Вероника Михеева
Ирина Харитонова

bhv

Microsoft® Access 2003



- Разработка объектов баз данных
- Архитектура сетевых приложений и проектов для SQL Server
- Обмен данными в формате XML

Наиболее полное руководство

+ CD-ROM

В ПОДЛИННИКЕ®

Вероника Михеева
Ирина Харитонова

Microsoft® Access 2003

Санкт-Петербург
«БХВ-Петербург»
2004

УДК 681.3.06
ББК 32.973.26-018.2
М69

Михеева В. Д., Харитонова И. А.

М69 Microsoft® Access 2003. — СПб.: БХВ-Петербург, 2004. — 1072 с.: ил.

ISBN 5-94157-366-9

В книге дается описание среды Access 2003, ее объектов и способов работы с данными, как своими, так и импортированными из других источников. Приводится краткое введение в теорию реляционных баз данных и систем управления базами данных (СУБД). Подробно описаны процедуры создания довольно сложных объектов баз данных. Приведены примеры типичных задач, которые постоянно приходится решать разработчикам приложений. Описаны основные средства программирования приложений Access: язык программирования VBA, объектные модели, разработка Web-приложений и способы интеграции приложений Access с другими приложениями Microsoft Office. Рассмотрены вопросы создания профессиональных многопользовательских сетевых баз данных и проектов, а также администрирования баз данных и проектов Access.

Для широкого круга пользователей

УДК 681.3.06
ББК 32.973.26-018.2

Группа подготовки издания:

Главный редактор	*Екатерина Кондукова*
Зам. главного редактора	*Евгений Рыбаков*
Зав. редакцией	*Григорий Добин*
Редактор	*Екатерина Капалыгина*
Компьютерная верстка	*Ольги Сергиенко*
Корректор	*Зинаида Дмитриева*
Дизайн обложки	*Игоря Цырульникова*
Зав. производством	*Николай Тверских*

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 25.08.04.
Формат 70×100 1/16. Печать офсетная. Усл. печ. л. 86,43.
Тираж 5000 экз. Заказ №
"БХВ-Петербург", 190005, Санкт-Петербург, Измайловский пр., 29.

Гигиеническое заключение на продукцию, товар № 77.99.02.953.Д.001537.03.02
от 13.03.2002 г. выдано Департаментом ГСЭН Минздрава России.

Отпечатано с готовых диапозитивов
в ОАО "Техническая книга"
190005, Санкт-Петербург, Измайловский пр., 29

ISBN 5-94157-366-9

© Михеева В. Д., Харитонова И. А., 2004
© Оформление, издательство "БХВ-Петербург", 2004

Содержание

Введение **19**

О чем эта книга 19
Для кого предназначена эта книга 19
Структура книги 20
Используемые обозначения 22
Компакт-диск с примерами 23
От издательства 24

ЧАСТЬ I. ОСНОВЫ РАЗРАБОТКИ БАЗ ДАННЫХ В MICROSOFT ACCESS 2003 ..... **25**

Глава 1. Общие сведения о Microsoft Access 2003 ..... **27**

Базы данных как средство хранения и обработки информации 28
Основные понятия 28
Microsoft Access как настольная СУБД реляционного типа 29
Различия между СУБД Access и электронными таблицами Excel 32
Запуск Access и открытие баз данных 33
Создание новой базы данных 38
Создание новой пустой базы данных 39
Создание копии существующей базы данных 40
Создание базы данных с использованием шаблона 40
Формат файла базы данных 43
Свойства файла базы данных 44
Рабочая среда Access 45
Окна рабочей среды Access 46
Меню и панели инструментов 48
Окно базы данных 51
Область задач 57
Использование справочной системы 57
Работа с областью задач *Справка Access* 57
Работа с Помощником 61
Другие средства получения справки 64

Глава 2. Работа с таблицами 65

Основные режимы работы с таблицами 66
Создание таблиц 71
Создание таблицы путем ввода данных 72
Создание таблиц с помощью Мастера 74
Создание таблицы с помощью Конструктора таблиц 78
Определение полей 79
Свойства полей таблицы 82
Использование Мастера подстановок 84
Форматы отображения данных 89
Определение маски ввода 94
Вставка, удаление и переименование полей таблицы 97
Определение ключевых полей 99
Создание и использование индексов 102
Связывание таблиц на схеме данных 104
Поддержка целостности данных 109
Ввод и проверка данных 110
Навигация по таблице 110
Работа с записями 113
Контроль и проверка введенных данных 114
Проверка орфографии и автозамена 118
Ввод данных с помощью копирования и перемещения 121
Использование списков значений 124
Отображение подчиненных таблиц 125
Изменение внешнего вида таблицы 129
Печать таблицы 134
Сортировка, поиск и фильтрация данных 139
Сортировка данных в таблице 139
Поиск и замена данных 141
Фильтрация данных 143

Глава 3. Использование внешних данных 149

Форматы данных, доступные для импорта, экспорта и связывания 150
Использование баз данных других форматов 152
Форматы баз данных 152
Присоединение и импорт внешних таблиц 154
Присоединение и импорт таблиц dBASE и Paradox 155
Присоединение и импорт таблиц баз данных Access 161
Импорт и присоединение электронных таблиц 162
Импорт и присоединение таблиц из текстовых файлов 170
Создание таблицы путем импорта из текстового файла 171
Особенности импорта текстовых файлов 173
Импорт и присоединение документов HTML 175
Создание спецификаций импорта 177
Импорт данных из документов и схем XML 180
Формат XML в Access. Общие сведения 180
Импорт документов XML 181

Импорт и присоединение списков из служб Windows SharePoint ... 183
Импорт и присоединение данных из адресных книг и папок Exchange ... 187
Импорт и присоединение таблиц из источника данных ODBC ... 191
Обновление связей с присоединенными таблицами ... 195
Экспорт данных в другие форматы ... 197
Экспорт в электронную таблицу или файлы dBASE, Paradox ... 197
Экспорт данных в текстовый файл ... 198
Быстрый экспорт данных в другие приложения Microsoft Office ... 202
Экспорт данных в форматы объектов WWW ... 205
Использование Буфера обмена Windows для импорта и экспорта данных ... 205

Глава 4. Отбор и сортировка записей с помощью запросов ... 211

Запросы и фильтры ... 212
Создание простого запроса с помощью Мастера ... 212
Создание и изменение запроса с помощью Конструктора ... 215
Создание вычисляемых полей в запросах ... 225
Виды соединений ... 227
Внутреннее соединение ... 228
Внешнее соединение ... 232
Рекурсивный запрос ... 234
Запросы с параметрами ... 235
Создание запроса из фильтра ... 237
Выполнение запроса ... 241
Изменение данных в результирующем множестве запроса ... 241
Печать результатов запроса ... 243
Передача результатов запроса в другие приложения ... 244
Использование выражений в запросах ... 244
Выражения в Access ... 244
Арифметические операторы ... 245
Операторы присваивания и сравнения ... 246
Логические операторы ... 246
Операторы слияния строковых значений (конкатенации) ... 248
Операторы идентификации "!" и "." ... 248
Операторы сравнения с образцом ... 248
Идентификаторы объектов ... 249
Константы ... 250
Функции ... 250
Создание выражений с помощью Построителя выражений ... 253

Глава 5. Создание форм для ввода данных ... 256

Автоматическое создание формы на основе таблицы или запроса ... 257
Создание формы с помощью Мастера ... 258
Режимы работы с формами ... 262
Печать формы ... 264
Разработка формы в режиме Конструктора ... 264
Окно приложения в режиме Конструктора ... 265
Структура формы ... 266
Панель инструментов *Конструктор форм* ... 268

Панель инструментов *Формат (форма/отчет)* ... 270
Параметры работы с формами ... 272
Основные элементы управления ... 273
Создание и удаление элементов управления ... 275
Создание элементов управления с помощью Панели элементов ... 275
Создание элементов управления с помощью кнопки *Список полей* ... 280
Удаление элементов формы ... 281
Выделение формы и элементов управления, отображение окна свойств ... 281
Задание последовательности перехода для элементов управления в форме ... 282
Перемещение и изменение размеров элементов управления ... 283
Перемещение и изменение размеров одного элемента управления ... 284
Выделение и перемещение нескольких элементов управления ... 285
Выравнивание одного элемента управления по сетке ... 286
Выравнивание и изменение размеров нескольких элементов управления ... 287
Изменение взаимного расположения и группировка элементов управления ... 288
Копирование, вставка и удаление элементов управления ... 289
Изменение надписей ... 290
Изменение типа элемента управления ... 290
Оформление формы и ее элементов ... 290
Применение автоформата ... 291
Изменение цвета фона формы и ее элементов ... 293
Изменение фонового рисунка формы ... 294
Форматирование элементов формы ... 296
Копирование формата элемента управления ... 297
Условное форматирование элементов формы ... 298
Использование дополнительных цветов ... 300
Сохранение формы ... 301
Просмотр и ввод данных в режиме Формы ... 302
Окно приложения при работе в режиме Формы ... 302
Добавление записи ... 305
Изменение записи ... 307
Отмена изменения записи ... 309
Удаление записи ... 309

Глава 6. Создание и печать отчетов ... 311

Автоматическое создание отчета на основе таблицы или запроса ... 312
Создание отчета с помощью Мастера ... 313
Преобразование отчетов ... 321
Преобразование отчета в формат документов Word и Excel ... 322
Преобразование отчета в другие форматы ... 322
Снимки отчетов ... 323
Режимы работы с отчетами ... 326
Режим Предварительного просмотра ... 327
Режим Просмотра образца ... 329
Режим Конструктора ... 329
Структура отчета ... 330
Изменение размеров отчета и его разделов ... 331

Инструментальные средства режима Конструктора ... 331
Панель инструментов *Конструктор отчетов* ... 332
Форматирование отчета ... 334
Панель элементов ... 334
Выделение отчета и его элементов ... 335
Изменение свойств отчета и его элементов ... 335
Создание и удаление элементов управления отчета ... 339
Создание элементов управления с помощью Панели элементов ... 339
Создание элементов управления с помощью кнопки *Список полей* ... 342
Удаление элементов отчета ... 342
Изменение размеров и перемещение элементов управления ... 342
Перемещение и изменение размера отдельного элемента управления ... 343
Выравнивание элемента управления по сетке ... 344
Выделение, перемещение и изменение свойств нескольких элементов управления ... 344
Выравнивание нескольких элементов управления ... 345
Изменение межстрочного интервала ... 345
Настройка расстояний между несколькими элементами ... 346
Копирование элементов управления через буфер обмена ... 347
Копирование формата элемента управления ... 347
Оформление страниц отчета ... 347
Добавление разрывов и нумерации страниц ... 348
Использование в отчете линий и прямоугольников ... 348
Создание в отчете вычисляемых полей ... 350
Сортировка и группировка данных ... 352
Применение шаблона отчетов ... 353
Просмотр и печать отчетов ... 354

ЧАСТЬ II. ПРОФЕССИОНАЛЬНАЯ РАБОТА С MICROSOFT ACCESS 2003 ... 357

Глава 7. Новые возможности Microsoft Access 2003 ... 359

Новые возможности комплекта приложений Microsoft Office 2003 ... 359
Новые требования к системе ... 361
Новый дизайн пользовательского интерфейса ... 361
Новая справочная система ... 366
Новый формат файлов баз данных ... 367
Новые средства защиты от макровирусов ... 368
Поддержка технологии Smart Tag ... 369
Поддержка XML и схем данных ... 370
Сотрудничество и обмен информацией ... 371
Новые функциональные возможности по разработке и сопровождению баз данных ... 372
Резервное копирование баз данных ... 372
Средства автозамены в виде Smart Tag ... 372
Отслеживание зависимостей объектов ... 373
Настройка шрифта в окне редактирования текста запроса ... 374
Автоматическое исправление ошибок ... 374

Преобразование связанных таблиц в локальные ... 375
Автоматическое распространение измененных свойств поля ... 375

Глава 8. Редактирование и анализ данных с помощью запросов ... 377

Создание таблиц с помощью запроса ... 378
Преобразование запроса на выборку в запрос на изменение ... 378
Выполнение запроса на изменение ... 379
Создание запросов на добавление данных ... 381
Создание запросов на обновление таблиц ... 384
Создание запросов на удаление записей ... 387
Анализ данных с помощью запросов ... 389
Перекрестные запросы ... 389
Использование фиксированных заголовков столбцов в перекрестных запросах ... 392
Изменение уровня детализации в запросе ... 394
Использование статистических функций для выполнения расчетов ... 395
Вычисления на всем диапазоне записей ... 397
Вычисления на выделенных записях таблицы ... 398
Отбор повторяющихся записей и записей, не имеющих подчиненных ... 401
Сводные таблицы ... 403
Создание сводной таблицы ... 404
Фильтрация данных сводной таблицы ... 408
Изменение сводной таблицы ... 411
Сортировка в сводной таблице ... 412
Изменение уровня детализации ... 413
Общие и промежуточные итоги ... 416
Форматирование сводной таблицы ... 419
Сводные диаграммы ... 420
Создание сводной диаграммы ... 420
Изменение вида диаграммы ... 423
Форматирование элементов сводной диаграммы ... 425
Настройка свойств полей в запросе и свойств самого запроса ... 426
Создание запросов SQL ... 430
Сравнение ANSI и Jet SQL ... 430
Зарезервированные слова Jet SQL ... 432
Функции и операторы Access, используемые вместо ключевых слов ANSI SQL ... 433
Зарезервированные слова Jet SQL и функции, не входящие в ANSI SQL ... 434
Эквивалентные типы данных Jet SQL и ANSI SQL ... 435
Подстановочные знаки при сравнении строк ... 437
Зарезервированные слова *DISTINCTROW* и *DISTINCT* ... 437
Использование разделителей и символов SQL ... 439
Использование инструкций SQL для создания запросов ... 440
Создание запросов на объединение записей (*Union*) ... 440
Создание запроса к серверу ... 442
Управляющие запросы ... 442
Создание подчиненных запросов ... 443

Глава 9. Создание составных форм ... 446

Использование Панели элементов ... 446
Встроенные элементы управления ... 447
Разновидности элементов управления по типу содержимого ... 448
Элемент управления *Надпись* ... 449
Элемент управления *Поле* ... 450
Элементы управления выбора: *Выключатель*, *Переключатель*, *Флажок* и *Группа* ... 451
Элемент управления *Список* ... 457
Элемент управления *Поле со списком* ... 461
Элемент управления *Кнопка* ... 463
Элементы управления *Свободная рамка объекта* и *Присоединенная рамка объекта* ... 467
Элемент управления *Рисунок* ... 467
Элемент управления *Разрыв страницы* ... 467
Элемент управления *Набор вкладок* ... 468
Элемент управления *Подчиненная форма/отчет* ... 469
Элементы управления *Линия* и *Прямоугольник* ... 469
Преобразование одного элемента управления в другой ... 470
Создание многотабличных форм ... 471
Создание главной и подчиненной форм с помощью Мастера автоформ ... 472
Использование Мастера форм для создания главной и подчиненной форм ... 474
Добавление подчиненной формы в главную форму ... 475
Связывание главной и подчиненной форм ... 478
Синхронизация данных в элементах формы ... 479
Переход на определенную запись в форме ... 480
Динамическая фильтрация данных ... 480
Автоматическая подстановка значения в поле ввода ... 482
Создание многостраничных форм ... 484
Создание многостраничных форм с элементом *Разрыв страницы* ... 484
Создание многостраничной формы с вкладками ... 487
Добавление и удаление страниц в наборе вкладок ... 489
Изменение порядка следования страниц ... 489
Свойства элемента управления *Набор вкладок* ... 490
Всплывающие формы и окна диалога ... 491
Вычисления в формах ... 493
Создание вычисляемых полей ... 493
Создание вычисляемых полей для отображения итоговых значений ... 495
Использование статистических функций в формах ... 497
Функция *DLookup* ... 498
Функции *DCount* и *DSum* ... 499
Функция *DMax* ... 499
Рекомендации по оформлению форм ... 499
Размещение элементов управления в форме ... 500
Размещение элементов управления в области данных ... 500
Внешний вид элементов управления ... 500
Формулировка надписей и инструкций ... 501
Навигация в форме ... 501

Использование цвета ... 502
Оформление в едином стиле ... 503
Использование в формах новых возможностей Access 2003 ... 503

Глава 10. Подготовка составных отчетов ... 505

Создание составных отчетов ... 506
Применение подчиненных отчетов ... 506
Создание подчиненных отчетов ... 510
Перекрестные отчеты ... 517
Печать отчета в несколько колонок ... 520
Сортировка и группировка данных ... 521
Создание базового запроса ... 521
Определение условий сортировки и группировки ... 522
Определение содержимого отчета ... 525
Настройка печати групп отчета ... 528
Расположение групп в отчетах, имеющих несколько колонок ... 528
Повторение заголовка группы на каждой странице или колонке ... 528
Скрытие повторяющихся данных в отчете ... 529
Проведение вычислений ... 531
Создание полей, содержащих сумму с накоплением ... 531
Вычисление процентов ... 532
Нумерация записей и подсчет количества записей в отчете ... 532
Объединение текстовых данных и печать условий отбора записей в отчете ... 533
Фильтрация данных в отчете ... 535

Глава 11. Публикация данных в Интернете и корпоративной сети ... 537

Интеграция Access со всемирной сетью Интернет ... 537
Основные сведения об Интернете ... 538
Программа просмотра Интернета ... 540
О примерах этой главы ... 540
Использование гиперссылок в объектах базы данных ... 541
Панель инструментов *Веб-узел* ... 545
Преобразование объектов базы данных в Web-страницы ... 546
Преобразование в формат HTML ... 548
Преобразование в статический формат HTML ... 549
Преобразование в динамический формат HTML ... 552
Преобразование в формат XML ... 554
Преобразование в страницу доступа к данным ... 558
Страницы доступа к данным ... 559
Создание страниц доступа к данным ... 561
Автоматическое создание на основе таблицы или запроса ... 561
Создание с помощью Мастера ... 563
Подключение существующей Web-страницы к базе данных ... 568
Подключение к источнику данных ... 570
Разработка страницы доступа к данным в режиме Конструктора ... 572
Структура страницы доступа к данным ... 572
Элементы управления ... 573
Добавление, удаление и перемещение элементов управления ... 574

Сортировка и группировка данных ... 579
Изменение свойств страницы доступа к данным и ее элементов ... 583
Добавление на страницу графических элементов и гиперссылок ... 586
Оформление страницы доступа к данным ... 590
Добавление на страницу элементов ActiveX ... 591
Использование Web-компонентов Microsoft Office ... 593
Программирование страниц доступа к данным ... 594
Создание XML-форм в InfoPath ... 595
Средства публикации данных ... 603
Публикация данных с помощью Web-страниц ... 604
Публикация данных на сервере Web ... 605
Отправка данных по электронной почте ... 607
Публикация данных на сервере FTP ... 608

Глава 12. Программирование с помощью макросов ... 610

Использование макросов в базе данных Access ... 610
Создание макросов ... 614
Конструктор макросов ... 614
Создание групп макросов ... 618
Применение условий в макросах ... 620
Выполнение и отладка макросов ... 622
Запуск макроса из окна Конструктора макроса ... 623
Запуск макроса из окна базы данных ... 623
Запуск макроса с помощью кнопки на панели инструментов ... 623
Запуск макроса с помощью команды меню ... 625
Запуск макроса с помощью комбинации клавиш ... 626
Запуск макроса при открытии базы данных ... 627
Запуск макроса из другого макроса ... 627
Назначение макроса событию ... 628
Вызов макроса из процедуры VBA ... 633
Отладка макросов и поиск ошибок ... 633
Применение макросов ... 635
Работа с данными в формах и отчетах ... 635
Ссылки на формы, отчеты и их свойства ... 635
Ссылки на элементы управления форм, отчетов и их свойства ... 636
Ссылки на подчиненные формы и отчеты ... 636
Фильтрация записей в формах, отчетах, таблицах ... 637
Перемещение по данным ... 638
Обновление данных в формах и элементах управления ... 640
Работа с объектами ... 642
Преобразование макросов в процедуры VBA ... 643
Уровни безопасности макросов ... 646

Глава 13. Программирование на языке VBA ... 648

Язык Visual Basic для приложений (VBA) ... 649
Процедуры и функции, их вызов и передача параметров ... 649
Переменные, константы и типы данных ... 652
Ссылки на объекты ... 657

Массивы ... 658
Область действия переменных и процедур ... 660
Управляющие конструкции ... 663
Ветвления ... 663
Циклы ... 665
Выход из циклов и процедур ... 668
Объектные модели Microsoft Access 2003 ... 669
Основные понятия ... 669
Объектные модели Microsoft Office 2003 ... 670
Объектная модель Microsoft Office Access 2003 ... 671
Объектные модели доступа к данным ... 679
Объектная модель Microsoft DAO 3.6 ... 679
Модель объектов ActiveX для управления данными ... 682
Объектная модель Visual Basic для приложений ... 686
Объектная модель Microsoft Office 2003 ... 687
Использование объектов и семейств ... 691
Ссылки на объекты ... 691
Работа с объектами в семействах ... 693
Свойства и методы объектов ... 695
Модули как объекты Access ... 696
Стандартные модули ... 696
Модули класса ... 697
Создание модулей ... 700
Ссылки на объект *Module* ... 701
Среда программирования VBA ... 702
Окно редактора кода ... 702
Использование окна просмотра объектов ... 713
Установка ссылок на объектные библиотеки ... 715
Инструментальные средства отладки ... 716
Меню *Debug* ... 716
Окно *Locals* ... 718
Окно *Watches* ... 720
Окно *Quick Watch* ... 721
Окно *Immediate* ... 722
Окно *Call Stack* ... 723
Использование комбинаций клавиш быстрого вызова ... 723
Программирование в формах и отчетах ... 724
События Access ... 725
События данных ... 725
События фокуса ... 729
События клавиатуры ... 730
События мыши ... 731
События печати ... 733
События фильтра ... 734
События окна ... 735
События ошибок ... 737
События таймера ... 737
События режима Сводная таблица ... 737

Последовательности событий Access ... 739
Последовательность событий фокуса для элементов управления в формах ... 740
Последовательность событий изменения данных в элементе управления ... 740
Последовательность событий в формах ... 742
Последовательность событий клавиатуры и мыши ... 743
Последовательность событий в отчетах ... 743
Создание процедур обработки событий ... 744
Синхронизация данных в связанных формах ... 745
Проверка дублирования значений первичного ключа ... 746
Создание перекрестного отчета с изменяющимся числом столбцов ... 747
Использование функций для обработки событий ... 751
Импорт файла XML ... 753
Экспорт файла XML ... 754
Преобразование файла XML ... 756

Глава 14. Настройка пользовательского интерфейса ... 758

Настройка параметров приложения Access ... 758
Настройка параметров запуска ... 772
Запуск приложения с параметрами командной строки ... 774
Командный интерфейс приложения ... 776
Использование кнопочной формы ... 777
Создание и изменение панелей команд ... 781
Создание меню ... 782
Создание и изменение панелей инструментов ... 787
Создание и удаление контекстных меню ... 790
Изменение порядка команд ... 791
Связывание строк меню, панелей инструментов и контекстных меню с формами и отчетами ... 791
Изменение меню и панелей инструментов с помощью VBA ... 793

Глава 15. Интеграция Access 2003 с компонентами Office 2003 ... 796

Связывание и внедрение объектов ... 797
Вставка рисунков в формы и отчеты ... 799
Вставка рабочего листа Excel в форму или отчет ... 807
Вставка документов Word в таблицы, формы и отчеты Access ... 810
Автоматизация ... 817
Клиенты и серверы автоматизации ... 817
Взаимодействие компонентов Microsoft Office 2003 ... 818
Использование Microsoft Access в качестве клиента автоматизации ... 818
Операции с объектами Microsoft Excel ... 820
Обмен данными с Microsoft Word в системе автоматизации ... 829
Использование Access в качестве сервера автоматизации ... 840
Использование элементов ActiveX в приложении ... 841
Добавление элемента управления *Календарь* в форму ... 842
Подключение специальных элементов управления в реестр Windows ... 847
Программный доступ к документам XML ... 849

ЧАСТЬ III. РАЗРАБОТКА И СОПРОВОЖДЕНИЕ ПРИЛОЖЕНИЙ ACCESS 2003 855

Глава 16. Разработка сетевых приложений Access 857

Способы совместного использования баз данных 857
Совместное использование базы данных 858
Совместное использование только данных 858
Совместное использование объектов базы данных через Интернет 858
Совместная работа с базой данных с помощью репликации 859
Совместный доступ к базе данных с помощью SQL Server 859
Режимы работы с базой данных в сети 859
Пример сетевого приложения 861
Правила игры 861
Сетевое решение в реализации архитектуры приложения 862
Организация совместного доступа к данным и объектам 864
Совместный просмотр и изменение данных 864
Методы управления блокировками записей 865
Предотвращение конфликтов при блокировке записей в сети 867
Отображение обновленных данных 867
Совместное изменение объектов 868
Совместное использование страниц доступа к данным 869
Организация обмена данными между компонентами сетевого приложения 870
Программирование с использованием интерфейса DAO 870
Создание и использование рабочей области 871
Открытие источника данных 873
Программное изменение структуры и схемы данных 877
Выполнение операций с данными 879
Программная синхронизация доступа к данным 880
Выполнение транзакций 883
Защита сетевого приложения Access 884
Защита базы данных на уровне пользователей 885
Рабочая группа пользователей базы данных 885
Встроенные учетные записи Access 886
Административные права доступа 887
Право на владение объектами базы данных 887
Установка системы защиты на уровне пользователей 888
Использование Мастера защиты 889
Создание и подключение файла рабочей группы 891
Снятие защиты на уровне пользователей 891
Создание учетных записей 892
Назначение прав доступа 892
Права доступа в приложении "Игра в доминирование" 893
Ограничение разрешений для встроенных учетных записей Access 894
Защита программного кода на VBA 894

Глава 17. Проекты Access 2003 897

Основные понятия 898
Отличие архитектуры "клиент-сервер" от архитектуры "файл-сервер" 898

Распределение функций в архитектуре "клиент-сервер" ... 899
Универсальный доступ к данным через OLE DB ... 900
Общие сведения ... 901
Компоненты OLE DB ... 902
Microsoft ActiveX Data Objects (ADO) ... 903
Установка связи с источником данных посредством интерфейса OLE DB ... 903
Настольная версия Microsoft SQL Server 2000 ... 907
Создание проекта Access 2003 ... 908
Создание проекта с использованием существующей базы данных ... 909
Создание проекта с новой базой данных ... 910
Открытие файла проекта ... 912
Работа с таблицами ... 912
Определение и изменение структуры таблицы ... 912
Диалоговое окно свойств таблицы ... 917
Создание триггеров ... 921
Работа со схемами базы данных ... 922
Конструктор баз данных ... 923
Создание таблицы в окне Конструктора базы данных ... 924
Изменение свойств существующей таблицы ... 925
Установление связей между таблицами ... 927
Создание индексов и ограничений ... 928
Сохранение и откат изменений в структуре базы данных ... 929
Создание новой схемы базы данных и добавление в нее таблиц ... 930
Изменение режима таблиц и расположения их на схеме базы данных ... 930
Предварительный просмотр и печать схемы данных ... 935
Копирование и перемещение таблиц между схемами данных ... 936
Удаление таблиц из схемы и из базы данных ... 936
Работа со схемами базы данных в многопользовательской среде ... 936
Работа с запросами ... 937
Конструктор запросов ... 937
Способы объединения таблиц в представлении ... 939
Группировка записей и суммирование значений в представлении ... 940
Использование подчиненных представлений ... 941
Использование параметров в инструкциях SQL ... 941
Диалоговое окно свойств представления ... 943
Создание и применение хранимых процедур ... 945
Создание и использование определяемой пользователем функции ... 949
Использование индексированных представлений ... 950
Сортировка и фильтрация данных в формах и отчетах ... 952
Повышение производительности приложения "клиент-сервер" ... 953
Минимизация количества обращений к серверу для доступа к данным ... 953
Осуществление обработки данных на сервере ... 953
Ограничение объема загружаемых с сервера данных ... 953
Повышение производительности при работе с отчетами ... 954
Повышение производительности при работе с формами ... 954
Повышение производительности внешней базы данных на SQL Server ... 955

Глава 18. Репликация баз данных ... 957

Портфельная репликация ... 958
Репликация баз данных Access ... 966

Создание реплик базы данных ... 967
Создание частичной реплики ... 969
Синхронизация реплик ... 973
Управление репликами ... 976
Репликация проектов Access ... 977
Репликация файлов на сервере Web ... 978
Репликация файлов в Internet Explorer и Windows ... 978
Репликация страниц доступа к данным ... 978

Глава 19. Миграция приложений ... 980

Преобразование настольного приложения Access в приложение с архитектурой "клиент-сервер" ... 981
Целесообразность перехода к архитектуре "клиент-сервер" ... 981
Надежность ... 981
Производительность ... 982
Масштабируемость ... 983
Безопасность ... 983
Способы преобразования приложений Access для использования их с SQL Server ... 983
Разделение баз данных ... 984
Перенос сетевого приложения на платформу SQL Server ... 987
Подготовительный этап ... 987
Выполнение переноса ... 989
Подготовка клиент-серверного приложения к эксплуатации ... 995
Использование приложений предыдущих версий Microsoft Access ... 995
Поддержка 2000 года в Microsoft Access ... 995
Совместимость форматов баз данных разных версий Microsoft Access ... 996
Совместное использование базы данных в нескольких версиях Access ... 997
Использование в Access баз данных предыдущих версий без преобразования формата ... 999
Использование файлов объектов приложения предыдущих версий Access ... 1000
Использование файлов объектов данных предыдущих версий Access ... 1001
Варианты преобразования баз данных в Access 2003 ... 1002
Сравнение преобразования и импорта файлов предыдущих версий Access ... 1003
Разрешение некорректных ссылок на библиотечные базы данных и надстройки Access ... 1004
Ограничения, связанные с преобразованием баз данных Access 2000 и 2002—2003 в формат Access 97 ... 1004

Глава 20. Администрирование баз данных ... 1006

Задачи администрирования баз данных ... 1006
Архивирование, сжатие и восстановление баз данных ... 1007
Создание резервной копии базы данных или проекта Access ... 1007
Сжатие базы данных или проекта Access ... 1008
Восстановление поврежденной базы данных ... 1010
Защита информации с помощью шифрования ... 1010

Администрирование защищенных баз данных и проектов Access ... 1011
Защита баз данных на SQL Server ... 1012
Защита объектов баз данных на SQL Server ... 1012
Управление учетными записями на SQL Server ... 1013
Защита базы данных Access с помощью пароля ... 1014
Установка и снятие пароля защиты базы данных ... 1014
Установка связи с таблицами базы данных, защищенной паролем ... 1015
Администрирование баз данных Access, защищенных на уровне пользователей ... 1016
Использование файла рабочей группы ... 1016
Администратор рабочих групп ... 1017
Подключение файла рабочей группы к приложению Access ... 1017
Создание и восстановление файла рабочей группы ... 1018
Управление учетными записями ... 1020
Назначение прав доступа к объектам базы данных ... 1023
Изменение прав доступа к объектам базы данных ... 1024
Предоставление права на владение объектами базы данных ... 1026

Приложение 1. Глоссарий ... 1031

Приложение 2. Сетевое приложение "Игра в доминирование" ... 1042

Правила игры в доминирование ... 1042
Архитектура приложения ... 1043
Функциональное описание приложения ... 1044
Краткое описание работы сервера ... 1044
Краткое описание работы клиента ... 1044
Завершение игры ... 1044
Сервер "Игры в доминирование" ... 1045
Клиент "Игры в доминирование" ... 1046
Интерфейс пользователя ... 1048
Установка сетевого приложения "Игра в доминирование" ... 1049
Параметры защиты приложения "Игра в доминирование" ... 1050

Приложение 3. Описание компакт-диска с примерами ... 1054

Предметный указатель ... 1059

Введение

Предлагаемая читателю книга "Microsoft Access 2003" входит в состав комплекта книг, посвященных семейству Microsoft Office 2003. Авторы уже в течение нескольких лет используют эту популярную СУБД в своей профессиональной деятельности. Это позволяет надеяться, что их опыт создания самых разных как с точки зрения проблемных областей, так и с точки зрения сложности разработки приложений, будет полезен читателям.

О чем эта книга

Предметом обсуждения в данной книге является настольная (desktop) СУБД Access 2003 — один из компонентов широко распространенного семейства офисных приложений Microsoft Office 2003. На сегодняшний день Access является одной из самых популярных настольных СУБД. Это связано с тем, что Access обладает очень широким диапазоном средств для ввода, анализа и представления данных. Эти средства являются не только простыми и удобными, но и высокопродуктивными, что обеспечивает высокую скорость разработки приложений. Изначально Access обладала рядом уникальных возможностей, такими как умение сводить воедино информацию из самых разных источников (электронных таблиц, текстовых файлов, других баз данных); представление данных в удобном для пользователя виде с помощью таблиц, диаграмм, отчетов; интеграция с другими компонентами Microsoft Office. Совершенствуясь от версии к версии, Access стала инструментом, который может удовлетворить самые разные категории пользователей: от новичка, которому нравится дружественный интерфейс системы, позволяющий ему справиться с малыми задачами, до профессионального разработчика, который имеет весь необходимый инструментарий построения уникального решения для конкретного предприятия среднего бизнеса.

Для кого предназначена эта книга

Поскольку СУБД Access предназначена для *разных* категорий пользователей, авторы старались написать книгу так, чтобы она была полезна и новичкам, и опытным пользователям.

Те пользователи, которые успешно применяют в своей деятельности другие компоненты Microsoft Office, смогут понять, что даст им освоение нового продукта, и достаточно быстро построить решение для новых задач, интегрируя его с уже существующими решениями.

Опытные пользователи Access могут рассчитывать на эту книгу как на самое полное руководство по продукту (такой статус имеют все книги серии "В подлиннике" издательства "БХВ-Петербург"). Им, наверное, будут наиболее интересны разделы книги, посвященные разработке сложных многопользовательских приложений и приложений с архитектурой "клиент-сервер".

Программисты-разработчики получат полное представление о библиотеках объектов, которые могут быть использованы в программах Visual Basic для приложений (VBA), о средствах программирования и отладки программ, о событиях, которые могут использоваться для того, чтобы обеспечить требуемую реакцию приложения на действия пользователя.

Администраторы баз данных найдут в ней информацию о том, какие средства Access позволят наиболее эффективным образом поддерживать работающие приложения.

Кроме того, в книге довольно много советов, в которых авторы делятся с читателями своим собственным опытом разработки приложений на базе Access. Авторы предполагают, что читатель этой книги имеет опыт работы в среде Microsoft Windows, умеет пользоваться мышью и знает основы интерфейса Windows, поэтому здесь не описывается выполнение операции с окнами и мышью (типа перемещения окна или перетаскивания объекта с помощью мыши).

Структура книги

Книга структурирована таким образом, чтобы провести читателя от самых основных (и при этом очень простых) операций через все функции и вспомогательные средства Access до разработки и администрирования сетевых многопользовательских приложений и проектов Access с архитектурой "клиент-сервер". Она состоит из трех частей, части состоят из глав, имеющих сквозную нумерацию, чтобы обеспечить оптимальное количество уровней разделов и подразделов книги. При написании глав авторы старались не повторяться в изложении материала, при этом везде, где это необходимо, приведены ссылки на материал, изложенный в других главах или разделах. Для удобства поиска необходимого материала ссылки, как правило, делаются не на главу, а на нужный раздел в главе, например:

(См. разд. "Объектные модели Microsoft Office Access 2003" гл. 13.)

Часть I "Основы разработки баз данных в Microsoft Access 2003" ориентирована на пользователей, которые никогда не работали с базой данных Access или даже вообще не работали с базами данных. Шесть глав этой части (1—6) должны дать такому читателю представление о продукте, базах данных, основных операциях с данными, основных объектах Access — таблицах, запросах, формах и отчетах — и операциях с ними. Примеры в этой части подобраны таким образом, чтобы объяснить стандартные приемы решения типовых задач, которые могут возникнуть у пользователя Access. При этом в книге мы постоянно делаем ссылки на последующие главы, ко-

торые, как правило, описывают и объясняют более сложные решения, поэтому любознательный читатель может знакомиться с книгой не обязательно последовательно. Тот же, кто считает себя опытным пользователем Access, может пропустить всю первую часть и начать читать книгу со второй части.

Часть II "Профессиональная работа с Microsoft Access 2003" ориентирована на пользователя, который уже имеет опыт работы с предыдущими версиями Microsoft Access, знаком с основами проектирования баз данных и которому требуется создать нетривиальное приложение в среде Access. Эта часть начинается с обзора новых возможностей Microsoft Office Access 2003 *(гл. 7)*, который позволит читателю быстро сориентироваться в том, какие главы представляют для него наибольший интерес. Дальше он может изучать эти главы в любом порядке, т. к. они являются достаточно независимыми и авторы редко использовали примеры из предыдущих глав для иллюстрации материала и подготовки новых примеров (в тех местах, где это все-таки было, есть ссылки на предыдущие главы). Начинающим разработчикам баз данных будет полезен материал всей второй части, где очень подробно описаны процедуры создания довольно сложных объектов баз данных. Эти примеры помогут справиться с типичными задачами, которые постоянно приходится решать разработчикам такого рода приложений. Изложение материала во второй части также построено по принципу нарастания сложности. В первых главах *(8—10)* выполнен возврат к тем объектам Access, которые уже описывались в первой части: запросам, формам, отчетам. Однако в этих главах объясняется работа с более сложными видами объектов: запросы, выполняющие множественные изменения данных (их нужно делать особенно аккуратно), перекрестные запросы, сводные таблицы и сводные диаграммы, составные формы и отчеты. Материал этих глав, так же как и примеры, не требует знания языков программирования — все задачи решаются с помощью очень развитых диалоговых средств интерфейса Access — Конструкторов, Мастеров, Построителей.

В *гл. 11* рассматриваются все вопросы применения технологий Интернет/интранет в СУБД Access. Это в первую очередь создание и использование страниц доступа к данным, а также возможности экспорта форм и отчетов Access в Web-страницы, создание динамических Web-страниц и т. д.

Две главы *(12 и 13)* посвящены вопросам программирования в Access. В них рассматриваются два языка программирования Access — язык макрокоманд и Visual Basic для приложений (VBA), а также соответствующие им объекты — макросы и модули. Особое внимание уделяется описанию событий в формах и отчетах.

После того как все объекты Access подробно описаны, читатель узнает, как настроить свое приложение, чтобы оно имело законченный вид *(гл. 14)*: создать систему меню, кнопочную форму, кнопки на панели команд.

И наконец, последняя глава второй части рассказывает о механизмах интеграции приложения Access с другими компонентами Microsoft Office 2003. При этом рассматриваются как диалоговые средства внедрения и связывания объектов из других приложений в таблицы, формы и отчеты Access, так и программные методы управления объектами других приложений из приложения Access.

Часть III "Разработка и сопровождение приложений Access 2003" ориентирована на профессиональных разработчиков-программистов и администраторов баз данных.

Основной материал этой части *(гл. 16 и 17)* описывает особенности разработки сложных многопользовательских приложений как в настольном варианте, т. е. с применением только СУБД Access, так и использование Access для разработки клиентской части приложения "клиент-сервер". К этим особенностям относятся: принципы проектирования сетевого многопользовательского приложения, способы защиты данных, управление блокировками записей и т. д. В *гл. 17* описывается механизм доступа к данным на сервере базы данных — OLE DB, объясняется работа с объектами в проектах Access.

Три главы этой части *(18, 19, 20)* посвящены вопросам сопровождения и администрирования приложений Access: здесь описываются средства Access, обеспечивающие репликацию баз данных, архивирование, сжатие и восстановление баз данных, защиту данных, администрирование рабочих групп. Кроме того, подробно рассматриваются вопросы использования баз данных, созданных в предыдущих версиях Access, преобразования баз данных, созданных в предыдущих версиях Access, в формат Access 2002—2003, возможности перемещения таблиц базы данных Access на Microsoft SQL Server и преобразования базы данных Access 2003 в проект Access 2003.

Книга также содержит три приложения и предметный указатель.

Приложение 1 представляет собой глоссарий.

Приложение 2 приводит сетевое многопользовательское приложение "Игра в доминирование", создание которого рассматривается в *гл. 16*.

Приложение 3 описывает содержимое компакт-диска с примерами, относящимися к данной книге.

Используемые обозначения

Некоторые слова и фразы в тексте выделяются специальным шрифтом. Поскольку книга написана по локализованной версии Access 2003, экранные термины являются русскоязычными, приводятся в тексте в точности так, как они видны на экране, и выделяются **полужирным шрифтом**. К экранным терминам относятся названия команд, диалоговых окон, элементов диалогов (полей ввода, флажков, кнопок, списков, переключателей и т. д.), панелей инструментов и кнопок, значения раскрывающихся списков. Для удобства пользователей англоязычной версии рядом в скобках приводится английский вариант термина (по крайней мере, для первого вхождения термина в раздел). Однако существуют исключения из этого правила. В некоторых главах используются программные продукты и средства, которые не локализуются: Microsoft SQL Server, редактор VBE. В этом случае основным является англоязычный термин, который и выделяется в тексте **полужирным шрифтом**. Рядом дается наш перевод этого термина или объяснение его назначения.

Курсивом в тексте выделяются определяющие вхождения новых понятий. Иногда авторы применяют *курсив* также для выделения отдельных слов или выражений, на которые хотят обратить внимание читателя.

Имена программ, средств, файлов и папок пишутся с прописной буквы без кавычек и без выделения, например: Проводник (Explorer), Панель задач, Мастер отчета. Собственные имена объектов Access — таблиц, запросов, форм, отчетов, полей в

таблицах, а также названия функций, макрокоманд, инструкций языка SQL (языка структурированных запросов), тексты программ и текст, вводимый пользователем с клавиатуры, выделяются моноширинным шрифтом.

Кроме шрифтовых выделений, используется два типа специальных абзацев: советы и замечания.

Совет

Прежде чем выполнять эту операцию, мы рекомендуем вам сделать резервную копию файла Борей.mdb.

Замечание

При описании пошаговых процедур мы чаще всего представляли самый быстрый способ выполнения операции, в большинстве случаев этого же результата можно достичь, используя соответствующую команду меню.

Книга снабжена большим количеством рисунков. Читатель скоро обнаружит, что они отличаются по стилю. Дело в том, что стиль рисунков сильно зависит от установленной на компьютере операционной системы. Авторы работали по локализованной версии Microsoft Office Access 2003, но в двух операционных средах: Microsoft Windows 2000 Professional и Microsoft Windows XP. Отсюда и различие в стиле рисунков.

Компакт-диск с примерами

В качестве примеров использовалась учебная база данных Борей (Northwind для англоязычной версии), которая входит в комплект поставки Microsoft Office Access 2003. В ряде случаев авторы просто объясняли, как построены те или иные объекты этой базы данных. В других случаях на основе таблиц базы данных Борей авторы создавали новые объекты: запросы, формы, отчеты, страницы доступа к данным. Примеры, так же как и излагаемый материал, усложняются от первых глав книги к последним. Используемые примеры разбираются в тексте книги с различной степенью подробности. В первых главах этот разбор очень детальный, и читатель может легко воспроизвести все примеры, выполняя приводимые в тексте пошаговые процедуры. В последних главах, во избежание повторения уже описанных процедур и ориентируясь на более подготовленного читателя, приводится достаточно беглый обзор представленных примеров. Кроме того, в последней части книги авторы ссылаются на сложное сетевое многопользовательское приложение "Игра в доминирование", подробное описание которого вынесено в *Приложение 2*. С большей частью его кода (кроме приведенных в тексте книги фрагментов) читатель может познакомиться, скопировав его с компакт-диска, который распространяется вместе с книгой. На этом же компакт-диске приведены некоторые примеры из других глав книги. При описании разработки приложений в архитектуре "клиент-сервер" авторы снова воспользовались учебной базой данных (файл Northwind.adp), которая входит в комплект демонстрационных приложений Access 2003.

От издательства

Чтобы получить дополнительные сведения или выразить свое отношение к данной книге, читатели могут обратиться в издательство "БХВ-Петербург" по адресу: **http://www.bhv.ru**. Кроме того, при необходимости можно связаться с авторами книги по электронной почте:

vm_bhv@mail.ru — Вероника Дмитриевна Михеева

kia@spbmegapolis.ru — Ирина Александровна Харитонова



ЧАСТЬ I

Основы разработки баз данных в Microsoft Access 2003

Глава 1. Общие сведения о Microsoft Access 2003

Глава 2. Работа с таблицами

Глава 3. Использование внешних данных

Глава 4. Отбор и сортировка записей с помощью запросов

Глава 5. Создание форм для ввода данных

Глава 6. Создание и печать отчетов



ГЛАВА 1

Общие сведения о Microsoft Access 2003

Система управления базами данных Microsoft Access является одним из самых популярных приложений в семействе настольных СУБД. Все версии Access имеют в своем арсенале средства, значительно упрощающие ввод и обработку данных, поиск данных и предоставление информации в виде таблиц, графиков и отчетов. Начиная с версии Access 2000, появились также Web-страницы доступа к данным, которые пользователь может просматривать с помощью программы Internet Explorer. Помимо этого, Access позволяет использовать электронные таблицы Excel, таблицы из разных баз данных, как настольных, так и серверных, и другие источники для хранения информации, необходимой приложению. Присоединив внешние таблицы, пользователь Access будет работать с данными в этих таблицах так, как если бы это были таблицы Access, причем другие пользователи могут продолжать работать с этими данными в той среде, в которой они были созданы.

Настоящая глава адресована тем пользователям, которые только приступают к использованию Access в своей работе. Поэтому мы даем в ней основные понятия, касающиеся баз данных и систем управления базами данных (СУБД), которые потребуются нам в дальнейшем изложении. Еще мы хотели бы показать разницу между обработкой данных в формате электронных таблиц, которые стали уже привычными для широкого круга людей, и возможностями систем управления базами данных. Мы объясним также, когда следует от работы с простыми электронными таблицами перейти к более сложному приложению Access. В дальнейшем вы убедитесь, что на самом деле все не так сложно, если уметь пользоваться теми инструментами, которые предоставляет Access в помощь пользователю. Мы дадим краткий обзор возможностей Access как системы управления реляционной базой данных, описание ее рабочей среды, справочной системы и операций, связанных с открытием уже существующих приложений Access и созданием нового файла базы данных.

Итак, здесь будут обсуждаться следующие темы:

- Базы данных как средство хранения и обработки информации
- Запуск Access и открытие баз данных
- Создание новой базы данных
- Рабочая среда Access
- Использование справочной системы

Базы данных как средство хранения и обработки информации

В этом разделе мы познакомим вас с основными понятиями баз данных и систем управления базами данных (СУБД), дадим краткий обзор приложения Access как системы управления реляционными базами данных. А также приведем сравнение возможностей программы электронных таблиц Excel и СУБД Access.

Основные понятия

Базы данных — это совокупность сведений (о реальных объектах, процессах, событиях или явлениях), относящихся к определенной теме или задаче, организованная таким образом, чтобы обеспечить удобное представление этой совокупности, как в целом, так и любой ее части. *Реляционная база данных* представляет собой множество взаимосвязанных таблиц, каждая из которых содержит информацию об объектах определенного типа. Каждая строка таблицы включает данные об одном объекте (например, клиенте, автомобиле, документе), а столбцы таблицы содержат различные характеристики этих объектов — атрибуты (например, наименования и адреса клиентов, марки и цены автомобилей). Строки таблицы называются *записями*; все записи имеют одинаковую структуру — они состоят из *полей*, в которых хранятся атрибуты объекта. Каждое поле записи содержит одну характеристику объекта и имеет строго определенный *тип данных* (например, текстовая строка, число, дата). Все записи имеют одни и те же поля, только в них содержатся разные значения атрибутов.

Для работы с данными используются *системы управления базами данных (СУБД)*. Основные функции СУБД — это определение данных (описание структуры базы данных), обработка данных и управление данными.

Прежде чем заносить данные в таблицы, нужно определить структуру этих таблиц. Под этим понимается не только описание наименований и типов полей, но и ряд других характеристик (например, формат, критерии проверки вводимых данных). Кроме описания структуры таблиц, обычно задаются *связи* между таблицами. Связи в реляционных базах данных определяются по совпадению значений полей в разных таблицах. Например, клиенты и заказы связаны отношением *"один-ко-многим"*, т. к. одной записи в таблице, содержащей сведения о клиентах, может соответствовать несколько записей в таблице заказов этих клиентов. Если же рассмотреть отношение между преподавателями и курсами лекций, которые они читают, то это будет отношение *"многие-ко-многим"*, т. к. один преподаватель может читать несколько курсов, но и один курс может читаться несколькими преподавателями. И последний тип связей между таблицами — это отношение *"один-к-одному"*. Такой тип отношений встречается гораздо реже. Как правило, это бывает в двух случаях: запись имеет большое количество полей, и тогда данные об одном типе объектов разносятся по двум связанным таблицам, или нужно определить дополнительные атрибуты для некоторого количества записей в таблице, тогда создается отдельная таблица для этих дополнительных атрибутов, которая связывается отношением "один-к-одному" с основной таблицей.

Любая СУБД позволяет выполнять четыре простейшие операции с данными:

- *добавлять* в таблицу одну или несколько записей;
- *удалять* из таблицы одну или несколько записей;

- *обновлять* значения некоторых полей в одной или нескольких записях;
- *находить* одну или несколько записей, удовлетворяющих заданному условию.

Для выполнения этих операций используется механизм *запросов*. Результатом выполнения запросов является либо отобранное по определенным критериям множество записей из таблицы, либо изменения в таблицах. Запросы к базе формируются на специально созданном для этого языке, который так и называется — *язык структурированных запросов* (Structured Query Language, SQL).

Выжной функцией СУБД является управление данными. Под этим обычно понимают защиту данных от несанкционированного доступа, поддержку многопользовательского режима работы с данными, а также обеспечение целостности и согласованности данных.

Защита от несанкционированного доступа позволяет каждому пользователю видеть и изменять только те данные, которые ему разрешено видеть или менять. Средства, обеспечивающие многопользовательскую работу, не позволяют нескольким пользователям одновременно изменять одни и те же данные. Средства обеспечения целостности и согласованности данных не дают выполнять такие изменения, после которых данные могут оказаться несогласованными. Например, если две таблицы связаны отношением "один-ко-многим", то нельзя внести запись в таблицу на стороне "многие" (ее обычно называют подчиненной), если в таблице на стороне "один" (главной) отсутствует соответствующая запись.

Microsoft Access как настольная СУБД реляционного типа

Microsoft Access является настольной СУБД реляционного типа, которая имеет все необходимые средства для выполнения перечисленных ранее функций. Достоинством Access является то, что она имеет очень простой графический интерфейс, который позволяет грамотному пользователю (не программисту) не только создавать собственные базы данных, но и разрабатывать приложения различной степени сложности.

В отличие от других настольных СУБД, Access хранит все данные в одном файле, хотя и распределяет их по разным таблицам. Вы можете создать сколько угодно таблиц, используя те правила, которые были описаны ранее. Самым важным правилом, которое требуется соблюдать, является то, что в базе данных нужно хранить только необходимую информацию, и при этом все данные должны храниться только в одном месте, т. е. не должно быть дублирования информации.

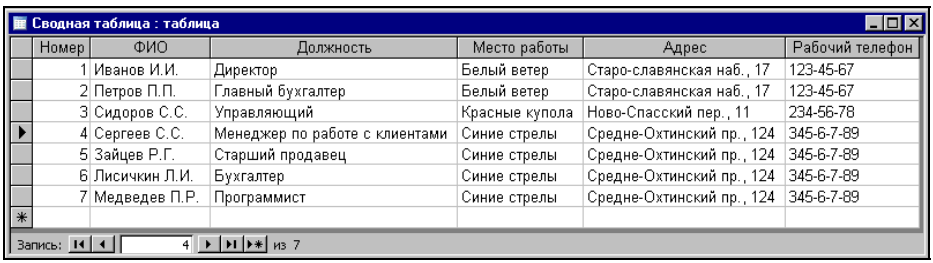
Замечание

Из всякого правила могут быть исключения, поэтому иногда требуется сделать отступление и хранить, например, промежуточные расчетные данные. Это диктуется обычно соображениями производительности. Однако общее правило все-таки следующее: если вы определяете какое-либо поле в таблице, проверьте, не хранятся ли уже эти данные где-нибудь еще. Если хранятся, то их, как правило, можно так или иначе извлечь.

"Почему же плохо иметь в таблицах поля с повторяющимися данными?" — может спросить неискушенный в области баз данных читатель. Дело в том, что это очень

неэффективный способ хранения данных. И не только потому, что они занимают лишнее место в памяти. Этот аргумент в последнее время не является таким сильным, как раньше, из-за значительного снижения цен на микросхемы памяти. Основная причина — это то, что такие данные долго вводить и трудно анализировать. Если случайно при вводе значения пользователь сделал грамматическую ошибку или даже просто ввел лишний пробел, то при запросах и группировках такое значение будет рассматриваться как самостоятельное и строка, содержащая это значение, не попадет в нужную группу или просто не будет выведена на экран. Именно поэтому при проектировании структуры баз данных стараются избегать повторения данных и создают для них отдельные таблицы. Этот процесс называется *нормализацией*.

Рассмотрим на примере, как можно избежать повторения данных в полях таблиц. На рис. 1.1 показан способ хранения данных об участниках семинара в одной таблице.



Сводная таблица : таблица

Номер	ФИО	Должность	Место работы	Адрес	Рабочий телефон
1	Иванов И.И.	Директор	Белый ветер	Старо-славянская наб., 17	123-45-67
2	Петров П.П.	Главный бухгалтер	Белый ветер	Старо-славянская наб., 17	123-45-67
3	Сидоров С.С.	Управляющий	Красные купола	Ново-Спасский пер., 11	234-56-78
4	Сергеев С.С.	Менеджер по работе с клиентами	Синие стрелы	Средне-Охтинский пр., 124	345-6-7-89
5	Зайцев Р.Г.	Старший продавец	Синие стрелы	Средне-Охтинский пр., 124	345-6-7-89
6	Лисичкин Л.И.	Бухгалтер	Синие стрелы	Средне-Охтинский пр., 124	345-6-7-89
7	Медведев П.Р.	Программист	Синие стрелы	Средне-Охтинский пр., 124	345-6-7-89

Запись: 4 из 7

Рис. 1.1. Способ хранения информации в одной таблице

Как видно из таблицы, сведения об организациях (их названия, адреса и телефоны) повторяются в разных строчках таблицы. Чтобы избежать дублирования данных, рекомендуется создать две таблицы и затем связать их. В первую таблицу нужно включить сведения об организации — и это будет *главная* таблица. Во вторую *(подчиненную)* войдут участники семинара и их должности, а вместо полной информации о месте работы — код организации, равный порядковому номеру организации в первой таблице (см. третий столбец на рис. 1.2). Далее устанавливаем связь между таблицами по полю `Код организации`. В результате главная таблица принимает вид, показанный на рис. 1.3, где реквизиты каждой организации записаны один раз и (под строкой организации) можно раскрыть список всех сотрудников данной организации, посетивших семинар *(о работе с таблицами см. также гл. 2)*.

Если вы правильно определили поля таблиц, то связи между ними Access установит сама. Для этого нужно, чтобы поля в связанных таблицах, содержащие совпадающие данные (например, `Код организации` на рис. 1.2), имели одинаковое наименование и согласующиеся типы данных.

Вы можете просматривать, изменять и удалять данные прямо в таблицах. Однако часто удобнее использовать для этого специально созданные *экранные формы*, которые могут иметь как табличный вид, когда в форме таблицы отображается сразу несколько записей, так и показывать только одну запись в виде некоторого бланка. В формах можно отображать информацию сразу из нескольких таблиц. В этом случае в основе формы будет не таблица, а запрос. *Запрос* позволяет объединять данные из нескольких связанных таблиц, выполнять вычисления над данными из других столбцов таблицы, добавлять, изменять и удалять записи в таблицах. При этом в

большинстве случаев вам не нужно изучать язык Access SQL, т. к. запрос будет сформирован автоматически в результате процесса конструирования в графическом конструкторе.

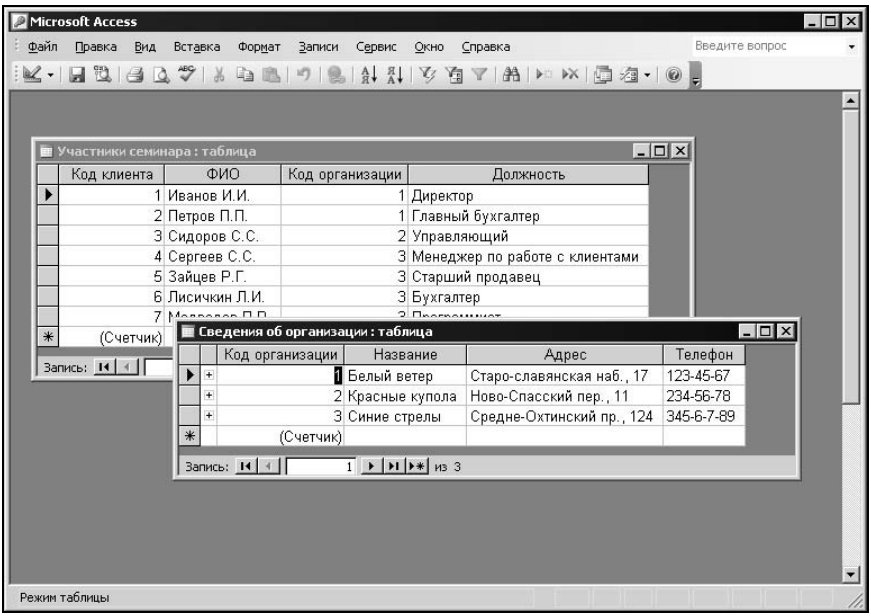


Рис. 1.2. Способ хранения информации в нескольких таблицах



Рис. 1.3. Внешний вид главной таблицы

Для того чтобы можно было анализировать хранящуюся в базе данных Access информацию, вы будете создавать разные *отчеты*, которые позволяют извлекать необходимые данные, группировать и сортировать их в нужном виде, вычислять итоговые значения по группам и в целом по всем отработанным записям. Отчеты могут быть дополнены рисунками, диаграммами, содержательными комментариями, могут быть выведены на печать, преобразованы в документ Word или опубликованы на Web-странице.

Другим удобным средством, которым вы можете пользоваться, чтобы публиковать данные в сети интранет (intranet) или Интернет (Internet), являются *страницы дос-*

типа к данным. Они не только могут содержать статическую информацию для просмотра, но могут быть динамическими, почти такими же, как формы, и позволяющими пользователю создавать запросы к базе данных, которые извлекают или изменяют нужные данные.

Для выполнения всех перечисленных операций Access предлагает большое количество Мастеров и Построителей, которые помогут быстро научиться работать с данными и добиваться желаемых результатов, а также избежать рутинных действий, которые присутствуют всегда, даже в такой творческой работе, как разработка приложений в Access.

Различия между СУБД Access и электронными таблицами Excel

На первый взгляд СУБД Access очень похожа по своему назначению и возможностям на программу электронных таблиц Microsoft Excel. Однако между ними существуют принципиальные различия.

- При работе с электронной таблицей Excel вы можете в ячейку таблицы внести любую информацию, вследствие чего на рабочем листе можно разместить и таблицу, и текст, и рисунок. В один столбец таблицы могут при необходимости вноситься разные данные — числа, текст, даты. Таблица в базе данных Access отличается от таблицы Excel тем, что в ней, как уже говорилось ранее, для каждого поля записи определен тип данных, т. е. нельзя в одном столбце таблицы в разные строки заносить данные разных типов.
- Access позволяет не только вводить данные в таблицы, но и контролировать их правильность. Для этого необходимо установить правила проверки прямо на уровне таблицы. Тогда каким бы образом не вводились данные — прямо в таблицу, через экранную форму или на странице доступа к данным, Access не позволит сохранить в записи те данные, которые не удовлетворяют заданным правилам.
- С таблицами Excel удобно работать, если они содержат ограниченное количество строк. Таблицы баз данных могут включать в себя огромное количество записей, и при этом СУБД обеспечивает удобные способы извлечения из этого множества нужной информации.
- Если все необходимые для работы данные вы будете хранить в документах Word и электронных таблицах, то по мере накопления информации вы просто можете запутаться в большом количестве файлов. Access позволяет хранить все данные в одном файле и осуществлять доступ к этим данным постранично, т. е. не превышая ограничений на ресурсы памяти компьютера.
- В Access возможно создание связей между таблицами, что позволяет совместно использовать данные из разных таблиц. При этом для пользователя они будут представляться одной таблицей. Реализовать такую возможность в системах управления электронными таблицами сложно, а иногда просто невозможно.
- Устанавливая взаимосвязи между отдельными таблицами, Access позволяет избежать ненужного дублирования данных, сэкономить память компьютера, а также увеличить скорость и точность обработки информации. Для этого таблицы, содержащие повторяющиеся данные, разбивают на несколько связанных таблиц.

- Excel позволяет работать нескольким пользователям с одним документом, однако эти возможности очень ограничены. Access может поддерживать одновременную работу с базой данных 50 пользователей, при этом все пользователи гарантированно будут работать с актуальными данными.
- Access имеет развитую систему защиты от несанкционированного доступа, которая позволяет каждому пользователю или группе пользователей видеть и изменять только те объекты, на которые ему выданы права администратором системы, например, можно запретить использование некоторых команд меню, открытие определенных форм, изменение данных в таблицах или формах. Электронные таблицы тоже позволяют защитить данные паролем или разрешить только просмотр данных, однако эти средства существенно проще.

Таким образом, СУБД Access применяется в тех случаях, когда прикладная задача требует хранения и обработки разнородной информации о большом количестве объектов и предполагает возможность многопользовательского режима. Примером может служить задача ведения складского учета. Электронные таблицы Excel являются удобным средством хранения ограниченного количества табличных данных с широкими возможностями выполнения расчетов и анализа данных. Документы Excel предназначаются, как правило, для индивидуального использования.

Замечание

Возможности Excel, конечно, не ограничиваются только электронными таблицами, например, Excel очень удобно использовать для анализа данных, которые хранятся на сервере базы данных (в частности, Microsoft SQL Server), но в этом случае Excel не является хранилищем данных, а обеспечивает только их обработку.

Запуск Access и открытие баз данных

После короткого обзора возможностей Access перейдем к описанию начальных действий по запуску Access и открытию файлов баз данных.

Замечание

Мы не описываем здесь процесс установки Microsoft Office Access 2003, т. к. подробную информацию об установке можно найти в документации, которая входит в комплект поставки.

Для того чтобы запустить Microsoft Office Access 2003, необходимо:

1. Нажать кнопку **Пуск** (Start) на Панели задач в нижней части рабочего стола.
2. Открыть в главном меню пункт **Программы** (Programms).
3. Выбрать программу **Microsoft Office Access 2003**.

Базой данных Access является файл, который имеет расширение mdb. Этот файл может содержать не только все таблицы, но и другие объекты приложений Access — запросы, формы, отчеты, страницы доступа к данным, макросы и модули.

Замечание

В дальнейшем вы узнаете, что это не всегда бывает так. Иногда приложение состоит из нескольких файлов, но пока мы рассматриваем простейший вариант.

При запуске Access появляется главное окно **Microsoft Access** и область задач справа.

Для того чтобы открыть существующую базу данных, можно воспользоваться одним из трех способов.

- Выбрать в меню **Файл** (File) нужную базу данных из списка ранее открывавшихся файлов (рис. 1.4).

Замечание

Если такой список в меню **Файл** (File) отсутствует, нужно с помощью команды **Сервис | Параметры** (Tools | Options) открыть диалоговое окно **Параметры** (Options), раскрыть вкладку **Общие** (General) и установить флажок **Помнить список файлов** (Recently used file list). По умолчанию хранятся названия последних четырех используемых файлов, но это число можно изменить, если выбрать из списка, находящегося рядом с этим флажком, другое число. Максимальное количество хранимых имен файлов 9.

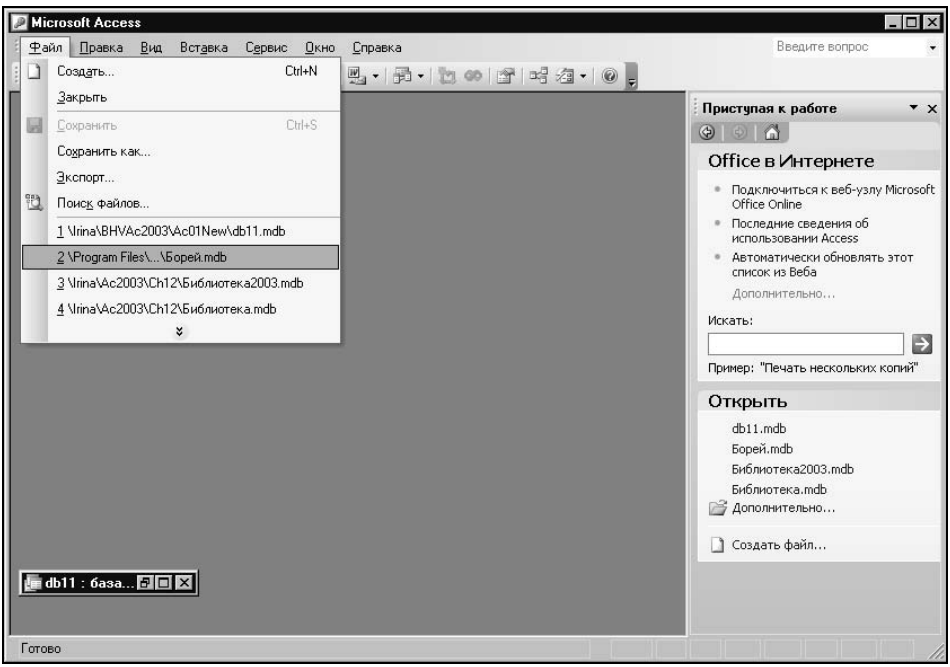


Рис. 1.4. Открытие базы данных из меню **Файл**

- Выбрать файл из списка **Открыть** (Open) в области задач **Приступая к работе** (Getting Started), которая расположена в правой части окна приложения (см. рис. 1.4). Если нужного файла нет в списке (а в нем отображаются те же файлы, что и в меню **Файл**), то нажмите кнопку **Дополнительно** (More). *(Об области задач см. разд. "Рабочая среда Access" данной главы.)*
- Выбрать команду **Открыть** (Open) в меню **Файл** (File) и затем найти нужный файл в диалоговом окне **Открытие файла базы данных** (Open) (рис. 1.5).

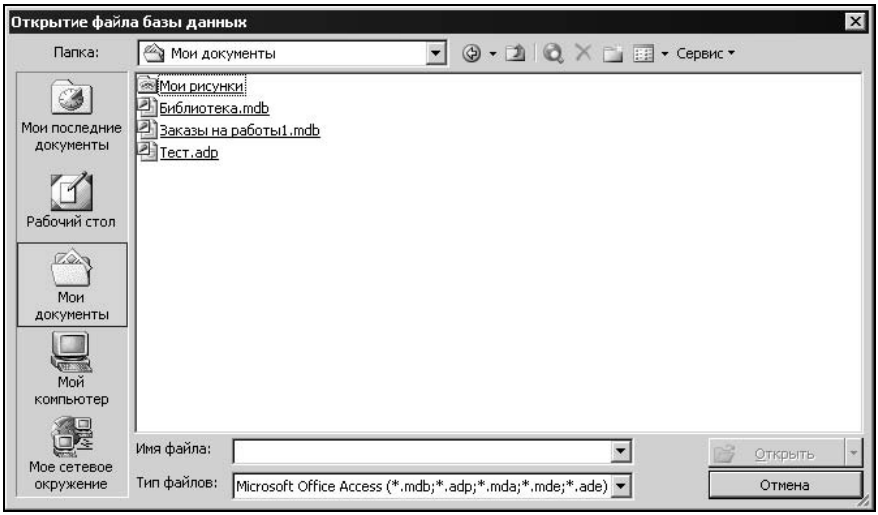


Рис. 1.5. Диалоговое окно **Открытие файла базы данных**

- Нажать кнопку **Открыть** (Open) на панели инструментов (рис. 1.6), что также приведет к открытию диалогового окна **Открытие файла базы данных** (Open).

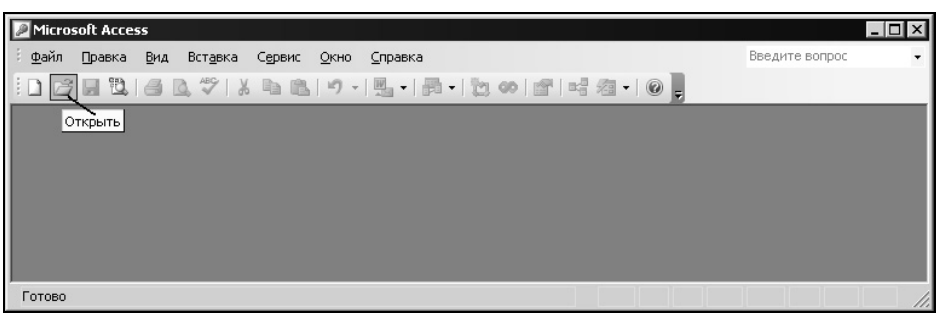


Рис. 1.6. Открытие базы данных с помощью кнопки **Открыть** на панели инструментов

В диалоговом окне **Открытие файла базы данных** обычно по умолчанию отображается содержимое папки Мои документы (My Documents) или Личная (Personal) (в зависимости от установленной на компьютере операционной системы). Найдя в списке необходимую базу данных, нужно выделить файл, щелкнув по нему, и нажать кнопку **Открыть** (Open) или дважды щелкнуть по этому файлу. В главном окне Microsoft Access появится окно выбранной базы данных (рис. 1.7).

Замечание

Вы можете легко в новой версии Access открывать файлы баз данных, созданные в версиях Access 2000 или Access 2002, и работать с ними, не преобразовывая в новый формат. Однако если вы хотите работать с файлами, созданными в более ранних версиях Access, рекомендуем вам ознакомиться предварительно с материалом *гл. 19*.

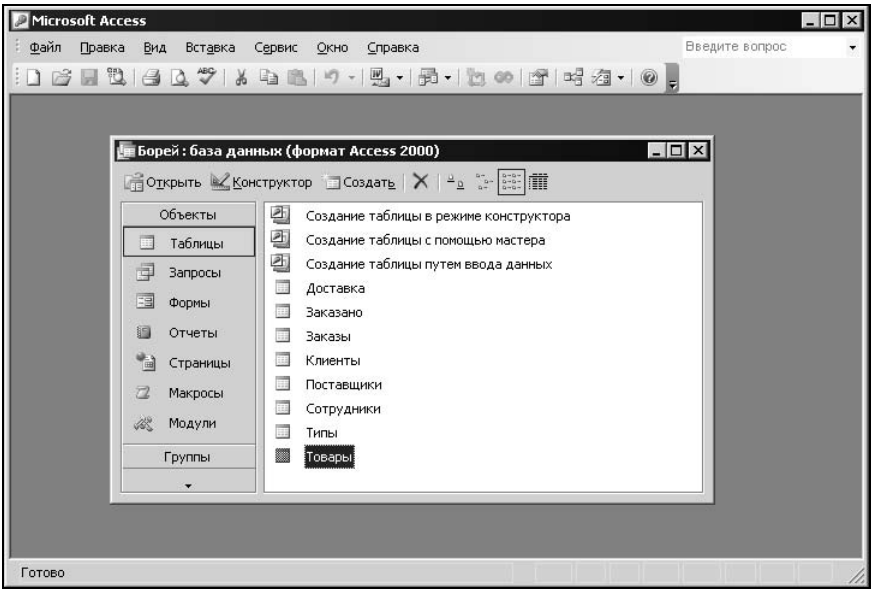


Рис. 1.7. Пример базы данных

Файлы можно выбирать не только из списка, который появляется в диалоговом окне **Открытие файла базы данных** (Open) при его открытии. Можно выбрать нужную папку либо из раскрывающегося списка **Папка** (Look in), либо использовать значки, находящиеся на так называемой панели адресов слева.

При выборе значка **Мои последние документы** (My Recent Documents) (см. рис. 1.5) в диалоговом окне появляется список ярлыков к последним открывавшимся файлам и папкам.

При выборе значка **Рабочий стол** (Desktop) появляется список, содержащий ярлыки папок и файлы, находящиеся в данный момент на рабочем столе Windows.

Можно открыть файл базы данных, находящийся на сетевом диске. Для этого используйте значок **Мое сетевое окружение** (My Network Places) на панели адресов. Для того чтобы подключить сетевой диск (если он отсутствует в списке ярлыков **Мое сетевое окружение**), необходимо выполнить команду **Подключить сетевой диск** (Map Network Drive) из списка команд кнопки **Сервис** (Tools) на панели инструментов.

Любую папку, которую вы нашли и содержимое которой отображается в данный момент в диалоговом окне, можно добавить на панель адресов. Потом ее легко будет находить. Для этого нужно выполнить команду **Добавить на панель адресов** (Add to "My Places") из списка команд кнопки **Сервис** (Tools).

Тип файлов, отображаемых в диалоговом окне, выбирается в раскрывающемся списке **Тип файлов** (File of type) в нижней части окна. По умолчанию это все типы файлов, с которыми может работать Microsoft Office Access.

Если вы не можете найти нужный файл базы данных, его можно разыскать, задавая критерии поиска в специальном диалоговом окне **Поиск файлов** (File Search), кото-

рое появляется, если вы щелкните по кнопке **Сервис** (Tools) и выберите из списка команду **Найти** (Search).

Замечание

Существует специальная *область задач* **Поиск файлов** (File Search), которая тоже может быть использована для обычного и расширенного поиска нужного файла. Она содержит те же самые критерии, что и диалоговое окно **Поиск файлов** (File Search). *(Подробную информацию об этой панели см. в гл. 7.)*

Открыть файл можно в одном из четырех режимов:

- **Открыть** (Open) — режим коллективного доступа. Когда вы открываете файл базы данных в этом режиме, доступ к нему будет разрешен другим пользователям, они могут открыть его и работать с объектами базы данных одновременно с вами. Причем все пользователи будут иметь возможность как просматривать данные, так и изменять их.
- **Открыть монопольно** (Open Exclusive) — режим монопольного доступа. В этом случае никто другой не сможет открыть этот файл, пока вы его используете.
- **Открыть для чтения** (Open Read-Only) — режим коллективного доступа только для чтения. В этом случае вы не сможете изменять ни данные в таблицах, ни другие объекты базы данных.
- **Открыть монопольно только для чтения** (Open Exclusive Read-Only) — режим монопольного доступа только для чтения. Вы открываете файл только для чтения и одновременно запрещаете другим пользователям открывать этот файл.

Выбрать нужный режим вы можете из списка, который раскрывается при нажатии стрелки справа от кнопки **Открыть** (Open). По умолчанию (при двойном щелчке мышью по имени файла или нажатии кнопки **Открыть** (Open)) файл базы данных открывается в режиме коллективного доступа.

При открытии файла базы данных вы, скорее всего, получите сообщение системы безопасности Access о том, что в приложении не заблокированы небезопасные выражения. Это новое средство борьбы с вирусами, которое появилось в Access 2003. В состав небезопасных выражений входят функции, которые могут быть использованы хакерами для несанкционированного доступа к ресурсам компьютера. Заблокировать небезопасные выражения можно только в том случае, если на компьютере установлен пакет обновления Microsoft Jet 4.0 Service Pack 8 или более поздней версии. Чтобы сделать базы данных (и проекты) Access более устойчивыми к атакам вирусов, необходимо использовать изолированный режим работы служб ядра базы данных Microsoft Jet 4.0. Изолированный режим позволяет сохранять полную функциональность приложения, блокируя при этом небезопасные выражения. Если пакет обновления установлен, то указанное сообщение не будет выдаваться. При этом в приложении (базе данных Access) будет использоваться изолированный режим и блокироваться ряд функций в том случае, если установлен высокий или средний уровень безопасности (по умолчанию устанавливается средний уровень безопасности). Если же установлен низкий уровень безопасности, то это сообщение выдаваться не будет, но при этом приложение не будет работать в изолированном режиме. Если, не смотря на полученное предупреждение, вы все-таки откроете файл базы данных, то она также не будет работать в изолированном режиме.

Уровень безопасности приложения устанавливается в диалоговом окне **Безопасность** (Security), которое открывается при выполнении команды меню **Сервис | Макрос | Безопасность** (Tools | Macro | Security).

Создание новой базы данных

Для создания новой базы данных вы можете:

- выбрать команду **Создать** (New) из меню **Файл** (File);
- нажать кнопку **Создать** (New) на панели инструментов. На рис. 1.6 она расположена слева от кнопки **Открыть** (Open);
- если на экране отображена область задач **Приступая к работе** (Getting Started), то выполнить команду **Создать файл** (Create a new file) (см. рис. 1.4).

Любое из этих действий приведет к открытию справа области задач **Создание файла** (New file) (рис. 1.8). Далее нужно использовать команды этой области.

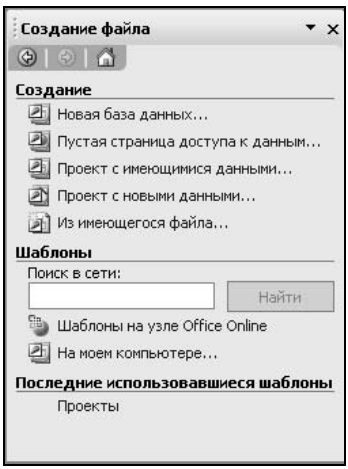


Рис. 1.8. Область задач **Создание файла**

Замечание

При создании новой базы данных в процессе работы в Access прежняя база данных будет автоматически закрыта. Обратите внимание, что две базы данных в одном окне приложения Access не могут быть открыты одновременно!

Создать новую базу данных можно тремя способами:

- создать пустую базу данных;
- создать копию существующей базы данных;
- создать базу данных с использованием шаблона.

Создание новой пустой базы данных

Для того чтобы создать новую пустую базу данных, необходимо выполнить ряд действий.

1. В области задач **Создание файла** (New file) из раздела **Создание** (New) выбрать элемент **Новая база данных** (Blank Database) (см. рис. 1.8). Появится диалоговое окно **Файл новой базы данных** (File New Database) с содержимым папки Мои документы (My Documents) (рис. 1.9).

Замечание

Если на вашем компьютере установлена операционная система Windows NT Workstation, то это будет папка Личная. Папки Личная и Мои документы по умолчанию предназначены для хранения документов Microsoft Office, поэтому их многие используют для хранения рабочих документов.

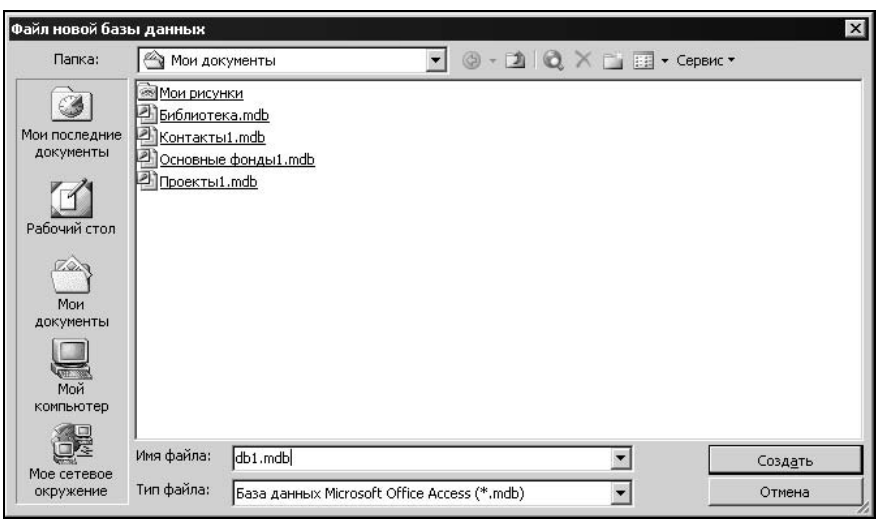


Рис. 1.9. Диалоговое окно **Файл новой базы данных**

2. В раскрывающемся списке **Папка** (Save in) открыть папку, в которой будет храниться создаваемая база данных, или выбрать папку одним из методов, описанных в *разд. "Запуск Access и открытие баз данных" этой главы.*

3. В поле ввода **Имя файла** (File name) указать имя создаваемого файла и нажать кнопку **Создать** (Create). По умолчанию Access присваивает новой базе данных имя db1, а если база с именем db1 уже существует, то db2 и т. д. После нажатия кнопки появится окно новой базы данных (рис. 1.10).

Дальше можно создавать таблицы и другие объекты базы данных. Работа с таблицами баз данных описана в *гл. 2*, в частности в *разд. "Создание таблиц".*

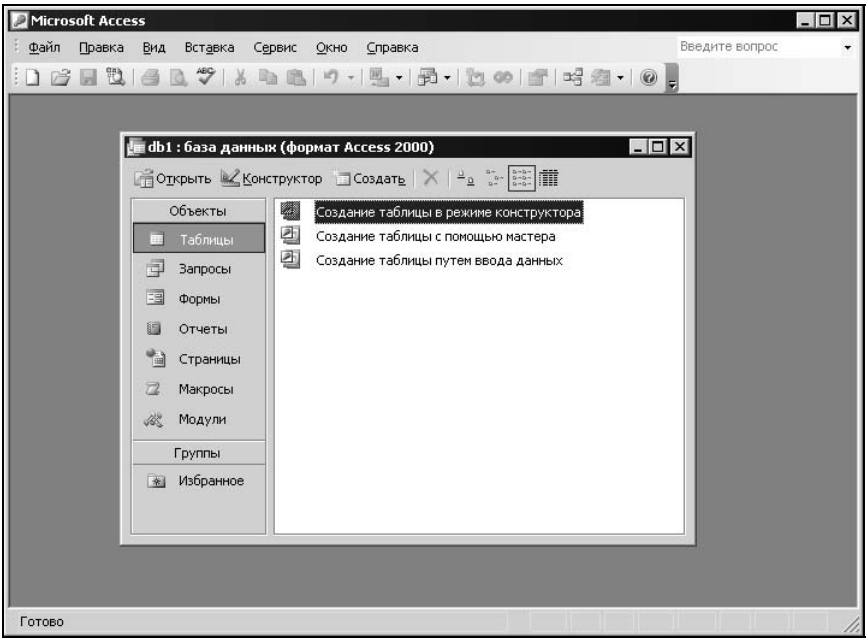


Рис. 1.10. Окно новой базы данных Access 2003

Создание копии существующей базы данных

Чтобы создать копию существующей базы данных, необходимо следовать приведенной далее инструкции.

1. В области задач **Создание файла** (New file) из раздела **Создание** (New) выбрать команду **Создание из имеющегося файла** (From existing file).
2. С помощью открывшегося диалогового окна **Создание из имеющегося файла** (New from existing file) выбрать файл базы данных, копию которого вы хотите создать. Это диалоговое окно будет идентично окну **Открытие файла базы данных** (Open) (см. рис. 1.5) *(см. разд. "Запуск Access и открытие баз данных" этой главы)*.
3. Щелкнуть мышью по кнопке **Создать новый** (Create New).

При этом будет создана копия выбранного файла базы данных, во всем идентичная оригиналу (т. е. и по данным, и по структуре базы данных), которая будет немедленно сохранена под именем `<имя файла>1.mdb` и открыта. Здесь `<имя файла>` — это имя выбранного файла.

Создание базы данных с использованием шаблона

Для создания новой базы данных можно воспользоваться соответствующим заранее заготовленным шаблоном. Этот шаблон теперь можно найти на локальном компьютере, в локальной сети и на узле Microsoft Office Online. Соответствующие команды

есть в области задач **Создание файла** (New file) в разделах **Шаблоны** (Templates) и **Последние использовавшиеся шаблоны**.

Чтобы поискать шаблон в локальной сети, введите в поле **Поиск в сети** (Search Online for) одно или несколько ключевых слов шаблона и нажмите кнопку **Найти** (Go). Если соответствующий шаблон будет найден, то ссылка на него будет выведена в область задач **Поиск** (Search Results).

Можно искать шаблон на узле Microsoft Office Online. Для этого выберите в разделе **Шаблоны** (Templates) команду **Шаблоны на узле Office Online** (Templates on Office Online). В этом случае вы попадаете на страничку Шаблоны Microsoft Office и там пытаетесь найти нужный шаблон.

Можно использовать шаблоны, которые установлены на вашем локальном компьютере. Они копируются при установке Access по умолчанию в папку Program Files\Microsoft Office\Templates\1049 и представляют собой файлы с расширением mdz.

Чтобы использовать шаблоны на локальном компьютере:

1. Выберите в разделе **Шаблоны** (Templates) команду **На моем компьютере** (On my computer). Появится диалоговое окно **Шаблоны** (Templates), в котором раскрыта вкладка **Базы данных** (Databases), которая содержит ярлыки для имеющихся шаблонов (рис. 1.11).

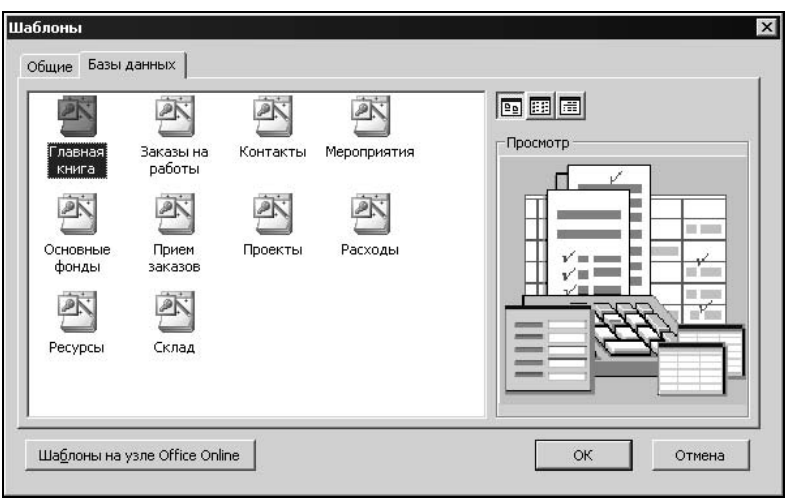


Рис. 1.11. Диалоговое окно **Шаблоны**

2. Выберите ярлык шаблона, который кажется вам более всего подходящим для создаваемой базы данных, и нажмите кнопку **ОК**. Запустится Мастер базы данных. Далее нужно следовать указаниям Мастера.
3. Вначале вас попросят задать название и место размещения новой базы данных, для чего нужно использовать диалоговое окно **Файл новой базы данных** (File New Database). После этого создается новая база данных (сначала пустая) и открыва-

ется. В диалоговом окне Мастера приводятся краткие сведения о будущем содержании базы. Нажмите на кнопку **Далее** (Next). После этого вам будут показаны таблицы, используемые в базе, и поля этих таблиц (рис. 1.12).

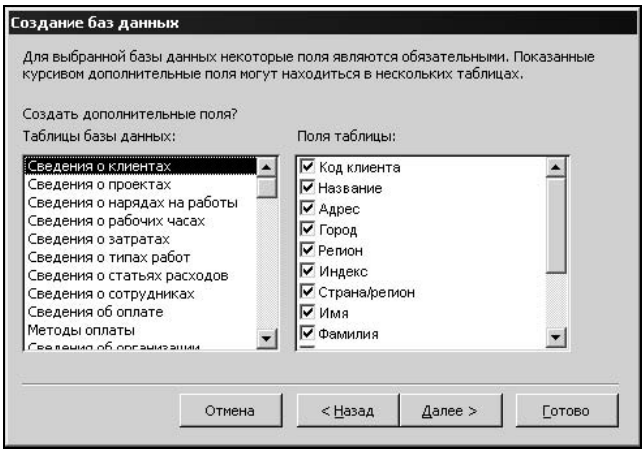


Рис. 1.12. Диалоговое окно выбора шаблонов для таблиц Мастера баз данных

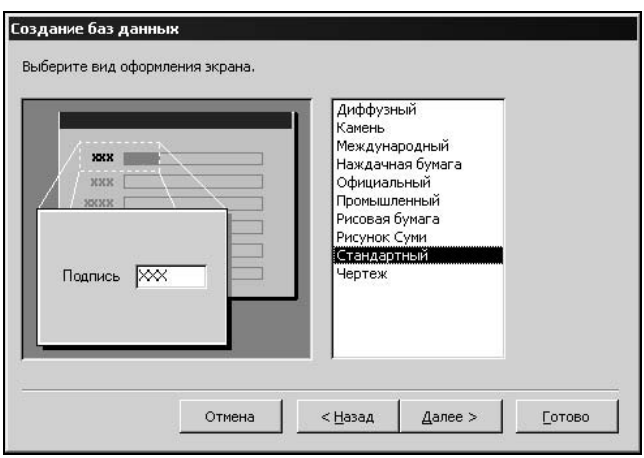


Рис. 1.13. Диалоговое окно стилевого оформления форм Мастера баз данных

4. Следующие шаги (переход к каждому следующему шагу осуществляется нажатием кнопки **Далее** (Next) позволяют выбрать стилевое оформление форм (рис. 1.13) и отчетов (рис. 1.14), а также задать заголовок главного меню новой базы данных и файл рисунка, используемого при печати отчетов (рис. 1.15).
5. Наконец, на последнем шаге можно уведомить Access об отказе немедленного открытия вновь созданной базы (для этого необходимо снять флажок **Да** (Yes) в последнем окне Мастера), после чего нажать кнопку **Готово** (Finish).

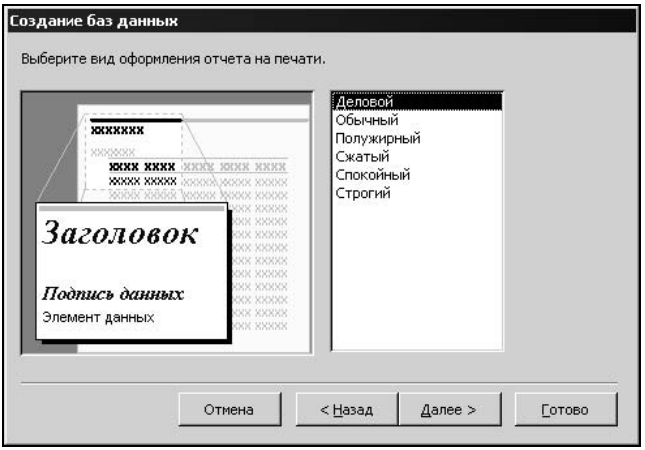


Рис. 1.14. Диалоговое окно стилевого оформления отчетов Мастера баз данных

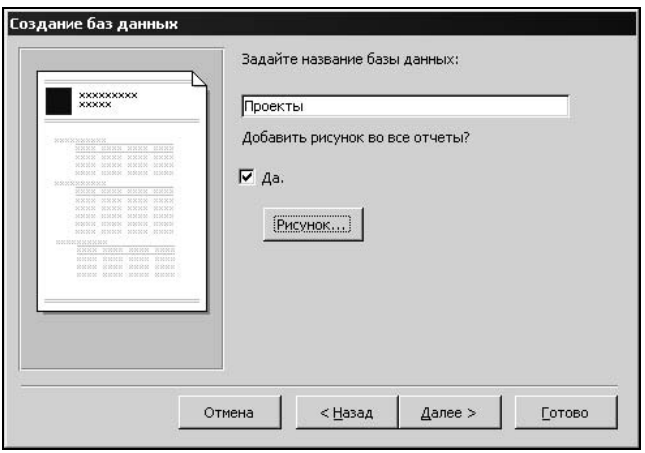


Рис. 1.15. Диалоговое окно Мастера баз данных, задающее заголовок базы данных и файл логотипа отчетов

Если на каком-то шаге вы прервете процесс, нажав кнопку **Отмена** (Cancel), то база данных создана не будет.

Формат файла базы данных

По умолчанию при создании новой базы данных создается файл в формате Access 2000. Файл этого формата можно открыть и работать с ним без преобразования в нескольких версиях Access: Access 2000, Access 2002 и Access 2003. Впервые в новой версии Access не сменился формат файла базы данных, т. е. "родным" форматом для Access 2003 является тот же формат, что был в Access 2002. Чтобы изменить формат файла, нужно выполнить его преобразование. *(Информацию о преобразовании формата см. в гл. 19.)*

Если вы хотите работать с файлами баз данных в формате Access 2002, то можно сменить установку по умолчанию. Для этого сделайте следующее:

1. Выполните команду меню **Сервис | Параметры** (Tools | Options).
2. В диалоговом окне **Параметры** (Options) откройте вкладку **Другие** (Advanced) (рис. 1.16).
3. В списке **Формат файла по умолчанию** (Default File Format) выберите **Access 2002 – 2003**.

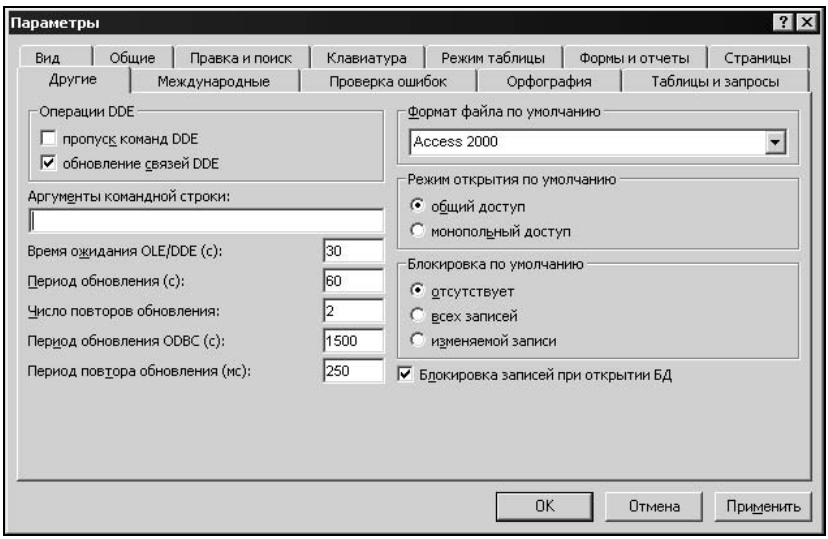


Рис. 1.16. Диалоговое окно **Параметры**

Свойства файла базы данных

Чтобы посмотреть свойства файла базы данных, нужно выполнить команду меню **Файл | Свойства базы данных** (File | Database Properties). Появится диалоговое окно, которое имеет 5 вкладок, отображающих свойства открытого в данный момент файла MDB.

На вкладке **Общие** (General) отображается полный путь и наименование файла, его размер, время создания и последней модификации, а также атрибуты файла — **Только чтение** (Read only), **Скрытый** (Hidden), **Архивный** (Archive), **Системный** (System) (рис. 1.17). Изменить атрибуты файла в этом окне нельзя — они открыты только для просмотра.

На вкладке **Документ** (Summery) отображается ряд свойств, которые можно использовать для поиска нужного файла: **Название**, **Тема**, **Автор**, **Ключевые слова** и т. д.

На вкладке **Статистика** (Statistics) указаны такие сведения, как дата и время создания файла, дата и время его последнего изменения, автор изменений, число правок, а также общее время правок.



Рис. 1.17. Вкладка **Общие** диалогового окна **Свойства**

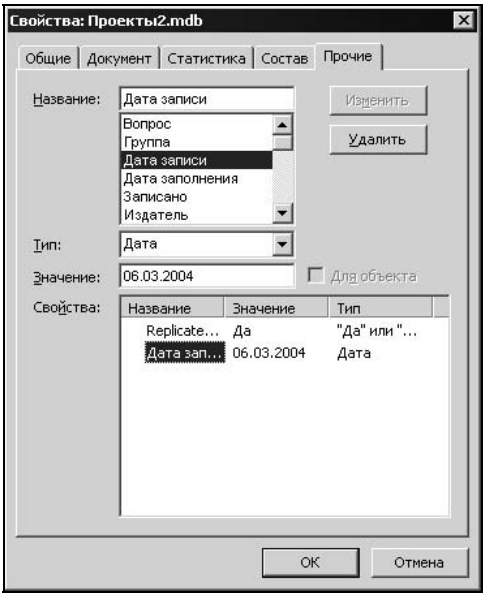


Рис. 1.18. Вкладка **Прочие** диалогового окна **Свойства**

На вкладке **Состав** (Contents) выводится упорядоченный список объектов приложения.

На последней вкладке **Прочие** (Custom) можно задать дополнительные свойства файла (они так и называются — пользовательские), которые можно использовать при поиске файла. Название свойства можно либо выбрать из списка, либо ввести свое. Затем нужно указать тип этого свойства и значение и нажать кнопку **Добавить** (Add) (рис. 1.18). Тогда это свойство появится в списке свойств в нижней половине окна. Лишние свойства можно убрать, если выбрать их в списке и нажать кнопку **Удалить** (Delete).

Чтобы изменить значение свойства, нужно выполнить следующие действия:

1. Выделить свойство в списке **Свойства** (Properties); тогда название, тип и значение свойства отобразятся в соответствующих полях.
2. Изменить тип или значение свойства.
3. Нажать кнопку **Изменить** (Modify).

Рабочая среда Access

Открыв базу данных Access, пользователь видит на экране главное окно Access и окно базы данных. Главное окно Access имеет вид, аналогичный окнам других приложений Microsoft Office. В верхней части окна размещается его заголовок, меню и панель инструментов, а в нижней части — строка состояния. Наиболее важным элементом интерфейса в среде Access является окно базы данных. Оно является

контейнером, содержащим все объекты базы данных. Доступ к этим объектам возможен только через окно базы данных.

На рис. 1.19 показано главное окно Access 2003, в котором открыто несколько таблиц в режиме Таблицы. Далее мы прокомментируем этот рисунок, а заодно определим используемые термины.

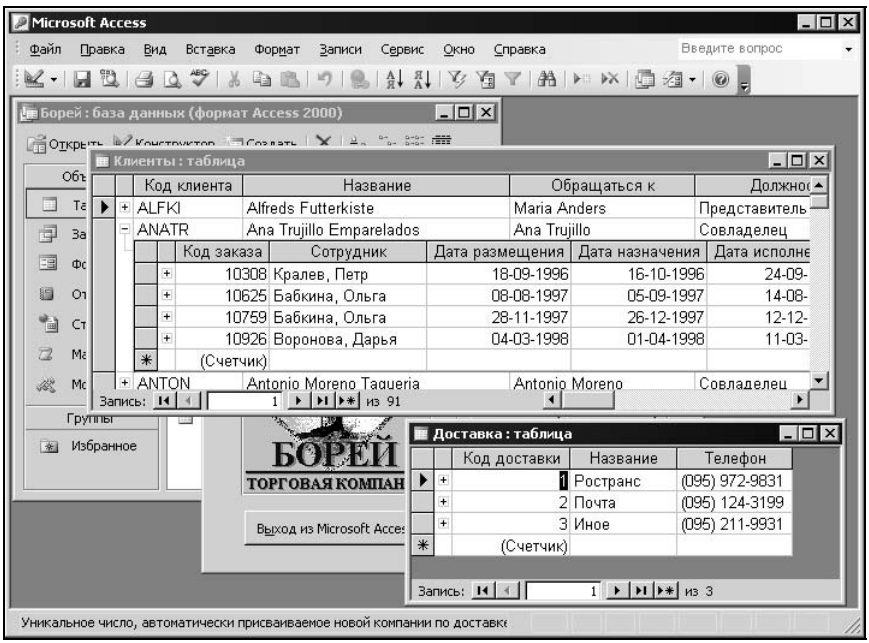


Рис. 1.19. Главное окно Access 2003 с окном базы данных и несколькими таблицами

Окна рабочей среды Access

Каждое запущенное приложение Windows имеет свое окно, в котором выводятся результаты его работы — *окно приложения*. В данном случае окном приложения является главное окно Access. Кроме окна приложения, в Access широко используются еще два типа окон: *диалоговые окна* и *окна документов*. Окна приложения и окна документов могут перемещаться по экрану монитора, имеют изменяемые размеры и могут быть свернуты, развернуты или развернуты до максимальных размеров. Диалоговые окна также можно перемещать по экрану, но они имеют постоянные размеры и их нельзя свернуть или развернуть.

Строка в верхней части окна приложения, содержащая название приложения, является *заголовком* окна приложения. Заголовок окна приложения используется для перемещения окна по экрану (только если окно приложения не свернуто и не развернуто до максимальных размеров). Окно документа, так же как и окно приложения, имеет заголовок, который содержит название этого документа.

Строка состояния в нижней части окна приложения или документа служит для отображения подсказок и сведений о выполняемых операциях.

Большинство объектов базы данных Access: таблицы, запросы, формы, отчеты, страницы доступа к данным и макросы — выводятся в окнах, которые в терминологии Windows являются окнами документов. Исключение составляют программные модули — для их представления используется специальная среда программирования *(см. гл. 13)*.

Access 2003 поддерживает *интерфейс множественных документов* (Multiple Document Interface, MDI), который позволяет открыть в рабочем поле окна приложения несколько окон документов одновременно.

На рис. 1.19 показаны в рабочем поле окна приложения окно базы данных и три окна документов, из которых одно окно развернуто и два — свернуты.

Окна приложения и все окна документов Access 2003 по умолчанию отображаются на Панели задач Windows. Любое из этих окон легко активизируется, путем щелчка кнопкой мыши по соответствующему значку.

Активным окном называется окно, в которое направляются все сообщения (действия) от мыши или клавиатуры. Заголовок окна активного приложения или документа подсвечивается цветом, выбранным при установке параметров оформления рабочего стола (обычно это синий цвет). Если окно приложения и окно документа в нем являются активными одновременно, то все сообщения от клавиатуры и мыши получает окно документа.

Одновременно могут быть активными только одно окно приложения и одно окно документа в этом приложении. Все остальные окна считаются *неактивными* (их заголовок выводится на сером фоне). Щелчок левой кнопкой мыши по неактивному окну делает его активным и переводит на передний план. Если нужное окно целиком закрыто другими окнами, то сделать его активным можно, выбрав название документа в меню **Окно** (Window), либо щелкнув по соответствующему значку на Панели задач.

Чтобы закрыть окно документа или приложения, необходимо нажать кнопку **Закрыть** в заголовке окна справа (на ней изображен крестик).

Нажатие кнопки **Развернуть** в заголовке окна приложения (на ней изображен квадратик) позволяет установить максимальные размеры окна приложения. Нажатие кнопки **Развернуть** в заголовке окна документа позволяет максимизировать его в окне приложения. В заголовке развернутого окна (как документа, так и приложения) вместо кнопки **Развернуть** появляется кнопка **Восстановить** (с изображением двух наложенных квадратиков), нажатие которой вызывает возвращение окну прежних размеров. Кнопка **Свернуть** (на ней изображен знак подчеркивания) позволяет минимизировать окно приложения или документа. Окно приложения при этом свертывается в значок на Панели задач Windows, а окно документа — в значок в нижней части окна приложения. Обратную ей функцию выполняет кнопка **Восстановить**. Она позволяет вернуть окну те размеры, которые оно имело до применения операций развертывания или свертывания.

Максимально развернутое окно документа занимает все поле окна приложения. Кнопки **Свернуть**, **Восстановить** и **Закрыть** при этом перемещаются в правую часть строки главного меню приложения.

Замечание

Будьте осторожны и не путайте кнопки окна приложения и окна документа, которые при этом оказываются близко (друг под другом), иначе вместо окна документа вы можете закрыть окно приложения.

Если развернуть до максимального размера одно из окон документов, то все другие окна документов будут открываться максимально развернутыми. Если одно из таких окон свернуть к обычному размеру с помощью кнопки **Восстановить**, то все остальные окна документов будут также восстановлены в своих обычных размерах.

Меню и панели инструментов

В Access 2003, так же как и в других приложениях Microsoft Office, используются два типа меню: раскрывающиеся и контекстные. Контекстное меню появляется на экране при щелчке правой кнопкой мыши по объекту, с которым вы хотите работать. Оно содержит перечень действий, которые можно выполнять с данным объектом в данном контексте.

Примером раскрывающегося меню может служить системное меню окна приложения или документа. *Системное меню окна приложения* появляется при щелчке левой кнопкой мыши по значку приложения, который находится в левой части заголовка окна приложения, или при нажатии комбинации клавиш <Alt>+<Пробел>. *Системное меню документа* открывается после щелчка левой кнопкой мыши по значку документа, который находится в левой части заголовка окна документа, или при нажатии комбинации клавиш <Alt>+<->. Системное меню предназначено для изменения параметров окна.

Строка под заголовком окна приложения содержит *главное меню приложения*, с помощью которого можно получить доступ к командам меню. Вид главного меню Access 2003 зависит от режима, в котором находится приложение. Например, когда активным окном является окно таблицы (см. рис. 1.19), в главном меню появляется команда **Записи** (Records), которая отсутствует в других режимах работы.

Ниже строки меню располагается *панель инструментов*, на которой находятся кнопки. Многие кнопки имеют тот же внешний вид и назначение, что и в других приложениях Microsoft Office, но все-таки большая часть кнопок предназначена для выполнения специфических функций Access. Кнопки панели инструментов часто дублируют команды меню. Access 2003 имеет 27 стандартных панелей инструментов, которые выводятся на экран в зависимости от того, какой объект активен в данный момент.

Все кнопки панелей инструментов имеют "всплывающие" подсказки, объясняющие их назначение. Подсказки появляются, если подвести к кнопке указатель мыши и немного задержать. Если подсказки не появляются, значит, они отключены. Чтобы включить режим вывода подсказок, выполните команду меню **Вид | Панели инструментов | Настройка** (View | Toobars | Customize) и в диалоговом окне **Настройка** (Customize) на вкладке **Параметры** (Options) установите флажок **Отображать подсказки для кнопок** (Show ScreenTips on toolbars).

Панели инструментов могут фиксироваться вдоль границы окна приложения, как показано на рис. 1.19, а могут быть плавающими, т. е. перемещаемыми по экрану.

В этом случае панель инструментов будет иметь заголовок и может менять свою форму и размеры. Левый край панели инструментов помечен вертикальной линией. Если подвести указатель мыши к этой линии, нажать левую кнопку мыши и, не отпуская, потянуть вправо, панель будет перемещаться вдоль границы окна. А если потянуть вниз, панель инструментов превратится в плавающую. Такую панель можно перемещать по экрану, зацепив за заголовок.

Меню и панели инструментов по умолчанию отображают лишь наиболее часто используемые команды и кнопки. Чтобы получить полный список команд меню, необходимо щелкнуть мышью по двойной стрелке у нижнего края раскрывающегося меню. Если не все кнопки поместились на панели инструментов, то у ее правого края тоже появляется значок двойной стрелки. Если щелкнуть мышью по нему, будут показаны оставшиеся кнопки. Если вы использовали какую-то кнопку или команду, которой не было по умолчанию в соответствующей панели или меню, то в дальнейшем она будет там видна.

Меню и панели инструментов Access являются настраиваемыми, т. е. можно изменять состав кнопок, удаляя ненужные или добавляя новые кнопки *(см. разд. "Командный интерфейс приложения" гл. 14)*.

При открытии файла базы данных в окне приложения отображается окно базы данных и соответствующая панель инструментов **База данных** (Database). Перечень кнопок этой панели приведен в табл. 1.1, а сама панель инструментов — на рис. 1.20.



Рис. 1.20. Панель инструментов **База данных**

Таблица 1.1. *Панель инструментов* ***База данных***

Кнопка	Описание	Команда меню
Создать (New)	Создание новой базы данных или проекта Access. Отображается область задач **Создание файла**	**Файл** | **Создать** (File | New)
Открыть (Open)	Открытие уже существующего файла базы данных	**Файл** | **Открыть** (File | Open)
Сохранить (Save)	Сохранение активной таблицы	**Файл** | **Сохранить** (File | Save)
Поиск файлов (File Search)	Поиск документов по содержанию	**Файл** | **Найти** (File | File Search)
Печать (Print)	Печать выделенного объекта Access без открытия диалогового окна **Печать**. Недоступна для страниц и модулей	**Файл** | **Найти** (File | Print)

Таблица 1.1 *(окончание)*

Кнопка	Описание	Команда меню
Предварительный просмотр (Print Preview)	Предварительный просмотр размещения объекта Access на листе перед печатью. Недоступна для страниц и модулей	**Файл** | **Предварительный просмотр** (File | Print Preview)
Орфография (Spelling)	Проверка орфографии в текстовых полях выделенного объекта. Доступна только для таблиц, запросов, форм	**Сервис** | **Орфография** (Tools | Spelling)
Вырезать (Cut)	Удаление выделенного объекта в Буфер обмена Windows	**Правка** | **Вырезать** (Edit | Cut)
Копировать (Copy)	Копирование выделенного объекта в Буфер обмена	**Правка** | **Копировать** (Edit | Copy)
Вставить (Paste)	Вставка объекта из Буфера обмена	**Правка** | **Вставить** (Edit | Paste)
Отменить (Undo)	Отмена последних действий	**Правка** | **Отменить** (Edit | Undo)
Связи с Office (Office Links)	Передача данных из выделенной таблицы или запроса в Word или Excel	**Сервис** | **Связи с Office** (Tools | Office Links)
Анализ (Analyze)	Несколько утилит для анализа базы данных	**Сервис** | **Анализ** (Tools | Analyze
Программа (Code)	Быстрое открытие редактора VBA для просмотра кода модуля	**Вид** | **Программа** (View | Code)
Редактор сценариев (Microsoft Script Editor)	Открытие окна редактора сценариев для просмотра и редактирования HTML-кода страницы доступа к данным	**Сервис** | **Макрос** | **Редактор сценариев** (Tools | Macro | Microsoft Script Editor)
Свойства (Properties)	Отображение свойств выделенного объекта	**Вид** | **Свойства** (View | Properties)
Схема данных (Relationships)	Отображение структуры базы данных	**Сервис** | **Схема данных** (Tools | Relationships)
Новый объект (New Object)	Создание нового объекта базы данных. Чтобы выбрать тип создаваемого объекта, нажмите стрелку справа от кнопки	**Вставка** (Insert)
Справка: Microsoft Office Access (Microsoft Office Access Help)	Вызов помощника и получение справки по Access 2003	**Справка** | **Справка: Microsoft Office Access** (Help | Microsoft Office Access Help)

Окно базы данных

Особым окном в Access является *окно базы данных*, которое позволяет получить доступ ко всем объектам базы данных и выбрать режим работы с объектом. В левой части окна находится панель объектов, которая содержит ярлыки для каждого типа *объектов* Access: **Таблицы** (Tables), **Запросы** (Queries), **Формы** (Forms), **Отчеты** (Reports), **Страницы** (Pages), **Макросы** (Macros), **Модули** (Modules) (рис. 1.21).

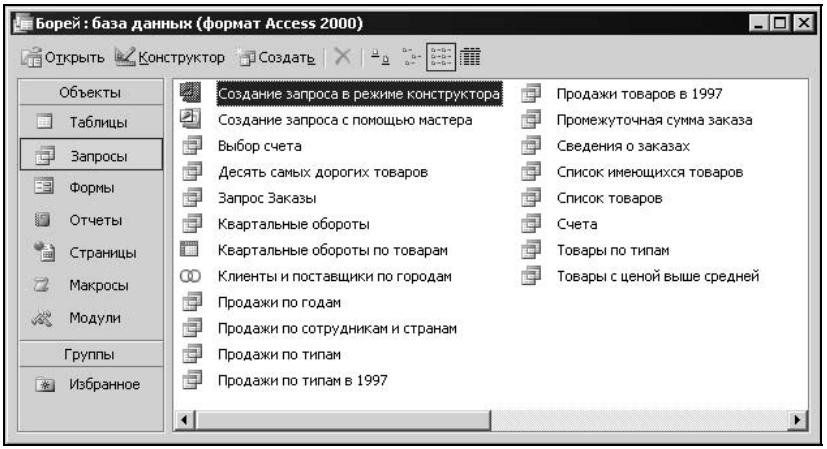


Рис. 1.21. Окно базы данных Access 2003

Щелкнув по ярлыку левой кнопкой мыши, в правой части окна вы откроете список соответствующих объектов. Список объектов может быть представлен в разных видах (обычных для папок операционной системы Windows):

- мелкие значки;
- крупные значки;
- список (рис. 1.21);
- таблица.

Переключение этих режимов отображения выполняется с помощью четырех правых кнопок на панели инструментов, расположенной вдоль верхнего края окна базы данных.

При представлении объектов в виде значков эти значки можно перетаскивать с помощью мыши и располагать внутри окна базы данных любым удобным образом. Если же вы хотите расположить значки так, чтобы они находились рядом друг с другом, нужно:

1. Щелкнуть правой кнопкой мыши на любом свободном участке окна базы данных.
2. Из контекстного меню выбрать **Выстроить значки** (Line Up Icons).

Представление перечня объектов в виде списка не позволяет располагать значки произвольным образом в окне базы данных, но их можно "перетаскивать" за преде-

лы окна базы данных (это один из способов активизировать объект, например, открыть таблицу). Представление в виде таблицы позволяет для каждого объекта посмотреть не только его имя, но и описание (столбец **Описание** (Description)), дату и время последнего изменения (столбец **Дата изменения** (Modified)), дату и время создания (столбец **Дата создания** (Created)), а также тип объекта.

При использовании любого вида представления объектов в окне базы данных их можно упорядочить по имени, типу, дате создания и дате изменения. Для этого необходимо:

1. Щелкнуть правой кнопкой мыши на любом свободном участке окна базы данных.
2. Из контекстного меню выбрать команду **Упорядочить значки** (Arrange Icons).
3. Из раскрывающегося меню выбрать способ упорядочения: **по имени** (By Name), **по типу** (By Type), **по дате создания** (By Created), **по дате изменения** (By Modified).

Чтобы объекты в окне базы данных упорядочились автоматически, необходимо:

1. Щелкнуть правой кнопкой мыши на любом свободном участке окна базы данных.
2. Из контекстного меню выбрать команду **Упорядочить значки** (Arrange Icons).
3. В раскрывающемся меню выбрать команду **автоматически** (Auto Arrange). Перед командой появится метка, свидетельствующая о том, что режим автоматического упорядочения включен.

Чтобы отменить автоматическое упорядочение, необходимо снова выбрать команду **автоматически** (Auto Arrange) в меню **Упорядочить значки** (Arrange Icons). Метка команды будет удалена, а режим автоматического упорядочения выключен.

С каждым объектом базы данных можно работать в двух режимах. Первый режим назовем *режимом выполнения* (условно, т. к. он немного различается для разных типов объектов):

- для таблиц, запросов, форм и страниц доступа к данным этот режим означает открытие соответствующего объекта и называется, соответственно, режим Таблицы (для таблиц и запросов), режим Формы, режим Страницы;
- для отчета — это режим предварительного просмотра;
- для макроса — это действительно режим выполнения;
- для модуля этот режим отключен.

Второй режим — это *режим Конструктора*. Данный режим применяется ко всем типам объектов и предназначен для создания и изменения объектов.

Выбрать нужный режим можно с помощью кнопок, которые находятся в левой части панели инструментов окна базы данных, с помощью команд меню **Вид** (View) или с помощью кнопки **Вид** (View) на панели инструментов Access.

На панели инструментов окна **База данных** (Database) слева имеются три основных кнопки:

- первая кнопка имеет изменяющееся название, в зависимости от того, какой объект выбран. Если выбраны таблица, запрос, форма, страница или группа **Избранное** (Favorites), кнопка приобретает вид **Открыть** (Open). Если выбран отчет — **Просмотреть** (Preview), если макрос или модуль — **Запустить** (Run). При этом

значок рядом с названием соответственно изменяется. Название этой кнопки в каждом варианте наглядно отражает назначение;

- вторая кнопка **Конструктор** (Design) постоянна для всех объектов и предназначена для редактирования ранее созданного объекта;
- третья кнопка **Создать** (New) имеет постоянное название, но при этом изменяются значки на ней, в соответствии с типом объекта.

Замечание

Для создания новых объектов можно использовать также специальные ярлыки, включенные в список объектов каждого типа.

Например, открыть таблицу в режиме Таблицы можно следующими способами:

- выделить ее в списке и нажать кнопку **Открыть** (Open);
- перетащить значок таблицы мышью за пределы окна базы данных (но, естественно, в пределах окна приложения);
- дважды щелкнуть кнопкой мыши по имени таблицы;
- один раз щелкнуть левой кнопкой мыши по имени таблицы, если установлен флажок **Одним щелчком** (Single-click ope) в диалоговом окне **Параметры** (Options).

Чтобы открыть любой объект в режиме Конструктора, необходимо выделить этот объект в списке и нажать кнопку **Конструктор** (Design).

Чтобы удалить объект из списка, нужно:

1. Выделить объект в списке и нажать кнопку **Удалить** (Delete) в верхней части окна базы данных или клавишу <Delete> на клавиатуре, либо щелкнуть правой кнопкой мыши по имени объекта и из контекстного меню выбрать команду **Удалить** (Delete), или выделить объект и выполнить команду меню **Правка | Удалить** (Edit | Delete).
2. При удалении объекта всегда появляется диалоговое окно, требующее подтверждение выполнения операции, т. к. удаленный объект восстановить невозможно. В этом окне необходимо нажать кнопку **Да** (Yes).

Чтобы переименовать объект, нужно:

1. Щелкнуть два раза на имени объекта (именно два раза, с некоторой задержкой, а не двойной щелчок) или щелкнуть правой кнопкой мыши на имени объекта и из контекстного меню выбрать команду **Переименовать** (Rename). Наименование объекта должно подсветиться и быть выделено рамкой (рис. 1.22).
2. В поле имени объекта ввести новое имя.

Нажать клавишу <Enter> или щелкнуть левой кнопкой мыши на любом свободном месте в правой части окна.

Чтобы скопировать или переместить объект, можно воспользоваться стандартными приемами копирования/вставки. Например, чтобы скопировать таблицу в буфер обмена (Clipboard), необходимо сделать одно из действий:

- выделить требуемую таблицу в списке и нажать кнопку **Копировать** (Copy) на стандартной панели инструментов **База данных** (Database);

Замечание

Эта панель отображается на экране, когда активным является окно базы данных.

- щелкнуть правой кнопкой мыши на имени таблицы и из контекстного меню выбрать команду **Копировать** (Copy);
- выделить требуемую таблицу в списке и выполнить команду меню **Правка** | **Копировать** (Edit | Copy));
- выделить требуемую таблицу в списке и нажать комбинацию клавиш <Ctrl>+ +<C> или <Ctrl>+<Ins>.

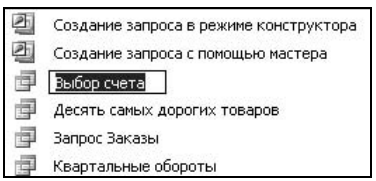


Рис. 1.22. Переименование объекта

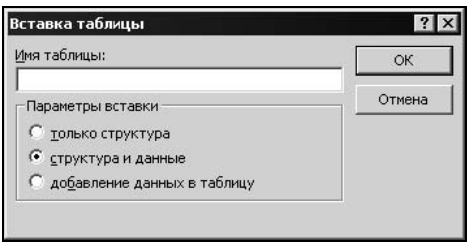


Рис. 1.23. Диалоговое окно **Вставка таблицы**

Чтобы вырезать таблицу и вставить ее в буфер обмена (Clipboard), нужно осуществить одну из операций:

- выделить необходимую таблицу в списке и нажать кнопку **Вырезать** (Cut) на панели инструментов **База данных** (Database);
- щелкнуть правой кнопкой мыши на имени таблицы и из контекстного меню выбрать команду **Вырезать** (Cut);
- выделить необходимую таблицу в списке и выполнить команду меню **Правка** | **Вырезать** (Edit | Cut);
- выделить требуемую таблицу в списке и нажать комбинацию клавиш <Ctrl>+<X>.

Чтобы вставить таблицу из буфера обмена, нужно:

1. Открыть список таблиц в окне базы данных.
2. Нажать кнопку **Вставить** (Paste) на панели инструментов **База данных** (Database) или щелкнуть правой кнопкой мыши на свободном поле в списке таблиц и из контекстного меню выбрать команду **Вставить** (Paste), либо выполнить команду меню **Правка** | **Вставить** (Edit | Paste) или нажать комбинацию клавиш <Ctrl>+<V> или <Shift>+<Ins>.
3. В появившемся окне **Вставка таблицы** (Paste Table As) (рис. 1.23) в поле **Имя таблицы** (Table Name) ввести имя новой таблицы, в группе **Параметры вставки** (Paste Option) выбрать вариант вставки таблицы: **только структура** (Structure only), **структура и данные** (Structure and Data), **добавление данных в таблицу** (Append Data to Existing Table) и нажать кнопку **OK**.

Режим вставки **только структура** (Structure only) предназначен для копирования структуры таблиц, т. е. списка полей таблицы с указанием типа каждого поля и его

описанием. При этом не предусматривается копирование данных. В режиме **структура и данные** (Structure and Data) таблица копируется целиком вместе с данными, которые в ней хранятся. В режиме **добавление данных в таблицу** (Append Data to Existing Table) необходимо в поле **Имя таблицы** (Table Name) ввести имя существующей таблицы, после чего будет осуществлено добавление данных из таблицы, находящейся в буфере обмена, в указанную таблицу.

Замечание

При добавлении данных в существующую таблицу структуры таблицы-источника и таблицы-приемника должны совпадать.

Аналогично копируются другие объекты, но при вставке объекта из буфера обмена запрашивается только имя нового объекта.

Чтобы посмотреть свойства объекта, необходимо проделать одну из следующих операций:

- щелкнуть правой кнопкой мыши на имени объекта и из контекстного меню выбрать команду **Свойства** (Properties);
- выделить объект из списка в окне базы данных и выбрать команду **Вид | Свойства** (View | Properties) из главного меню Access.

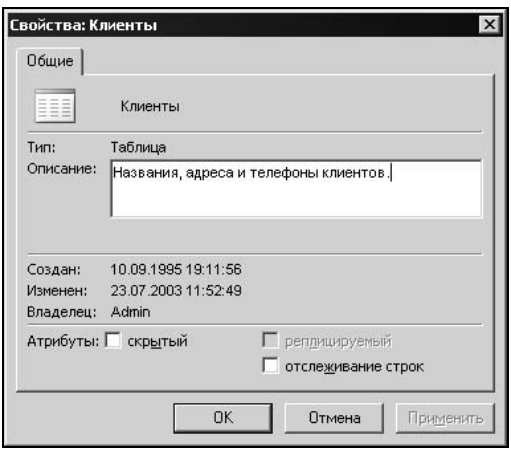


Рис. 1.24. Окно свойств таблицы

На рис. 1.24 показано окно свойств таблицы. В нем отражается следующая информация:

- **Тип** (Type) — тип объекта (в данном случае **Таблица** (Table));
- **Описание** (Description) — описание таблицы, определяемое пользователем;
- **Создан** (Created) — дата создания таблицы;
- **Изменен** (Modified) — дата последнего изменения таблицы;
- **Владелец** (Owner) — владелец (создатель) таблицы;

- **Атрибуты** (Attributes): **скрытый** (Hidden) — позволяет скрыть таблицу из окна базы данных, **реплицируемый** (Replicated) и **отслеживание строк** (Row Level Tracking) — позволяют управлять реплицируемостью объекта *(см. разд. "Репликация баз данных" гл. 18)*.

Пользователь может изменять в окне свойств только описание таблицы и значения ее атрибутов.

На панели объектов можно размещать также папки, которые содержат ярлыки для различных объектов баз данных. Таким образом, можно объединять разные типы объектов в группы. По умолчанию в этой части панели объектов размещается одна папка — Избранное (Favorites). Щелкнув мышью по папке, можно увидеть список объектов, входящих в данную группу (рис. 1.25).



Рис. 1.25. Ярлыки объектов, входящих в группу **Поставщики**

Чтобы добавить новую папку на панель объектов, нужно:

1. Щелкнуть правой кнопкой мыши на панели объектов и выбрать из контекстного меню команду **Новая группа** (New Group) или выполнить команду меню **Правка | Группы | Новая группа** (Edit | Groups | New Group).
2. В диалоговом окне **Новая группа** (New Group) ввести имя создаваемой папки и нажать кнопку **OK**.

Самый простой способ добавления ярлыка объекта в группу следующий: необходимо раскрыть список объектов данного типа, найти в нем нужный объект и перетащить его мышью в соответствующую папку на панели объектов.

Другой способ добавления объекта в группу:

1. Откройте список объектов нужного типа в окне базы данных.
2. Щелкните правой кнопкой мыши по нужному объекту и из контекстного меню выберите команду **Добавить в группу** (Add to Group), или выделите нужный

объект в списке и выполните команду меню **Правка | Добавить в группу** (Edit | Add to Group)).

3. Выберите в раскрывшемся меню нужную папку или создайте новую папку, выбрав команду **Новая группа** (New Group).

Замечание

Обратите внимание, что в группу включаются только ярлыки объектов; сами объекты никуда не перемещаются и видны в своих списках.

Так же как и отдельные объекты базы данных, группы могут удаляться и переименовываться. Ярлыки в группе тоже могут удаляться, переименовываться, копироваться. Осуществляется это с помощью соответствующих команд контекстного меню, которое появляется, если щелкнуть правой кнопкой мыши по объекту, который нужно удалить, переименовать или скопировать.

Область задач

Еще одним важным элементом среды Access 2003 является область задач, которая отображается в правой части окна приложения Access. Эта область состоит из нескольких сменных панелей и так же как меню и панели инструментов предназначена для выбора и выполнения ряда действий в системе. Вид этой области был сильно изменен в Access 2003 по сравнению с предыдущей версией Access, поэтому она подробно описана в *гл. 7.*

Использование справочной системы

Access содержит обширную справочную систему и средства ее визуализации, готовые дать ответ на подавляющее большинство вопросов, возникающих у пользователя. При этом сведения, предоставляемые справочной системой, можно не только прочесть, но и скопировать в буфер обмена или вывести на печать.

Существует несколько способов получить ответы на свои вопросы.

- С помощью специального поля для ввода вопроса в строке главного меню Access.
- С помощью области задач **Справка Access** (Microsoft Access Help).
- Обратившись к Помощнику (Office Assistant).

Справочная система в комплекте приложений Microsoft Office 2003 также претерпела существенные изменения, самое главное из которых состоит в том, что пользователь теперь имеет доступ не только к локальной справочной информации, но и к огромному объему ресурсов на Web-узле компании Microsoft. *(См. разд. "Новая справочная система" гл. 7.)*

Работа с областью задач *Справка Access*

Если на экране уже отображается одна из областей задач, достаточно щелкнуть по ее заголовку и выбрать из раскрывающегося списка **Справка** (Help), чтобы отобра-

зить область **Справка Access** (Microsoft Access Help) (рис. 1.26). Если область задач не отображается, выполните одно из действий:

- выберите команду меню **Справка | Справка Microsoft Office Access** (Help | Microsoft Office Access Help);
- нажмите кнопку **Справка Microsoft Access** (Microsoft Office Access Help) на стандартной панели инструментов Access (той, которая отображается в данный момент на экране);
- выберите команду **Вид | Область задач** (View | Task Pane), чтобы отобразить область задач, а затем выберите **Справка Access** (Access Help).

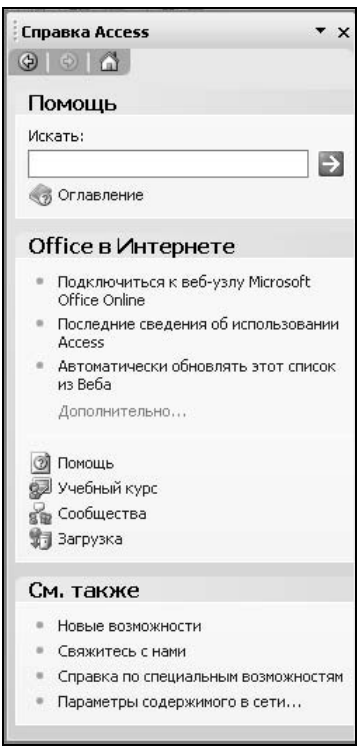


Рис. 1.26. Область задач **Справка Access**

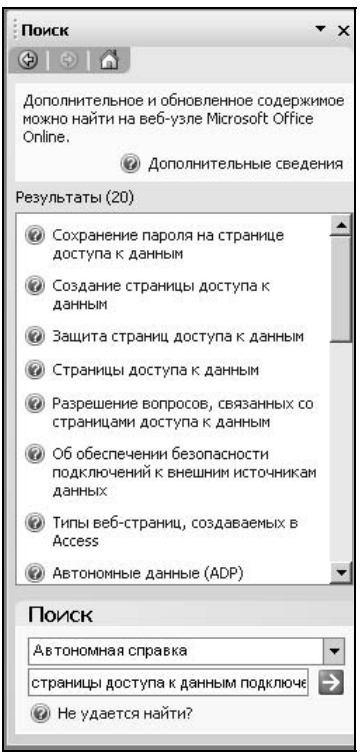


Рис. 1.27. Область задач **Поиск**

В верхней части находится поле ввода **Искать** (Search). Чтобы найти нужные сведения, введите в это поле ключевые слова для поиска, например `страницы доступа к данным подключение` и нажмите кнопку со стрелкой справа.

Если вы подключены к Интернету, то поиск будет вестись на Web-узле Microsoft Office Online, если нет, то только на локальном компьютере. Результаты поиска отобразятся в другой области задач — **Поиск** (Search Results) в виде ссылок на найденные разделы справки (рис. 1.27).

В нижней части области **Поиск** (Search Results) отображаются критерии поиска и та область, в которой был выполнен поиск, в данном примере это **Автономная справка** (Offline Help), т. е. поиск осуществлялся на локальном компьютере. Чтобы просмотреть раздел справки, щелкните мышью на нужной ссылке. Появится окно **Справка Microsoft Office Access** (Microsoft Office Access Help) с текстом выбранного раздела (рис. 1.28). Этот текст в большинстве случаев содержит ссылки на другие разделы, которые называются гиперссылками и выделяются по умолчанию синим цветом. Текст, скрытый за гиперссылкой, можно увидеть только щелкнув по ней. Причем, если гиперссылка помечена треугольником слева, текст вставляется прямо в то место, где она расположена (исходный текст раздвигается). Щелкните по гиперссылке второй раз, и соответствующий текст будет убран с экрана.

Замечание

Обратите внимание на маленькую гиперссылку **Показать все** (Show all), которая часто присутствует в правом верхнем углу окна. Щелкнув на ней мышью, вы можете раскрыть сразу все гиперссылки, присутствующие в данном разделе. А потом точно так же вы можете их все скрыть, щелкнув по гиперссылке **Скрыть все** (Hide All)

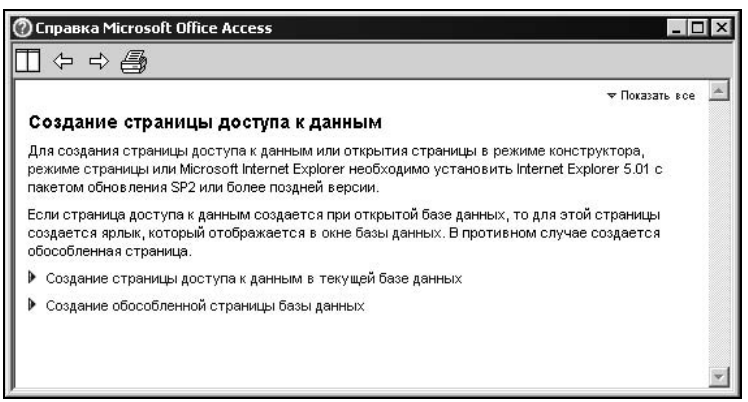


Рис. 1.28. Окно **Справка Microsoft Office Access**

Кроме гиперссылок в тексте разделов справки, вам могут встретиться различные элементы управления, предоставляющие разные возможности, например мини-учебник. При наведении на элемент управления указатель мыши меняет форму. Чтобы активизировать элемент управления, необходимо щелкнуть на нем левой кнопкой мыши.

Совет

Если вы встретите в разделе справки нечто, не похожее на текст, подведите к нему указатель мыши и, если он изменит свою форму, щелкните левой кнопкой мыши.

Если необходимые сведения не найдены, попробуйте нажать самую нижнюю ссылку **Не удается найти?** (Can't find it?). При этом будет отображен раздел справки **Советы по достижению лучших результатов поиска** (Tips for better search results). Если у вас есть доступ в Интернет, можете к нему подключиться. Как это сделать, вы узнаете,

если щелкните по ссылке **Дополнительные сведения** в верхней части области задач **Поиск**.

Если есть возможность постоянного доступа в Интернет, можно настроить Access так, чтобы всегда получать сведения с узла Microsoft Office Online. Для этого щелкните на самую нижнюю ссылку в области задач **Справка Access — Параметры содержимого в сети** (Online Content Setting) и в диалоговом окне **Параметры служб** (Service Options) установите флажок **Показать содержимое и ссылки с веб-узла Microsoft Office Online** (Show Microsoft Office Online Featured links). Это очень полезная возможность, т. к. содержимое этого узла постоянно обновляется и дополняется.

Замечание

Этот флажок отмечен по умолчанию при установке Microsoft Office, но его можно сбросить, если у вас нет постоянного доступа в Интернет. Эти установки действуют на все приложения Office, поэтому, отмечая или сбрасывая флажок, вы определяете режим поиска и для всех остальных приложений Office, которые установлены на вашем компьютере.

Чтобы посмотреть оглавление справки Microsoft Office Access, нужно щелкнуть на ссылке **Оглавление** (Table of Contents) в области задач **Справка Access**. Оглавление справки будет выведено на этой же панели. И так же, как и при поиске, если вы подключены к Интернету и установлен флажок **Показать содержимое и ссылки с веб-узла Microsoft Office Online** (Show Microsoft Office Online Featured links), оглавление ищется на Web-узле, в противном случае оно выводится из автономной справки на локальном компьютере. Оглавление, как и раньше, представляет собой иерархическую структуру тем и разделов справки. Темы обозначаются значком книги, а разделы — кружочком со знаком вопроса. Наименование темы и раздела представляет собой гиперссылку, поэтому, чтобы открыть нужную тему, достаточно щелкнуть на ее названии. Тогда могут открыться другие темы и/или разделы справки. Разделы справки находятся на нижнем уровне иерархии. Щелкнув по названию раздела, вы увидите окно **Справка Microsoft Office Access** с текстом раздела.

В окне **Справка Microsoft Office Access** сразу под заголовком находится небольшая панель инструментов с кнопками, имеющими следующее назначение.

- Кнопка **Автомозаика/Разобрать** (Auto Tile)/(Untile) позволяет изменять расположение окон **Microsoft Access** и **Справка Microsoft Office Access** на рабочем столе операционной системы. Режим **Автомозаика** (Auto Tile) означает, что рабочий стол будет разделен между двумя этими окнами. Режим **Разобрать** (Untile) означает, что окно приложения **Microsoft Access** будет иметь максимальный экранный размер (или любой другой, который был установлен вручную). Окно **Справка Microsoft Office Access** будет располагаться независимо поверх него.
- Кнопки **Назад** (Back) и **Вперед** (Forward) позволяют возвращаться к тем разделам, которые вы смотрели, и перемещаться между ними в обоих направлениях.
- Кнопка **Печать** (Print) дает возможность вывести на печать содержание текущего раздела.

Остальные ссылки в области задач **Справка Access** предназначены для работы с Интернетом. Нажав любую из них, вы попадаете на узел Microsoft Office Online на раз-

ные странички. Например, щелкнув ссылку **Учебный курс** (Training) вы попадете на главную страничку обучения, где можете выбрать продукт или курс, который хотите пройти. Нажав ссылку **Сообщества** (Communities), вы попадете на интерактивный форум, где можете обмениваться информацией с другими пользователями Office — членами сообщества. Ссылка **Помощь** (Assistance) приводит на страничку, которая содержит инструкции, тематические разделы и советы по вопросам эффективного использования приложений Office 2003.

Работа с Помощником

Кроме справочной системы все приложения Microsoft Office имеют еще одно средство — Помощник по Офису (Office Assistant). Он дает советы, причем весьма уместные, ибо советы начинающим чередуются с напоминаниями, полезными и тем, кто давно знаком с Access. Помощник помогает быстро разыскать нужную справочную информацию, часто пытается заранее предугадать, какая помощь может понадобиться в данный момент. Во многих случаях он берет на себя диалог с пользователем. Обычно пользователи настраивают его в соответствии со своим вкусом.

Замечание

Поскольку Помощник работает во всех приложениях Microsoft Office, настройки, сделанные для Access, распространятся и на все другие приложения.

Чтобы активизировать Помощника, нужно выполнить команду меню **Справка | Показать Помощника** (Help | Show Assistant). Работать с Помощником очень просто. Если вы хотите к нему обратиться, щелкните на нем левой кнопкой мыши. Появится диалоговое окно, в котором он спросит, что вы хотите сделать (рис. 1.29). Введите в поле этого окна свой вопрос, слово или словосочетание и нажмите кнопку **Найти** (Search). Результат поиска будет выведен в области задач **Поиск** (Search Results), после чего можно выбрать и просмотреть нужный раздел справки.

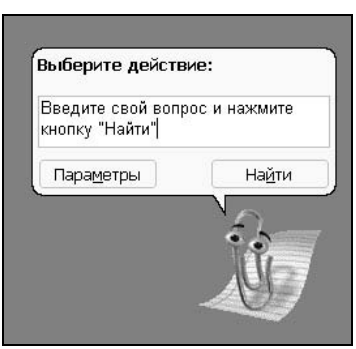


Рис. 1.29. Окно вопросов-ответов Помощника

Иногда Помощник старается привлечь внимание пользователя необычным поведением (например, над ним может зажечься лампочка). Это означает, что он готов поделиться полезным советом, уместным в данной ситуации. Если вы хотите принять его помощь, щелкните по лампочке, чтобы прочитать совет.

Если вам кажется, что Помощник больше не нужен, удалите его с экрана. Для этого щелкните на нем правой кнопкой мыши и выберите команду **Скрыть** (Hide).

Как и другие средства Microsoft Office, поведение Помощника можно настраивать. Для этого выберите из контекстного меню команду **Параметры** (Options). Появится диалоговое окно **Помощник** (Office Assistant) (рис. 1.30), в котором автоматически будет открыта вкладка **Параметры** (Options).

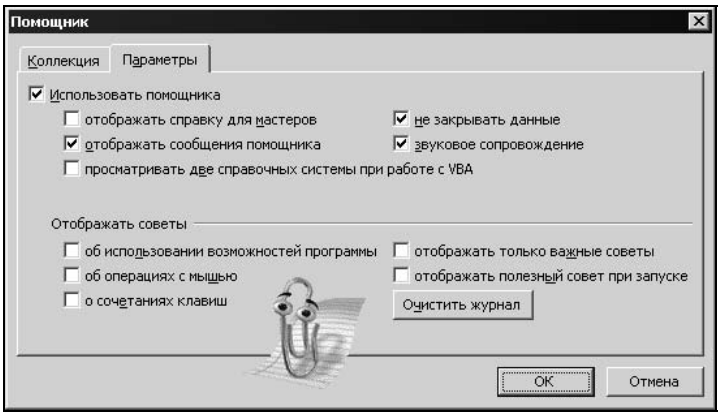


Рис. 1.30. Вкладка **Параметры** диалогового окна **Помощник**

Если вы совсем хотите отказаться от услуг Помощника, сбросьте флажок **Использовать помощника** (Use the Office Assistant). С помощью других флажков можно отрегулировать, какие именно справки и советы будет выдавать Помощник. Назначение этих флажков представлено в табл. 1.2.

Таблица 1.2. *Флажки в диалоговом окне* ***Помощник***

Название флажка	Назначение
Использовать помощника (Use the Office Assistant)	Установить, если нужно, чтобы при нажатой клавише <F1> отображался Помощник, а не диалоговое окно справочной системы
отображать справку для мастеров (Help with Wizards)	Установить, если нужно, чтобы при обращении к мастерам Помощник предоставлял информацию по работе с ними. Однако справка, которую показывает Помощник, имеется не у всех мастеров
отображать сообщения помощника (Display Alerts)	Установить, если нужно, чтобы системные сообщения и сообщения приложений отображались в выноске Помощника, если он активизирован, а не в диалоговом окне
просматривать две справочных системы при работе с VBA (Search for Both Product and Programming Help When Programming)	Установить, если нужно, чтобы при работе с Visual Basic для приложений (VBA) справки выдавались как по программированию, так и по продукту. Если нужны справки только по программированию, флажок необходимо сбросить. *(Подробную информацию о VBA см. в гл. 13.)*

Таблица 1.2 *(окончание)*

Название флажка	Назначение
не закрывать данные (Move When in the Way)	Установить, чтобы Помощник автоматически перемещался, если он перекрывает диалоговые окна или другие элементы экрана. Если Помощник не используется в течение пяти минут, он уменьшается в размерах
звуковое сопровождение (Make Sounds)	Включить звуки, издаваемые Помощником
об использовании возможностей программы (Using Features More Effectively)	Установить, чтобы отображались сведения о средствах, возможно, неизвестных пользователю, и советы об эффективном использовании уже известных ему средств
об операциях с мышью (Using the Mouse More Effectively)	Установить, чтобы отображались советы по более эффективному использованию мыши
о сочетаниях клавиш (Keyboard Shortcuts)	Установить, чтобы отображались комбинации клавиш для используемых средств
отображать только важные советы (Only Show High Priority Tips)	Установить, чтобы отображались только важные советы: например, о средствах, сберегающих время
отображать полезный совет при запуске (Show the Tip of the Day at Startup)	Установить, чтобы при запуске приложения всегда отображался один из советов Помощника

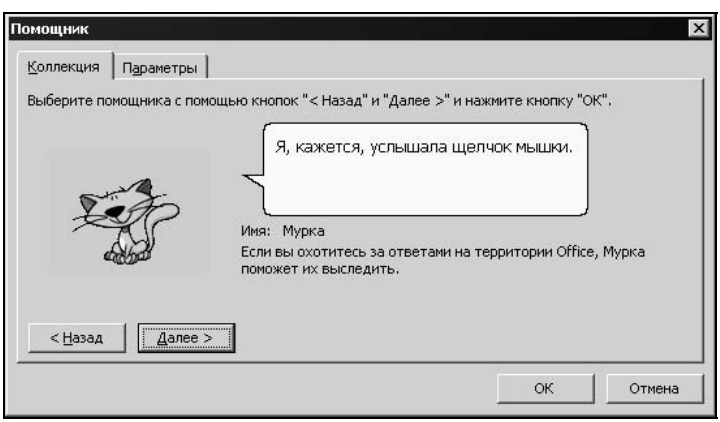


Рис. 1.31. Вкладка **Коллекция** диалогового окна **Помощник**

Кнопка **Очистить журнал** (Reset My Tips) позволяет очистить журнал Помощника, содержащий полученные ранее советы.

Помощник Microsoft Office имеет целую коллекцию разных персонажей (по умолчанию отображается известный Скрепыш). Чтобы сменить персонаж Помощника,

выберите из контекстного меню команду **Выбрать помощника** (Choose Assistant). Откроется диалоговое окно **Помощник** (Office Assistant) с автоматически раскрытой вкладкой **Коллекция** (Gallery) (рис. 1.31).

Используя кнопки **Далее** (Next) и **Назад** (Back), познакомьтесь со всеми персонажами и выберите того, кто вам понравится.

Замечание

С анимацией персонажей можно познакомиться, выбирая команду **Мотор!** (Animate!) из контекстного меню Помощника.

Другие средства получения справки

Наиболее часто используемым и естественным способом получения помощи является нажатие клавиши <F1>. Но в Access 2003 ее удобно использовать при работе в редакторе Visual Basic для приложений. При работе в среде Access нажатие клавиши <F1> в большинстве случаев приводит либо к отображению диалогового окна справки с надписью **Hidden Language Element** (Скрытый элемент языка), либо в окне справки отображается раздел справки на английском языке, либо отображается область задач **Справка Access**.

Поэтому в среде Access удобнее использовать поле **Введите вопрос**, которое отображается в правой части строки главного меню Access. В это поле можно ввести слово, словосочетание или предложение и нажать клавишу <Enter>. Результат поиска ответа на введенный вопрос отображается в области задач **Поиск** (Search Results). Выбор нужного действия раскрывает соответствующий раздел справки Microsoft Access.

Если ваш компьютер подключен к Интернету, то можно воспользоваться командами меню **Справка | Office в Интернете** (Help | Office on the Web) и **Справка | Ресурсы разработчиков Access** (Access Developer Resources)).

ГЛАВА 2



Работа с таблицами

Предметом рассмотрения данной главы являются таблицы — основной объект Access. Список таблиц, составляющих базу данных приложения, появляется в окне базы данных при первом открытии приложения. Кроме этого, Access создает системные таблицы, в которых хранится информация обо всех объектах приложения, и эти таблицы при необходимости также могут отображаться в окне базы данных.

В данной главе мы остановимся на двух важнейших аспектах работы с таблицами:

- описании структуры таблиц и установке связи между ними;
- работе с данными в таблицах: вводе, просмотре, изменении, поиске и т. д.

При описании структуры таблиц мы будем опираться на те сведения из теории баз данных, которые были приведены в начале предыдущей главы, и по мере необходимости дополнять их. Этого будет достаточно для понимания приводимого материала.

В настоящей главе будут рассмотрены:

- различные способы создания таблиц, которые предоставляет Access;
- определение полей в таблице, свойства полей, а также свойства самих таблиц;
- способы, с помощью которых можно задать условия проверки корректности ввода данных на уровне поля и на уровне записи;
- определение ключевых полей, установка связей между таблицами, работа со схемой данных;
- создание и использование индексов;
- способы добавления, обновления, замены и удаления записей при работе с таблицами;
- способы сортировки данных в таблице;
- возможности поиска записей по значению поля, а также замены вхождений образца в поле;
- способы отбора данных при помощи фильтров, копирования, экспорта и рассылки отсортированных и отфильтрованных данных.

Основные режимы работы с таблицами

В Access 2003 существует четыре режима работы с таблицами: режим Таблицы (Datasheet View), режим Конструктора (Design View), режим Сводной таблицы (PivotTable View) и режим Сводной диаграммы (PivotChart View). В режиме Таблицы осуществляется работа с данными, находящимися в таблице: просмотр, изменение, добавление, сортировка и т. п. В режиме Конструктора создается или модифицируется структура таблицы, т. е. задаются имена полей таблицы и их типы, определяются их свойства. В режимах Сводной таблицы и Сводной диаграммы удобно выполнять анализ данных, динамически изменяя способы их представления. Существует также дополнительный режим — режим предварительного просмотра, который позволяет увидеть расположение данных на листе перед осуществлением печати таблицы.

Для быстрого перехода из одного режима в другой служит кнопка **Вид** (View) на панелях инструментов **Таблица в режиме таблицы** (Table Datasheet), **Конструктор таблиц** (Table Design), **Сводная таблица** (PivotTable) и **Сводная диаграмма** (PivotChart). Чтобы переключиться в нужный режим, достаточно нажать эту кнопку или щелкнуть на стрелке на этой кнопке и выбрать из выпадающего списка нужный режим.

Открыть таблицу в режиме Таблицы можно несколькими способами:

- дважды щелкнуть мышью на имени таблицы в списке таблиц в окне базы данных;
- выделить таблицу в списке таблиц в окне базы данных и нажать кнопку **Открыть** (Open) в верхней части окна базы данных;
- щелкнуть правой кнопкой мыши на имени таблицы и из контекстного меню выбрать команду **Открыть** (Open).

В любом случае на экране появится окно с содержимым таблицы (рис. 2.1).


Клиенты : таблица

Код клиента	Название	Обращаться к	Должность
ALFKI	Alfreds Futterkiste	Maria Anders	Представитель
ANATR	Ana Trujillo Emparelados	Ana Trujillo	Совладелец
ANTON	Antonio Moreno Taqueria	Antonio Moreno	Совладелец
AROUT	Around the Horn	Thomas Hardy	Представитель
BERGS	Berglunds snabbkop	Christina Berglund	Координатор
BLAUS	Blauer See Delikatessen	Hanna Moos	Представитель
BLONP	Blondel pere et fils	Frederique Citeaux	Главный менеджер
BOLID	Bolido Comidas preparadas	Martin Sommer	Совладелец
BONAP	Bon app'	Laurence Lebihan	Совладелец
BOTTM	Bottom-Dollar Markets	Elizabeth Lincoln	Бухгалтер
BSBEV	B's Beverages	Victoria Ashworth	Представитель
CACTU	Cactus Comidas para llevar	Patricio Simpson	Продавец
CENTC	Centro comercial Moctezuma	Francisco Chang	Главный менеджер
CHOPS	Chop-suey Chinese	Yang Wang	Совладелец
COMMI	Comercio Mineiro	Pedro Afonso	Ученик продавца
CONSH	Consolidated Holdings	Elizabeth Brown	Представитель
DRACD	Drachenblut Delikatessen	Sven Ottlieb	Координатор
DUMON	Du monde entier	Janine Labrune	Совладелец
EASTC	Eastern Connection	Ann Devon	Продавец

Запись: 1 из 91

Рис. 2.1. Таблица Клиенты, открытая в режиме Таблицы

В верхней части таблицы располагаются имена *полей* (ячеек, находящихся в одном столбце таблицы), ниже следуют *записи* (строки таблицы), в которые вносятся данные. Одна запись всегда является *текущей*, и рядом с ней расположен *указатель текущей записи* (стрелка в поле выделения в левой части окна). В нижней части окна расположены *кнопки навигации*, позволяющие перемещать указатель текущей записи по таблице (на первую запись, на предыдущую запись, на следующую запись, на последнюю запись). Там же находятся поле номера текущей записи, кнопка создания новой записи и указатель общего количества записей в таблице. Для создания новой записи также служит последняя строка таблицы, отмеченная в поле выделения звездочкой (на рис. 2.1 она не видна).

Горизонтальная *полоса прокрутки* полей таблицы позволяет увидеть те поля таблицы, которые не поместились в окно таблицы. Аналогично вертикальная полоса прокрутки записей таблицы позволяет увидеть записи, находящиеся за пределами окна.

При открытии таблицы в верхней части окна приложения появляется стандартная панель инструментов **Таблица в режиме таблицы** (Table Datasheet) (рис. 2.2). Назначение кнопок этой панели и соответствующие им команды меню описаны в табл. 2.1.



Рис. 2.2. Панель инструментов **Таблица в режиме таблицы**

Таблица 2.1. *Панель инструментов* ***Таблица в режиме таблицы***

Кнопка	Описание	Команда меню
Вид (View)	Отображение таблицы в различных режимах. Чтобы изменить режим отображения таблицы, нажмите стрелку справа от кнопки и в появившемся списке выделите нужный элемент. Если просто нажать эту кнопку, таблица будет отображена в режиме Таблицы	**Вид** | **Режим таблицы** (View | Datasheet View)
Сохранить (Save)	Сохранение активной таблицы	**Файл** | **Сохранить** (File | Save)
Поиск файлов (File Search)	Поиск документов по содержанию	**Файл** | **Поиск файлов** (File | File Search)
Печать (Print)	Печать таблицы без открытия диалогового окна **Печать**	Нет
Предварительный просмотр (Print Preview)	Предварительный просмотр размещения таблицы на листе перед печатью	**Файл** | **Предварительный просмотр** (File | Print Preview)
Орфография (Spelling)	Проверка орфографии в текстовых полях таблицы	**Сервис** | **Орфография** (Tools | Spelling)

Таблица 2.1 *(продолжение)*

Кнопка	Описание	Команда меню
Вырезать (Cut)	Удаление выделенных объектов из таблицы в буфер обмена Windows	**Правка** | **Вырезать** (Edit | Cut)
Копировать (Copy)	Копирование выделенных объектов в буфер обмена	**Правка** | **Копировать** (Edit | Copy)
Вставить (Paste)	Вставка содержимого в буфер обмена	**Правка** | **Вставить** (Edit | Paste)
Отменить (Undo)	Отмена последних действий	**Правка** | **Отменить** (Edit | Undo)
Добавить гиперссылку (Insert Hyperlink)	Вставка поля типа гиперссылка	**Вставка** | **Гиперссылка** (Inser | Hyperlink Column)
Сортировка по возрастанию (Sort Ascending)	Сортировка записей по возрастанию значений в активном поле	**Записи** | **Сортировка** | **Сортировка по возрастанию** (Records | Sort | Sort Ascending
Сортировка по убыванию (Sort Descending)	Сортировка записей по убыванию значений в активном поле	**Записи** | **Сортировка** | **Сортировка по убыванию** (Records | Sort | Sort Descending
Фильтр по выделенному (Filter by selection)	Отбор записей, соответствующих выделенному фрагменту	**Записи** | **Фильтр** | **Фильтр по выделенному** (Records | Filter | Filter by selection)
Изменить фильтр (Filter by form)	Отображение диалогового окна для ввода критериев отбора записей	**Записи** | **Фильтр** | **Изменить фильтр** (Records | Filter | Filter by form)
Применение фильтра (Apply filter)	Отображение отфильтрованных записей	**Записи** | **Применить фильтр** (Records | Apply filter)
Найти (Find)	Поиск записи по заданным критериям поиска	**Правка** | **Найти** (Edit | Find)
Новая запись (New Record)	Добавление новой записи в таблицу	**Вставка** | **Новая запись** (Insert | New Record)
Удалить запись (Delete Record)	Удаление текущей записи из таблицы	**Правка** | **Удалить запись** (Edit | Delete Record)
Окно базы данных (Database Window)	Отображение окна базы данных	**Окно** | **1** (Window | 1)

Таблица 2.1 *(окончание)*

Кнопка	Описание	Команда меню
Новый объект (New Object)	Создание нового объекта базы данных. Чтобы выбрать тип создаваемого объекта, нажмите стрелку справа от кнопки	Нет
Справка: Microsoft Office Access (Microsoft Office Access Help)	Вызов помощника и получение справки по Access 2003	**Справка	Справка: Microsoft Office Access** (Help | Microsoft Office Access Help)

Открыть таблицу в режиме Конструктора можно следующими способами:

- выделить таблицу в списке таблиц в окне базы данных и нажать кнопку **Конструктор** (Design) в верхней части окна базы данных;
- щелкнуть правой кнопкой мыши на имени таблицы и из контекстного меню выбрать команду **Конструктор** (Design).

После выполнения указанных действий на экране появится окно таблицы в режиме Конструктора, а также панель инструментов **Конструктор таблиц** (Table Design) (рис. 2.3).

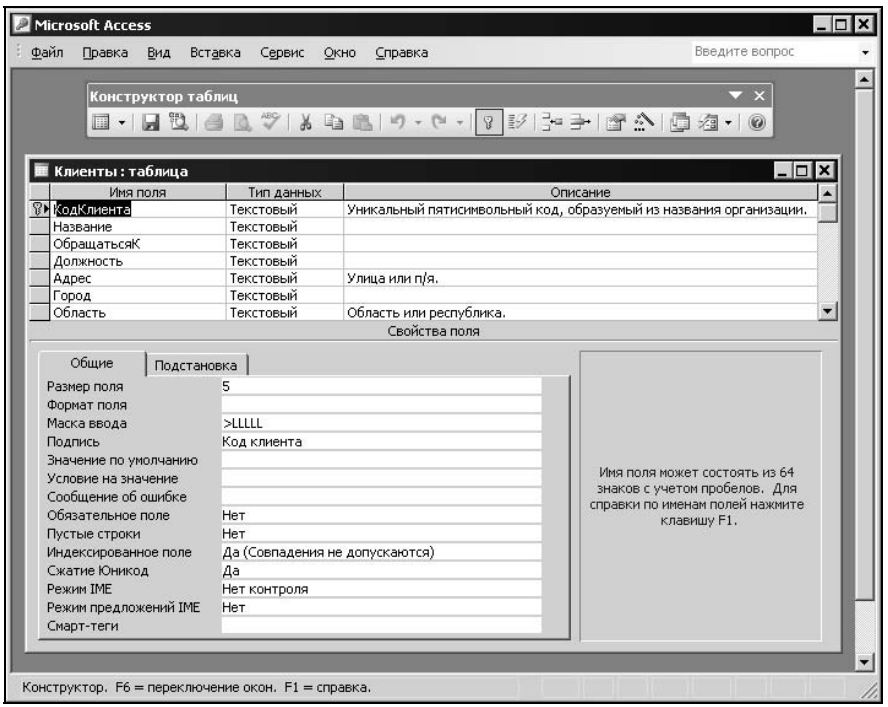


Рис. 2.3. Таблица Клиенты, открытая в режиме Конструктора

Окно таблицы в режиме Конструктора состоит из двух панелей.

- Верхняя панель содержит таблицу из трех столбцов: **Имя поля** (Field Name), **Тип данных** (Data Type) и **Описание** (Description). В столбце **Имя поля** (Field Name) содержатся имена полей таблицы, в столбце **Тип данных** (Data Type) указывается тип данных для каждого поля. Заполнение двух первых столбцов является обязательным.
- На нижней панели располагаются две вкладки, содержащие свойства полей: **Общие** (General) и **Подстановка** (Lookup), а также область контекстной справки.

Ключевое поле таблицы помечается специальным значком — ключик в поле выделения в левой части окна. Чтобы выделить поле, достаточно просто щелкнуть мышью по строке, в которой описывается данное поле. При этом в нижней части окна будут показаны параметры именно для этого поля. Перемещаться между столбцами в верхней панели окна можно с помощью клавиши <Tab>, а для перемещения по списку полей вверх и вниз пользуйтесь клавишами со стрелками. С помощью клавиши <F6> можно перемещаться между панелями.

В верхней части окна приложения появляется стандартная панель инструментов **Конструктор таблиц**. В табл. 2.2 описаны основные кнопки этой панели (только кнопки, уникальные для данного режима).

Таблица 2.2. *Панель инструментов* ***Конструктор таблиц***

Кнопка	Описание	Команда меню
Ключевое поле (Primary Key)	Добавляет в таблицу ключевое поле или удаляет его	**Правка | Ключевое поле** (Edit | Primary Key)
Индексы (Indexes)	Открывает окно редактирования индексов	**Вид | Индексы** (View | Indexes)
Вставить строки (Insert Rows)	Позволяет добавлять поля в таблицу	**Вставка | Строки** (Insert | Rows)
Удалить строки (Delete Rows)	Позволяет удалять поля из таблицы	**Правка | Удалить строки** (Edit | Delete Rows)
Свойства (Properties)	Открывает окно свойств таблицы	**Вид | Свойства** (View | Properties)
Построить (Build)	Открывает окно Построителя полей	Нет

Остальные кнопки на этой панели те же самые, что и на описанной ранее панели **Таблица в режиме таблицы** (Table Datasheet).

Режим Сводной таблицы позволяет отображать табличные данные в более удобном и обозримом виде. Сводная таблица позволяет группировать, суммировать или каким-то другим образом обрабатывать данные из обычной таблицы Access. Этот режим является, с одной стороны, аналогом сводных таблиц в Excel, а с другой — развитием уже давно используемых в Access перекрестных запросов.

Режим Сводной диаграммы тесно связан со сводной таблицей. Это просто графическое представление сводной таблицы. В предыдущих версиях Access диаграммы мог-

ли использоваться только в отчетах. Теперь они стали динамическими, и есть возможность создавать эти диаграммы и управлять ими в интерактивном режиме прямо на экране монитора.

Однако хотя можно создать сводную таблицу для обычной таблицы Access, на наш взгляд, наиболее эффективен этот режим для запросов и форм. Таблицы Access, если они правильно спроектированы, как правило, не бывают сложными. А вот запросы и формы обычно собирают информацию из разных таблиц. И поэтому именно для анализа этих данных удобно создавать сводные таблицы и диаграммы. По этой причине мы пока отложим рассмотрение настоящих режимов и вернемся к ним в *части II* книги, когда будем говорить о построении сложных запросов и форм. А в данной главе мы подробно остановимся на первых двух режимах.

Создание таблиц

В Access используются три способа создания таблиц: путем ввода данных (by entering data), с помощью Конструктора таблиц (in Design view) и с помощью Мастера создания таблиц (by using wizard). Для каждого из этих способов существует специальный ярлык новых объектов в списке таблиц (рис. 2.4). Если вы создали новый файл базы данных, то кроме этих ярлыков в списке таблиц больше ничего нет.

Замечание

Если эти ярлыки не видны, необходимо открыть окно **Параметры** (Options) (с помощью команды **Сервис | Параметры** (Tools | Option), затем раскрыть вкладку **Вид** (View) и установить флажок **новые ярлыки объектов** (New Object Shortcuts).

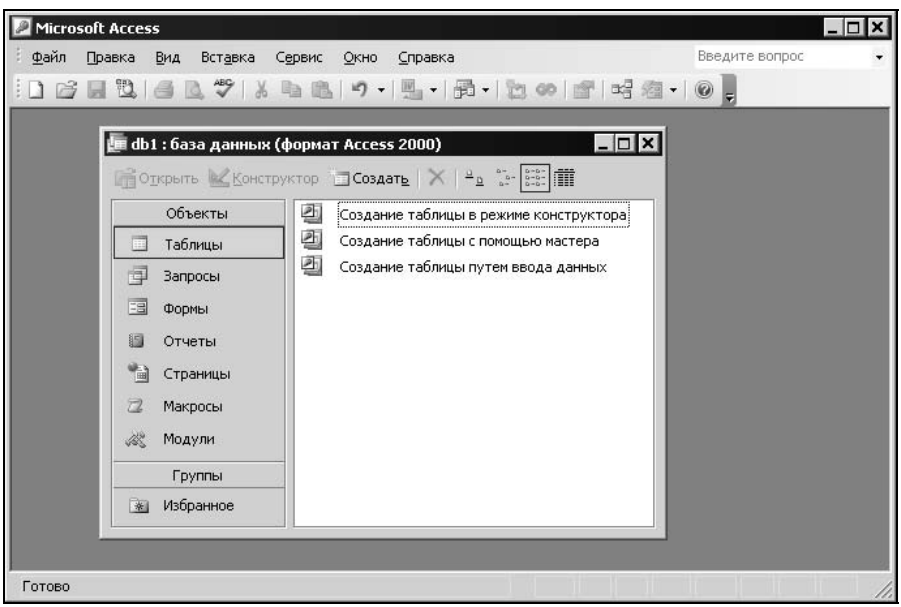


Рис. 2.4. Ярлыки новых объектов в списке таблиц пустой базы данных

Можно вместо ярлыка нажать кнопку **Создать** (New) в окне базы данных. Тогда появится диалоговое окно **Новая таблица** (New Table), в котором и нужно выбрать способ создания таблицы (рис. 2.5).

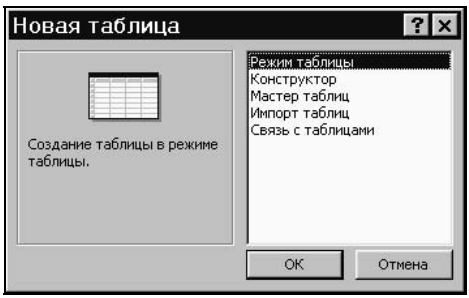


Рис. 2.5. Диалоговое окно **Новая таблица**

И наконец, на панели инструментов **База данных** (Database) расположена кнопка **Новый объект** (New Object). Если щелкнуть мышью по стрелке на этой кнопке и выбрать из списка объект **Таблица** (Table), откроется то же диалоговое окно **Новая таблица** (New Table).

Для создания новой таблицы можно воспользоваться любым из описанных способов. Обычно создание таблиц путем ввода данных используют, когда структура таблицы очень проста, например справочник. Тогда многие свойства полей устанавливаются по умолчанию. Потом можно переключить таблицу в режим Конструктора и посмотреть, как эти свойства будут установлены. Использование Мастера таблиц оправдано, если нужно создать какую-то типовую таблицу, тогда можно воспользоваться шаблонами, которые он предложит. Наиболее часто применяется режим Конструктора, поэтому даже если на первых порах вы воспользуетесь другими способами, все равно этот режим знать нужно обязательно. Он используется не только для просмотра структуры таблицы, но и для ее изменения.

Создание таблицы путем ввода данных

Если вы затрудняетесь сразу определить структуру таблицы, Microsoft Access позволяет создать таблицу путем ввода данных в окно с традиционной табличной формой (рис. 2.6).

Для этого необходимо:

1. Открыть окно новой или существующей базы данных и в нем открыть список таблиц.
2. Выделить в списке таблиц ярлык **Создание таблицы путем ввода данных** (Create table by entering data) и дважды щелкнуть по нему левой кнопкой мыши. Появится пустая таблица со стандартными названиями столбцов: **Поле1** (Field1), **Поле2** (Field2) и т. д. По умолчанию для создания таблицы предоставляется десять полей.
3. В поля этой таблицы нужно ввести требуемые данные. Тип данных в одном поле (столбце) во всех записях должен быть одинаковым.

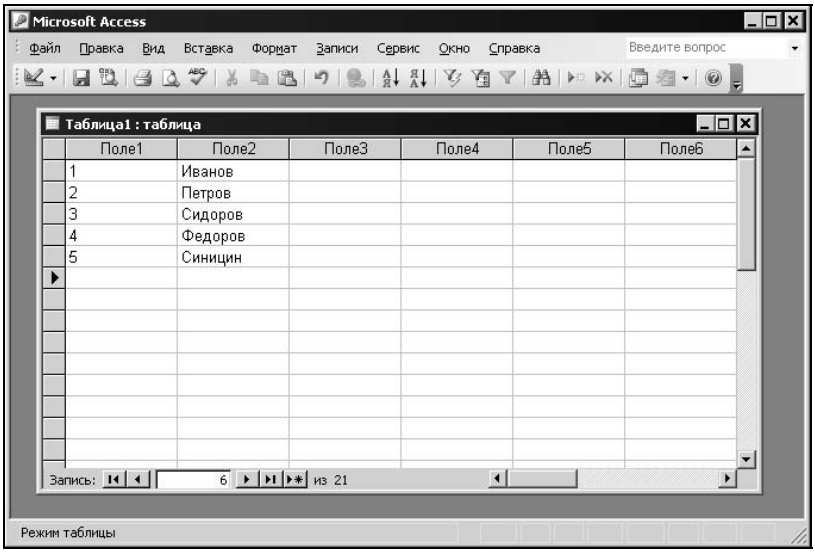


Рис. 2.6. Создание таблицы путем ввода данных

4. Можно ввести свои заголовки столбцов, для чего дважды щелкните по заголовку столбца, или выберите из контекстного меню команду **Переименовать столбец** (Rename Column). После этого стандартное название столбца подсвечивается, и вы можете вводить свое название.
5. По окончании ввода данных в таблицу нажать на кнопку **Закрыть** (в верхнем правом углу окна таблицы).
6. В ответ на вопрос **Сохранить изменения макета или структуры таблицы <*имя таблицы*>?** (Do you want to save changes to the design of table <*имя таблицы*>?) нажать кнопку **Да** (Yes).
7. В окне **Сохранение** (Save As) в поле **Имя таблицы** (Table Name) ввести имя новой таблицы и нажать кнопку **OK** (рис. 2.7).

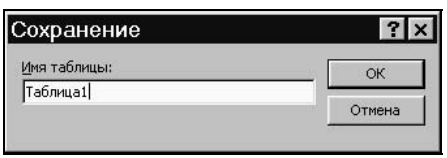


Рис. 2.7. Диалоговое окно **Сохранение**

8. Microsoft Access выдаст сообщение **Ключевые поля не заданы** (There is no primary key defined) и вопрос **Создать ключевое поле сейчас?** (Do you want to create a primary key now?). Нажать кнопку **Нет** (No), если один или несколько полей в таблице могут однозначно идентифицировать записи в таблице и служить первичным ключом, или кнопку **Да** (Yes), и тогда Access создаст дополнительное поле, которое сделает ключевым. *(К определению ключевого поля таблицы мы еще вернемся в разд. "Определение ключевых полей" данной главы.)*

В списке таблиц появится новая таблица, которая будет содержать введенные данные. Таблица при этом создается автоматически. Поля ее будут иметь либо стандартные названия, либо те, которые вы ввели, а их типы будут определяться по введенным данным. В случае если в один столбец были введены данные разных типов, например числа, даты и текст, тип поля определяется как **Текстовый** (Text).

Ввод данных в таблицу Access мало чем отличается от ввода в таблицу Excel. Для перехода между полями вы можете использовать клавишу <Tab>. Данные, введенные в текущую строку, сохраняются при переходе к следующей строке. *(Более подробно режим ввода и изменения данных в режиме таблицы описан в разд. "Ввод и проверка данных" этой главы.)*

Создание таблиц с помощью Мастера

Рассмотрим еще один способ создания таблиц, когда используется Мастер таблиц. В качестве примера построим новую таблицу `Список рассылки оповещений` в учебной базе данных `Борей` (Northwind), содержащую поля `Имя` (FirstName), `Фамилия` (LastName), `ИмяОрганизация` (OrganizationName), `АдресЭлектроннойПочты` (EmailAddress), `Заметки` (Notes).

Для этого необходимо:

1. Дважды щелкнуть левой кнопкой мыши на ярлыке **Создание таблицы с помощью мастера** (Create table by using wizard).
2. В диалоговом окне **Создание таблиц** (Table Wizard) (рис. 2.8) выбрать назначение таблицы: **Деловые** (Business) (для иных целей можно выбрать и **Личные** (Personal)).
3. В поле **Образцы таблиц** (Sample Tables) выбрать подходящую таблицу — **Список рассылки** (MailingList).
4. Переместить нужные поля из списка **Образцы полей** (Sample Fields) в список **Поля новой таблицы** (Fields in my new table). Для этого следует воспользоваться кнопками со стрелками: > — переместить одно поле, >> — переместить все поля.

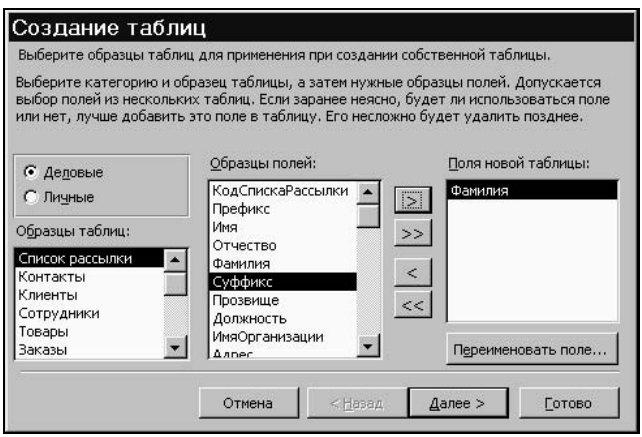


Рис. 2.8. Первое диалоговое окно Мастера таблиц

5. Повторять описанные в шагах 3—4 действия до тех пор, пока не будет создан подходящий перечень полей для новой таблицы.
6. При необходимости переименовать какое-либо поле нужно выделить его в списке **Поля новой таблицы** (Fields in my new table) и нажать кнопку **Переименовать поле** (Rename Field), затем ввести в диалоговом окне **Переименование поля** (Rename Field) новое имя поля и нажать кнопку **OK** (рис. 2.9). Нажать кнопку **Далее** (Next).

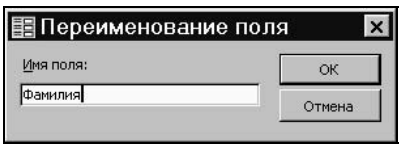


Рис. 2.9. Диалоговое окно **Переименование поля**

7. В следующем диалоговом окне в поле **Задайте имя для новой таблицы** (What do you want to name your table?) ввести имя новой таблицы `Список рассылки оповещений` (рис. 2.10).

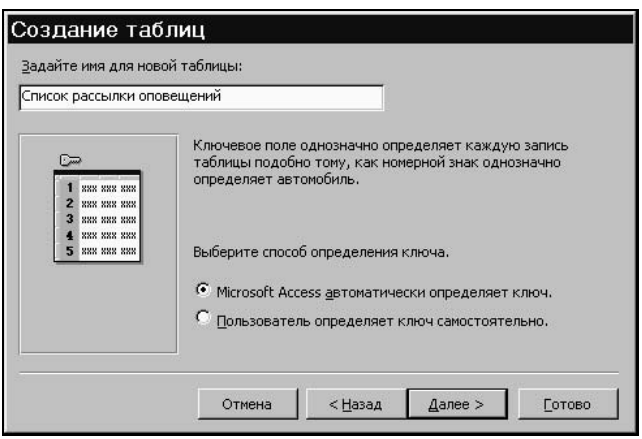


Рис. 2.10. Второе диалоговое окно Мастера таблиц

8. Выбрать способ определения ключа **Microsoft Access автоматически определяет ключ** (Yes, set a primary key for me) (имеется альтернатива **Пользователь определяет ключ самостоятельно** (No, I'll set the primary key)). Тогда в таблицу будет добавлено поле автоматической нумерации. Нажать на кнопку **Далее** (Next).
9. В случае если было выбрано самостоятельное определение ключа, в поле со списком **Выберите поле с уникальными для каждой записи данными** (What field will hold data that is unique for each record?) необходимо выбрать имя поля, которое станет ключевым. Затем нужно определить тип данных, которые будут содержаться в ключевом поле. Мастер предлагает выбрать один из трех вариантов: **Последовательные числа, автоматически присваиваемые каждой новой записи** (Consecutive Numbers Microsoft Access assigns automatically to new record); **Числа,**

вводимые пользователем при добавлении новых записей (Numbers I enter when I add new records); **Сочетание чисел и символов, вводимое пользователем при добавлении новой записи** (Numbers and/or letters I enter when I add new records). Нажать кнопку **Далее** (Next).

10. В следующем окне Мастер таблиц предлагает установить связи между существующими таблицами и создаваемой таблицей (рис. 2.11). Он пытается сам создать связи между таблицами и предоставляет информацию о создаваемых связях пользователю в виде списка. При необходимости исправить связи нужно выбрать соответствующую строку списка и нажать кнопку **Связи** (Relationship). Например, выберите в списке строку **не связана с Клиенты** и нажмите кнопку **Связи** (Relationships).

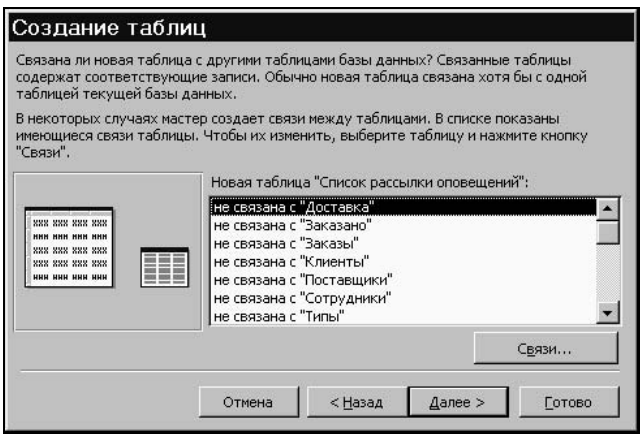


Рис. 2.11. Третье диалоговое окно Мастера таблиц

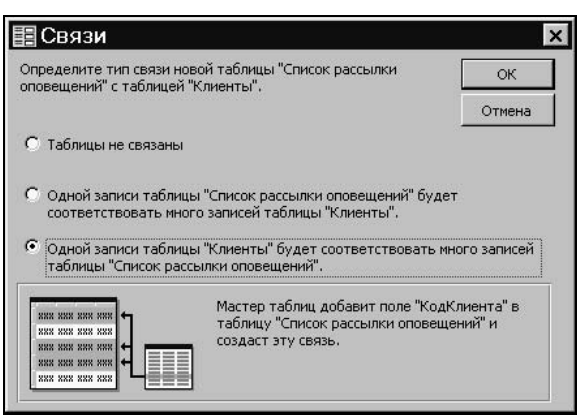


Рис. 2.12. Диалоговое окно **Связи**

11. В окне **Связи** (Relationships) нужно выбрать тип связи с данной таблицей (рис. 2.12). Выберите переключатель **Одной записи таблицы "Клиенты" будет со-**

ответствовать много записей таблицы "Список рассылки оповещений". Нажать кнопку **OK**.

12. В диалоговом окне Мастера (см. рис. 2.11) нажать кнопку **Далее** (Next).
13. Открыв последнее диалоговое окно, можно решить, что вы будете делать с таблицей дальше (рис. 2.13). Выберите переключатель **Изменить структуру таблицы**, чтобы перейти в режим Конструктора таблиц, и нажмите на кнопку **Готово** (Finish). Откроется окно Конструктора таблиц. Структура таблицы будет выглядеть так, как представлено на рис. 2.14.

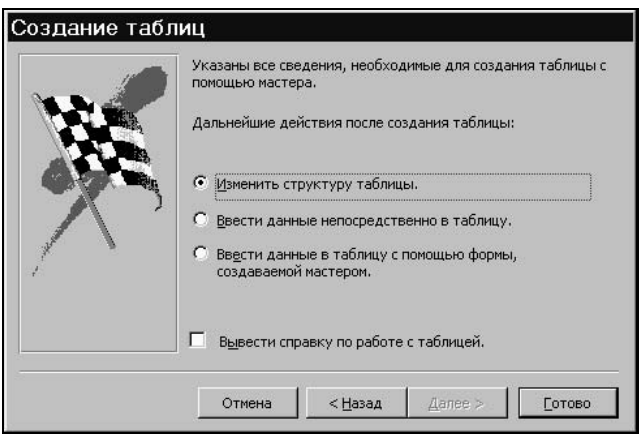


Рис. 2.13. Четвертое окно Мастера таблиц

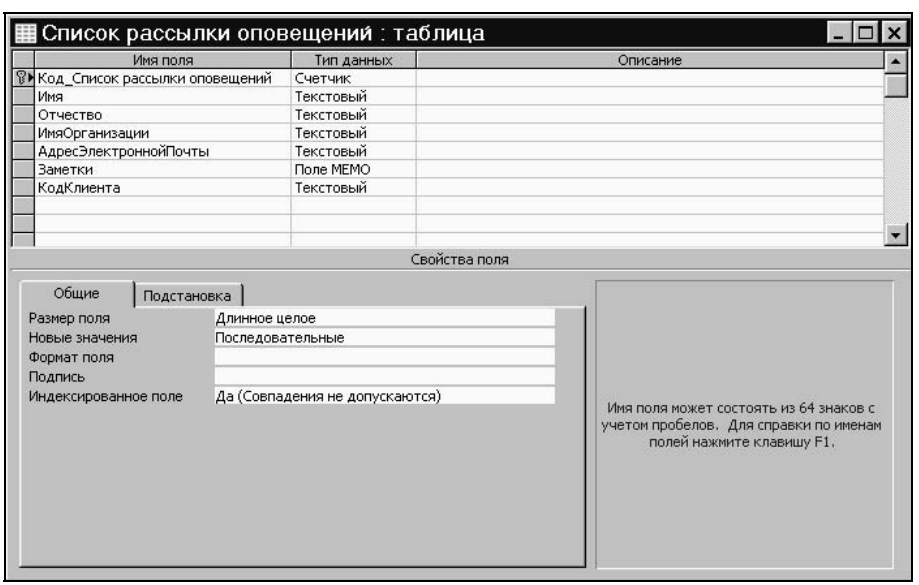


Рис. 2.14. Структура таблицы Список рассылки оповещений в режиме Конструктора

Создание таблицы с помощью Конструктора таблиц

Теперь мы познакомимся с режимом Конструктора. В режиме Конструктора таблицы создаются путем задания имен полей, их типов и свойств. Чтобы создать таблицу в режиме Конструктора, необходимо:

1. Дважды щелкнуть левой кнопкой мыши на ярлыке **Создание таблицы с помощью конструктора** (Create table in Design View) или нажать кнопку **Создать** (New) в верхней части окна базы данных, или выбрать в списке на кнопке **Новый объект** (New Object) элемент **Таблица** (Table). В первом случае сразу открывается окно Конструктора таблиц (рис. 2.15). В двух других случаях сначала появляется окно **Новая таблица** (New Table). В нем нужно выбрать элемент **Конструктор** (Design View) и нажать на кнопку **ОК**. Тогда тоже откроется окно Конструктора таблиц.

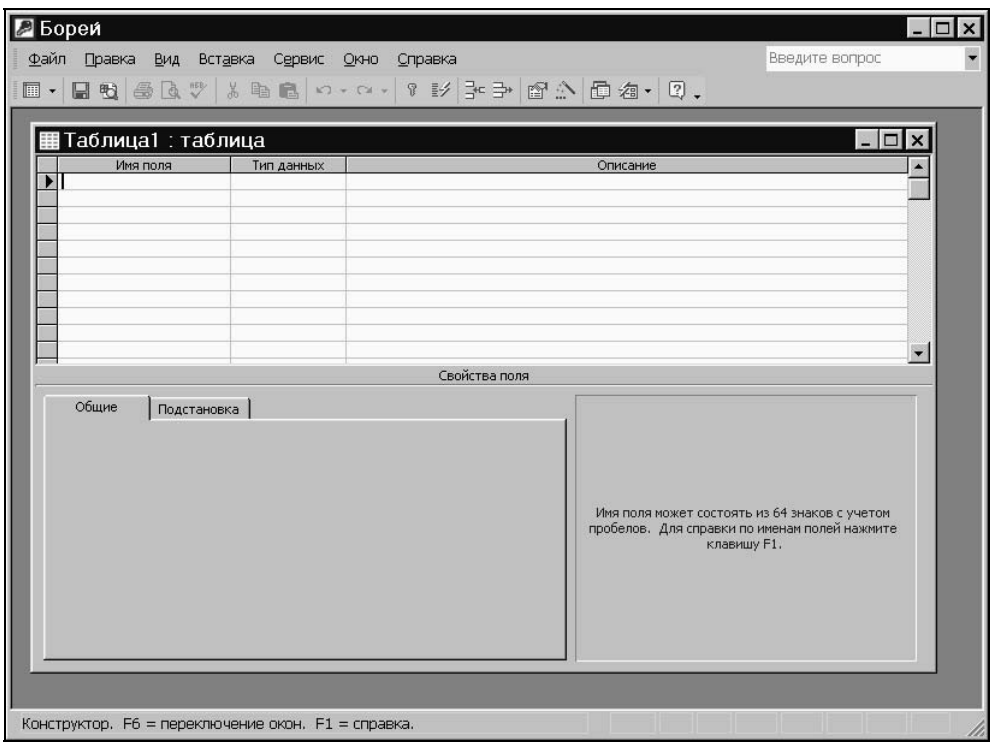


Рис. 2.15. Окно новой таблицы в режиме Конструктора

2. В окне Конструктора таблиц в столбец **Имя поля** (Field Name) ввести имена полей создаваемой таблицы.
3. В столбце **Тип данных** (Data Type) для каждого поля таблицы выбрать из раскрывающегося списка тип данных, которые будут содержаться в этом поле.
4. В столбце **Описание** (Description) можно ввести описание данного поля (не обязательно).

Совет

Хотя Access не требует, чтобы для поля вводилось описание, мы рекомендуем не экономить время и описать большинство полей в таблицах. При выборе поля в форме или таблице это описание выводится в строке состояния окна приложения. Поэтому оно может служить справкой для пользователя вашего приложения.

5. В нижней части окна Конструктора таблиц на вкладках **Общие** (General) и **Подстановка** (Lookup) ввести свойства каждого поля или оставить значения свойств, установленные по умолчанию.
6. После описания всех полей будущей таблицы нажать кнопку **Закрыть** (в верхнем правом углу окна таблицы) или выполнить команду меню **Файл | Закрыть** (File | Close).
7. На вопрос **Сохранить изменения макета или структуры таблицы <*имя таблицы*>?** (Do you want to save changes to the design of table <*имя таблицы*>?) нажать кнопку **Да** (Yes).
8. В окне **Сохранить как** (Save As) в поле **Имя таблицы** (Table Name) ввести имя создаваемой таблицы и нажать кнопку **ОК**.
9. В ответ на сообщение **Ключевые поля не заданы** (There is no primary key defined) и вопрос **Создать ключевое поле сейчас?** (Do you want to create a primary key now?) нажмите **Да** (Yes), если ключевое поле необходимо, или **Нет** (No), если такого не требуется.

После указанных действий в списке таблиц в окне базы данных появится имя и значок новой таблицы. Ввести данные в созданную таблицу можно, открыв ее в режиме Таблицы.

Существуют также и другие варианты создания таблиц. Можно, например, создать таблицу с помощью запроса на создание таблицы *(см. гл. 8)*.

Можно создавать таблицы путем импорта из других баз данных или создавая связи с такими же внешними таблицами *(см. гл. 3)*.

Определение полей

При создании таблиц в режиме Конструктора приходится задавать имена таблиц и полей в таблицах. При этом необходимо придерживаться ряда правил:

- имена полей в таблице не должны повторяться, т. е. должны быть уникальными;
- имена полей могут содержать не более 64 символов, включая пробелы;
- желательно избегать употребления имен полей, совпадающих с именами встроенных функций или свойств Access (например, `Name` — имя);
- имя поля не должно начинаться с пробела или управляющего символа (коды ASCII 00—31);
- имена полей могут содержать любые символы, включая буквы, цифры, пробелы, специальные символы, за исключением точки (.), восклицательного знака (!), апострофа (') и квадратных скобок ([), (]).

Эти же правила действительны и для имен таблиц, а также других объектов Microsoft Access.

Совет

Рекомендуется не использовать очень длинные имена, т. к. они плохо запоминаются, и не включать в имена большое количество пробелов, особенно если с именами предполагается работать с помощью процедур Visual Basic для приложений. Чтобы следовать данной рекомендации и при этом получать понятные имена, можно либо использовать вместо пробелов между словами знак подчеркивания (_), либо писать все слова слитно, но каждое начинать с заглавной буквы.

После ввода имени поля необходимо задать тип данных, которые будут находиться в этом поле. Наиболее удобным способом является выбор типа из списка, но наименование типа можно ввести и вручную. Обычно при ручном вводе Конструктор автоматически предлагает закончить вводимое название типа. В Microsoft Office Access имеются следующие типы данных:

- **Текстовый** (Text) — символьные или числовые данные, не требующие вычислений. Поле данного типа может содержать до 255 символов. Размер текстового поля задается с помощью свойства **Размер поля** (FieldSize), в котором указывается максимальное количество символов, которые могут быть введены в данное поле. При необходимости ввести строку большего размера, чем указано в данном свойстве, нужно значение свойства изменить, т. к. место под дополнительные символы не резервируется. Независимо от длины текстового поля, оно хранится в базе данных Access в записи переменной длины. Access не занимает память под неиспользуемую часть текстового поля в записи, что уменьшает общий размер базы данных.

Замечание

По умолчанию длина текстового поля равна 50 символам. Но это значение можно изменить. Для этого откройте окно **Свойства** (Properties) с помощью команды меню **Сервис | Параметры** (Tools | Options), раскройте вкладку **Таблицы и запросы** (Tables/Queries) и установите нужное значение в поле **текстовое** (Text) группы полей **Размеры полей по умолчанию** (Default field sizes).

- **Поле MEMO** (MEMO) — поле MEMO предназначено для ввода текстовой информации по объему превышающей 255 символов. Такое поле может содержать до 65 535 символов. Этот тип данных отличается от типа **Текстовый** (Text) тем, что в таблице хранятся не сами данные, а ссылки на блоки данных, хранящиеся отдельно. За счет этого ускоряется обработка таблиц (сортировка, поиск и т. п.). Поле типа MEMO не может быть ключевым.

Совет

При сортировке (и группировке тоже) значений в поле типа MEMO нужно учитывать, что Access при этом использует только первые 255 символов поля.

- **Числовой** (Number) — числовой тип применяется для хранения числовых данных, используемых в математических расчетах. Имеет много подтипов. От выбора подтипа (размера) данных числового типа зависит точность вычислений. Для установки подтипа числовых данных служит свойство **Размер поля** (FieldSize). Данные этого типа могут содержаться в 1, 2, 4, 8 или 16 байтах. Обычно по умолчанию используется подтип **Длинное целое** (LongInteger), который занимает

4 байта и представляет собой число в пределах от −2 147 483 648 до +2 147 483 647. Но, кроме этого типа, можно указать **Байт** (Byte) — 1 байт, **Целое** (Integer) — 2 байта, **Одинарное с плавающей точкой** (Single) — 4 байта, **Двойное с плавающей точкой** (Double) — 8 байт, **Десятичное** (Decimal) — 12 байт, **Код репликации** (Replication ID) — 16 байт.

Замечание

Значение подтипа, устанавливаемое по умолчанию, можно изменить так же, как и длину текстового поля — на вкладке **Таблицы и запросы** (Tables/Queries) диалогового окна **Свойства** (Properties) изменить значение в поле **числовое** (Text) группы полей **Размеры полей по умолчанию** (Default field sizes).

- **Денежный** (Currency) — тип данных, который рекомендуется использовать при выполнении денежных расчетов, т. к. он предотвращает округления в процессе вычислений. Точность денежного типа данных колеблется от 1 до 4 десятичных знаков. Целая часть может содержать до 15 десятичных знаков. Денежный тип данных занимает 8 байт памяти.
- **Дата/Время** (Date/Time) — тип для представления даты и времени. Позволяет вводить даты с 100 по 9999 год. Размер поля — 8 байт. Даты и время хранятся в специальном фиксированном числовом формате. Дата является целой частью значения поля типа **Дата/Время**, а время — его дробной частью. Access предоставляет большой выбор форматов отображения даты и времени.
- **Счетчик** (AutoNumber) — поле содержит уникальный номер, определяемый Access автоматически для каждой новой записи либо случайным образом, либо путем увеличения предыдущего значения на 1, либо путем формирования GUID (Global Unic ID, глобальный уникальный идентификатор). Этот тап данных используется для создания уникального идентификатора записей в таблице. Значения полей типа счетчика обновлять нельзя. Счетчик обычно занимает 4 байта памяти и, следовательно, максимальное число записей в таблице с полем счетчика не должно превышать двух миллиардов. Поле счетчика типа GUID занимает 16 байт и позволяет обеспечить синхронизацию различных копий базы данных при репликации, поэтому он еще называется *код репликации.*
- **Логический** (Yes/No) — логическое поле, которое может содержать только два значения, интерпретируемых как Да/Нет, Истина/Ложь, Включено/Выключено. Поля логического типа не могут быть ключевыми, но их можно индексировать.

Внимание!

Access использует величину −1 для представления значения **Истина** и величину 0 — для значения **Ложь**.

- **Поле объекта OLE** (OLE object) — содержит ссылку на OLE-объект (лист Microsoft Excel, документ Microsoft Word, звук, рисунок и т. п.). Объем объекта ограничивается имеющимся в наличии дисковым пространством. Тип объекта OLE не указывается в свойствах поля объекта OLE. Он зависит от приложения, использованного для создания объектов OLE, которые хранятся в этом поле. Упаковщик объектов позволяет внедрять файлы, созданные приложениями, которые не являются серверами (поставщиками) объектов OLE. Например, он позволяет

внедрить текстовые файлы в качестве объектов в таблицы Access 2003. В поле объекта OLE могут храниться произвольные данные, в том числе и данные нескольких типов. Это позволяет обойти основное ограничение реляционных баз данных, которое требует, чтобы в каждом поле хранились данные только одного типа.

- **Гиперссылка** (Hyperlink) — дает возможность хранить в поле ссылку, с помощью которой можно ссылаться на произвольный фрагмент данных внутри файла или Web-страницы на том же компьютере, на компьютере в локальной сети, в интранет или в Интернет. Гиперссылка состоит из четырех частей: отображаемый текст, адрес (путь к файлу или странице), дополнительный адрес (положение внутри файла или страницы) и текст всплывающей подсказки. Поле типа **Гиперссылка** не может быть ключевым или индексированным.

В поле типа можно также выбрать значение **Мастер подстановок** (Lookup Wizard), который запускает Мастера подстановок (Lookup Wizard), создающего поле подстановок. *Поле подстановок* позволяет выбирать значение поля из списка, содержащего набор постоянных значений или значений из другой таблицы. *(О поле подстановок мы подробно будем говорить позже в разд. "Использование Мастера подстановок" данной главы.)*

Как только вы выбираете тип данных для поля, так на нижней панели окна отображается список свойств этого поля. Данный список зависит от типа поля. Для того чтобы просмотреть или изменить свойства конкретного поля таблицы, необходимо:

1. Выделить нужное поле таблицы (т. е. установить курсор в строку с именем поля) в окне Конструктора.
2. Раскрыть вкладку **Общие** (General) для просмотра общих свойств поля или вкладку **Подстановка** (Lookup) для просмотра параметров подстановки.

Рассмотрим основные свойства полей для каждого типа данных.

Свойства полей таблицы

Для большинства типов данных характерно свойство **Подпись** (Caption). С помощью этого свойства можно задать названия полей таблицы, которые выводятся в различных режимах (в надписях, присоединенных к элементам управления формы, в заголовке столбца в режиме Таблицы; в строке заголовка в режиме Формы; в заголовке отчета, выводящемся в режиме предварительного просмотра; текст, который выводится в элементе управления). Поле может содержать до 2048 символов.

Кроме того, для большинства типов данных существует свойство **Обязательное поле** (Required), которое определяет необходимость ввода данных в это поле.

Совет

Настоятельно рекомендуем устанавливать значение этого свойства равным **Да** (Yes) для тех полей таблицы, которые не должны быть пустыми. Это обеспечивает автоматический контроль ввода данных в такие поля, т. е. ни при каких обстоятельствах пользователь не сможет создать запись, в которой данное поле останется пустым.

Свойство **Формат поля** (Format) указывает формат отображения данных из поля в режиме Таблицы. Для определения формата полей текстового типа используются

специальные символы форматирования. Для числовых полей значение формата можно выбрать из раскрывающегося списка. Для логических полей можно выбрать из списка следующие варианты: **Да/Нет** (Yes/No), **Истина/Ложь** (True/False), **Вкл/Выкл** (On/Off). *(Форматы полей мы будем подробно рассматривать в разд. "Форматы отображения данных" этой главы.)*

С помощью свойства **Маска ввода** (Input Mask) указывается маска, позволяющая автоматизировать проверку ввода символов в поле. Она применяется к таким полям, как номер телефона, дата и т. д. Задавать маску ввода можно вручную или с помощью Мастера. *(К маскам ввода мы еще вернемся в разд. "Форматы отображения данных" этой главы.)*

Свойство **Индексированное поле** (Indexed) определяет, является ли данное поле индексированным, и если является, то в каком режиме. Существуют два режима индексирования: **Совпадения допускаются** (Duplicates OK) и **Совпадения не допускаются** (No duplicates). В первом случае поле может содержать повторяющиеся значения, во втором — нет.

Для большинства типов полей определено свойство **Значение по умолчанию** (Default Value). В этом свойстве указывается значение, автоматически добавляемое в поле для каждой новой записи, если это значение не введено пользователем.

Внимание!

Обратите внимание, что значение по умолчанию присваивается только при вводе новой записи. Если вы установите это значение для таблицы, в которой уже существуют записи, то в старых записях значение этого поля не изменится.

Два свойства, которые тоже определены для большинства полей, позволяют выполнять проверку данных, вводимых в поле.

- **Условие на значение** (Validation Rule) — свойство определяет условие (ограничение), накладываемое на вводимые в это поле данные. При несоответствии вводимых данных указанному условию выдается сообщение об ошибке.
- **Сообщение об ошибке** (Validation Text) — свойство определяет то сообщение, которое будет выдаваться пользователю, если при вводе данных не соблюдается условие, указанное в свойстве **Условие на значение** (Validation Rule).

Замечание

Если это свойство не указано, будет выдано стандартное сообщение об ошибке.

- У числовых, текстовых полей и поля типа **Счетчик** (AutoNumber) есть свойство **Размер поля** (Field Size). Для текстового поля значение этого свойства определяет максимальное количество символов, которые можно хранить в данном поле, а для числовых полей — тип и размер чисел, которые могут в нем содержаться.

Разработчики приложений Microsoft Access 2003 могут присоединить к любым полям таблицы смарт-теги — специальные надстройки, которые могут распознавать в процессе их ввода определенные строки и сопоставлять им некоторые действия, которые пользователь может выполнить сразу же после ввода. *(Подробную информацию о смарт-тегах см. в гл. 7.)*

В табл. 2.3—2.5 приводятся свойства, специфичные для полей определенного типа.

Таблица 2.3. *Дополнительные свойства текстового поля и поля MEMO*

Наименование свойства	Описание
Пустые строки (Allow Zero Length)	Свойство определяет, допустимо ли оставлять это поле пустым — значением поля будет являться пустая строка""
Сжатие Юникод (Unicode Compression)	Свойство определяет, выполняется ли сжатие данных при сохранении для полей текстового типа (при кодировке Unicode удаляются все первые байты символов, если они равны 0)
Режим IME (IME Mode)	Input Method Editor (IME) — программа, позволяющая распознавать вводимые с помощью специальных кодов символы азиатских алфавитов: китайского, корейского и японского. Свойство позволяет указать режим преобразования, которое выполняется при получении полем фокуса
Режим предложений IME (IME Sentence Mode)	Определяет режим предложений IME, который применяется при получении полем фокуса

Таблица 2.4. *Дополнительные свойства полей числового и денежного типа*

Наименование свойства	Описание
Число десятичных знаков (Decimal Places)	Свойство определяет число десятичных знаков справа от десятичного разделителя. Значение свойства можно выбрать из раскрывающегося списка. Чтобы число знаков определялось автоматически, необходимо установить значение **Авто** (Auto)

Таблица 2.5. *Дополнительные свойства полей типа* ***Счетчик***

Наименование свойства	Описание
Новые значения (New Values)	Указывается способ формирования следующего значения счетчика. Из списка можно выбрать два варианта: **Последовательные** (Increment) и **Случайные** (Random)

Использование Мастера подстановок

Сделать более простым ввод значений в поле позволяет операция подстановки. Используя эту операцию, можно выбирать значения поля из списка. Список значений может быть как фиксированным, так и динамическим, т. е. он может выбираться из таблицы или при запросе. Сформировать столбец подстановок для поля помогает Мастер подстановок (Lookup Wizard). Давайте создадим столбец подстановок для

поля КодКлиента в таблице Список рассылки оповещений. Это даст нам возможность при вводе данных в эту таблицу вводить не коды клиентов, которые мы не знаем, а выбирать из списка название организации, в которой работает данное лицо. Чтобы использовать Мастер подстановок (Lookup Wizard), необходимо:

1. Открыть таблицу Список рассылки оповещений в режиме Конструктора.
2. Выделить поле КодКлиента и выбрать из списка в столбце **Тип данных** (Data Type) значение **Мастер подстановок** (Lookup Wizard) (рис. 2.16).

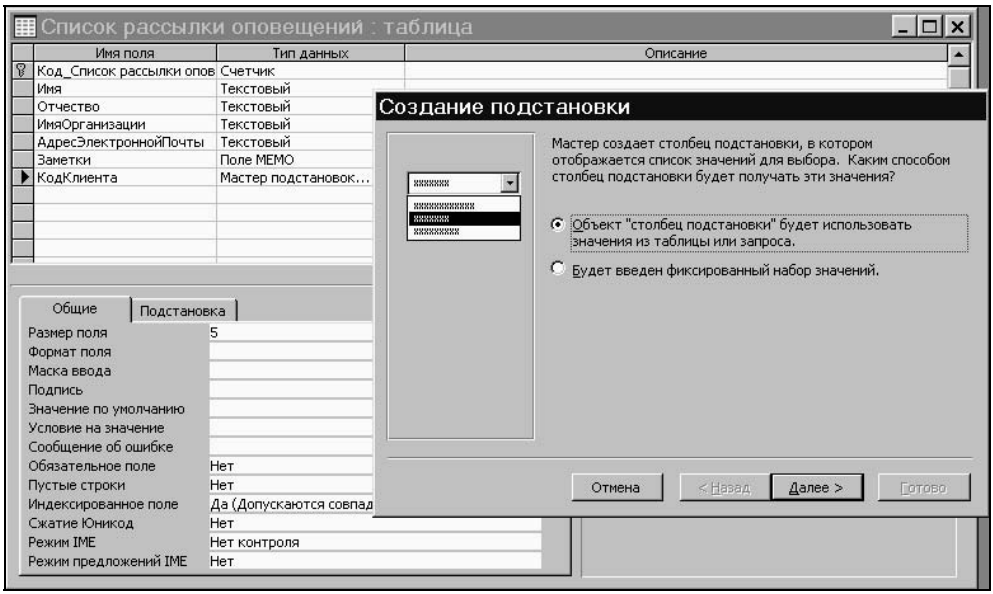


Рис. 2.16. Первое диалоговое окно Мастера подстановок

В открывшемся диалоговом окне Мастера подстановок (Lookup Wizard) выбрать способ задания значений: **Объект "столбец подстановки" будет использовать значения из таблицы или запроса** (I want the lookup column to look up the values in a table or query), т. к. в данном случае мы должны использовать данные из таблицы Клиенты (Customers). Нажать кнопку **Далее** (Next).

В следующем диалоговом окне можно выбрать из списка таблицу или запрос, из которого будет осуществляться подстановка. Выбрать таблицу Клиенты (Customers) и нажать кнопку **Далее** (Next) (рис. 2.17).

В списке **Доступные поля** (Available Fields) выводятся все поля таблицы Клиенты (Customers). Переместить из списка доступных полей в список подстановки (Selected Fields) поля КодКлиента и Название, нажать кнопку **Далее** (Next) (рис. 2.18).

Просмотреть столбец подстановки, с помощью перетаскивания границы, выбрать ширину столбца, а также оставить установленный по умолчанию флажок **Скрыть ключевой столбец** (Hide key column). Действительно, нам не нужно в раскрывающемся списке видеть коды клиентов. Нажать на кнопку **Далее** (Next) (рис. 2.19).

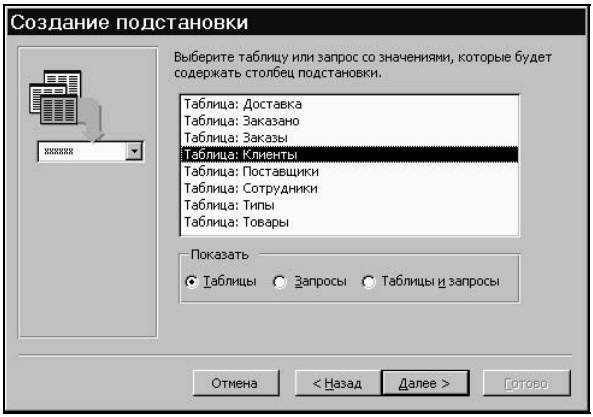


Рис. 2.17. Второе диалоговое окно Мастера подстановок

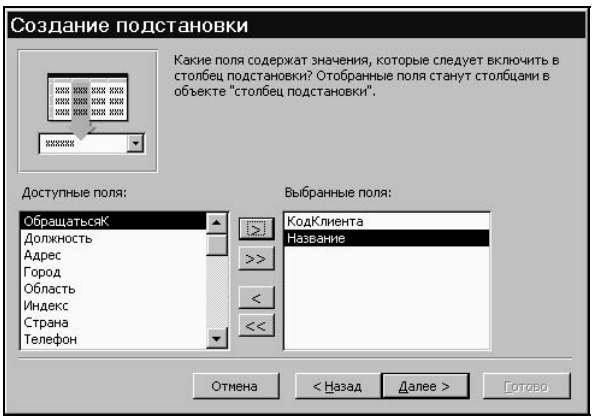


Рис. 2.18. Третье диалоговое окно Мастера подстановок

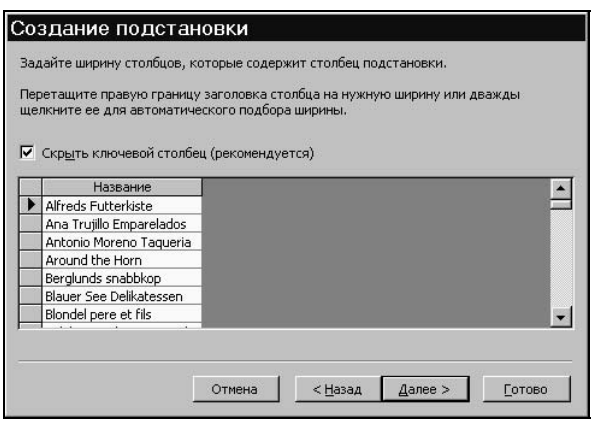


Рис. 2.19. Четвертое диалоговое окно Мастера подстановок

Ввести название столбца подстановок и нажать на кнопку **Готово** (Finish) (рис. 2.20). При необходимости Мастер подстановок попросит сохранить те изменения, которые вы внесли в таблицу, прежде чем окончательно установит подстановки — ответьте **Да** (Yes). На этом процедура создания столбца подстановок заканчивается.

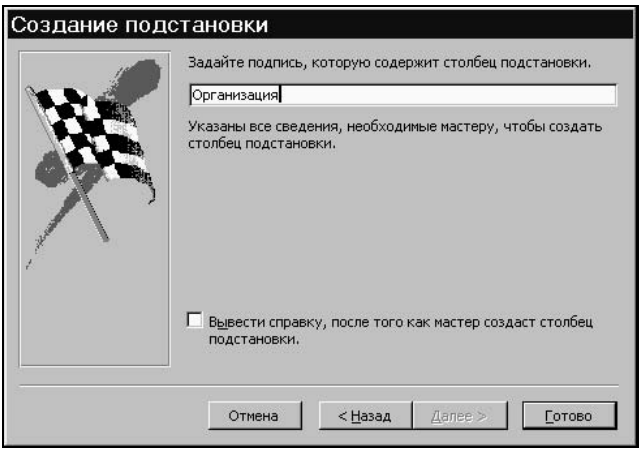


Рис. 2.20. Пятое диалоговое окно Мастера подстановок

Замечание

Применять операцию подстановки можно только к полям, содержащим текстовые и числовые данные, а также к логическим полям. Другие типы полей не могут использовать подстановку.

На нижней панели окна Конструктора таблиц находится вкладка **Подстановка** (Lookup), в которой определяются свойства поля, относящиеся к подстановке. Если столбец подстановки для поля не определен, значение свойства **Тип элемента управления** (Display Control) для текстовых и числовых полей равно **Поле** (Text Box), для логического типа — **Флажок** (Check Box). При этом остальные свойства на вкладке **Подстановка** (Lookup) отсутствуют.

Если вы хотите добавить к полю столбец подстановок, то установите для свойства **Тип элемента управления** (Display Control) значение **Поле со списком** (Combo Box) или **Список** (ListBox). В этом случае на вкладке **Подстановка** (Lookup) появится ряд свойств, позволяющих управлять параметрами подстановки (рис. 2.21). Описание этих свойств приведено в табл. 2.6.

Замечание

Для просмотра таблицы в режиме Таблицы описанные ранее поля абсолютно равноценны, различаются они в формах. Поле **Список** отображается на экране сразу в виде прокручиваемого списка значений, а **Поле со списком** — в виде раскрывающегося списка. *(О других отличиях см. разд. "Встроенные элементы управления" гл. 9.)*

Таблица 2.6. *Свойства подстановки для поля со списком*

Наименование свойства	Описание
Тип источника строк (Row Source Type) *	Свойство определяет тип источника строк для списка или поля со списком. Может принимать три значения: **Таблица или запрос** (Table/Query), **Список значений** (Value List) и **Список полей** (Field List)
Источник строк (Row Source) *	Свойство определяет источник данных для формирования строк списка. При выборе типа источника строк **Список значений** (Value List) оно представляет собой список элементов, разделяемых точкой с запятой (;), в противном случае — имя таблицы или запроса либо инструкцию языка SQL
Присоединенный столбец (Bound Column) *	Свойство определяет номер столбца списка или поля со списком, содержащего значение, которое должно храниться в соответствующем поле таблицы. Если это свойство имеет значение 0, то в поле будет сохраняться номер строки списка, если 1 и больше, то в поле передаются данные из соответствующего столбца списка. Значение этого свойства не может превышать значение свойства **Число столбцов** (Column Count)
Число столбцов (Column Count) *	Свойство определяет число столбцов, выводимых в списке или в поле со списком. Если в качестве источника строк используется **Таблица или запрос** (Table/Query), то в списке будут выводиться столбцы таблицы, начиная с первого и заканчивая тем, который указан в значении данного свойства. Если в качестве источника строк используется список значений, то в списке будет выводиться указанное число столбцов, а данные в них будут располагаться построчно в последовательности, указанной в свойстве **Источник строк** (Row Source)
Заглавия столбцов (Column Heads) *	Свойство определяет, используются ли первые элементы списка или поля со списком как заголовки столбцов
Ширина столбцов (Column Widths) *	Свойство определяет ширину столбцов списка или поля со списком. Если в списке несколько столбцов, значения разделяются точкой с запятой (;). Задав нулевое значение можно скрыть столбец
Число строк списка (List Rows)	Свойство определяет максимальное число строк для поля со списком. Если количество элементов списка превысит указанное число строк, в списке появится линейка прокрутки
Ширина списка (List Width)	Свойство определяет ширину раскрывающегося списка для поля со списком
Ограничиться списком (Limit to List)	Свойство определяет, могут ли вводиться в поле со списком значения, не являющиеся элементами списка.

Если значение свойства **Тип элемента управления** (Display Control) равно **Список** (List Box) (а это возможно только для текстовых и числовых полей), на вкладке **Подстановка** появляются лишь свойства, отмеченные в табл. 2.6 звездочкой (*).

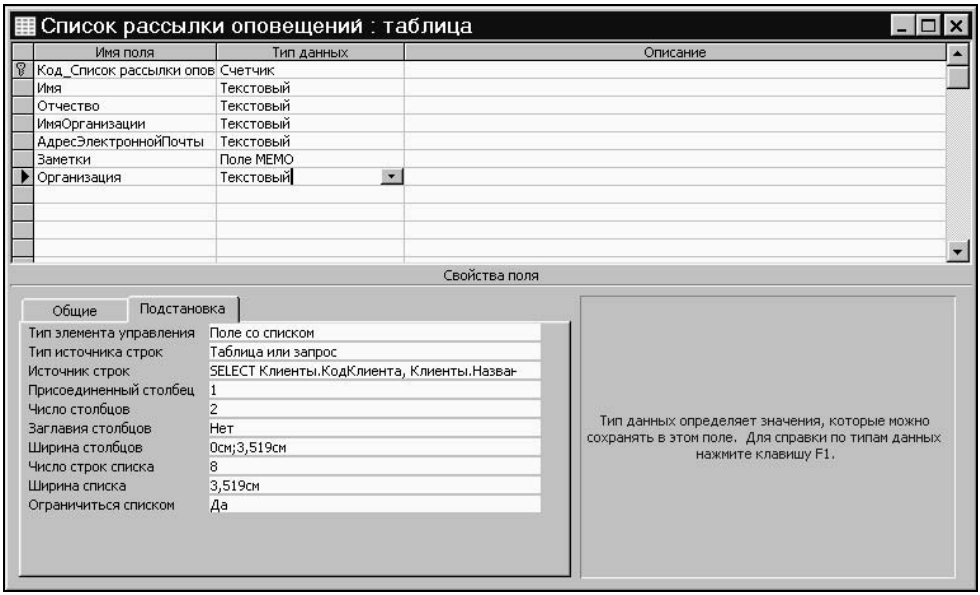


Рис. 2.21. Столбец подстановок в таблице `Список рассылки оповещений`

Форматы отображения данных

Свойство **Формат поля** (Format) на вкладке **Общие** (General) позволяет указать форматы вывода текста, чисел, дат, значений времени на экран и на печать. Для каждого типа поля существует свой набор встроенных стандартных форматов. Эти форматы используют региональные установки компьютера, т. е. те параметры, которые установлены на соответствующих вкладках окна **Язык и стандарты** (Regional and Language Options) Панели управления Windows. Но вы можете определить собственные (специальные) форматы отображения, пользуясь специальными символами форматирования.

Свойство **Формат поля** (Format) используется для отображения данных не только в режиме Таблицы, но и в связанных с данным полем элементах управления, в формах и отчетах, если для них не установлен другой формат.

Замечание

Свойство **Формат поля** (Format) определяет только способ отображения данных, не влияя на способ их сохранения.

Свойство **Формат поля** (Format) имеет различные значения для разных типов данных. В табл. 2.7—2.13 приведены стандартные значения этого свойства для допустимых типов, а также специальные символы, которые используются при определении нестандартных форматов.

По умолчанию, т. е. если значением свойства **Число десятичных знаков** является **Авто** (Auto), форматы **Фиксированный**, **С разделителями разрядов** и **Процентный** выводят числа с двумя десятичными знаками.

Таблица 2.7. *Встроенные числовые форматы для числовых и денежных типов данных*

Значение	Описание
Основной (General Number)	Используется в качестве значения по умолчанию. Числа отображаются так, как они были введены. Пример: 12345,67
Денежный (Currency)	Число выводится с разделителями разрядов и символом национальной валюты, введенным в региональных установках. Пример: 3 457р
Евро (Euro)	Частный случай предыдущего формата со знаком валюты евро. Пример: 3 457€
Фиксированный (Fixed)	Числа выводятся без разделителей групп разрядов. Пример: 3456,79
С разделителями разрядов (Standard)	Числа выводятся с разделителями групп разрядов. Пример: 3 456,79
Процентный (Percent)	Значение умножается на 100; добавляется символ процента (%). Пример: 98,00%
Экспоненциальный (Scientific)	Числа выводятся в экспоненциальном виде. Пример: 3,46E+03

Числовые форматы также можно определять самостоятельно с помощью специальных символов. Они могут включать в себя от одного до четырех разделов, отделенных друг от друга точкой с запятой (;). В первом разделе описывается формат положительных чисел, во втором — формат отрицательных чисел, в третьем — формат нулевых значений, в четвертом — формат пустых (Null) значений.

Таблица 2.8. *Символы форматирования для числовых типов данных*

Символ	Описание
. (точка)	Десятичный разделитель. Символ десятичного разделителя выбирается в окне **Язык и стандарты** (Regional and Language Options) Панели управления Windows
, (запятая)	Разделитель групп разрядов
`0`	Прототип разряда. Выводится цифра или 0
`#`	Прототип разряда. Выводится цифра или ничего не выводится
`$`	Выводится символ доллара ($)
`%`	Процентный формат. Число умножается на 100 и к нему добавляется символ процентов
`E-` или `e-`	Экспоненциальная нотация (перед отрицательными показателями степени изображается знак минус, а перед положительными ничего). Этот символ используется только вместе с другими символами (например, `0`, `00E-00`).

Таблица 2.8 *(окончание)*

Символ	Описание
`E+` или `e+`	Экспоненциальная нотация (перед отрицательными показателями степени изображается знак минус, а перед положительными знак плюс). Этот символ используется только вместе с другими символами (например, `0,00E+00`).

Замечание

Для добавления в специальный формат запятой или другого символа разделителя следует ввести этот символ в кавычках, например, формат `"#"##` задает последовательности вида `#1, #35, #, #99`.

Таблица 2.9. *Встроенные форматы для полей типа* ***Дата/Время***

Значение	Описание
Полный формат даты (General Date)	Значение используется по умолчанию. Если значение содержит только дату, то время не отображается; а если — время, то не отображается дата. Данный формат является комбинацией двух форматов: **Краткий формат даты** и **Длинный формат времени**. Примеры: 01.01.1999 12:08:02 и 23.02.1999 00:00:00
Длинный формат даты (Long Date)	Совпадает с настройкой, задающейся в окне **Язык и стандарты** (Regional and Language Options) панели управления Windows. Пример: 12 января 1999 г.
Средний формат даты (Medium Date)	В российской практике используется редко. Пример: 03-янв-1999
Краткий формат даты (Short Date)	Пример: 11.06.2002
Длинный формат времени (Long Time)	Пример: 20:58:10
Средний формат времени (Medium Time)	В российской практике обычно не используется. Пример: 5:34 PM
Краткий формат времени (Short Time)	Пример: 17:34

Специальные форматы даты и времени определяются с помощью символов форматирования, перечисленных в табл. 2.10.

Таблица 2.10. *Символы форматирования для типа данных* ***Дата/Время***

Символ	Описание
: (двоеточие)	Разделитель компонентов времени. Символ разделителя выбирается в окне **Язык и стандарты** (Regional and Language Options) Панели управления Windows

Таблица 2.10 *(продолжение)*

Символ	Описание
/	Разделитель компонентов даты
c	Задает встроенный **Полный формат даты**
d	Номер дня месяца, состоящий из 1 или 2 цифр (1—31)
dd	Номер дня месяца, состоящий из 2 цифр (01—31)
ddd	Сокращенное название дня недели (Пн—Вс)
dddd	Полное название дня недели (понедельник—воскресенье)
ddddd	Задает встроенный **Краткий формат даты**
dddddd	Задает встроенный **Длинный формат даты**
w	Номер дня недели (1—7)
ww	Номер недели в году (1—53)
m	Номер месяца, состоящий из 1 или 2 цифр (1—12)
mm	Номер месяца, состоящий из 2 цифр (01—12)
mmm	Первые три буквы названия месяца (янв—дек)
mmmm	Полное название месяца (Январь—Декабрь)
q	Номер квартала в году (1—4)
y	Номер дня в году (1—366)
yy	Последние две цифры номера года (01—99)
yyyy	Полный номер года (0100—9999)
h	Число часов, состоящее из 1 или 2 цифр (0—23)
hh	Число часов, состоящее из 2 цифр (00—23)
n	Число минут, состоящее из 1 или 2 цифр (0—59)
nn	Число минут, состоящее из 2 цифр (00—59)
s	Число секунд, состоящее из 1 или 2 цифр (0—59)
ss	Число секунд, состоящее из 2 цифр (00—59)
ttttt	Задает встроенный **Длинный формат времени**
AM/PM	12-часовой формат времени с добавлением прописных букв **AM** или **PM**
am/pm	12-часовой формат времени с добавлением строчных букв **am** или **pm**
A/P	12-часовой формат времени с добавлением прописных букв **A** или **P**
a/p	12-часовой формат времени с добавлением строчных букв **a** или **p**

Таблица 2.10 *(окончание)*

Символ	Описание
AMPM	12-часовой формат времени; используется индикатор "утро/день", выбранный в окне **Язык и стандарты** (Regional and Language Options) Панели управления Windows
"A.D. " #;# " B.C."	Формат, позволяющий выводить даты "до нашей эры" и "нашей эры". Отрицательное число означает год до нашей эры, поэтому будет отображаться число (без знака минус) и символы **B.C.** после него. Положительное число означает год нашей эры и перед ним будут выводиться символы **A.D.**

Примеры:

ddd", "d\ mmmm\ yyyy	Пн, 1 января 2001
dd\ mmm\ yy	01 янв 01
dddd", "dd\ mmmmm\ yyyy	понедельник, 01 января 2001

Замечание

Специальные форматы выводятся в соответствии со значениями, установленными в окне **Язык и стандарты** (Regional and Language Options) панели управления Windows. Специальные форматы, противоречащие настройкам окна **Язык и стандарты**, игнорируются.

Специальные форматы для текстовых полей и полей MEMO могут включать один или два раздела, разделяемых точкой с запятой (;). В первом разделе описывается формат отображения текста, во втором — формат отображения строк нулевой длины и пустых (Null) значений.

Замечание

В Access различаются понятия пустого значения и строки нулевой длины. Если поле имеет значение Null — пустое, то это означает, что поле не имеет никакого значения. Если значением текстового или MEMO-поля является строка нулевой длины, то значение этого поля известно — пустая строка. Пустое значение не может быть равно никакому значению, в том числе другому пустому значению, поэтому связывать таблицы по значению Null нельзя. Пустые строки считаются равными друг другу, поэтому по пустым строкам таблицы связывать можно.

Таблица 2.11. *Символы форматирования для текстовых и MEMO-полей*

Символ	Описание
@	Обязательный текстовый символ или пробел
&	Необязательный текстовый символ
<	Преобразует все символы в строчные
>	Преобразует все символы в прописные

Для логических полей свойство **Формат** (Format) позволяет указать использование встроенных форматов: **Истина/Ложь** (True/False), **Да/Нет** (Yes/No) или **Вкл/Выкл** (On/Off). В Access элементом управления, используемым по умолчанию для логических полей, является **Флажок** (Check Box). При его применении игнорируются встроенные и специальные форматы.

Специальный логический формат может содержать от одного до трех разделов, разделяемых точкой с запятой (;). Первый раздел не влияет на формат логических значений, но символ точки с запятой указать необходимо. Во втором разделе задается строковое выражение, определяющее значение **Истина** (True), **Да** (Yes) или **Вкл** (On), а в третьем — строковое выражение, определяющее значение **Ложь** (False), **Нет** (No) или **Выкл** (Off).

Не разрешается смешивать в одном формате специальные символы, предназначенные для определения числовых форматов, форматов даты/времени и текстовых форматов. Но существует ряд символов форматирования, которые используются для любых типов данных. Эти символы представлены в табл. 2.12.

Таблица 2.12. *Символы форматирования для всех типов данных*

Символ	Описание
(Пробел)	Выводит пробел как символьную константу, т. е. этот пробел выводится всегда
"ABC"	Символы, заключенные в кавычки, всегда отображаются в поле
!	Определяет выравнивание символов по левому краю
*	Позволяет заполнить доступное пустое пространство следующим символом
\	Выводит следующий символ как символьную константу
[цвет]	Задает цвет, название которого указано в скобках. Допустимые имена цветов: (Черный) (Black), (Синий) (Blue), (Зеленый) (Green), (Бирюзовый) (Cyan), (Красный) (Red), (Лиловый) (Magenta), (Желтый) (Yellow), (Белый) (White)

Определение маски ввода

Если свойство **Формат** (Format) управляет тем, как данные отображаются в поле таблицы, то свойство **Маска ввода** (Input Mask) позволяет контролировать ввод данных в таблицу. И не только контролировать, но и упрощать процесс ввода. Маску используют в том случае, когда вводимые данные должны содержать определенные символы в некоторых позициях вводимой строки. Самым простым и ярким примером таких данных являются номера телефонов. Маска ввода должна обеспечить возможность вводить только цифры номера, а остальные символы (скобки вокруг кода города, дефис между цифрами номера) будут добавляться автоматически. Маска задается с помощью символов, приведенных в табл. 2.13.

Для того чтобы сформировать маску ввода, можно использовать Мастер масок ввода (Input Mask Wizard). Откройте в режиме Конструктора таблицу Заказы (Orders) (рис. 2.22). Выделите поле ДатаРазмещения (OverDate). Чтобы ввести маску ввода для

Таблица 2.13. *Символы для создания масок ввода*

Символ маски	Описание
0	В данную позицию должна быть введена цифра. Знаки плюс (+) и минус (-) не допускаются
9	В данную позицию должна быть введена цифра или пробел. Знаки плюс (+) и минус (-) не допускаются
#	В данную позицию должна быть введена цифра, пробел, знаки плюс (+) или минус (-)
L	В данную позицию должна быть введена буква
?	В данную позицию может быть введена буква или пробел
A	В данную позицию должна быть введена буква или цифра
a	В данную позицию должна быть введена буква, цифра или пробел
&	В данную позицию должен быть введен произвольный символ или пробел
C	В данную позицию может быть введен произвольный символ или пробел. Если пользователь ничего не введет, Access не занесет в эту позиции никаких данных
.	Десятичный разделитель (зависит от региональных установок в окне **Язык и стандарты** (Regional and Language Options) Панели управления Windows)
,	Разделитель групп разрядов (зависит от региональных установок в окне **Язык и стандарты** (Regional and Language Options) Панели управления Windows)
: - /	Разделители в значениях даты и времени (зависят от региональных установок в окне **Язык и стандарты** (Regional and Language Options) Панели управления Windows)
<	Преобразует все символы справа к нижнему регистру
>	Преобразует все символы справа к верхнему регистру
!	Указывает, что маску нужно заполнять справа налево. Этот символ следует использовать в том случае, когда символы в левой части маски являются необязательными. Его можно помещать в любой позиции маски
\	Указывает, что следующий символ необходимо рассматривать в качестве постоянного символа, даже если он является специальным символом маски. Например, \A будет выводить в маске букву A
"литерал"	Вместо того чтобы многократно использовать символ обратного слэша (\), можно просто заключить любой литерал в двойные кавычки
Пароль	Слово "Пароль" в маске ввода позволяет скрывать вводимые в поле символы. Вместо них будет отображаться знак звездочка (*). Сохраняться при этом в поле будут сами символы

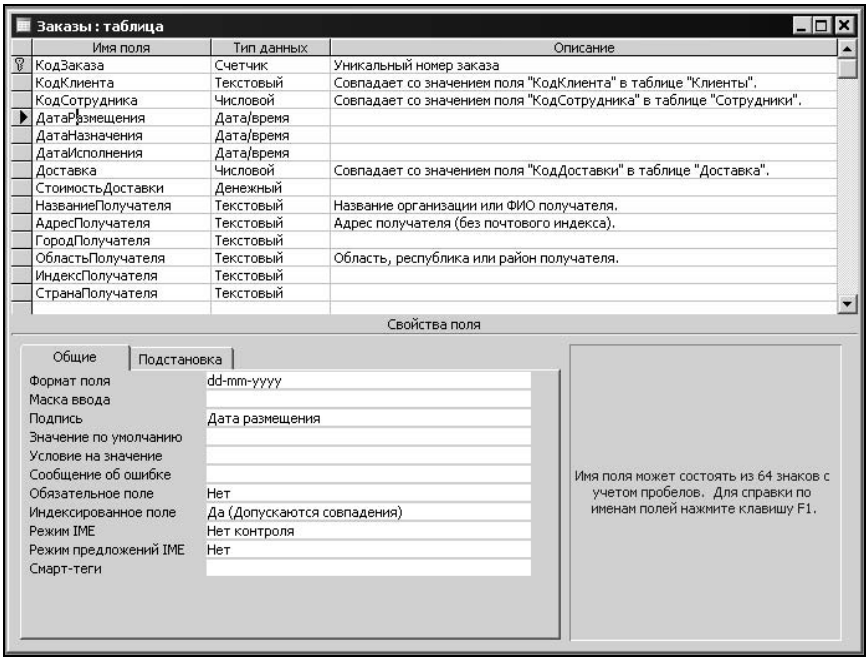


Рис. 2.22. Таблица `Заказы` в режиме Конструктора

этого поля, щелкните мышью по ячейке свойства **Маска ввода** (Input Mask). Нажмите небольшую кнопку с тремя точками, которая появится справа (эта кнопка называется кнопкой Построителя и будет встречаться еще во многих местах), и дальше следуйте указаниям Мастера:

1. В первом диалоговом окне вы можете выбрать нужную маску из списка (рис. 2.23). Если вы попытаетесь вводить данные в поле **Проба** (Try It), то сможете увидеть, как будут форматироваться вводимые вами символы. Нажмите кнопку **Далее** (Next).
2. Во втором диалоговом окне вы увидите название маски, соответствующую ей строку символов, поле со списком **Заполнитель** (Placeholder character), в котором можно выбрать символ-указатель заполняемых при вводе позиций, и поле проверки работы маски (рис. 2.24). Вы можете скорректировать символьную строку маски, чтобы добиться желаемого эффекта. Нажмите кнопку **Далее** (Next).
3. В последнем диалоговом окне нажмите кнопку **Готово**.

Сформированная символьная строка маски отображается в поле **Маска ввода** (Input Mask) для поля `ДатаРазмещения` (OrderDate) (рис. 2.25). Маска ввода состоит из трех частей:

- символьной строки маски;
- символа `0` (Это означает, что введенные в поле данные будут сохраняться вместе со специальными символами. Можно заменить `0` на `1` — тогда специальные символы маски не будут запоминаться в поле.);
- символа заполнителя.

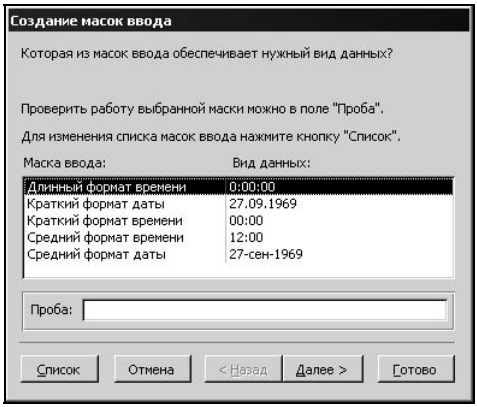


Рис. 2.23. Первое диалоговое окно Мастера масок ввода

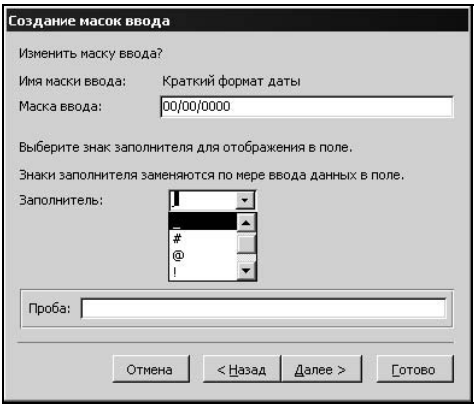


Рис. 2.24. Второе диалоговое окно Мастера масок ввода

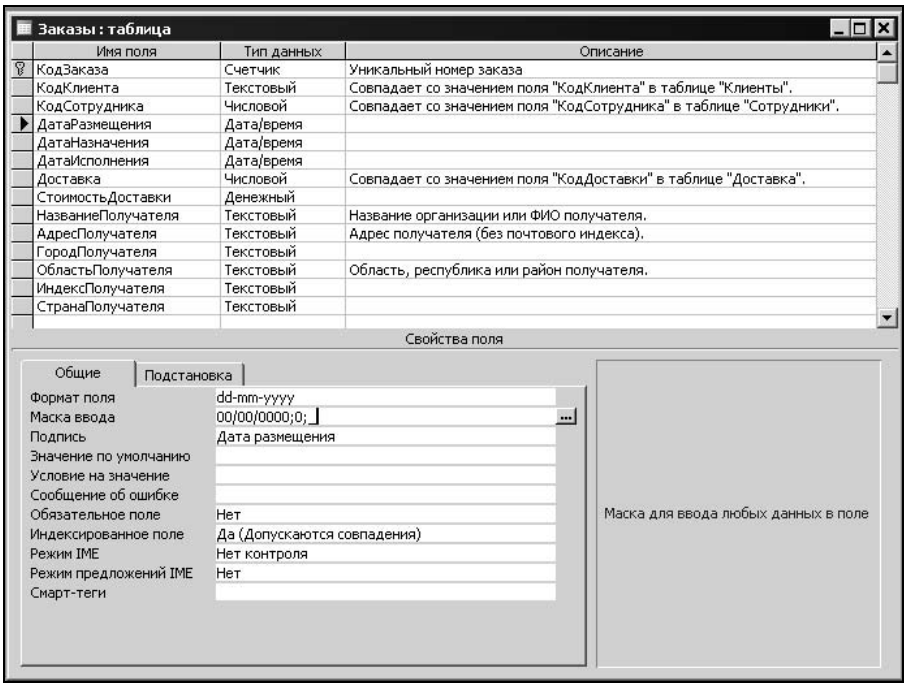


Рис. 2.25. Маска ввода для поля `ДатаРазмещения`

Вставка, удаление и переименование полей таблицы

Рассмотрим теперь, каким образом можно переименовывать, вставлять и удалять поля в режиме Конструктора таблиц.

Чтобы переименовать поле, необходимо:

1. Щелкнуть левой кнопкой мыши в столбце **Имя поля** (Field Name) на имени поля, подлежащего переименованию.
2. Ввести новое имя поля или отредактировать старое имя.
3. Нажать клавишу <Enter> или переместить указатель текущей записи на любую другую запись.

Чтобы вставить поле, надо сначала выделить строку, перед которой необходимо вставить поле. Для этого нужно щелкнуть левой кнопкой мыши в области выделения данной строки. После чего вставить новое поле можно одним из следующих способов:

- нажать клавишу <Insert>;
- нажать кнопку **Добавить строки** (Insert Rows) на панели инструментов **Конструктор таблиц** (Table Design);
- выбрать в главном меню команду **Вставка** (Insert) и из раскрывающегося меню — команду **Строки** (Rows);
- щелкнуть правой кнопкой мыши на выделенной записи и из контекстного меню выбрать команду **Добавить строки** (Insert Rows).

Затем в полученную пустую строку нужно ввести имя нового поля, тип и описание.

Замечание

Чтобы вставить сразу несколько строк, необходимо выделить требуемое количество существующих строк, перед которыми нужно вставить новые, а затем выполнить команду вставки строк в соответствии с приведенной выше процедурой.

Чтобы удалить поле, выделите строку (строки) описания поля (полей), которую необходимо удалить. Затем используйте любой из трех вариантов удаления выделенной строки (строк):

- нажать клавишу <Delete>;
- выбрать из главного меню команду **Правка** (Edit) и из раскрывающегося меню — команду **Удалить** (Delete) или **Удалить строки** (Delete Rows);
- щелкнуть правой кнопкой мыши на выделенной строке (строках) и из контекстного меню выбрать команду **Удалить строки** (Delete Rows).

Замечание

При удалении полей, в которых содержатся или могут содержаться данные, Access потребует подтверждения на удаление данного поля.

Переименование, добавление и удаление полей возможны не только в режиме Конструктора, но и в режиме Таблицы. Чтобы переименовать поле в режиме Таблицы, необходимо:

1. Выделить столбец, который необходимо переименовать. Для этого нужно щелкнуть левой кнопкой мыши на имени поля (столбца).
2. Щелкнуть правой кнопкой мыши на выделенном столбце и из контекстного меню выбрать команду **Переименовать столбец** (Rename Column) или выбрать из

главного меню команду **Формат** (Format) и из раскрывающегося меню — команду **Переименовать столбец** (Rename Column).

3. В поле заголовка столбца ввести новое имя поля и нажать клавишу <Enter> или щелкнуть левой кнопкой мыши в любом другом месте таблицы.

Замечание

Самый быстрый способ переименования поля в режиме Таблицы — это дважды щелкнуть левой кнопкой мыши в поле заголовка столбца, ввести новое имя поля и нажать клавишу <Enter>.

Чтобы добавить поле в таблицу в режиме Таблицы, необходимо:

1. Выделить столбец, перед которым требуется вставить новое поле.
2. Щелкнуть правой кнопкой мыши на выделенном столбце и из контекстного меню выбрать команду **Добавить столбец** (Insert Column) или из главного меню выбрать команду **Вставка** (Insert) и из раскрывающегося меню — команду **Столбец** (Column).

Перед выделенным столбцом появится новый столбец, имя которого будет состоять из слова **Поле** (Field) и порядкового номера — **Поле1** (Field1), **Поле2** (Field2) и т. д., — а тип будет иметь значение **Текстовый** (Text) со значениями свойств, заданными по умолчанию.

Чтобы удалить поле из таблицы в режиме Таблицы, необходимо:

1. Выделить столбец, который нужно удалить.
2. Щелкнуть правой кнопкой мыши на выделенном столбце и из контекстного меню выбрать команду **Удалить столбец** (Delete Column) или выбрать из главного меню команду **Правка** (Edit) и из раскрывающегося меню — команду **Удалить столбец** (Delete Column).

Определение ключевых полей

Ранее неоднократно упоминалось понятие ключевого поля. *Ключевое поле* (первичный ключ) — это одно или несколько полей, комбинация значений которых однозначно определяет каждую запись в таблице. Если для таблицы определены ключевые поля, то Access сама предотвращает дублирование или ввод пустых значений в ключевое поле. Ключевые поля используются для быстрого поиска и связи данных из разных таблиц при помощи запросов, форм и отчетов, поэтому рекомендуется их определять.

В Access можно выделить три типа ключевых полей: простой ключ, составной ключ и счетчик. Рассмотрим каждый из этих типов.

Для создания *простого* ключа достаточно иметь в таблице значимое поле (атрибут), которое содержит уникальные значения (например, коды или номера). Именно оно и будет играть роль первичного ключа.

Если окажется, что выбранное поле содержит повторяющиеся или пустые значения, его нельзя определить как ключевое. Чтобы определить записи, содержащие повторяющиеся данные, можно выполнить запрос на поиск повторяющихся записей. Для этого следует выбрать в списке таблиц нужную и нажать кнопку **Новый объект | За-**

прос (New Object | Query) на панели инструментов **База данных** (Datasheet). В окне **Новый запрос** (New Query) выберите элемент **Повторяющиеся записи** (Find Duplicates Query Wizard).

Если устранить повторы путем изменения значений невозможно, следует либо добавить в таблицу поле счетчика и сделать его ключевым, либо определить составной ключ.

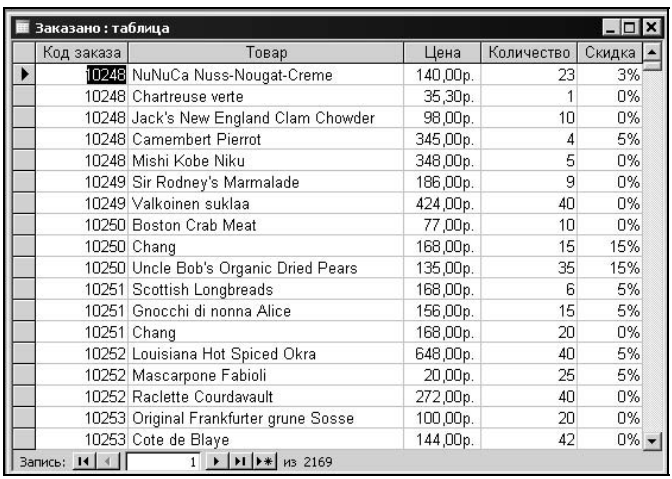
Составной ключ необходим в случае, если невозможно гарантировать уникальность записи с помощью одного поля. Он представляет собой комбинацию нескольких полей. Для определения составного ключа необходимо:

1. Открыть таблицу в режиме Конструктора.
2. Выделить поля, которые нужно определить как ключевые.
3. Нажать кнопку **Ключевое поле** (Primary Key) на панели инструментов **Конструктор таблиц** (Table Design).

Замечание

Для составного ключа существенным может оказаться порядок образующих ключ полей. Сортировка записей осуществляется в соответствии с порядком ключевых полей в окне Конструктора таблицы. Если необходимо указать другой порядок сортировки без изменения порядка ключевых полей, то сначала нужно определить ключ, а затем нажать кнопку **Индексы** (Indexes) на панели инструментов **Конструктор Таблиц** (Table Design). Потом в появившемся окне **Индексы** (Indexes) надо указать другой порядок полей для индекса с именем **Ключевое поле** (PrimaryKey).

Рассмотрим применение составного ключа для таблицы `Заказано` (OrderDetails) базы данных `Борей` (Northwind) (рис. 2.26).



Заказано : таблица

Код заказа	Товар	Цена	Количество	Скидка
10248	NuNuCa Nuss-Nougat-Creme	140,00р.	23	3%
10248	Chartreuse verte	35,30р.	1	0%
10248	Jack's New England Clam Chowder	98,00р.	10	0%
10248	Camembert Pierrot	345,00р.	4	5%
10248	Mishi Kobe Niku	348,00р.	5	0%
10249	Sir Rodney's Marmalade	186,00р.	9	0%
10249	Valkoinen suklaa	424,00р.	40	0%
10250	Boston Crab Meat	77,00р.	10	0%
10250	Chang	168,00р.	15	15%
10250	Uncle Bob's Organic Dried Pears	135,00р.	35	15%
10251	Scottish Longbreads	168,00р.	6	5%
10251	Gnocchi di nonna Alice	156,00р.	15	5%
10251	Chang	168,00р.	20	0%
10252	Louisiana Hot Spiced Okra	648,00р.	40	5%
10252	Mascarpone Fabioli	20,00р.	25	5%
10252	Raclette Courdavault	272,00р.	40	0%
10253	Original Frankfurter grune Sosse	100,00р.	20	0%
10253	Cote de Blaye	144,00р.	42	0%

Запись: 1 из 2169

Рис. 2.26. Пример таблицы с использованием составного ключа

В данном случае в качестве составного ключа используются поля `Код заказа` (OrderID) и `КодТовара` (ProductID), т. к. ни одно из этих полей в отдельности не га-

рантирует уникальность записи. При этом в таблице выводится не код товара, а наименование товара, т. к. поле КодТовара (ProductID) данной таблицы содержит подстановку из таблицы Товары (Products), а значения полей КодТовара (ProductID) этих таблиц связаны отношением "один-ко-многим" (одной записи таблицы Товары (Products) может соответствовать несколько записей таблицы Заказано (Order-Details)). Оба поля могут содержать повторяющиеся значения. Так, один заказ может включать в себя несколько товаров, а в разные заказы могут включаться одинаковые товары. В то же время сочетание полей КодЗаказа (OrderID) и КодТовара (ProductID) однозначно определяет каждую запись таблицы Заказы (OrderDetails).

Если найти в таблице поля, которые могли бы уникально идентифицировать запись, сложно, то прибегают к созданию специального поля — счетчика записей.

Для создания ключевого поля типа *счетчик* необходимо в режиме Конструктора таблиц:

1. Включить в таблицу поле счетчика.
2. Задать для него значение свойства **Новые значения** (New Values) равным **Последовательные** (Increment).
3. Указать это поле в качестве ключевого путем нажатия кнопки **Ключевое поле** (Primary Key) на панели инструментов **Конструктор таблиц** (Table Design.)

При таком задании ключевого поля значение в этом поле будет увеличиваться на 1 для каждой новой записи в таблице.

Если при создании таблицы ключевые поля не были определены, то при ее сохранении будет выдано сообщение, предлагающее определить ключевое поле. При нажатии кнопки **Да** (Yes) будет создано ключевое поле счетчика с именем **Код** (ID) и типом данных **Счетчик** (AutoNumber) именно такого типа, как мы описали.

Замечание

Кроме такого простого счетчика, можно создать счетчик, значения которого будут представлять собой случайные числа (свойство **Новые значения** (New Values) должно иметь значение **Случайные** (Random)) или глобальный уникальный идентификатор (GUID) (в этом случае свойство **Размер поля** (Field Size) должно иметь значение **Код репликации** (Replication ID)).

Чтобы изменить ключ, необходимо:

1. Открыть таблицу в режиме Конструктора.
2. Выбрать имеющиеся ключевые поля.
3. Нажать кнопку **Ключевое поле** (Primary Key), при этом она должна принять положение **Выкл**, а из области выделения должны исчезнуть значки ключевого поля.
4. Выбрать поле, которое необходимо сделать ключевым.
5. Нажать кнопку **Ключевое поле** (Primary Key). При этом в области выделения должен появиться значок ключевого поля.

Чтобы удалить ключ, необходимо:

1. Открыть таблицу в режиме Конструктора.
2. Выбрать имеющееся ключевое поле (ключевые поля).

3. Нажать кнопку **Ключевое поле** (Primary Key), при этом она должна принять положение **Выкл**, а из области выделения должен исчезнуть значок (значки) ключевого поля.

Замечание

Если ключевое поле таблицы используется в какой-нибудь связи с другой таблицей, то так просто ключ не удалить — будет выдано сообщение об ошибке. Поэтому прежде чем удалить ключевое поле, нужно удалить связь.

Создание и использование индексов

С целью ускорения поиска и сортировки данных в любой СУБД используются индексы. *Индекс* является средством, которое обеспечивает быстрый доступ к данным в таблице на основе значений одного или нескольких полей. Индекс представляет собой упорядоченный список значений и ссылок на те записи, в которых хранятся эти значения. Чтобы найти нужные записи, СУБД сначала ищет требуемое значение в индексе, а затем по ссылкам быстро отбирает соответствующие записи. Индексы бывают двух типов: простые и составные. *Простые* индексы представляют собой индексы, созданные по одному полю. Индекс, построенный по нескольким полям, называется *составным*. Примером составного индекса может быть индекс, построенный по полям `Фамилия` и `Имя`.

Однако применение индексов приносит не только преимущества, но и недостатки. Главным среди них является тот, что при добавлении и удалении записей или при обновлении значений в индексном столбце требуется обновлять индекс, что при большом количестве индексов в таблице может замедлять работу. Поэтому индексы обычно рекомендуется создавать только для тех столбцов таблицы, по которым наиболее часто выполняется поиск записей или сортировка. Во многих СУБД (например, FoxPro) индексы хранятся в отдельных файлах и являются предметом заботы разработчиков, т. к. при нарушении индекса поиск данных выполняется некорректно. В Access индексы хранятся в том же файле базы данных, что и таблицы и другие объекты Access. Индексировать можно любые поля, кроме объектов OLE, однако если в поле содержится много одинаковых значений (например, в поле типа **Логический** (Yes/No)), то индексирование не дает существенного эффекта.

Чтобы создать простой индекс, необходимо:

1. Открыть таблицу в режиме Конструктора.
2. Выбрать поле, для которого необходимо создать индекс.
3. Открыть вкладку **Общие** (General) и выбрать для свойства **Индексированное поле** (Indexed) значение **Да (Допускаются совпадения)** (Yes (Duplicates OK)) или **Да (Совпадения не допускаются)** (Yes (No duplicates)) (рис. 2.27).

Ключевое поле таблицы автоматически индексируется и свойству **Индексированное поле** (Indexed) присваивается значение **Да (Совпадения не допускаются)** (Yes (No duplicates)).

Составной индекс создается в специальном диалоговом окне. Чтобы создать составной индекс, необходимо:

1. Открыть таблицу в режиме Конструктора.
2. На панели инструментов **Конструктор таблиц** (Table Design) нажать кнопку **Индексы** (Indexes).

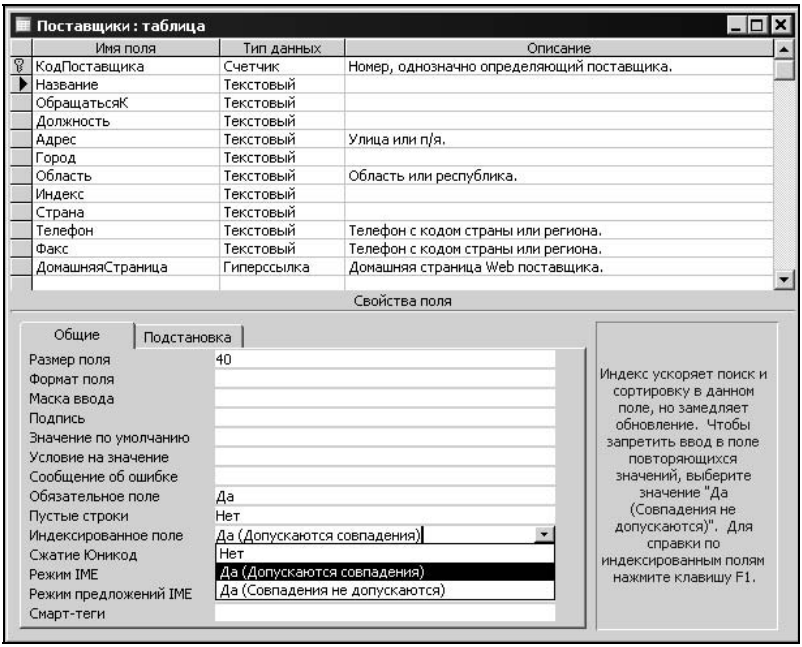


Рис. 2.27. Установка индекса для поля

3. В первой пустой строке (рис. 2.28) поля **Индекс** (Index Name) ввести имя индекса.

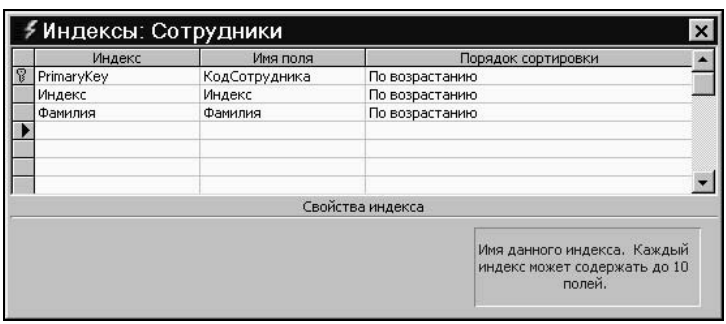


Рис. 2.28. Диалоговое окно **Индексы**

4. В поле **Имя поля** (Field Name) нажать на стрелку и выбрать первое поле, для которого необходимо создать индекс.
5. В следующей строке поля **Имя поля** (Field Name) указать второе индексируемое поле. (Для данной строки поле **Индекс** (Index Name) должно оставаться пустым.) Повторите эту операцию для всех полей, которые необходимо включить в индекс. (В индексе может быть использовано до 10 полей.)

Замечание

По умолчанию устанавливается порядок сортировки **По возрастанию** (Ascending). Для сортировки данных полей по убыванию в поле **Порядок сортировки** (Sort Order) в окне индексов укажите значение **По убыванию** (Descending).

Диалоговое окно **Индексы** (Indexes) используется также для просмотра, изменения и удаления существующих индексов. Изменить можно:

- название индекса в поле **Индекс** (Index Name);
- поле таблицы, соответствующее данному индексу, выбрав новое поле из списка в поле **Имя поля** (Field Name);
- порядок сортировки в поле **Порядок сортировки** (Sort Order);
- свойства данного индекса в нижней части окна (рис. 2.29):
  - **Ключевое поле** (Primary) определяет, является ли индексированное поле ключевым;
  - **Уникальный индекс** (Unique) определяет, должно ли быть каждое значение в этом поле уникальным;
  - **Пропуск пустых полей** (Ignore Nulls) определяет, включаются или нет в индекс записи с пустым (Null) значением данного поля.

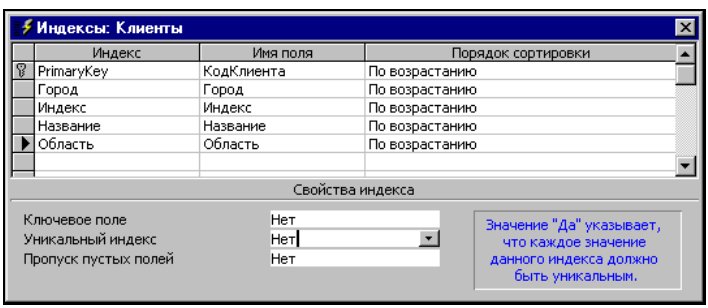


Рис. 2.29. Диалоговое окно индексов для таблицы `Клиенты`

Удаление индекса выполняется точно так же, как удаление поля в Конструкторе таблиц. В диалоговом окне **Индексы** (Indexes) выделите строку (или строки в случае составного индекса) с нужным индексом и нажмите клавишу <Delete> или воспользуйтесь контекстным меню.

Связывание таблиц на схеме данных

Для того чтобы было удобно просматривать, создавать, удалять и изменять связи между таблицами, в Access используется **Схема данных** (Relationships) (рис. 2.30).

Чтобы открыть схему данных, необходимо выполнить команду **Сервис | Схема данных** (Tools | Relationships) или нажать кнопку **Схема данных** (Relationships) на панели инструментов **База данных** (Database). По умолчанию схема будет содержать все таблицы со связями. Поэтому сюда будет включена, например, наша новая таблица

Список рассылки оповещений, которая была связана с таблицей Клиенты (Customers) при помощи Мастера подстановок *(см. разд. "Использование Мастера подстановок" данной главы)*.

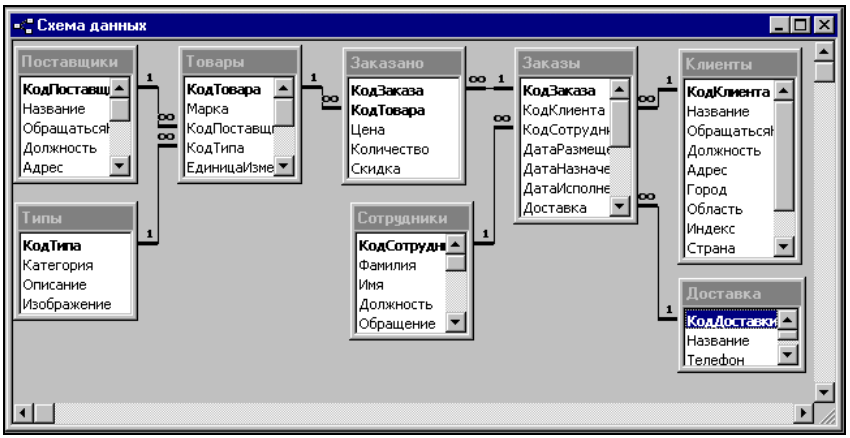


Рис. 2.30. Пример схемы данных

Чтобы убрать какую-либо таблицу из схемы данных, необходимо щелкнуть правой кнопкой мыши на любом месте этой таблицы и из контекстного меню выбрать команду **Скрыть таблицу** (Hide table)) или выделить эту таблицу и нажать клавишу <Delete>. Например, на рис. 2.30 таблица Список рассылки оповещений скрыта.

Замечание

Удаление таблицы из схемы данных не означает удаление ее из базы данных, просто она не присутствует в схеме.

Чтобы вновь добавить в эту схему спрятанную нами таблицу (Список рассылки оповещений) или таблицу, у которой связи еще не установлены, необходимо:

1. Щелкнуть правой кнопкой мыши на свободном пространстве схемы данных и из контекстного меню выбрать команду **Добавить таблицу** (Show table) или нажать кнопку **Отобразить таблицу** на панели инструментов **Связь** (Relationship).
2. В диалоговом окне **Добавление таблицы** (Show table) раскрыть вкладку **Таблицы** (Tables), выбрать из списка таблицу Список рассылки оповещений и нажать кнопку **Добавить** (Add) или просто дважды щелкнуть по ней (рис. 2.31).

Замечание

Если нужно добавить к схеме несколько таблиц, выберите в списке все нужные таблицы, пользуясь клавишами <Shift> или <Ctrl>, и только потом нажмите кнопку **Добавить** (Add) или добавляйте каждую таблицу двойным щелчком мыши.

3. Нажать кнопку **Закрыть** (Close), чтобы закрыть диалоговое окно **Добавление таблицы** (Show table).

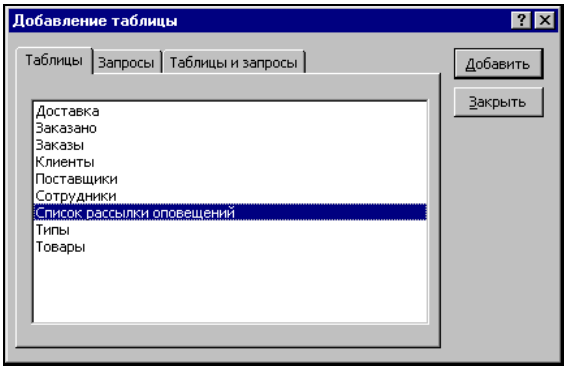


Рис. 2.31. Диалоговое окно **Добавление таблицы**

В данном случае связь между таблицами Access строит автоматически. Это происходит тогда, когда две таблицы имеют одинаковые названия связанных полей и согласованные типы данных, причем хотя бы в одной из таблиц связанное поле является ключевым.

Замечание

Согласованные типы данных означают следующее: если ключевое поле имеет тип данных **Счетчик** (AutoNumber), то соответствующее ему поле в другой таблице должно иметь тип **Длинное целое** (Long Integer). В остальных случаях типы данных должны просто совпадать.

Если нужная связь автоматически не создана, ее можно создать двумя путями. Первый путь — графический. Следует выбрать в главной таблице поле для связи, нажать левую кнопку мыши и перетащить поле во вторую таблицу. Отпустить левую кнопку мыши над тем полем подчиненной таблицы, с которым устанавливается связь. После этого появится диалоговое окно **Изменение связей** (Edit Relationships) (рис. 2.32).

Альтернативный вариант — выбрать команду **Связи | Изменить связь** (Relationships | Edit Relationships) из главного меню Access. Эта команда появляется в меню, когда открыто окно **Схема данных** (Relationships).

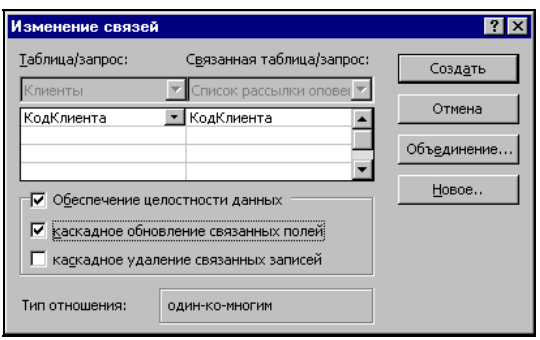


Рис. 2.32. Окно **Изменение связей**

В диалоговом окне **Изменение связей** (Edit Relationships) можно выбрать из списков названия связанных таблиц и полей для связывания. Если отношение между таблицами "один-ко-многим", то слева из списка **Таблица/запрос** (Table/Query) выбирается главная таблица и поле в этой таблице, а справа из списка **Связанная таблица/запрос** — подчиненная и соответственно поле в ней. Если отношение "один-к-одному", то порядок таблиц значения не имеет. Если вы устанавливали связь графически, то все поля в списках уже выбраны, и нужно только определить правила ссылочной целостности.

Для этого устанавливают флажок: **Обеспечение целостности данных** (Enforce Referential Integrity) и один или оба флажка: **каскадное обновление связанных полей** (Cascade Update Related Fields) и **каскадное удаление связанных записей** (Cascade Delete Related Records) *(см. следующий раздел)*.

При необходимости можно изменить параметры объединения. Для этого нужно нажать на кнопку **Объединение** (Join Type) и выбрать один из переключателей (рис. 2.33):

1. **Объединение только тех записей, в которых связанные поля обеих таблиц совпадают**.
2. **Объединение ВСЕХ записей из *<имя первой таблицы>* и только тех записей из *<имя второй таблицы>*, в которых связанные поля совпадают**.
3. **Объединение ВСЕХ записей из *<имя второй таблицы>* и только тех записей из *<имя первой таблицы>*, в которых связанные поля совпадают**.

По умолчанию устанавливается переключатель **1**. Нажать кнопку **OK**.

(Подробную информацию о параметрах объединения см. в разд. "Виды соединений" гл. 4.)

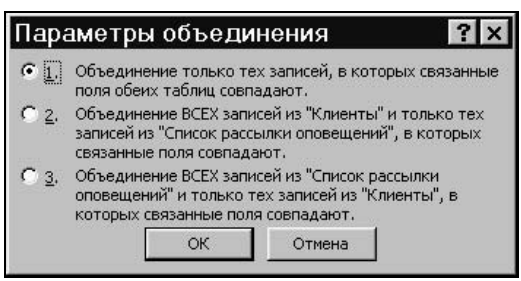


Рис. 2.33. Диалоговое окно **Параметры объединения**

Когда создается новая связь, можно воспользоваться кнопкой **Новое** (Create New) и в окне **Создание** (Create New) ввести имена связываемых таблиц, а также имена полей, используемых для связи, после чего нажать кнопку **OK**.

После установки всех параметров связи нажать кнопку **OK** в окне **Изменение связей** (Edit Relationship).

Когда таблица удаляется из макета, связи ее с другими таблицами не удаляются, они только не отображаются. Если нужно удалить связь между таблицами, необходимо выделить эту связь на схеме (она должна стать жирной) и нажать клавишу <Delete> или выполнить команду из контекстного меню **Удалить** (Delete).

Как мы уже сказали, в окне **Схема данных** (Relationships) могут показываться не все таблицы и не все связи между таблицами. Поэтому если нужно посмотреть все связи, существующие в базе данных, необходимо нажать кнопку **Отобразить все связи** (Show all Relationships) на панели инструментов **Связь** (Relationship) или выполнить соответствующую команду меню **Связи** (Relationships). При этом на схеме появятся все связанные таблицы.

Если база данных содержит много таблиц, то такая схема может оказаться очень запутанной, поэтому иногда бывает интересно посмотреть только часть общей схемы. Например, можно посмотреть только связи одной конкретной таблицы. Для этого выполните следующие действия:

1. Нажмите кнопку **Очистить макет** (Clear Layout) на панели инструментов **Связь** (Relationship). Вся схема будет удалена, поле окна станет пустым.
2. Теперь добавьте к схеме нужную таблицу, как было описано ранее, и нажмите кнопку **Отобразить прямые связи** (Show Direct Relationships) на панели инструментов **Связь** (Relationship).

На рис. 2.34 показана схема, отображающая все связи для таблицы `Заказы`.

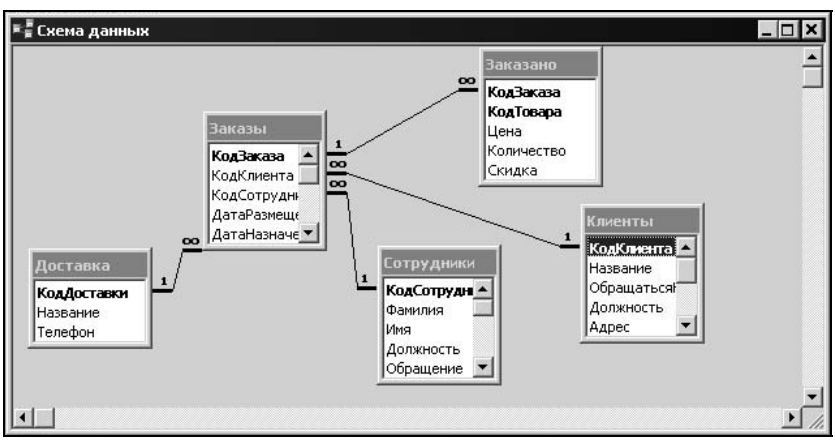


Рис. 2.34. Схема прямых связей таблицы `Заказы`

Созданную схему данных (полную или частичную) можно сохранить. Для этого просто закройте окно схемы данных и на вопрос, нужно ли сохранять макет, ответьте **Да** (Yes) или нажмите перед закрытием кнопку **Сохранить** (Save) на панели инструментов **Связь** (Relationship).

Замечание

К сожалению, сохраняется только единственная схема данных. Для каждой базы данных одновременно сохранить несколько схем невозможно.

Схему данных можно вывести на печать. Для этого существует специальная команда в меню **Файл** (File) — **Печать схемы данных** (Print Relationships). В результате будет создан отчет Access, на котором в графическом виде отображена схема. Этот отчет открывается в режиме предварительного просмотра и его можно напечатать, нажав

кнопку **Печать** (Print) на панели инструментов, или сохранить в виде отчета для дальнейшего применения, для чего используйте команду меню **Файл | Сохранить** (File | Save). На рис. 2.35 показан такой отчет для базы данных Борей.

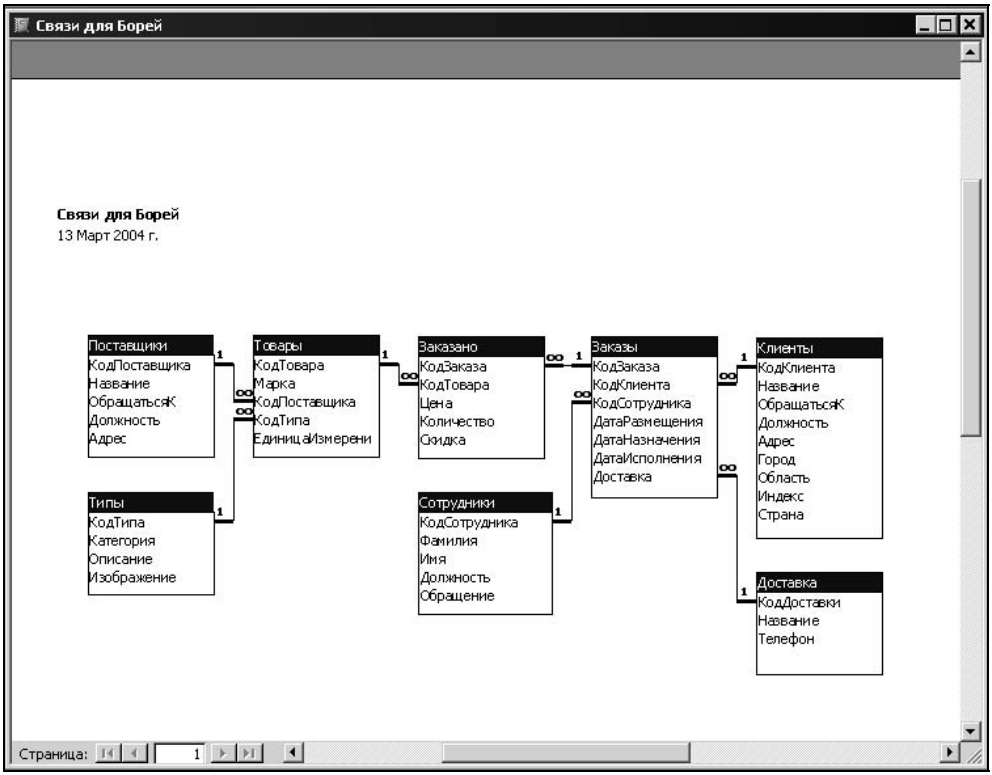


Рис. 2.35. Отчет, созданный при печати схемы данных

Поддержка целостности данных

В Access обеспечивается возможность автоматической проверки целостности данных в связанных полях. *Целостность данных* означает систему правил, используемых для поддержания связей между записями в связанных таблицах, а также для обеспечения защиты от случайного удаления или изменения связанных данных. Установить проверку целостности данных можно, если выполнены следующие условия:

- связанное поле главной таблицы является ключевым полем или имеет уникальный индекс;
- связанные поля имеют один тип данных;

Замечание

Здесь существует два исключения. Поле счетчика может быть связано с числовым полем, если в последнем поле в свойстве **Размер поля** (FieldSize) указано значение

Длинное целое (Long Integer), или в обоих полях свойство **Размер поля** (FieldSize) имеет значение **Код репликации** (Replication ID).

- обе таблицы принадлежат одной базе данных Microsoft Access.

Замечание

Таблицы одного приложения могут содержаться и в нескольких файлах базы данных *(см. также гл. 3)*.

Чтобы обеспечить целостность, работа с данными должна производиться с учетом перечисленных далее правил.

- Невозможно ввести в связанное поле подчиненной таблицы значение, отсутствующее в связанном поле главной таблицы. Однако можно ввести пустое значение, показывающее, что для данной записи связь отсутствует.
- Не допускается удаление записи из главной таблицы, если существуют связанные с ней записи в подчиненной таблице.
- Невозможно изменить значение ключевого поля в главной таблице, если существуют записи, связанные с данной таблицей.

Чтобы эти правила контролировались для конкретной связи, при ее создании следует установить флажок **Обеспечение целостности данных** (Enforce Referential Integrity). Тогда любая попытка выполнить действие, нарушающее одно из перечисленных ранее правил, приведет к выводу на экран предупреждения, а само действие выполнено не будет.

Чтобы преодолеть ограничения на удаление или изменение связанных записей, сохраняя при этом целостность данных, следует установить флажки **каскадное обновление связанных полей** (Cascade Update Related Fields) и **каскадное удаление связанных записей** (Cascade Delete Related Records). Если установлен флажок **каскадное обновление связанных полей** (Cascade Update Related Fields), то при изменении ключевого поля главной таблицы автоматически будут изменены и соответствующие значения поля в связанных записях. Если установлен флажок **каскадное удаление связанных записей** (Cascade Delete Related Records), то при удалении записи в главной таблице удаляются и все связанные записи в подчиненной таблице.

Ввод и проверка данных

До сих пор мы обсуждали все вопросы, связанные с созданием таблиц, определением и изменением их структуры. Эти изменения выполняются в режиме Конструктора таблиц. Теперь мы переходим к описанию возможностей, которые предоставляет Access при работе с таблицами в режиме Таблицы. Этот режим связан с вводом и обновлением данных в таблице.

Навигация по таблице

Для перемещения по записям используются кнопки, расположенные на нижней границе окна таблицы (рис. 2.36). Описание этих кнопок приведено в табл. 2.14.



Рис. 2.36. Кнопки навигации в окне таблицы

Таблица 2.14. *Кнопки перемещения по записям таблицы*

Кнопка	Описание
|<	Перемещение на первую запись таблицы
<	Перемещение на предыдущую запись таблицы
>	Перемещение на следующую запись таблицы
>|	Перемещение на последнюю запись таблицы
>*	Создание новой записи

Помимо кнопок перемещения в нижней части окна таблицы расположено поле, в котором отображается номер текущей записи. Для перехода на запись с известным номером можно его ввести в данное поле и нажать клавишу <Enter>. Но обычно номер записи не известен. Поиск конкретной записи можно осуществлять разными способами, в том числе с помощью полос прокрутки.

Для перемещения по полям записи и между записями в режиме Таблицы можно использовать также сочетания клавиш, показанные в табл. 2.15.

Таблица 2.15. *Навигация по таблице с помощью клавиш*

Сочетания клавиш	Действие
<F5>	Переход в поле номера записи
<Tab>, <Enter> или <→>	Переход к следующему полю
<End>	Переход к последнему полю в текущей записи
<Shift>+<Tab> или <←>	Переход к предыдущему полю
<Home>	Переход к первому полю в текущей записи в режиме перемещения
<↓>	Переход к текущему полю в следующей записи
<Ctrl>+<↓>	Переход к текущему полю в последней записи
<Ctrl>+<END>	Переход к последнему полю в последней записи
<↑>	Переход к текущему полю в предыдущей записи
<Ctrl>+<↑>	Переход к текущему полю в первой записи
<Ctrl>+<Home>	Переход к первому полю в первой записи
<Page Down>	На один экран вниз
<Page Up>	На один экран вверх

Таблица 2.15 *(окончание)*

Сочетания клавиш	Действие
<Ctrl>+<Page Down>	На один экран вправо
<Ctrl>+<Page Up>	На один экран влево

Перемещение в режиме таблицы по полям в длинных записях можно осуществлять с помощью поля со списком **Перейти к полю** (Go to Field) панели инструментов **Формат (режим таблицы)** (Formatting (Datasheet)). Чтобы отобразить на экране эту панель инструментов, выполните следующее:

1. В главном меню выберите команду **Вид | Панели инструментов** (View | Toolbars).
2. Установите флажок для панели **Формат (режим таблицы)** (Formatting (Datasheet)) (рис. 2.37).

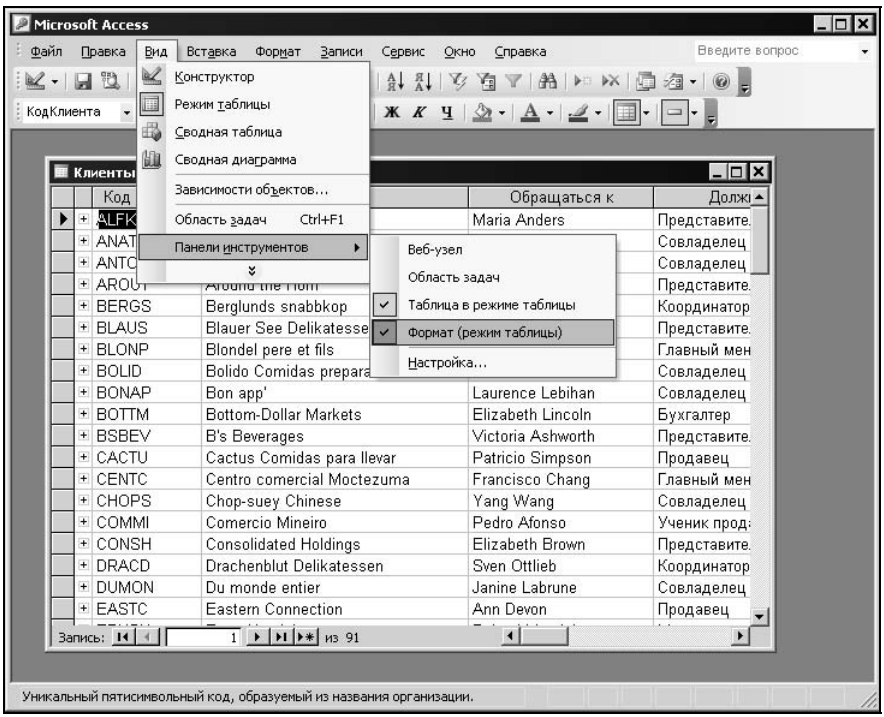


Рис. 2.37. Выбор панели инструментов **Формат (режим таблицы)**

Поле со списком **Переход к полю** (Go to Field) расположено слева на панели инструментов **Формат (режим таблицы)** (Formatting (Datasheet)). Для перехода на нужное поле в текущей записи необходимо выбрать в раскрывающемся списке его имя (рис. 2.38). Остальные элементы панели форматирования можно использовать для изменения шрифта, цвета, линий сетки в текущей таблице.

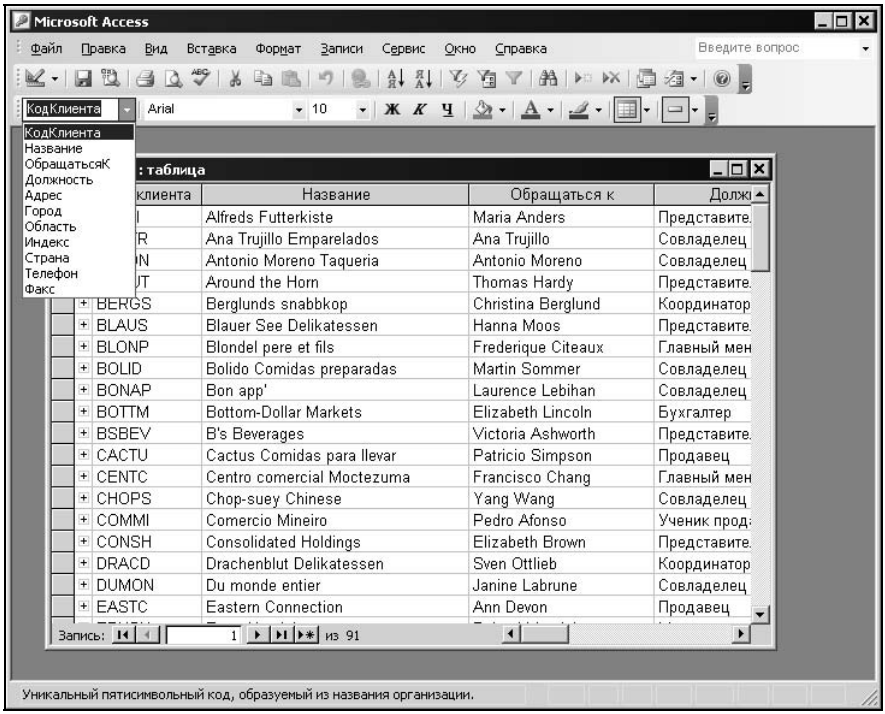


Рис. 2.38. Список полей на панели инструментов **Формат (режим таблицы)**

Работа с записями

Рассмотрим работу с записями в режиме Таблицы. Существуют три операции с записями: добавление, изменение и удаление. Чтобы добавить в таблицу новую запись, необходимо нажать кнопку **Новая запись** (New Record) на панели инструментов **Таблица** (Table Datasheet), а затем ввести данные в поля новой записи. Возможно также использовать один из следующих вариантов добавления записи в таблицу:

- нажать кнопку добавления новой записи >* в нижней части окна таблицы;
- переместить указатель на запись, отмеченную в области выделения звездочкой;
- выбрать в главном меню команду **Вставка | Новая запись** (Insert | New Record);
- щелкнуть правой кнопкой мыши на области выделения и из контекстного меню выбрать команду **Новая запись** (New Record).

Замечание

Microsoft Access сохраняет изменения в полях текущей записи только при переходе к другой записи.

Чтобы изменить существующую запись, достаточно поместить указатель мыши на поле, данные в котором необходимо изменить. После этого ввести новые данные и

нажать клавишу <Tab> для перехода на следующее поле. Окончательно данные будут изменены только после перехода на следующую или любую другую запись. До тех пор, пока работа выполняется над одной записью, все изменения можно отменить, нажав клавишу <Esc>. При необходимости отменить изменения, произведенные в текущем поле, можно также воспользоваться клавишей <Esc>.

Замечание

Если нужно внести изменения сразу в несколько записей, например увеличить цены в прайс-листе на некоторый коэффициент, используется запрос к базе данных специального вида — запрос на обновление *(см. гл. 8)*.

Удалить запись можно тоже несколькими способами. Сначала нужно в любом случае выделить запись, а затем выполнить одно из следующих действий:

- нажать на панели инструментов кнопку **Удалить запись** (Delete Record);
- щелкнуть правой кнопкой мыши на области выделения удаляемых записей и из контекстного меню выбрать команду **Удалить запись** (Delete Record);
- выбрать из меню **Правка** (Edit) команду **Удалить** (Delete) или **Удалить запись** (Delete Record);
- нажать клавишу <Delete>.

При удалении записей Access запрашивает подтверждение, чтобы предотвратить случайное удаление (рис. 2.39).

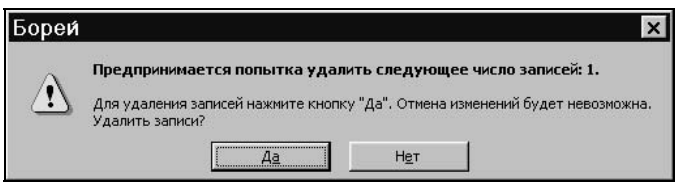


Рис. 2.39. Диалоговое окно подтверждения удаления записей

Замечание

Иногда при удалении данных требуется удалить связанные с ними данные из других таблиц. Если при определении связи между таблицами был установлен флажок **каскадное удаление связанных записей** (Cascade Delete Related Records), то это удаление выполняется автоматически, но будет выдано предупреждение, в противном случае Access не позволит удалить данные в главной таблице, если для них существуют связанные записи в подчиненной таблице.

Контроль и проверка введенных данных

Microsoft Access может осуществлять проверку данных, вводимых в поле или запись таблицы. Чтобы значения, вводимые в поля записи, автоматически контролировались, необходимо задать условия на значения. Кроме того, можно указать текст сообщения, которое должно появиться на экране, если введенные данные не соответствуют определенному условию.

В Access существует два типа условий на значение:

- условия на значение поля;
- условия на значения записи.

Очень важно, что оба типа условий вводятся в систему на уровне таблицы. Это означает, что проверка вводимых данных на соответствие этим условиям будет выполняться независимо от способа ввода данных, будь то прямое редактирование записей в режиме Таблицы или импорт (копирование) из другой таблицы или редактирование записи в форме (на странице доступа к данным). Это позволяет надежно защищать данные от ошибок ввода.

Условие на значение поля позволяет обеспечить ввод корректных данных в поле записи. Проверка введенного значения осуществляется при переходе к другому полю или другой записи. Если введенное значение не соответствует заданному условию, на экран выводится сообщение, указанное пользователем, или стандартное сообщение, если пользователь не определил свое. Чтобы определить условие на значение поля, необходимо:

1. Открыть таблицу в режиме Конструктора.
2. В верхней части окна выбрать нужное поле.
3. Открыть на панели свойств поля вкладку **Общие** (General) и выбрать свойство **Условие на значение** (Validation Rule), а затем ввести необходимое условие. Задать условие на значение можно при помощи Построителя выражений (рис. 2.40), нажав на кнопку Построителя справа. Выражения, используемые в условиях, возвращают значения **Истина** (True), и в этом случае данные корректны, или **Ложь** (False) — данные некорректны. Одним из самых простых выражений, которое может встретиться в условии, является `>0`, т. е. в поле могут вводиться только положительные числа. Для ввода более сложных выражений нужно знать правила построения выражений в Access. Эти правила, а также правила пользования Построителем выражений, подробно изложены в *разд. "Создание выражений с помощью Построителя выражений" гл. 4.*
4. Выбрать свойство **Сообщение об ошибке** (Validation Text) и ввести сообщение, которое будет выводиться на экран, если данные не соответствуют заданному условию.

Если условие на значение накладывается на поле, которое уже содержит данные, то при сохранении таблицы Microsoft Access выдаст вопрос, нужно ли применять новое условие к уже существующим данным.

Замечание

Чтобы проверить соответствие текущих данных в таблице условиям на значение, следует на заголовке таблицы в режиме Конструктора нажать правую кнопку мыши, а затем выбрать команду **Проверка условий** (Test Validation Rules).

Условия на значение записи отличаются от условий на значение поля тем, что в условиях на значение записи допускаются ссылки на разные поля. Это позволяет использовать такие условия для сравнения значений, введенных в разные поля таблицы. Условия на значение записи проверяются при сохранении записи.

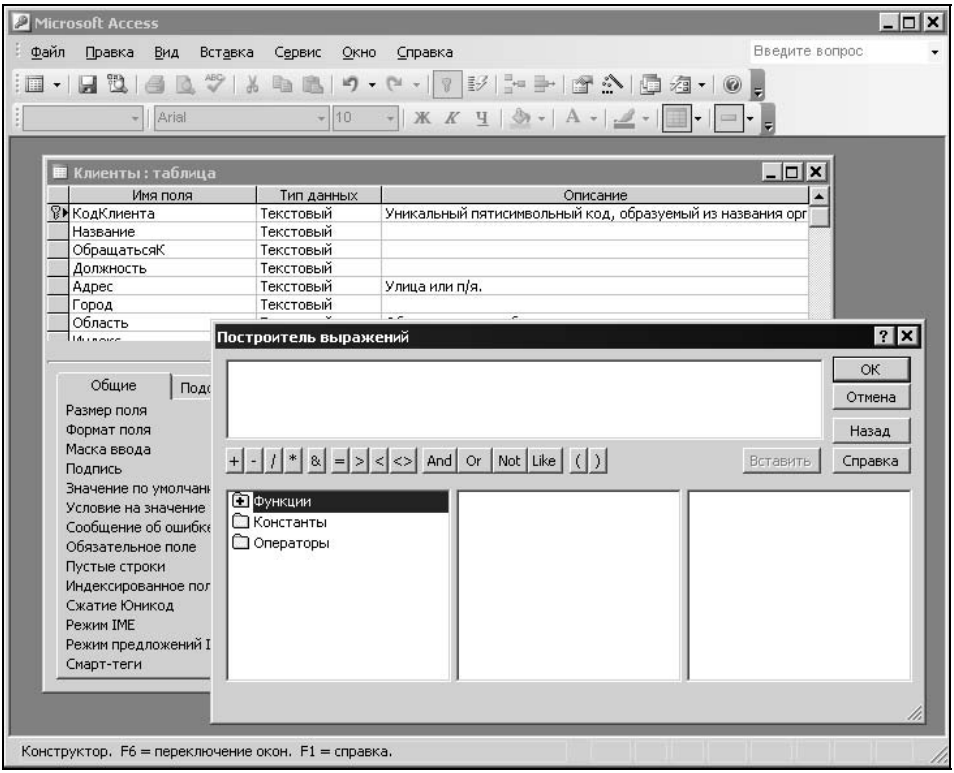


Рис. 2.40. Применение Построителя выражений при задании свойства **Условие на значение**

Чтобы определить условие на значение записи, необходимо:

1. Открыть таблицу в режиме Конструктора.
2. Нажать на панели инструментов **Конструктор таблиц** (Table Design) кнопку **Свойства** (Properties) или, щелкнув правой кнопкой мыши по заголовку таблицы, выбрать из контекстного меню команду **Свойства** (Properties). Откроется диалоговое окно **Свойства таблицы** (Table Properties).
3. Третья строка в этом окне — **Условие на значение** (Validation Rule), которое позволяет ввести выражение для условия на значение записи. Например, в таблице `Заказы` (Orders) нужно указать, что дата исполнения заказа должна быть больше, чем дата размещения. Если воспользоваться Построителем выражений, то видно, что в окне Построителя появилась папка Заказы (Orders), а рядом — список полей таблицы `Заказы` (Orders) (рис. 2.41).

   Выделите в списке значение `ДатаИсполнения` и нажмите кнопку **Вставить** (Paste) или просто дважды щелкните мышью по этому элементу. В верхнем поле окна Построителя появится имя поля в квадратных скобках. Нажмите кнопку >. Аналогичным способом выберите из списка элемент `ДатаРазмещения`. В результате в верхнем поле появится выражение `[ДатаИсполнения] > [ДатаРазмещения]`. Нажмите кнопку **OK**.

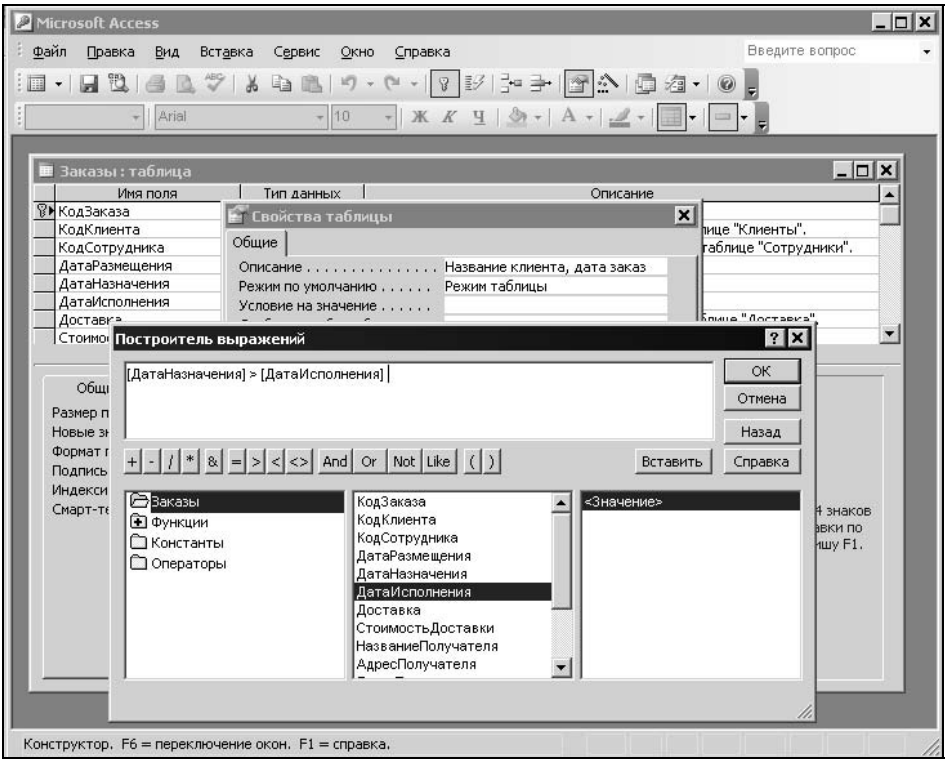


Рис. 2.41. Ввод условия с помощью Построителя выражений

Замечание

Если в выражениях используются ссылки на название полей таблицы, то они окружаются квадратными скобками.

4. Выбрать свойство **Сообщение об ошибке** (Validation Text) и ввести сообщение, которое будет выдаваться на экран, если данные не соответствуют указанному условию.

Замечание

Для таблицы можно указать только одно выражение, определяющее условия на значения. Если же необходимо наложить более одного условия, то в ячейке свойства **Условие на значение** (Validation Rule) выражения можно объединить при помощи оператора **And**.

Вне зависимости от того, заданы условия на значение поля или записи, при попытке выхода из текущего поля, после ввода данных Microsoft Access проверяет, являются ли введенные данные допустимыми для этого поля. Если значение не является допустимым, появляется предупреждающее сообщение и выход из поля блокируется. Чтобы выйти из поля, необходимо ввести правильное значение или отменить внесенные изменения, нажав клавишу <Esc>.

Значение поля может оказаться недопустимым по следующим причинам:

- значение несовместимо с типом данных поля (например, в числовых и текстовых полях или в поле даты). Если введено значение, не соответствующее типу данных поля, Access пытается преобразовать его в правильный тип данных. Однако некоторые данные нельзя преобразовать в данные другого типа. Например, нельзя хранить текстовое значение в числовом поле;
- данные нарушают условие на значение, заданное в свойстве **Условие на значение** (Validation Rule) этого поля;
- в поле необходимо ввести данные, а они не были введены.

Данные проверяются также при переходе на другую запись. Перед сохранением записи Access проверяет все условия, заданные в свойстве **Условие на значение** (Validation Rule) записи. Кроме того, если таблица, в которую вносятся изменения, имеет ключ или уникальный индекс, то проверяется, что изменяемая запись не содержит в этом поле (или полях) значений, совпадающих со значениями в других записях таблицы. Если запись нельзя сохранить, то выдается сообщение об ошибке. Тогда необходимо внести требуемые изменения или нажать кнопку **Восстановить текущее поле или запись** (Undo) на панели инструментов для отмены всех изменений, внесенных в запись.

Проверка орфографии и автозамена

Помимо проверки условий на значения и корректности ввода данных, в Microsoft Access имеется возможность проверки орфографии. Чтобы проверить орфографию в режиме таблицы, необходимо:

1. Выбрать записи, столбцы, поля или часть текста поля, в которых требуется проверить орфографию.
2. Нажать на кнопку **Орфография** (Spelling) на панели инструментов.
3. При проверке орфографии, если в тексте будет найдено неправильное слово, появится диалоговое окно **Орфография** (Spelling) (рис. 2.42). При этом предлагается несколько вариантов действий пользователя:
   - заменить это слово на слово из словаря (в приведенном примере **диссертацию**);
   - выполнить такую замену для всех вхождений этого слова в выделенной области таблицы;
   - пропустить (не заменять) данное слово или все вхождения данного слова;
   - добавить это слово в словарь, тогда оно будет считаться правильным, и в следующий раз окно с ошибкой выдаваться не будет.

Управлять режимом проверки орфографии можно с помощью параметров на вкладке **Орфография** (Spelling) диалогового окна **Параметры** (Options) (рис. 2.43).

Для настройки используйте флажки, размещенные на этой вкладке. Кроме того, можно указать словарь, который будет использован для добавления слов при проверке орфографии и язык этого словаря. По умолчанию используется словарь CUSTOM.DIC, содержащий список слов, которые считаются правильными. Вы мо-

жете добавлять слова либо в этот стандартный словарь, либо создать свой собственный словарь.

Замечание

Создать новый словарь можно только в Microsoft Office Word. Здесь можно только выбрать его из списка в поле **Добавлять слова в** (Add words to). Размер пользовательского словаря ограничен — 5000 слов или 64 Кбайт.

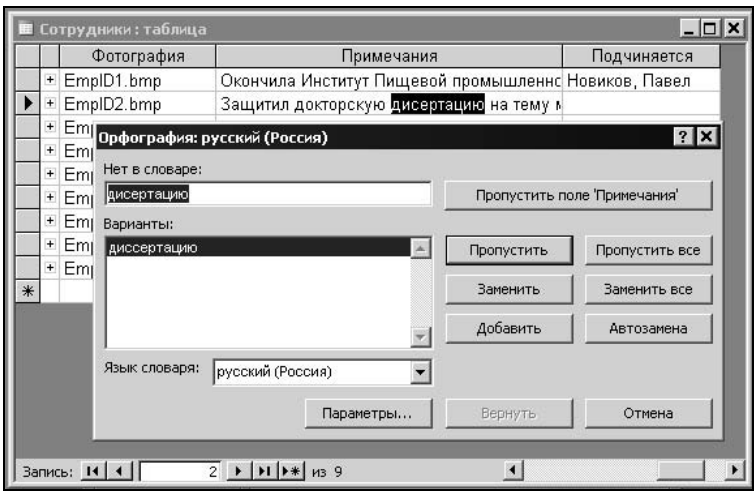


Рис. 2.42. Диалоговое окно **Орфография**

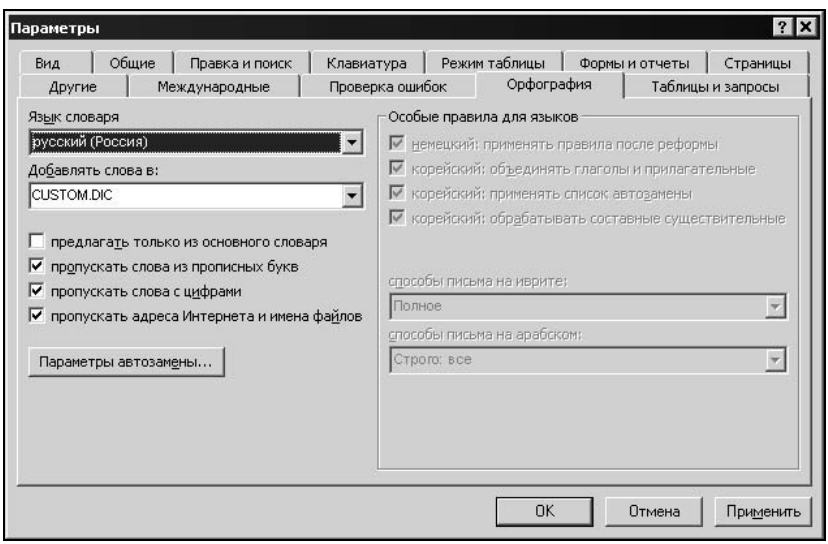


Рис. 2.43. Вкладка **Орфография** диалогового окна **Параметры**

Слово, добавляемое в словарь, не может превышать 64 символов и не должно содержать пробелов. Если вы решили использовать свой словарь для добавления тех слов, которые средства проверки орфографии должны воспринимать как правильные, то нужно, чтобы флажок **предлагать только из основного словаря** (Suggest from main dictionary only) был сброшен. Тогда будет происходить сравнение слов в выбранной области таблицы как со словами из основного словаря, так и с вашим пользовательским словарем.

Кроме проверки орфографии, в Microsoft Access возможно автоматическое исправление орфографических ошибок при вводе данных. Для этого служит режим Автозамены (AutoCorrect), который позволяет автоматически исправлять слова, при вводе которых часто допускаются ошибки, а также автоматически заменять сокращения полными значениями. То есть можно ввести аббревиатуру, а Access сама подставит расшифрованное наименование в поле. Чтобы включить режим **Автозамены** (AutoCorrect), необходимо:

1. Выбрать в меню **Сервис** (Tools) команду **Параметры автозамены** (AutoCorrect Options). Появится диалоговое окно **Автозамена** (AutoCorrect) (рис. 2.44).
2. Установить флажок **Заменять при вводе** (Replace text as you type).

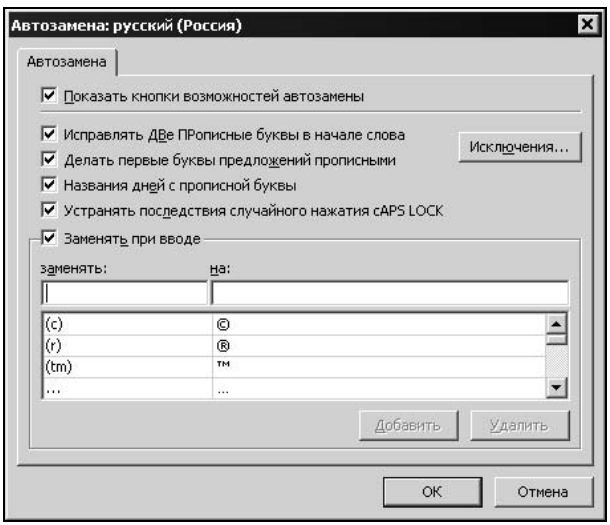


Рис. 2.44. Диалоговое окно **Автозамена**

В нижней части окна содержится список слов, которые автоматически заменяются при вводе. Этот список является пополняемым, т. е. вы можете дополнить его нужными словами. Для этого выполните следующее:

1. В поле **заменять** (Replace) введите образец, который должен заменяться автоматически.
2. В поле **на** (With) введите текст, который подставляется вместо образца.
3. Нажмите кнопку **Добавить** (Add).

Замечание

Текст, введенный до установки флажка **Заменять при вводе** (Replace text as you type) или до добавления слова в список, исправлен не будет.

Флажки в верхней части окна **Автозамена** (AutoCorrect) указывают, в каких случаях будет выполняться автозамена. Эти флажки влияют на регистр вводимых букв. Устанавливая или сбрасывая соответствующий флажок можно настраивать режим **Автозамены** (AutoCorrect). Если какой-то флажок снимать нежелательно, но тем не менее в некоторых словах вы не хотите, чтобы выполнялась соответствующая автозамена, например, не заменялась бы первая буква на прописную, можно ввести исключения. Для этого выполните следующее:

1. Нажмите кнопку **Исключения** (Exeptions) в диалоговом окне **Автозамена** (AutoCorrect).
2. Раскройте нужную вкладку в окне **Исключения при автозамене** (AutoCorrect Exeptions).
3. Добавьте исключение или выберите из списка и удалите существующее исключение.
4. Нажмите кнопку **OK**.

Если включен режим автозамены, то каждый раз, когда выполняется автозамена, рядом с заменяемым текстом появляется смарт-тег автозамены. Если такая замена не желательна, можно щелкнуть на смарт-теге и либо вернуть первоначальный текст обратно, либо вообще отменить такую автозамену, либо открыть диалоговое окно **Автозамена** (AutoCorrect), чтобы изменить параметры автозамены (рис. 2.45). Но помните, что если вы отключите автозамену, то данный образец будет удален из словаря и не будет заменяться в дальнейшем нигде.

Чтобы смарт-теги выводились при выполнении автозамены, должен быть установлен флажок **Показать кнопки возможностей автозамены** (Show AutoCorrect Options buttons) в диалоговом окне **Автозамена** (AutoCorrect) (см. рис. 2.44). Он обычно установлен по умолчанию.



Рис. 2.45. Смарт-тег автозамены

Ввод данных с помощью копирования и перемещения

Ввод данных в таблицу можно осуществлять не только непосредственно с клавиатуры, но и с помощью команд копирования и перемещения. Чтобы вырезать или скопировать текст, необходимо выделить его и нажать соответствующую кнопку на панели инструментов: **Вырезать** (Cut) или **Копировать** (Copy). При этом выделенная

информация будет перемещена или скопирована в буфер обмена. Чтобы вставить скопированную информацию, необходимо установить курсор на то место, куда ее предполагается вставить, и нажать кнопку **Вставить** (Paste) на панели инструментов.

В таблице Access можно, так же как в рабочем листе Excel, выделить диапазон ячеек (рис. 2.46). Для этого нужно сначала подвести указатель мыши к левой границе левой верхней ячейки, дождаться, когда курсор примет форму крестика, после этого нажать левую кнопку мыши и, удерживая ее, переместить указатель в нижнюю правую ячейку, потом отпустить кнопку мыши.

Если такой диапазон ячеек вставляется в середину таблицы, необходимо выделить в ней область с тем же числом полей и записей, что и у копируемых данных. Если вставка должна выполняться в конец таблицы, достаточно выделить первую пустую запись.

Если нужно выделить для копирования несколько записей целиком, то щелкните левой кнопкой мыши в области выделения первой записи и, удерживая кнопку, переместите указатель мыши до последней выделяемой записи, затем отпустите кнопку. Аналогично выделяются столбцы таблицы, только щелкать мышью нужно по области заголовка столбца.

Если записи копируются или перемещаются между двумя таблицами, то нужно следить за тем, чтобы обе таблицы имели поля одинакового типа данных, иначе при операции вставки могут возникнуть ошибки (имена полей не обязательно должны совпадать). Если запись в исходной таблице содержит больше полей, чем таблица-получатель, лишние поля не вставляются (обрезаются).

Для замены существующих записей копируемыми нужно выделить заменяемые записи и затем выполнить вставку. При этом вставляется ровно столько записей, сколько выделено заменяемых. Для добавления записей в конец таблицы нужно выделить последнюю строку в таблице, помеченную знаком (*) в области выделения, и выполнить вставку — вставляются все записи из буфера.



Рис. 2.46. Выделение диапазона ячеек в таблице

Замечание

Команда меню **Правка | Добавить из буфера** (Edit | Paste Append) вставляет все записи из буфера в конец таблицы, независимо от того, какая запись таблицы в данный момент является текущей, но вместе с названиями столбцов, которые тоже хранятся в буфере.

Вставить записи в середину таблицы нельзя.

Буфер обмена в Microsoft Office 2003 позволяет накапливать копируемые элементы, а уже потом вставлять их. В этом случае буфер обмена необходимо отобразить на экране. Для чего нужно выполнить команду меню **Правка | Буфер обмена Office** (Edit | Office Clipboard)) и тогда справа будет отображена область задач **Буфер обмена** (Clipboard) (рис. 2.47). Если одна из панелей области задач уже отображается, то можно щелкнуть на заголовке этой панели и выбрать из выпадающего списка панель **Буфер обмена** (Clipboard).

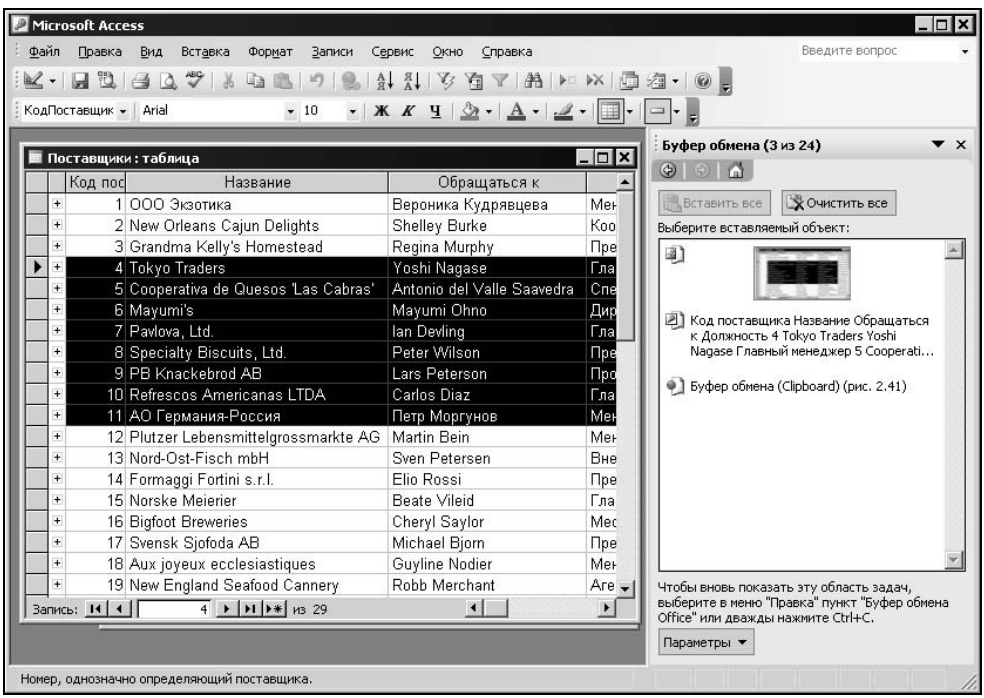


Рис. 2.47. Область задач **Буфер обмена**

При копировании или вырезании информации на панели буфера обмена будет появляться очередной значок, обозначающий скопированные данные, а также начальные буквы, если был скопирован текст, или уменьшенное изображение, если был скопирован рисунок. Общее количество блоков скопированной информации не превышает 24. При копировании 25-го элемента будет произведено удаление первого вставленного элемента. Порядок замещения блоков информации в буфере обмена организован по принципу очереди FIFO (First In First Out, "первый пришел — пер-

вый ушел"). Скопированные элементы остаются в буфере обмена Office до тех пор, пока не будут закрыты все запущенные на компьютере приложения Office.

При нажатии кнопки **Вставить** (Paste) на панели инструментов вставляется последний скопированный или вырезанный блок информации. Чтобы вставить любой из 24 последних скопированных блоков, необходимо щелкнуть левой кнопкой мыши на значке, соответствующем этому блоку. Или, если задержать указатель мыши на значке блока, справа появится кнопка выпадающего меню, из которого можно выбрать:

- **Вставить** (Paste) — вставить данный блок в текст;
- **Удалить** (Delete) — удалить данный блок из буфера обмена.

Чтобы вставить сразу все скопированные блоки, необходимо нажать кнопку **Вставить все** (Paste All) на панели инструментов **Буфер обмена** (Clipboard). Чтобы удалить из буфера обмена всю информацию, необходимо нажать кнопку **Очистить все** (Clear All) на той же панели.

Работой буфера обмена управляют пять параметров (флажков), которые можно установить или снять, выбрав их из выпадающего меню на кнопке **Параметры** (Options) в области задач **Буфер обмена** (Clipboard):

- **Автоматическое отображение буфера обмена Office** (Show Office Clipboard Automatically) — автоматически показывать буфер обмена во время копирования;
- **Открывать буфер обмена Office при двойном нажатии Ctrl+C** (Show Office Clipboard When Ctrl+C Pressed Twice) — область задач **Буфер обмена** (Clipboard) будет отображаться после двойного нажатия клавиш <Ctrl>+<C>;
- **Собирать данные без автоматического отображения буфера обмена Office** (Collect Without Show Office Clipboard) — автоматически копировать объекты в буфер обмена без его отображения;
- **Показать значок буфера обмена Office на панели задач** (Show Office Clipboard Icon on Taskbar) — поместить значок буфера обмена на панель задач операционной системы Windows;
- **Показывать состояние около панели задач при копировании** (Show Status Near Taskbar When Copying) — выдавать сообщение о состоянии буфера обмена рядом со значком на панели задач при копировании в него элемента.

Чтобы закрыть область задач **Буфер обмена** (Clipboard), необходимо нажать кнопку **Закрыть** (Close), которая находится справа на строке меню этой области.

Использование списков значений

Для удобства ввода данных в поле таблицы в Access предусмотрена возможность создания *списков подстановок*, из которых можно выбирать значения для ввода в данное поле. Списки подстановок могут быть двух типов:

- список, в котором выводятся значения из существующей таблицы или отчета;
- список постоянных значений, которые вводятся при создании поля.

Создавать новые поля со списком подстановок можно как в режиме Конструктора, так и в режиме Таблицы.

Чтобы создать поле со списком подстановок в режиме Конструктора, необходимо воспользоваться **Мастером подстановок** (Lookup Wizard), выбрав его в столбце **Тип данных** (Data Type) или определив параметры списка на вкладке свойств **Подстановка** (Lookup) *(см. разд. "Использование Мастера подстановок" данной главы)*.

Чтобы создать поле со списком подстановок в режиме Таблицы, необходимо:

1. Поместить указатель мыши в любую ячейку столбца, перед которым требуется вставить поле, и выбрать в меню **Вставка** (Insert) команду **Поле подстановок** (Lookup Column).
2. Следовать инструкциям Мастера подстановок (Lookup Wizard).

После завершения работы Мастера подстановок в таблицу будет добавлено поле со списком подстановок.

Когда таблица открыта в режиме Таблицы, поле со списком ничем не отличается от обычного текстового поля. Но когда курсор попадает в это поле, справа появляется кнопка со стрелкой. Если нажать данную кнопку, откроется список, как это показано на рис. 2.48. Для ввода значения в поле нужно просто выбрать подходящее значение в списке.



Рис. 2.48. Поле со списком в режиме Таблицы

Отображение подчиненных таблиц

В Access 2003 можно отобразить несколько связанных таблиц в окне главной таблицы. Рассмотрим возможности отображения на примере таблиц `Типы` (Categories), `Товары` (Products) и `Заказано` (Order Details) базы данных `Борей`. На рис. 2.49 изображена схема данных указанной базы. На схеме видно, что таблицы `Типы` (Categories) и `Товары` (Products) связаны отношением "один-ко-многим", т. е. одной записи в таблице `Типы` (Categories) может соответствовать множество записей таблицы `Товары` (Products). Таблица `Товары` (Products) связана с таблицей `Заказано` (Order Details) тоже отношением "один-ко-многим".

На рис. 2.50 представлена таблица `Типы` (Categories), которая имеет подчиненную таблицу `Товары` (Products). В данном случае между областью выделения и первым

столбцом таблицы существует узкий столбец, в котором для каждой записи отображается значок плюс (+). Если щелкнуть на нем левой кнопкой мыши, откроется часть подчиненной таблицы, содержащая только те записи, которые связаны с записью в главной таблице. Значок плюс при этом меняется на минус.

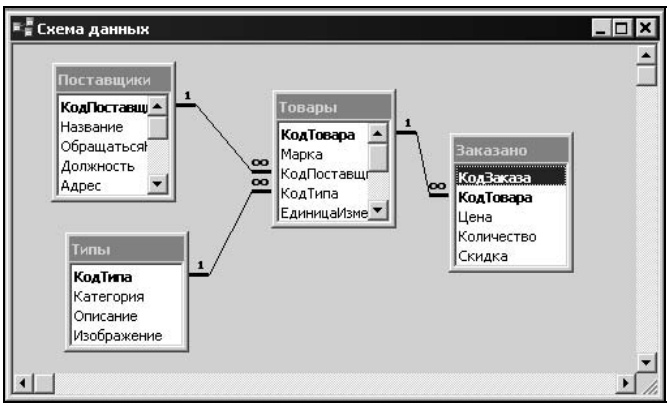


Рис. 2.49. Схема данных базы Борей

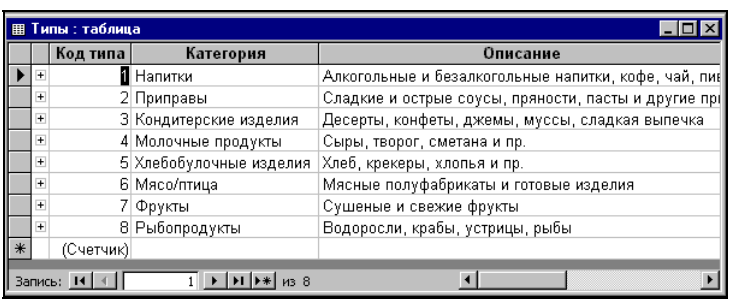
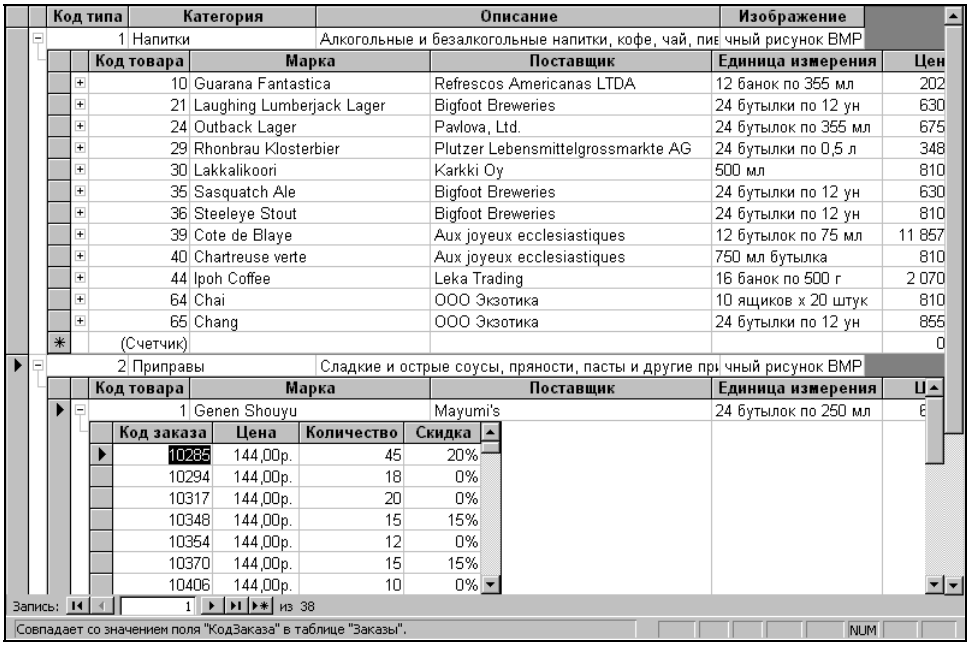


Рис. 2.50. Представление данных для таблиц, имеющих подчиненные таблицы

Изображение окна с таблицей, имеющей подчиненные таблицы, представлено на рис. 2.51. В данном случае на экран выводятся записи таблицы Типы (Categories) и для записей 1 и 2 открыты окна подчиненных таблиц. В окне подчиненной таблицы отображаются только те записи, которые соответствуют данной записи основной таблицы (Типы (Categories)). В свою очередь в окне подчиненной таблицы можно также открыть вложенное окно для следующей подчиненной таблицы. В данном случае это таблица Заказано (OrderDetails).

Чтобы развернуть все подтаблицы в таблице (или подтаблице), имеющей фокус, выполните команду меню **Формат | Подтаблица | Развернуть все** (Format | Subdatasheet | Expand All). Соответственно, чтобы свернуть все подтаблицы, нужно выполнить команду меню **Формат | Подтаблица | Свернуть все** (Format | Subdatasheet | Collapse All).

Если таблица, открытая в режиме Таблицы, не имеет подчиненных таблиц, то столбец со знаками плюс и минус отсутствует.



Код типа	Категория	Описание	Изображение
1	Напитки	Алкогольные и безалкогольные напитки, кофе, чай, пив	чный рисунок BMP

Код товара	Марка	Поставщик	Единица измерения	Цен
10	Guarana Fantastica	Refrescos Americanas LTDA	12 банок по 355 мл	202
21	Laughing Lumberjack Lager	Bigfoot Breweries	24 бутылки по 12 ун	630
24	Outback Lager	Pavlova, Ltd.	24 бутылок по 355 мл	675
29	Rhonbrau Klosterbier	Plutzer Lebensmittelgrossmarkte AG	24 бутылки по 0,5 л	348
30	Lakkalikoori	Karkki Oy	500 мл	810
35	Sasquatch Ale	Bigfoot Breweries	24 бутылки по 12 ун	630
36	Steeleye Stout	Bigfoot Breweries	24 бутылки по 12 ун	810
39	Cote de Blaye	Aux joyeux ecclesiastiques	12 бутылок по 75 мл	11 857
40	Chartreuse verte	Aux joyeux ecclesiastiques	750 мл бутылка	810
44	Ipoh Coffee	Leka Trading	16 банок по 500 г	2 070
64	Chai	ООО Экзотика	10 ящиков х 20 штук	810
65	Chang	ООО Экзотика	24 бутылки по 12 ун	855
(Счетчик)				0

2	Приправы	Сладкие и острые соусы, пряности, пасты и другие пр	чный рисунок BMP

Код товара	Марка	Поставщик	Единица измерения	Ц
1	Genen Shouyu	Mayumi's	24 бутылок по 250 мл	6

Код заказа	Цена	Количество	Скидка
10285	144,00р.	45	20%
10294	144,00р.	18	0%
10317	144,00р.	20	0%
10348	144,00р.	15	15%
10354	144,00р.	12	0%
10370	144,00р.	15	15%
10406	144,00р.	10	0%

Запись: 1 из 38

Совпадает со значением поля "КодЗаказа" в таблице "Заказы". NUM

Рис. 2.51. Отображение подчиненных таблиц

Бывают случаи, когда таблица имеет несколько подчиненных таблиц. Например, в базе данных `Библиотека` таблица `Издания` содержит данные о всех библиотечных книгах, журналах и других изданиях. Эта таблица может быть связана с таблицей `Авторы`, содержащей данные об авторах, отношением "многие-ко-многим", т. к. у книги может быть несколько авторов. С другой стороны, она связана с таблицей `Формуляры`, содержащей данные о книгах, выдаваемых читателям, также отношением "один-ко-многим". Чтобы "развязать" таблицы `Издания` и `Авторы` создается промежуточная таблица `АвторыИздания`. И вместо одной связи "многие-ко-многим" появляются две связи "один-ко-многим". Связи между этими таблицами представлены схемой данных на рис. 2.52.

Таким образом, таблица `Издания` имеет две подчиненные таблицы: `АвторыИздания` и `Формуляры`. Если открыть эту таблицу в режиме Таблицы и попробовать открыть записи в подчиненной таблице, появится диалоговое окно **Вставка подтаблицы** (Insert Subdatasheet), в котором нужно указать, какую из подчиненных таблиц отображать (рис. 2.53). Если в базе данных правильно установлены связи между таблицами, то при выборе таблицы из списка в полях **Подчиненные поля** (Link Child Fields) и **Основные поля** (Link Master Fields) автоматически отобразятся наименования связывающих полей. В противном случае, эти поля нужно задать самим, выбрав их названия из списка. После нажатия кнопки **OK** выбранная подтаблица будет отображаться под строкой в таблице `Издания`.

Если потребуется отобразить другую подчиненную таблицу, выполните команду меню **Вставка | Подтаблица** (Insert | Subdatasheet). Снова появится окно **Вставка подтаблицы** (Insert Subdatasheet) и можно выбрать из списка другую таблицу.

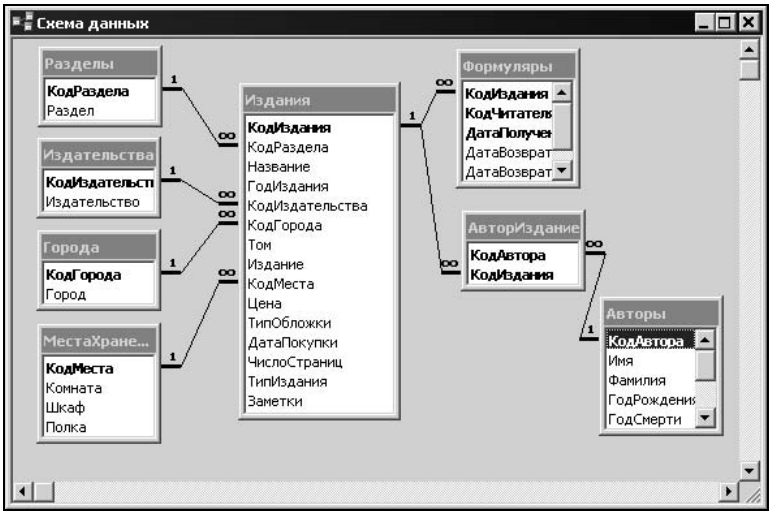


Рис. 2.52. Схема данных для базы данных `Библиотека`

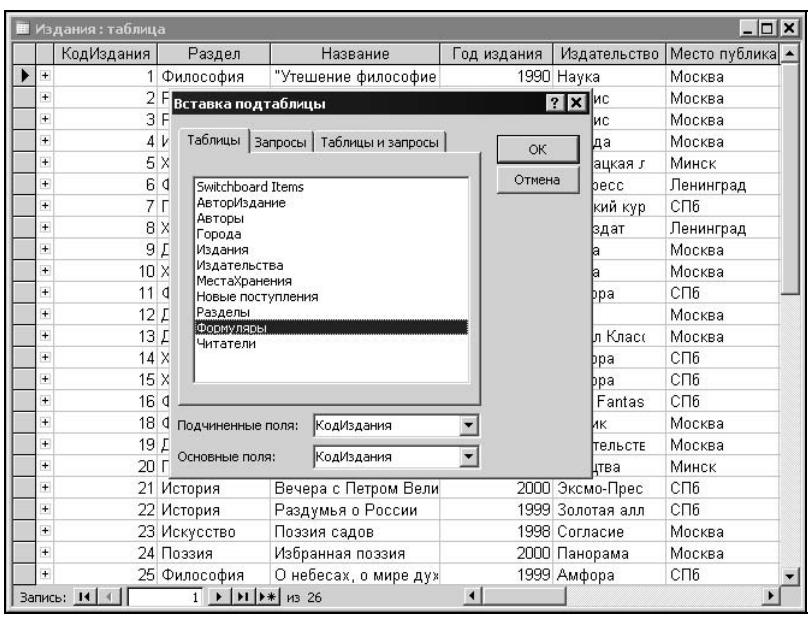


Рис. 2.53. Диалоговое окно **Вставка подтаблицы**

Если вы не хотите, чтобы подчиненная таблица отображалась в режиме таблицы основной таблицы, выполните команду меню **Формат | Подтаблица | Удалить** (Format | Subdatasheet | Remove). При этом связь между таблицами не удаляется, меняется только форма отображения.

Настройки, связанные с отображением подчиненной таблицы сохраняются в свойствах таблицы (рис. 2.54).

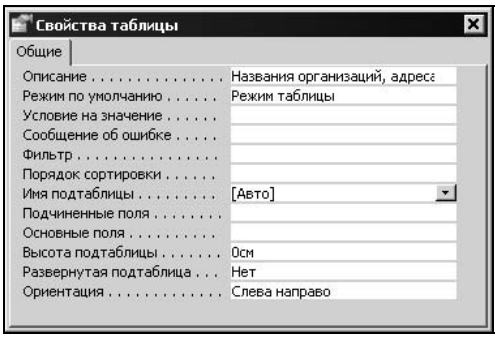


Рис. 2.54. Диалоговое окно **Свойства таблицы**

По умолчанию свойство **Имя подтаблицы** (SubdatasheetName) имеет значение **Авто** (Auto). Поэтому подчиненная таблица, как правило, создается автоматически, но в этом поле может стоять имя конкретной таблицы или запроса или слово **Нет** (None). В последнем случае подтаблица отображаться не будет. В других свойствах таблицы можно определить имена связывающих полей, высоту подчиненной таблицы и будет ли она разворачиваться полностью по умолчанию при открытии таблицы.

Изменение внешнего вида таблицы

В Microsoft Access существует понятие макета таблицы. *Макет таблицы* — это вид, в котором данные конкретной таблицы отображаются в окне в режиме Таблицы. По умолчанию формируется стандартный макет таблицы, который можно увидеть на экране, если открыть, например, таблицу `Типы` (Categories) базы данных `Борей` в режиме Таблицы. Access позволяет изменить макет таблицы. При этом можно изменить его как для всех таблиц данного приложения, так и для конкретной таблицы. Для того чтобы изменить макет для всех таблиц, необходимо изменить параметры Access. Это выполняется с помощью команды **Сервис | Параметры** (Tools | Options) *(см. гл. 14)*.

Для изменения формата отображения конкретной таблицы служит панель инструментов **Формат (режим таблицы)** (Formatting (Datasheet)) (рис. 2.55), а также команда главного меню **Формат** (Format). Для отображения панели инструментов **Формат (режим таблицы)** (Formatting (Datasheet)) необходимо в меню **Вид** (View) выбрать команду **Панели инструментов** (Toolbars) и установить флажок **Формат (режим таблицы)** (Formatting (Datasheet)) *(см. разд. "Навигация по таблице" данной главы)*.



Рис. 2.55. Панель инструментов **Формат (режим таблицы)**

Описание элементов панели форматирования и соответствующих команд меню **Формат** (Format) приведено в табл. 2.16.

Таблица 2.16. *Элементы панели инструментов* ***Формат (режим таблицы)***

Название кнопки или списка	Команда меню	Описание
Перейти на поле (Go to Field)		Поле со списком служит для быстрого перехода к нужному полю в текущей записи
Шрифт (Font)	**Формат** | **Шрифт** (Format | Font)	Поле со списком служит для выбора шрифта, который используется для отображения данных
Размер (Font Size)	**Формат** | **Шрифт** (Format | Font)	Поле со списком, в котором можно выбрать или ввести размер шрифта для данных таблицы
Полужирный (Bold)	**Формат** | **Шрифт** (Format | Font)	Применение полужирного шрифта
Курсив (Italic)	**Формат** | **Шрифт** (Format | Font)	Применение курсива
Подчеркнутый (Underline)	**Формат** | **Шрифт** (Format | Font)	Применение шрифта с подчеркиванием
Цвет заливки/фона (Fill/Back Color)	**Формат** | **Режим таблицы** (Format | Datasheet)	Позволяет изменить цвет фона таблицы. При нажатии стрелки справа от кнопки выводится окно с палитрой цветов, в котором можно выбрать цвет фона
Цвет текста (Font/Fore Color)	**Формат** | **Шрифт** (Format | Font)	Позволяет изменить цвет шрифта для данных таблицы с помощью палитры цветов
Цвет линии/границы (Line/Border Color)	**Формат** | **Режим таблицы** (Format | Datasheet)	Позволяет изменить цвет линий сетки с помощью палитры цветов
Сетка (Gridlines)	**Формат** | **Режим таблицы** (Format | Datasheet)	Позволяет выбирать разные типы сетки таблицы: вывод всех линий, только горизонтальных, только вертикальных и отсутствие линий
Оформление (Special Effect)	**Формат** | **Режим таблицы** (Format | Datasheet)	Позволяет выбрать вид оформления ячеек таблицы: обычное, приподнятое или утопленное

На рис. 2.56 приведено диалоговое окно **Формат таблицы** (Datasheet Formatting), которое открывается при выполнении команды **Формат** | **Режим таблицы** (Format | Datasheet). Элементы этого окна (списки и переключатели) позволяют изменить вид таблицы. Списки в полях группы **Вид границы и линий** (Border and Line Styles) вклю-

чают существенно больше вариантов оформления, чем выведено на кнопки панели форматирования. В поле **Образец** можно проверять результат выбора.

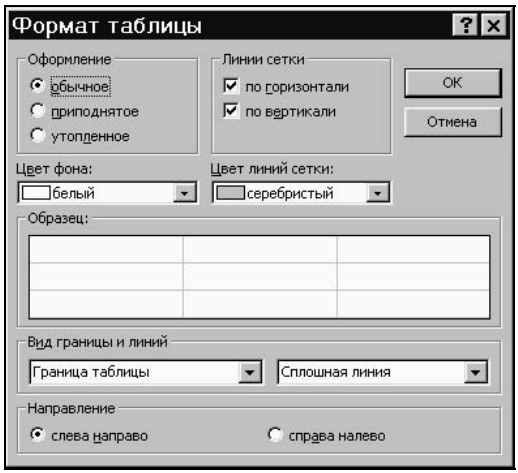


Рис. 2.56. Диалоговое окно **Формат таблицы**

Изменить высоту строки таблицы можно путем перетаскивания нижней границы строки в области выделения с помощью мыши (высота меняется сразу для всех строк в таблице). Можно сделать это и с помощью команды меню:

1. В меню **Формат** (Format) выбрать команду **Высота строки** (Row Height).
2. В окне **Высота строки** (Row Height) в поле **Высота строки** (Row Height) ввести высоту строки в пунктах (рис. 2.57).
3. При необходимости вернуться к стандартной высоте строки установить флажок **Стандартная** (Standard Height).
4. Нажать кнопку **OK**.

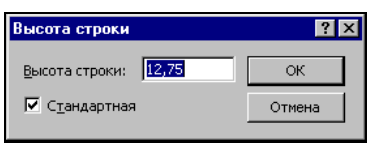


Рис. 2.57. Диалоговое окно **Высота строки**

Аналогично изменение ширины столбца возможно путем перетаскивания правой границы столбца в области заголовка столбца с помощью мыши или команды меню **Формат | Ширина столбца** (Format | Column Width).

Чтобы установить ширину столбца в соответствии с шириной данных (самая длинная строка), необходимо дважды щелкнуть левой кнопкой мыши на правой границе столбца в области заголовка. Или открыть диалоговое окно **Ширина столбца** (Column Width) с помощью команды **Формат | Ширина столбца** (Format, Column Width) и нажать кнопку **По ширине данных** (Best Fit).

Чтобы скрыть столбцы таблицы с экрана, необходимо:

1. Выделить столбцы, которые требуется скрыть.
2. Выбрать в меню **Формат** (Format) команду **Скрыть столбцы** (Hide Columns).

Или нажать правую кнопку мыши на выделенном столбце и в контекстном меню выбрать команду **Скрыть столбцы** (Hide Columns) (рис. 2.58).

Замечание

Скрыть столбец можно также, перетащив правую границу столбца к левой с помощью мыши так, чтобы они превратились в одну линию.

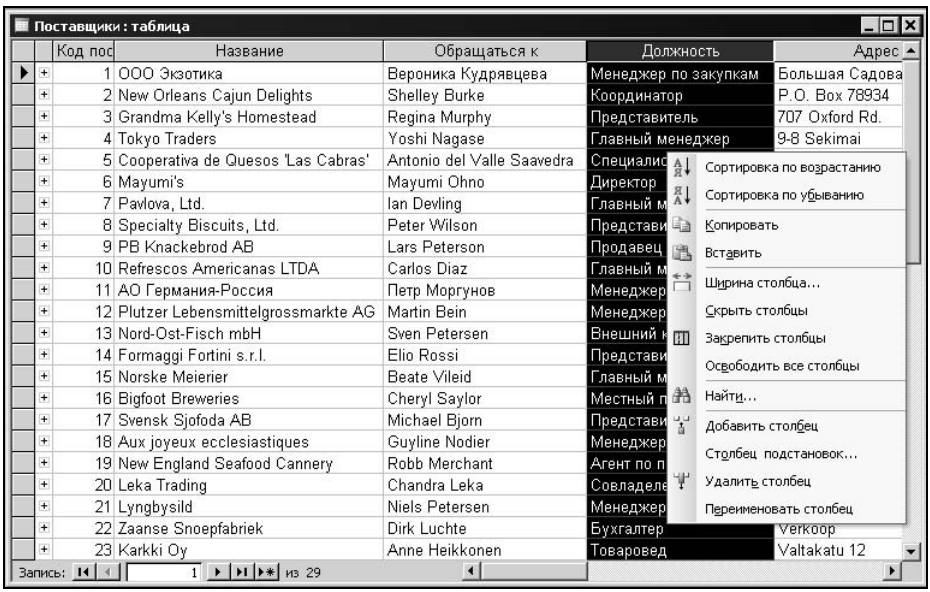


Рис. 2.58. Использование контекстного меню для скрытия столбцов

Чтобы отобразить скрытые столбцы, необходимо:

1. Выбрать в меню **Формат** (Format) команду **Отобразить столбцы** (Unhide Columns).
2. В окне **Отобразить столбцы** (Unhide Columns) установить флажки соответствующих полей для отображения столбцов или снять флажки для того, чтобы их скрыть (рис. 2.59).
3. Нажать кнопку **Закрыть** (Close).

Замечание

Отобразить столбец можно также, установив указатель мыши на левую границу столбца, следующего за скрытым, и переместив ее вправо.

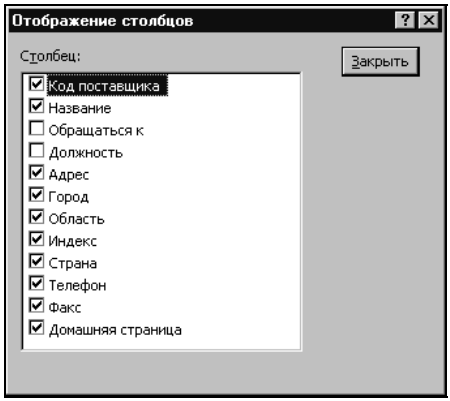


Рис. 2.59. Диалоговое окно **Отображение столбцов**

Один или несколько столбцов, расположенных в левой части объекта в режиме Таблицы, можно закрепить таким образом, что они будут оставаться на экране при горизонтальной прокрутке таблицы. Для этого необходимо:

1. Выделить закрепляемые столбцы.
2. Выбрать в меню **Формат** (Format) команду **Закрепить столбцы** (Freeze Columns) или щелкнуть правой кнопкой мыши на выделенном столбце (столбцах) и в контекстном меню выбрать команду **Закрепить столбцы** (Freeze Columns). При этом выделенные столбцы перемещаются в левую часть экрана.

Для освобождения закрепленных столбцов необходимо выбрать в меню **Формат** (Format) команду **Освободить все столбцы** (Unfreeze All Columns).

Чтобы переместить столбцы внутри таблицы в режиме Таблицы, необходимо:

1. Выделить нужный столбец (столбцы).
2. Установить указатель мыши на область выделения над выбранными столбцами.
3. Нажать левую кнопку мыши и перетащить столбцы на новое место.
4. Отпустить левую кнопку мыши.

Чтобы переместить столбцы внутри таблицы в режиме Конструктора, необходимо:

1. Открыть таблицу в режиме Конструктора.
2. Выделить строку (строки), соответствующую перемещаемым столбцам.
3. Установить указатель мыши в область выделения перемещаемых строк и нажать левую кнопку мыши.
4. Переместить строки на новое место.
5. Отпустить левую кнопку мыши.

После изменения макета таблицы в режиме Таблицы или в режиме Конструктора Microsoft Access потребует подтверждения изменения макета. В этом случае необходимо нажать кнопку **Да** (Yes), если нужно сохранить изменения, или кнопку **Нет** (No), если изменения сохранять не нужно.

Печать таблицы

Самым простым способом вывода таблицы на печать является использование кнопки **Печать** (Print) на панели инструментов **База данных** (Database). Чтобы вывести таблицу на печать таким способом, необходимо:

1. Выделить таблицу в окне базы данных.
2. Нажать кнопку **Печать** (Print) на панели инструментов **База данных** (Database.)

При этом производится печать всей таблицы. Для вывода всей таблицы можно также щелкнуть правой кнопкой мыши на значке таблицы в окне базы данных и из контекстного меню выбрать команду **Печать** (Print).

Для того чтобы иметь возможность изменять параметры печати, необходимо воспользоваться возможностями печати из режима Таблицы.

Для этого нужно:

1. Открыть таблицу в режиме Таблицы.
2. Выбрать в меню **Файл** (File) команду **Печать** (Print).
3. В окне **Печать** (Print) (рис. 2.60), в блоке **Принтер** (Printer), в поле **имя** (Name) выбрать принтер, на котором предполагается печатать таблицу.

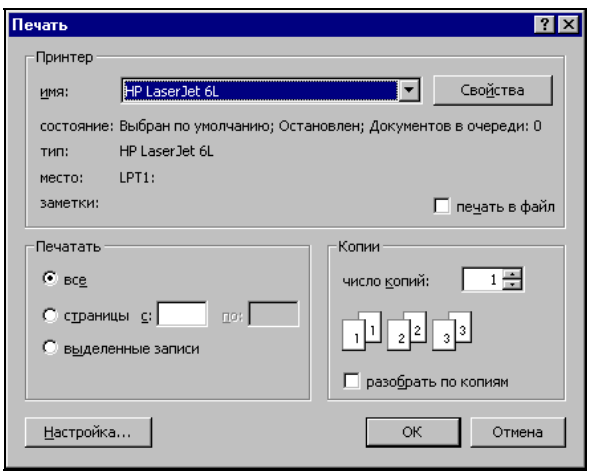


Рис. 2.60. Окно **Печать**

4. При необходимости нажать кнопку **Свойства** (Properties) для получения доступа к свойствам принтера.
5. При необходимости установить флажок **печать в файл** (Print to file) для создания файла на диске.
6. В блоке **Печатать** (Print Range) выбрать режим печати: **все** (All) — всю таблицу, **страницы с ... по...** (Pages From ... To...) — диапазон страниц, **выделенные записи** (Selected Record(s)) — только выделенные записи таблицы.

7. В блоке **Копии** (Copies) установить количество копий в поле **число копий** (Number of copies).
8. При необходимости установить флажок **разобрать по копиям** (Collate) для включения режима печати по копиям, т. е. сначала будут печататься все листы первого экземпляра, затем — все листы второго и т. д.
9. При необходимости нажать кнопку **Настройка** (Setup), которая открывает окно **Параметры страницы** (Page Setup) для установки параметров страницы.
10. Нажать кнопку **OK**.

Замечание

Кнопка **Печать** (Print) панели инструментов **Таблица в режиме таблицы** (Datasheet) также распечатает всю таблицу.

Чтобы иметь возможность настроить расположение информации на листе, нужно установить параметры страницы:

1. Открыть таблицу в режиме Таблицы.
2. Выбрать из меню **Файл** (File) команду **Параметры страницы** (Page Setup).
3. Открыть вкладку **Поля** (Margins) и установить значения отступов от края страницы, соответственно: **верхнее** (Top), **нижнее** (Bottom), **левое** (Left) и **правое** (Right) (рис. 2.61).
4. При необходимости установить флажок **Печатать заголовки** (Print Headings) для вывода на печать заголовков столбцов.
5. Открыть вкладку **Страница** (Page) и в блоке **Ориентация** (Orientation) выбрать ориентацию листа: **Книжная** (Portrait) или **Альбомная** (Landscape).

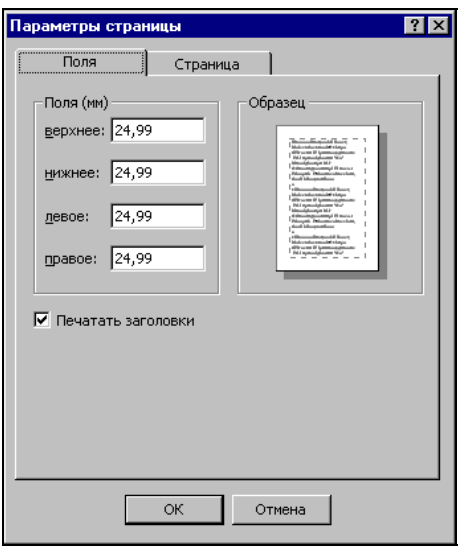


Рис. 2.61. Окно **Параметры страницы**

6. В блоке **Бумага** (Paper) выбрать в поле **Размер** (Size) размер листа, а в поле **Подача** (Source) — вариант подачи листов на принтер.
7. В блоке **Принтер для объекта <*имя таблицы*>** (Printer for <*имя таблицы*>) выбрать **Текущий принтер** (Default Printer) или **Другой принтер** (Use specific printer). В последнем случае при нажатии кнопки **Принтер** (Printer) открывается окно выбора принтера.
8. Нажать кнопку **ОК**.

Замечание

В зависимости от типа принтера вкладка **Страница** (Page) может отсутствовать. В этом случае, вместо пунктов 5, 6 и 7 следует щелкнуть мышью по кнопке **Свойства** (Properties) в окне **Печать** (Print) и выбрать необходимые значения размера бумаги, ориентации листа и т. д. из настроек самого принтера *(см. руководство по эксплуатации принтера)*.

Чтобы иметь возможность посмотреть расположение данных на страницах, прежде чем они будут отправлены на печать, можно воспользоваться функцией предварительного просмотра. Войти в режим Предварительного просмотра можно как из окна базы данных, так и из режима Таблицы. Чтобы просмотреть таблицу в режиме Предварительного просмотра из окна базы данных, необходимо выделить таблицу в списке таблиц окна базы данных, а затем выполнить одно из действий:

- нажать кнопку **Предварительный просмотр** (Print Preview) на панели инструментов **База данных** (Database);
- выбрать из меню **Файл** (File) команду **Предварительный просмотр** (Print Preview);
- щелкнуть правой кнопкой мыши на значке таблицы и из контекстного меню выбрать команду **Предварительный просмотр** (Print Preview).

Чтобы войти в окно предварительного просмотра из режима Таблицы, необходимо нажать кнопку **Предварительный просмотр** (Print Preview) на панели инструментов **Режим таблицы** (Table Datasheet) или выбрать команду **Предварительный просмотр** (Print Preview) из меню **Файл** (File).

После перехода в режим Предварительного просмотра на экране появляется окно предварительного просмотра и панель инструментов **Предварительный просмотр** (Print Preview) (рис. 2.62). В нижней части окна присутствует ряд кнопок перемещения по страницам, аналогичных кнопкам перемещения по записям в режиме Таблицы. Там же находится поле номера страницы, в котором отображается номер текущей страницы и куда можно вручную ввести номер страницы и перейти на нее.

Чтобы увеличить страницу для детального просмотра, достаточно установить указатель мыши на изображение страницы (он принимает форму лупы со знаком плюс (+)) и нажать левую кнопку мыши. Для просмотра листа целиком необходимо установить указатель мыши на страницу (лупа со знаком минус (-)) и нажать левую кнопку мыши. При просмотре страницы в режиме увеличения в окне появляются линейки прокрутки, позволяющие перемещаться по листу вверх-вниз и вправо-влево.

Действия, которые можно совершить с помощью кнопок панели инструментов **Предварительный просмотр** (Print Preview), приведены в табл. 2.17.

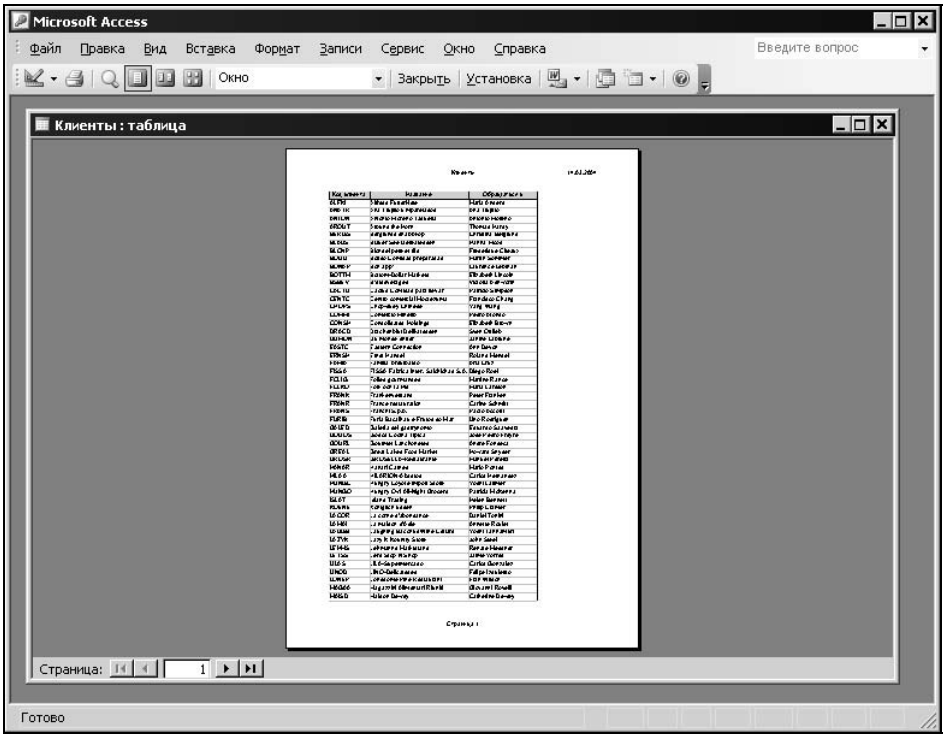


Рис. 2.62. Окно предварительного просмотра документа

Таблица 2.17. *Описание кнопок панели инструментов* ***Предварительный просмотр***

Название кнопки или списка	**Описание**	**Команда меню**
Вид (View)	Используется для переключения в режим Таблицы, в режим Конструктора, в режим Сводной таблицы и в режим Сводной диаграммы	**Вид** | **Конструктор** (View | Design) **Вид** | **Режим таблицы** (View | Datasheet) **Вид** | **Сводная таблица** (View | Pivot Table) **Вид** | **Сводная диаграмма** (View | PivotChart)
Печать (Print)	Применяется для печати таблицы	**Файл** | **Печать** (File | Print)
Масштаб (Zoom)	Позволяет изменять масштаб отображения страницы	**Вид** | **Масштаб** (View | Zoom)
Одна страница (One Page)	Позволяет просматривать одну страницу (режим по умолчанию)	**Вид** | **Число страниц** (View | Pages)

Таблица 2.17 *(окончание)*

Название кнопки или списка	Описание	Команда меню
Две страницы (Two Page)	Позволяет просматривать две страницы одновременно	**Вид	Число страниц** (View | Pages)
Несколько страниц (Multiply Pages)	Позволяет выбрать количество отображаемых одновременно страниц	**Вид	Число страниц** (View | Pages)
Масштаб (Zoom)	Позволяет ввести масштаб отображения в процентах или выбрать его из списка	**Вид	Масштаб** (View | Zoom)
Закрыть (Close)	Позволяет выйти из режима просмотра и вернуться в режим таблицы или в окно базы данных	**Файл	Закрыть** (File | Close)
Установка (Setup)	Позволяет изменять параметры страницы непосредственно из режима предварительного просмотра (см. рис. 2.62)	**Файл	Параметры страницы** (File | Page Setup)
Связи с Office (Office Links)	Позволяет экспортировать данные из таблицы в другие документы Microsoft Office	**Сервис	Связи с Office** (Tools | Office Links)
Окно базы данных (Database Window)	Активизирует окно базы данных	
Новый объект (New Object)	Позволяет создать на основе данной или любых других таблиц новую таблицу, запрос, форму, отчет, страницу, макрос, модуль	
Справка: Microsoft Office Access (Microsoft Office Access Help)	Вызов Помощника и получение справки по Access 2003	**Справка	Справка: Microsoft Office Access** (Help | Microsoft Office Access Help)

Как видно из таблицы, многие действия, выполняемые при нажатии кнопок панели инструментов, дублируются соответствующими командами меню. Эти действия являются стандартными и используются в других приложениях Microsoft Office. Характерной именно для Access является кнопка **Связи с Office** (Office Links). Если нажать стрелку справа на этой кнопке, то появляется список из трех команд:

- **Слияние в Microsoft Office Word** (Merge It with Microsoft Office Word) — обеспечивает слияние данных из таблицы Access с документом Microsoft Office Word;
- **Публикация в Microsoft Office Word** (Publish It with Microsoft Office Word) — преобразует таблицу Access в документ Microsoft Office Word;
- **Анализ в Microsoft Office Excel** (Analyze It with Microsoft Office Excel) — преобразует таблицу Access в рабочий лист Microsoft Office Excel.

Сортировка, поиск и фильтрация данных

Данные, которые хранятся в таблицах базы данных, могут быть эффективно использованы только при условии, если существуют простые и доступные средства поиска нужной информации. Таблицы могут содержать огромное число записей, и хорошая СУБД должна обеспечить возможность быстро и легко найти и отобрать среди них нужные. Access 2003 является примером того, как можно при выполнении этих операций обеспечить максимальное удобство для пользователей и в то же время эффективность обработки достаточно больших объемов данных.

Сортировка данных в таблице

Операция сортировки данных используется всегда для удобства нахождения нужной информации. Когда на экране (или на бумаге) отображается таблица, гораздо легче найти нужную строку, если эти строки упорядочены. Мы привыкли к тому, что табличные данные упорядочены по алфавиту, по дате, по увеличению или уменьшению значений в столбцах, содержащих числа. Но в разных ситуациях хотелось бы сортировать строки по разным признакам (столбцам таблицы). В идеале это должно выполняться легким движением руки. Именно так и позволяет делать Access. По умолчанию, когда таблица открывается в режиме Таблицы, она упорядочивается по значению ключевого поля. Если ключевое поле для таблицы не определено, записи выводятся в порядке их ввода в таблицу. Если нужно отсортировать записи по значению другого поля, достаточно установить курсор на любую строку соответствующего столбца и нажать одну из кнопок на панели инструментов: **Сортировка по возрастанию** (Sort Ascending) или **Сортировка по убыванию** (Sort Descending).

Другой способ выполнения этой операции: щелкнуть правой кнопкой мыши по любой строке нужного столбца и выбрать из контекстного меню (рис. 2.63) соответствующую команду.

Таким образом можно упорядочить записи по любому столбцу таблицы. Ограничения существуют только на тип данных, которые можно упорядочивать — нельзя сортировать значения в поле объекта OLE.

Чтобы правильно применять сортировку, нужно знать несколько простых правил.

- При сортировке в возрастающем порядке записи, содержащие пустые поля (с пустыми значениями), указываются в списке первыми.
- Числа, находящиеся в текстовых полях, сортируются как строки символов, а не как числовые значения. Если нужно отсортировать их в числовом порядке, все текстовые строки должны содержать одинаковое количество символов. Если строка содержит меньшее количество символов, то сначала нужно вставить незначащие нули.
- При сохранении таблицы сохраняется и порядок ее сортировки.

Замечание

Порядок сортировки зависит от настроек языка, определенных в диалоговом окне **Параметры** (Options) при создании базы данных. Чтобы изменить эту настройку, необходимо выбрать в меню **Сервис** (Tools) команду **Параметры** (Options), открыть вкладку **Общие** (General) и в поле **Порядок сортировки базы данных** (New database sort order) выбрать нужный язык *(см. гл. 14)*.

Совет

Процесс сортировки значительно ускоряется, если предварительно по данному полю был построен индекс.

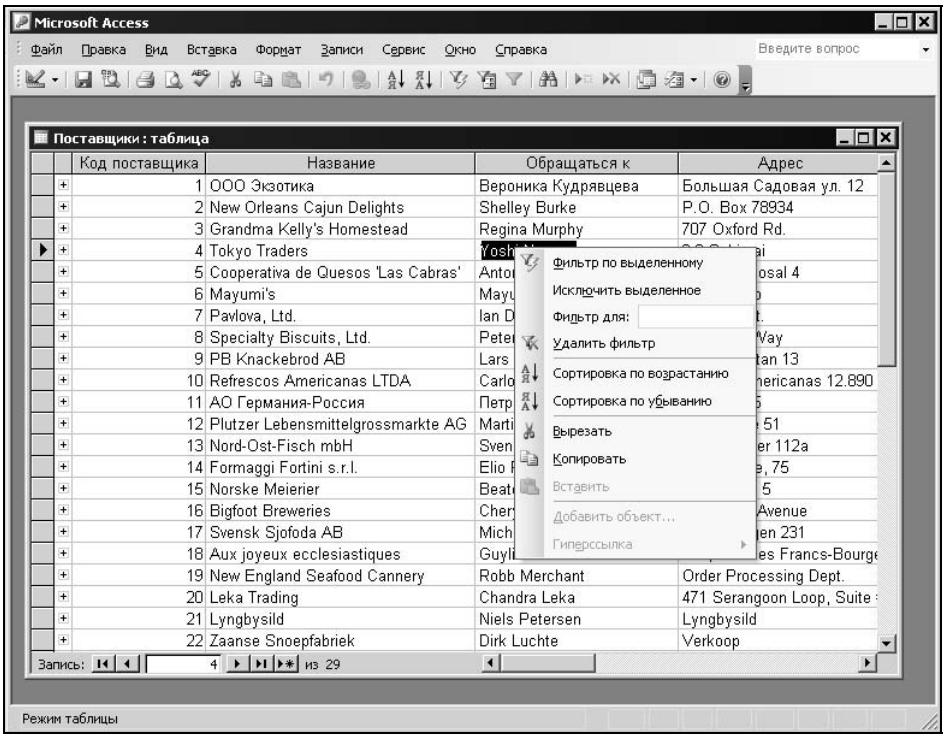


Рис. 2.63. Контекстное меню в режиме Таблицы

Если при просмотре таблицы вы изменили порядок сортировки записей в ней, а потом решили ее закрыть, Access запрашивает, нужно ли сохранить изменения макета. Если вы ответите **Да** (Yes), то новый порядок сортировки сохранится и воспроизведется, когда таблица будет открыта в следующий раз. Для того чтобы вернуться к первоначальному порядку сортировки (по значению ключевого поля), необходимо выполнить команду меню **Записи | Удалить фильтр** (Records | Remove Filter/Sort).

Иногда нужно выполнить сортировку по значению нескольких полей. Для этого переместите сначала сортируемые столбцы таким образом, чтобы они оказались, во-первых, рядом, а во-вторых, с учетом приоритетов — приоритеты устанавливаются слева направо, т. к. первыми будут сортироваться значения в крайнем левом столбце. После этого нужно выделить все столбцы и нажать, соответственно, кнопку **Сортировка по возрастанию** (Sort Ascending) или **Сортировка по убыванию** (Sort Descending) на панели инструментов.

Поиск и замена данных

В режиме Таблицы предусмотрены возможности поиска данных по образцу либо в конкретном поле, либо во всей таблице.

Чтобы найти необходимые данные, нужно:

1. Открыть таблицу в режиме Таблицы.
2. Если известно, в каком столбце нужно вести поиск, поместить курсор на любую строку этого столбца.
3. Нажать кнопку **Найти** (Find) на панели инструментов **Таблица в режиме таблицы** (Table Datasheet) и открыть вкладку **Поиск** (Find) или выполнить команду меню **Правка | Найти** (Edit | Find). Появится диалоговое окно **Поиск и замена** (Find and Replace) (рис. 2.64).

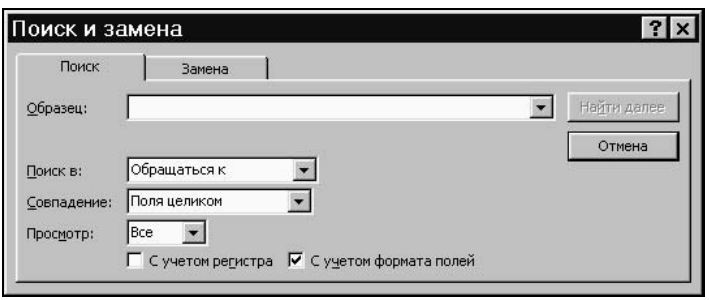


Рис. 2.64. Диалоговое окно **Поиск и замена**, вкладка **Поиск**

4. Ввести значение, которое требуется найти, в поле **Образец** (Find What). Если точное значение не известно, можно использовать подстановочные знаки.
5. Остальные параметры в окне можно изменить или оставить так, как они установлены по умолчанию:
   - значение в поле со списком **Поиск в** (Look In), определяющее место поиска, по умолчанию содержит название выбранного столбца;
   - значение в поле со списком **Совпадение** (Match) определяет один из трех вариантов совпадения образца со значением поля — **С любой частью поля** (Any Part of Field), **Поля целиком** (Whole Field), **С начала поля** (Start of Field);
   - поле со списком **Просмотр** (Search), в котором можно задать направление поиска: **Все** (All), **Вверх** (Up), **Вниз** (Down);
   - флажок **С учетом регистра** (Match Case) позволяет при поиске учитывать начертание букв — прописные или строчные;
   - флажок **С учетом формата полей** (Search Fields As Formatted) позволяет выполнять поиск данных с учетом формата отображения.

Чтобы найти первое вхождение указанного образца, нажмите кнопку **Найти далее** (Find Next).

Для поиска следующих вхождений этого значения нажимайте кнопку **Найти далее** (Find Next) до тех пор, пока не будет найдено нужное вхождение.

Для доступа к окну **Поиск и замена** (Find and Replace) можно использовать также команду меню **Правка | Найти** (Edit | Find).

Замечание

Из всех видов поиска самым медленным будет поиск по совпадению с любой частью поля, при этом не используется индекс, даже если поле, по которому выполняется поиск, проиндексировано.

Чтобы выполнить поиск по всем столбцам таблицы, нужно из списка **Поиск в** (Look In) выбрать значение **<*Имя таблицы*>: таблица**.

Замечание

Если таблица имеет подчиненную таблицу, поиск выполняется *только* в основной таблице.

Чтобы выполнить не только поиск, но и изменение данных в найденном поле, необходимо в том же диалоговом окне **Поиск и замена** (Find and Replace) раскрыть вкладку **Замена** (Replace) (рис. 2.65) и в поле **Заменить на** (Replace With) указать значение, на которое нужно заменить искомое значение. После этого нажмите кнопку **Заменить** (Replace) для замены одного значения или **Заменить все** (Replace All) для замены всех вхождений. Все остальные поля имеют тот же смысл, что и на вкладке **Поиск** (Find).

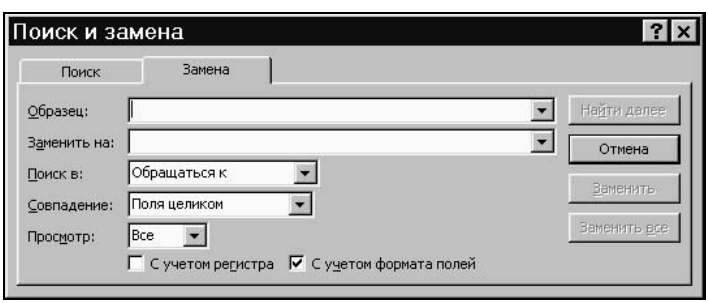


Рис. 2.65. Диалоговое окно **Поиск и замена**, вкладка **Замена**

Есть возможность установить параметры поиска/замены по умолчанию. Для этого необходимо:

1. В меню **Сервис** (Tools) выбрать команду **Параметры** (Options).
2. В окне **Параметры** (Options) открыть вкладку **Правка и поиск** (Edit/Find) (рис. 2.66).
3. В группе **Поиск и замена по умолчанию** (Default find/replace behavior) выбрать нужный тип поиска: **поле - целиком** (Fast search), **везде - любая часть** (General search), **поле - с начала** (Start of field search).
4. Нажать кнопку **ОК**.

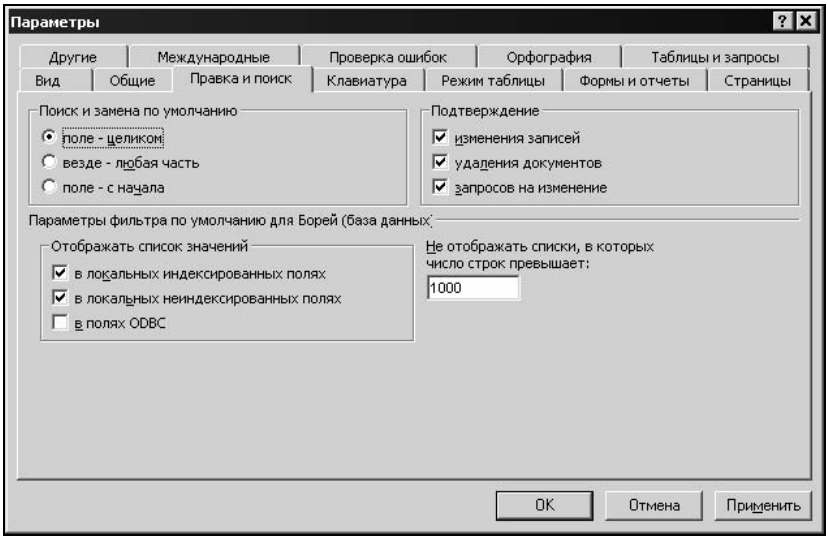


Рис. 2.66. Настройка параметров поиска по умолчанию

Фильтрация данных

В Microsoft Access предусмотрено четыре способа отбора записей с помощью фильтров: фильтр по выделенному фрагменту, обычный фильтр, поле **Фильтр для** (Filter For) и расширенный фильтр.

Фильтр по выделенному фрагменту, обычный фильтр и поле **Фильтр для** (Filter For) являются очень простыми способами отбора записей, причем самым простым является фильтр по выделенному фрагменту — он позволяет найти все записи, содержащие выделенное значение в выбранном поле. Обычный фильтр используется для отбора записей по значениям нескольких полей. Поле **Фильтр для** (Filter For) используется для ввода выражения, результат которого будет использоваться в качестве условия отбора. Для создания сложных фильтров, следует применять окно расширенного фильтра.

Набор записей, которые отобраны в процессе фильтрации, называется *результирующим набором*.

Чтобы использовать фильтр по выделенному фрагменту, необходимо:

1. В поле объекта в режиме Таблицы найти и выделить значение, которое должно содержаться в отбираемых записях.
2. Нажать кнопку **Фильтр по выделенному** (Filter by Selection) на панели инструментов **Таблица в режиме таблицы** (Table Datasheet).

Фильтры сохраняются автоматически при сохранении таблицы в свойстве **Фильтр** (Filter). Таким образом, при повторном ее открытии можно снова применить последний фильтр. Правда, для этого нужно будет нажать кнопку **Применение фильтра** (Apply Filter) на панели инструментов **Таблица в режиме таблицы** (Table Datasheet).

Совет

Фильтр позволяет отбирать записи, не содержащие заданного значения. Для этого необходимо выделить значение, нажать правую кнопку мыши и выполнить команду **Исключить выделенное** (Filter Excluding Selection).

На рис. 2.67 показан результирующий набор, полученный применением фильтра по выделенному к таблице `Заказы` (Orders) демонстрационной базы данных `Борей` (Northwind). Чтобы получить такой результат, выполните следующее:

1. Откройте таблицу `Заказы` в режиме Таблицы.
2. Выделите в столбце `Клиент` какое-нибудь значение, например `Emst Handel`.
3. Нажмите кнопку **Фильтр по выделенному** (Filter by Selection) на панели инструментов.

Заказы : таблица

Код заказа	Клиент	Сотрудник	Дата размещения	Дата назначения
10258	Ernst Handel	Белова, Мария	17-07-1996	14-08-1996
10263	Ernst Handel	Ясенева, Инна	23-07-1996	20-08-1996
10351	Ernst Handel	Белова, Мария	11-11-1996	09-12-1996
10368	Ernst Handel	Новиков, Павел	29-11-1996	27-12-1996
10382	Ernst Handel	Воронова, Дарья	13-12-1996	10-01-1997
10390	Ernst Handel	Акбаев, Иван	23-12-1996	20-01-1997
10402	Ernst Handel	Крылова, Анна	02-01-1997	13-02-1997
10403	Ernst Handel	Воронова, Дарья	03-01-1997	31-01-1997
10430	Ernst Handel	Воронова, Дарья	30-01-1997	13-02-1997
10442	Ernst Handel	Бабкина, Ольга	11-02-1997	11-03-1997
10514	Ernst Handel	Бабкина, Ольга	22-04-1997	20-05-1997
10571	Ernst Handel	Крылова, Анна	17-06-1997	29-07-1997
10595	Ernst Handel	Новиков, Павел	10-07-1997	07-08-1997
10633	Ernst Handel	Кралев, Петр	15-08-1997	12-09-1997
10667	Ernst Handel	Кралев, Петр	12-09-1997	10-10-1997
10698	Ernst Handel	Воронова, Дарья	09-10-1997	06-11-1997
10764	Ernst Handel	Акбаев, Иван	03-12-1997	31-12-1997
10771	Ernst Handel	Ясенева, Инна	10-12-1997	07-01-1998
10773	Ernst Handel	Белова, Мария	11-12-1997	08-01-1998
10776	Ernst Handel	Белова, Мария	15-12-1997	12-01-1998

Запись: 1 из 30 (Фильтр)

Рис. 2.67. Результирующий набор записей после применения фильтра по выделенному

Чтобы отменить фильтр, необходимо нажать кнопку **Удалить фильтр** (Remove Filter) на панели инструментов **Режим таблицы** (Table Datasheet) или воспользоваться одноименной командой контекстного меню.

Теперь откройте таблицу `Заказы` в режиме Конструктора и выведите окно **Свойства таблицы** (Table Properties), нажав кнопку **Свойства** (Properties) на панели инструментов. Значение свойства **Фильтр** (Filter) представляет собой последний из примененных фильтров (рис. 2.68).

Замечание

Сохранить можно только один фильтр, каждый новый фильтр заменяет предыдущий.

Кнопки **Применение фильтра** (Apply Filter) и **Удалить фильтр** (Remove Filter) являются по сути одной кнопкой, состояние (нажата, отжата) и подпись которой меня-

ются в зависимости от того, установлен или нет фильтр для таблицы. Еще одним признаком, по которому можно узнать, что в таблице отображаются отфильтрованные записи, является слово **ФЛТР** (FLTR) в строке состояния окна приложения.

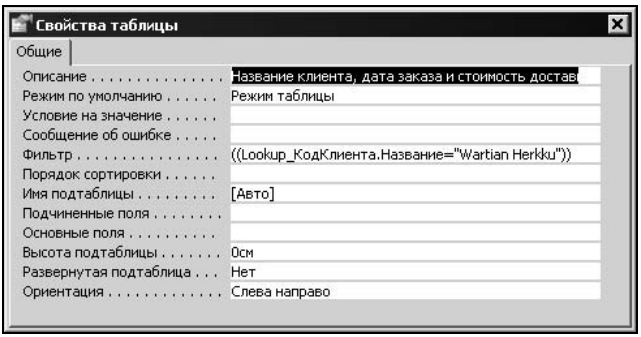


Рис. 2.68. Диалоговое окно **Свойства таблицы** с сохраненным фильтром

Для того чтобы установить фильтр по выделенному фрагменту, можно воспользоваться также командой контекстного меню **Фильтр по выделенному** (Filter by Selection) или выбрать из меню **Записи** (Records) команду **Фильтр | Фильтр по выделенному** (Filter | Filter by Selection).

Похожей возможностью фильтрации записей обладает поле **Фильтр для** (Filter For). Его можно использовать в том случае, когда не удается быстро найти образец отбора в столбце таблицы и его нужно просто ввести.

Чтобы использовать возможности поля **Фильтр для** (Filter For), необходимо:

1. Открыть таблицу в режиме Таблицы.
2. Щелкнуть правой кнопкой мыши на поле, для которого нужно указать условие отбора, а затем ввести значение условия отбора в поле **Фильтр для** (Filter For) в контекстном меню (рис. 2.69).
3. Чтобы применить фильтр и закрыть контекстное меню, нужно нажать клавишу <Enter>, а чтобы применить фильтр и оставить контекстное меню открытым — клавишу <Tab>. При этом можно вводить в поле **Фильтр для** (Filter For) новые значения и по нажатию клавиши <Tab> обновлять содержимое выборки.

Замечание

В поле **Фильтр для** (Filter For) можно указывать не только конкретные значения, но и выражения, требующие вычислений.

Чтобы применить обычный фильтр, необходимо:

1. Открыть таблицу в режиме Таблицы (например, откройте таблицу `Заказы` (Orders)).
2. Нажать кнопку **Изменить фильтр** (Filter by Form) на панели инструментов **Таблица в режиме таблицы** (Table Datasheet). Появится форма **фильтр** (Filter by form) — специальное окно для изменения фильтра (рис. 2.70).

Форма содержит линейку полей таблицы. В любое из этих полей можно ввести или выбрать из списка значение, которое и будет являться условием отбора. Если условия ввести в несколько полей, они будут объединяться с помощью логического оператора И. Для того чтобы объединить условия по ИЛИ, нужно раскрыть другую вкладку формы, щелкнув по ярлычку **ИЛИ** в нижней части формы.

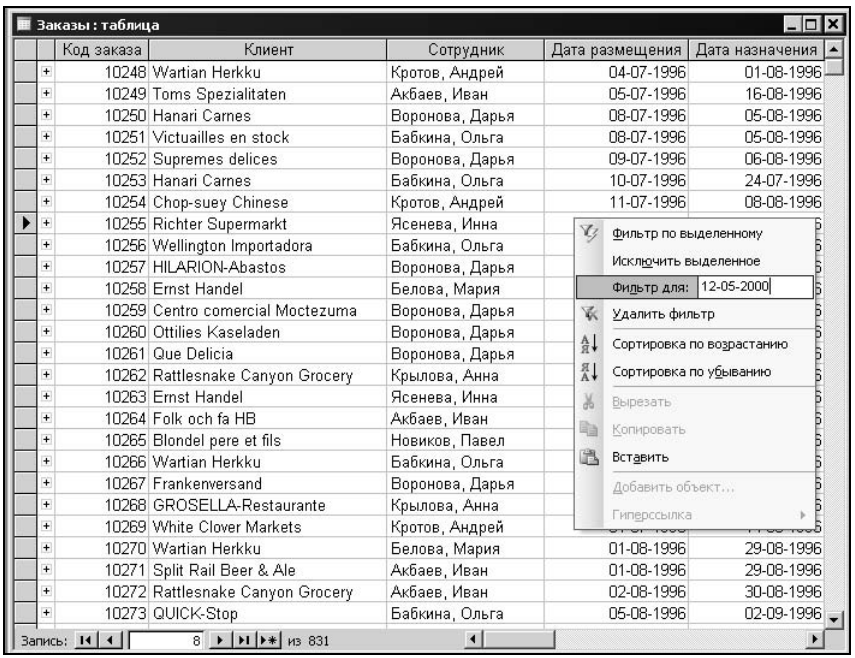


Рис. 2.69. Поле **Фильтр для** в контекстном меню



Рис. 2.70. Окно изменения фильтра

3. Выберите, например, значение `Ernst Handel` из списка в поле `Клиент` (Customer) и дату `1.01.98` в поле `Дата размещения`. Добавьте значок > перед датой. При этих условиях Access будет отбирать все заказы для Ernst Handel, размещенные после 1 января 1998 года.

4. Щелкните мышью по ярлыку **ИЛИ** и раскройте вторую вкладку.
5. Выберите из списка в поле `Клиент` (Customer) значение `Alfreds Futterkiste`, а в поле `Дата размещения` снова введите значение `> 1.01.98`. Теперь будут отбираться заказы двух клиентов: Ernst Handel и Alfreds Futterkiste, размещенные после 1 января 1998 года.
6. Нажмите кнопку **Применение фильтра** (Apply Filter) на панели инструментов.

Результирующий набор записей показан на рис. 2.71.



Рис. 2.71. Результирующий набор записей после применения обычного фильтра

Замечание

При вводе условий отбора в поля формы фильтра можно использовать любые выражения, которые возможны в Access *(см. разд. "Выражения в Access" гл. 4)*.

Условия отбора записей, которые вводятся в поля формы **фильтр** (Filter by Form), можно сохранить в базе данных в виде запроса. Для этого необходимо при открытой форме **фильтр** (Filter by Form) нажать кнопку **Сохранить как запрос** (Save As Query) на панели инструментов **Фильтр** (Filter). Затем в диалоговом окне **Сохранение в виде запроса** (Save As Query) ввести имя запроса и нажать кнопку **ОК**. Когда потребуется повторить установку такого фильтра, нужно опять же при открытой форме **фильтр** (Filter by Form) нажать кнопку **Загрузить из запроса** (Load from Query) на панели инструментов.

Создание расширенного фильтра, как более сложного, будет рассматриваться в *гл. 4*.

Отобранные с помощью фильтра данные можно копировать, экспортировать и рассылать. Копирование данных из выборки осуществляется так же, как копирование данных в таблице.

Экспорт данных из выборки позволяет сохранить их в отдельном файле. Чтобы экспортировать данные, необходимо:

1. Выбрать в меню **Файл** (File) команду **Экспорт** (Export).
2. В окне **Экспорт объекта: Таблица <*имя таблицы*> в** (Export Table <*имя таблицы*> to) выбрать папку для экспорта файла.

3. В поле **Имя файла** (File Name) ввести имя файла, в который осуществляется экспорт.
4. В поле **Тип файла** (Save as Type) выбрать тип файла, в который осуществляется экспорт.
5. Нажать кнопку **Сохранить все** (Export All).

В отличие от обычного экспорта данных, рассылка позволяет сразу отправить результаты выборки по электронной почте разным адресатам для дальнейшего ознакомления с материалами, их анализа и т. п. Чтобы отослать результаты выборки, необходимо:

1. Выбрать в меню **Файл** (File) команду **Отправить** (Send To).
2. Из раскрывающегося меню выбрать **Почтовый клиент** (Mail Recipient (as Attachment)).
3. В окне **Послать** (Send) выбрать тип файла.
4. Нажать кнопку **OK** и далее выполнить стандартную процедуру для отправки почтового сообщения в своем почтовом клиенте.

ГЛАВА 3



Использование внешних данных

При работе с любым приложением обработки данных всегда актуален вопрос, как использовать те данные, которые уже были накоплены раньше другими программными средствами и, следовательно, имеют другой формат.

Access 2003 позволяет решить эту проблему стандартным способом — путем *импорта* существующей таблицы базы данных, рабочего листа электронной таблицы или текстового файла, созданных приложениями MS-DOS или Windows, во внутренний формат базы данных Access (MDB). Естественно, что Access 2003 может также *экспортировать* данные из таблиц базы данных формата MDB в любой формат, из которого можно импортировать данные.

Однако Access является в этом смысле уникальной системой, т. к. она имеет еще один способ использования данных, которые хранятся в других форматах. Система позволяет присоединять таблицы из баз данных других форматов к базе данных Access и работать с ними в исходном формате. После создания связи базы данных с внешней таблицей присоединенную таблицу можно просматривать, изменять ее содержимое, т. е. работать с ней как с внутренней таблицей базы данных Access. При этом другие пользователи могут использовать файл таблицы в своих приложениях.

Помимо файлов баз данных, Access может работать непосредственно с файлами электронных таблиц, текстовыми файлами, документами HTML, адресными книгами или импортировать данные из этих файлов и документов XML.

Для этого используются либо встроенные драйверы ISAM (Index-Sequential Access Method, индексно-последовательный метод доступа), либо драйверы ODBC (Open Database Connectivity, открытый доступ к данным), либо поставщики данных OLE DB.

Все встроенные драйверы устанавливаются автоматически в процессе инсталляции Access 2003. Из драйверов ODBC в комплект поставки Microsoft Access входят три драйвера — Microsoft SQL Server ODBC driver (Sqlsrv32.dll), FoxPro ODBC driver (vfpodbc.dll) и Oracle ODBC driver (msorcl32.dll). Кроме того, устанавливается еще четыре провайдера OLE DB (Microsoft Jet 4.0 OLE DB Provider, Microsoft OLE DB Provider for SQL Server, OLE DB Provider for ODBC Drivers, OLE DB Provider for Oracle). Взаимодействие с внешними данными через интерфейсы OLE DB будет

подробно рассматриваться в *гл. 17* на примере взаимодействия с Microsoft SQL Server. Использование данных из остальных источников мы рассмотрим в этой главе.

Форматы данных, доступные для импорта, экспорта и связывания

Типы файлов, данные из которых могут быть импортированы в базу данных Access или которые могут быть связаны с базой данных Access, представлены в табл. 3.1. Их можно увидеть, если в меню **Файл** (File) выбрать команду **Внешние данные | Импорт** (Get External Data | Import), а затем щелкнуть мышью по расширению поля **Типы файлов** (Files of type) в диалоговом окне **Импорт** (Import). Форматы, в которые можно экспортировать данные из базы данных Access, приводятся в табл. 3.2. Их можно увидеть, если в меню **Файл** (File) выбрать команду **Экспорт** (Exort) и затем щелкнуть мышью по расширению поля **Типы файлов** (Files of type). В обоих случаях первая строчка содержит шаблоны файлов баз данных и проектов самой Microsoft Access, остальные приведены в таблицах.

Таблица 3.1. *Типы файлов, импортируемых в Access 2003*

Типы файлов	Расширения	Описание
dBASE 5	dbf	Файлы баз данных различных версий СУБД dBASE
dBASE III	dbf	
dBASE IV	dbf	
Microsoft Excel	xls	Файлы электронных таблиц Microsoft Excel
Exchange	–	Файлы адресных книг и папок Microsoft Exchange
Документы HTML (HTML Documents)	html; htm	Документы в формате HTML
Lotus 1-2-3/DOS	wj*	Файлы электронных таблиц Lotus 1-2-3
Lotus 1-2-3	wk*	
Outlook	—	Файлы адресных книг и папок Microsoft Outlook
Paradox	db	Файлы баз данных СУБД Paradox
Текстовые файлы (Text Files)	txt; csv; tab; asc	Текстовые файлы различных кодовых страниц (Windows-1281, ASCII и др.)
Windows SharePoint Services 2.0	–	Списки Microsoft Windows SharePoint Services
XML (Документы XML)	xml; xsd	Документы в формате XML
Базы данных ODBC (ODBC Databases)	–	Visual FoxPro 2.*x*, 3.0, 5.0 и 6.*x* (только импорт), таблицы SQL Server и других баз данных, для которых установлены драйверы ODBC

По умолчанию встроенные драйверы ISAM используются для доступа к файлам следующих типов: другим базам данных и проектам Access, dBASE, Paradox, электронным таблицам Microsoft Excel и Lotus 1-2-3, адресным книгам и папкам Microsoft Exchange и Outlook, спискам Microsoft Windows SharePoint Services, текстовым файлам и документам HTML и XML. И хотя для этого могут быть использованы и соответствующие драйверы ODBC, рекомендуется применять именно встроенные драйверы.

Для доступа к другим базам данных используются драйверы ODBC. Вы сможете импортировать данные из любого источника ODBC, который имеет драйвер, удовлетворяющий спецификации ODBC уровня 1.

Таблица 3.2. *Типы файлов, экспортируемых из Access 2003*

Типы файлов	Расширения	Описание
dBASE 5	dbf	Файлы баз данных различных версий СУБД dBASE
dBASE III	dbf	
dBASE IV	dbf	
Microsoft Excel 3	xls	Файлы электронных таблиц разных версий Microsoft Excel
Microsoft Excel 4	xls	
Microsoft Excel 5-7	xls	
Microsoft Excel 97-2003	xls	
Документы HTML (HTML Documents)	html; htm	Web-страницы в формате HTML
Lotus 1-2-3 WJ2	wj2	Файлы электронных таблиц Lotus 1-2-3
Lotus 1-2-3 WK1	wk1	
Lotus 1-2-3 WK3	wk3	
Paradox 3	db	Файлы баз данных разных версий СУБД Paradox
Paradox 4	db	
Paradox 5	db	
Paradox 7-8	db	
Текстовые файлы (Text Files)	txt; csv; tab; asc	Текстовые файлы различных кодовых страниц (Windows-1281, ASCII и др.)
Windows SharePoint Services 2.0	–	Списки Microsoft Windows SharePoint Services
Microsoft Active Server Pages	asp	Web-страницы в формате ASP
Microsoft IIS 1-2	htx; idc	Web-страницы в формате HTX/IDC
Формат RTF (Rich Text Format)	rtf	Текстовый файл универсального формата RTF (сохраняющий стили и разметку), который пригоден для импорта в Microsoft Word любых версий

Таблица 3.2 *(окончание)*

Типы файлов	Расширения	Описание
Microsoft Word Merge	txt	Файл данных для использования в документах слияния Microsoft Word
XML (Документы XML)	xml	Документы в формате XML
Базы данных ODBC (ODBC Databases)	–	Visual FoxPro 3.0, 5.0 и 6.*x*, таблицы SQL Server и других баз данных, для которых установлены драйверы ODBC

Использование баз данных других форматов

Системы управления реляционными базами данных различных производителей работают с файлами баз данных определенного формата. Многие из них используют специальные индексные файлы для ускорения поиска и сортировки данных в таблицах *(см. разд. "Создание и использование индексов" гл. 2)*.

Access 2003 позволяет работать не только непосредственно с таблицами баз данных других форматов, но и с индексными файлами этих таблиц.

В этом разделе будут описаны форматы баз данных, которые можно импортировать (присоединять) в Access 2003, особенности работы с таблицами баз данных других форматов, а также использование индексных файлов, преобразование типов данных и типичные ошибки, возникающие при импорте таблиц.

Форматы баз данных

В большинстве настольных СУБД каждая таблица базы данных хранится в отдельном файле, который содержит служебную информацию и сами данные в соответствующем формате. К служебной относится информация, описывающая структуру файла, например имена и типы данных полей таблицы, число записей в файле, длина файла и т. д.

В отличие от других СУБД, Microsoft Access хранит все объекты (кроме страниц доступа к данным) в одном файле с расширением mdb. Кроме того, MDB-файл включает набор системных таблиц, которые содержат всю необходимую информацию об объектах базы данных, в том числе информацию о расположении и параметрах присоединенных таблиц других форматов. По умолчанию системные таблицы не выводятся в окне базы данных Access 2003. Чтобы получить доступ к этим таблицам, необходимо изменить один из параметров Access. Для этого нужно выбрать команду **Сервис | Параметры** (Tools | Options), раскрыть вкладку **Вид** (View) диалогового окна **Параметры** (Options) и в группе **Отображать** (Show) установить флажок **Системные объекты** (System Objects). Системные таблицы Access имеют доступ только на чтение, поэтому они не могут модифицироваться пользователем.

Далее описаны форматы файлов баз данных, которые могут быть присоединены или импортированы Access 2003.

- Таблицы Access 2000 и Access 2002 (MDB) обрабатываются совершенно аналогично таблицам Access 2003. Отличие будет состоять только в том, что после названия открытого файла в заголовке окна базы данных в скобках появится: **Формат Access 2000** (Access 2000 format file). Поэтому при импорте этих таблиц не выполняется преобразование формата файла.
- Таблицы из первых версий Access (2.0, 7.0/95, 8.0/97) преобразуются при импорте в тот формат, который имеет текущий файл MDB).
- Таблицы dBASE III, III+, IV, 5 и 7 (DBF), файлы полей типа MEMO (DBT), индексные файлы dBASE III (NDX) и индексные файлы dBASE IV (MDX).
- Табличные (DB) и индексные (PX) файлы Paradox 3.*x*, 4.*x*, 5.0 и 7.0—8.0.

Замечание

Драйверы ISAM, входящие в стандартную поставку Microsoft Office в составе компонента Microsoft Data Access Component (MDAC) 2.1 и выше, обеспечивают доступ к файлам формата DBF только на чтение. Чтобы иметь доступ на чтение и запись, требуется установить Borland Database Engine (BDE) 4.*x* или выше. Существуют также драйверы от фирмы Microsoft, которые обеспечивают доступ на чтение и запись и не требуют установки BDE, но для этого необходимо обратиться в службу Microsoft Technical Support.

Формат файлов таблиц и индексов dBASE III и dBASE IV является стандартным для систем управления базами данных, работающих на персональных компьютерах. Система управления базами данных Access 2003 позволяет присоединять индексные файлы форматов NDX и MDX. При изменении присоединенной проиндексированной таблицы dBASE Access автоматически обновляет индексные файлы.

Замечание

Наличие индексных файлов NDX (dBASE III), MDX (dBASE IV) не обязательно. Тем не менее, если эти файлы доступны, лучше их использовать. В противном случае при изменении присоединенной таблицы dBASE эти изменения не отражаются в индексном файле, что может вызвать ошибки при попытке использования проиндексированных таблиц, измененных в Access, в приложениях dBASE.

Если импортируемый файл dBASE содержит поля типа MEMO, то необходимо, чтобы файлы с данными этих полей (DBT) были доступны, т. е. они должны находиться в той же папке, что и сама таблица. Если Access не удается найти файл DBT, то присоединение или импорт данных из такой таблицы вызовет ошибку. При экспорте данных из таблицы Access, содержащей поле типа MEMO, в таблицу формата dBASE автоматически создается файл DBT и ему присваивается то же имя, что и файлу таблицы.

Access позволяет присоединять табличные и индексные файлы Paradox 3.*x* и 4.*x* for MS-DOS, а также Paradox for Windows 5.0, 7.0. и 8.0. При обновлении данных в таблице Paradox из Access индексный файл должен быть доступен наряду с файлом таблицы. В процессе присоединения таблицы Access автоматически присоединяет и индексный файл (если он существует).

Все дополнительные файлы, используемые при импорте или присоединении таблиц баз данных, должны быть размещены в одной папке с таблицей. Стоит также заметить, что большинство приложений, работающих с файлами таблиц и индексов, используют для этих файлов стандартные расширения, приведенные ранее, хотя это и не является обязательным. Поэтому определение конкретного формата файла таблицы базы данных иногда может оказаться нетривиальным.

Импорт данных из баз данных FoxPro может выполняться через драйверы ODBC, поэтому нужно указывать источник данных ODBC.

Присоединение и импорт внешних таблиц

Прежде чем перейти к вопросам, связанным с импортом и присоединением таблиц, нужно понять разницу между этими возможностями Access.

При импорте таблицы из другой базы данных в текущем файле MDB создается новая таблица Access, которая имеет ту же структуру (состав и типы полей), что и исходная таблица, и содержит все данные исходной таблицы. После этого с новой таблицей можно работать, и она никак не зависит от исходной таблицы.

При присоединении внешней таблицы вы просто получаете доступ к таблице другого приложения, можете использовать данные из этой таблицы, но при этом она остается в старом приложении, в Access хранится только информация о связи. Оба приложения (и то, в котором эта таблица была создана, и приложение Access) могут работать с этой таблицей одновременно.

С присоединенной к базе данных Access внешней таблицей можно работать точно так же, как если бы она содержалась в этой базе. Единственным ограничением является то, что невозможно изменить структуру присоединенной таблицы (имена полей, типы данных и т. д.). Однако название таблицы можно изменить (при этом имя файла внешней таблицы не изменяется, изменяется только название (псевдоним) таблицы в базе данных Access), например, сделать его более понятным, т. к. ограничения на имя таблицы в Access менее жесткие, чем в других СУБД.

Кроме того, в присоединенной таблице можно в режиме Конструктора изменять ряд свойств полей (формат, надпись, маску ввода, число десятичных знаков), которые влияют только на то, как эта таблица будет отображаться в Access. При этом в "родном" приложении свойства полей не меняются.

Замечание

В первых версиях Access для операции присоединения внешней таблицы использовался термин *присоединение* (attachment). В последних версиях используется термин *связывание* (linking). Таким образом, внешние таблицы теперь становятся связанными. Мы используем в этой книге старый термин *присоединение* и, соответственно, присоединенные таблицы, чтобы не путать со *связанными* таблицами в одной базе данных, которые содержат связанные данные.

Использование присоединения внешних таблиц к базе данных Access бывает выгодным в следующих случаях:

- таблица используется совместно несколькими пользователями;
- невозможен отказ от использования другой СУБД для модификации данных;

- таблица хранится на другой машине (например, на сервере локальной сети) и содержит очень большой объем информации.

Использование импорта внешних таблиц в базе данных Access оправдано в следующих случаях:

- если при разработке приложений необходимо снять ограничения на изменение свойств внешних таблиц;
- отсутствует непосредственный доступ к файлу таблицы;
- необходимо переопределить ключевые поля таблиц Paradox. Такая ситуация может возникнуть в случае, если структура одной или нескольких таблиц серьезно противоречит правилам нормализации;
- в таблицу предполагается вводить повторяющиеся данные, а первичный ключ внешней таблицы этого не допускает.

При разработке приложения, которое должно работать с присоединенными таблицами, можно временно импортировать таблицу в базу данных, а затем на заключительной стадии разработки перейти к использованию присоединенной таблицы. При этом чтобы исключить проблемы на этапе перехода от работы с импортированной таблицей на работу с присоединенной, не изменяйте имена и типы данных полей импортированной таблицы. В противном случае может потребоваться произвести множество изменений в запросах, формах, отчетах, макросах и модулях VBA. Если разрабатываемое приложение будет использовать таблицы Paradox, то не изменяйте и поля, образующие первичный ключ. Если же предполагается использовать таблицы dBASE, необходимо убедиться в том, что используемые индексы соответствуют индексным файлам (NDX и MDX) таблицы dBASE.

Присоединение и импорт таблиц dBASE и Paradox

Теперь можно перейти непосредственно к описанию процесса присоединения и импортирования внешних таблиц.

Совет

Будьте осторожны во время присоединения файла таблицы и не работайте непосредственно с оригиналом файла до тех пор, пока не убедитесь в том, что все изменения таблицы, производимые из Access, не вызывают ошибок в исходном приложении, работающем с этой таблицей. Используйте для опытов резервную копию файла таблицы.

Чтобы присоединить или импортировать таблицу dBASE или Paradox в открытую базу данных Access 2003:

1. Откройте любую базу данных, например демонстрационную базу данных Борей (Northwind). При этом не требуется, чтобы окна объектов базы данных (форм, запросов или отчетов) были закрыты в процессе импорта или присоединения таблиц.
2. Чтобы присоединить внешнюю таблицу к базе данных, выберите команду **Файл | Внешние данные | Связь с таблицами** (File | Get External Data | Link Tables). Появится диалоговое окно **Связь** (Link), изображенное на рис. 3.1. Чтобы импортировать внешнюю таблицу в базу данных, выберите команду **Файл | Внешние дан-**

ные | Импорт (File | Get External Data | Import). Появится диалоговое окно **Импорт** (Import) (рис. 3.2). Эти диалоговые окна являются модифицированными стандартными окнами открытия документа Microsoft Office.

Замечание

Можно начать процесс присоединения или импорта, щелкнув правой кнопкой мыши на свободном поле окна базы данных и выбрав из контекстного меню команду **Связь с таблицами** (Link Tables) или **Импорт** (Import).

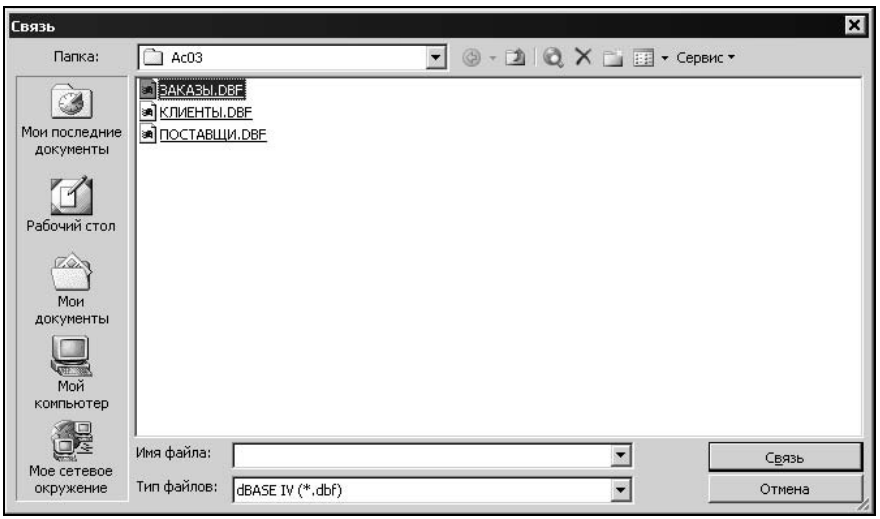


Рис. 3.1. Диалоговое окно **Связь**

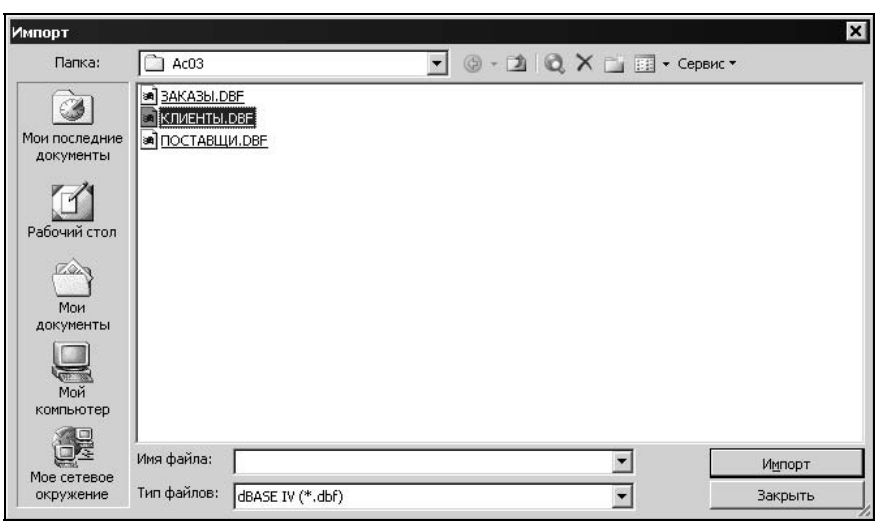


Рис. 3.2. Диалоговое окно **Импорт**

3. Раскройте папку, в которой находится файл нужной таблицы. В раскрывающемся списке **Тип файла** (Files of type) выберите тип файла, который вы хотите импортировать или присоединить к базе данных.

Замечание

Если присоединяемая таблица находится на другом компьютере локальной сети, рекомендуется в поле **Имя файла** (File name) вводить полный путь к файлу сети, например `\\Tyger\Dbase\Tables\Clients.dbf`, где `Tyger` — имя компьютера в сети. Использование вместо этого имени логического диска чревато тем, что переназначение логического диска может нарушить связь с таблицей.

4. Дважды щелкните левой кнопкой мыши по имени файла, который необходимо импортировать или присоединить к базе данных. Можно также выделить имя файла и нажать кнопку **Связь** (Link) или **Импорт** (Import). При этом предполагается, что файлы таблиц имеют стандартные расширения.
5. Если выбранная таблица зашифрована (закодирована), появится диалоговое окно, запрашивающее пароль. Введите пароль и нажмите клавишу <Enter>.
6. В отличие от предыдущих версий Access, теперь не выдается диалоговое окно **Выбор индексного файла** (Select Index Files). Если присоединяется таблица dBASE III, то чтобы присоединились индексные файлы, необходимо создать специальный файл с тем же именем, что и присоединяемый файл с расширением inf. Это конфигурационный файл, который должен содержать перечень всех индексных файлов таблицы dBASE. Пример такого файла для таблицы Клиенты.dbf приведен далее:

```
[dbase]
NDX=PRIMARYK.NDX
NDX1=Город.NDX
NDX2=Индекс.NDX
NDX3=Название.NDX
NDX4=Область.NDX
UIDX=PRIMARYK
PRIMARY=PRIMARYK
```

В первой строке описывается индексный файл для первичного ключа таблицы. Далее перечисляются все индексные файлы. Потом указывается, что индекс в файле PRIMARYK является уникальным первичным ключом таблицы.

Системы управления базами данных dBASE IV и dBASE 5 позволяют хранить несколько индексов в одном файле (MDX). Имя индексного файла обычно совпадает с именем таблицы.

Когда таблица dBASE будет импортироваться или присоединяться, Access будет искать файл INF или MDX, а также нужные файлы NDX в той же папке, что и сама таблица, и подключать индексы. Если открыть потом эту таблицу и посмотреть в режиме Конструктора ее индексы, то это будет видно (рис. 3.3). Если индексные файлы отсутствуют, то Access присоединяет таблицу без индексов и не создает первичного ключа. Тем не менее эта таблица редактируется и в ней возможно добавление и удаление записей.

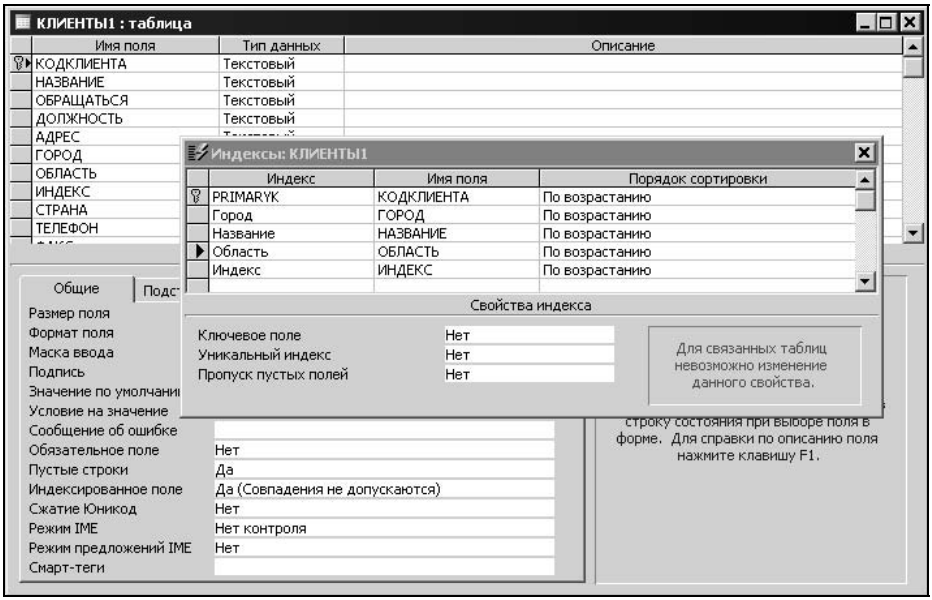


Рис. 3.3. Присоединение индексных файлов

Учтите также, что в процессе присоединения Access не проверяет индексный файл на соответствие структуре присоединяемой таблицы dBASE. Если индексный файл не соответствует таблице, его обновление производиться не будет. К сожалению, Access не выводит предупреждения об использовании индексного файла, несоответствующего таблице. Использование таблицы, для которой был неверно указан индексный файл, может вызвать ошибки в приложении dBASE. Чтобы устранить такую ошибку (если она возникнет), необходимо переиндексировать таблицу в приложении, создавшем ее. Если присоединяемая таблица использует несколько индексов и предполагается изменять содержимое таблицы средствами Access, необходимо добавить все индексы. В противном случае при изменении значения поля, по которому проиндексирована таблица, соответствующий индексный файл не будет обновлен.

Замечание

В предыдущих версиях Access конфигурационный файл INF автоматически создавался в процессе импорта или связывания таблицы на основе данных, вводимых пользователем в диалоговом окне **Выбор индексного файла** (Select Index Files). В данном случае такого окна получить не удалось, и в случае отсутствия файла INF результирующая таблица (импортированная или связанная) просто не имела индексов. Если таблица импортируется, то это не страшно — можно создать для нее индексы уже в Access. Но если таблица присоединяется, и вы собираетесь изменять в ней данные, то отсутствие индексов чревато нежелательными последствиями, как и в случае некорректных индексных файлов. По той же причине нельзя никуда перемещать файл INF и файлы индексов, т. к. Access не сможет обновлять эти файлы.

Если присоединяется файл Paradox, в котором определен первичный ключ, то Access автоматически ищет и присоединяет индексный файл (файл с расшире-

нием px), из которого и определяет первичный ключ. Если файл Paradox не имеет первичного ключа, то в отличие от файлов DBF, вы не сможете изменять данные в этой таблице из Access.

Если присоединяемая таблица имеет поля типа MEMO, Access будет искать в той же папке, что и основной файл, соответствующий файл с расширением dbt (для таблицы dBASE) или md (для таблицы Paradox). Если таких полей не окажется, будет выдано сообщение об ошибке, и операция присоединения или импорта будет прервана.

7. После того как операция присоединения (или импорта) будет завершена, появится окно с сообщением об успешном окончании операции.
8. Диалоговое окно **Связь** (или **Импорт**) остается открытым после завершения операции, позволяя, если это необходимо, присоединить или импортировать еще какие-либо внешние таблицы.
9. Чтобы закрыть диалоговое окно **Связь** (или **Импорт**), нажмите кнопку **Закрыть** (Close). Присоединенные и импортированные таблицы появятся в окне базы данных. Слева от значка присоединенной таблицы выводится стрелка, указывающая на то, что таблица хранится во внешнем файле (рис. 3.4), а сам значок определяет тип (формат) присоединенного файла.

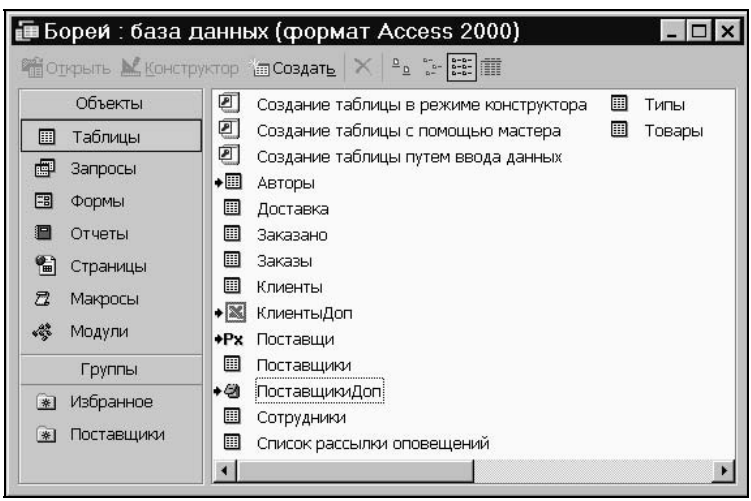


Рис. 3.4. Присоединенные таблицы в окне базы данных

10. Дважды щелкните левой кнопкой мыши по присоединенной таблице в окне базы данных. Внешняя таблица будет открыта в режиме таблицы (рис. 3.5).

Замечание

Access не умеет обновлять индексные файлы таблиц Xbase, хранящиеся в формате, отличном от формата dBASE. Например, она не может работать с индексными файлами приложений, разработанных при помощи компилятора CA-Clipper (формат NTX). Тем не менее, CA-Clipper (5+) дает возможность создавать и использовать

индексные файлы в стандартном формате dBASE (NDX). Access не может применять и дополнительные индексные файлы Paradox. Поэтому при создании приложений, данные из которых предполагается использовать и изменять в Access, необходимо работать только со стандартными форматами индексных файлов. Модификация существующего приложения с целью использования стандартных индексных файлов не должна вызвать затруднений, но производительность может существенно понизиться.

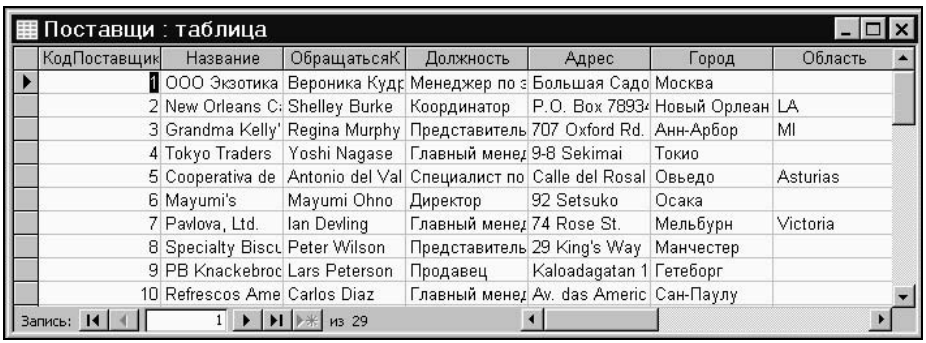


Рис. 3.5. Присоединенная таблица Paradox открыта в режиме Таблицы

При присоединении или импорте внешней таблицы Access читает заголовок файла таблицы и преобразует типы данных полей внешней таблицы в типы данных Access. Поскольку Access поддерживает больше типов данных, чем другие настольные СУБД, преобразование типов происходит правильно. В табл. 3.3 приведено соответствие между типами данных таблиц dBASE, Paradox и Access.

Таблица 3.3. *Соответствие типов данных некоторых СУБД*

dBASE III/IV	Paradox 3.0, 4.0, 5.0	Access
Character	Alphanumeric	Текстовый (Text)
Numeric, Float	Number, Money, BCD	С плавающей точкой (8 байт) (Double)
		С плавающей точкой (4 байт) (Single)
		Байт (Byte)
	Short Number	Целое (Integer)
	Long Number	Длинное целое (Long)
	AutoIncrement	Счетчик (AutoNumber)
Logical	Logical	Логический (Yes/No)
Date	Date, Time, Timestamp	Дата/время (Date/Time)
Memo	Memo, Formatted memo	Поле MEMO (Memo)
	Binary, OLE	Поле объекта OLE (OLE)

При импорте из многопользовательской базы данных необходимо иметь монопольный доступ к импортируемому файлу таблицы. При этом другие пользователи не будут иметь доступа к файлу таблицы, пока не будет закрыто диалоговое окно **Импорт** (Import).

Замечание

В Access 2003 есть возможность непосредственно открывать файлы других форматов из диалогового окна **Открытие файла базы данных** (Open). Открыв список в поле **Типы файлов** этого окна, вы увидите все те форматы, которые мы только что описывали. При выборе файла одного из указанных в списке форматов Access создает новый файл базы данных (в формате Access) в той же папке, в которой находится выбранный файл, и присоединяет (импортирует) все таблицы из выбранной базы данных.

Импорт и присоединение таблиц других форматов — достаточно сложный процесс, который не всегда завершается успешно. При обнаружении проблем в процессе импорта файла Access создает таблицу **Ошибки импорта** (Import Errors), которая отображается в списке таблиц окна базы данных. Чтобы посмотреть обнаруженные ошибки, откройте эту таблицу.

Присоединение и импорт таблиц баз данных Access

Присоединять к базе данных Access можно не только файлы баз данных других форматов, но и отдельные таблицы из других баз данных Access. Процедура присоединения таких таблиц очень проста. Начало этой процедуры аналогично описанному ранее стандартному процессу присоединения внешних таблиц.

1. Выберите команду **Файл | Внешние данные | Связь с таблицами** (File | Get External Data | Link Tables) или одноименную команду контекстного меню.
2. В появившемся диалоговом окне **Связь** (Link) раскройте папку, в которой находится файл MDB, содержащий нужную таблицу. В раскрывающемся списке **Тип файла** (Files of type) выберите **Microsoft Access (*.mdb; *.adp; *.mda; *.mde; *.ade)** (обычно тип файла устанавливается в списке по умолчанию). Выделите файл MDB и нажмите кнопку **Связь** (Link). Появится диалоговое окно **Связь с таблицами** (Link Tables), изображенное на рис. 3.6.

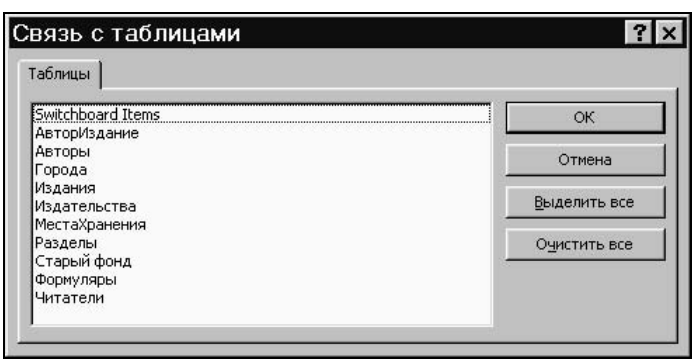


Рис. 3.6. Диалоговое окно **Связь с таблицами**

3. В списке таблиц этого окна выделите имена тех таблиц, которые нужно присоединить к текущей базе данных. Чтобы выделить все таблицы, нажмите кнопку **Выделить все** (Select All). Нажмите кнопку **OK**. Диалоговое окно **Связь с таблицами** (Link Tables) будет закрыто, а выделенные таблицы присоединены, и их названия появятся в окне базы данных.

Если присоединяемые таблицы являются связанными в своей базе данных, то эта связь между ними сохраняется и в той базе данных, к которой они были присоединены. Однако изменить свойства этой связи или удалить ее можно только в "родной" базе данных.

Импорт таблиц из других баз данных Access выполняется аналогично присоединению, только используется команда меню **Файл | Внешние данные | Импорт** (File | Get External Data | Import) и диалоговые окна **Импорт** (Import) и **Импорт объектов** (Import Objects).

Импорт и присоединение электронных таблиц

Файлы, созданные приложениями электронных таблиц, управления проектами и т. д., тоже представляют собой таблицы специального вида. Access 2003 поддерживает следующие форматы импортируемых файлов:

- файлы рабочих листов (XLS) Excel 2.*x*, 3.0, 4.0, 5.0, 7.0 (Excel 95), 8.0 (Excel 97), 9.0 (Excel 2000), 10.0 (Excel 2002), Excel 2003;
- электронные таблицы Lotus 1-2-3 в форматах WKS, WK1, WK2, WRK3, WRK4.

Этих форматов вполне достаточно, потому что практически все приложения, предназначенные для работы с электронными таблицами, позволяют экспортировать данные в один из этих форматов.

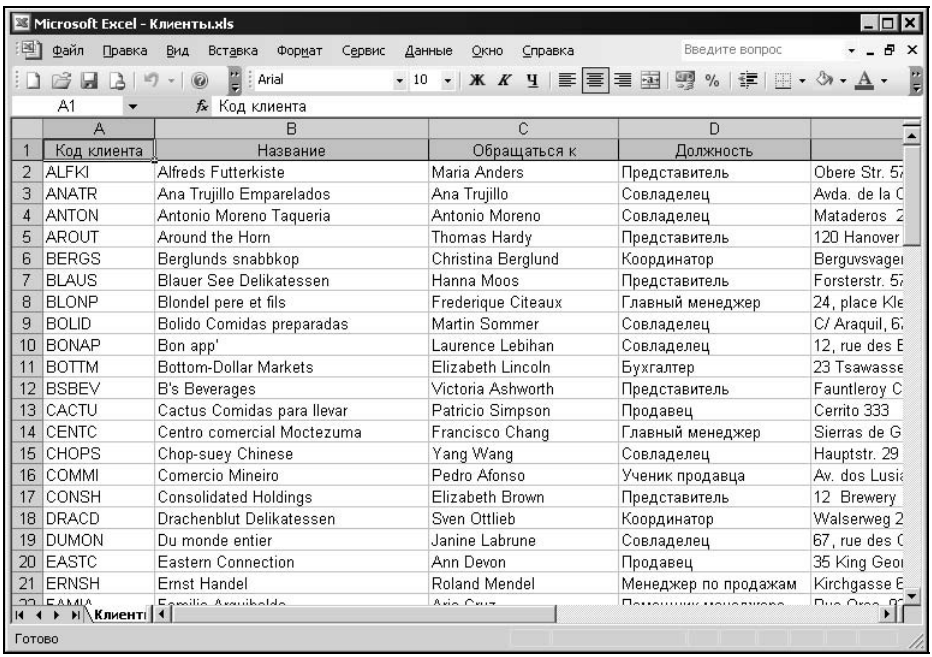
Access позволяет создавать таблицы в базе данных, импортируя их из рабочего листа Excel. При этом можно импортировать как целый рабочий лист, так и именованный диапазон ячеек этого листа. В процессе импорта может быть создана новая таблица Access, которая будет содержать все импортируемые данные, или же эти данные будут добавлены к уже существующей таблице Access. При добавлении данных в существующую таблицу заголовки столбцов рабочего листа Excel (во всяком случае, для тех столбцов, которые должны импортироваться) должны совпадать с названиями полей этой таблицы. Если таблица Excel не имеет заголовков столбцов, то порядок полей и данные в них должны соответствовать типам данных в соответствующих столбцах таблицы Access.

Замечание

Импортирование отдельного рабочего листа из рабочей книги возможно, начиная с Microsoft Excel 5.0 и выше. При необходимости импортировать только один рабочий лист из других файлов электронных таблиц, содержащих несколько листов, нужно сначала каждый лист сохранить в отдельном файле.

На рис. 3.7 показан формат представления данных на рабочем листе Excel, который подготовлен для импорта в базу данных. Обратите внимание на то, что имена всех

полей находятся в первой строке таблицы, а оставшиеся строки содержат данные. Тип данных в каждой ячейке одного столбца должен быть одинаковым, и в каждой строке должны использоваться одни и те же поля. Ни одна из строк электронной таблицы не должна иметь больше полей, чем первая строка, содержащая наименования столбцов. В таком виде таблица на рабочем листе максимально соответствует таблице Access, что позволит упростить процесс импорта.



	A	B	C	D	
1	Код клиента	Название	Обращаться к	Должность	
2	ALFKI	Alfreds Futterkiste	Maria Anders	Представитель	Obere Str. 57
3	ANATR	Ana Trujillo Emparelados	Ana Trujillo	Совладелец	Avda. de la C
4	ANTON	Antonio Moreno Taqueria	Antonio Moreno	Совладелец	Mataderos 2
5	AROUT	Around the Horn	Thomas Hardy	Представитель	120 Hanover
6	BERGS	Berglunds snabbkop	Christina Berglund	Координатор	Berguvsvagen
7	BLAUS	Blauer See Delikatessen	Hanna Moos	Представитель	Forsterstr. 57
8	BLONP	Blondel pere et fils	Frederique Citeaux	Главный менеджер	24, place Kle
9	BOLID	Bolido Comidas preparadas	Martin Sommer	Совладелец	C/ Araquil, 67
10	BONAP	Bon app'	Laurence Lebihan	Совладелец	12, rue des B
11	BOTTM	Bottom-Dollar Markets	Elizabeth Lincoln	Бухгалтер	23 Tsawasse
12	BSBEV	B's Beverages	Victoria Ashworth	Представитель	Fauntleroy C
13	CACTU	Cactus Comidas para llevar	Patricio Simpson	Продавец	Cerrito 333
14	CENTC	Centro comercial Moctezuma	Francisco Chang	Главный менеджер	Sierras de G
15	CHOPS	Chop-suey Chinese	Yang Wang	Совладелец	Hauptstr. 29
16	COMMI	Comercio Mineiro	Pedro Afonso	Ученик продавца	Av. dos Lusia
17	CONSH	Consolidated Holdings	Elizabeth Brown	Представитель	12 Brewery
18	DRACD	Drachenblut Delikatessen	Sven Ottlieb	Координатор	Walserweg 2
19	DUMON	Du monde entier	Janine Labrune	Совладелец	67, rue des C
20	EASTC	Eastern Connection	Ann Devon	Продавец	35 King Geo
21	ERNSH	Ernst Handel	Roland Mendel	Менеджер по продажам	Kirchgasse 6

Рис. 3.7. Рабочий лист Excel

Для того чтобы подготовить данные на рабочем листе для импорта, может потребоваться выполнить несколько предварительных действий.

1. Запустите Microsoft Excel и откройте рабочий лист, данные из которого нужно импортировать.
2. Добавьте, если это необходимо, заголовки столбцов над первой строкой импортируемого диапазона данных. Заголовки столбцов будут применяться в качестве имен полей, поэтому в них нельзя использовать точку (.), восклицательный знак (!) и прямоугольные скобки ([]). Нельзя использовать одно имя дважды. Если в заголовках столбцов используются запрещенные символы или один заголовок применяется несколько раз, в процессе импорта данных это вызовет сообщение об ошибке.
3. Если необходимо импортировать не все данные рабочего листа, выделите нужный диапазон ячеек, а затем выберите команду **Вставка | Имя | Присвоить** (Insert | Name | Define) и присвойте имя выделенному диапазону.
4. Сохраните рабочий лист и закройте приложение Excel.

Замечание

Если ячейки рабочего листа содержали формулы, по которым вычислялись значения, то в таблицу Access будут импортированы только вычисленные значения.

Теперь данные рабочего листа можно импортировать в таблицу Access. Для этого:

1. Запустите Access и откройте базу данных, в которую необходимо импортировать данные. Активизируйте окно **База данных** (Database), щелкнув по его заголовку левой кнопкой мыши.
2. Выберите команду **Файл | Внешние данные | Импорт** (File | Get External Data | Import). Появится диалоговое окно **Импорт**.
3. Перейдите в папку, содержащую файл рабочего листа Excel. В раскрывающемся списке **Тип файла** (File of Type) выделите элемент **Microsoft Excel (*.xls)**. Выделите имя файла и нажмите кнопку **Импорт** (Import) или просто дважды щелкните по имени файла левой кнопкой мыши. Запустится Мастер импорта электронной таблицы (Import Spreadsheet Wizard), первое диалоговое окно которого показано на рис. 3.8.
4. Если необходимо импортировать весь рабочий лист, выберите переключатель **листы** (Show Worksheets). Если же предполагается импортировать именованный диапазон рабочего листа, то выберите переключатель **именованные диапазоны** (Show Named Ranges). В списке первого диалогового окна Мастера импорта электронных таблиц будут выведены имена рабочих листов или диапазонов, соответственно. На рис. 3.8 список объектов состоит только из одного наименования `Клиенты`, т. к. импортируемая таблица имеет один рабочий лист.

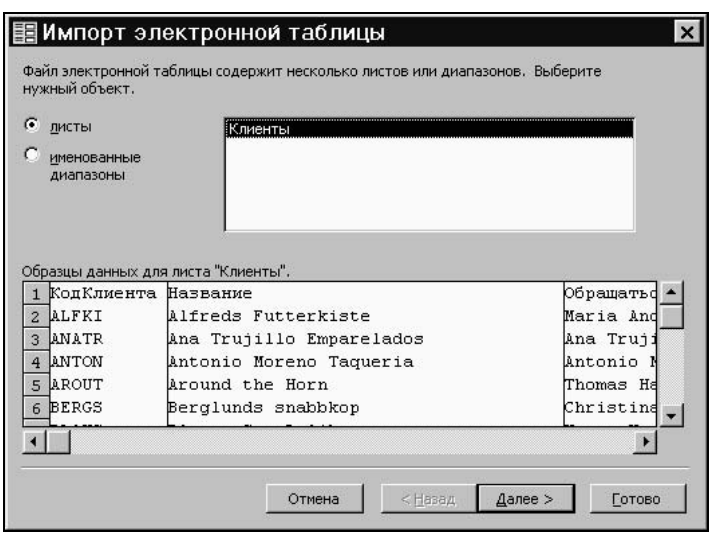


Рис. 3.8. Первое диалоговое окно Мастера импорта электронной таблицы

5. Выделите нужное имя рабочего листа или диапазона ячеек. В нижней части первого окна Мастера импорта выводится образец данных из импортируемого листа

или диапазона. Нажмите кнопку **Далее** (Next), чтобы перейти к следующему шагу Мастера. Появится диалоговое окно, изображенное на рис. 3.9.

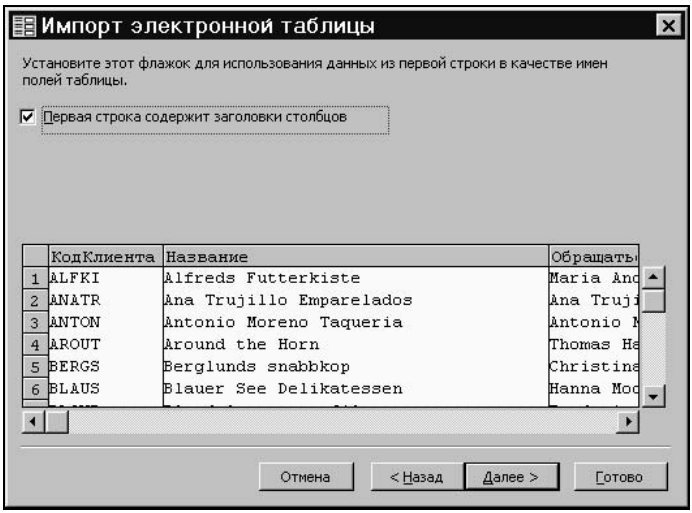


Рис. 3.9. Второе диалоговое окно Мастера импорта электронной таблицы

6. Если первая строка импортируемых данных содержит заголовки столбцов, то их можно использовать в качестве имен полей. Для этого установите флажок **Первая строка содержит заголовки столбцов** (First Row Contains Column Headings). Снова нажмите кнопку **Далее** (Next). Появится диалоговое окно, показанное на рис. 3.10.

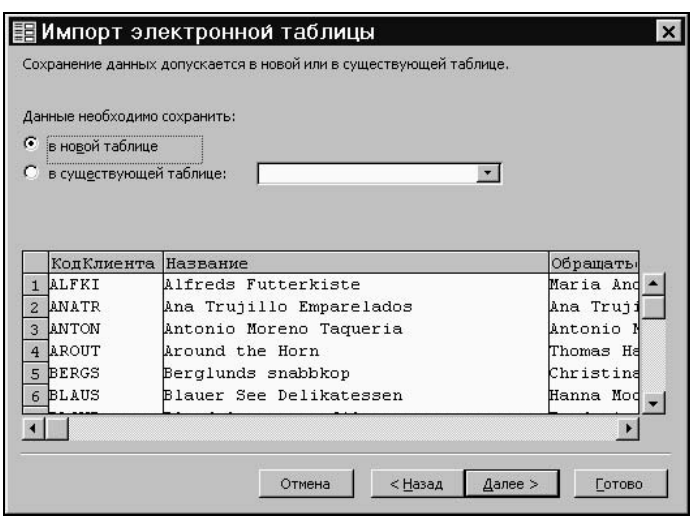


Рис. 3.10. Третье диалоговое окно Мастера импорта электронной таблицы

7. В этом диалоговом окне можно указать, где необходимо сохранить импортируемые данные — в новой таблице или в существующей. Выберите нужный переключатель и, при необходимости, нужную таблицу в списке **в существующей таблице** (In an Existing Table). Нажмите кнопку **Далее** (Next). Если вы добавляете данные к уже существующей таблице, то все промежуточные шаги будут пропущены и появится последнее диалоговое окно Мастера (см. шаг 14). Если же вы создаете новую таблицу, то появится диалоговое окно, показанное на рис. 3.11.

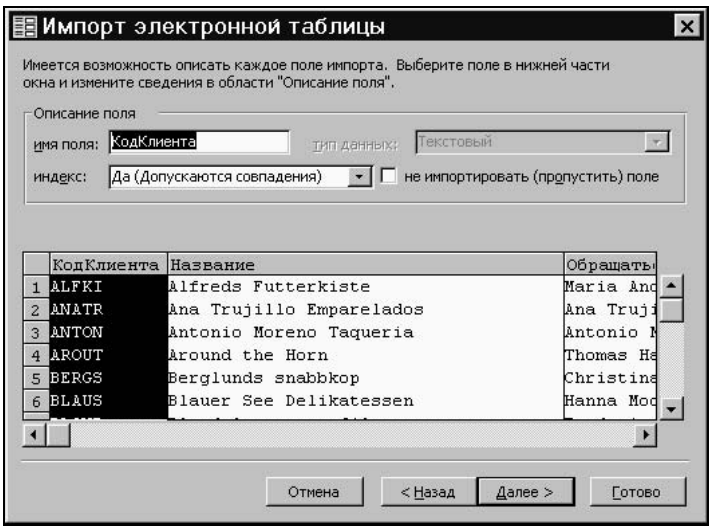


Рис. 3.11. Четвертое диалоговое окно Мастера импорта электронной таблицы

8. В этом окне вы можете выбрать те столбцы, которые нужно импортировать (это не обязательно все столбцы рабочего листа), и ввести некоторые параметры для полей будущей таблицы. Чтобы не импортировать (пропустить) какое-либо поле (столбец рабочего листа), выделите его и установите флажок **не импортировать (пропустить) поле** (Do not import field).

9. Мастер импорта электронных таблиц позволяет изменить или добавить (если они не определены в первой строке рабочего листа) имена полей таблицы, соответствующих столбцам рабочего листа. Для этого выделите столбец, щелкнув по нему левой кнопкой мыши, а затем укажите имя в текстовом поле **имя поля** (Field Name) (горизонтальная полоса прокрутки в диалоговом окне позволяет просмотреть все столбцы листа).

10. Если какое-либо поле нужно проиндексировать, выберите требуемый тип индекса в раскрывающемся списке **индекс** (Indexed).

11. Если данные рабочего листа не форматировались или были отформатированы как текст, Мастер позволяет пользователю изменить тип данных для каждого поля при помощи раскрывающегося списка **тип данных** (Data Type). На рис. 3.11 этот список недоступен, поскольку импортируемый рабочий лист содержал форматирование, которое Мастер определил автоматически.

12. Повторите шаги 9—11 для каждого столбца импортируемых данных. После того как заданы все имена и типы данных полей, определены проиндексированные поля, а также столбцы, которые нужно пропустить при импорте, нажмите кнопку **Далее**. Появится следующее диалоговое окно Мастера импорта (рис. 3.12).

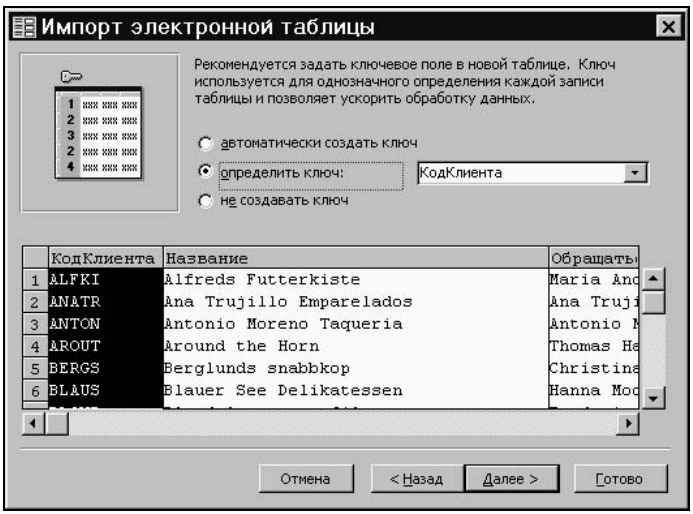


Рис. 3.12. Пятое диалоговое окно Мастера импорта электронной таблицы

13. На пятом шаге можно определить ключевое поле новой таблицы. Чтобы Access автоматически добавил к импортируемой таблице ключевое поле с типом данных **Счетчик** и присвоил уникальный номер каждой строке таблицы, выберите переключатель **автоматически создать ключ** (Let Access add Primary Key). Чтобы определить один из столбцов импортируемого листа или диапазона ячеек в качестве ключевого поля таблицы, выберите переключатель **определить ключ** (Choose my own Primary Key) и выделите в раскрывающемся списке название столбца. Если для импортируемой таблицы не требуется определение ключевого поля, выберите переключатель **не создавать ключ** (No Primary Key). Нажмите кнопку **Далее**. Появится последнее диалоговое окно.

Совет

Прежде чем определить ключевое поле на основе одного из столбцов таблицы, убедитесь в том, что ячейки этого столбца содержат уникальные значения.

14. В поле ввода **Импорт в таблицу** (Import to Table) укажите имя новой таблицы. По умолчанию Мастер использует имя импортируемого рабочего листа или диапазона ячеек в качестве имени новой таблицы. Сразу после завершения импорта данных можно вызвать Анализатор таблиц, позволяющий оптимизировать структуру таблицы, например исключать повторяющиеся данные. В этом случае необходимо установить флажок **Проанализировать таблицу после импорта данных** (I would like a wizard to analyze my table after importing the data). Нажмите кнопку **Готово** (Finish), чтобы завершить процесс импорта. После того как им-

порт данных будет завершен, появится сообщение об успешном окончании операции.

Замечание

Если имя таблицы, которое вы указали на последнем шаге Мастера импорта электронной таблицы, совпадет с именем уже существующей в базе данных таблицы, то будет выведено диалоговое окно, в котором Мастер спросит у вас, заменить ли уже существующую таблицу новой или переименовать создаваемую таблицу.

При автоматическом определении типов данных полей импортируемой таблицы Access просматривает первые 20 строк импортируемого диапазона ячеек. Например, если каждый столбец в импортируемом диапазоне содержит числовые данные, то полю, соответствующему этому столбцу, присвоится один из числовых подтипов данных (выбор подтипа зависит от параметров форматирования рабочего листа Excel). Если же в первых 20 ячейках столбца содержатся числа, а в остальных встречается текст, то Access уже не изменяет тип данных поля, а создает *таблицу ошибок импорта*, в которой каждая запись соответствует одной ошибке. После окончания процесса импорта в случае возникновения ошибок выдается сообщение об этом (рис. 3.13) и указывается имя таблицы ошибок импорта.

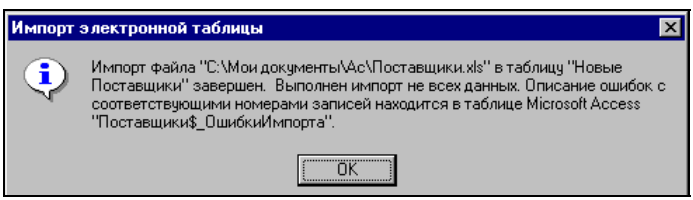


Рис. 3.13. Сообщение о возникновении ошибок импорта

Таблица ошибок импорта (рис. 3.14) позволяет найти записи, в которых возникли ошибки, и исправить их. В таблице каждая встреченная ошибка описывается, и указываются номера строк, в которых она возникла.

Наиболее частыми причинами ошибок, помимо описанной ранее ситуации, когда в поле оказались данные неправильного типа, являются:

- одна или несколько строк таблицы содержат больше полей, чем строка заголовков;

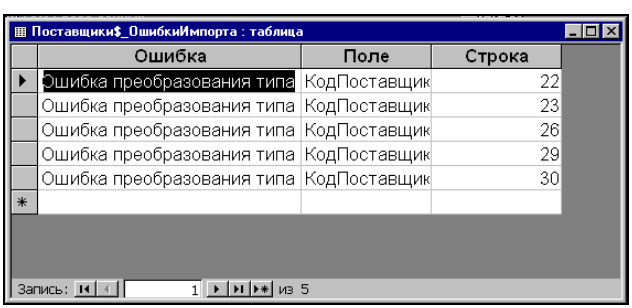


Рис. 3.14. Таблица ошибок импорта

- импортируемые строки таблицы содержат повторяющиеся значения либо в поле, которое определено в таблице Access как первичный ключ, либо для которого свойство **Индексированное поле** (Indexed) имеет значение **Да (Совпадения не допускаются)**;
- числовые данные в одном из полей превышают значение свойства **Размер поля** (Field Size), который указан для соответствующего поля в таблице Access;
- при вставке записей из электронной таблицы в существующую таблицу Access причиной ошибки может быть также несовпадение имен полей в таблице Access и заголовков столбцов в импортируемой таблице.

В большинстве случаев для исправления ошибок требуется исправить рабочий лист Excel и повторить импорт данных.

Импортированная таблица (даже если не все строки импортировались успешно) появится в окне базы данных. Чтобы проверить, что желаемый результат достигнут, откройте импортированную таблицу в режиме Таблицы, дважды щелкнув по ней левой кнопкой мыши.

Чтобы узнать, какие типы данных полей были выбраны при импорте рабочего листа, в окне базы данных откройте созданную таблицу в режиме Конструктора (рис. 3.15). В отличие от присоединенных таблиц, все свойства полей импортированных таблиц можно изменить.

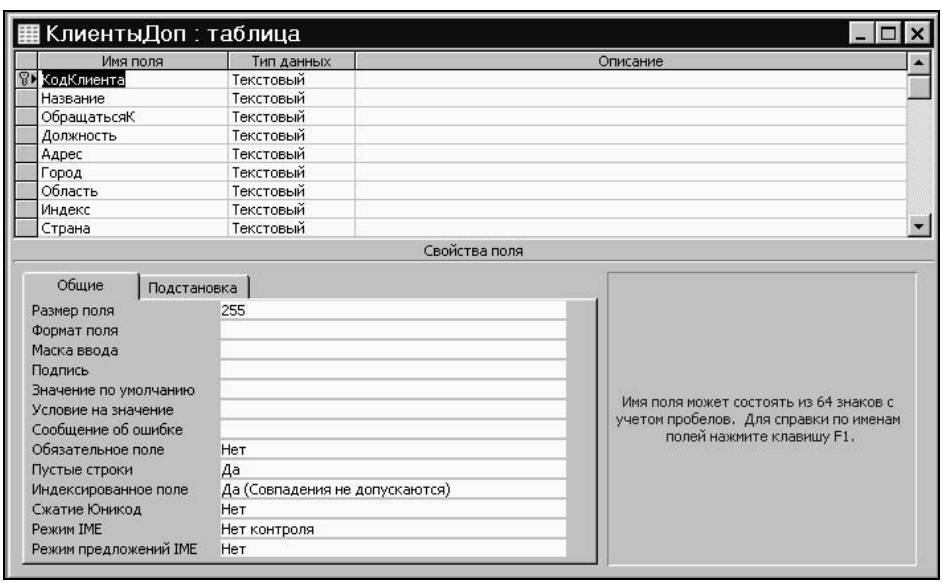


Рис. 3.15. Импортированная таблица в режиме Конструктора

Access позволяет также связать рабочий лист Excel с базой данных Access. Благодаря этой возможности с присоединенным рабочим листом Excel можно работать как в исходном формате средствами Excel, так и в формате таблицы базы данных Access, причем это можно делать одновременно. Что касается изменения структуры рабоче-

го листа, то на него распространяются правила работы с присоединенными таблицами, т. е. в режиме Конструктора можно изменить только те свойства полей, которые связаны с отображением этих полей в таблицах Access. При изменении структуры таблицы или перемещении ее в другую папку нужно обновить связь с ней.

Присоединение рабочего листа Excel выполняется практически так же, как и импорт, только используется команда **Связь с таблицами** (Link Tables). Мастер связывания электронной таблицы (Link Spreadsheet Wizard) аналогичен Мастеру импорта, однако он не позволяет выбирать столбцы рабочего листа и изменять их параметры.

Импорт и присоединение таблиц из текстовых файлов

В некоторых случаях для того, чтобы получить данные из какого-нибудь приложения, невозможно использовать ни один из приведенных ранее форматов в силу отсутствия в нем функции экспорта в данный формат. Тогда остается применить самое простое решение — передать данные посредством обычного текстового файла. Access позволяет импортировать в базы данных табличные данные из текстовых файлов двух стандартных форматов:

- с полями фиксированной ширины — строки состоят из полей фиксированной ширины, пустые позиции при этом заполняются пробелами;
- с полями переменной ширины — со специальными символами-разделителями полей.

Для файлов с фиксированной шириной полей начало каждого поля определяется по его позиции относительно начала строки. Строки имеют одинаковую длину и разделяются символами новой строки (обычно это пара символов — возврат каретки и перевод строки).

Для файлов с разделителями полей существует несколько стандартных вариантов разделения полей. Они приведены в табл. 3.4.

Таблица 3.4. *Форматы текстовых файлов, поддерживаемые Access 2003*

Формат	Описание
Записи в одну строку, поля разделены запятыми	Разделителем записей является символ новой строки. Некоторые приложения заключают все значения полей в двойные кавычки, другие приложения ограничиваются заключением в кавычки лишь текстовых значений, чтобы отличить их от числовых
Записи в одну строку, поля разделены знаками табуляции	Разделителем записей является символ новой строки. Предполагается, что все значения полей текстовые, а сами поля разделены знаками табуляции. Этот формат поддерживается большей частью текстовых процессоров
Записи в одну строку, поля разделены пробелами	Access позволяет использовать пробел в качестве символа разделителя полей, однако использование такого разделителя очень не удобно для файлов, содержащих текстовые данные (в которые также может входить пробел)

Создание таблицы путем импорта из текстового файла

Процесс импорта данных, хранящихся в текстовом файле, выполняется с помощью Мастера импорта текстов, и этот процесс очень похож на импорт файла электронной таблицы.

1. Откройте базу данных, в которую нужно импортировать данные из текстового файла. Активизируйте окно базы данных, щелкнув по его заголовку левой кнопкой мыши.
2. Выберите команду **Файл | Внешние данные | Импорт** (File | Get External Data | Import). Появится диалоговое окно **Импорт** (Import).
3. В раскрывающемся списке **Тип файла** (Files of Type) диалогового окна **Импорт** выделите элемент **Текстовые файлы (*.txt; *.csv; *.tab; *asc)** (Text Files). Откройте папку, содержащую текстовый файл, который нужно импортировать, и выделите его. Нажмите кнопку **Импорт** (Import). Появится первое диалоговое окно Мастера импорта текстов (рис. 3.16).

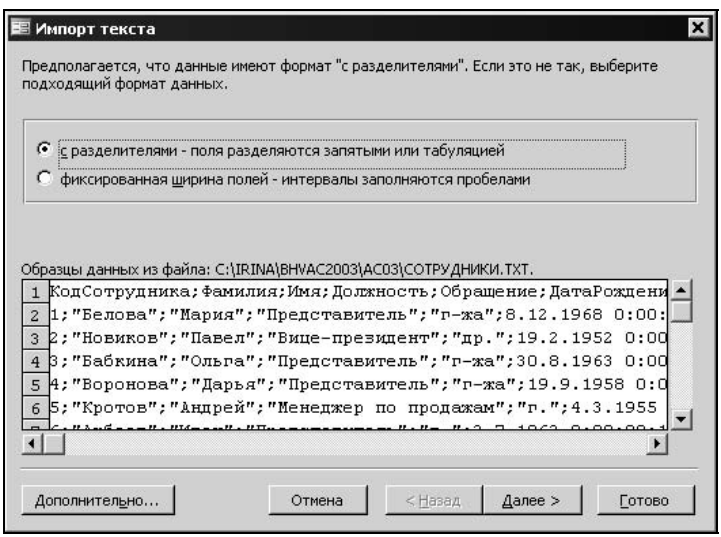


Рис. 3.16. Первое диалоговое окно Мастера импорта текстов

4. В этом окне вы должны выбрать один из двух основных форматов текстовых файлов. Если в текстовом файле для определения конца поля используется разделительный символ, выберите переключатель **с разделителями** (Delimited). Если же текстовый файл содержит записи с полями фиксированной ширины, выберите переключатель **фиксированная ширина полей** (Fixed Width). В нижней части диалогового окна Мастера выводится образец данных из файла, позволяющий визуально определить формат импортируемых данных. Нажмите кнопку **Далее** (Next), чтобы перейти к следующему шагу. Если на первом шаге был выбран переключатель **с разделителями**, появится диалоговое окно, изображенное на рис. 3.17. Если же был выбран переключатель **фиксированная ширина полей**, то на втором шаге Мастера появится диалоговое окно, показанное на рис. 3.18.

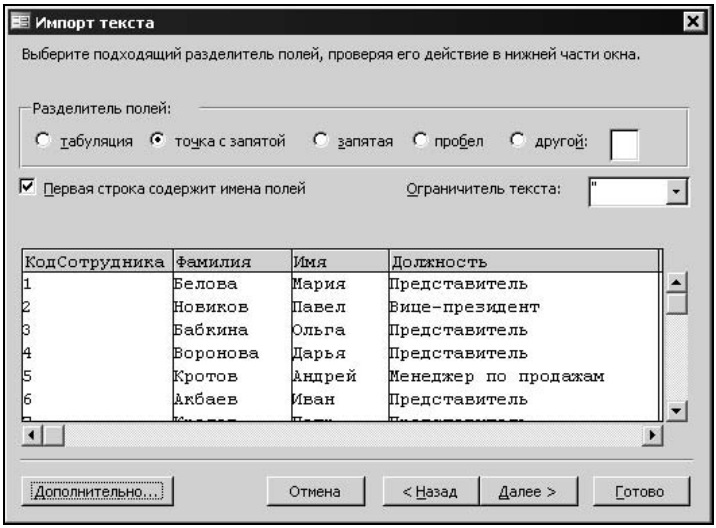


Рис. 3.17. Второе диалоговое окно Мастера импорта текстов для текстовых файлов с разделителями

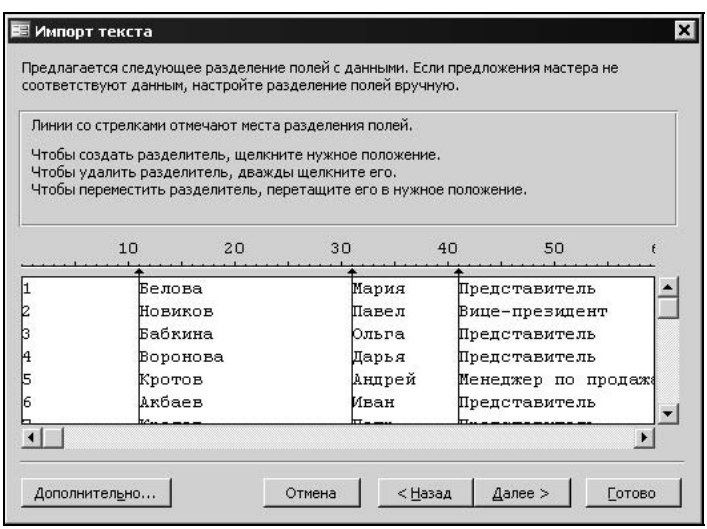


Рис. 3.18. Третье диалоговое окно Мастера импорта текстов для текстовых файлов с фиксированной шириной полей

5. Если файл содержит разделители полей, то на втором шаге укажите символ разделителя. Если в импортируемом файле текстовые значения ограничены символом, отличным от двойных кавычек, то укажите ограничитель текстовых значений, выделив его в раскрывающемся списке **Ограничитель текста** (Text Qualifier). Если в первой строке импортируемого файла содержатся заголовки полей, установите флажок **Первая строка содержит имена полей** (First Row Contains Field

Names). В нижней части окна виден образец, по которому можно определить правильность сделанного выбора. Нажмите кнопку **Далее** (Next), чтобы перейти к следующему шагу Мастера импорта текстов (шаг 7).

6. При импорте текстовых файлов с фиксированной шириной полей Мастер импорта попытается автоматически разделить поля. Просмотрев образец данных, можно указать дополнительное место разделения полей (черную стрелку), а также переместить или удалить существующий разделитель. Для этого внимательно прочитайте указания Мастера. После установки ширины всех полей нажмите кнопку **Далее** (Next) для перехода к следующему шагу.
7. Дальнейшие шаги полностью совпадают с процессом импорта электронной таблицы: указывается, где сохранить данные (в новой или существующей таблицах); изменяется, если нужно, описание каждого поля импорта: определяется ключевое поле и указывается имя результирующей таблицы, если импорт выполняется в новую таблицу.

В результате будет создана таблица с указанным на последнем шаге названием. Если в процессе импорта текстового файла обнаруживаются ошибки, то они, как обычно, фиксируются в таблице ошибок импорта. По окончании импорта данных будет выведено сообщение об успешном или неуспешном завершении операции. Причины ошибок, как правило, те же самые, что и в таблице Excel. Только в текстовых файлах с разделителями нужно быть особенно внимательными, т. к. лишний знак разделителя в строке приведет к тому, что строка не будет импортирована. Кроме того, нужно обращать внимание на кавычки, заключающие поля в импортируемом файле.

Отличием процесса импорта текстового файла является только наличие кнопки **Дополнительно** (Advanced) почти во всех диалоговых окнах Мастера (см. рис. 3.17). Эти кнопки используются для создания Спецификаций импорта *(см. разд. "Создание спецификаций импорта" далее).*

Особенности импорта текстовых файлов

Большинство текстовых процессоров позволяют сохранить данные документов слияния в текстовых файлах без форматирования, которые затем можно использовать для импорта в таблицы Access. Например, в Word 2003 нужно просто открыть документ и сохранить его как текстовый файл, выбрав команду **Файл | Сохранить как** (File | Save As).

Если есть возможность выбора формата текстовых файлов, предназначенных для импорта в таблицы Access, рекомендуется использовать файлы с разделителями полей. Недостатком файлов с фиксированной шириной полей является, во-первых, большой объем, т. к. в них содержится много пробелов, а во-вторых, сложности с определением позиций начала столбцов. Если же все-таки нужно использовать файл с фиксированной шириной полей, будьте внимательны и помните, что Access анализирует только первые 20 строк текстового файла, поэтому позиции разделения полей, определяемые автоматически, могут оказаться неверными.

В качестве символов разделителей полей в текстовых файлах могут использоваться самые разнообразные символы. Access позволяет работать как со стандартными символами разделителей (запятыми, пробелами и символами табуляции), так и указать

какой-то другой символ в качестве разделителя (см. рис. 3.17). Однако нестандартные разделители полей (в том числе и пробелы) на практике встречаются крайне редко.

Кроме того, в текстовых файлах, которые получаются при экспорте данных из разных приложений, могут использоваться двойные кавычки (") для выделения текстовых полей. При импорте данных из таких файлов Access полагает, что поля, значения которых не выделены кавычками, являются числовыми (в том случае, если они содержат числа). Далеко не все приложения выделяют кавычками текстовые значения при импорте в текстовый файл. Например, Excel при экспорте данных в текстовый файл в формате CSV (Comma Delimited) не заключает текстовые значения в кавычки (рис. 3.19).

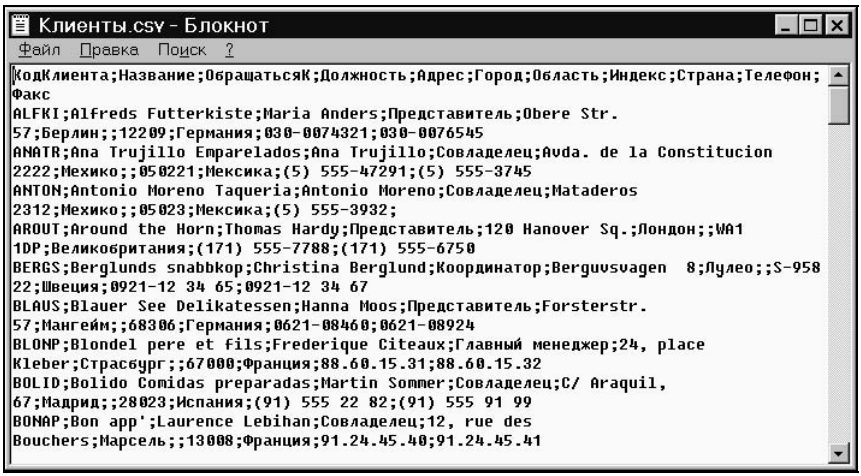


Рис. 3.19. Текстовый файл в формате CSV открыт в приложении Блокнот

Замечание

С помощью приложения Блокнот можно быстро определить тип текстового файла, который предполагается импортировать в таблицу Access. Однако приложение Блокнот может работать только с файлами, размер которых не превышает 60 Кбайт. Для просмотра больших текстовых файлов можно воспользоваться редактором WordPad. Но при использовании этого редактора для просмотра или редактирования текстовых файлов нельзя применять форматирование и сохранять файлы в формате DOC, т. к. в этом случае текстовый файл нельзя будет импортировать.

В текстовом файле SDF (Standard Data Format), создаваемом командой dBASE `COPY TO` *ИмяФайла* `DELIMITED`, все текстовые значения заключаются в прямые кавычки (рис. 3.20). Значения полей типа Numeric и Date в кавычки не заключаются. Этот формат текстовых файлов является стандартом для многих СУБД и широко распространен в других приложениях.

Многие текстовые процессоры в качестве разделителя полей применяют символы табуляции. Символ табуляции используется также при преобразовании обычного текста в таблицу в качестве разделителя ячеек. В текстовых файлах с разделением

полей символами табуляции редко используется выделение текстовых значений кавычками. Access при экспорте данных в текстовые файлы, в которых в качестве разделителя используется символ табуляции, заключает текстовые значения в кавычки, чтобы символы новой строки, которые могут содержаться в текстовых полях, не воспринимались как разделители записей. Тем не менее не рекомендуется использовать символы новой строки в текстовых полях.

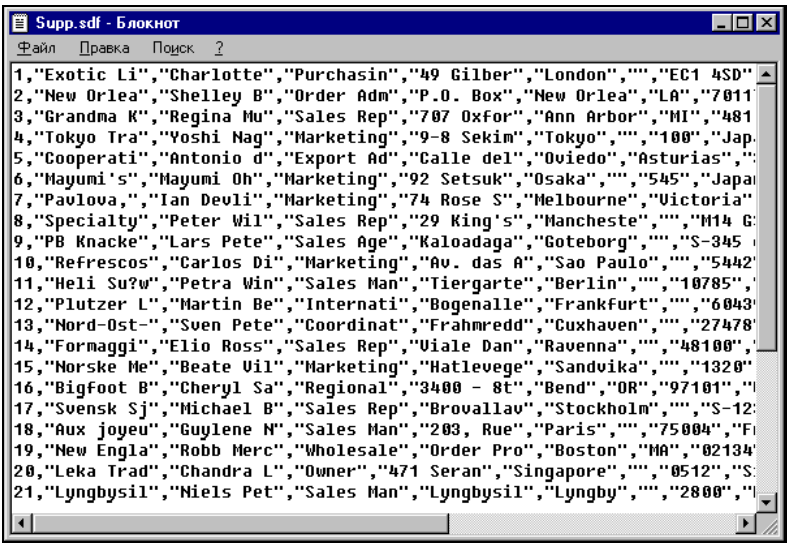


Рис. 3.20. Текстовый файл в формате SDF открыт в приложении Блокнот

Импорт и присоединение документов HTML

Access 2003 позволяет импортировать и присоединять таблицы из документов HTML. Для этого используется Мастер импорта (присоединения) HTML, который очень похож на Мастер импорта (присоединения) электронных таблиц. Чтобы присоединить таблицу из документа HTML, необходимо:

1. Открыть базу данных, в которую требуется импортировать или присоединить таблицу, после чего активизировать окно базы данных, щелкнув по его заголовку левой кнопкой мыши.
2. Выбрать команду **Файл | Внешние данные | Связь с таблицами** (File | Get External Data | Link) или **Файл | Внешние данные | Импорт** (File | Get External Data | Import). Появится диалоговое окно **Связь** (Link) или **Импорт** (Import).
3. В раскрывающемся списке **Тип файла** (Files of Type) появившегося окна нужно выделить элемент **Документы HTML** (HTML Files), открыть папку, содержащую документ HTML, который требуется импортировать, и выделить его. Нажать кнопку **Связь** (Link) или **Импорт** (Import). Появится первое диалоговое окно Мастера (рис. 3.21).

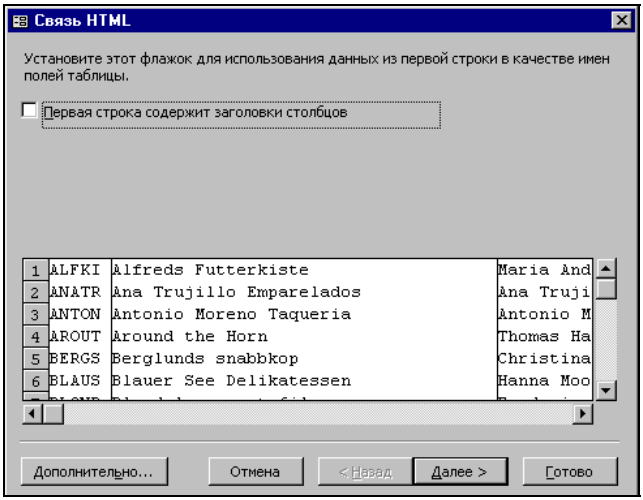


Рис. 3.21. Первое диалоговое окно Мастера присоединения документов HTML

Замечание

Access позволяет также присоединять и импортировать документы HTML, которые находятся на удаленных FTP- и HTTP-серверах. Для этого требуется в поле **Имя файла** (File Name) диалогового окна **Связь** (Link) или **Импорт** (Import) указать интернет-адрес нужного файла.

4. Если первая строка таблицы содержит заголовки столбцов, их можно использовать в качестве имен полей. Для этого необходимо установить флажок **Первая строка содержит заголовки столбцов** (First Row Contains Column Headings). В нашем примере заголовков столбцов не было, поэтому флажок не установлен. Нажмите кнопку **Далее**, чтобы перейти к следующему шагу.
5. При импорте данных из документа HTML тоже можно либо создавать новую таблицу Access, либо добавлять данные в существующую таблицу, поэтому далее выводится диалоговое окно, аналогичное представленному на рис. 3.10, в котором и нужно определить, где должны сохраняться импортируемые данные. При присоединении страницы HTML этот шаг пропускается.
6. В следующем диалоговом окне Мастера можно описать каждое поле импорта: имя поля, тип данных. Индексироваться поля в данном случае не могут (рис. 3.22).
7. На следующем шаге Мастер запрашивает указание о том, какое поле можно считать ключевым для новой таблицы.

Замечание

При присоединении таблицы шаги 5 и 7 пропускаются, ключевое поле не определяется и присоединенная таблица из документа HTML будет доступна только в режиме чтения.

8. В последнем диалоговом окне нужно указать имя таблицы Access.

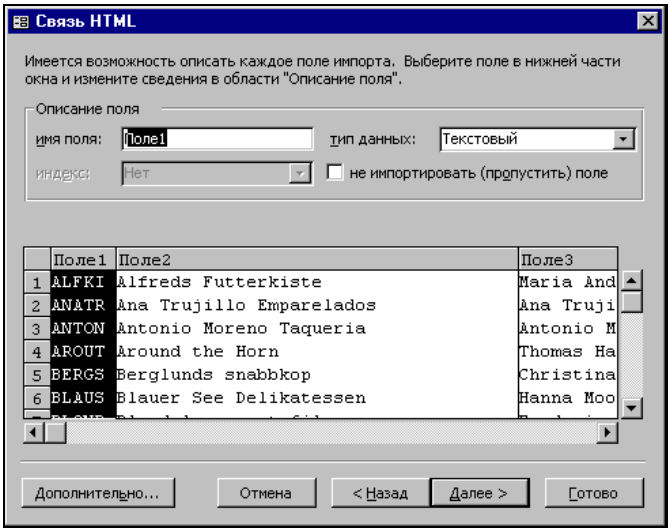


Рис. 3.22. Второе диалоговое окно Мастера присоединения документов HTML

Гиперссылки, которые могут содержаться в файле HTML, преобразуются по умолчанию в поля типа Гиперссылка, но только в том случае, если они определены тегом `<A HREF>`. Если в таблице содержатся вложенные рисунки в формате GIF или JREG, они будут игнорироваться. Корректно импортируются только обычные таблицы (не вложенные). Если в ячейку таблицы на странице HTML вложен список значений, то все эти значения при импорте склеиваются без каких-либо разделителей (в Справке Microsoft Access говорится, что разделяются символами конца строки, но в наших экспериментах этого не наблюдалось).

Создание спецификаций импорта

Спецификация импорта — специальный текстовый файл, который используется при повторном выполнении импорта одной и той же таблицы. Чтобы создать спецификацию импорта, нажмите кнопку **Дополнительно** (Advanced) в диалоговом окне Мастера импорта на любом шаге процесса. В появившемся диалоговом окне ***<имя файла>* спецификация импорта** (...Import Specification) можно изменить имена и типы полей, а также другие параметры импортируемых данных. Окно спецификации импорта для файлов с разделителями изображено на рис. 3.23, а для файлов с полями фиксированной ширины — на рис. 3.24. В табл. 3.5 приведены все параметры импорта, которые можно указать в спецификации. После того как создание спецификации импорта завершено, нажмите кнопку **Сохранить как** (Save As). В появившемся диалоговом окне **Сохранение спецификации импорта/экспорта** (Save Import/Export Specification) укажите имя спецификации и нажмите кнопку **OK**.

Для того чтобы воспользоваться сохраненной спецификацией при очередном импорте данных, после запуска Мастера импорта текстовых файлов нажмите кнопку

Дополнительно (Advanced), а затем в диалоговом окне **Спецификация импорта** (Import Specification) нажмите кнопку **Спецификация** (Specs), выделите нужную спецификацию и нажмите кнопку **OK**.

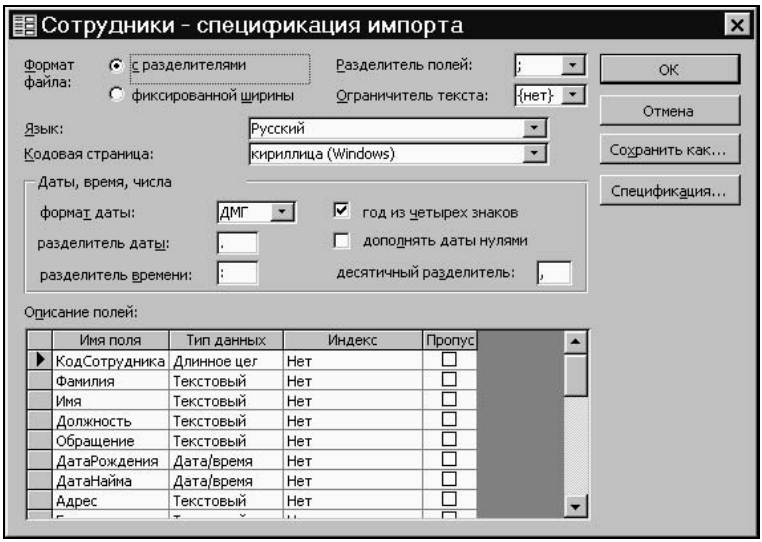


Рис. 3.23. Окно спецификации импорта для текстовых файлов с разделителями полей

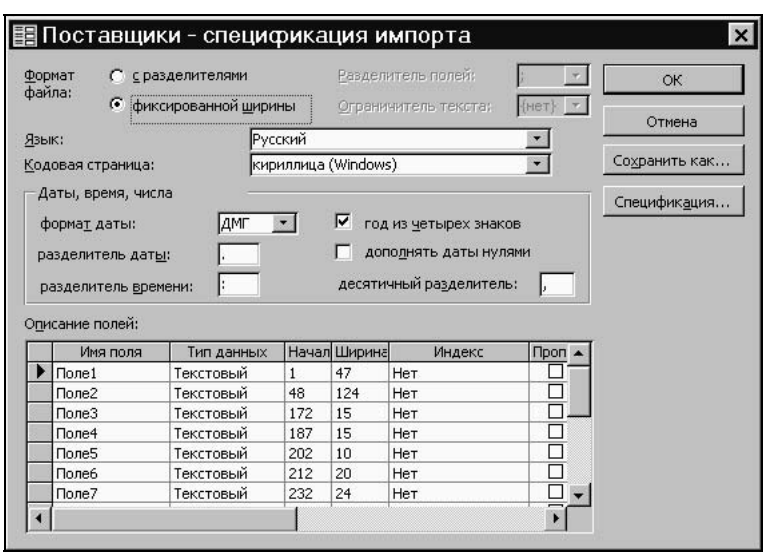


Рис. 3.24. Окно спецификации импорта для текстовых файлов с полями фиксированной ширины

Таблица 3.5. *Параметры импорта, определяемые в спецификации*

Параметр	Описание
Формат файла (File Format)	Позволяет указать тип импортируемого текстового файла: с разделителями полей или с полями фиксированной ширины. В зависимости от выбранного формата файла изменяется набор параметров спецификации импорта
Разделитель полей (File Delimiter)	Этот раскрывающийся список используется для определения символа разделителя полей импортируемого файла. Для файлов с фиксированной шириной полей этот параметр не доступен. Не доступен он и при импорте данных из HTML-файлов
Ограничитель текста (Text Qualifier)	Для определения символа, которым отмечается начало и конец текстовых полей, выделите соответствующий символ в раскрывающемся списке. Для файлов с фиксированной шириной полей этот параметр также не доступен. Не доступен он и при импорте данных из HTML-файлов
Язык (Language) **Кодовая страница** (Code Page)	Эти параметры используются для выбора кодировки импортируемого текстового файла
формат даты (Data Order)	Если в импортируемом файле для дат используется формат, отличный от европейского **ДМГ** (например, американский **МДГ**), то выберите соответствующий формат представления дат в раскрывающемся списке
разделитель даты (Data Delimiter) **разделитель времени** (Time Delimiter)	В спецификации импорта можно указать символы, используемые в качестве разделителей компонентов даты и времени. Для этого просто введите их в поля **разделитель даты** и **разделитель времени**. В качестве разделителя компонентов даты чаще всего используется точка, а в качестве разделителя компонентов времени — двоеточие
год из четырех знаков (Four Digit Years)	Этот флажок позволяет указать, что в текстовом файле для года используется четыре знака (например, 23.03.1999)
дополнять даты нулями (Leading Zeros in dates)	Установите этот флажок, если в текстовом файле в датах используются ведущие нули (например, 04.01.96)
десятичный разделитель (Decimal Simbol)	Введите символ, используемый в импортируемом файле в качестве разделителя целой и дробной части (десятичной точки)
Описание полей (Field Information)	Вид бланка описания полей зависит от выбранного формата файла. Для текстовых файлов с разделителями и файлов HTML можно указать имя поля, тип данных, необходимость индексации поля (см. рис. 3.23). Можно также исключить (пропустить) поле при импорте. Для текстовых файлов с фиксированной шириной полей можно дополнительно указать смещение относительно начала файла и ширину каждого поля (см. рис. 3.24)

Импорт данных из документов и схем XML

В новой версии Microsoft Office Access 2003 усовершенствована поддержка формата XML как универсального формата для обмена данными между различными приложениями. Microsoft Access 2003 поддерживает возможность как импорта данных из файлов формата XML, так и экспорта в формат XML основных типов объектов — таблиц, запросов, форм, отчетов.

Формат XML в Access. Общие сведения

XML (Extensible Markup Language) — это расширяемый язык разметки гипертекста, разработанный на базе HTML как формат универсального представления данных в глобальных сетях. Этот формат позволяет совершенно различным приложениям обмениваться данными через Интернет. XML расширяет возможности HTML, т. к. позволяет создавать дополнительные элементы, с помощью которых можно описывать и определять новые данные, объекты и их свойства, отделяя данные от представления их в виде HTML, благодаря чему удается преодолеть ограничения HTML в возможности описания нестандартных объектов. XML официально принят консорциумом W3C (World Wide Web Consortium), который занимается стандартами, относящимися к World Wide Web.

Теперь поговорим о структуре документа XML. Дело в том, что такой документ может состоять не из одного файла XML, а из нескольких файлов разных типов. Это могут быть файлы следующих типов:

- Web-страница для просмотра полученного документа (в формате HTML или ASP);
- файл XSL, содержащий описание структуры внешнего представления документа;
- файл XML, включающий в себя данные, являющиеся источником для заполнения документа;
- файл XSD, содержащий описание структуры данных, обычно его называют *схемой данных*. Схема данных может не выделяться в отдельный файл, а быть включена прямо в файл XML, который содержит сами данные.

Это обусловлено логическим разделением документа XML на отдельные части: данные (XML), структуру данных (XSD) и представление данных (HTML или ASP, XSL).

Подчеркнем, что все эти файлы вместе образуют единый документ Web, который можно просмотреть с помощью обозревателя Internet Explorer 5 и выше. Они лишь описывают разные его части. Однако файл XML (возможно, вместе с файлом XSD) может быть использован приложениями, распознающими этот формат данных, независимо от других частей документа. Файл в формате HTML или ASP содержит лишь сценарий, который активизируется при загрузке страницы и загружает данные на страницу из источника в формате XML.

Отделение самих данных от их представления (в формате HTML) и помещение их в отдельный файл в формате XML открывает возможность другим приложения, воспринимающим этот специально разработанный универсальный формат, получать и обрабатывать данные из такого документа независимо от их представления. Отделение данных от их представления позволяет также приложениям применять различ-

ные способы для отображения одних и тех же данных XML с помощью нескольких различных схем представления данных.

Файл XSD называется *схемой XML*. Его содержание удовлетворяет стандарту XML Schema standard (XSD), официально принятому консорциумом W3C. Файл схемы XML описывает структуру данных в универсальном виде, включая информацию о названиях элементов, типах данных, комбинациях элементов, об атрибутах элементов. Схема XML определяет модель представления данных в формате XML: задает правила для тегов и текста. Применение схемы XML гарантирует правильное восприятие данных в формате XML другими приложениями и правильное преобразование этих данных в другие форматы данных.

Схема XML содержит описание данных, но не содержит описание того, как они должны быть отображены в программе просмотра. Эта информация предоставляется отдельно, в файлах CSS на языке описания стилей Cascading Style Sheet. Однако это не слишком удобно, т. к. разработчику приходится изучать еще и язык CSS в дополнение к XML, к тому же CSS предоставляет недостаточно средств для контроля над выводом данных. В современных приложениях чаще применяется более гибкое средство для описания внешнего представления данных XML — язык XSL (Extensible Stylesheet Language). Он позволяет точно выбрать данные, которые требуется отобразить, задать порядок расположения элементов данных, модифицировать и добавить дополнительную информацию. Кроме того, этот язык похож на XML: в нем для создания шаблона стиля вывода данных используются теги, подобные тегам в XML, и конструкции HTML. Заметим, что для отображения данных XML в Internet Explorer 5 или выше не обязательно присоединять файлы CSS или XSL, т. к. эта программа просмотра имеет собственное описание стиля, применяющееся по умолчанию. Используйте собственные файлы описания стилей, чтобы обеспечить единообразный внешний вид ваших Web-страниц, основанных на данных XML.

И наконец, в Access 2003 используется еще разрабатываемый в настоящее время консорциумом W3C язык преобразования стилей XSL for Transformation (XSLT). Он основан на языке XML и применяется для преобразования внешнего вида документа XML одного формата в другой. Присоединение файла XSLT к документу XML позволяет интерпретировать и преобразовать этот документ в другой формат, который может распознать другое приложение.

Импорт документов XML

Access 2003 обеспечивает возможность импорта данных в формате XML, и процедура эта аналогична процедуре импорта данных из таблиц Access, т. е. вы можете импортировать:

- только структуру таблицы;
- таблицу вместе с данными;
- только данные, которые добавляются в уже существующую таблицу.

Чтобы выполнить импорт таблиц из файла XML, нужно:

1. Открыть базу данных, в которую требуется импортировать данные. Активизировать окно базы данных, щелкнув по его заголовку левой кнопкой мыши. Выбрать команду **Файл | Внешние данные | Импорт** (File | Get External Data | Import) или раскрыть список таблиц в окне базы данных, щелкнуть правой кнопкой мыши

на свободном поле в этом списке и выбрать из контекстного меню команду **Импорт** (Import).

2. В диалоговом окне **Импорт** (Import) выбрать в раскрывающемся списке **Тип файла** (Files of Type) элемент **XML** (XML Documents).
3. В поле **Папка** (Look in) выбрать папку, содержащую файл XML, который нужно импортировать, и выделить его в списке. Нажать кнопку **Импорт** (Import).
4. Появится первое диалоговое окно **Импорт XML** (Import XML) (рис. 3.25), в котором будет отображен список таблиц, содержащихся в выбранном файле XML. Импортироваться будут все перечисленные таблицы, вы не можете выбрать в этом списке одну или несколько таблиц. По умолчанию выполняется импорт таблицы вместе с данными. Если же нужно импортировать только структуру таблицы или добавить данные из файла XML в уже существующую таблицу, нажмите кнопку **Параметры** (Options). В нижней части окна появится группа переключателей, в которой необходимо выбрать требующийся вариант. Нажмите кнопку **ОК**.

Access выполняет импорт данных, причем информация о структуре данных берется либо прямо из импортируемого файла XML, либо из файла схемы данных XSD, который должен размещаться в той же папке и иметь то же имя, что и файл, содержащий данные.

Как видите, процесс импорта данных из XML-файла очень прост, но и менее гибок, чем импорт HTML-файла или текстового файла — таблицы всегда импортируются полностью и нельзя изменить свойства столбцов, вы даже не видите тех данных, которые импортируете. В случае необходимости выполнить преобразование XML-файла в процессе импорта в Access 2003 можно использовать файл преобразований XSLT.

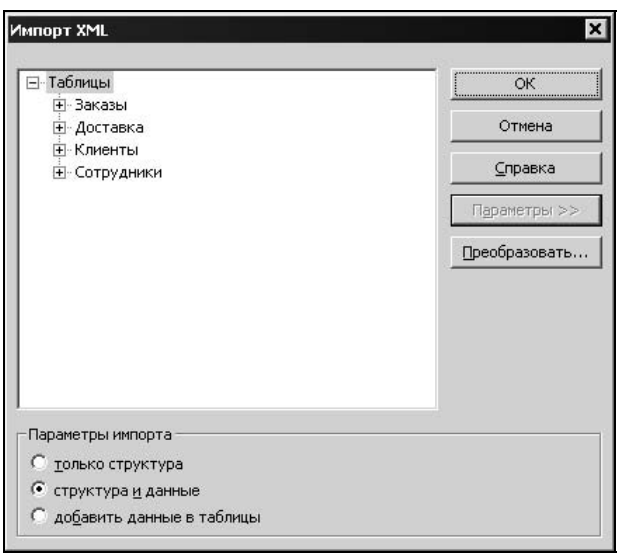


Рис. 3.25. Диалоговое окно импорта документа XML

В отличие от предыдущей версии Access, в Access 2003 можно выполнить это преобразование в процессе импорта, например изменить структуру таблицы или отсортировать ее определенным образом. Чтобы указать файл преобразований, нужно нажать кнопку **Преобразовать** (Transform) в окне **Импорт XML** (Import XML). Тогда появится еще одно диалоговое окно — **Преобразования для импорта** (Import Transforms, рис. 3.26). В нем нужно выбрать то преобразование, которое требуется применить к исходному файлу. Если в той же папке присутствует одноименный файл с расширением xsl, то имя этого файла будет указано в списке, если нет, то нужно найти файл преобразований XSL. Для этого нажмите кнопку **Добавить** (Add) и выберите в стандартном окне Windows требуемый файл. Затем нажмите кнопку **OK**.

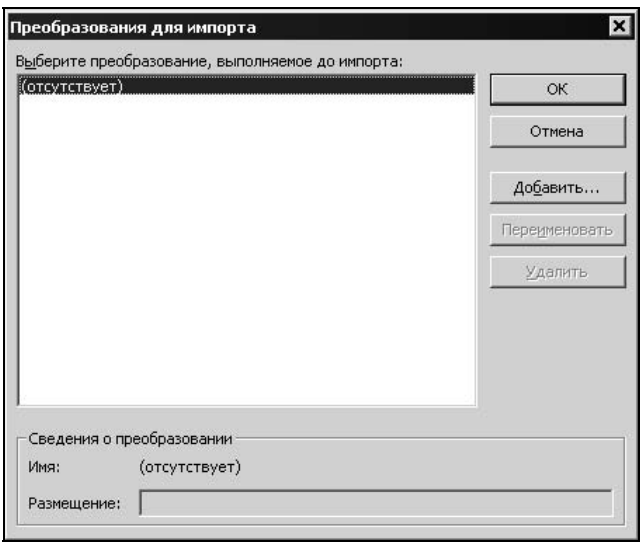


Рис. 3.26. Диалоговое окно **Преобразования для импорта**

Файл XSLT может быть создан либо в процессе экспорта данных из другого источника, либо в любом текстовом редакторе.

Примечание

Подробные сведения о языке XSLT можно найти на Web-узле Microsoft Online XML Developer Center — **http://www.microsoft.com/xml/**.

Импорт и присоединение списков из служб Windows SharePoint

Windows SharePoint Services (службы Windows SharePoint) представляют собой набор служб Microsoft Windows Server 2003, которые используются для организации совместного доступа к разного рода информации различным группам пользователей. Эти службы обеспечивают общее рабочее пространство в форме Web-узлов, которое содержит библиотеки документов, форм, рисунков, а также разнообразную полезную

информацию, организованную в виде списков — контактов, задач, ссылок, событий, извещений и т. д. Список — компонент Web-узла, хранящий и отображающий сведения, которые могут быть добавлены пользователями с помощью обозревателей или приложений Microsoft Office 2003.

Windows SharePoint Services тесно интегрированы со всеми приложениями Microsoft Office 2003. С помощью этих служб можно создавать узлы, отслеживать версии документов, совместно пользоваться информацией, которая хранится в списках узла. Создать узел, который будет служить рабочей областью для коллектива сотрудников, можно, например, прямо из Word 2003 или Excel 2003. Затем в эти рабочие области можно выложить документы для совместной работы — файлы Word, Excel, XML-файлы, содержащие данные, которые теперь доступны как в Word 2003, так и в Excel 2003, раздать права на эти документы. При создании узла автоматически создаются стандартные списки Windows SharePoint Services — ссылки, извещения, контакты, события, задачи, вопросы — но эти ссылки сначала пустые. Дальше нужно заполнить их данными. Для этого можно использовать электронные таблицы Excel или таблицы базы данных Access. Функция экспорта Access позволяет выполнить экспорт таблицы или запроса в список Windows SharePoint Services. С другой стороны, можно импортировать в базу данных Access любой список с узла Windows SharePoint Services. Или присоединить его так, чтобы соответствующая таблица Access всегда содержала актуальные данные из этого списка.

Один и тот же список Windows SharePoint Services может иметь несколько разных представлений — полная аналогия с таблицами и запросами Access. *Представление* — это определенным образом обработанный список, например, отсортированный, или с группировкой строк, или с ограниченным числом столбцов. Импортировать и связывать можно как сам список, так и его представление.

Подробнее о службах Windows SharePoint Services можно познакомиться на сайте Microsoft TechNet в разделе **/prodtechnol/windowsserver2003/technologies/sharepoint/default.mspx**.

Для импорта (связывания) списка Windows SharePoint Services в базу данных Access необходимо:

1. Открыть базу данных, в которую нужно импортировать список. Раскрыть в окне базы данных список таблиц, щелкнуть правой кнопкой мыши на свободном поле и выбрать из контекстного меню команду **Импорт** (Import) или команду меню **Файл | Внешние данные | Импорт** (File | Get External Data | Import).
2. В раскрывающемся списке **Тип файла** (Files of Type) диалогового окна **Импорт** выделить элемент **Windows SharePoint Services** и нажать кнопку **Импорт** (Import). После этого будет запущен Мастер импорта из Windows SharePoint Services и откроется первое диалоговое окно **Выбор узла** (Select a Site).
3. На первом шаге нужно указать узел, с которого будут импортироваться данные. Этот узел можно либо выбрать из списка ранее используемых узлов, либо ввести ссылку в поле **Узел** (Site) (рис. 3.27). Нажмите кнопку **Далее** (Next).
4. В следующем окне нужно выбрать список или его представление для импорта (рис. 3.28). Мастер импорта позволяет импортировать одновременно несколько списков или несколько представлений одного списка. Выберите нужный вариант импорта, а затем списки или их представления, которые хотите импортировать. Если импортируемый список содержит столбец подстановок (или несколько та-

ких столбцов), то можно импортировать не только сам список, но и все связанные списки, которые содержат подстановочные значения для этих столбцов. Если нужно импортировать именно такой список и вы хотите импортировать связанные списки, установите флажок **Загрузить коды для столбцов подстановки** (Retrieve IDs for lookup columns). Нажмите кнопку **Далее** (Next).

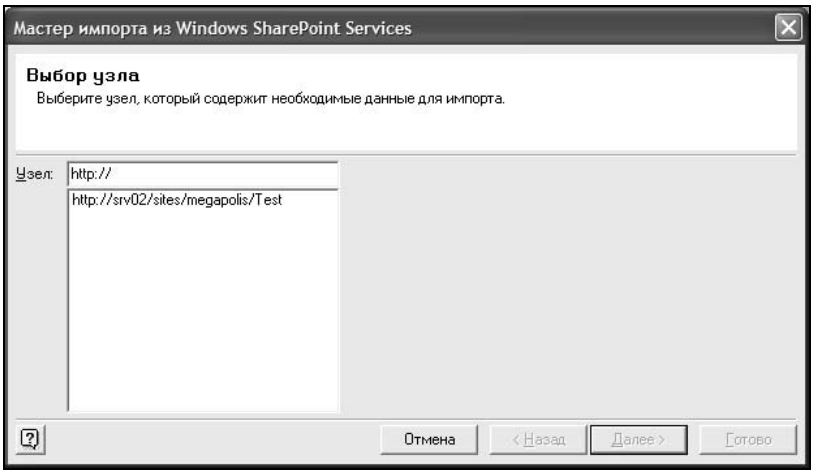


Рис. 3.27. Диалоговое окно **Выбор узла** Мастера импорта из Windows SharePoint Services

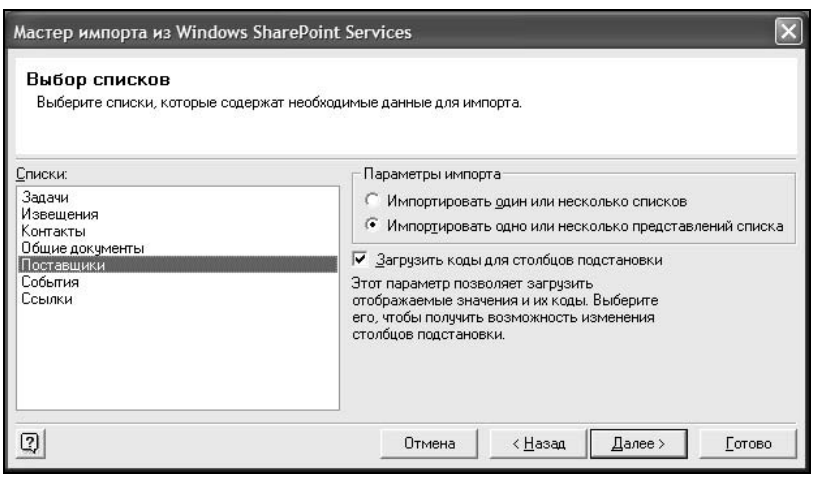


Рис. 3.28. Диалоговое окно **Выбор списков** Мастера импорта из Windows SharePoint Services

5. Если на предыдущем шаге был выбран вариант импорта нескольких представлений одного списка, то следующее окно — **Выбор представлений** (Select View) (рис. 3.29) дает возможность выбрать те представления, которые требуется импортировать (для выбора нескольких представлений нужно использовать клавиши <Ctrl> и <Shift>). Нажмите кнопку **Далее** (Next).

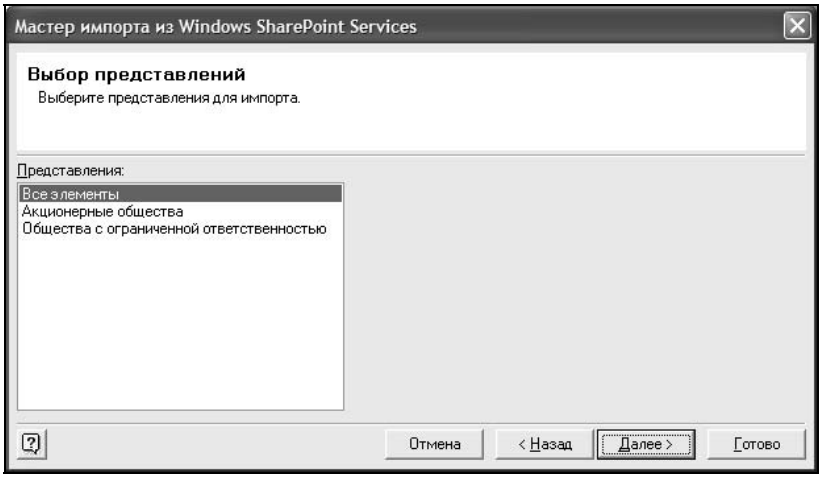


Рис. 3.29. Диалоговое окно **Выбор представлений** Мастера импорта из Windows SharePoint Services

6. Если должны импортироваться связанные списки, то следующим окном будет — **Выбор связанных списков** (Select Related Lists) (рис. 3.30). В нем нужно выбрать требуемые списки (здесь просто щелкните мышью на нужных списках) и в группе **Параметры связи и импорта** (Link/Import Options) укажите, хотите вы их импортировать или связывать. Нажмите кнопку **Далее** (Next).

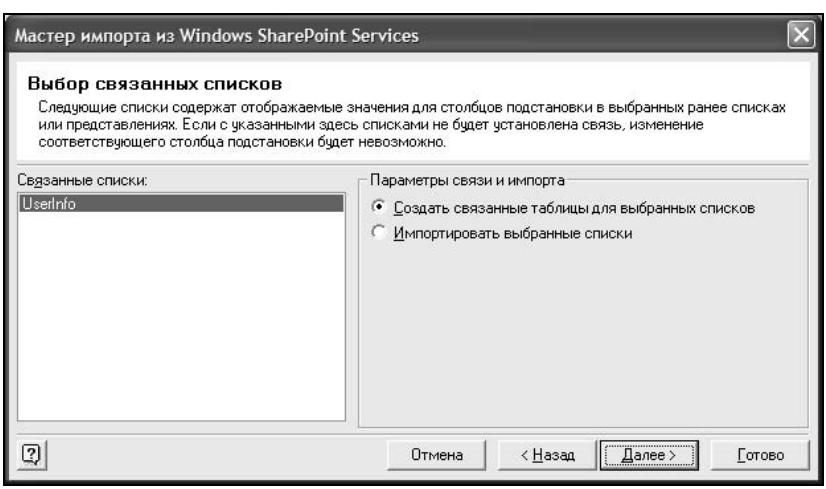


Рис. 3.30. Диалоговое окно **Выбор связанных списков** Мастера импорта из Windows SharePoint Services

7. Последнее окно **Готово** (Finish) будет отображать все выбранные списки и представления, включая связанные (рис. 3.31). Нажмите кнопку **Готово** (Finish). Все выбранные для импорта списки и/или представления будут импортированы в ба-

зу данных Access. Если для связанных списков был выбран параметр **Импорт**, то они также будут импортированы в таблицы Access с автоматической установкой связей между ними. Если для связанных списков выбран параметр **Связь**, то для них в базе данных Access будут созданы присоединенные таблицы.

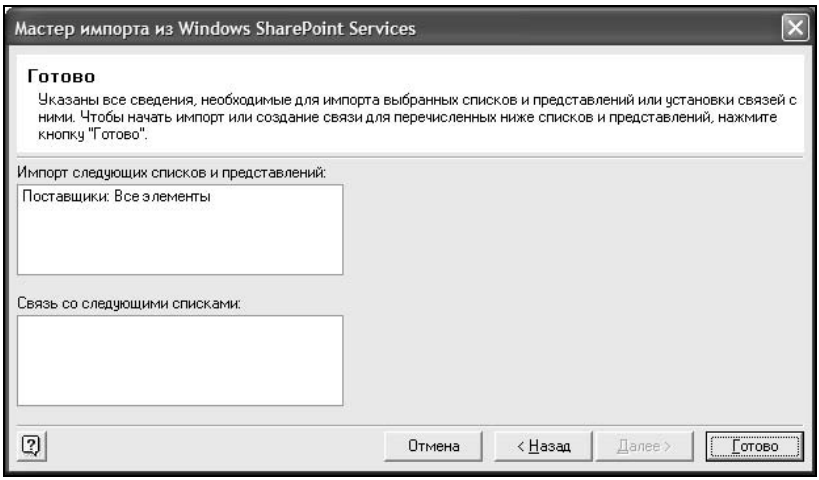


Рис. 3.31. Диалоговое окно **Готово** Мастера импорта из Windows SharePoint Services

Присоединение списков и представлений Windows SharePoint Services к базе данных Access выполняется аналогично. Мастер связи с Windows SharePoint Services будет отображать окна, аналогичные тем, что показаны на приведенных ранее рисунках для импорта списков. Отличие будет состоять только в том, что связанные списки для столбцов подстановки основного списка или представления будут всегда присоединяться, а не импортироваться, поэтому будет отсутствовать группа **Параметры связи и импорта** (Link/Import Options).

Типы данных в списках Windows SharePoint Services и таблицах Access не совпадают. В процессе импорта происходит преобразование типов данных.

Подробную информацию о соответствии типов данных Access и Windows SharePoint Services смотрите в разделе справки Access 2003 *"Использование Microsoft Access при работе в Интернете", "Access и службы Windows SharePoint Services", "Преобразование типов данных Access и SharePoint в программе Access".*

Замечание

Операции импорта и экспорта списков возможны только с узлов на Windows SharePoint Services 2.0.

Импорт и присоединение данных из адресных книг и папок Exchange

Access 2003 позволяет также импортировать данные из адресных книг Microsoft Outlook и папок Microsoft Exchange или присоединять эти книги и папки в виде таб-

лицы Access. Чтобы выполнить импорт данных из адресной книги Outlook, необходимо:

1. Открыть базу данных, в которую нужно импортировать данные. Выбрать команду **Файл | Внешние данные | Импорт** (File | Get External Data | Import) или раскрыть список таблиц и выполнить команду контекстного меню **Импорт** (Import).
2. В раскрывающемся списке **Тип файла** (Files of Type) диалогового окна **Импорт** (Import) нужно выделить элемент **Exchange**. После этого будет запущен Мастер импорта из Exchange/Outlook и откроется диалоговое окно, содержащее список адресных книг и папок Exchange, если вы используете Exchange как систему электронной почты (рис. 3.32). Раскройте список адресных книг и выберите в нем нужную книгу или папку и нажмите кнопку **Далее** (Next).

Замечание

Последовательность диалоговых окон Мастера импорта из Exchange/Outlook аналогична той последовательности, которая выдается Мастером импорта электронных таблиц.

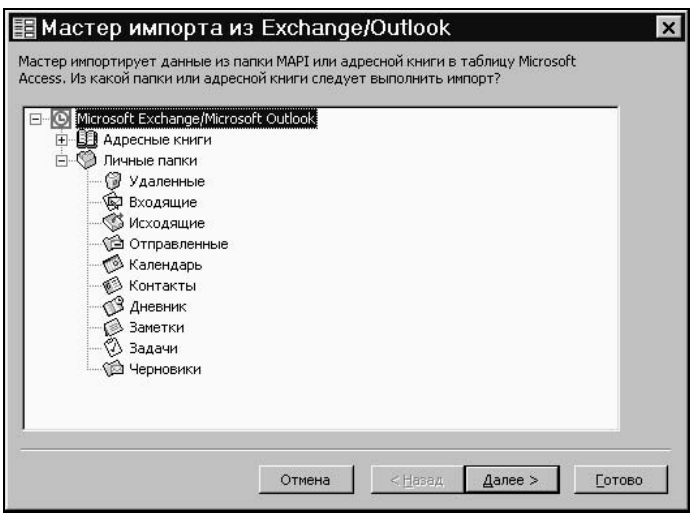


Рис. 3.32. Первое диалоговое окно Мастера импорта из Exchange/Outlook

3. В следующем диалоговом окне необходимо установить переключатель, указывающий, где нужно сохранить импортируемые данные (рис. 3.33). Нажмите кнопку **Далее** (Next).
4. Теперь можно описать каждое поле адресной книги, которое нужно импортировать. Мы в примере изменили имя первого поля (рис. 3.34). Нажмите кнопку **Далее** (Next).
5. В четвертом диалоговом окне нужно определить ключевое поле (рис. 3.35). В данном случае лучше выбрать переключатель **автоматически создать ключ**. Нажмите кнопку **Далее** (Next).

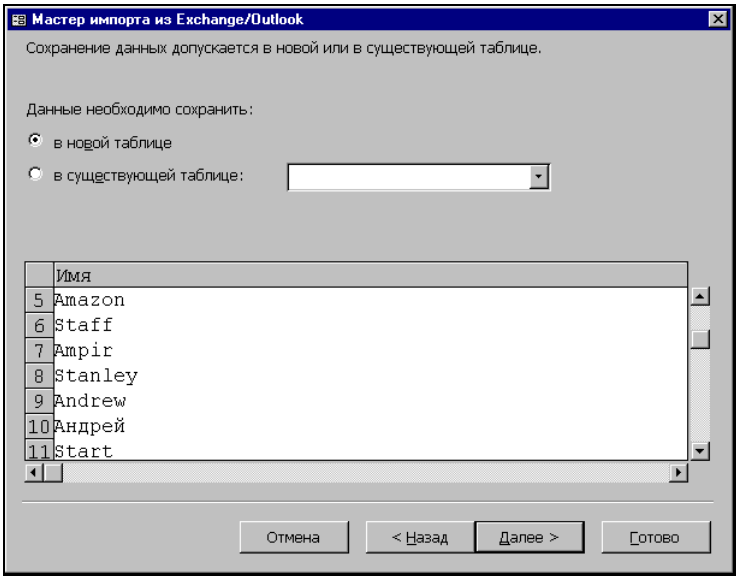


Рис. 3.33. Второе диалоговое окно Мастера импорта из Exchange/Outlook

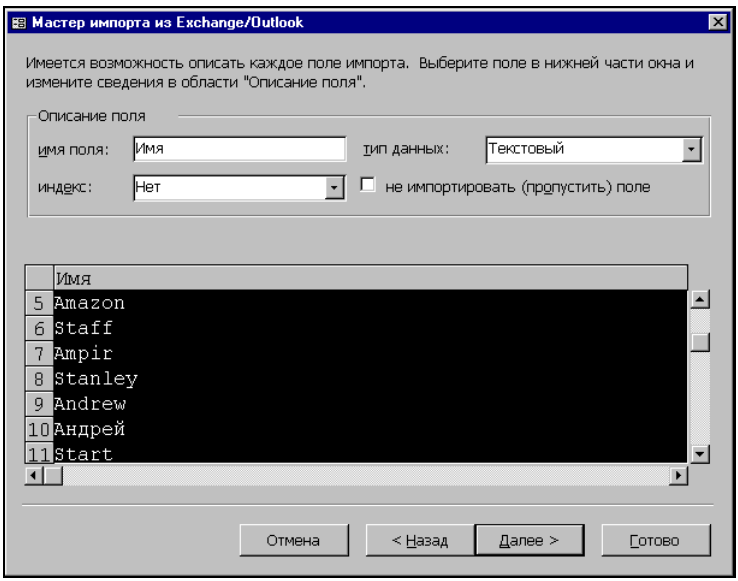


Рис. 3.34. Третье диалоговое окно Мастера импорта из Exchange/Outlook

6. В последнем диалоговом окне требуется только определить имя новой таблицы и нажать кнопку **Готово** (Finish).

Откройте получившуюся таблицу (в нашем случае, `Контакты`). Она будет выглядеть так, как показано на рис. 3.36.

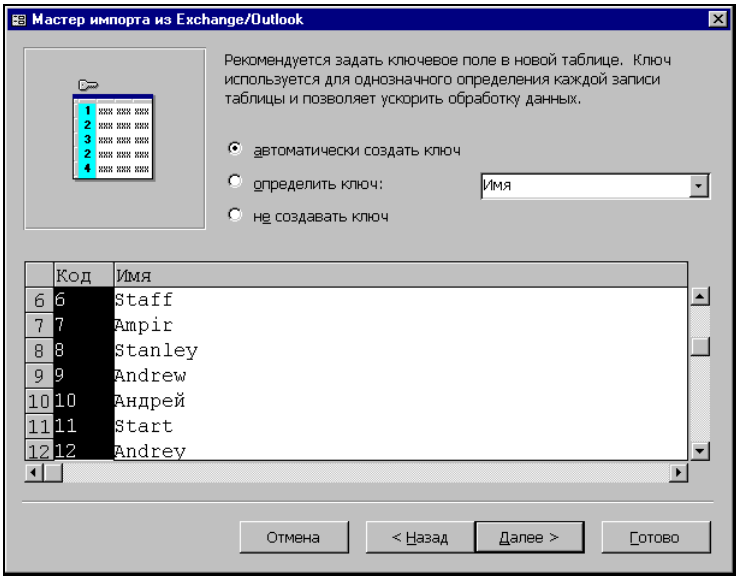


Рис. 3.35. Четвертое диалоговое окно Мастера импорта из Exchange/Outlook

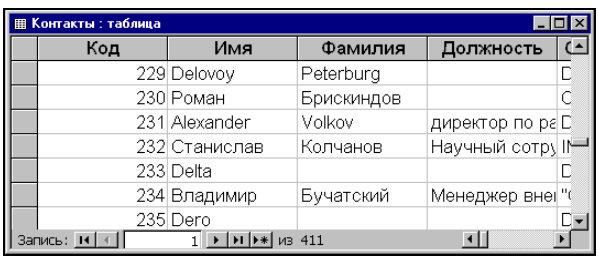


Рис. 3.36. Импортированные данные из папки Контакты Outlook

Присоединение адресных книг и папок Exchange/Outlook выполняется аналогично.

1. Открыть базу данных, к которой нужно присоединить адресную книгу или папку Exchange. Активизировать окно базы данных, щелкнув по его заголовку левой кнопкой мыши, и выбрать команду **Файл | Внешние данные | Связь с таблицами** (File | Get External Data | Link Tables).
2. В раскрывающемся списке **Тип файла** (Files of Type) диалогового окна **Связь** нужно выделить элемент **Exchange**. Появляется диалоговое окно Мастера, которое выглядит так же, как при импорте из адресной книги (см. рис. 3.32). Выберите нужную папку Exchange/Outlook (рис. 3.37) и нажмите кнопку **Далее**. В примере мы выбрали папку Контакты.
3. Сразу появится последнее диалоговое окно Мастера, в которое необходимо ввести имя присоединенной таблицы.

После завершения операции присоединенная папка будет выглядеть так, как это показано на рис. 3.38.

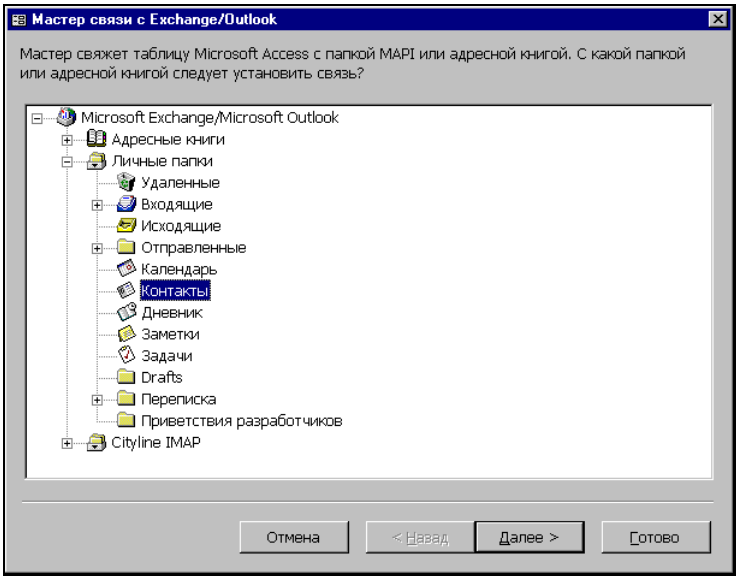


Рис. 3.37. Выбор папки Exchange/Outlook

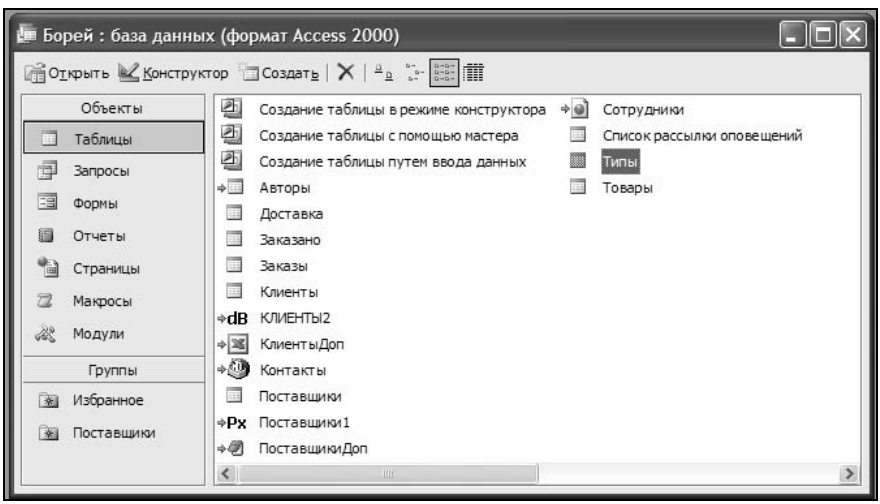


Рис. 3.38. Присоединенные файлы разных форматов

Импорт и присоединение таблиц из источника данных ODBC

Последнее, что нам осталось рассмотреть, — это импорт данных из источников ODBC и установка связи с данными ODBC. Драйверы ODBC являются универсаль-

ным способом доступа к данным различных форматов, обычно серверов баз данных. Если вы имеете драйвер ODBC для определенного источника, вы всегда сможете получить доступ к этим данным. Для этого следует установить драйвер и определить имя источника данных DSN (Data Source Name), который содержит всю необходимую информацию для установки соединения с источником данных. Для этого можно использовать утилиту ODBC Data Source Administrator (Администратор источников данных ODBC), которая запускается из папки Администрирование Панели управления Windows. Диалоговое окно Администратора состоит из нескольких вкладок (рис. 3.39).

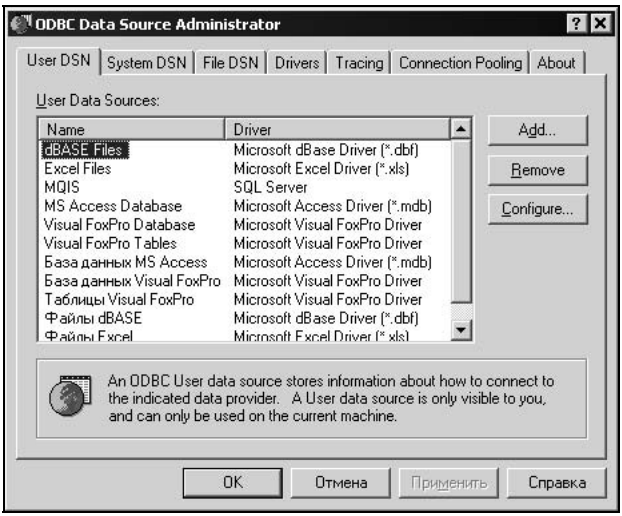


Рис. 3.39. Диалоговое окно Администратора источников данных ODBC

Первые три вкладки соответствуют трем типам источников данных. **User DSN** (Источники данных пользователя) и **System DSN** (Системные источники данных) доступны только на данном компьютере и их описания хранятся в Реестре Windows. Причем **User DSN** доступны только текущему пользователю и видны только ему, **System DSN** доступны всем пользователям на данном компьютере и системным сервисам.

File DSN (Файловые источники данных) хранятся в текстовых файлах с расширением dsn, могут быть размещены в сети и использоваться совместно разными пользователями для доступа к базам данных. Чтобы создать новый источник данных, нужно выбрать соответствующую вкладку и нажать кнопку **Add** (Добавить). Чтобы изменить описание уже существующего источника, нужно выбрать его в соответствующем списке и нажать кнопку **Configure** (Конфигурировать).

При создании нового источника DSN сначала появляется окно, в котором нужно выбрать тип источника, а по сути драйвер для связи с источником. Выберите строку с именем драйвера и нажмите кнопку **Finish** (Готово).

Вид следующего окна зависит от выбранного драйвера, например для Microsoft FoxPro VFP Driver это окно выглядит так, как показано на рис. 3.40.

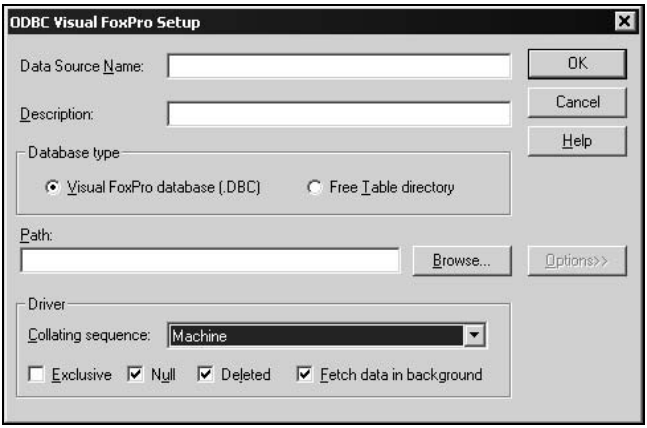


Рис. 3.40. Диалоговое окно настройки источника данных ODBC

В нем необходимо указать имя источника, его описание (пояснение к имени), тип базы данных (это может быть файл базы данных, содержащий все таблицы и представления) или одиночная таблица в файле DBF. В поле **Path** (Путь) нужно указать путь к этой базе данных или таблице. Несколько дополнительных параметров открываются, если нажать кнопку **Options** (Параметры). Это порядок сортировки, режим работы с источником (монопольный), могут ли столбцы содержать Null и т. д. Установив все параметры, нажмите кнопку **OK**. Новый источник данных ODBC будет создан и появится в списке.

При импорте данных из этого источника можно применять соответствующее описание.

1. Откройте базу данных, в которую нужно импортировать данные. Выберите команду **Файл | Внешние данные | Импорт** (File | Get External Data | Import) или раскройте список таблиц и выполните команду контекстного меню **Импорт** (Import).
2. В раскрывающемся списке **Тип файла** (Files of Type) диалогового окна **Импорт** нужно выделить элемент **Базы данных ODBC** (Databases ODBC) и нажать кнопку **Импорт** (Import). После этого откроется диалоговое окно, содержащее список источников данных ODBC (рис. 3.41).
3. Найдите нужный источник и щелкните на этой строке. Затем нажмите кнопку **OK**. Появится диалоговое окно **Импорт объектов** (Import Objects), содержащее либо список таблиц в базе данных, либо, как в данном примере, имя импортируемой таблицы, содержащейся в выбранном файле (рис. 3.42).
4. Выделите в списке объект или несколько объектов и нажмите кнопку **OK**. Данные из выбранных таблиц будут импортированы в таблицы Access.

Аналогично выполняется присоединение таблицы из источника ODBC к базе данных Access. Отличие состоит только в следующем: если импортируемая таблица не имеет уникального индекса, то будет выведено дополнительное окно со списком полей таблицы, в котором будет предложено определить идентифицирующее поле (поля). Присоединенные таблицы из источника данных ODBC выглядят в окне базы данных Access так, как представлено на рис. 3.43.

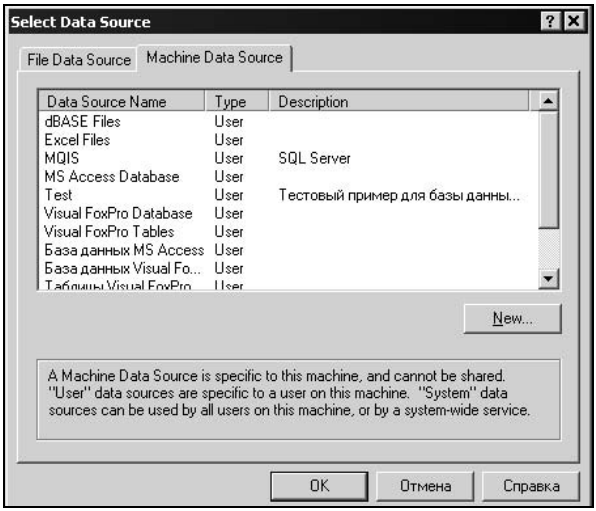


Рис. 3.41. Диалоговое окно выбора источников данных ODBC

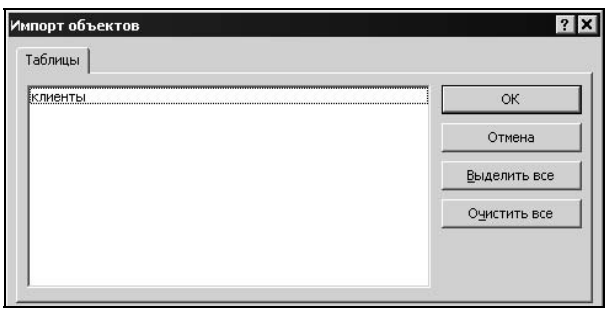


Рис. 3.42. Диалоговое окно **Импорт объектов**

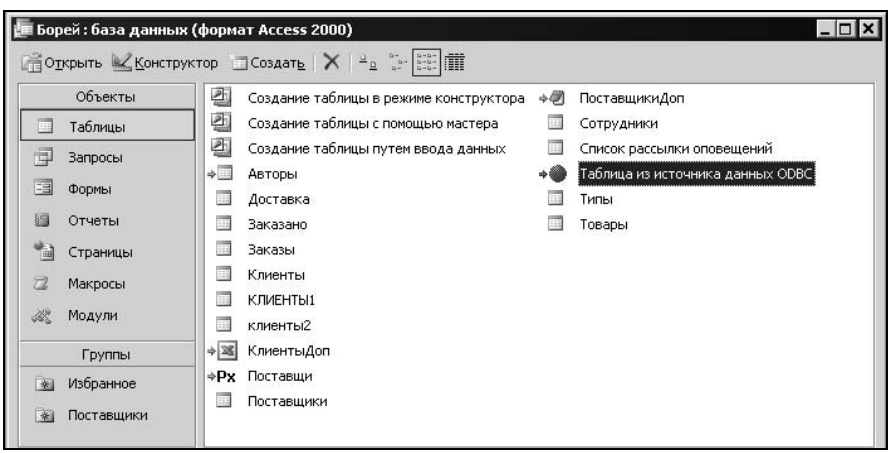


Рис. 3.43. Присоединенная таблица из базы данных ODBC

Обновление связей с присоединенными таблицами

При присоединении таблиц любых форматов Access сохраняет информацию о расположении и структуре внешних файлов. Если изменить местоположение или имя файла, то Access не сможет найти нужную таблицу и появится сообщение об ошибке (рис. 3.44).

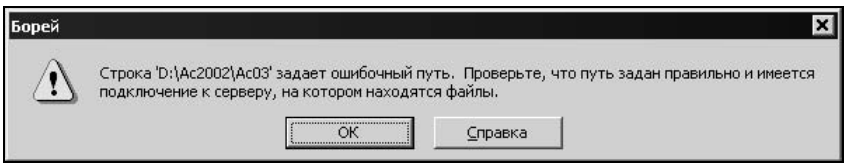


Рис. 3.44. Сообщение об отсутствии внешней таблицы

В этом случае связь нужно обновить. То же самое необходимо сделать, если структура внешней таблицы была изменена, например, было добавлено поле или изменился тип данных какого-нибудь поля.

Обновить связь с таблицей можно двумя способами.

- Первый способ состоит в том, что нужно просто удалить присоединенную таблицу из базы данных, а потом присоединить вновь. При этом не беспокойтесь — удаляя таблицу из окна базы данных, вы просто удаляете связь с таблицей, сама же таблица не стирается.

Примечание

Вы можете легко проверить, в каком файле содержится присоединенная таблица. Для этого достаточно навести указатель мыши на таблицу, и вы увидите всплывающую подсказку, в которой указан тип файла и полный путь к нему (рис. 3.45).

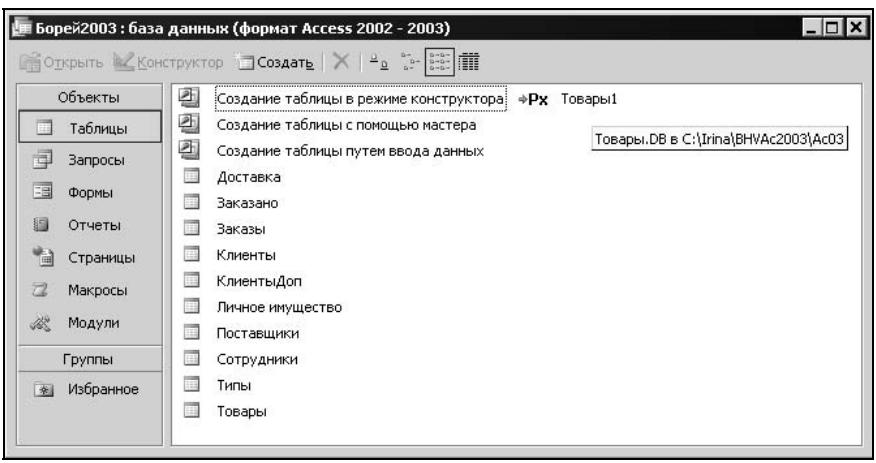


Рис. 3.45. Всплывающая подсказка о местоположении внешней таблицы

- Второй способ обновления связи с внешней таблицей — это использование Диспетчера связанных таблиц (Linked Table Manager).

Чтобы воспользоваться Диспетчером связанных таблиц:

1. Выберите команду меню **Сервис | Служебные программы | Диспетчер связанных таблиц** (Tools | Database Utilities | Linked Table Manager).
2. В появившемся диалоговом окне **Диспетчер связанных таблиц** (Linked Table Manager, рис. 3.46) выводится список всех присоединенных таблиц, а также полный путь и имя файла таблицы. Установите флажки слева у тех таблиц, файлы которых были перемещены. Если все присоединенные таблицы размещаются в одном внешнем MDB-файле или в одной папке, нужно выбрать сразу все таблицы, для чего нажмите кнопку **Выделить все** (Select All). Когда все флажки будут установлены, нажмите кнопку **OK**.

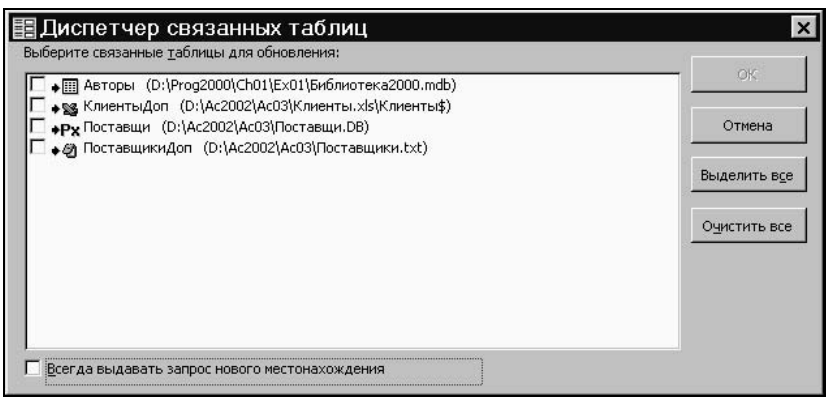


Рис. 3.46. Диалоговое окно **Диспетчер связанных таблиц**

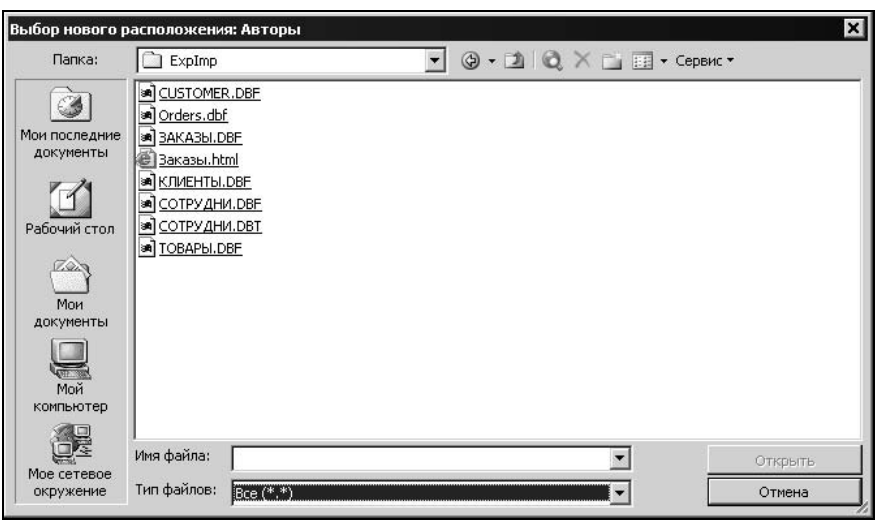


Рис. 3.47. Диалоговое окно **Выбор нового расположения**

3. Появится диалоговое окно **Выбор нового расположения** (Select New Location of), изображенное на рис. 3.47. Откройте папку, в которую перемещен файл таблиц, и выделите его. Нажмите кнопку **Открыть** (Open). Если связь будет успешно обновлена, то Access выведет окно с соответствующим сообщением.
4. Нажмите кнопку **ОК**, чтобы закрыть окно сообщения, а затем кнопку **Закрыть** (Close), чтобы закрыть окно Диспетчера связанных таблиц.

Диспетчер связанных таблиц можно использовать не только в том случае, когда файл был перемещен или переименован, но и тогда, когда была изменена структура внешней таблицы. После этого связь должна быть обязательно обновлена. В этом случае диалоговое окно **Выбор нового расположения** (Select New Location of) открываться не будет. Чтобы оно открывалось всегда, в диалоговом окне **Диспетчер связанных таблиц** (Linked Table Manager) можно установить флажок **Всегда выдавать запрос нового местонахождения** (Always Prompt for New Location).

Экспорт данных в другие форматы

Возможности экспорта данных также расширены в новой версии Access по сравнению с предыдущими версиями. Экспорт из таблиц базы данных может осуществляться в любой из форматов, для которых поддерживается операция импорта данных. Но экспортировать данные Access можно не только из таблицы, но и из запроса. Можно экспортировать не только данные, но и объекты Access — формы, отчеты. В данной главе мы рассмотрим возможность экспорта именно данных из таблиц Access. Об экспорте других объектов мы будем говорить в главах, посвященных этим объектам.

Экспорт в электронную таблицу или файлы dBASE, Paradox

Для экспорта данных в электронную таблицу Microsoft Excel или Lotus 1-2-3, а также в файлы базы данных dBASE или Paradox можно использовать следующую процедуру:

1. Откройте базу данных Access, из которой нужно экспортировать таблицу.
2. Раскройте список таблиц, щелкнув по ярлыку **Таблицы** на панели объектов окна базы данных, и выберите в этом списке экспортируемый объект.
3. Выберите команду меню **Файл | Экспорт** (File | Export) или команду **Экспорт** (Export) из контекстного меню. Появится диалоговое окно **Экспорт объекта** (Export Table), изображенное на рис. 3.48.
4. В раскрывающемся списке **Тип файла** (Save as Type) этого диалогового окна выберите тип файла, в который вы хотите экспортировать таблицу Access. Найдите папку, в которую нужно сохранить экспортируемый файл, а в текстовом поле **Имя файла** (File Name) укажите имя файла. При экспорте в таблицу Excel формата Excel 5—7 или Excel 97—2003 можно установить еще два флажка: **Сохранить формат** (Save formatted) и **Автозагрузка** (Autostart) (второй флажок становится доступным, только если вы установили первый). Нажмите кнопку **Экспорт** (Export).

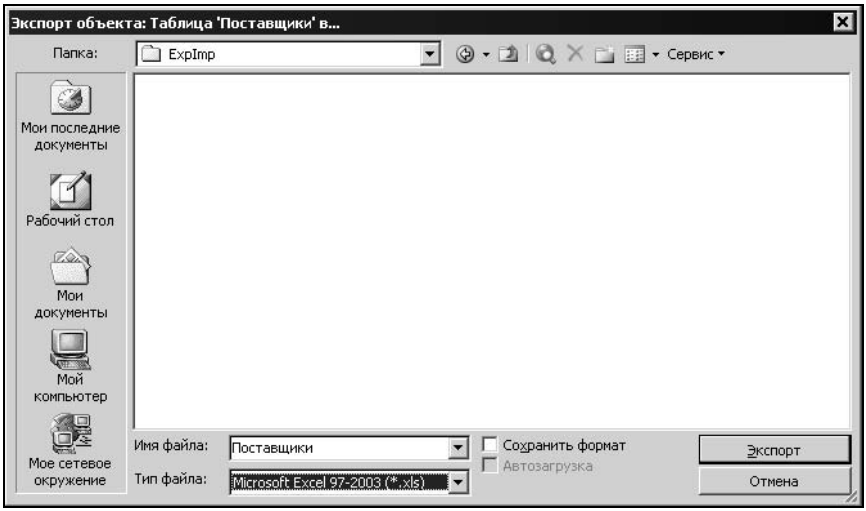


Рис. 3.48. Диалоговое окно **Экспорт объекта**

5. Если процедура экспорта закончится успешно, Access создаст новый файл, с которым вы сможете работать с помощью электронной таблицы или соответствующей СУБД. Если был установлен флажок **Автозагрузка** (Autostart) файл сразу загружается и можно видеть результат экспорта.

Замечание

СУБД Paradox и dBASE не позволяют иметь такие длинные имена полей, как Access, поэтому в процессе импорта длинные имена полей усекаются. Это в некоторых случаях может привести к ошибке, т. к. в результате усечения имена полей могут оказаться дублирующимися. Поэтому перед экспортом лучше создать временную таблицу, в которой изменить имена полей так, чтобы они удовлетворяли требованиям соответствующей СУБД, и затем выполнить экспорт именно из временной таблицы. Для экспорта данных в формат Paradox 8.0 или dBASE 7 необходимо установить на компьютере Borland Database Engine (BDE) версии 4.*x* или выше. Или нужно получить в службе технической поддержки Microsoft специальные драйверы ISAM, которые не требуют наличия BDE.

При экспорте в таблицу Excel форматов Excel 5—7 или Excel 97—2003 можно экспортировать не только всю таблицу, но и отдельный ее фрагмент. Для этого надо сначала открыть таблицу, выделить в ней нужный фрагмент, а потом уже выполнить команду меню **Файл | Экспорт** (File | Export) (в контекстном меню соответствующей команды нет). При этом в окне **Экспорт объекта** (Export Table) нужно установить флажок **Сохранить формат** (Save formatted). Тогда вы получите возможность выбрать один из вариантов экспорта — при нажатии на кнопку **Экспорт всех** (Export All) будет экспортирована вся таблица, а при выборе из списка на этой кнопке команды **Сохранить фрагмент** (Save Selection) только выделенный фрагмент.

Экспорт данных в текстовый файл

Экспорт данных из таблицы Access в текстовый формат применяется тогда, когда приложение, которому нужны эти данные, может импортировать только текстовые

файлы. Для экспорта могут применяться как файлы с разделителями, так и файлы с фиксированной шириной полей. Процедура экспорта похожа на описанную ранее процедуру экспорта в другие типы файлов:

1. Откройте базу данных Access, из которой требуется экспортировать таблицу. Активизируйте окно базы данных, щелкнув по его заголовку левой кнопкой мыши.
2. Раскройте список таблиц и выделите в нем нужную таблицу.
3. Выберите команду **Файл | Экспорт** (File | Export).
4. В раскрывающемся списке **Тип файла** (Save as Type) диалогового окна **Экспорт объекта** (Export Table) выберите элемент **Текстовые файлы** (Text Files). Найдите папку, в которой нужно сохранить экспортируемый файл, в текстовом поле **Имя файла** (File Name) укажите имя файла и нажмите кнопку **Экспорт** (Export).
5. Запустится Мастер экспорта текстов. Первое окно Мастера показано на рис. 3.49. Выберите тип текстового файла: с разделителями или с фиксированной шириной полей. Нажмите кнопку **Далее** (Next).

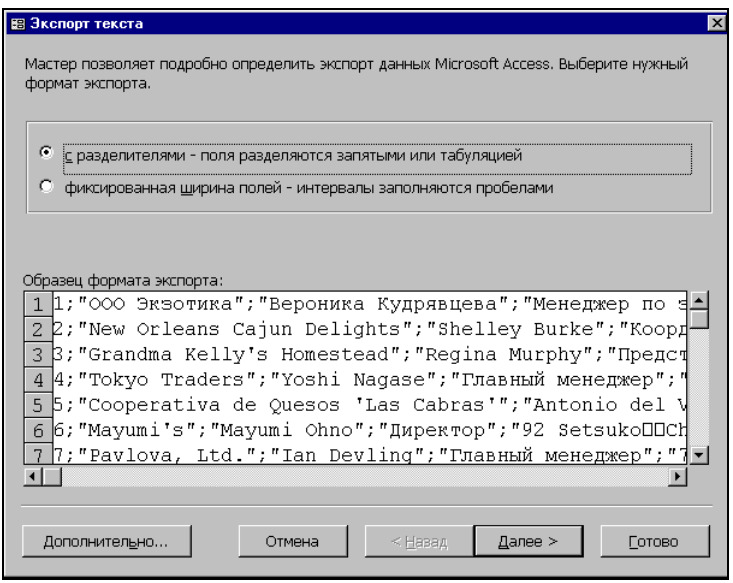


Рис. 3.49. Первое диалоговое окно **Экспорт текста**

6. Если вы выбрали текстовый файл с разделителями, то в следующем диалоговом окне можно будет выбрать символы для разделителей, включить в первую строку текстового файла имена полей таблицы и указать, нужно ли текстовые поля заключать в кавычки. По умолчанию разделителем будет точка с запятой (;), наименования полей не включаются в файл и текстовые поля заключаются в двойные кавычки (рис. 3.50). Если вы выбрали текстовый файл с фиксированной шириной полей, то в следующем окне предлагается настроить разделение полей (по умолчанию Access предлагает разделение полей в соответствии с данными в таблице). Нажмите кнопку **Далее** (Next).

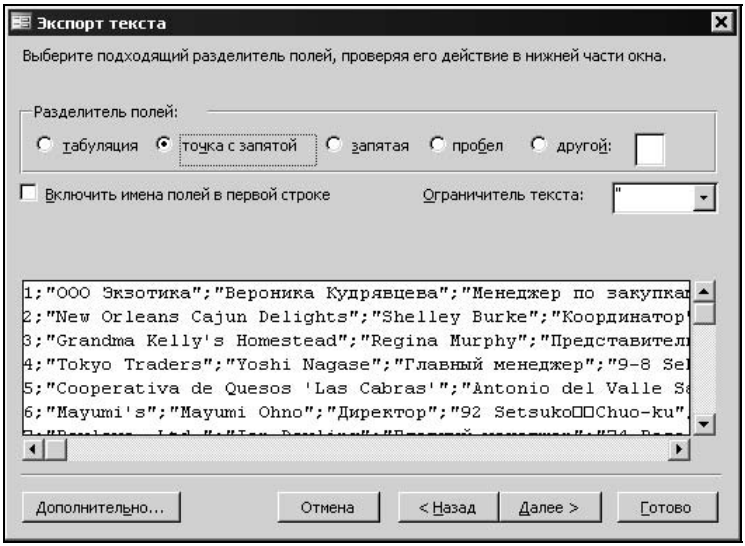


Рис. 3.50. Второе диалоговое окно **Экспорт текста** для текстового файла с разделителями

7. В последнем диалоговом окне Мастера вы определяете имя текстового файла. После нажатия кнопки **Готово** (Finish), при успешном выполнении процедуры, вы получите текстовый файл выбранного формата. Порядок записей в созданном путем экспорта текстовом файле определяется первичным ключом таблицы. Если у экспортируемой таблицы нет первичного ключа, записи экспортируются в том порядке, в котором они были добавлены в таблицу. На рис. 3.51 показан текстовый файл с разделителями, экспортированный из таблицы `Поставщики`.

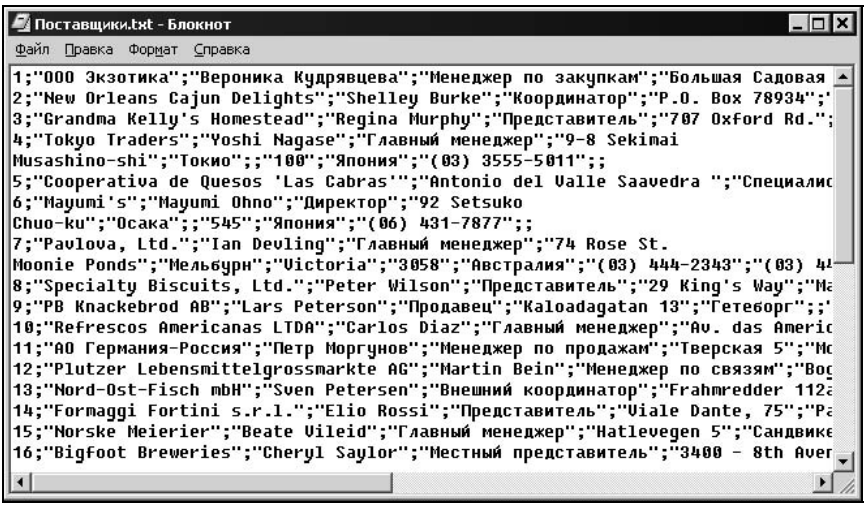
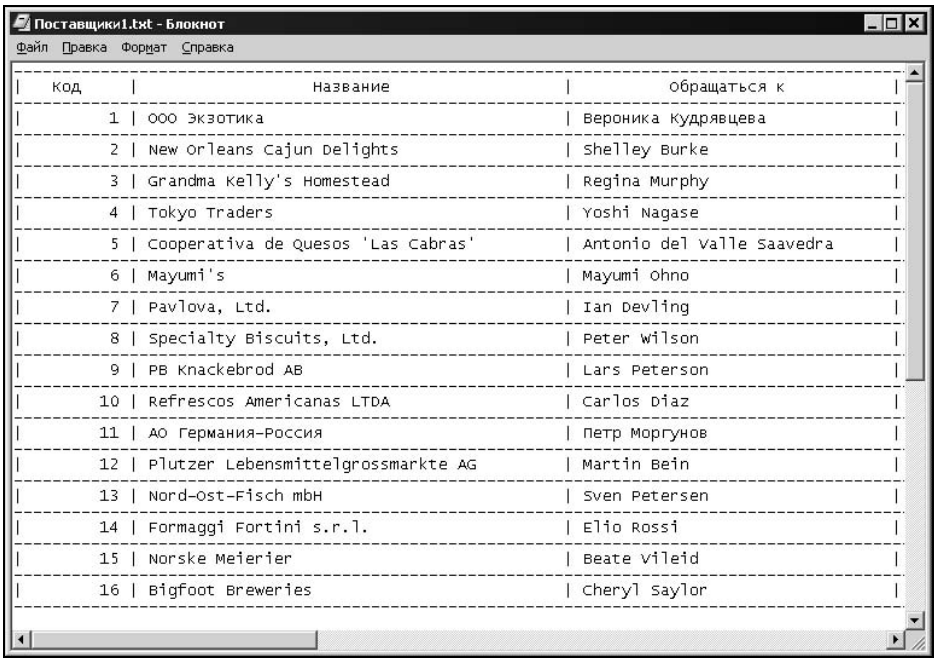


Рис. 3.51. Текстовый файл Поставщики, открытый в приложении Блокнот

Замечание

Использование символа новой строки в текстовых полях таблицы Access 2003 может привести к возникновению ошибок при экспорте данных в другие приложения.



Поставщики1.txt - Блокнот

Файл Правка Формат Справка

Код	Название	Обращаться к
1	ООО Экзотика	Вероника Кудрявцева
2	New Orleans Cajun Delights	Shelley Burke
3	Grandma Kelly's Homestead	Regina Murphy
4	Tokyo Traders	Yoshi Nagase
5	Cooperativa de Quesos 'Las Cabras'	Antonio del Valle Saavedra
6	Mayumi's	Mayumi Ohno
7	Pavlova, Ltd.	Ian Devling
8	Specialty Biscuits, Ltd.	Peter Wilson
9	PB Knackebrod AB	Lars Peterson
10	Refrescos Americanas LTDA	Carlos Diaz
11	АО Германия-Россия	Петр Моргунов
12	Plutzer Lebensmittelgrossmarkte AG	Martin Bein
13	Nord-Ost-Fisch mbH	Sven Petersen
14	Formaggi Fortini s.r.l.	Elio Rossi
15	Norske Meierier	Beate Vileid
16	Bigfoot Breweries	Cheryl Saylor

Рис. 3.52. Текстовый файл Поставщики, экспортированный с сохранением формата

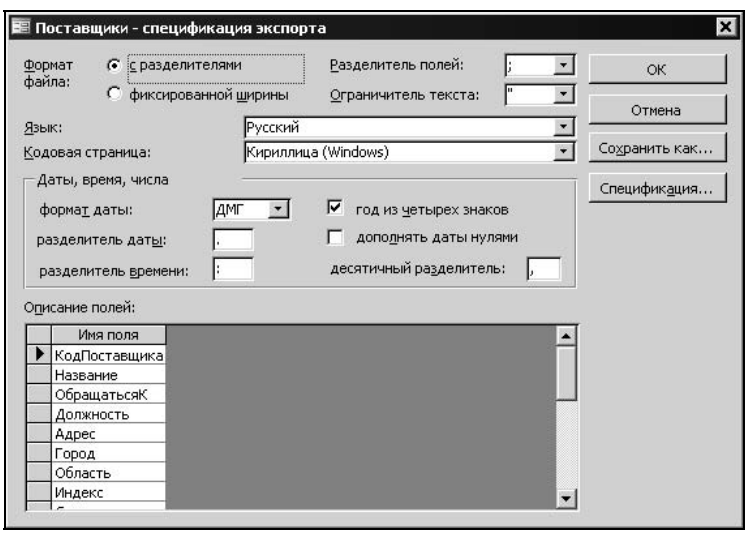


Рис. 3.53. Диалоговое окно спецификации экспорта

Можно экспортировать таблицу в текстовый файл с сохранением формата. Для этого в диалоговом окне **Экспорт объекта** (Export Table), выбрав тип файла **Текстовые файлы** (Text Files), нужно установить флажок **Сохранить формат** (Save formatted) и нажать кнопку **Экспорт** (Export). Далее в диалоговом окне **Кодировка для файла** (Encode As) можно выбрать тип кодировки — Microsoft Windows, MS-DOS, Unicode, Unicode (UTF-8). В результате будет создан текстовый файл с таблицей, вид которой представлен на рис. 3.52.

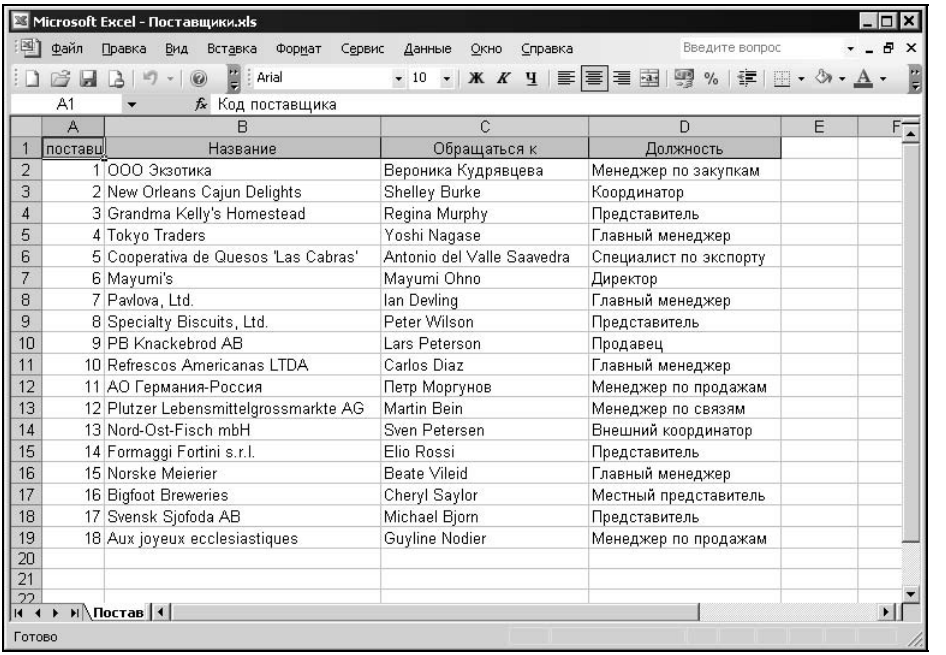
Так же, как и при импорте из текстовых файлов, можно создать Спецификации экспорта. Для этого в процессе экспорта в любом окне мастера нажмите кнопку **Дополнительно** (Advanced). Появится окно, похожее на окно для спецификаций импорта, только при экспорте вы не можете изменить тип поля, пропустить какие-то поля (рис. 3.53). Параметры, указываемые в спецификации те же, что и в Спецификации импорта (см. табл. 3.5).

Быстрый экспорт данных в другие приложения Microsoft Office

Access позволяет быстро экспортировать данные из таблицы в файл Microsoft Excel. Выполняется это с помощью команды **Связи с Office** (Office Links). При этом можно экспортировать как всю таблицу, так и ее часть. Если нужно экспортировать всю таблицу, выделите эту таблицу в списке таблиц окна базы данных и нажмите стрелку справа от кнопки **Связи с Office** (Office Links) на панели инструментов. Выберите из списка команду **Анализ в MS Excel** (Analyze It with Microsoft Excel). Access создаст новый файл с именем *<имя таблицы>*.xls, скопирует в него данные из выделенной таблицы, запустит приложение Excel и откроет в нем этот файл. Аналогично можно экспортировать только часть таблицы Access, например подмножество строк или даже диапазон ячеек. Для этого необходимо открыть таблицу в режиме Таблицы, выбрать подмножество строк, например с помощью фильтра, или выделить диапазон ячеек. Кнопки **Связи с Office** (Office Links) на панели инструментов в этом режиме нет, но можно выполнить команду меню **Сервис | Связи с Office | Анализ в Microsoft Office Excel** (Tools | Office Links | Analyze It with Microsoft Office Excel), и выбранные данные будут экспортированы в Excel. На рис. 3.54 показан диапазон ячеек таблицы `Поставщики`, который мы таким способом экспортировали в Excel.

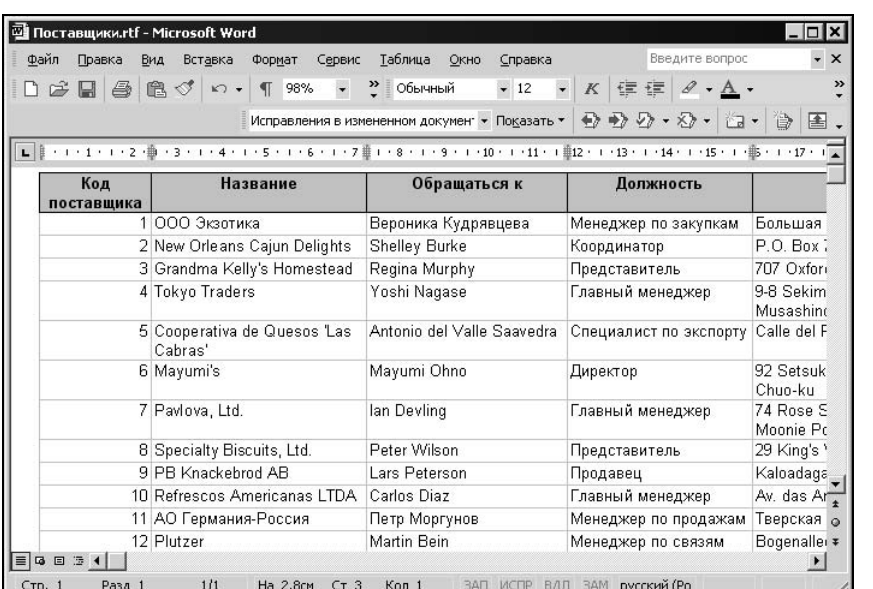
Так же легко можно экспортировать таблицу Access (целиком или часть) в таблицу Word. Для этого нужно выбрать таблицу, открыть ее и выделить диапазон строк, ячеек, столбцов, если требуется выполнить экспорт только части таблицы и выполнить команду меню **Сервис | Связи с Office | Публикация в Microsoft Office Word** (Tools | Office Links | Publish It with Microsoft Office Word). Будет создан документ Word в формате RTF, в нем создана таблица с данными из таблицы Access и документ будет открыт. Пример результирующей таблицы приведен на рис. 3.55. Экспортировалась та же таблица `Поставщики`. Поля типа MEMO и Гиперссылка экспортируются корректно, а вот рисунки и другие объекты в поле OLE в файл RTF не передаются.

Третья команда в группе команд **Связи с Office** (Office Links) — **Слияние с Microsoft Office Word** (Merge It with Microsoft Office Word) — используется для создания составных документов. Примерами таких документов являются письма, конверты, наклейки и т. д. В таблице Access могут храниться адреса клиентов для рассылки, а



	A	B	C	D	E	F
1	поставщ	Название	Обращаться к	Должность		
2	1	ООО Экзотика	Вероника Кудрявцева	Менеджер по закупкам		
3	2	New Orleans Cajun Delights	Shelley Burke	Координатор		
4	3	Grandma Kelly's Homestead	Regina Murphy	Представитель		
5	4	Tokyo Traders	Yoshi Nagase	Главный менеджер		
6	5	Cooperativa de Quesos 'Las Cabras'	Antonio del Valle Saavedra	Специалист по экспорту		
7	6	Mayumi's	Mayumi Ohno	Директор		
8	7	Pavlova, Ltd.	Ian Devling	Главный менеджер		
9	8	Specialty Biscuits, Ltd.	Peter Wilson	Представитель		
10	9	PB Knackebrod AB	Lars Peterson	Продавец		
11	10	Refrescos Americanas LTDA	Carlos Diaz	Главный менеджер		
12	11	АО Германия-Россия	Петр Моргунов	Менеджер по продажам		
13	12	Plutzer Lebensmittelgrossmarkte AG	Martin Bein	Менеджер по связям		
14	13	Nord-Ost-Fisch mbH	Sven Petersen	Внешний координатор		
15	14	Formaggi Fortini s.r.l.	Elio Rossi	Представитель		
16	15	Norske Meierier	Beate Vileid	Главный менеджер		
17	16	Bigfoot Breweries	Cheryl Saylor	Местный представитель		
18	17	Svensk Sjofoda AB	Michael Bjorn	Представитель		
19	18	Aux joyeux ecclesiastiques	Guyline Nodier	Менеджер по продажам		
20						
21						
22						

Рис. 3.54. Диапазон ячеек таблицы Access после экспорта в Excel



Код поставщика	Название	Обращаться к	Должность	
1	ООО Экзотика	Вероника Кудрявцева	Менеджер по закупкам	Большая
2	New Orleans Cajun Delights	Shelley Burke	Координатор	P.O. Box 7
3	Grandma Kelly's Homestead	Regina Murphy	Представитель	707 Oxfor
4	Tokyo Traders	Yoshi Nagase	Главный менеджер	9-8 Sekim Musashin
5	Cooperativa de Quesos 'Las Cabras'	Antonio del Valle Saavedra	Специалист по экспорту	Calle del F
6	Mayumi's	Mayumi Ohno	Директор	92 Setsuk Chuo-ku
7	Pavlova, Ltd.	Ian Devling	Главный менеджер	74 Rose S Moonie Po
8	Specialty Biscuits, Ltd.	Peter Wilson	Представитель	29 King's
9	PB Knackebrod AB	Lars Peterson	Продавец	Kaloadaga
10	Refrescos Americanas LTDA	Carlos Diaz	Главный менеджер	Av. das Ar
11	АО Германия-Россия	Петр Моргунов	Менеджер по продажам	Тверская
12	Plutzer	Martin Bein	Менеджер по связям	Bogenalle

Рис. 3.55. Таблица Access после экспорта в документ Word

документ Word представляет собой посылаемый документ. В процессе слияния можно вставить в документ из таблицы Access имя клиента, его адрес или другую нужную информацию. После установления связи с источником данных (в данном случае таблицей Access) документ может быть в любое время открыт в Microsoft Word и использован для печати стандартных писем или почтовых наклеек. При этом в документ будут попадать всегда актуальные данные из таблицы, т. к. запрос к этим данным будет выполняться в момент печати документа. Подробнее создание документов слияния мы будем рассматривать в *гл. 15*.

Замечание

Создание составных документов Word возможно только при работе с Microsoft Word версии 7.0/95 или выше.

Существуют еще несколько способов использования данных Microsoft Access в Microsoft Word.

- При работе с любой версией Microsoft Word допускается экспорт данных из Microsoft Access в файл данных составного документа, который будет в этом случае содержать статические данные, т. е. мгновенный снимок данных на момент экспорта.
- Допускается сохранение данных в файле формата RTF. При этом сохраняются такие атрибуты форматирования, как шрифты и стили. Файл RTF может быть открыт в Microsoft Word, в других текстовых процессорах на платформе Windows или в издательских программах.
- Можно экспортировать данные из Access в файл XML, который затем просто читается приложением Word 2003.

Для того чтобы осуществить экспорт таблицы Access в файл данных составного документа Microsoft Word, необходимо:

1. В окне базы данных выбрать имя экспортируемого запроса, а затем выполнить команду **Экспорт** (Export) из меню **Файл** (File).
2. В окне **Экспорт объекта** (Export Object) выбрать тип файла **Microsoft Word Merge** в поле **Тип файла** (File Type), а в поле **Имя файла** (File name) ввести имя файла и нажать кнопку **Экспорт** (Export).

Microsoft Access автоматически создает файл данных, содержащий имена полей и все данные из таблицы. Пример такого файла, полученного путем экспорта таблицы `Поставщики`, приведен на рис. 3.56.

Чтобы выполнить экспорт таблицы Access в файл формата RTF, необходимо:

1. В окне базы данных выбрать экспортируемую таблицу, а затем выполнить команду **Экспорт** (Export) из меню **Файл** (File) или из контекстного меню.
2. В окне **Экспорт объекта** (Export Object) в поле **Тип файла** (File Type) выбрать тип файла **Формат RTF** (Rich Text Fiormat), а в поле **Имя файла** (File name) ввести имя файла. Можно также установить флажок **Автозагрузка** (Autostart), чтобы сразу увидеть результат экспорта. Затем нажать кнопку **Экспорт** (Export).

Будет создан файл формата RTF, который содержит таблицу с полями, соответствующими полям таблицы Access, точно такую же, как при выполнении команды

меню **Сервис | Связи с Office | Публикация в Microsoft Office Word** (Publish It with Microsoft Office Word) (см. рис. 3.55).

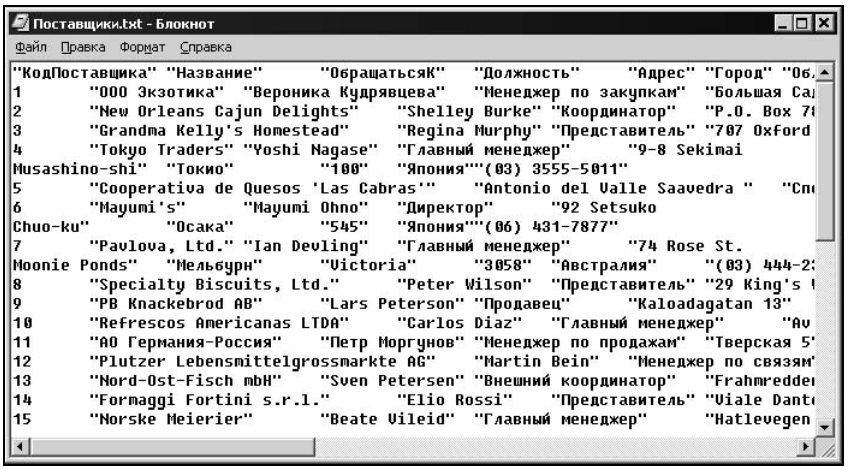


Рис. 3.56. Файл данных составного документа Word

Экспорт данных в форматы объектов WWW

Access 2003 предоставляет очень широкий выбор форматов объектов WWW (World Wide Web), в которые можно экспортировать данные. Во-первых, это статические страницы в формате HTML. Полученные таким образом документы будут являться мгновенными снимками таблиц на момент их экспорта. Кроме того, таблицы Access 2003 можно экспортировать либо в файлы формата IDC/HTX, либо в файлы формата ASP. И в этом случае отображаемые программой просмотра Интернет данные будут актуальными, однако их обновление не может производиться средствами программы просмотра. Можно формировать из таблиц Access документы XML для обмена с другими приложениями вместе со схемами данных и файлами представления данных. Можно экспортировать данные из таблиц Access списка служб Microsoft SharePoint Services. Более подробные сведения об экспорте данных в форматы объектов WWW приведены в *гл. 11.*

Использование Буфера обмена Windows для импорта и экспорта данных

При импорте или экспорте данных, созданных с помощью приложения Windows, можно воспользоваться Буфером обмена Windows. Это средство позволяет применять стандартные команды Windows и быстро перемещать данные из одного приложения Windows в другое. Это особенно удобно при копировании данных между приложениями Microsoft Office.

Однако этот метод имеет одно существенное ограничение — структура таблицы, в которую импортируются (копируются) данные, должна соответствовать данным,

передаваемым через Буфер обмена. Например, для правильной вставки данных из строк рабочего листа Excel в таблицу Access необходимо, чтобы типы данных полей таблицы Access соответствовали типам данных каждого столбца, скопированного в буфер обмена диапазона ячеек.

Чтобы добавить новые записи в таблицу Access 2003 при помощи Буфера обмена Windows:

1. Запустите приложение, которое будет использоваться для копирования данных в буфер обмена (в нашем примере это Microsoft Excel), и откройте файл с данными.
2. Выделите данные, которые нужно добавить в таблицу Access. Например, выделите диапазон ячеек рабочего листа, начиная со столбца, соответствующего первому полю таблицы Access (рис. 3.57). При этом не обязательно выделять все оставшиеся столбцы, соответствующие полям таблицы Access, поскольку Access добавит пустые значения или значения по умолчанию в поля, соответствующие столбцам, не включенным в выделенный диапазон.

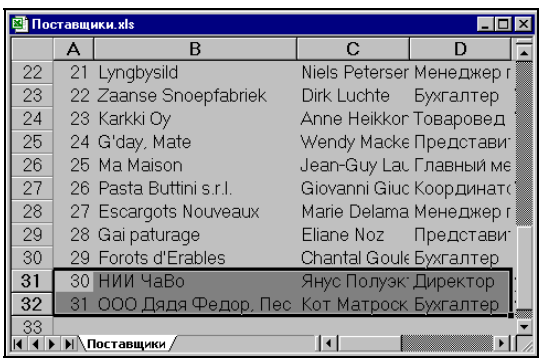


Рис. 3.57. В таблицу Access будет добавлен выделенный диапазон ячеек

3. Выберите команду **Правка | Копировать** (Edit | Copy) или нажмите комбинацию клавиш <Ctrl>+<C>, чтобы скопировать выделенный диапазон ячеек в Буфер обмена Windows.
4. Откройте таблицу, в которую нужно добавить данные, в режиме Таблицы.

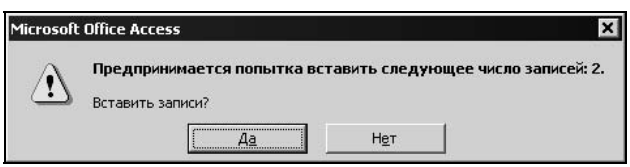


Рис. 3.58. Подтвердите вставку записей из буфера обмена

5. Выберите команду **Правка | Добавить из буфера** (Edit | Paste Append). Если в процессе добавления записей не возникнет ошибок, появится окно, запрашивающее подтверждение на добавление записей в таблицу (рис. 3.58). Нажмите кнопку **Да** (Yes). Новые записи будут добавлены в конец таблицы Access (рис. 3.59). Чтобы

разместить добавленные записи в правильной последовательности, выберите команду **Записи | Удалить фильтр** (Records | Remove Filter/Sort).



Поставщики : таблица

	Код поставщика	Название	Обращаться к	Должность
+	18	Aux joyeux ecclesiastiques	Guyline Nodier	Менеджер по продажам
+	19	New England Seafood Cannery	Robb Merchant	Агент по продажам
+	20	Leka Trading	Chandra Leka	Совладелец
+	21	Lyngbysild	Niels Petersen	Менеджер по продажам
+	22	Zaanse Snoepfabriek	Dirk Luchte	Бухгалтер
+	23	Karkki Oy	Anne Heikkonen	Товаровед
+	24	G'day, Mate	Wendy Mackenzie	Представитель
+	25	Ma Maison	Jean-Guy Lauzon	Главный менеджер
+	26	Pasta Buttini s.r.l.	Giovanni Giudici	Координатор
+	27	Escargots Nouveaux	Marie Delamare	Менеджер по продажам
+	28	Gai paturage	Eliane Noz	Представитель
+	29	Forots d'Erables	Chantal Goulet	Бухгалтер
+	30	НИИ ЧаВо	Янус Полуэктов	Директор
+	31	ООО Дядя Федор	Кот Матроскин	Бухгалтер
*	(Счетчик)			

Запись: 30 из 31

Рис. 3.59. Выделенные записи добавлены из Буфера обмена Windows

При добавлении в таблицу Access данных с помощью Буфера обмена Windows могут возникнуть ошибки по одной из двух причин:

- типы данных ячеек Excel не соответствуют типам данных полей таблицы Access;
- в первичном ключе добавляемых записей содержатся значения, которые уже есть в таблице Access.

В любом случае Access создаст таблицу `Ошибки вставки` (Paste Errors) и запишет в нее все записи, вызвавшие ошибки. Использование таблицы `Ошибки вставки` аналогично применению таблицы ошибок импорта данных, описанной ранее в этой главе.

Ошибки, вызванные наличием повторяющихся значений в первичном ключе таблицы Access, приводят к появлению на экране диалоговых окон с сообщениями об ошибках. Читая сообщения в этих окнах, вы можете получить более подробную информацию о каждой ошибке, если нажмете кнопку **Справка** (Help), можете запретить вывод дальнейших сообщений об ошибках, а можете просто прервать процесс добавления записей. В любом случае все ошибки в обработанных записях будут зафиксированы в таблице `Ошибки вставки` (на рис. 3.60 приведена таблица, которая была получена при повторной вставке в базу данных шести уже имеющихся в ней записей).

После завершения добавления записей в таблицу появится сообщение о том, что записи, которые не удалось добавить из буфера обмена, помещены в таблицу `Ошибки вставки`.

Совет

Если в процессе вставки выдается много сообщений об ошибках, вызванных повторяющимися значениями в ключевом поле, лучше прервать этот процесс (реальная вставка записей происходит только после подтверждения в конце процесса), найти

записи с повторяющимися значениями, исправить их и заново выполнить процесс вставки. Для того чтобы быстро найти повторяющиеся данные, можно выполнить запрос специального вида — Повторяющиеся записи. Раскройте список запросов в окне базы данных, нажмите кнопку **Создать** (New), в появившемся диалоговом окне **Новый запрос** (New Query) выделите элемент **Повторяющиеся записи** (Find Duplicates Query Wizard) и нажмите кнопку **OK**. *(Подробную информацию о создании запросов см. в гл. 4.)*



Рис. 3.60. Таблица Ошибки вставки

Замечание

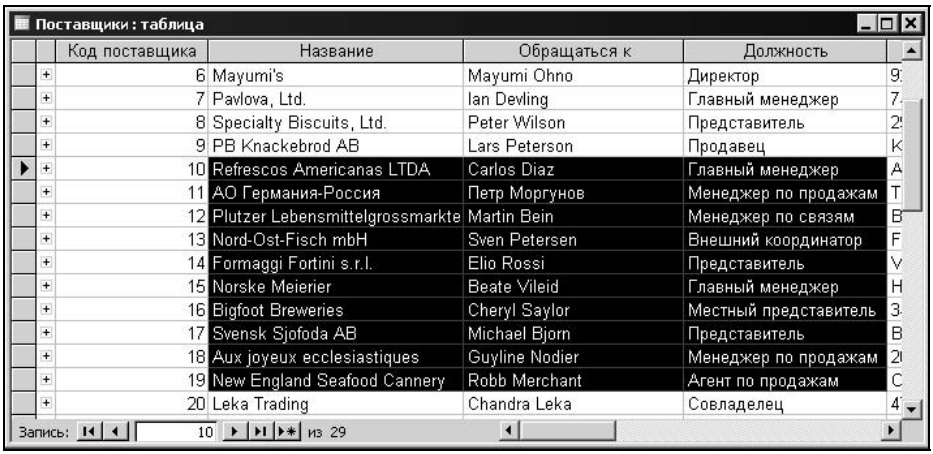
Если записи вставляются в таблицу, ключом которой является поле типа **Счетчик** (AutoNumber), то при добавлении новых записей они перенумеровываются в соответствии с последним значением счетчика. То есть, даже если значения этого поля в добавляемых записях будут совпадать с ключами в уже существующих записях, они будут вставлены, при этом значения ключей в новых записях будут отличаться от тех, что они имели до вставки.

Access позволяет не только добавить в таблицу записи из буфера обмена, но и заменить уже существующие в таблице записи. При этом все столбцы данных, скопированные в буфер обмена, должны соответствовать полям таблицы Access, причем первый столбец обязательно должен соответствовать первому полю. Однако в буфере обмена могут быть не все столбцы таблицы Access. В этом случае данные в полях таблицы, для которых отсутствуют столбцы в буфере обмена, останутся неизменными. А вот число строк в буфере обмена должно соответствовать числу выделенных заменяемых строк в таблице. Это число не должно быть меньше, чем число строк в буфере. В противном случае данные из последних строк буфера будут игнорироваться. Процедура замены записей аналогична описанной ранее процедуре добавления записей, только перед вставкой нужно выделить заменяемые записи в таблице.

Простейшим способом экспорта данных из таблицы Access в другое приложение Windows также является копирование содержимого таблицы Access в буфер обмена с последующей вставкой в документ другого приложения. Например, чтобы скопировать таблицу Access в документ Word через буфер обмена:

1. Откройте нужную таблицу в режиме Таблицы (мы использовали для примера таблицу Поставщики).
2. Выделите в этой таблице необходимый диапазон ячеек, нужное количество записей или все записи. Таблица Поставщики примет вид, изображенный на рис. 3.61.

Если необходимо выделить все поля во всех записях таблицы, выберите команду **Правка | Выделить все записи** (Edit | Select All Records) или щелкните левой кнопкой мыши по серому прямоугольнику в левом верхнем углу окна на пересечении строки заголовков столбцов и столбца выделения записей.



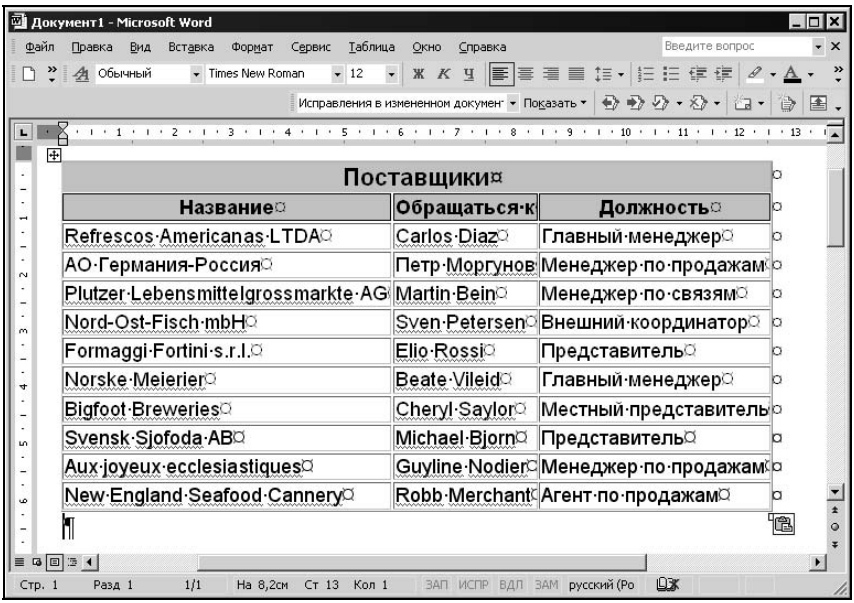
Поставщики : таблица

Код поставщика	Название	Обращаться к	Должность
6	Mayumi's	Mayumi Ohno	Директор
7	Pavlova, Ltd.	Ian Devling	Главный менеджер
8	Specialty Biscuits, Ltd.	Peter Wilson	Представитель
9	PB Knackebrod AB	Lars Peterson	Продавец
10	Refrescos Americanas LTDA	Carlos Diaz	Главный менеджер
11	АО Германия-Россия	Петр Моргунов	Менеджер по продажам
12	Plutzer Lebensmittelgrossmarkte	Martin Bein	Менеджер по связям
13	Nord-Ost-Fisch mbH	Sven Petersen	Внешний координатор
14	Formaggi Fortini s.r.l.	Elio Rossi	Представитель
15	Norske Meierier	Beate Vileid	Главный менеджер
16	Bigfoot Breweries	Cheryl Saylor	Местный представитель
17	Svensk Sjofoda AB	Michael Bjorn	Представитель
18	Aux joyeux ecclesiastiques	Guyline Nodier	Менеджер по продажам
19	New England Seafood Cannery	Robb Merchant	Агент по продажам
20	Leka Trading	Chandra Leka	Совладелец

Запись: 10 из 29

Рис. 3.61. Выделенный диапазон ячеек в таблице `Поставщики`

3. Нажмите комбинацию клавиш <Ctrl>+<C> или выберите команду **Правка | Копировать** (Edit | Copy), чтобы скопировать выделенные данные в буфер обмена.



Документ1 - Microsoft Word

Файл Правка Вид Вставка Формат Сервис Таблица Окно Справка

Поставщики

Название	Обращаться к	Должность
Refrescos Americanas LTDA	Carlos Diaz	Главный менеджер
АО Германия-Россия	Петр Моргунов	Менеджер по продажам
Plutzer Lebensmittelgrossmarkte AG	Martin Bein	Менеджер по связям
Nord-Ost-Fisch mbH	Sven Petersen	Внешний координатор
Formaggi Fortini s.r.l.	Elio Rossi	Представитель
Norske Meierier	Beate Vileid	Главный менеджер
Bigfoot Breweries	Cheryl Saylor	Местный представитель
Svensk Sjofoda AB	Michael Bjorn	Представитель
Aux joyeux ecclesiastiques	Guyline Nodier	Менеджер по продажам
New England Seafood Cannery	Robb Merchant	Агент по продажам

Стр. 1 Разд 1 1/1 На 8,2см Ст 13 Кол 1 ЗАП ИСПР ВДЛ ЗАМ русский (Ро

Рис. 3.62. Данные, импортированные в документ Word в виде таблицы

4. Запустите Microsoft Word и создайте в нем новый документ, выбрав команду **Файл | Создать** (File | New).
5. Выберите команду **Правка | Вставить** (Edit | Paste) или нажмите комбинацию клавиш <Ctrl>+<V>. Данные из буфера обмена будут вставлены в документ Word в виде отформатированной таблицы (рис. 3.62).

Заметьте, что первая строка данных, копируемых Access в буфер обмена, всегда содержит названия столбцов таблицы. Если документ Word создается при помощи нескольких операций копирования, ненужные заголовки столбцов таблицы требуется удалить вручную.

ГЛАВА 4



Отбор и сортировка записей с помощью запросов

Одним из семи стандартных объектов Microsoft Access является *запрос*. Запросы используются для просмотра, анализа и изменения данных в одной или нескольких таблицах. Например, можно использовать запрос для отображения данных из одной или нескольких таблиц, их сортировки в определенном порядке, выполнения вычислений над группой записей, осуществления выборки из таблицы по определенным условиям. Запросы могут служить источником данных для форм и отчетов Microsoft Access. Сам запрос не содержит данных, но позволяет выбирать их из таблиц и выполнять над ними ряд операций. В Microsoft Access существует несколько видов запросов: запросы к серверу, которые используются для выборки данных с сервера; запросы на автоподстановку, автоматически заполняющие поля для новой записи; запросы на выборку, выполняющие выборку данных из таблиц; запросы на изменение, которые дают возможность модифицировать данные в таблицах (в том числе удалять, обновлять и добавлять записи); запросы на создание таблицы, организующие новую таблицу на основе данных одной или нескольких существующих таблиц, а также другие типы запросов. В данной главе основное внимание уделяется запросу на выборку, как наиболее часто применяемому. Будут рассматриваться следующие вопросы:

- Создание запросов с помощью Мастера и Конструктора запросов
- Особенности выполнения запросов и их изменение
- Использование вычисляемых полей в запросе
- Преобразование фильтра в запрос
- Виды соединений таблиц в запросе
- Выполнение запросов с параметрами
- Способы вывода результатов запроса на печать
- Передача результатов запросов в другие приложения
- Использование выражений в запросах

Описание других типов запросов приводится в *гл. 8*.

Запросы и фильтры

Запрос на выборку содержит условия отбора данных и возвращает выборку, соответствующую указанным условиям, без изменения возвращаемых данных. В Access существует также понятие *фильтра*, который в свою очередь является набором условий, позволяющих отбирать подмножество записей или сортировать их. Сходство между запросами на выборку и фильтрами заключается в том, что и в тех и в других производится извлечение подмножества записей из базовой таблицы или запроса. Однако между ними существуют различия, которые нужно понимать, чтобы правильно сделать выбор, в каком случае использовать запрос, а в каком — фильтр.

Основные отличия запросов и фильтров заключаются в следующем:

- фильтры не позволяют в одной строке отображать данные из нескольких таблиц, т. е. объединять таблицы;
- фильтры не дают возможности указывать поля, которые должны отображаться в результирующем наборе записей, они всегда отображают все поля базовой таблицы;
- фильтры не могут быть сохранены как отдельный объект в окне базы данных (они сохраняются только в виде запроса);
- фильтры не позволяют вычислять суммы, средние значения, подсчитывать количество записей и находить другие итоговые значения.

Запросы могут использоваться только с закрытой таблицей или запросом. Фильтры обычно применяются при работе в режиме Формы или в режиме Таблицы для просмотра или изменения подмножества записей. Запрос можно использовать:

- для просмотра подмножества записей таблицы без предварительного открытия этой таблицы или формы;
- для того чтобы объединить в виде одной таблицы на экране данные из нескольких таблиц;
- для просмотра отдельных полей таблицы;
- для выполнения вычислений над значениями полей.

Создание простого запроса с помощью Мастера

Наиболее просто создается запрос при помощи Мастера запросов. Чтобы создать такой запрос необходимо:

1. В окне базы данных на панели объектов выбрать ярлык **Запросы** (Queries).
2. В списке запросов дважды щелкнуть левой кнопкой мыши на ярлыке **Создание запроса с помощью мастера** (Create query by using wizard) или нажать кнопку **Создать** (New) в окне базы данных, и в появившемся диалоговом окне **Новый запрос** (New Query) выбрать **Простой запрос** (Simple Query Wizard) и нажать кнопку **OK** (рис. 4.1).

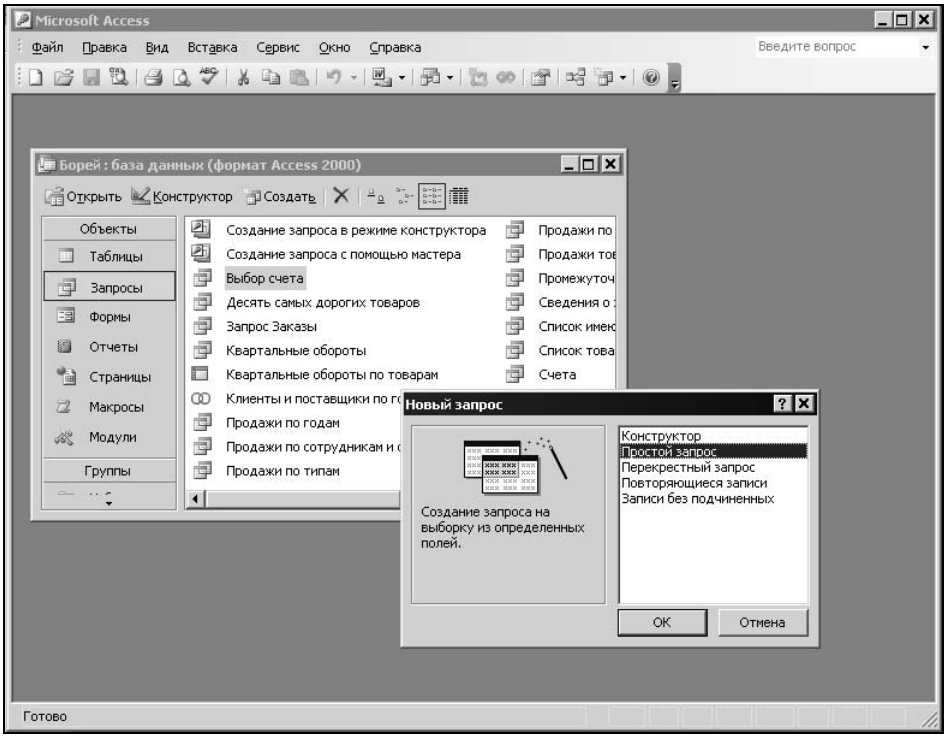


Рис. 4.1. Окно создания нового запроса

3. В появившемся окне **Создание простых запросов** (Simple Query Wizard) (рис. 4.2) в поле со списком **Таблицы и запросы** (Tables/Queries) выбрать таблицу или запрос, которые будут служить источником данных для создаваемого запроса.
4. С помощью стрелок вправо и влево переместить из списка **Доступные поля** (Available Fields) в список **Выбранные поля** (Selected Fields) те поля, которые необходимо иметь в конструируемом запросе. При этом порядок полей в списке **Выбранные поля** (Selected Fields) должен соответствовать тому порядку, который вы хотите видеть в запросе. Если нужно включить в запрос все поля, можно воспользоваться кнопкой с двумя стрелками вправо.
5. Нажать кнопку **Далее** (Next).
6. Следующее диалоговое окно будет последним. В нем нужно ввести имя создаваемого запроса (рис. 4.3) в поле **Задайте имя запроса** (What title do you want to your query?) и выбрать дальнейшие действия: **Открыть запрос для просмотра данных** (Open the query to view information) или **Изменить макет запроса** (Modify the query design).
7. При необходимости можно установить флажок **Вывести справку по работе с запросом?** (Display Help on working with the query?) для вывода справочной информации по работе с запросами.
8. Нажать кнопку **Готово** (Finish).

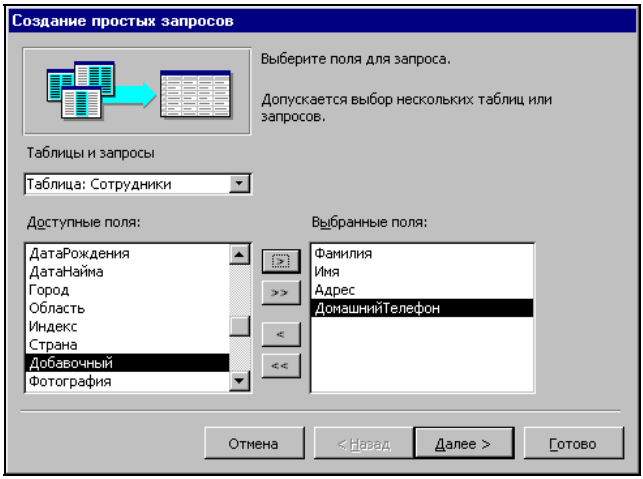


Рис. 4.2. Первое диалоговое окно Мастера простых запросов

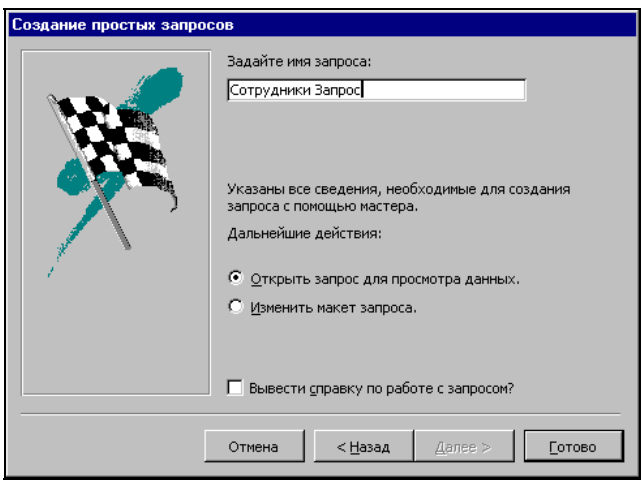


Рис. 4.3. Окно Мастера простых запросов на втором шаге

По окончании работы Мастера простых запросов в зависимости от выбора способа дальнейшей работы с запросом откроется или окно запроса в режиме просмотра (рис. 4.4), или окно Конструктора запросов, в котором можно модифицировать запрос. В качестве примера построения простого запроса с помощью Мастера простых запросов рассматривается создание запроса, содержащего имена, фамилии, адреса и телефоны сотрудников фирмы Борей (Northwind) (в качестве исходных данных взяты таблицы демонстрационной базы данных Борей). Этот запрос строится на основе таблицы Сотрудники (Employees). На первом шаге Мастера простых запросов была выбрана исходная таблица Сотрудники (Employees) и в список **Выбранные поля** (Selected Fields) были перенесены следующие поля: Имя (First Name), Фамилия (Last

Name), Адрес (Address), Домашний телефон (Home Phone). На втором шаге Мастера в поле названия запроса было введено имя Сотрудники Запрос и выбран способ дальнейшего отображения запроса: **Открыть запрос для просмотра данных**. Результатом работы Мастера стал запрос Сотрудники Запрос, изображенный на рис. 4.4.

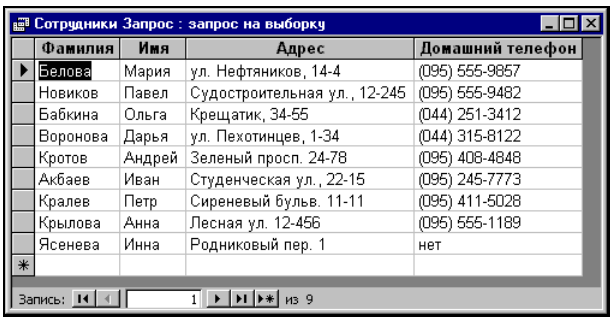


Рис. 4.4. Окно запроса в режиме просмотра

Создание и изменение запроса с помощью Конструктора

Для изменения уже существующих запросов и для создания новых запросов используется Конструктор запросов. Для того чтобы открыть запрос в режиме Конструктора, выделите в списке один из существующих запросов, например, только что созданный запрос Сотрудники Запрос, и нажмите кнопку **Конструктор** (Design) на панели инструментов окна **База данных** (Database).

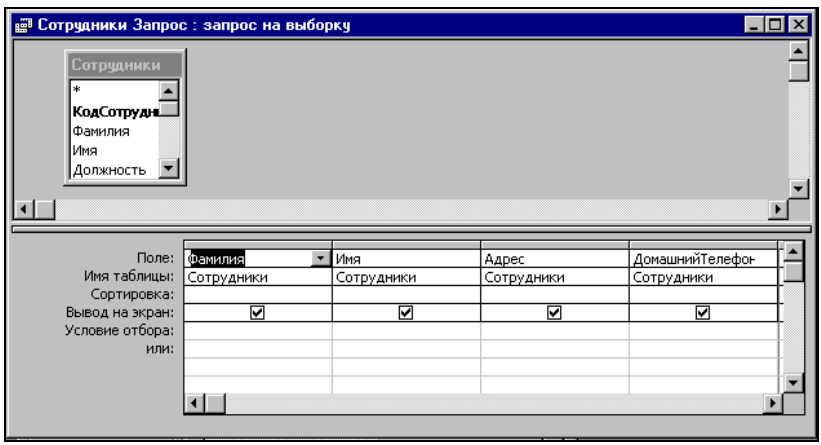


Рис. 4.5. Запрос Сотрудники Запрос в режиме Конструктора запросов

Появляется окно Конструктора запросов (рис. 4.5). В верхней части окна отображается таблица (или несколько таблиц, если запрос многотабличный) в том виде,

в каком таблицы отображаются в окне **Схема данных** (Relationship). Таблицы — источники данных для запроса, мы будем называть *базовыми* таблицами запроса. В нижней части окна находится *бланк запроса* — таблица, ячейки которой используются для определения запроса. В бланке отображаются все столбцы, включенные в результирующее множество запроса.

Для того чтобы просматривать полностью бланк запроса и все исходные таблицы, используют линейки прокрутки.

В области панелей инструментов Access отображается панель инструментов **Конструктор запросов** (Queries Design). Эта панель представлена на рис. 4.6, а в табл. 4.1 приведено описание кнопок этой панели инструментов и соответствующие им команды меню.



Рис. 4.6. Панель инструментов **Конструктор запросов**

Таблица 4.1. *Панель инструментов* ***Конструктор запросов***

Кнопка	Описание	Команда меню
Вид (View)	Отображение запроса в различных режимах. Чтобы изменить режим отображения запроса, нажмите стрелку справа от кнопки и в появившемся списке выделите нужный элемент. Если просто нажать эту кнопку, запрос будет отображен в режиме Таблицы	**Вид** | **Режим таблицы** (View | Datasheet View)
Сохранить (Save)	Сохранение активного запроса	**Файл** | **Сохранить** (File | Save)
Поиск файлов (File Search)	Поиск документов по содержанию	**Файл** | **Поиск файлов** (File | File Search)
Печать (Print)	Печать результатов запроса без открытия диалогового окна **Печать**	
Предварительный просмотр (Print Preview)	Предварительный просмотр запроса перед печатью	**Файл** | **Предварительный просмотр** (File | Print Preview)
Орфография (Spelling)	Проверка орфографии в текстовых полях запроса	**Сервис** | **Орфография** (Tools | Spelling)
Вырезать (Cut)	Удаление выделенных фрагментов из запроса в буфер обмена Windows	**Правка** | **Вырезать** (Edit | Cut)
Копировать (Copy)	Копирование выделенных фрагментов запроса в буфер обмена	**Правка** | **Копировать** (Edit | Copy)
Вставить (Paste)	Вставка содержимого буфера обмена в запрос	**Правка** | **Вставить** (Edit | Paste)

Таблица 4.1 *(окончание)*

Кнопка	Описание	Команда меню
Отменить (Undo)	Отмена изменений в запросе (позволяет отменить несколько последовательных действий, т. к. хранится стек изменений)	**Правка | Отменить** (Edit | Undo)
Вернуть (Redo)	Вернуть отмененные изменения (позволяет вернуть несколько выполненных последовательно отмен)	
Тип запроса (Query Type)	Изменение типа запроса. Для изменения типа запроса щелкните по стрелке справа от кнопки и выберите из списка нужный тип запроса	**Запрос | Выборка** (Query | Select Query)
Запуск (Run)	Выполнение запроса	**Запрос | Запуск** (Query | Run)
Отобразить таблицу (Show Table)	Выводится диалоговое окно **Добавление таблицы** (Show Table)	**Запрос | Добавить таблицу** (Query | Show Table)
Групповые операции (Totals)	Группировка записей в запросе и расчет итоговых значений	**Вид | Групповые операции** (View | Totals)
Набор значений (Top Values)	Отобразить только первые записи запроса. Количество отображаемых записей указывается в поле ввода либо в штуках, либо в процентах	
Свойства (Properties)	Открытие окон свойств выделенных объектов: запроса или поля запроса	**Вид | Свойства** (View | Properties)
Построитель (Build)	Вызов Построителя для создания выражения. Кнопка доступна только тогда, когда активизировано свойство запроса или поля запроса, которое допускает ввод выражения, например **Условие отбора** (Criteria)	
Окно базы данных (Database Window)	Отображение окна базы данных	**Окно | 1** (Window | 1)
Новый объект (New Object)	Создание нового объекта базы данных. Чтобы выбрать тип создаваемого объекта, нажмите стрелку справа от кнопки	
Справка: Microsoft Office Access (Microsoft Office Access Help)	Вызов помощника и получение справки по Access 2003	**Справка | Справка: Microsoft Office Access** (Help | Microsoft Office Access Help)

Вы освоите использование кнопок панели инструментов по мере знакомства с запросами как в данной главе, так и в *гл. 8*.

А теперь рассмотрим, как создать новый запрос с помощью Конструктора запросов. Для этого необходимо:

1. В окне базы данных на панели объектов выбрать ярлык **Запросы** (Queries).
2. В списке запросов выбрать ярлык **Создание запроса в режиме конструктора** (Create query in Design view) или нажать кнопку **Создать** (New), в появившемся окне **Новый запрос** (New Query) выбрать **Конструктор** (Design view) и нажать кнопку **ОК**.
3. В окне **Добавление таблицы** (Show Table) (рис. 4.7) выбрать одну или несколько таблиц или запросов для построения нового запроса и нажать кнопку **Добавить** (Add). Для удобства выбора таблиц и запросов в окне существуют следующие вкладки: **Таблицы** (Tables), на которой отображается список таблиц; **Запросы** (Queries), на которой отображается список запросов; **Таблицы и запросы** (Both), на которой отображается список таблиц и запросов вместе.

Совет

Если нужно добавить несколько таблиц, пользуйтесь клавишами <Ctrl> и <Shift>.

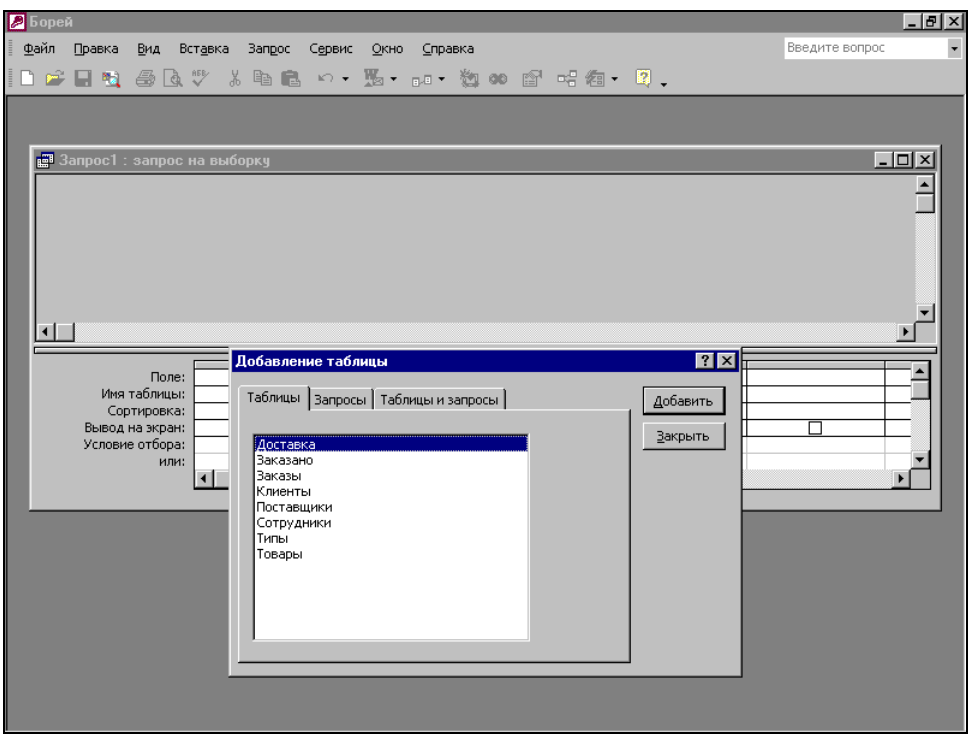


Рис. 4.7. Окно **Добавление таблицы** Конструктора запросов

4. После добавления всех необходимых таблиц нажать кнопку **Закрыть** (Close) в окне **Добавление таблицы** (Show Table).

Все выбранные таблицы оказываются помещенными на верхней панели окна Конструктора запросов. Если таблицы связаны между собой, т. е. связи присутствуют явно на схеме данных, то эти связи также отображаются (рис. 4.8). Если связи на схеме данных не установлены, то Конструктор запросов автоматически устанавливает связи между таблицами, если они содержат поля, которые имеют одинаковые имена и согласованные типы *(см. разд. "Связывание таблиц на схеме данных" гл. 2)*.

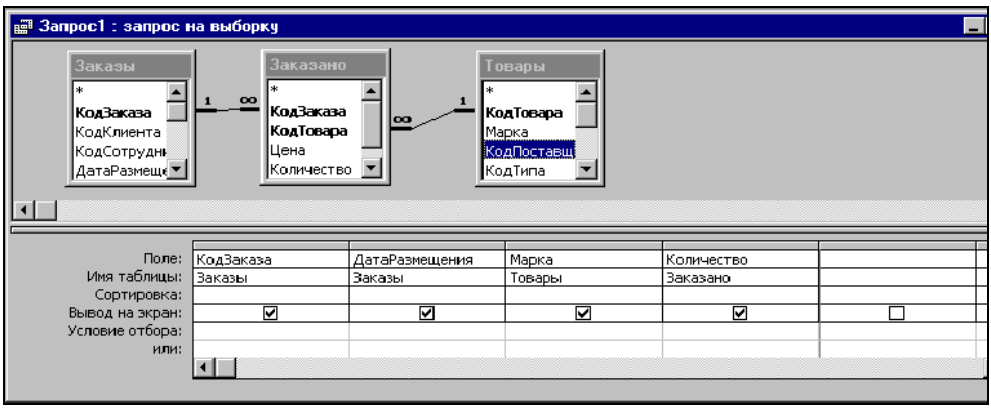


Рис. 4.8. Запрос по нескольким связанным таблицам

Внимание!

Иногда Конструктор запросов устанавливает лишние связи, основываясь только на именах и типах полей. Это может привести к некорректным результатам запроса, поэтому нужно обязательно проверять, как отображаются связи между таблицами в окне Конструктора запросов, и удалить вручную лишние связи. Для этого выделите лишнюю связь, щелкнув по ней левой кнопкой мыши, и нажмите клавишу <Delete>. Работа со связями в окне Конструктора запросов выполняется точно так же, как в окне **Схема данных** (Relationship) *(см. разд. "Связывание таблиц на схеме данных" гл. 2)*.

Замечание

Добавить таблицу или еще один запрос в окно Конструктора запросов можно также путем перетаскивания мышью названия таблицы или запроса из окна базы данных.

5. Затем нужно указать, какие поля из базовых таблиц будут отображаться в запросе. Включать в запрос можно поля из любой таблицы. Способов включения полей в запрос существует несколько:

   - Выделите нужное поле в таблице-источнике (можно выделить несколько полей, пользуясь клавишами <Shift> и <Ctrl>). Если требуется включить в запрос все поля базовой таблицы, выделите поле, обозначенное звездочкой (*). Дважды щелкните левой кнопкой мыши на выделенном поле. При этом в бланке запроса появятся столбцы, соответствующие выбранным полям. Затем

аналогично добавьте другие поля. Столбцы в бланке запроса при этом заполняются слева направо.

- Можно подвести указатель мыши к выделенному полю (одному из выделенных полей), нажать левую кнопку мыши и перетащить поле (поля) в нужное место бланка запроса (указатель мыши при этом должен принять вид трех прямоугольников). Последний способ позволяет помещать поля в любое место бланка запроса.
- И наконец, вместо перетаскивания полей в бланк запроса из таблицы можно просто использовать раскрывающийся список полей в строке **Поле** (Field) бланка запроса.

Примеры разных способов включения полей в результат запроса в режиме Конструктора запросов приведены на рис. 4.9—4.11.

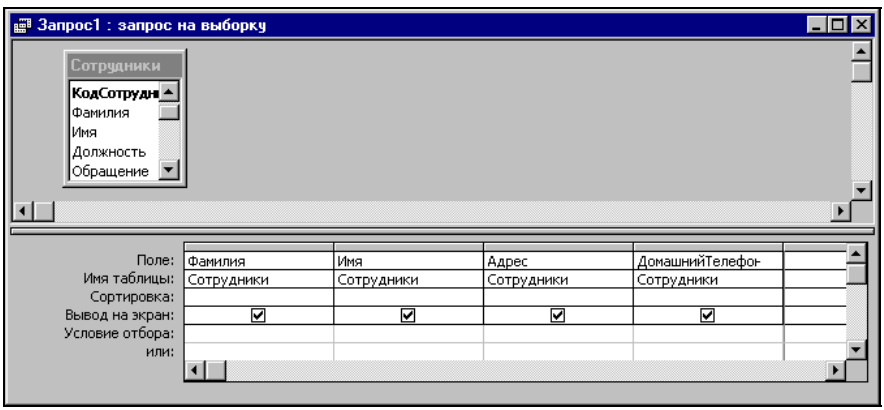


Рис. 4.9. Пример отбора полей для нового запроса в режиме Конструктора

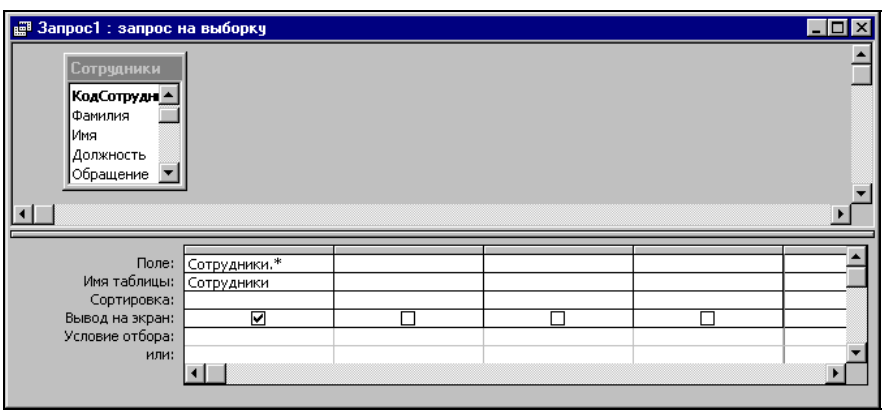


Рис. 4.10. Вид окна Конструктора при выборе всех полей таблицы

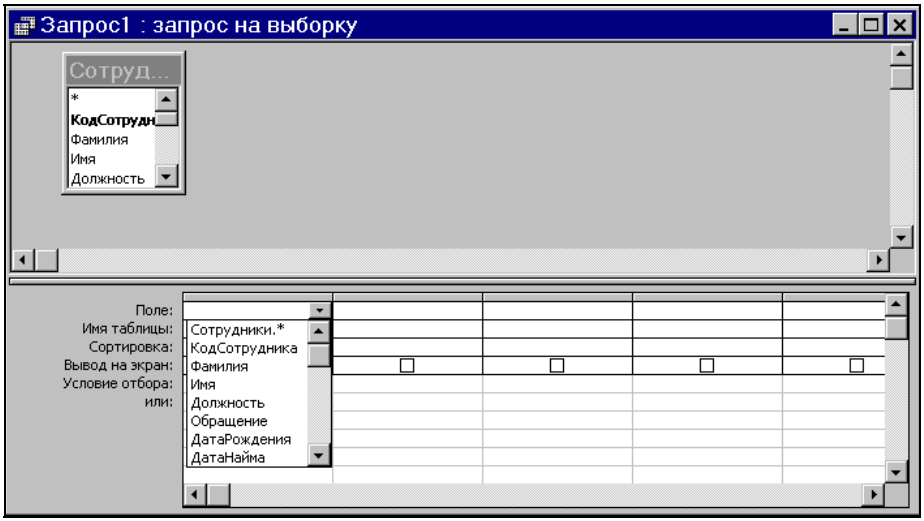


Рис. 4.11. Выбор полей запроса из раскрывающегося списка

Чтобы добавить в запрос еще одну таблицу или другой запрос, необходимо:

1. Нажать кнопку **Добавить таблицу** (Show Table) на панели инструментов или выполнить команду меню **Запрос | Добавить таблицу** (Query | Show Table).
2. В окне **Добавление таблицы** (Show Table) (см. рис. 4.7) выбрать вкладку, содержащую требуемые объекты.

Замечание

Если нужная таблица находится в другой базе данных или другом приложении, необходимо сначала присоединить эту таблицу к текущей базе данных.

3. Выбрать имя объекта, добавляемого в запрос.
4. Нажать кнопку **Добавить** (Add), а затем кнопку **Закрыть** (Close).

Чтобы удалить базовую таблицу из запроса, необходимо выделить ее, щелкнув на любом месте в списке ее полей, и нажать клавишу <Delete>.

Чтобы удалить поле из запроса, выделите нужный столбец в бланке запроса, а затем нажмите клавишу <Delete>. Чтобы выделить столбец, пользуйтесь областью выделения столбцов — узкой серой полоской над столбцами в бланке запроса. Когда вы подводите указатель мыши к этой области, он преобразуется в жирную стрелку, направленную вниз.

Поля в таблице, являющейся результатом запроса, отображаются в том порядке, в котором они следуют в бланке запроса. Если требуется изменить порядок их следования, переставьте соответствующим образом столбцы в бланке запроса. Делается это стандартным способом, т. е. сначала выделяется нужный столбец или несколько столбцов, а затем они перетаскиваются мышью на новое место. При этом отпускать кнопку мыши нужно тогда, когда указатель мыши окажется перед тем столбцом, который должен быть справа от вставляемого столбца. После вставки все столбцы,

расположенные справа от указателя, сдвигаются вправо. Если вы перемещаете столбцы в конец запроса, то отпускайте кнопку мыши, когда указатель окажется перед первым свободным столбцом.

В режиме Конструктора запросов можно изменять имена полей запроса. Чтобы переименовать поле, необходимо установить курсор в бланке запроса перед первой буквой его имени и ввести новое имя и символ двоеточия. Пример запроса с измененными именами полей приведен на рис. 4.12. Изменение имени поля в бланке запроса приводит к изменению заголовка столбца при просмотре запроса в режиме Таблицы. Кроме того, если на основе запроса создать новый объект, например форму или отчет, в новом объекте будет использовано новое имя поля. Имя поля базовой таблицы при этом не изменяется. Это имя также не изменяется в тех формах и отчетах, которые были созданы на основе запроса до изменения имени поля. Имена полей в запросах должны соответствовать соглашениям об именах Microsoft Access.

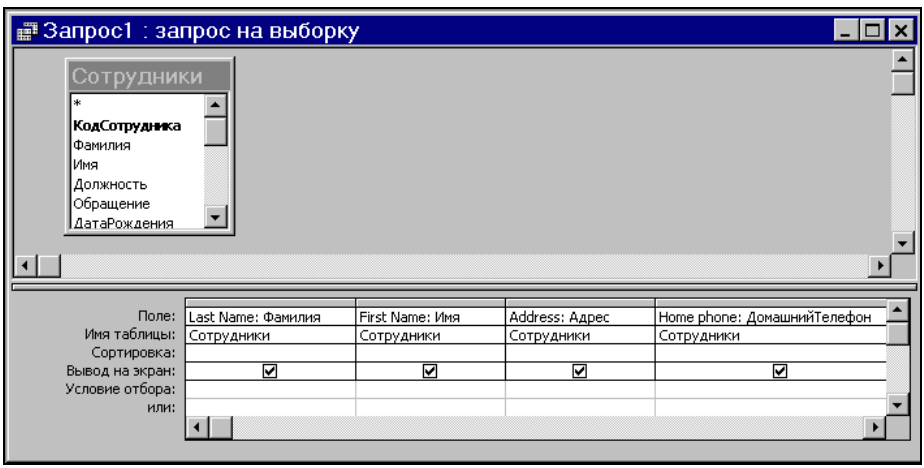


Рис. 4.12. Изменение названия поля в запросе

Совет

Чтобы использовать новое имя поля только в заголовках столбцов в режиме Таблицы или как подпись полей в формах и отчетах, следует задать для этого поля свойство **Подпись** (Caption), а не переименовывать поле в бланке. *(Об установке свойств поля см. разд. "Настройка свойств полей в запросе и свойств самого запроса" гл. 8.)*

Ширина столбцов в бланке запроса также может меняться. Осуществляется это простым перетаскиванием мышью границы столбца или двойным щелчком мыши по линии границы, как это делается в таблице Excel.

Замечание

Изменение ширины столбцов в окне Конструктора запросов или в окне расширенного фильтра не влияет на ширину столбцов запроса в режиме Таблицы или объекта, для которого применен фильтр.

В строке **Условие отбора** (Criteria) и в строке **или** (Or) указываются условия отбора записей. Такими условиями могут быть логические выражения. Например, (>30), (='Иванов'), (=10) и т. п. *(О создании выражений в условиях запроса см. разд. "Использование выражений в запросах" далее в этой главе.)*

Условия, находящиеся в одной строке, но в разных столбцах бланка, объединяются по И (And). Если нужно объединить условия отбора по ИЛИ (Or), разместите эти условия в разных строках бланка запроса. Пример использования нескольких условий отбора приведен на рис. 4.13.

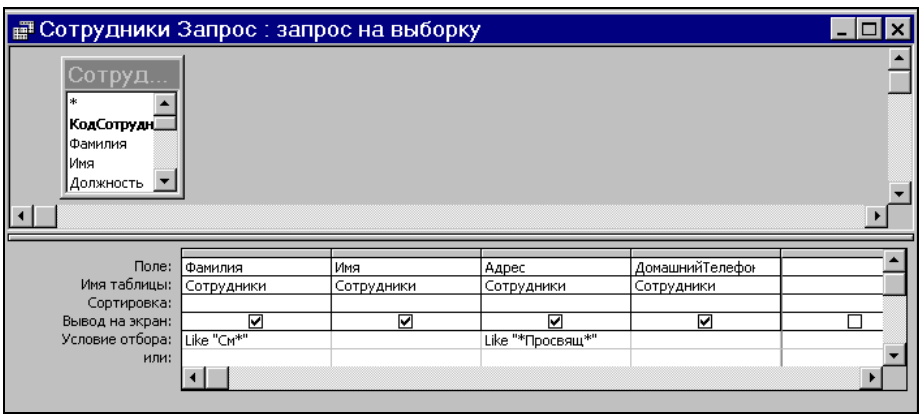


Рис. 4.13. Задание критериев отбора записей в запросе

Если критерий отбора очень сложный, можно вставлять дополнительные строки условий. Для этого необходимо:

1. Выделить строку, которая должна оказаться под вставляемой строкой. Для чего нужно подвести указатель мыши к ее левой границе (он должен превратиться в жирную стрелку, указывающую вправо) и щелкнуть левой кнопкой мыши.
2. Выполнить команду **Вставка | Строки** (Insert | Rows) или нажать клавишу <Ins>. Новая пустая строка вставляется над выбранной. В нее можно вносить условия отбора.

Для удаления строки условий отбора:

1. Щелкните левой кнопкой мыши в любом месте строки.
2. Выполните команду **Правка | Удалить строки** (Edit | Delete Rows).

Чтобы установить порядок сортировки записей в запросе, используйте строку **Сортировка** (Sort). Для каждого поля, по которому должны сортироваться записи, выберите из списка соответствующий порядок сортировки: **по возрастанию** (Ascending) или **по убыванию** (Descending). По умолчанию во всех полях запроса устанавливается значение **(отсутствует)** (Not sorted). Пример использования поля со списком **Сортировка** (Sort) приведен на рис. 4.14.

Если нужно отсортировать записи в запросе по нескольким полям, разместите их в бланке запроса таким образом, чтобы они были упорядочены слева направо — именно в таком порядке будет выполняться сортировка.

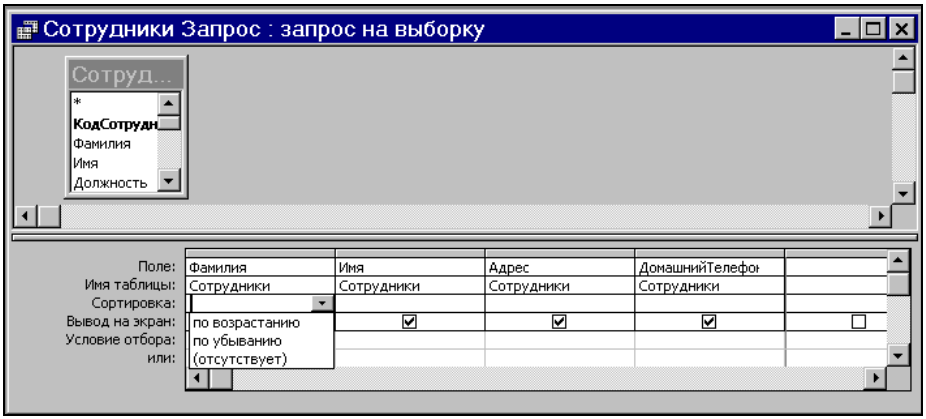


Рис. 4.14. Установка порядка сортировки записей в запросе

Можно легко удалить поле из результирующей таблицы запроса, если просто сбросить флажок **Вывод на экран** (Show) в этом столбце бланка запроса. По умолчанию данный флажок установлен для всех полей запроса. Ситуация, когда поле не должно отображаться в результате запроса, возникает обычно, когда оно включается в бланк запроса только для того, чтобы указать условие отбора или сортировки записей. Пример такого запроса приведен на рис. 4.15. В запрос включены все поля таблицы `Товары` (Products) (это задано в первом столбце бланка запроса), а поля `КодТипа` и `Марка` включены только для того, чтобы указать порядок сортировки. Поэтому флажок **Вывод на экран** (Show) у этих полей сброшен. В противном случае эти поля вывелись бы дважды.

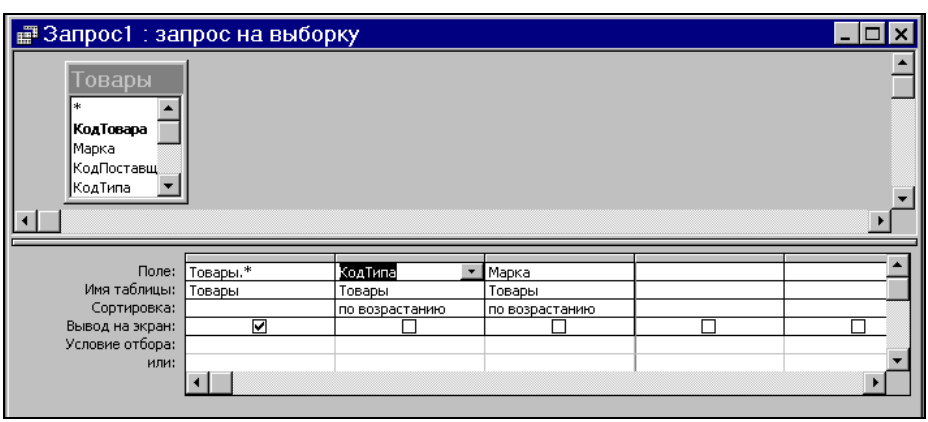


Рис. 4.15. Запрос в режиме Конструктора

Если вы хотите очистить бланк запроса для того чтобы создать новый, нужно выполнить команду **Правка | Очистить бланк** (Edit | Clear Grid).

После формирования бланка запроса его можно сохранить, нажав кнопку **Сохранить** (Save) на панели инструментов или выполнив команду меню **Файл | Сохранить** (File |

Save). При этом появляется диалоговое окно, в котором нужно ввести имя сохраняемого запроса (рис. 4.16). Просто закрыв запрос, вы тоже его сохраните.

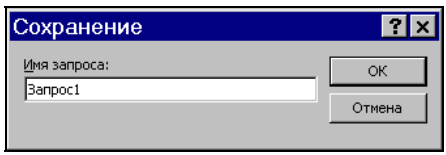


Рис. 4.16. Диалоговое окно **Сохранение**

Результаты выполнения запроса можно увидеть, выбрав **Режим таблицы** (Datasheet View) с помощью кнопки **Вид** (View) или нажав кнопку **Запуск** (Run) на панели инструментов.

Результат выполнения запроса, созданного в режиме Конструктора, приведен на рис. 4.17.

Запрос1 : запрос на выборку

Код товара	Марка	Поставщик	Тип	Единица измерения
64	Chai	ООО Экзотика	Напитки	10 коробок по 20 шт.
65	Chang	ООО Экзотика	Напитки	24 бутылки по 355 мл
40	Chartreuse verte	Aux joyeux ecclesiastiques	Напитки	750 мл бутылка
39	Cote de Blaye	Aux joyeux ecclesiastiques	Напитки	12 бутылок по 75 мл
10	Guarana Fantastica	Refrescos Americanas LTDA	Напитки	12 банок по 355 мл
44	Ipoh Coffee	Leka Trading	Напитки	16 банок по 500 г
30	Lakkalikoori	Karkki Oy	Напитки	500 мл
21	Laughing Lumberjack Lager	Bigfoot Breweries	Напитки	24 бутылки по 355 мл
24	Outback Lager	Pavlova, Ltd.	Напитки	24 бутылки по 355 мл
29	Rhonbrau Klosterbier	Plutzer Lebensmittelgrossmar	Напитки	24 бутылки по 0,5 л
35	Sasquatch Ale	Bigfoot Breweries	Напитки	24 бутылки по 355 мл
36	Steeleye Stout	Bigfoot Breweries	Напитки	24 бутылки по 355 мл
66	Aniseed Syrup	ООО Экзотика	Приправы	12 бутылок по 550 мл
67	Chef Anton's Cajun Seasonii	New Orleans Cajun Delights	Приправы	48 банок по 177 мл
68	Chef Anton's Gumbo Mix	New Orleans Cajun Delights	Приправы	36 коробок
1	Genen Shouyu	Mayumi's	Приправы	24 бутылки по 250 мл
69	Grandma's Boysenberry Spr	Karkki Oy	Приправы	12 банок по 237 мл
45	Gula Malacca	Leka Trading	Приправы	20 пакетов по 2 кг
19	Louisiana Fiery Hot Pepper S	New Orleans Cajun Delights	Приправы	32 бутылки по 237 мл
20	Louisiana Hot Spiced Okra	New Orleans Cajun Delights	Приправы	24 банки по 237 мл
71	Northwoods Cranberry Sauc	Grandma Kelly's Homestead	Приправы	12 банок по 355 мл

Запись: 1 из 77

Рис. 4.17. Результат выполнения запроса, созданного в режиме Конструктора

Создание вычисляемых полей в запросах

Создание запросов интересно не только тем, что вы можете в виде одной таблицы представить данные из нескольких связанных таблиц и отобрать нужные записи из этих таблиц. Вы можете создавать столбцы в запросе, которые являются результатом вычислений над значениями других столбцов. Такие столбцы называются *вычисляемыми*. Это существенно расширяет возможности запросов. Простейшим примером вычисляемого поля в запросе может быть поле, которое объединяет имя и фамилию

человека. На рис. 4.18 показан пример такого поля в запросе, созданном на базе таблиц Сотрудники (Employees) и Заказы (Orders).

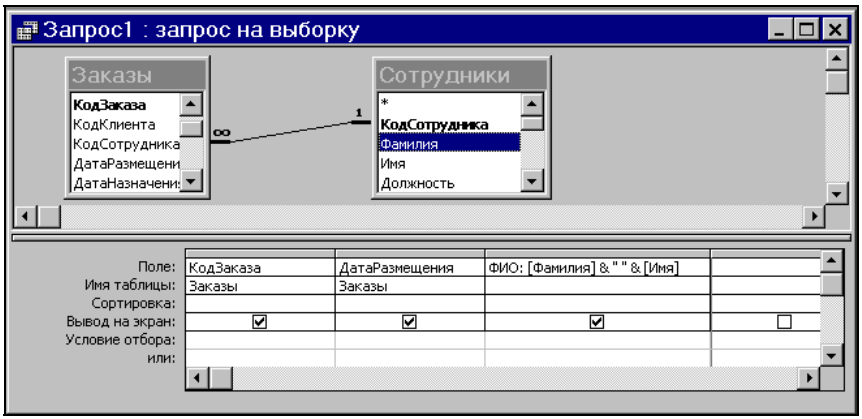


Рис. 4.18. Вычисляемое поле в запросе

Чтобы создать вычисляемое поле, нужно ввести выражение, которое вычисляет требуемое значение, в строку **Поле** (Field) свободного столбца бланка запроса. В данном примере это выражение представляет собой конкатенацию полей, содержащих имя и фамилию сотрудника, с пробелом между ними. Здесь используются ссылки на поля таблицы, которые в выражении заключаются в квадратные скобки. Перед выражением нужно написать имя поля ФИО и поставить двоеточие.

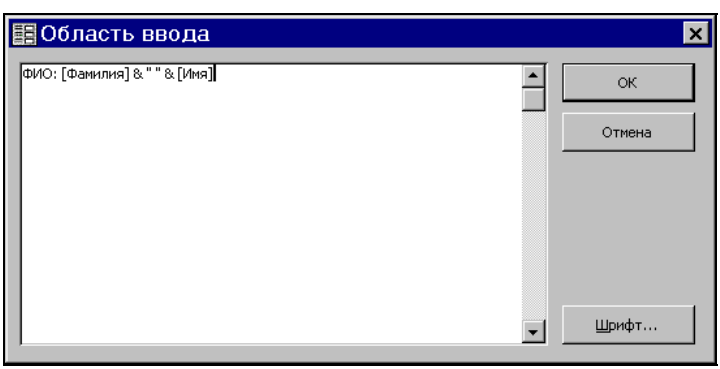


Рис. 4.19. Диалоговое окно **Область ввода**

Совет

Если выражение длинное, его неудобно писать в строке **Поле** (Field). Нажмите комбинацию клавиш <Shift>+<F2>. Появится диалоговое окно **Область ввода** (Zoom) (рис. 4.19), в котором вводить выражение удобнее. Можно также использовать Построитель выражений, вызвав его щелчком по кнопке **Построить** (Build) на панели

инструментов. *(Подробную информацию об использовании Построителя выражений и о правилах составления выражений в Access см. в последнем разделе данной главы.)*

Вы можете создать любое количество вычисляемых столбцов в запросе, используя при этом сколь угодно сложные выражения. С другими примерами создания таких столбцов вы познакомитесь в следующих главах книги.

Виды соединений

В зависимости от решаемых задач соединение между таблицами в запросах может быть нескольких видов.

Наиболее распространенным является *внутреннее соединение* (эквисоединение). Если таблицы связаны отношением "один-ко-многим", соединения основываются на совпадении уникального значении поля первичного ключа в одной таблице и значений поля внешнего ключа в другой таблице. В результирующее множество запроса попадают все записи из главной таблицы (таблицы на стороне "один"), для которых имеются соответствующие записи в подчиненной таблице (таблице на стороне "многие"). Если в подчиненной таблице записи с заданной величиной отсутствуют, то соответствующие записи в главной таблице в результирующее множество не включаются. С другой стороны, если одной записи в главной таблице соответствует несколько записей в подчиненной таблице, то в запросе запись из главной таблицы повторится столько раз, сколько ей соответствует подчиненных записей.

Подобного рода соединения между таблицами Access создает автоматически, если:

- в таблицах имеются поля с одинаковыми именами и согласованными типами, причем одно из полей является ключевым;
- соединение было явно задано в окне **Схема данных** (Relationships).

Замечание

"Согласованные типы" в большинстве случаев означают одинаковые типы. Исключением является поле типа `Счетчик`, которое может иметь размер `Длинное целое` или `Код репликации` и может связываться с числовыми полями соответствующего размера.

Для создания запроса, объединяющего все записи из одной таблицы и только те записи из второй, в которых связанные поля совпадают, используют *внешнее соединение*. В этом случае независимо от того, имеются ли соответствующие записи во второй таблице, все записи первой попадают в результирующее множество запроса.

Если необходимо связать данные любым отношением, кроме отношения равенства, используют *соединение по отношению* или *тета-соединение*). Соединение по отношению не отображается в окне **Схема данных** (Relationships) и не выводится в окне Конструктора запросов.

Для обозначения внутренних соединений в Access используются зарезервированные слова ANSI SQL `INNER JOIN`, а для указания внешних соединений — слова `LEFT JOIN` или `RIGHT JOIN`. Выражение `WHERE <поле1>=<поле2>` соответствует внутреннему соединению, а `WHERE поле1>поле2` — соединению по отношению (одному из таких соединений).

Для связывания записей из одной таблицы применяют *рекурсивное соединение*. Оно создается путем добавления в запрос копии таблицы (в результате чего Access назначает псевдоним для копии) и связывания полей идентичных таблиц.

Внутреннее соединение

Внутреннее соединение двух таблиц по одному полю в реляционной базе данных строится на основе отношения "один-ко-многим". Примером может служить демонстрационная база данных Борей (Northwind), в которой все соединения являются внутренними соединениями по одному полю на основе указанного отношения. В ходе разработки баз данных, в которых предполагается использование запросов на основе внутренних соединений, придерживайтесь перечисленных далее правил.

- Каждая таблица "один" должна иметь первичный ключ с уникальными значениями. Отсутствие повторений значений поля или полей первичного ключа в таблице Access устанавливает автоматически.
- Отношение "многие-ко-многим" реализуйте на основе промежуточной таблицы, которая связана с каждой из двух таблиц отношением "один-ко-многим". Для обеих связей промежуточная таблица будет находиться со стороны "многие".
- Необходимо извлечь повторяющиеся данные в новую таблицу и связать ее с таблицей, из которой эти данные были получены, отношением "многие-к-одному". Основная цель — однозначно определить извлеченные данные. Часто для этого приходится использовать первичный ключ, состоящий из нескольких полей. Для автоматизации работы по нахождению и извлечению повторяющейся информации можно использовать Мастер по анализу таблиц Microsoft Access. Этот Мастер запускается при выборе команды меню **Сервис | Анализ | Таблица** (Tools | Analysis | Table).

В качестве примера использования запроса на основе внутреннего соединения по одному полю в базе данных Борей (Northwind) построим отчет с указанием марки товара, его поставщика, единицы измерения товара и его цены.

1. Если база данных Борей открыта, закройте все окна, кроме окна **База данных** (Database). Если база данных Борей не загружена, загрузите ее.
2. Щелкните по ярлыку **Запросы** (Queries) на панели объектов окна **База данных** (Database), чтобы открыть список запросов, а затем нажмите кнопку **Создать** (New) для создания нового запроса. В появившемся диалоговом окне **Новый запрос** (New Query) выделите элемент **Конструктор** (Design View) и нажмите кнопку **ОК**. Access отображает диалоговое окно **Добавление таблицы** (Show Table) поверх пустого окна запроса.
3. Выберите таблицы Поставщики (Suppliers) и Товары (Products) и закройте диалоговое окно **Добавление таблицы** (Show Table). Access покажет список полей обеих таблиц в верхней панели окна запроса (рис. 4.20). Соединение создается автоматически, т. к. поле КодПоставщика является ключевым в таблице Поставщики и в таблице Товары найдено поле с тем же именем (внешний ключ). Отображаемое на рис. 4.20 отношение является отношением "один-ко-многим".
4. Выберите поле Марка (ProductName) в таблице Товары (Products) и перетащите его в строку **Поле** (Field) первого столбца бланка запроса.

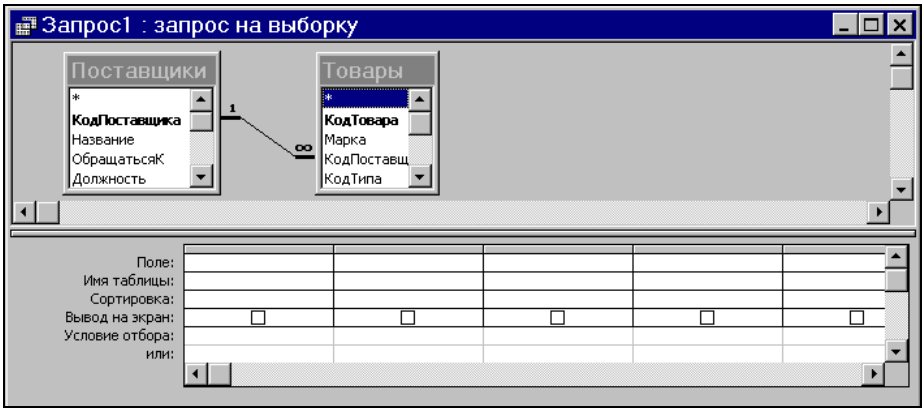


Рис. 4.20. Окно Конструктора запроса с изображением соединения полей таблиц

5. Выберите поле `Название` (CompanyName) в таблице `Поставщики` (Suppliers) и перетащите в строку **Поле** (Field) второго столбца. Также перетащите поля `ЕдиницаИзмерения` (QuantityPerUnit) и `Цена` (UnitPrice) таблицы `Товары` (Products) в строку **Поле** (Field) третьего и четвертого столбца бланка запроса, соответственно (рис. 4.21).

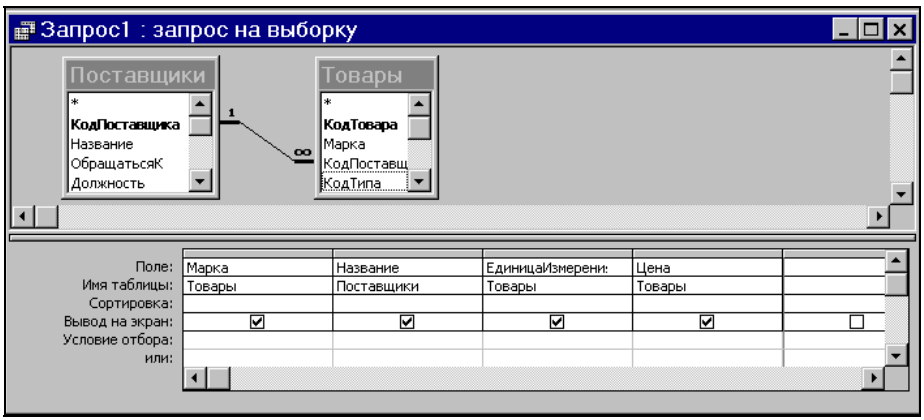
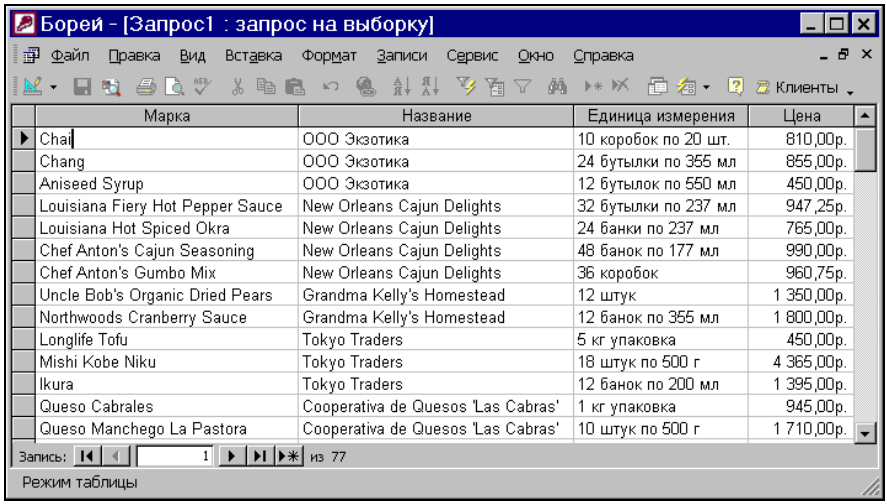


Рис. 4.21. Бланк запроса с заполненными полями в режиме Конструктора

6. Нажмите кнопку **Запуск** (Run) либо выберите **Режим таблицы** (Query View) в списке кнопки **Вид** (View) на панели инструментов для отображения результата запроса (рис. 4.22). Обратите внимание на то, что заголовками полей в созданном наборе записей являются подписи полей таблицы (включающие пробелы), а не их имена, в которых пробелы обычно не используются.

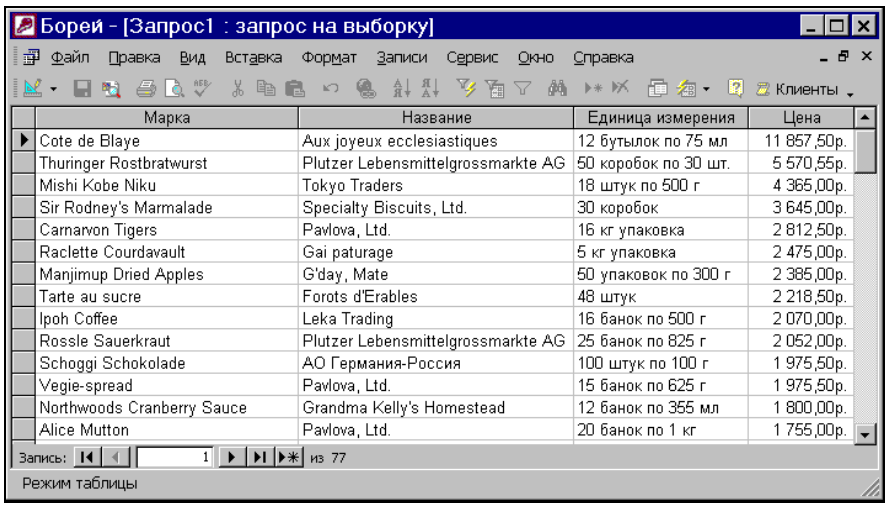
После выполнения запроса Access выводит на экран результирующее множество, записи которого упорядочены по значению первичного ключа (так же, как и при просмотре таблицы). Чтобы изменить порядок сортировки, например упорядочить записи по убывающей цене товара, выполните следующее:



Марка	Название	Единица измерения	Цена
Chai	ООО Экзотика	10 коробок по 20 шт.	810,00р.
Chang	ООО Экзотика	24 бутылки по 355 мл	855,00р.
Aniseed Syrup	ООО Экзотика	12 бутылок по 550 мл	450,00р.
Louisiana Fiery Hot Pepper Sauce	New Orleans Cajun Delights	32 бутылки по 237 мл	947,25р.
Louisiana Hot Spiced Okra	New Orleans Cajun Delights	24 банки по 237 мл	765,00р.
Chef Anton's Cajun Seasoning	New Orleans Cajun Delights	48 банок по 177 мл	990,00р.
Chef Anton's Gumbo Mix	New Orleans Cajun Delights	36 коробок	960,75р.
Uncle Bob's Organic Dried Pears	Grandma Kelly's Homestead	12 штук	1 350,00р.
Northwoods Cranberry Sauce	Grandma Kelly's Homestead	12 банок по 355 мл	1 800,00р.
Longlife Tofu	Tokyo Traders	5 кг упаковка	450,00р.
Mishi Kobe Niku	Tokyo Traders	18 штук по 500 г	4 365,00р.
Ikura	Tokyo Traders	12 банок по 200 мл	1 395,00р.
Queso Cabrales	Cooperativa de Quesos 'Las Cabras'	1 кг упаковка	945,00р.
Queso Manchego La Pastora	Cooperativa de Quesos 'Las Cabras'	10 штук по 500 г	1 710,00р.

Рис. 4.22. Результирующее множество запроса

1. Переключитесь в режим Конструктора запроса.
2. Установите курсор в поле **Сортировка** (Sort) столбца `Цена`, а затем нажмите клавишу <F4> или стрелку справа для вывода содержимого списка.
3. Выберите элемент **по убыванию** (Descending).
4. Нажмите кнопку **Запуск** (Run) для вывода нового результирующего множества запроса (рис. 4.23).
5. Сохраните запрос, дав ему название: `Отсортированный список товаров`.



Марка	Название	Единица измерения	Цена
Cote de Blaye	Aux joyeux ecclesiastiques	12 бутылок по 75 мл	11 857,50р.
Thuringer Rostbratwurst	Plutzer Lebensmittelgrossmarkte AG	50 коробок по 30 шт.	5 570,55р.
Mishi Kobe Niku	Tokyo Traders	18 штук по 500 г	4 365,00р.
Sir Rodney's Marmalade	Specialty Biscuits, Ltd.	30 коробок	3 645,00р.
Carnarvon Tigers	Pavlova, Ltd.	16 кг упаковка	2 812,50р.
Raclette Courdavault	Gai paturage	5 кг упаковка	2 475,00р.
Manjimup Dried Apples	G'day, Mate	50 упаковок по 300 г	2 385,00р.
Tarte au sucre	Forots d'Erables	48 штук	2 218,50р.
Ipoh Coffee	Leka Trading	16 банок по 500 г	2 070,00р.
Rossle Sauerkraut	Plutzer Lebensmittelgrossmarkte AG	25 банок по 825 г	2 052,00р.
Schoggi Schokolade	АО Германия-Россия	100 штук по 100 г	1 975,50р.
Vegie-spread	Pavlova, Ltd.	15 банок по 625 г	1 975,50р.
Northwoods Cranberry Sauce	Grandma Kelly's Homestead	12 банок по 355 мл	1 800,00р.
Alice Mutton	Pavlova, Ltd.	20 банок по 1 кг	1 755,00р.

Рис. 4.23. Результат запроса с сортировкой по убывающей цене товара

Аналогично создаются запросы, в которых участвует несколько связанных таблиц, образующих цепочку. При этом в результат запроса могут быть включены поля из всех таблиц, участвующих в запросе, или только поля из таблиц, находящихся на концах такой цепочки (рис. 4.24). Во втором случае таблицы на концах цепочки оказываются связанными косвенно, и чтобы правильно построить запрос, соединяющий поля этих таблиц, необходимо включить в запрос каждую таблицу, участвующую в соединении. Например, можно вывести страны, товары из которых покупаются клиентами. В данном случае косвенно оказываются связанными таблицы `Клиенты` (Customers) и `Поставщики` (Suppliers), а промежуточными таблицами оказываются таблицы `Заказы` (Orders), `Заказано` (Order Details), `Товары` (Products). Access автоматически показывает связи, в том числе и промежуточные, между таблицами.

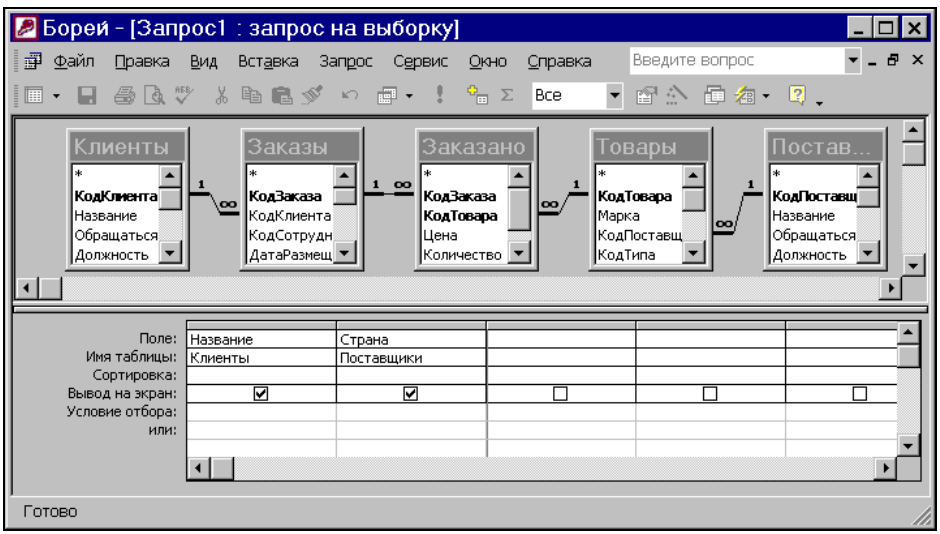


Рис. 4.24. Соединение косвенно связанных записей

После добавления в бланк запроса полей `Название` (CompanyName) из таблицы `Клиенты` (Customers) и `Страна` (Country) из таблицы `Поставщики` (Suppliers) выберите команду **Вид | Режим SQL** (View | SQL View) для того, чтобы просмотреть инструкцию SQL, соответствующую данному запросу (рис. 4.25). Соединения таблиц задаются операцией `INNER JOIN...ON...` При таком положении косвенные соединения основываются на выражении `INNER JOIN...ON...ON...` *(Подробную информацию об инструкциях языка SQL см. в разд. "Создание запросов SQL", гл. 8.)*

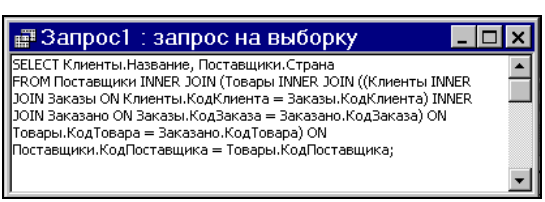


Рис. 4.25. Эквивалентная запросу инструкция SQL

Запросы, соединяющие косвенно связанные записи, часто используются при анализе данных с помощью статистических функций SQL или перекрестных запросов Access.

Внешнее соединение

Внешние соединения бывают *левыми* или *правыми*. Запрос, в котором участвуют таблицы с левым внешним соединением (`LEFT JOIN` или `*=` в SQL), выводит все записи таблицы "один", в независимости от того, имеются ли соответствующие им записи в таблице "многие". И наоборот, запрос, в котором участвуют таблицы с правым внешним соединением (`RIGHT JOIN` или `=*` в SQL), выводит все записи таблицы "многие", в независимости от того, имеются ли соответствующие им записи в таблице "один".

Рассмотрим в качестве примера, как с помощью левого внешнего соединения создать запрос в базе данных `Борей` (Northwind), который обнаруживает клиентов, не сделавших ни одного заказа:

1. Создайте новый запрос и добавьте в него таблицы `Клиенты` (Customers) и `Заказы` (Orders). Поскольку эти таблицы связаны отношением "один-ко-многим", между ними сразу появляется соединяющая линия.
2. Выделите и перетащите в бланк запроса поле `Название` (CompanyName) таблицы `Клиенты` и поле `КодЗаказа` (OrderId) таблицы `Заказы`.
3. Выполните запрос и посмотрите, сколько записей будет возвращать запрос: 830.
4. Вернитесь в режим Конструктора. Щелкните правой кнопкой мыши на линии, соединяющей таблицы, и выберите из контекстного меню команду **Параметры объединения** (Join Properties). Появится диалоговое окно, представленное на рис. 4.26. В данном окне показаны имена связанных полей в таблицах. При этом таблица на стороне "один" считается левой (независимо от того, как она расположена в окне Конструктора), а таблица на стороне "многие" — правой. Ниже расположены три переключателя, которые задают:
   - **1** — внутреннее соединение;
   - **2** — левое внешнее соединение;
   - **3** — правое внешнее соединение.

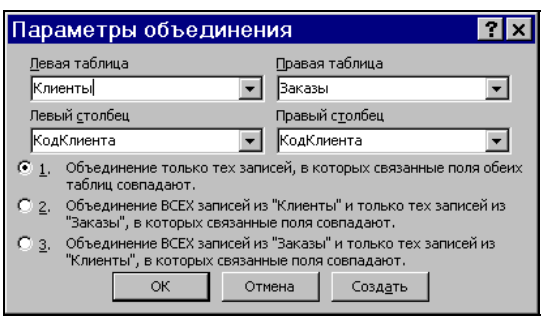


Рис. 4.26. Диалоговое окно **Параметры объединения**

Замечание

Традиционно термин JOIN, который применяется к операции соединения таблиц, переводился как "объединение", хотя на самом деле операция объединения таблиц — это UNION, которая позволяет объединить все записи из одной таблицы и все записи из другой, а потом удалить дублирующиеся записи. Тем не менее, при локализации диалоговых окон Access по-прежнему, как мы видим, используется термин "объединение" по отношению к операции JOIN.

5. Задайте левое внешнее соединение, выбрав значение **2**. Нажмите кнопку **OK** для закрытия диалогового окна. При этом на конце линии соединения появится стрелка в сторону таблицы "многие", что указывает на левое внешнее соединение (рис. 4.27).

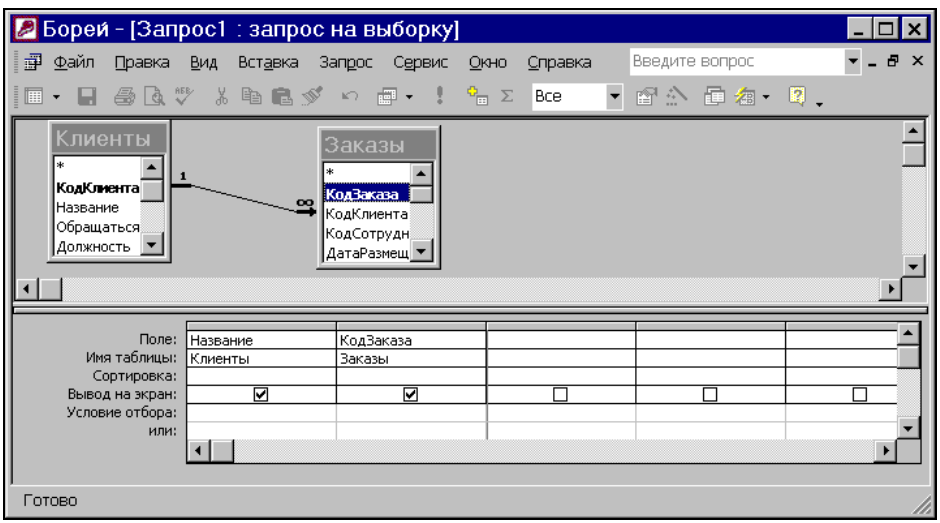


Рис. 4.27. Отображение левого внешнего соединения в окне Конструктора запросов

6. Установите сортировку по возрастанию для поля КодЗаказа (OrderId) во втором столбце бланка запроса.
7. Нажмите на панели инструментов кнопку **Запуск** (Run) для вывода результирующего множества запроса с левым внешним соединением (рис. 4.28). Количество записей стало равно 832, и первые две записи содержат пустое поле КодЗаказа.
8. Сохраните запрос под именем Холодные клиенты. Он нам потребуется в дальнейшем.

Запросы с правым внешним соединением используются гораздо реже, т. к., если при составлении схемы данных для связи между таблицами установлена ссылочная целостность *(см. разд. "Обеспечение целостности данных" гл. 2)*, записей в таблице "многие" (подчиненной), не связанных с записями в таблице "один" (главной), просто не может быть.

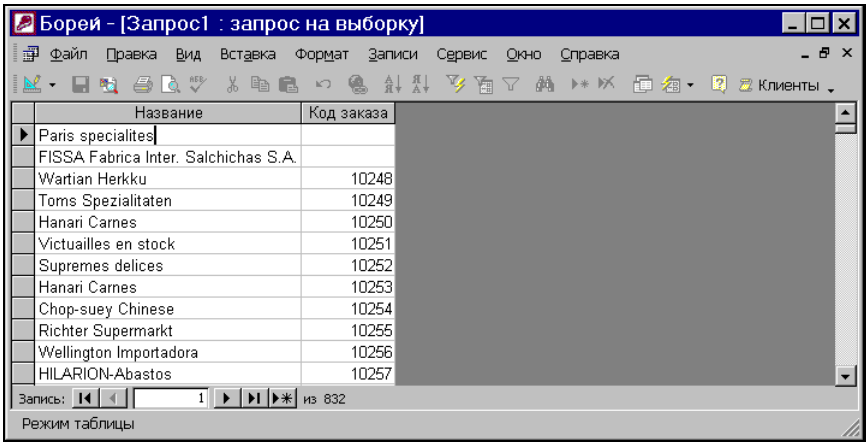


Рис. 4.28. Результат выполнения запроса с левым внешним соединением

Рекурсивный запрос

Рекурсивный запрос используется, когда в таблице есть записи, которые ссылаются на другие записи этой же таблицы. Примером такой таблицы в базе данных `Борей` является таблица `Сотрудники` (Employees). В этой таблице есть поле `Подчиняется` (ReportsTo). Значением этого поля является код сотрудника, которому подчиняется данный сотрудник. Поэтому, чтобы увидеть иерархические связи между сотрудниками, построим запрос.

1. Создайте новый запрос и добавьте в него таблицу `Сотрудники`.
2. Добавьте в запрос копию таблицы, повторно нажав кнопку **Добавить** (Show Table). Access присваивает копии имя `Сотрудники_1`. Закройте диалоговое окно **Добавление таблицы** (Show Table).
3. Создайте связь между таблицей и ее копией, как показано на рис. 4.29.
4. Чтобы в запросе отображались все сотрудники, в том числе и те, которые никому не подчиняются, нужно сделать это соединение внешним. Поэтому выделите линию соединения и откройте окно **Параметры объединения** (Join Properties) *(см. разд. "Внешнее соединение" ранее)*. Выберите переключатель **2**.
5. Включите поля `Должность` (Title), `Фамилия` (LastName) и `Имя` (First Name) исходной таблицы `Сотрудники` в бланк заказа. И затем перетащите поле `Должность` из таблицы `Сотрудники_1`. Чтобы не было в запросе два поля с одним именем, изменим наименование столбца в запросе для поля `Должность` из таблицы `Сотрудники_1`. Введите имя `Начальник` и поставьте двоеточие, как на рис. 4.29.
6. Нажмите кнопку **Запуск** (Run). Вы должны получить таблицу сотрудников, из которой видно, что Менеджер по продажам Кротов Андрей командует тремя сотрудниками и сам подчиняется вице-президенту компании. Все остальные сотрудники непосредственно подчиняются вице-президенту, а вице-президент не подчиняется никому (рис. 4.30).

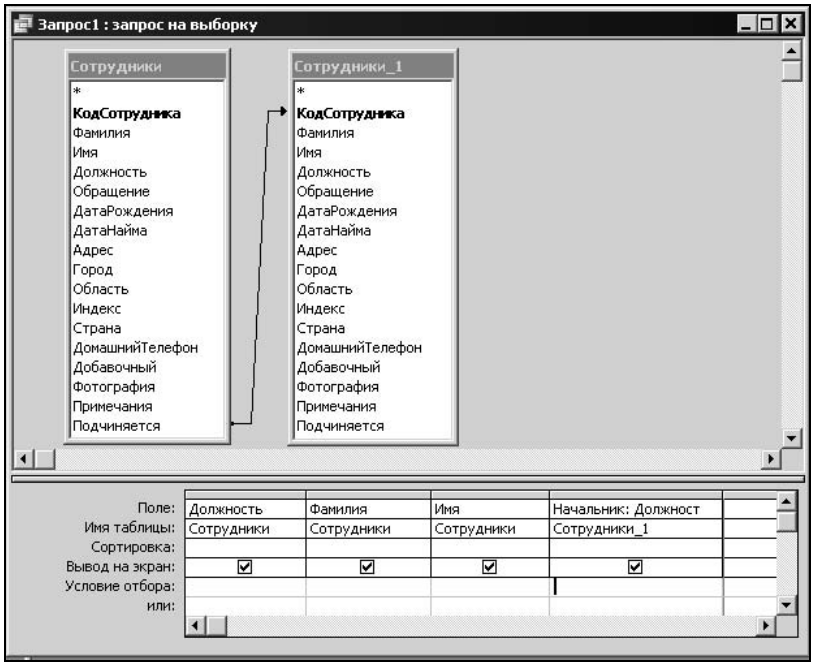


Рис. 4.29. Отображение запроса с рекурсивным соединением в окне Конструктора запросов



Рис. 4.30. Запрос с рекурсивным соединением

Данный пример, конечно, очень простой, но если нужно передать несколько уровней иерархии и таблица содержит больше записей, такое представление может быть полезно вследствие своей наглядности.

Запросы с параметрами

Запрос в Access является объектом, который сохраняется в файле базы данных и может многократно повторяться. Все запросы, которые мы демонстрировали до сих пор, содержали конкретные значения дат, названий, имен и т. д. Если требуется

повторить такой запрос с другими значениями в условиях отбора, его нужно открыть в режиме Конструктора, изменить условие и выполнить. Чтобы не делать многократно этих операций, можно создать запрос с параметрами. При выполнении такого запроса выдается диалоговое окно **Введите значение параметра** (Enter Parameter Value), в котором пользователь может ввести конкретное значение и затем получить нужный результат.

Покажем, как создавать запросы с параметрами на примере запроса `Отсортированный список товаров`, который мы создавали ранее. Теперь с помощью этого запроса попробуем отобрать товары, поставляемые определенным поставщиком. Для этого выполните следующее:

1. Откройте данный запрос в режиме Конструктора.
2. Чтобы определить параметр запроса, введите в строку **Условие отбора** (Criteria) для столбца `Название` (CompanyName) вместо конкретного значения слово или фразу и заключите их в квадратные скобки, например `[Поставщик:]`. Эта фраза будет выдаваться в виде приглашения в диалоговом окне при выполнении запроса.
3. Если вы хотите, чтобы Access проверяла данные, вводимые в качестве параметра запроса, нужно указать тип данных для этого параметра. Обычно в этом нет необходимости при работе с текстовыми полями, т. к. по умолчанию параметру присваивается тип данных **Текстовый** (Text). Если же данные в поле запроса представляют собой даты или числа, рекомендуется тип данных для параметра определять. Для этого щелкните правой кнопкой мыши на свободном поле в верхней части запроса и выберите из контекстного меню команду **Параметры** (Parameters) или выполните команду меню **Запрос | Параметры** (Query | Parameters). Появляется диалоговое окно **Параметры запроса** (Query Parameters), представленное на рис. 4.31.

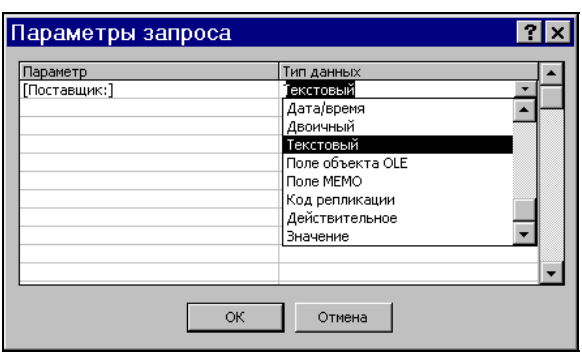


Рис. 4.31. Диалоговое окно **Параметры запроса**

4. В столбец **Параметр** (Parameter) нужно ввести название параметра точно так, как он определен в бланке запроса (легче всего это сделать путем копирования через буфер обмена), только можно не вводить квадратные скобки. В столбце **Тип данных** (Data Type) выберите из раскрывающегося списка необходимый тип данных. Нажмите кнопку **OK**.

5. Нажмите кнопку **Запуск** (Run) на панели инструментов, чтобы выполнить запрос. При выполнении запроса появляется диалоговое окно **Введите значение параметра** (Input Parameter Value) (рис. 4.32), в которое нужно ввести значение, например `Tokyo Traders`. Результат выполнения запроса представлен на рис. 4.33. В него попадают только те товары, которые поставляются данным поставщиком.

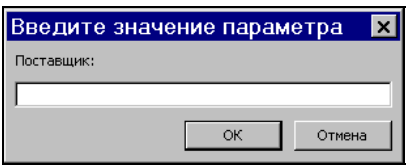


Рис. 4.32. Диалоговое окно **Введите значение параметра**



Рис. 4.33. Результат выполнения запроса с параметром

В одном запросе можно ввести несколько параметров. При выполнении такого запроса для каждого из параметров будут поочередно выводиться диалоговые окна **Введите значение параметра** в том порядке, в котором параметры перечислены в бланке запроса.

Совет

При использовании запроса с параметрами делайте понятными их имена, т. к. именно они будут выводиться в диалоговом окне для пользователя.

Создание запроса из фильтра

Одним из способов создания запроса является сохранение фильтра в виде запроса. Ранее мы говорили, что фильтр используется при просмотре таблицы для отбора нужных записей. Если вы создали достаточно сложный фильтр в окне **Фильтр** (Filter by Form) и знаете, что его придется использовать в дальнейшем, можно сохранить его в базе данных в виде запроса. Для этого достаточно нажать кнопку **Сохранить как запрос** (Save as query) на панели инструментов при открытом окне **Фильтр** (Filter by Form).

1. Откройте в режиме Таблицы таблицу `Клиенты` (Customers).
2. Нажмите кнопку **Изменить фильтр** (Filter by form) на панели инструментов **Таблица в режиме таблицы** (Table datasheet). Появится окно, в котором можно задать фильтр.

3. Выберите из списка в поле Должность (ContactTitle) значение Менеджер по продажам (Sales Manager).
4. Проверьте, как работает фильтр, нажав кнопку **Применение фильтра** (Apply filter) на панели инструментов **Фильтр** (Filter). Убедившись, что фильтр работает правильно, вы можете сохранить его для дальнейшего использования.
5. Нажмите кнопку **Изменить фильтр** (Filter by form) на панели инструментов еще раз, чтобы вернуться в окно определения фильтра.
6. Нажмите кнопку **Сохранить как запрос** (Save As Query) на панели инструментов **Фильтр** (Filter) или выполните команду **Файл | Сохранить как запрос** (File | Save As Query). Появится диалоговое окно (рис. 4.34), в котором необходимо указать имя запроса, например Менеджеры по продажам, и нажать кнопку **OK**.
7. Снимите фильтр, нажав кнопку **Очистить бланк** (Clear Grid) на панели инструментов **Фильтр** (Filter), и закройте окно фильтра, нажав кнопку **Закрыть** (Close).



Рис. 4.34. Сохранение фильтра в виде запроса

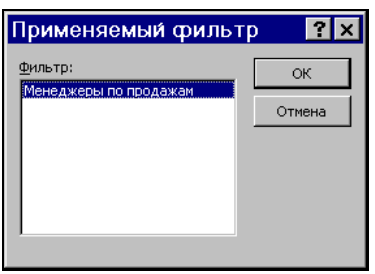


Рис. 4.35. Диалоговое окно **Применяемый фильтр**

Сохраненный фильтр может быть легко использован в дальнейшем, причем двумя способами:

1. Можно установить сохраненный фильтр, снова открыв таблицу Клиенты (Customers). Для этого нужно сначала открыть окно определения фильтра, нажав кнопку **Изменить фильтр** (Filter by form). Затем нажать кнопку **Загрузить из запроса** (Load from Query) на панели инструментов или выполнить команду **Файл | Загрузить из запроса** (File | Load from Query). При этом появится диалоговое окно **Применяемый фильтр** (Applicable Filter) (рис. 4.35), в котором требуется выделить в списке необходимый фильтр и нажать кнопку **OK** или просто дважды щелкнуть левой кнопкой мыши по названию фильтра. Чтобы применить фильтр, нужно нажать кнопку **Применение фильтра** (Apply filter) на панели инструментов.

Замечание

Обратите внимание, что в списке фильтров не присутствуют другие запросы, есть только фильтр, сохраненный как запрос.

2. Выполнить запрос Менеджеры по продажам. Для этого нужно раскрыть список запросов в окне базы данных, найти запрос Менеджеры по продажам и открыть его в режиме Таблицы, нажав кнопку **Открыть** (Open). Вы получите тот же результат,

что и в первом случае, когда применяется фильтр к открытой таблице. Когда запрос открыт в режиме Таблицы, или применен фильтр к таблице, нажмите кнопку **Вид | Конструктор** (View | Design view) на панели инструментов. Появится окно Конструктора запроса, в котором отображается запрос, соответствующий фильтру (рис. 4.36).

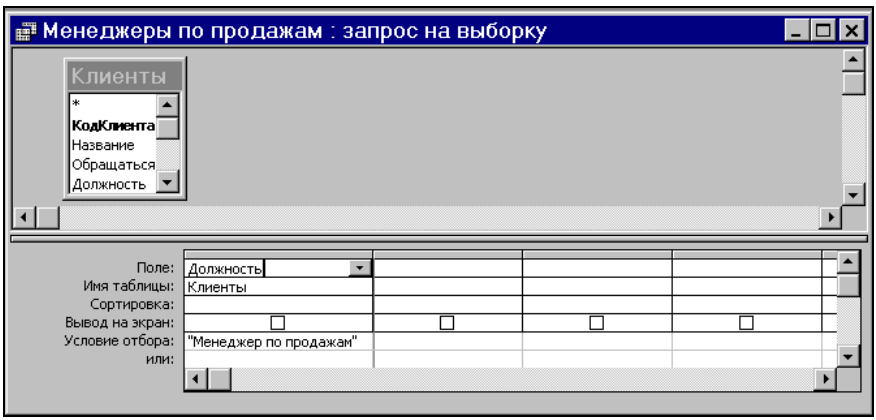


Рис. 4.36. Запрос Менеджеры по продажам в режиме Конструктора

Обратите внимание, что хотя в режиме Таблицы запрос отображал все столбцы исходной таблицы, в бланке запроса указан только один столбец Должность (ContactTitle), который содержит условие отбора, и тот не включен в результат запроса. Чтобы понять, почему так происходит, откройте запрос в режиме SQL. Для этого нужно нажать кнопку **Вид | Режим SQL** (View | SQL View). Появится диалоговое окно, содержащее инструкцию языка SQL, которая и обрабатывается при выполнении запроса (рис. 4.37). Эта инструкция имеет вид:

```
SELECT *
FROM Клиенты
WHERE (((Клиенты.Должность)= "Менеджер по продажам"));
```

Значок * означает, что выбираются *все* поля из таблицы Клиенты.

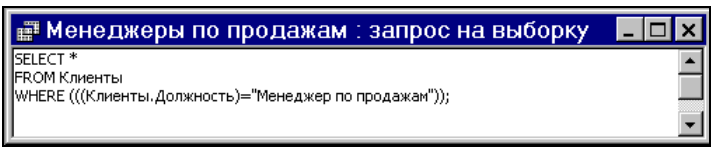


Рис. 4.37. Запрос Менеджеры по продажам в режиме SQL

В виде запроса можно сохранить так называемый *расширенный фильтр*. Расширенный фильтр используется в том случае, если требуется задать сложные критерии отбора и сортировки записей. Чтобы создать расширенный фильтр для открытой таблицы, необходимо выполнить команду **Записи | Фильтр | Расширенный фильтр** (Records | Filter | Advanced Filter/Sort). При этом появляется окно расширенного

фильтра, которое очень напоминает окно Конструктора запроса. На рис. 4.38 представлено окно расширенного фильтра для таблицы `Заказы` (Orders). Чтобы создать такой фильтр, мы выполнили следующие действия:

1. Открыли в режиме Таблицы таблицу `Заказы` (Orders).
2. Выделили в поле `Сотрудник` значение `Воронова, Дарья` и нажали кнопку **Фильтр по выделенному** (Filter by Selection) на панели инструментов, отфильтровав таким образом заказы, обработанные конкретным сотрудником.
3. Выполнили команду **Записи | Фильтр | Расширенный фильтр** (Records | Filter | Advanced Filter/Sort).
4. Перенесли в бланк фильтра два поля: `ДатаРазмещения` (OrderedDate) и `НазваниеПолучателя` (ShipName).
5. Для поля `ДатаРазмещения` (OrderedDate) указали критерий выборки `Between #01.03.1998# And #30.06.1998#`, означающий, что отбираться будут только заказы, поступившие во втором квартале 1998 года.
6. Определили порядок сортировки записей таким образом, что сначала будут сортироваться заказы по дате размещения, а в пределах одной даты — по наименованиям клиентов (в алфавитном порядке).

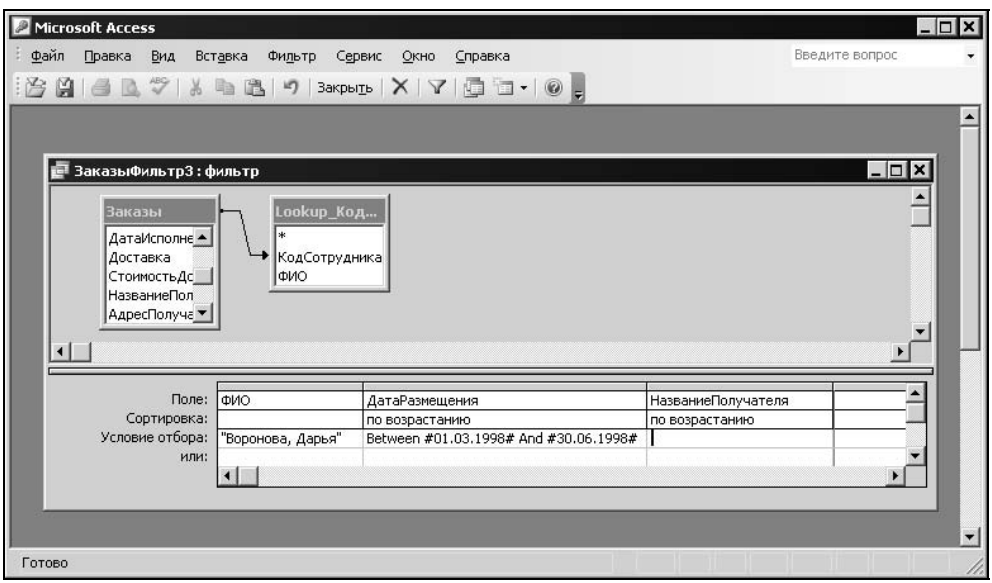


Рис. 4.38. Окно расширенного фильтра

Окно расширенного фильтра действительно очень похоже на окно Конструктора запросов, однако панель инструментов очень сильно отличается от панели инструментов Конструктора запросов. Например, нельзя добавить еще одну таблицу (расширенный фильтр строится только для одной таблицы — в данном примере вторая таблица отображает поле подстановки) или выполнить группировку записей. На панели инструментов есть только две важные кнопки: **Применение фильтра** (Apply

filter) и **Сохранить как запрос** (Save as query). В бланке фильтра отсутствует строка, содержащая флажки включения поля в запрос. Это означает, что фильтр не влияет на состав столбцов в результирующей таблице — всегда отображаются все столбцы, фильтруются только записи. Однако критерии отбора записей могут быть установлены сколь угодно сложные. Задаются они по тем же правилам, что и в запросах.

Выполнение запроса

Чтобы посмотреть результаты запроса, достаточно открыть его в режиме Таблицы. Для этого необходимо:

1. В окне базы данных на панели объектов выбрать ярлык **Запросы** (Queries).
2. Выделить нужный запрос в списке запросов и нажать кнопку **Открыть** (Open) или дважды щелкнуть левой кнопкой мыши на нужном запросе.

После этого на экране появляется таблица, которая содержит только те записи, которые удовлетворяют критериям отбора, указанным в запросе, и те поля, которые указаны в бланке запроса. Если в запросе указан порядок сортировки записей, они выводятся на экран в соответствующем порядке.

Внешне отображение результатов запроса не отличается от отображения таблицы. Все режимы работы с таблицей, описанные в *гл. 2*, применимы и к результирующей таблице запроса. С ней можно работать так же, как и с обычной таблицей, в частности не только просматривать, но и изменять данные. Изменения сохраняются в записях базовой таблицы, на основе которой построен запрос. Если запрос создан на основе двух и более связанных таблиц, то не всегда столбцы в запросе можно редактировать. *(Подробную информацию об этом см. в следующем разделе.)*

Для выполнения запроса из режима Конструктора достаточно просто переключиться в **Режим таблицы** (Datasheet View) с помощью кнопки **Вид** (View) на панели инструментов, как мы это уже не раз делали, или с помощью команды **Режим таблицы** (Datasheet View) из меню **Вид** (View).

В Access 2003 запрос можно открыть не только в режиме Таблицы. Вы можете выполнить обработку результатов запроса и получить на их основе сводную таблицу или диаграмму, аналогичные тем, которые создаются на базе таблицы Excel. Эту возможность мы рассмотрим подробно в *гл. 8*.

Изменение данных в результирующем множестве запроса

Когда результат запроса отображается в виде таблицы, возникает желание не только просматривать, но и изменять данные в этой таблице. Поскольку данная таблица является виртуальной, на самом деле редактирование полей запроса означает редактирование полей в таблицах, на основе которых строился запрос. Однако такое редактирование не всегда возможно, и при попытке изменить значение некоторого поля вы можете получить сообщение, что данные в запросе не являются обновляемыми, или просто звуковой сигнал. В настоящем разделе мы рассмотрим условия, при которых данные в запросе могут обновляться.

Является ли запрос обновляемым, можно достаточно легко обнаружить визуально: при отображении результатов запроса в конце записей должна быть пустая строка, которая помечена значком звездочки (*) в области выделения записи слева. Тогда в эту строку можно вводить данные, которые создадут новую запись. Если такая строка отсутствует, добавлять записи в запрос и изменять поля запроса нельзя.

Любой запрос имеет два свойства: **Уникальные значения** (Unique Values) и **Уникальные записи** (Unique Rows). Свойство **Уникальные записи** (Unique Row) исключает из результирующего набора дублированные записи, т. е. в результат запроса будут включены записи, которые имеют отличающиеся значения хотя бы в одном из полей. Свойство **Уникальные значения** (Unique Values) требует включения в результирующий набор только тех записей, которые имеют уникальные значения во всех полях. Значения этих свойств не могут одновременно быть равными **Да** (Yes), хотя оба могут иметь значения **Нет** (No). Значения этих свойств могут устанавливаться в окне свойств запроса. Большинство запросов, свойство **Уникальные записи** (Unique Row) которых имеет значение **Да** (Yes), могут использоваться для обновления данных. Запросы, свойство **Уникальные значения** (Unique Values) которых имеет значение **Да** (Yes), не допускают ни обновления имеющихся в них записей, ни добавления новых.

Вы не сможете добавлять и изменять записи в запросах, если:

- таблица в запросе является связанной и для нее не определен первичный ключ;
- две таблицы запроса связаны отношением "один-ко-многим" и при этом в таблице "один" не задано полей первичного ключа;
- в запросе используются рекурсивные соединения;
- в запросе применяются статистические функции SQL;
- база данных открыта в режиме "только для чтения" (read only) либо находится в папке, к которой текущему пользователю предоставлено разрешение только на чтение данных;
- в запросе объединены несколько (больше двух) таблиц отношением "один-ко-многим";
- если таблицы связаны отношением "один-ко-многим", то невозможно добавление записей, когда в запросе не отображается связывающее поле таблицы на стороне "многие";
- если таблицы в запросе связаны отношением "один-ко-многим", то не все поля в запросе могут обновляться.

Условия, при которых можно добавить или обновить записи в запросе:

- таблица является единственной в запросе;
- таблицы в запросе связаны отношением "один-к-одному";
- если таблицы в запросе связаны отношением "один-ко-многим", то установите каскадное обновление для этих таблиц;
- если не обновляются поля таблицы на стороне "многие" запроса, попробуйте сначала сохранить изменения, сделанные в таблице на стороне "один".

Печать результатов запроса

Результаты запроса можно не только посмотреть на экране, но и напечатать. Сделать это можно несколькими способами:

1. Открыть запрос в режиме Таблицы и нажать кнопку **Печать** (Print) на панели инструментов. При этом таблица выводится на принтер, установленный по умолчанию, и используются параметры печати, заданные по умолчанию.
2. Открыть запрос в режиме Таблицы и выполнить команду **Файл | Печать** (File | Print). В этом случае перед печатью таблицы на экран выводится стандартное окно **Печать** (Print) (рис. 4.39), в котором можно изменить параметры печати, заданные по умолчанию. Например, можно выделить в результирующей таблице только некоторые записи, а затем в диалоговом окне **Печать** (Print) установить переключатель **выделенные записи** (Selected Record(s)). Тогда будут напечатаны только выделенные записи.

Настройка параметров печати при печати запроса выполняется так же, как и при печати обычной таблицы *(см. разд. "Работа с таблицами" гл. 2)*.

Совет

Перед выводом запроса на печать полезно посмотреть его в режиме предварительного просмотра. Для этого нужно нажать кнопку **Предварительный просмотр** (Print Preview) на панели инструментов. Назначение кнопок панели инструментов и возможные действия с запросом в режиме предварительного просмотра аналогичны описанным в *разд. "Печать таблицы" гл. 2*.

Можно напечатать результат запроса и прямо из окна базы данных, не открывая его. Для этого необходимо:

1. Выделить требуемый запрос в списке запросов окна базы данных.
2. Щелкнуть правой кнопкой мыши по названию запроса.
3. Выбрать из контекстного меню команду **Печать** (Print).

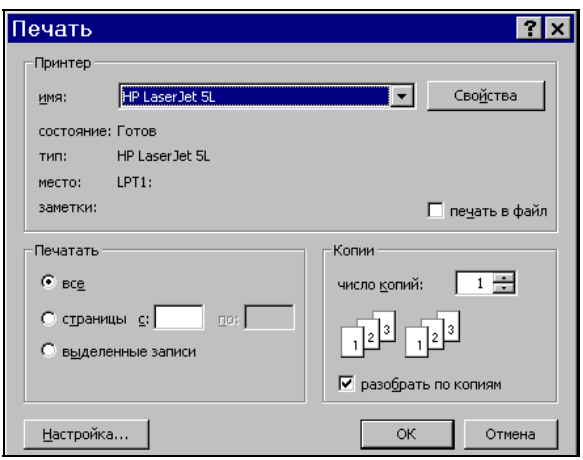


Рис. 4.39. Диалоговое окно **Печать**

Передача результатов запроса в другие приложения

Поскольку результатом запроса является виртуальная таблица, то данные из этой таблицы могут экспортироваться точно так же, как данные из обычной таблицы, т. е. все форматы экспорта, применимые к таблице, применяются и к запросу. В силу этого не имеет смысла описывать способы передачи результатов запросов в данной главе, вместо этого мы отсылаем читателя к *гл. 3*. Сам запрос, как объект базы данных Access, может быть экспортирован в другую базу данных Access (но не в проект Access (.adp)).

Использование выражений в запросах

При работе с объектами Access очень широко используются *выражения*. Примеры простых выражений приводились ранее, например, при вводе условий на значение поля в таблице *(см. "Работа с таблицами" гл. 2)*.

Выражения активно используются в запросах для описания критериев выборки записей, как уже упоминалось раньше в данной главе. В последующих главах книги при описании объектов Access мы будем постоянно указывать, где и как могут использоваться выражения. В этом разделе мы опишем правила создания выражений в Access, которые должны использоваться в дальнейшем во всех случаях применения выражений.

Выражения в Access

Каждое выражение может содержать один или несколько операторов и одну или несколько констант, идентификаторов или функций. Выражение может быть сколь угодно сложным.

- *Константы* — характеризуют неизменные значения. Их часто используют для создания значений по умолчанию и для сравнения значений в полях таблиц. Значения констант определяются пользователем при вводе выражения. (Например, `09`, страна фирмы изготовителя `Швеция`.)
- *Идентификаторы* — это имена объектов в Access (например, полей таблиц или запросов), которые при вычислении выражений заменяются их текущими значениями (для определения этой операции обычно используется термин *возвращают*). Например, идентификатор имени поля `[Household Inventory]` в выражении возвращает значение поля `Household Inventory` из текущей выделенной записи. Именованные константы и переменные, применяемые в программах Visual Basic для приложений (VBA), тоже являются идентификаторами. Существует несколько встроенных именованных констант, служащих идентификаторами: `True`, `False`, `Yes`, `No` и `Null`.

Замечание

Если имя поля или таблицы содержит пробелы, его идентификатор обязан в выражении заключаться в квадратные скобки. Чтобы упростить ввод идентификаторов,

рекомендуется не использовать пробелы в названиях таблиц, полей в таблицах и названиях других объектов Access.

- *Функции* — возвращают в выражение значение вместо имени функции. В отличие от идентификаторов, большинство функций требуют заключать в скобки свои аргументы — идентификаторы или значения подвыражений. Например, функция `Date()`, имеющая пустой список аргументов, возвращает текущую дату.
- *Операторы* — обычные значки арифметических операций (+, −, *, /) и другие символы и аббревиатуры. Большинство из них эквивалентны операторам традиционных языков программирования типа Basic. Некоторые специфичны для Access или SQL, например `Between`, или `Like`. Используемые в операции константы, идентификаторы и функции называются *операндами*.

Для создания выражений в Access существует шесть категорий операторов: арифметические, логические, присваивания, конкатенации, идентификации и сравнения с образцом.

Рассмотрим далее типы операторов, а также идентификаторы объектов и функций.

Арифметические операторы

Арифметические операторы, как следует из названия, выполняют сложение, вычитание, умножение и деление. Арифметические операторы оперируют только с числовыми значениями и должны, за исключением *унарного минуса*, иметь два числовых операнда.

В табл. 4.2 приведен список арифметических операторов для выражений Access.

Таблица 4.2. *Арифметические операторы*

Оператор	Пример	Описание
+	`[Итог] + [Надбавка]`	Складывает два операнда
−	`Date() - 7`	Считает разность двух операндов
− (унарный)	`-12345`	Меняет знак операнда
*	`[Коробок] * [Цена коробки]`	Перемножает два операнда
/	`[Количество] / 12.55`	Делит один операнд на другой
\	`[Коробок] \ 2`	Делит один целый операнд на другой нацело. При использовании деления нацело операнды с десятичными дробями округляются до целого, а дробные части отбрасываются
Mod	`[Коробок] Mod 12`	Возвращает остаток от деления нацело. Например, `15 Mod 12` равно 3
^	`[Основание] ^ [Показатель]`	Возводит операнд `Основание` в степень `Показатель`

В таблице отсутствует знак равенства (=), поскольку он отнесен к группе операторов присваивания и сравнения.

Операторы присваивания и сравнения

Обычно в качестве оператора присваивания значения объекту, переменной или константе используется знак равенства (=). Например, выражение `=Now()` может присваивать полю таблицы значение по умолчанию, и тогда знак равенства действует как оператор присваивания. С другой стороны, знак = представляет собой оператор сравнения, определяющий, равны ли два операнда.

Оператор сравнения соотносит значения двух операндов и возвращает логические значения (`True` или `False`), соответствующие результату сравнения. Основное назначение операторов сравнения — создание условий на значение, установление критериев выборки записей в запросах, определение действий макросов и контроль выполнения программ в VBA. В табл. 4.3 приведен список операторов сравнения Access.

Таблица 4.3. *Операторы сравнения*

Оператор	Пример	Результат	Описание
<	1 < 100	True	Меньше
<=	1 <= 1	True	Меньше либо равно
=	1 = 100	False	Равно
>=	100 >= 1	True	Больше либо равно
>	100 > 100	False	Больше
<>	1 <> 100	True	Неравно

Замечание

Если один из операндов имеет значение `Null` (пустое значение), то любое сравнение возвращает значение `Null` (тоже пустое значение).

Логические операторы

Логические (булевы) операторы используются для объединения результатов двух или более выражений сравнения в единое целое:

- `And` — конъюнкция (логическое И);
- `Or` — дизъюнкция (логическое ИЛИ);
- `Not` — логическое отрицание;
- `Xor` — исключающее ИЛИ;
- `Eqv` — логическая эквивалентность;
- `Imp` — логическая импликация.

Они могут состоять только из выражений, возвращающих логические значения `True`, `False` или `Null`. В противном случае выполняется побитовое сравнение. Логические операторы всегда требуют двух операндов, за исключением `Not` — логического эквивалента унарного минуса.

В табл. 4.4—4.6 приведены результаты логических операторов Access в зависимости от значения операндов.

Таблица 4.4. *Зависимость значения логических операторов от значения операндов*

	A=True B=False	A=True B=True	A=False B=False	A=False B=True
A And B	False	True	False	False
A Or B	True	True	False	True
Not A	False	False	True	True
A Xor B	True	False	False	True
A Eqv B	False	True	True	False
A Imp B	False	True	True	True

Для всех логических операторов, кроме `Imp`, значение одного из операндов `Null` приводит к значению результата `Null`.

Таблица 4.5. *Значения оператора* `Imp` *с* `Null`*-операндами*

	A=Null B=False	A=Null B=True	A=False B=Null	A=True B=Null	A=Null B=Null
A Imp B	Null	True	True	Null	Null

В случае если сравниваются два выражения и, по крайней мере, одно из них не является константой из множества (`True`, `False`, `Null`), логические операторы приводят к побитовому сравнению выражений-операндов (табл. 4.6).

Таблица 4.6. *Значения логических операторов для однобитовых операндов*

Значения соответствующего бита результата	бит A=1 бит B=0	бит A=1 бит B=1	бит A=0 бит B=0	бит A=0 бит B=1
A And B	0	1	0	0
A Or B	1	1	0	1
Not A	0	0	1	1
A Xor B	1	0	0	1
A Eqv B	0	1	1	0
A Imp B	0	1	1	1

Операторы слияния строковых значений (конкатенации)

Стандартный значок оператора конкатенации SQL амперсанд (&) является более предпочтительным, чем значок плюса (+), хотя оба они приводят к одинаковому результату: объединению двух текстовых значений в единую строку символов. Применение значка плюс (+) двусмысленно, его основное назначение — сложение двух числовых операндов.

Пример: слияние `"Visual " & "Basic"` дает `"Visual Basic"`. Обратите внимание на дополнительный пробел в первом слове, без него результат выглядел бы несколько иначе: `"VisualBasic"`.

Операторы идентификации "!" и "."

Операторы идентификации применяются в качестве разделителей в ссылках на объекты (оператор "!"), их методы или свойства (оператор "."):

- `КлассОбъекта!ИмяОбъекта` или
- `КлассОбъекта!ИмяОбъекта.Свойство` или
- `КлассОбъекта!ИмяОбъекта.Метод()` или
- `ИмяОбъекта.Свойство` или
- `ИмяОбъекта.Метод()`.

Эти операторы позволяют объединять имена объектов и классов объектов для отбора специфических объектов или их свойств, различать имена объектов и их свойств, идентифицировать определенные поля в таблицах.

Например:

- `Forms!Категории, Tables!Категории`

  идентификация формы и таблицы с одинаковыми именами.

- `MyTextbox.Caption = "Будьте внимательны!"`

  здесь `MyTextbox` — объект управления, а `Caption` — свойство.

- `Заказы![Код клиента]`

  определяет поле `Код клиента` (CustomerID) в таблице `Заказы` (Customers).

Благодаря наличию операторов идентификации можно присваивать полям разных объектов одни и те же имена (поэтому, например, свойство `Caption` есть у большинства объектов).

Операторы сравнения с образцом

Остальные операторы Access (табл. 4.7) упрощают создание выражений для выборки записей в запросах и относятся к операторам сравнения с образцом. Эти операторы возвращают `True` или `False`, в зависимости от соответствия значения в поле выбранной спецификации оператора. Наличие этих операторов в условиях на значение позволяет либо включать запись в запрос, если логическое значение, возвращаемое выражением, равно `True`, либо отвергать, если это значение — `False`.

Таблица 4.7. *Другие операторы, используемые в выражениях Access*

Оператор	Пример	Описание
`Between`	`Between (-100) And (100)`	Определяет, находится ли числовое значение в определенном диапазоне значений
`Is`	`Is Null` `Is Not Null`	При использовании вместе с `Null` определяет, является ли значение `Null` или `Not Null`
`In`	`In("Москва", "Киев", "Санкт-Петербург")`	Определяет, является ли строковое значение элементом списка значений
`Like`	`Like "Ив*"` `Like "db??"`	Определяет, начинается ли строковое значение с указанных символов (для правильной работы `Like` нужно добавить символ шаблона "`*`" или один или несколько символов "`?`")

Для пользователей, знакомых с шаблонами командной строки DOS, применение знаков (символов шаблона) "*" и "?" в операторах `Like` не вызовет затруднений. Точно так же, как в DOS, символ "*" замещает любое число знаков, а символ шаблона "?" замещает только один знак, поэтому `Like "Ив*"` вернет `True` для значений "Иванов", "Иващенко", "Иволгин". Для "db1" или "db100" значение оператора `Like "db??"` будет `False`, но тот же оператор вернет для "db10" и "dbXX" значение `True`. Символы шаблона "`*`" и "`?`" могут стоять в любом месте шаблонной строки, например:

- `Like "*ms*.?*"` — возвращает `True` только для значений, содержащих в своем теле подстроку "ms", после которой (может быть сразу, а может быть через несколько символов) следует не менее двух символов, первый из которых точка (.);
- `Like "?1???*"` — возвращает `True` только для значений, состоящих не менее чем из пяти символов, вторым из которых является "1".

Идентификаторы объектов

У любого объекта Access есть имя, по которому его можно однозначно идентифицировать в некоторой системе объектов. Кроме обозначения коротким именем объекта, идентификатор можно обозначить *квалифицированным* (или *полным*) именем, когда объект идентифицируется как один из объектов в семействе объектов. В этом случае имя идентификатора состоит из *имени семейства* (класс объекта), отделенного от *присвоенного имени* (имени объекта) восклицательным знаком или точкой (символами операции идентификации "!" и "."). Поэтому имена объектов не должны содержать символов "!" и ".". В SQL разделитель имен объектов — точка, в Access для разделения имен таблиц и имен полей используется "!", а точка разделяет объекты и их свойства. Используя идентификаторы, можно возвращать значения полей в объекты форм и отчетов, а также строить новые выражения.

Пример идентификаторов объектов в базе данных `Борей` (Northwind):

`[Заказы]![ДатаНазначения]`

— идентификатор поля `ДатаНазначения`, находящегося в таблице `Заказы` (Orders).

Константы

В этом разделе описываются константы в их явном представлении. Кроме явного представления константы могут быть именованными. Именованные константы создаются пользователем и используются в программах на Visual Basic для приложений (VBA). Кроме того, Access имеет много встроенных констант, имена которых рассматриваются как зарезервированные слова и не могут использоваться для других целей.

Access различает три вида констант: числовые, строковые, константы даты и времени.

- *Числовыми* константами называются последовательности цифр, и, при необходимости, символы знака числа (+) и (−), символ десятичной точки (.), а для записи числа в экспоненциальной нотации, символы порядка (`E`) или (`e`). Для положительного числа знак (+) не обязателен.

  Пример записи числовых констант: `12345`; `-12.345`; `-6.76E-23`.

- *Строковыми* или *(текстовыми)* константами называются любые символы, возвращаемые функцией `Chr$()`, и их комбинации. Функция `Chr$()` возвращает символ в кодировке ANSI, которой пользуется Windows. Кроме печатаемых символов, в которые входят буквы, цифры, знаки пунктуации и другие специальные символы клавиатуры (например, "@", "~" и т. п.), результатом функции могут быть и управляющие символы, такие как символ табуляции <Tab>, возврат каретки и перевод строки, аналогичный по результату нажатию клавиши <Enter> и др. Печатаемые символы должны быть заключены в двойные прямые кавычки. Часто, например при вводе строковых констант в ячейки таблиц или бланки запросов, Access сам добавляет кавычки, в иных случаях это придется сделать вручную.

  Примером сложной строковой константы может служить выражение:

  ```
  Chr$(9) & "Отступ" & Chr$(10)& Chr$(13)& "Новая строка"
  ```

  Здесь `Chr$(9)` — символ табуляции <Tab>; `Chr$(10)` — символ возврата каретки; `Chr$(13)` — символ перевода строки. Все подстроки объединены с помощью оператора конкатенации.

- *Константы даты/времени* в Access выделяются символами "#". Как и в случае остальных констант, если программа распознает ввод в бланк запроса даты или времени в стандартном формате Access, эти символы будут добавлены автоматически.

  Пример констант даты и времени: `#26/04/75#`, `#22-Mar-74#`, `#10:35:30#`.

Функции

Функция, так же как и идентификатор, используется для возвращения значения в точку вызова. Значение, возвращаемое функцией, определяется ее типом, например функция `Now ()` возвращает дату и время часов компьютера. Синтаксически функция выделяется круглыми скобками, следующими сразу же за ее идентификатором. Многие функции требуют наличия аргументов, которые записываются в этих скобках через запятую при обращении к функции. Функции можно использовать для создания нового выражения или функции. *(Об использовании функций рассказывается в гл. 13.)*

Встроенные в Access функции можно сгруппировать по следующим категориям (забегая вперед, скажем, что именно так они сгруппированы в специальном инструменте, помогающем правильно создавать сложные выражения, который называется Построитель выражений (Expression Builder)).

- Функции *работы с массивами* — используются для определения границ размерностей массивов при программировании на VBA.
- Функции *преобразования типов данных* — применяются для преобразования одного типа данных в другой. Например, для преобразования числа в строку служит функция `Str()`, а обратно — функция `Val()`.
- С помощью функций *работы с объектами баз данных* осуществляется обращение к объектам баз данных: таблицам, запросам, формам, отчетам, макросам и модулям. Например, функция `CurrentDB()` возвращает ссылку на объект `Database`, с которым работает Access в данный момент. *(Для более подробной информации о работе с этими функциями обратитесь к справочной системе Access или к гл. 13.)*
- Функции *даты и времени* — служат для проведения операций со значениями даты и времени. Например, функция `DateDiff()` вычисляет промежуток между двумя датами, а функция `Date()` возвращает значение текущей даты.
- Функции *динамического обмена данными (DDE)* между приложениями Windows и функции *работы с использованием технологии связывания и внедрения (OLE)* (например, `DDE()` и `DDESend ()`) — используются для перемещения данных из одного приложения в другое и обратно. Для приложений, поддерживающих технологию OLE, в частности Microsoft Office, особенно важными являются функции `GetObject()` и `CreateObject()`, с помощью которых начинается основная работа с любыми объектами *(см. разд. "Интеграция Access 2003 с другими компонентами Office 2003" гл. 15)*.
- *Статистические функции SQL* — используются чаще всего для многотабличных запросов как источника данных для форм. Статистические функции SQL возвращают статистические данные из записей, выбранных по запросу. Эти функции нельзя использовать в макросах или вызывать их из программ VBA иначе, как внутри выражений в кавычках, используемых в инструкциях SQL для создания объектов `Recordset`.

  Статистические функции по подмножествам записей — это те же статистические функции SQL, но работающие по большей части с вычисляемыми значениями, а не со значениями, содержащимися в полях запросов.

  Примером статистической функции SQL является `StDev()`, а соответствующей ей статистической функции по подмножеству записей — `DStDev()`. Обе они вычисляют стандартное отклонение для заданного множества значений.
- Функции *обработки ошибок* — используются, как следует из названия, для отслеживания ошибок. Например, функция `Error()` выводит сообщение об ошибке по ее номеру. Наличие таких функций в выражении можно считать признаком хорошего тона, поскольку используемое выражение, хоть и приводит к ошибочному результату, но продолжает вести себя корректно, не забыв предупредить пользователя.
- *Финансовые* функции — идентичны своим двойникам в Microsoft Excel и используют те же аргументы. Например, функция `Rate()` возвращает процентную став-

ку, необходимую для получения путем регулярных взносов указанной суммы на базе, имеющейся за определенный срок.

- Функции *общего назначения* — используются в основном при программировании на VBA для извлечения вспомогательной информации и управления ходом выполнения программы. Например, функция `Command()` служит для извлечения аргументов командной строки при выполнении программы, написанной на VBA.
- Функции *сообщений* и *ввода/вывода* — позволяют выводить сообщения или вводить новые данные, а также устанавливать различные параметры ввода/вывода. Иллюстрацией может служить функция ввода данных `InputBox()` или функция проверки существования каталога или файла `Dir()`.
- Функции *проверки* — особая группа функций, отвечающих на вопрос: "Является ли аргумент... ?". Например, `IsNumeric()` — возвращает `True`, если аргумент имеет один из числовых типов данных, и `False` — в противном случае; `IsObject()` — возвращает `True`, если аргумент — объект OLE Automation, и `False` — в противном случае.
- Функции *ветвления* — используются для выбора из нескольких альтернатив. На них стоит остановиться подробнее.
  - `IIf (Expr, Truepart, Falsepart)` — возвращает значение выражения `Truepart`, если значение выражения `Expr` равно `True`, или значение выражения `Falsepart`, если значение `Expr` есть `False`. Особо важно то, что вне зависимости от значения выражения `Expr` будут вычислены как выражение `Truepart`, так и `Falsepart`. Стоит быть очень внимательным к возможным побочным эффектам, связанным с этой особенностью функции `IIf()`. Отметим, что здесь, как и в большинстве функций, в качестве каждого из аргументов стоит выражение, которое, в свою очередь, тоже может быть сложным выражением.
  - Функция `Choose()` возвращает значение, соответствующее заданному положению в списке значений. `Switch()` возвращает значение, связанное с первым из последовательности выражением, имеющим значение `True`.
  - Функции `Choose()` и `Switch()` похожи на команду `Select Case` из VBA и других диалектов Basic, которая не раз встретится вам в *гл. 13*.
- *Математические и тригонометрические* функции — используются для выполнения простых (и не слишком) математических операций, например вычисления логарифма `Log()` или синуса числа `Sin()`.
- *Текстовые* функции — позволяют проводить различные операции над строками. Например, функция `Trim()` возвращает строку, заданную в качестве аргумента, без начальных и заключительных пробелов.

Замечание

Еще раз обратим внимание, что описание всех функций (а их более 100) с подробными примерами выходит за рамки этой книги, тем более что справочная система Access и VBA предлагает развернутое изложение использования каждой функции, сопровождаемое примерами. При необходимости использования в дальнейшем изложении какой-либо незнакомой функции мы будем стараться кратко описать ее назначение и применение.

Создание выражений с помощью Построителя выражений

Ввод выражений возможен в среде Access не только вручную, но и с помощью удобного инструмента, называемого Построитель выражений (Expression Builder).

Построитель выражений (Expression Builder) вызывается всякий раз, когда в поле свойства объекта Access, например в ячейке бланка Конструктора запросов, вы щелкаете кнопку Построителя (кнопка с тремя точками) или нажимаете кнопку **Построить** (Build) на панели инструментов.

Для демонстрации работы Построителя выражений воспользуемся демонстрационной базой `Борей` (Northwind). После создания и проверки запроса можно применить критерий отбора, чтобы ограничить количество записей в результирующем множестве запроса:

1. Выберите запрос `Счета` (Invoices) и, нажав кнопку **Конструктор** панели инструментов в окне базы данных, перейдите в режим Конструктора запроса.
2. Поместите курсор на строку **Условие отбора** (Criteria) поля, для которого вы хотите установить критерий выборки, скажем, поля `СтранаПолучателя` (Ship Country).
3. Нажмите кнопку Построителя выражений **Построить** (Build) на панели инструментов Access для вывода окна **Построитель выражений** (Expression Builder). Другой способ — нажмите правую кнопку мыши в строке **Условие отбора** (Criteria) и выберите из меню команду **Построить** (Build).
4. В поле выражений вверху окна Построителя выражений введите требуемое выражение. На рис. 4.40 показан пример выражения `Like Ф*`, которое работает в строке **Условие отбора** (Criteria) поля `СтранаПолучателя` (Ship Country). Для ускорения ввода `Like` можно воспользоваться кнопкой **Like** под окном ввода выражений.

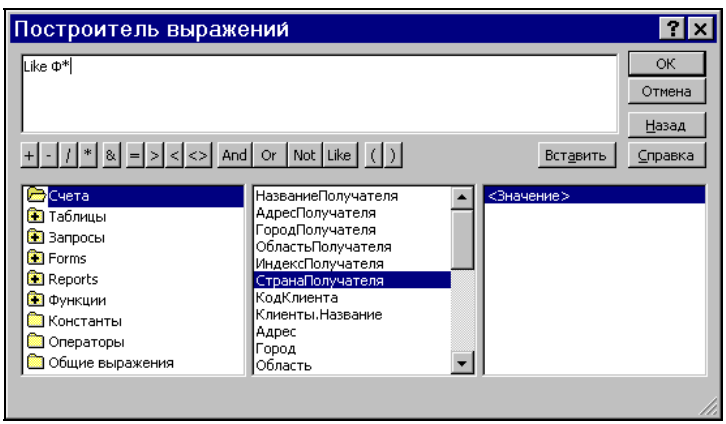


Рис. 4.40. Диалоговое окно **Построитель выражений**

5. Нажмите кнопку **OK** для возврата в бланк запроса. В поле, где расположен курсор, Построитель выражений помещает построенное выражение (рис. 4.41).

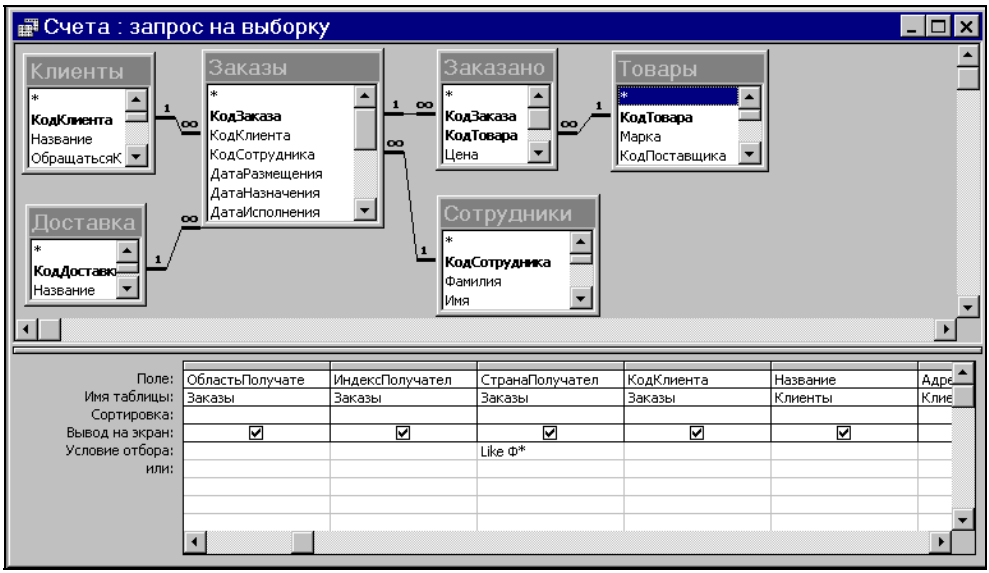


Рис. 4.41. Бланк запроса с выражением, созданным Построителем выражений

6. Для отображения результатов выполненного запроса нажмите кнопку **Запуск** (Open) на панели инструментов (рис. 4.42).

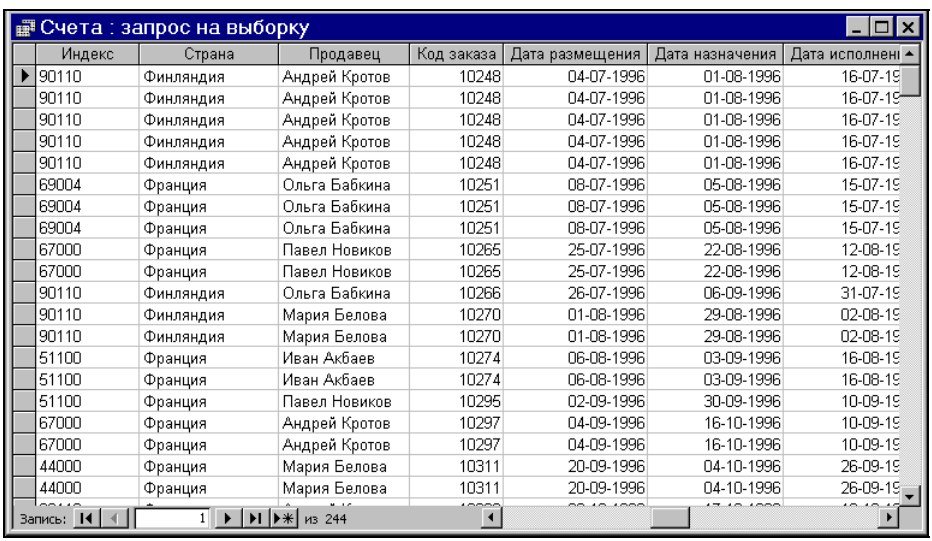


Рис. 4.42. Запрос, созданный в результате добавления условия отбора

7. При желании изменить условие отбора возвратитесь к режиму Конструктор, выделите ненужное выражение и сотрите его нажатием клавиши <Delete>.
8. Повторите шаги со 2 по 7 для заполнения всех условий отбора, которые хотите попробовать. Если вы совершите ошибку в синтаксисе выражений, Access известит о ней сообщением об ошибке. Смелее экспериментируйте!
9. По завершении экспериментов сохраните ваш запрос с описательным именем.

Как видите, создание выражений является не таким уж сложным делом, особенно с таким помощником, как Построитель выражений.

ГЛАВА 5



Создание форм для ввода данных

Для сбора информации самым удобным средством является заполнение специально заготовленной формы — бланка с перечнем вопросов, на которые нужно ответить, пояснениями и пустыми строками или клетками для записи. Такие бумажные формы знакомы всем: их не раз приходилось заполнять при оформлении документов, скажем, в банке, поликлинике или библиотеке. Аналогом таких бумажных форм являются электронные формы в базах данных Access. В Access *формами* называются настраиваемые диалоговые окна, сохраняемые в базе данных в виде объектов специального типа. Формы Access являются объектами базы данных так же, как, например, таблицы и запросы. Формы позволяют организовать удобный и интуитивно понятный интерфейс пользователя для работы с данными. Формы используются в приложении преимущественно для ввода и редактирования данных, иногда их применяют для поиска необходимой информации. Формы предоставляют доступ к данным, содержащимся в одной или сразу в нескольких таблицах. В Access существует возможность управлять формами программно с помощью процедур на языке VBA.

С помощью форм намного проще просматривать и изменять данные, чем при работе с данными в режиме Таблицы. При этом число ошибок ввода существенно сокращается, ведь кроме удобного представления информации формы обеспечивают автоматическую проверку корректности введенных данных. Кроме того, в формах можно отображать результаты вычислений и обеспечить доступ к данным сразу в нескольких связанных таблицах.

На формах располагаются так называемые *элементы управления*, с помощью которых и осуществляется доступ к полям таблиц с данными. Например, элементами управления являются текстовые поля для ввода, кнопки, флажки, переключатели, списки, надписи, а также рамки объектов для отображения графики и объектов OLE.

В этой главе даны основные понятия, касающиеся форм, пройден весь путь от создания простой формы до ее использования. По нашему мнению, этих сведений достаточно, чтобы начать создавать несложные формы в Access. Для получения дополнительной информации, касающейся создания многотабличных и многостраничных форм, обратитесь к *гл. 9*, которая является логическим продолжением настоящей главы.

Автоматическое создание формы на основе таблицы или запроса

В Access 2003 предлагается несколько способов создания форм. Самым простым из них является использование средств автоматического создания форм на основе таблицы или запроса. Автоматически создаваемые формы *(автоформы)* бывают нескольких видов, каждый из которых отличается способом отображения данных:

- Автоформа, организованная **в столбец** (Columnar). В такой форме поля каждой записи отображаются в виде набора элементов управления, расположенных в один или несколько столбцов. Это компактное и, пожалуй, самое удачное представление для быстрого создания формы.
- **Табличная** (Datasheet). Форма будет выглядеть так же, как обычная таблица Access.
- **Ленточная** (Tabular). В такой форме поля каждой записи располагаются в отдельной строке. Это очень удобно для работы с большими массивами данных, поскольку данные располагаются в таком же порядке, как в простой таблице. Преимуществом именно этого способа представления формы по сравнению с табличным является то, что каждое поле представлено в виде отдельного элемента управления, которое можно оформить в любом стиле по вашему вкусу и для которого можно определить функции обработки событий, т. е. "оживить" поле с помощью программирования.
- Автоформа в виде **сводной таблицы** (PivotTable) или **сводной диаграммы** (PivotChart). Эти два варианта появились в предыдущей версии Access 2002. Об этих представлениях речь пойдет в *части II* книги *(см. гл. 9)*.

Автоматически созданная форма включает все поля выбранного источника данных.

Чтобы создать форму с помощью средства автоматического создания форм:

1. Щелкните по ярлыку **Формы** (Forms) в окне **База данных** (Database) и нажмите кнопку **Создать** (New). Появится диалоговое окно **Новая форма** (New Form), представленное на рис. 5.1.

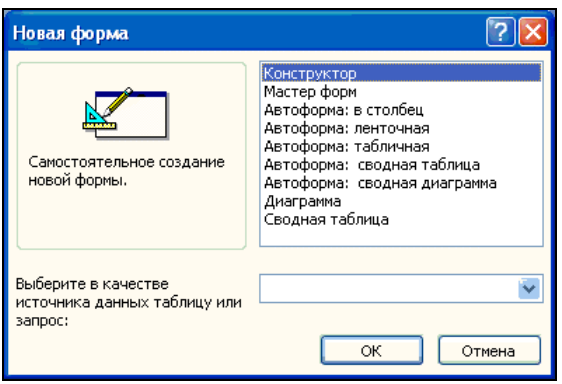


Рис. 5.1. Диалоговое окно **Новая форма**

2. В списке диалогового окна **Новая форма** выделите один из вариантов автоформы, например: **Автоформа: в столбец** (AutoForm: Columnar).
3. В поле со списком, находящемся в нижней части диалогового окна **Новая форма**, содержатся имена всех таблиц и запросов базы данных, которые могут быть использованы в качестве источника данных для формы. Щелкните левой кнопкой мыши по кнопке со стрелкой, чтобы раскрыть список, и выберите в нем нужный элемент.
4. Нажмите кнопку **OK**.

В результате будет автоматически создана и открыта форма выбранного вида. Чтобы созданную форму можно было использовать в дальнейшем, ее необходимо сохранить. Для сохранения формы выберите команду **Файл | Сохранить** (File | Save) или нажмите кнопку **Сохранить** (Save) на панели инструментов **Режим формы** (Form View). В поле **Имя формы** (Form Name) появившегося диалогового окна **Сохранение** (Save As) введите нужное название и нажмите кнопку **OK**.

Совет

Для любой формы можно изменить ее представление, определяющее режим отображения данных. Можно выбрать один из пяти вариантов: **Одиночная форма** (Single Form), **Ленточные формы** (Continuous Form), **Режим таблицы** (Datasheet), **Сводная таблица** (PivotTable) и **Сводная диаграмма** (PivotChart). Чтобы изменить режим отображения данных, откройте форму в режиме Конструктора, затем откройте окно свойств формы *(как это сделать, описано в разд. "Выделение формы и элементов управления, отображение окна свойств" данной главы)* и выберите соответствующий элемент в раскрывающемся списке **Режим по умолчанию** (Default View).

Создание формы с помощью Мастера

Другим простым и быстрым способом создания формы является использование Мастера форм. С помощью Мастера можно создавать формы на основе одной таблицы и более сложные формы на основе нескольких таблиц и запросов, имеющие подчиненные формы. Намного проще и быстрее создавать формы с помощью Мастера, а затем усовершенствовать их в режиме Конструктора. Поэтому Мастер форм полезен не только начинающим пользователям, но и профессиональным разработчикам.

Мастер форм разбивает процесс создания формы на несколько этапов. На каждом этапе требуется установить определенные параметры в одном из диалоговых окон Мастера, которые определяют создание формы по шагам. Если на каком-либо шаге была допущена ошибка или возникла необходимость изменения каких-либо установленных параметров, то для возвращения к предыдущему шагу нажмите кнопку **Назад** (Back). Кроме того, в любой момент можно нажать кнопку **Отмена** (Cancel) для отказа от создания формы и возвращения к окну базы данных.

Чтобы лучше представить, как создавать простые формы с помощью Мастера форм, опишем процедуру создания формы `Сотрудники` (Employees) для базы данных `Борей` (Northwind). Источником данных для этой формы будет таблица `Сотрудники` (Employees). Эта форма предназначена для ввода и редактирования информации

о сотрудниках компании. Чтобы с помощью Мастера создать простую форму, не содержащую подчиненных форм:

1. Щелкните по ярлыку **Формы** (Forms) в окне базы данных.
2. Нажмите кнопку **Создать** (New) на панели инструментов окна базы данных. В появившемся диалоговом окне **Новая форма** (New Form) (см. рис. 5.1) в списке вариантов выделите элемент **Мастер форм** (Form Wizard) и нажмите кнопку **OK**.

Замечание

То же самое можно сделать просто дважды щелкнув по ярлыку **Создание формы с помощью мастера** (Create form by using wizard), находящемуся перед списком существующих форм в базе данных.

3. Появится первое диалоговое окно Мастера форм (рис. 5.2). В поле со списком **Таблицы и запросы** (Tables/Queries), как и в раскрывающемся списке в окне **Новая форма**, отображаются имена всех таблиц и запросов базы данных, которые могут быть использованы в качестве источника данных для формы. Раскройте этот список и выберите имя таблицы или запроса. В нашем примере — это таблица `Сотрудники` (Employees).

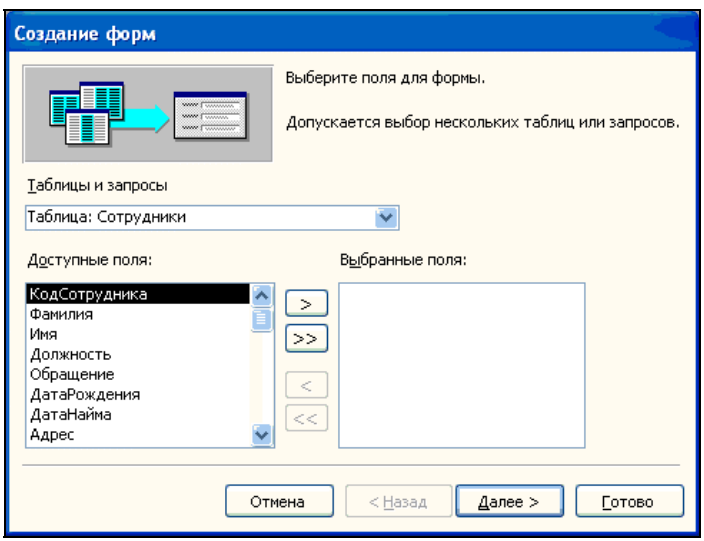


Рис. 5.2. Первое диалоговое окно Мастера форм

4. В списке **Доступные поля** (Available Fields) этого диалогового окна отображаются все поля выбранной таблицы или запроса. Добавление полей в форму позволит просматривать и редактировать данные выбранной таблицы. Чтобы добавить в создаваемую форму только некоторые поля, выделите каждое из этих полей и нажмите кнопку >. Выделенное поле будет перемещено из списка **Доступные поля** в список **Выбранные поля** (Selected Fields). Чтобы добавить в создаваемую форму сразу все поля из выбранной таблицы или запроса, нажмите кнопку >>.

Нажмите кнопку **Далее** (Next) для отображения второго диалогового окна Мастера форм.

5. Во втором диалоговом окне Мастера (рис. 5.3) можно определить вид формы. Как уже говорилось в предыдущем разделе, существует несколько видов форм, определяющих представление данных на ней. Чтобы задать внешний вид формы, выберите один из переключателей: **в один столбец** (Columnar), **ленточный** (Tabular), **табличный** (Datasheet), **выровненный** (Justified), **сводная таблица** (PivotTable) или **сводная диаграмма** (PivotChart).

Замечание

Попробуйте создать ленточную форму и оформить ее подходящим образом с помощью форматирования элементов управления. Вы увидите, что в хорошо оформленной ленточной форме данные просматривать намного удобнее, чем в табличной форме.

Вид **выровненный** аналогичен виду **ленточный**, но отличается тем, что все поля записи будут отображены на одной странице.

После выбора подходящего режима отображения данных в форме нажмите кнопку **Далее** для отображения следующего диалогового окна Мастера форм.

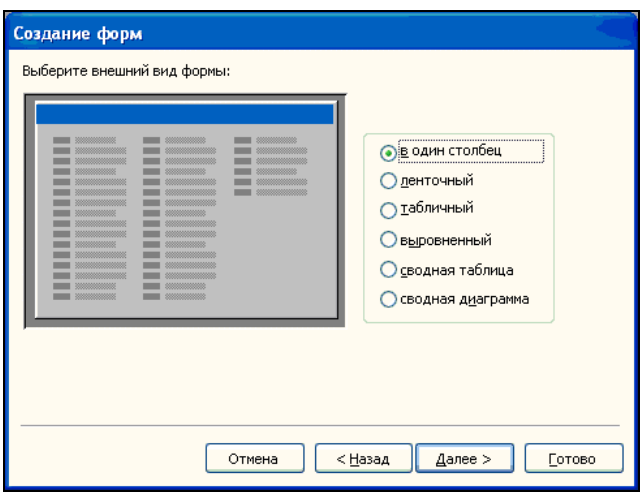


Рис. 5.3. Второе диалоговое окно Мастера форм

6. Третье диалоговое окно Мастера форм (рис. 5.4) предназначено для выбора стиля оформления новой формы. Мастер предлагает несколько стандартных стилей оформления. Можно определить собственные стили с помощью диалогового окна **Автоформат** (Autoformat). Тогда они будут отображены вместе со стандартными в этом диалоговом окне Мастера форм. Выберите один из предлагаемых стилей и нажмите кнопку **Далее**. *(Способ создания собственного стиля описан в разд. "Применение автоформата" этой главы.)*

7. В последнем диалоговом окне Мастера форм (рис. 5.5) требуется указать название формы. В поле ввода этого диалогового окна введите название формы:

Форма_Сотрудники. Чтобы отобразить созданную Мастером форму в режиме Формы, выберите переключатель **Открыть форму для просмотра и ввода данных** (Open the form to view or enter information). А если после автоматического создания формы с помощью Мастера требуется внести собственные изменения, выберите переключатель **Изменить макет формы** (Modify the form's design), тогда созданная форма будет открыта в режиме Конструктора. Если необходима справка о работе с созданной формой, установите флажок **Вывести справку по работе с формой?** (Display Help on working with the form?). После этого нажмите кнопку **Готово** (Finish).

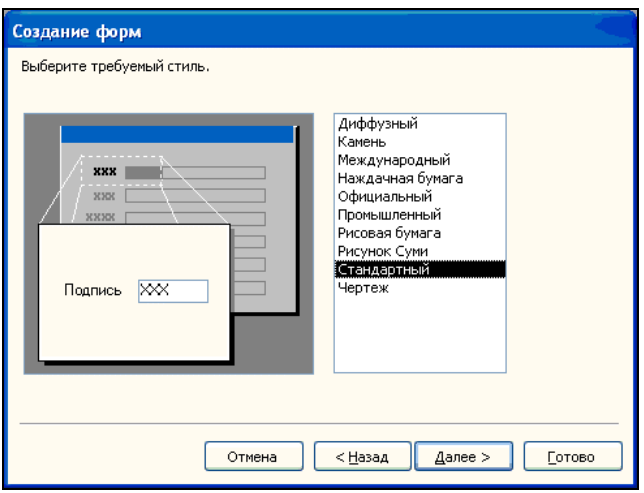


Рис. 5.4. Третье диалоговое окно Мастера форм

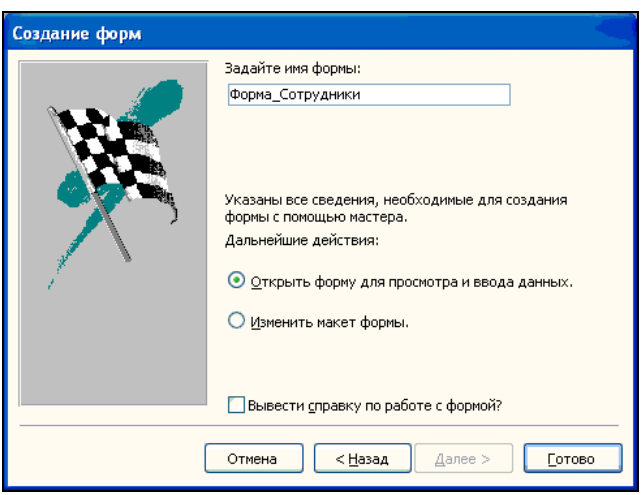


Рис. 5.5. Последнее диалоговое окно Мастера форм

В результате Мастером будет создана форма в соответствии с выбранными параметрами и сохранена с указанным именем, затем эта форма будет открыта в заданном режиме. На рис. 5.6 представлена созданная форма, открытая в режиме Формы.

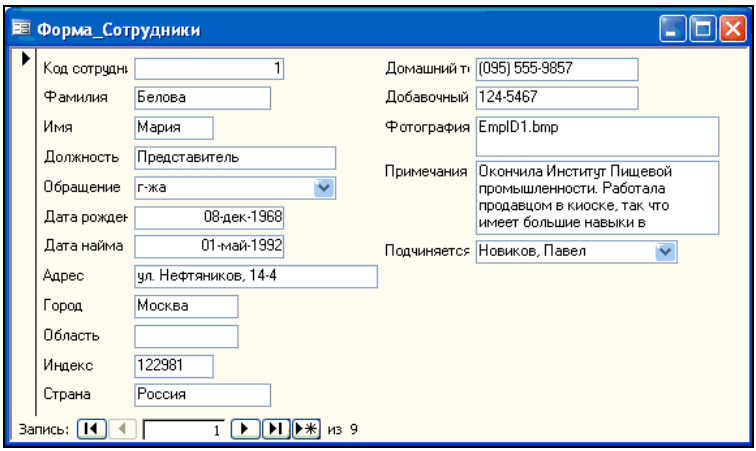


Рис. 5.6. Созданная с помощью Мастера `Форма_Сотрудники`

Форма `Форма_Сотрудники`, созданная с помощью Мастера, готова к использованию. Однако можно улучшить ее внешний вид, изменив некоторые параметры оформления. В следующих разделах приведены примеры изменения внешнего вида формы в режиме Конструктора.

Режимы работы с формами

Работа с формами Access 2003 может происходить в пяти режимах: Формы, Таблицы, Конструктора, Сводной таблицы и Сводной диаграммы. Выбрать режим работы с открытой формой можно либо с помощью кнопки **Вид** (View) на панели инструментов текущего режима работы с формой (например, **Конструктор форм** (Form Design) — одна из таких панелей), либо с помощью соответствующей команды меню **Вид** (View).

Режим Формы является "рабочим" для пользователя базы данных. В этом режиме осуществляется просмотр и редактирование записей, удаление записей или добавление новых. Таким способом пользователь работает с таблицей или запросом, являющимся источником данных для формы. Чаще всего пользователи приложений Access работают с формами именно в этом режиме. Здесь по умолчанию открывается форма из окна базы данных. Если форма была открыта в другом режиме, то для перехода в режим Формы выберите команду **Вид | Режим формы** (View | Form View) или нажмите на стрелку, расположенную справа от кнопки **Вид** (View) на панели инструментов, и в открывшемся списке выберите элемент **Режим формы** (Form View). На рис. 5.6 показана форма, создание которой описано в предыдущем разделе, открытая в режиме Формы.

Для пользователя может оказаться удобным работать с формой в режиме Таблицы. В этом режиме, как и в режиме Формы, можно просматривать и редактировать, до-

бавлять и удалять записи в таблице или запросе, являющемся источником данных для формы. Однако здесь не применяются параметры форматирования элементов управления. Чтобы перейти в режим Таблицы, выберите команду **Вид | Режим таблицы** (View | Datasheet View) или нажмите на стрелку, расположенную справа от кнопки **Вид** (View) панели инструментов, и в открывшемся списке выберите элемент **Режим таблицы** (Datasheet View). На рис. 5.7 показана форма, создание которой описано в предыдущем разделе, открытая в режиме Таблицы.

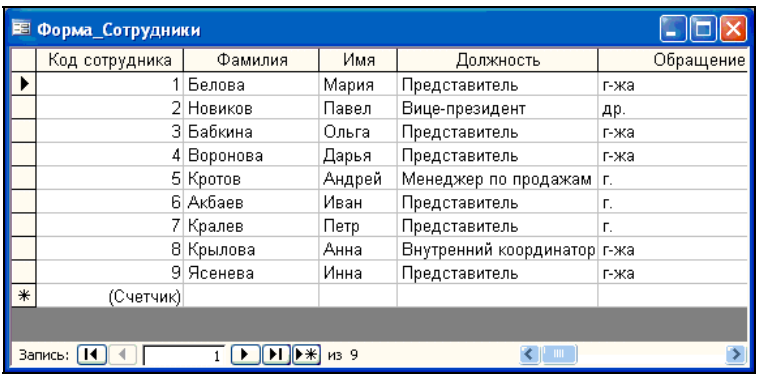


Рис. 5.7. Форма `Форма_Сотрудники` в режиме Таблицы

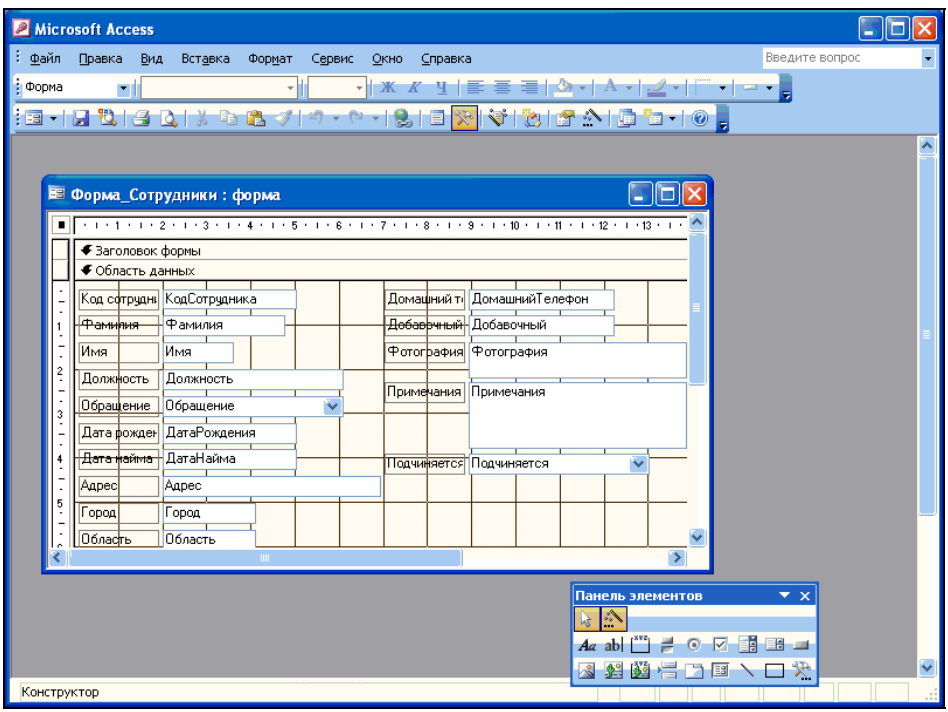


Рис. 5.8. Форма `Форма_Сотрудники` в режиме Конструктора

В любое время можно настроить различные свойства формы, изменить ее внешний вид, структуру и функциональность, работая с ней в режиме Конструктора (им обычно пользуются разработчики). Чтобы перейти в режим Конструктора, выберите команду **Вид | Конструктор** (View | Design View) или нажмите на стрелку, расположенную справа от кнопки **Вид** (View) на панели инструментов, и в открывшемся списке выберите элемент **Конструктор** (Design View). На рис. 5.8 показана форма, создание которой описано в предыдущем разделе, открытая в режиме Конструктора.

Начиная с предыдущей версии Access 2002, разработчики и пользователи Access могут работать с формой еще в двух режимах: Сводной таблицы или Сводной диаграммы. Но эти режимы мы рассмотрим в *части II* книги.

Печать формы

Форма, как таблица, запрос или отчет базы данных Access, может служить документом для печати. Напечатать форму можно, находясь в любом режиме работы с ней, например, в режиме Таблицы или режиме Формы. При этом внешний вид печатаемой формы будет соответствовать текущему режиму. Команды печати и способ настройки параметров печати точно такие же, как при печати таблиц и отчетов Access, или, например, документов Word. Это стандартные команды меню **Файл** (File): **Параметры страницы** (Page Setup), **Предварительный просмотр** (Print Preview) и **Печать** (Print), а также кнопки на панели инструментов текущего режима работы с формой и панели инструментов **Предварительный просмотр** (Print Preview). Печать формы, как и других объектов базы данных, можно выполнить и программным путем с помощью процедуры на языке VBA или макроса (о программировании речь пойдет в *части II* книги).

Чтобы увидеть, как форма будет выглядеть при печати, следует перейти в режим Предварительного просмотра. Для этого выберите команду **Файл | Предварительный просмотр** (File | Print Preview) или нажмите кнопку **Предварительный просмотр** (Print Preview) на панели инструментов. Чтобы вернуться в прежний режим работы с формой, снова выберите команду меню **Файл | Предварительный просмотр** или нажмите кнопку **Закрыть** (Close) на панели инструментов. А также можно нажать на стрелку справа от кнопки **Вид** (View) на панели инструментов и выбрать один из режимов для дальнейшей работы с формой.

Разработка формы в режиме Конструктора

Разработку структуры формы, того, как она будет выглядеть в режиме Таблицы и режиме Формы, можно выполнить только в *режиме Конструктора*. Здесь можно изменить источник данных для формы, количество отображаемых полей, внешний вид формы и элементов управления, добавить или удалить элементы управления, настроить их свойства.

Панель элементов (Toolbox) (см. рис. 5.8), отображаемая в режиме Конструктора, позволяет создавать элементы управления, при этом некоторые элементы управления можно создавать с помощью Мастера элементов, осуществляющего пошаговое руководство процессом создания.

К форме можно применить один из предлагаемых Access стилей оформления, позволяющих быстро и качественно настроить ее внешний вид. Более того, Access позволяет создавать свои собственные стили оформления и применять их к другим формам. Все это позволяет сделать диалоговое окно **Автоформат** (AutoFormat), которое описано в *разд. "Применение автоформата" данной главы.*

Чтобы создать пустую форму, не пользуясь автоматическим созданием форм и Мастером форм, в окне базы данных щелкните по ярлыку **Формы** (Forms), нажмите кнопку **Создать** (New) на панели в окне базы данных, в появившемся диалоговом окне **Новая форма** (New Form) выделите элемент **Конструктор** (Design View) и нажмите кнопку **ОК**. То же самое можно сделать просто дважды щелкнув по ярлыку **Создание формы в режиме конструктора** (Create form in Design view), находящемуся перед списком существующих форм в базе данных.

Можно использовать созданную ранее форму `Форма_Сотрудники` для проведения экспериментов, связанных с изменением ее внешнего вида. Однако не стоит сохранять измененную форму, чтобы не испортить ее первоначальный вид. Или прежде чем проводить эксперименты, создайте копию исходной формы.

Окно приложения в режиме Конструктора

Откройте форму `Форма_Сотрудники` в режиме Конструктора (см. рис. 5.8), для чего выделите эту форму в окне базы данных и нажмите кнопку **Конструктор** (Design).

При работе в режиме Конструктора форм используются следующие элементы окна приложения:

- **Панель элементов** (Toolbox). При переходе в режим Конструктора на экране отображается данная панель инструментов, предназначенная для добавления в форму новых элементов управления. Использование панели элементов описано далее в этой главе. Здесь мы обсудим другие особенности работы с формой в режиме Конструктора, а панель элементов можно закрыть. Для этого нажмите кнопку **Панель элементов** (Toolbox) на панели инструментов **Конструктор форм** (Form Design) или кнопку **Закрыть** (Close) — кнопка с изображением крестика в правом верхнем углу панели инструментов.

Замечание

Панель элементов появляется на экране всякий раз при переходе в режим Конструктора. Если закрыть эту панель инструментов, в следующий раз она не появится. Тогда, чтобы отобразить ее, нажмите кнопку **Панель элементов** на панели инструментов **Конструктор форм** или выберите команду **Вид | Панели инструментов** (View | Toolbars), а затем установите флажок **Панель элементов** (Toolbox).

- Панель инструментов **Конструктор форм** (Form Design). Основные команды, меню, используемые в режиме Конструктора форм, продублированы в виде кнопок на панели инструментов **Конструктор форм**. *(Назначение кнопок этой панели инструментов подробно описано в разд. "Панель инструментов Конструктор форм" данной главы.)*
- Панель инструментов **Формат (форма/отчет)** (Formatting (Form/Report)) предназначена для форматирования выделенных элементов управления. Форматирование элемента управления включает такие действия, как задание цвета фона и ра-

мок, задание стиля и цвета текста. *(Назначение кнопок этой панели инструментов описано в разд. "Панель инструментов Формат (форма/отчет)" данной главы.)*

- Вертикальная и горизонтальная линейки предназначены для разметки формы по заданным размерам и выделения нескольких соседних элементов управления, лежащих в одном вертикальном или горизонтальном сегменте макета формы от одной отметки на линейке до другой.
- Правая граница макета формы позволяет задать ширину формы. Чтобы изменить ширину формы, перетащите правую границу макета формы с помощью мыши вправо или влево. На рис. 5.8 правая граница макета формы находится слева от панели инструментов **Панель элементов**.
- Нижняя граница макета формы позволяет определить высоту формы. Чтобы изменить высоту формы, перетащите нижнюю границу макета формы с помощью мыши вниз или вверх. На рис. 5.8 нижняя граница макета формы совпадает с нижней границей примечания формы.
- Вертикальная и горизонтальная полосы прокрутки позволяют просматривать части формы, оказавшиеся за границами экрана в режиме Конструктора.

Структура формы

Макет формы состоит из разделов. Любая форма может включать следующие разделы:

- **Заголовок формы** (Form Header) определяет верхнюю часть формы. Этот раздел добавляется в форму вместе с разделом примечания формы. В область заголовка формы можно поместить текст, графику и другие элементы управления. При печати многостраничной формы раздел заголовка отображается только на первой странице.
- **Верхний колонтитул** (Page Header) определяет верхний колонтитул страницы при печати формы. Этот раздел добавляется в форму вместе с разделом, определяющим нижний колонтитул страницы. Данный раздел отображается только, когда форма открыта в режиме Предварительного просмотра. При печати многостраничной форме верхний колонтитул отображается вверху каждой страницы.
- **Область данных** (Detail) определяет основную часть формы, содержащую данные, полученные из источника. Этот раздел может содержать элементы управления, отображающие данные из таблиц и запросов, а также неизменяемые данные, например надписи. При печати многостраничной формы данный раздел отображается на каждой странице.
- **Нижний колонтитул** (Page Footer) определяет нижний колонтитул страницы при печати формы. Этот раздел добавляется в форму вместе с разделом, определяющим верхний колонтитул страницы. Он отображается только, когда форма открыта в режиме Предварительного просмотра. При печати многостраничной формы нижний колонтитул отображается внизу каждой страницы.
- **Примечание формы** (Form Footer) определяет нижнюю часть формы. Этот раздел добавляется в форму вместе с разделом заголовка формы. При печати многостраничной формы примечание формы будет отображено только внизу последней страницы.

Разделы формы в режиме Конструктора представлены на рис. 5.9. Чтобы надписи, обозначающие разделы формы, были лучше видны, мы скрыли сетку, облегчающую разметку формы (сетка видна на рис. 5.8), с помощью команды меню **Вид | Сетка** (View | Grid).

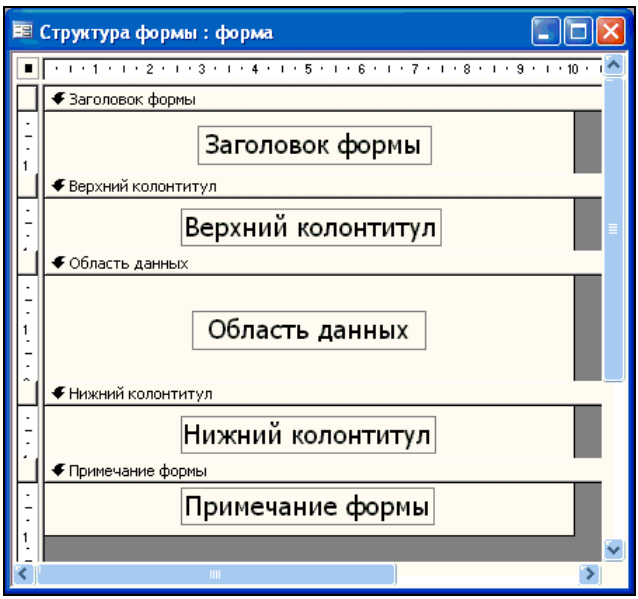


Рис. 5.9. Разделы формы Access

Из всех перечисленных разделов обязательным является только раздел области данных. Высоту области любого раздела формы можно изменить с помощью мыши, перетаскивая нижнюю часть раздела вверх или вниз.

Замечание

Хотя форма, изображенная на рис. 5.8, имеет оба раздела заголовка и примечания формы, эти области не отображаются в режиме Формы, поскольку оба эти раздела имеют нулевую высоту. При создании новой пустой формы при помощи Конструктора форм разделы заголовка и примечания не добавляются.

Для добавления или удаления разделов заголовка и примечания формы выберите команду **Вид | Заголовок/примечание формы** (View | Form Header/Footer). Если форма уже содержит эти разделы, то команда **Заголовок/примечание формы** будет отмечена флажком. Для удаления разделов заголовка и примечания формы сбросьте этот флажок.

Для добавления или удаления разделов верхнего и нижнего колонтитула страницы выберите команду **Вид | Колонтитулы** (View | Page Header/Footer). Если форма уже содержит эти разделы, то команда **Колонтитулы** будет отмечена флажком. Для удаления разделов верхнего и нижнего колонтитула страницы сбросьте этот флажок.

Замечание

Если разделы заголовка и примечания формы или разделы верхнего и нижнего колонтитула содержат текст либо другие элементы управления формы, то перед попыткой удалить эти разделы будет отображено диалоговое окно, предупреждающее об удалении содержимого заголовка и примечания.

Высоту раздела формы можно изменить, перетаскивая границу раздела при помощи мыши. Для этого поместите указатель мыши над границей раздела так, чтобы он принял вид горизонтальной черты с двумя вертикальными стрелками. Нажмите левую кнопку мыши и перетащите границу раздела для получения нужного размера.

Высота раздела области данных определяется вертикальным размером окна формы, а также размером заголовка и примечания формы. Для изменения высоты раздела области данных переместите его нижнюю границу при помощи мыши вверх (для уменьшения высоты) или вниз (для увеличения). В режиме Формы вертикальная полоса прокрутки в окне формы появляется только для просмотра раздела области данных.

Панель инструментов *Конструктор форм*

Панель инструментов **Конструктор форм** (Form Design) в Access 2003 содержит кнопки, предназначенные для разработки форм. На рис. 5.10 изображена панель инструментов **Конструктор форм**, а в табл. 5.1 приведены описания кнопок этой панели инструментов и соответствующие им команды меню.



Рис. 5.10. Панель инструментов **Конструктор форм**

Таблица 5.1. *Панель инструментов* ***Конструктор форм***

Кнопка	Описание	Команда меню
Вид (View) и элементы списка: **Режим формы** (Form View), **Режим таблицы** (Datasheet View), **Сводная таблица** (Pivot Table View), **Сводная диаграмма** (Pivot Chart View), **Конструктор** (Design View)	Отображение формы в выбранном режиме. Например, чтобы открыть форму в режиме Таблицы, нажмите стрелку справа от кнопки и в появившемся списке выделите элемент **Режим таблицы** (Datasheet View)	**Вид** | **Режим формы** (View | Form View), **Вид** | **Режим таблицы** (View | Datasheet View), **Вид** | **Сводная таблица** (View | Pivot Table View), **Вид** | **Сводная диаграмма** (View | Pivot Chart View), **Вид** | **Конструктор** (View | Design View)
Сохранить (Save)	Сохранение активной формы	**Файл** | **Сохранить** (File | Save)

Таблица 5.1 *(продолжение)*

Кнопка	Описание	Команда меню
Поиск файлов (Search)	Отобразить панель поиска документов Microsoft Office	**Файл** | **Поиск файлов** (File | Search)
Печать (факс) (Print (Fax))	Печать или отправка по факсу содержимого активной формы с текущими параметрами печати без открытия диалогового окна **Печать** (Print)	
Предварительный просмотр (Print Preview)	Предварительный просмотр формы перед печатью. Форма может быть напечатана из окна предварительного просмотра	**Файл** | **Предварительный просмотр** (File | Print Preview)
Вырезать (Cut)	Удаление выделенных объектов из формы в Буфер обмена Windows	**Правка** | **Вырезать** (Edit | Cut)
Копировать (Copy)	Копирование выделенных объектов формы в Буфер обмена	**Правка** | **Копировать** (Edit | Copy)
Вставить (Paste)	Вставка содержимого Буфера обмена в форму	**Правка** | **Вставить** (Edit | Paste)
Формат по образцу (Format Painter)	Копирование параметров форматирования из одного выделенного объекта в другой такого же типа	
Отменить (Undo)	Отмена последнего действия	**Правка** | **Отменить** (Edit | Undo)
Вернуть (Redo)	Повторить выполнение последнего отмененного действия	**Правка** | **Вернуть** (Edit | Redo)
Добавить гиперссылку (Insert Hyperlink)	Вставка в форму гиперссылки	**Вставка** | **Гиперссылка** (Insert | Hyperlink)
Список полей (Field List)	Вывод списка полей, содержащихся в базовом источнике записей (запросе или таблице) основной формы	**Вид** | **Список полей** (View | Field List)
Панель элементов (Toolbox)	Отображение или скрытие панели элементов	**Вид** | **Панели инструментов** | **Панель элементов** (View | Toolbars | Toolbox)
Автоформат (AutoFormat)	Выбор заранее определенных форматов для форм, включая такие характеристики, как фоновый рисунок для форм и шрифты для элементов управления	**Формат** | **Автоформат** (Format | AutoFormat)
Программа (Code)	Открытие модуля процедур обработки событий формы в окне редактора VBA	**Вид** | **Программа** (View | Code)

Таблица 5.1 *(окончание)*

Кнопка	Описание	Команда меню
Свойства (Properties)	Открытие окон свойств выделенных объектов: формы, раздела (например, верхнего или нижнего колонтитула), а также элементов управления	**Вид** | **Свойства** (View | Properties)
Построить (Build)	Вызов Построителя для выделенного элемента или свойства в форме, если Построитель есть	
Окно базы данных (Database Window)	Отображение окна базы данных	**Окно** | **1** (Window | 1)
Новый объект (New Object)	Создание нового объекта базы данных. Чтобы выбрать тип создаваемого объекта, нажмите стрелку справа от кнопки	
Справка: Microsoft Office Access (Microsoft Access Help)	Получение справки по Microsoft Access 2003	**Справка** | **Справка: Microsoft Office Access** (Help | Microsoft Access Help)

Панель инструментов *Формат (форма/отчет)*

Кнопки и раскрывающиеся списки, облегчающие выбор параметров форматирования (цвета, шрифта, толщины линий и т. п.), расположены на отдельной панели инструментов, которая называется **Формат (форма/отчет)** (Formatting (Form/Report)). Эта панель инструментов применяется для форматирования элементов форм и отчетов.

Функции кнопок, используемых для форматирования текста, аналогичны функциям кнопок, которые можно увидеть на панелях инструментов Microsoft Word. Раскрывающийся список **Объект** (Object) отображает имя выделенного элемента управления формы и позволяет быстро выделить другой объект в форме, выбрав его имя в списке. По умолчанию, для элементов управления форм используется такой же шрифт, как для таблиц и запросов. В раскрывающемся списке **Шрифт** (Font) можно выбрать любой из установленных на компьютере шрифтов. В раскрывающемся списке **Размер** (Font Size) можно выбрать нужный размер шрифта или ввести его значение (в пунктах).

На рис. 5.11 изображена панель форматирования, а в табл. 5.2 приведены названия и функции кнопок, предназначенных для форматирования текста элементов управления, а также значение соответствующих свойств элемента управления.

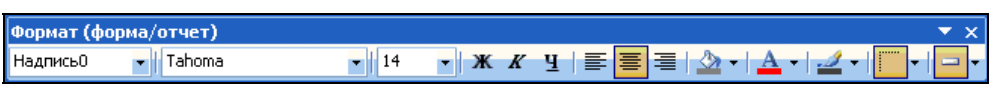


Рис. 5.11. Панель инструментов **Формат (форма/отчет)**

Таблица 5.2. *Кнопки панели инструментов* ***Формат (форма/отчет)***

Кнопка	Описание	Свойство и значение
Объект (Object)	Имя выделенного объекта	Имя (Name) = <*имя объекта*>
Шрифт (Font)	Шрифт текста в элементе	Шрифт (Font Name) = <*имя шрифта*>
Размер (Font Size)	Размер шрифта текста в элементе	Размер шрифта (Font Size) = <*число*>
Полужирный (Bold)	Форматирование текста выделенного элемента управления полужирным шрифтом	Насыщенность (Font Weight) = Полужирный (Semi-bold)
Курсив (Italic)	Форматирование текста выделенного элемента управления курсивом	Курсив (Font Italic) = Да (Yes)
Подчеркнутый (Underline)	Делает текст элемента управления подчеркнутым	Подчеркнутый (Font Underline) = Да (Yes)
По левому краю (Align Left)	Выравнивание текста по левому краю	Выравнивание текста (Text Align) = По левому краю (Left)
По центру (Center)	Выравнивание текста по центру	Выравнивание текста (Text Align) = По центру (Center)
По правому краю (Align Right)	Выравнивание текста по правому краю	Выравнивание текста (Text Align) = По правому краю (Right)
Цвет заливки/фона (Fill/Back Color)	Открывает палитру, содержащую образцы цветов для фона выделенного объекта или раздела формы	Цвет фона (Back Color) = <*номер*>
Цвет текста (Font/Fore Color)	Открывает палитру, содержащую образцы цветов текста выделенного элемента управления формы	Цвет текста (Fore Color) = <*номер*>
Цвет линии/границы (Line/Border Color)	Открывает палитру, содержащую образцы цветов границы выделенного элемента управления формы	Цвет границы (Border Color) = <*номер*>
Толщина линии/границы (Line/Border Width)	Открывает палитру образцов ширины границы для выделенного элемента управления. Ширина границы может быть выбрана сверхтонкой или толщиной от 1 до 6 пунктов	Ширина границы (Border Width) = <*ширина*>

***Таблица 5.2** (окончание)*

Кнопка	Описание	Свойство и значение
Оформление (Special Effect)	Открывает панель стилей оформления элементов управления, предназначенных для оформления выделенного объекта. Можно выбрать обычное представление элемента управления, с тенью, приподнятое, утопленное, вдавленное или рельефное	`Оформление (Special Effect) = <оформление>`

Параметры работы с формами

При создании новых форм используются параметры, заданные с помощью диалогового окна **Параметры** (Options). Для отображения этого окна выберите команду **Сервис | Параметры** (Tools | Options), в появившемся диалоговом окне раскройте вкладку **Формы и отчеты** (Forms/Reports) (рис. 5.12). С помощью этой вкладки можно задать имя существующей формы (в текстовом поле **Шаблон формы** (Form template)), которая будет использована в качестве образца оформления новых форм, исключая формы, созданные с помощью Мастера. Можно также задать способ выделения объектов в форме или отчете, выбрав соответствующий элемент в группе **Выделение объектов** (Selection behavior), и еще назначить использование по умолчанию процедур VBA для обработки событий форм, элементов управления или отчетов, установив флажок **Всегда использовать процедуры обработки событий** (Always use event procedure).

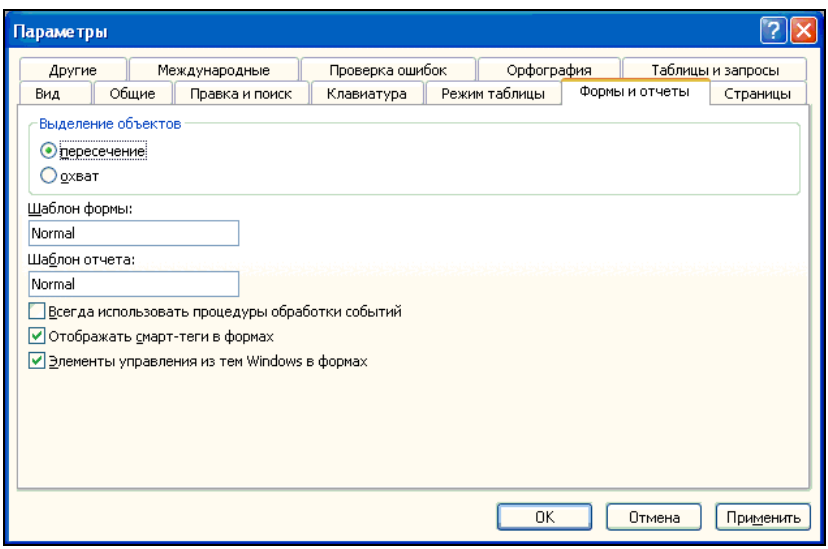


Рис. 5.12. Вкладка **Формы и отчеты** диалогового окна **Параметры**

Обратите внимание в этом окне на два флажка: **Отображать смарт-теги в формах** (Show Smart Tags on Forms) и **Элементы управления из тем Windows в формах** (Use Windows Themed Controls on Forms). Эти два параметра являются новыми для Access, они появились в версии Access 2003.

Первый из этих параметров позволяет отключить возможность отображения смарт-тегов в формах. С одной стороны это может увеличить скорость загрузки формы и редактирования данных, с другой стороны, сбросив этот флажок, вы лишите приложение Access 2003 дополнительных возможностей, предоставляемых присоединенными к элементам управления смарт-тегами. Аналогичный параметр добавлен и на вкладку **Режим таблицы** (Datasheet View) этого окна для возможности включения или отключения смарт-тегов в режиме Таблицы.

Второй новый параметр (**Элементы управления из тем Windows в формах**) предназначен для управления внешним видом форм. Дело в том, что в Access 2003 многие элементы управления форм способны наследовать графические свойства, соответствующие текущему стилю оформления (теме) операционной системы Windows XP (если установлен пакет обновлений Service Pack 1). Если вы хотите избежать лишних графических украшений в элементах управления, снимите этот флажок. А если захотите вообще вернуться к классическому стилю оформления Windows, то с помощью Панели управления (Control Panel) откройте окно **Отображение** (Display Properties) и на вкладке **Темы** (Themes) в раскрывающемся списке **Тема** (Theme) выберите элемент **Windows Classic**. Тогда и формы Access 2003 будут выглядеть соответственно.

Основные элементы управления

Здесь мы рассмотрим наиболее простые и часто используемые элементы управления: текстовое поле, надпись, поле со списком, переключатель, выключатель и флажок. Формы с вкладками и многостраничные формы будут подробно рассмотрены в *гл. 9*.

Чтобы сделать форму более понятной и легко читаемой, на нее добавляют надписи, поясняющие содержимое формы. Этот текст является неизменяемым и создается с помощью элемента управления **Надпись** (Label).

Для отображения, ввода или изменения в форме текстовых данных, например примечаний, используются текстовые поля, которые создаются с помощью элемента управления **Поле** (Text Box).

Текстовые поля позволяют вводить произвольные значения (хотя можно ограничить множество допустимых значений для поля, задав с помощью окна свойств текстового поля, например, маску ввода или правило проверки введенного значения). Чтобы предоставить пользователю выбор из определенного набора значений, вместо текстовых полей применяют такие элементы управления, как флажки, переключатели, выключатели, объединенные в группы однотипных элементов, а также списки.

Группа (рис. 5.13) тоже представляет собой элемент управления, который имеет название **Группа переключателей** (Option group). Каждому элементу управления в группе сопоставляется уникальное число, однозначно определяющее данный элемент в группе, это число и сохраняется в качестве значения свойства **Значение** (Option Value) элемента управления. Отдельный элемент управления **Флажок** (Check box),

Переключатель (Option button) или **Выключатель** (Toggle button), не включенный в группу подобных элементов, определяет значение логического типа: **Да** (Yes) или **Нет** (No). В данном случае вместо значения **Да** (Yes) допускается использовать значение **Истина** (True) или любое положительное число, а вместо значения **Нет** (No) — значение **Ложь** (False) или ноль. Элементы управления **Флажок** (Check box), **Переключатель** (Option button), **Выключатель** (Toggle button) и **Группа переключателей** (Option group) имеют свойство **Значение по умолчанию** (Default Value), позволяющее задать первоначальное состояние элемента управления или группы. Для отдельного элемента управления в качестве значения этого свойства используется значение логического типа, а для группы — число, сопоставленное одному из элементов группы (тому элементу, который будет выбран в группе по умолчанию).

Кроме групп элементов для предоставления выбора значения какого-либо параметра из заданного набора можно использовать элемент управления **Список** (List Box) или **Поле со списком** (Combo Box). Ввести значение в поле со списком можно двумя способами: ввести значение в поле или выбрать значение в раскрывающемся списке.

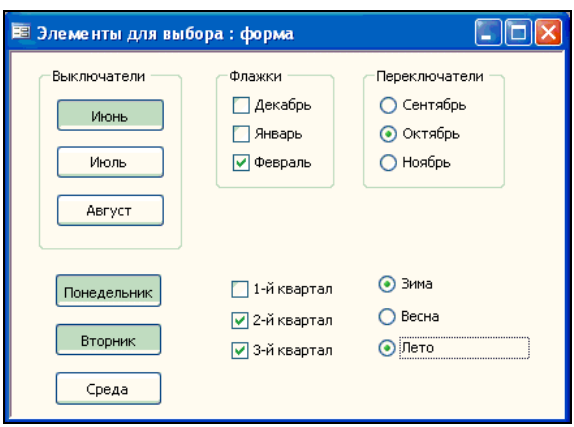


Рис. 5.13. Отдельные флажки, переключатели, выключатели и группы элементов

Совет

Пользователю намного удобнее работать со списком или группой элементов, чем с текстовым полем, когда требуется выбрать одно из нескольких возможных значений. Во-первых, чтобы сделать выбор, пользователю не нужно вводить значение полностью, а достаточно лишь щелкнуть кнопкой мыши, чтобы выделить нужный элемент. Во-вторых, нет необходимости помнить все допустимые при выборе значения, поскольку они явно перечислены в элементах группы или списка. Поэтому, чтобы сделать интерфейс приложения более удобным для пользователя, используйте в формах группы или списки вместо текстовых полей всегда, когда нужно обеспечить выбор одного из нескольких значений. А также применяйте отдельные флажки, выключатели или переключатели вместо текстовых полей, когда требуется предоставить пользователю выбор между альтернативами "да" — "нет" для заданных параметров.

Создание и удаление элементов управления

Access предоставляет два простых метода создания элементов управления: посредством панели инструментов **Панель элементов** (Toolbox) и кнопки **Список полей** (Field List). Кроме того, при создании элементов управления с помощью панели элементов можно воспользоваться Мастером элементов, позволяющим пройти шаг за шагом процесс создания более сложных элементов управления, например, таких как списки, поля со списком и группы элементов управления.

Создание элементов управления с помощью Панели элементов

Надпись — самый простой для использования тип элементов управления панели элементов. Для добавления надписи в какой-либо раздел формы:

1. На панели элементов нажмите кнопку **Надпись** (Label). Когда указатель мыши попадет в область формы, он примет вид крестика со значком элемента управления **Надпись** (Label). Центр крестика определяет позицию верхнего левого угла элемента управления **Надпись** (Label).
2. Поместите указатель мыши в виде крестика в область нужного раздела формы. Нажмите левую кнопку мыши и, удерживая ее, перетащите указатель мыши в правый нижний угол надписи (рис. 5.14).

Замечание

Вместе с перемещением указателя мыши будет изменяться и контур надписи. Число строк и количество символов текущего типа шрифта, которые может отобразить надпись, выводятся в строке состояния.

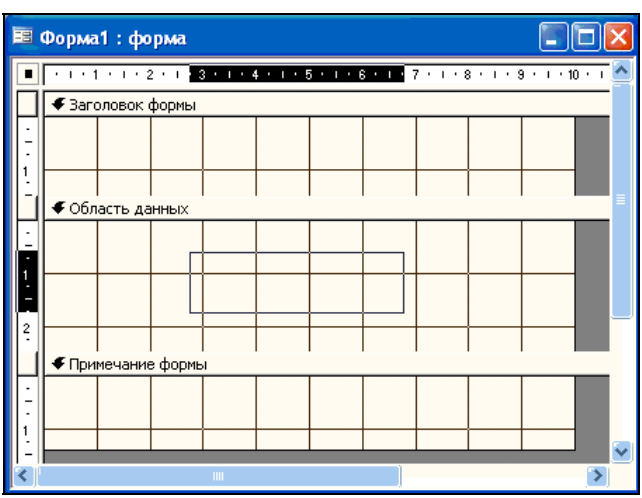


Рис. 5.14. Добавление в форму элемента управления **Надпись**

3. При достижении элементом управления **Надпись** (Label) нужных размеров отпустите левую кнопку мыши. Если надпись будет иметь размеры, превышающие об-

ласть того раздела, к которому она добавляется, то область раздела формы будет увеличена, чтобы поместить надпись целиком.

4. Внутри контура надписи появится текстовый курсор. Введите нужный текст. Если, создав надпись, не ввести в нее хотя бы одного символа, то после следующего щелчка кнопкой мыши этот элемент управления исчезнет.

После добавления элемента управления для его перемещения и изменения размеров используется угловой маркер перемещения и маркеры изменения размеров. С помощью маркера перемещения можно изменить положение левого верхнего угла элемента. Положение левого верхнего угла элемента определяют также значения свойств элемента управления **От левого края** (Left) и **От верхнего края** (Top). Маркеры изменения размеров устанавливают значения свойств элемента управления **Ширина** (Width) и **Высота** (Height).

Добавление в форму большинства других типов элементов управления (выключателей, переключателей, флажков) аналогично, описанному ранее. Некоторые кнопки панели элементов, например **Список** (List Box) или **Поле со списком** (Combo Box), при создании элемента управления вызывают Мастер элементов, если кнопка **Мастера** (Control Wizards) нажата.

Чтобы создать элемент управления **Список** с помощью Мастера:

1. На панели элементов нажмите кнопку **Мастера** (Control Wizards), если она еще не нажата.
2. На панели элементов нажмите кнопку **Список** (List Box). Когда указатель мыши попадет в область формы, он примет вид крестика со значком элемента управления **Список** (List Box). Центр крестика определяет позицию верхнего левого угла элемента управления **Список** (List Box).
3. Поместите указатель мыши в виде крестика в область нужного раздела формы. Нажмите левую кнопку мыши и, удерживая ее, перетащите указатель мыши в правый нижний угол списка.

Замечание

Вместе с перемещением указателя мыши будет изменяться и контур списка. Число строк и количество символов текущего типа шрифта, которые может отобразить список, выводятся в строке состояния.

4. При достижении элементом управления **Список** (List Box) нужных размеров отпустите левую кнопку мыши. Появится первое диалоговое окно Мастера элементов. В этом диалоговом окне выберите переключатель **Объект "список" будет использовать значения из таблицы или запроса** (I want the list box to look up the values in a table or query). Нажмите кнопку **Далее** (Next). Появится второе диалоговое окно Мастера списков (рис. 5.15).

Замечание

Список может также содержать набор фиксированных значений, который можно будет ввести на следующем шаге Мастера, или список значений одного из полей таблицы или запроса, который является базовым для формы.

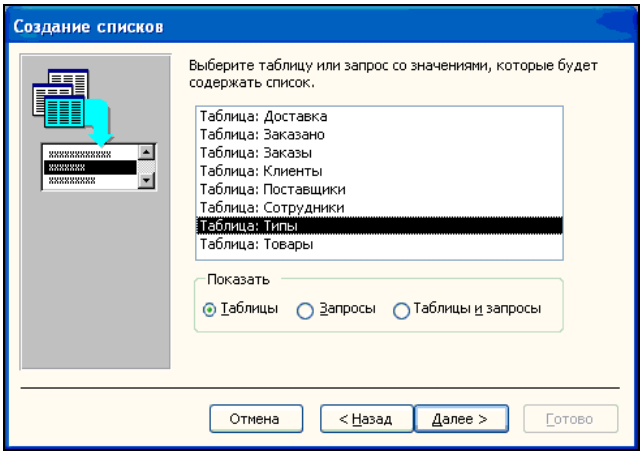


Рис. 5.15. Второе диалоговое окно Мастера списков

Совет

Довольно часто при разработке формы для ускорения процесса перехода по большому количеству записей возникает потребность в создании специального поля со списком, при выборе элемента в котором в форме будет отображаться соответствующая запись из источника данных. В Access с помощью Мастера можно создать список или поле со списком специально для поиска нужной записи в форме. Для этого следует выбрать соответствующий переключатель в первом диалоговом окне Мастера элемента: например, для списка — **Поиск записи в форме на основе значения, которое содержит список** (Find a record on my form based on the value I selected in my list box). Этот переключатель отображается в случае, если для формы задан источник записей: таблица или запрос. В режиме Формы выделение элемента в этом списке будет приводить к переходу на соответствующую запись в форме. Переход реализуется с помощью процедуры обработки события **После обновления** (After Update), созданной Мастером.

5. Во втором диалоговом окне Мастера списков нужно указать таблицу или запрос, значения из которого должны появляться в списке. Выделите в списке нужную таблицу или запрос и нажмите кнопку **Далее** (Next). Появится третье диалоговое окно Мастера списков (рис. 5.16).
6. В этом диалоговом окне нужно определить, какие поля таблицы (в нашем примере, таблицы Типы (Categories)) будут отображаться в списке. Чаще всего в списке должно отображаться одно поле, значение из которого будет выбираться пользователем, но необходимо включить в список еще и ключевое поле, т. к. именно его значение будет использовано в качестве значения данного элемента управления. В нашем примере полей всего два. Поэтому достаточно нажать кнопку >>, чтобы переместить их из списка доступных полей в список выбранных. Затем нажмите кнопку **Далее** (Next).
7. Появится диалоговое окно Мастера, позволяющее задать порядок сортировки элементов в списке (рис. 5.17). Это новое средство, появившееся в Access 2003. Если элементы списка состоят из нескольких полей, их можно отсортировать

последовательно: от одного до четырех полей. Чтобы отсортировать наш список по названию категории, выберите в первом раскрывающемся списке название поля Категория (CategoryName). Щелкнув по переключателю напротив, можно изменить порядок сортировки, например, если требуется отсортировать элементы по убыванию. Нажмите кнопку **Далее** (Next).

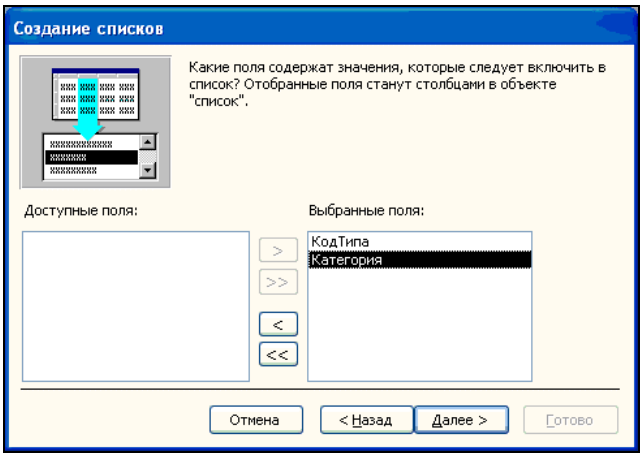


Рис. 5.16. Третье диалоговое окно Мастера списков

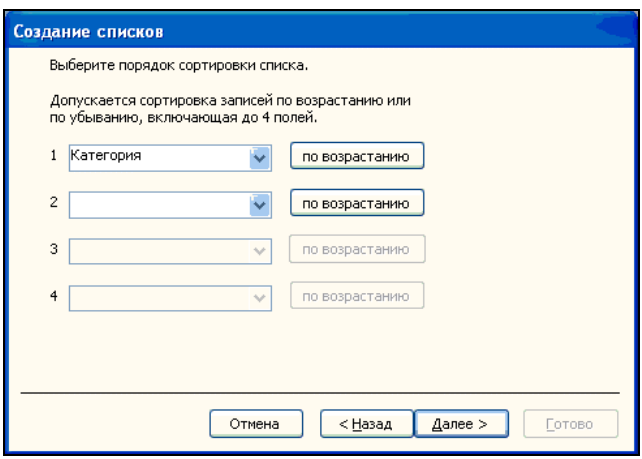


Рис. 5.17. Четвертое диалоговое окно Мастера списков

8. В следующем диалоговом окне (рис. 5.18) отображаются выбранные поля, причем поле КодТипа (CategoryID), которое показывать пользователю не нужно, скрыто. Если ключевое поле содержит нужную пользователю информацию, его можно отобразить в списке (список может содержать два и более полей). Для этого достаточно сбросить флажок **Скрыть ключевой столбец** (Hide key column).

Перемещая с помощью мыши правую границу столбцов, можно задать их ширину. Чтобы настроить ширину столбца по ширине самого длинного значения, достаточно подвести указатель мыши к правой границе столбца и дважды щелкнуть левой кнопкой мыши. Нажмите кнопку **Далее** (Next).

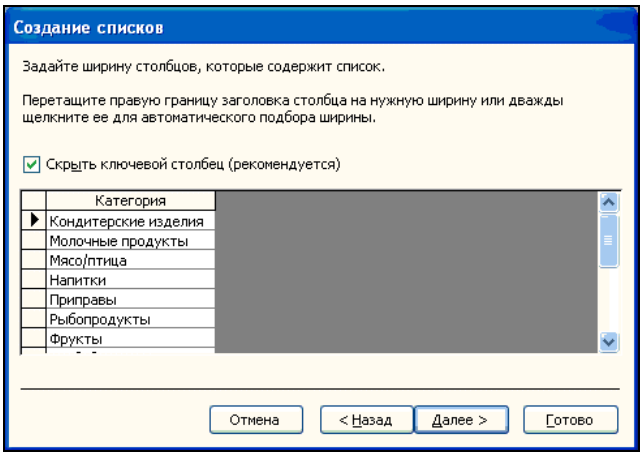


Рис. 5.18. Пятое диалоговое окно Мастера списков

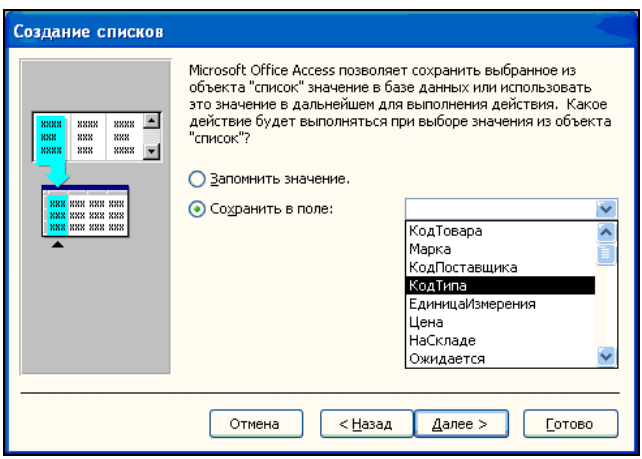


Рис. 5.19. Шестое диалоговое окно Мастера списков

9. Если для формы задан источник записей: таблица или запрос, появится пятое диалоговое окно Мастера списков, представленное на рис. 5.19 (иначе отобразится сразу последнее диалоговое окно Мастера списка, описанное на следующем шаге 10). В этом диалоговом окне нужно выбрать, требуется ли сохранить выбранное значение списка в поле источника данных формы. Если требуется (как в нашем примере), выберите переключатель **Сохранить в поле** (Store that value in

this field). Тогда в раскрывающемся списке справа отобразятся все поля источника данных формы. В нашем примере с использованием формы `Список товаров` (Product List) этим источником данных является таблица `Товары` (Products), и из списка нужно получить значение поля `КодТипа` (CategoryID), поэтому следует выбрать в раскрывающемся списке имя поля `КодТипа` (CategoryID). После этого нажмите кнопку **Далее** (Next). Появится последнее диалоговое окно Мастера списков.

10. В последнем диалоговом окне можно указать название, которое будет использоваться в качестве метки для создаваемого списка. Чтобы после создания списка получить справку по возможностям его настройки, установите флажок, расположенный внизу диалогового окна.
11. Для завершения процесса создания списка нажмите кнопку **Готово** (Finish).

При создании таких элементов управления без помощи Мастера элементов свойства создаваемых элементов нужно настраивать вручную с помощью окна свойств элемента управления. Чтобы открыть окно свойств какого-либо элемента управления формы, выделите его и нажмите кнопку **Свойства** (Properties) на панели инструментов **Конструктор форм** (Form Design), либо дважды щелкните по элементу управления. На рис. 5.20 изображено окно свойств списка.

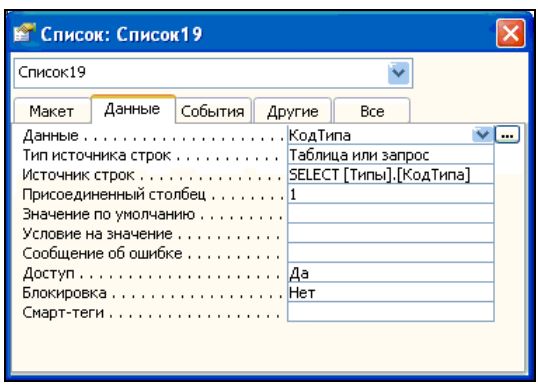


Рис. 5.20. Окно свойств списка

Создание элементов управления с помощью кнопки *Список полей*

Еще одним достаточно простым и удобным средством создания элементов управления является окно **Список полей** (Field List).

Чтобы создать элемент управления, связанный с каким-либо полем источника данных разрабатываемой формы:

1. На панели инструментов **Конструктор форм** (Form Design) нажмите кнопку **Список полей** (Field List). Появится всплывающее окно, содержащее поля из источника данных формы (рис. 5.21).
2. В открывшемся окне выделите нужное поле и с помощью мыши перетащите его в область одного из разделов формы. В форме появится выбранное поле с меткой, содержащей название поля таблицы или запроса. Теперь его можно пере-

мещать и форматировать стандартными методами, используемыми для модификации элементов управления.

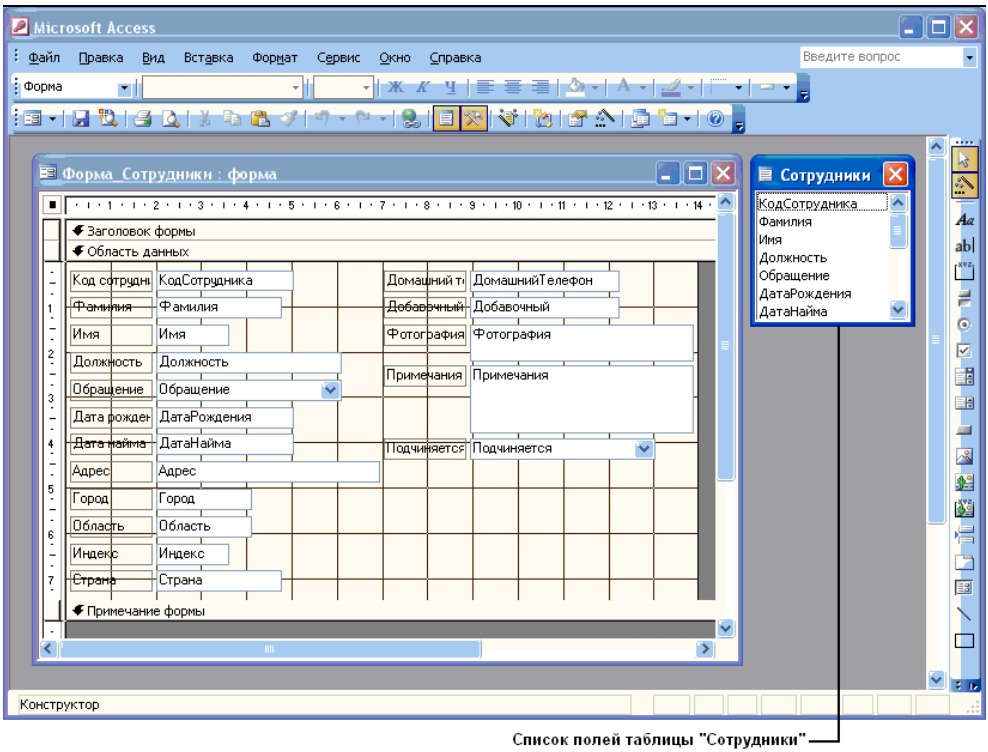


Рис. 5.21. Список полей в режиме Конструктора формы `Форма_Сотрудники`

Удаление элементов формы

Чтобы удалить из формы какой-либо элемент управления:

1. Выделите элемент управления, щелкнув по нему левой кнопкой мыши.
2. Нажмите клавишу <Delete> или выберите команду **Правка | Удалить** (Edit | Delete).

Чтобы удалить сразу несколько элементов управления, выделите их, как описано далее в *разд. "Выделение и перемещение нескольких элементов управления"*, и нажмите клавишу <Delete> или выберите команду **Правка | Удалить** (Edit | Delete).

Выделение формы и элементов управления, отображение окна свойств

Значения свойств формы, ее разделов и каждого элемента управления в форме задаются в окнах свойств соответствующих элементов. Для отображения на экране

окна свойств формы, раздела или элемента управления выделите его и нажмите кнопку **Свойства** (Properties) на панели инструментов **Конструктор форм** (Form Design) или выберите команду **Свойства** (Properties) в контекстном меню.

Выделение формы или элемента управления выполняется описанным далее способом.

- *Форма*. Щелкните левой кнопкой мыши по области формы, находящейся справа от вертикальной линии, позволяющей устанавливать правую границу формы; либо щелкните левой кнопкой мыши по маленькому серому квадрату в верхнем левом углу окна Конструктора формы, где пересекаются вертикальная и горизонтальная линейки; либо выберите команду **Правка | Выделить форму** (Edit | Select Form). Чтобы быстро выделить форму и открыть окно свойств, дважды щелкните левой кнопкой мыши по квадратику в верхнем левом углу окна Конструктора формы. Окно свойств формы позволяет установить значения свойства всей формы.
- *Разделы заголовка формы, примечания формы, разделы верхнего и нижнего колонтитулов*. Щелкните левой кнопкой мыши по области заголовка формы, примечания формы, либо верхнего или нижнего колонтитулов.
- *Раздел области данных*. Щелкните левой кнопкой мыши по области данных. Установка значений свойств раздела области данных происходит так же, как и установка свойств раздела заголовка формы, но новые значения будут применены только к разделу области данных.
- *Элемент управления*. Щелкните по элементу управления левой кнопкой мыши или выделите с помощью мыши прямоугольную область формы, в которую попадает один или несколько элементов управления, требующие выделения. Способ выделения зависит от установленных параметров, которые были описаны в *разд. "Параметры работы с формами"*: область выделения должна либо пересекать выделяемые элементы, либо полностью их охватывать. Еще один способ выделения отдельного элемента управления — выбрать имя необходимого элемента в раскрывающемся списке **Объект** (Object) на панели инструментов **Формат (форма/отчет)** (Formatting (Form/Report)), в котором отображаются имена всех элементов управления в форме. При выделении элемента управления вокруг него появляется рамка с маркером перемещения (в левом верхнем углу) и маркерами изменения размеров (рис. 5.22).

Замечание

Текстовые поля, флажки, переключатели и кнопки имеют связанные с ними метки. При выделении таких элементов управления и метка, и элемент управления выделяются одновременно.

Задание последовательности перехода для элементов управления в форме

При работе с формой переход от одного элемента управления к другому осуществляется с помощью мыши или клавиатуры. Чтобы перейти к следующему элементу управления, щелкните по нему левой кнопкой мыши или нажмите клавишу <Tab>.

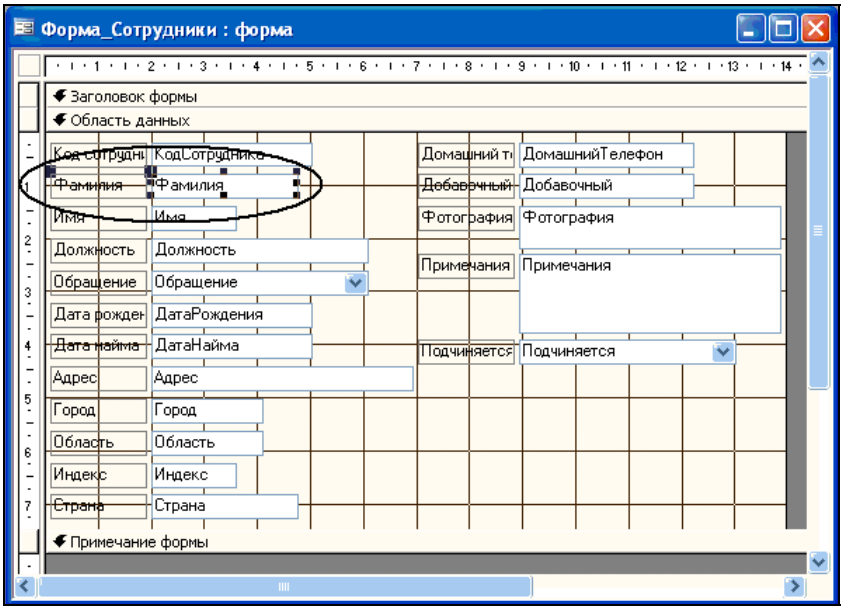


Рис. 5.22. Выделенный элемент управления с именем **Примечание**

Чтобы вернуться к предыдущему элементу управления, щелкните по нему или нажмите комбинацию клавиш <Shift>+<Tab>. С помощью мыши можно сразу перейти к любому элементу управления в форме. С помощью клавиатуры переход от одного элемента управления к другому осуществляется только в заданной последовательности. По умолчанию последовательность перехода определяется порядком создания элементов управления в режиме Конструктора. Изменить эту последовательность позволяет диалоговое окно **Последовательность перехода** (Tab Order).

Чтобы отобразить это диалоговое окно в режиме Конструктора формы, выберите команду **Вид | Последовательность перехода** (View | Tab Order) или щелкните правой кнопкой мыши по форме и выберите в контекстном меню команду **Последовательность перехода** (Tab Order). В этом диалоговом окне отображается список всех элементов управления в форме. Чтобы изменить последовательность перехода, нужно изменить порядок следования элементов в списке, перетаскивая элементы списка с помощью мыши. Перетаскивать нужно, щелкнув по серому квадратику слева от названия поля и удерживая нажатой левую кнопку мыши.

Перемещение и изменение размеров элементов управления

При разработке формы, даже созданной с помощью Мастера, наверняка потребуется изменить расположение и размеры элементов управления. Можно изменить размеры и местоположение отдельного элемента, а можно настроить сразу несколько элементов так, чтобы они были одинакового размера и чтобы были расположены на форме ровно и на равном расстоянии друг от друга. От этого форма сразу приобретет привлекательный вид, и работать с ней будет намного удобнее.

Перемещение и изменение размеров одного элемента управления

Перемещение и изменение размеров элемента управления производится следующим образом:

- Чтобы переместить элемент управления (вместе с его меткой), поместите указатель мыши в любую точку на границе выделенного элемента управления, отличную от маркеров изменения размеров. Когда указатель мыши находится в этой области, предназначенной для перемещения элемента управления, он превращается в инструмент перемещения и меняет свой вид на изображение руки. Нажмите левую кнопку мыши и, не отпуская ее, перетащите элемент управления на новое место. При перетаскивании элемента управления его граница перемещается одновременно с указателем мыши. Когда элемент управления окажется на нужном месте, отпустите левую кнопку мыши.

Замечание

Если элемент управления не имеет связанной с ним метки, его можно перетаскивать также за маркер перемещения, находящийся в левом верхнем углу элемента управления.

- Если нужно переместить элемент управления отдельно от связанной с ним метки, его следует перетаскивать за маркер перемещения, находящийся в левом верхнем углу элемента управления. Указатель мыши, помещенный над маркером перемещения, примет вид руки с вытянутым вверх указательным пальцем. Щелкните по маркеру перемещения и, удерживая нажатой левую кнопку мыши, перетащите элемент управления на новое место, а затем отпустите кнопку мыши.
- Чтобы изменить ширину и высоту элемента управления, поместите указатель мыши над одним из трех угловых маркеров изменения размера выделенного элемента управления так, чтобы он превратился в наклонную двустороннюю стрелку. Нажмите левую кнопку мыши и, удерживая ее нажатой, перетащите границу элемента управления, а затем отпустите кнопку мыши.
- Чтобы изменить только высоту (или только ширину) выделенного элемента управления, поместите указатель мыши над верхним или нижним (левым или правым) маркером изменения размера выделенного элемента управления так, чтобы он превратился в вертикальную (горизонтальную) двустороннюю стрелку. Нажмите левую кнопку мыши и, удерживая ее нажатой, перетащите границу элемента управления, а затем отпустите левую кнопку мыши.

Есть еще способ изменения размеров выделенного элемента управления: с помощью команд меню **Формат | Размер | по размеру данных** (Format | Size | To Fit) и **Формат | Размер | по узлам сетки** (Format | Size | To Grid). С помощью первой команды можно автоматически изменить размеры элемента в соответствии с размером отображаемых в нем данных. К примеру, для текстового поля меняется размер по вертикали в зависимости от заданного размера шрифта для текста. Вторая команда выполняет автоматическое изменение размеров и положения элемента так, чтобы его границы лежали на ближайших линиях, соединяющих узлы сетки. Действие этой команды заметно, если элементы имеют такие размеры, что их границы лежат между узлами сетки, например, когда первоначальные размеры элементов устанавливались при отключенном режиме выравнивания элементов по сетке (о выравнивании по сетке рассказано в следующих разделах).

Выделение и перемещение нескольких элементов управления

Выделение нескольких элементов управления позволяет задавать для них общие свойства, перемещать их и изменять размеры одновременно. Отображение окна свойств для нескольких выделенных элементов производится так же, как и для одного выделенного элемента *(это было описано в разд. "Выделение формы и элементов управления, отображение окна свойств")*.

Для выделения нескольких элементов управления одновременно можно использовать один из описанных далее способов.

- При помощи мыши нарисуйте область выделения вокруг группы элементов управления, как показано на рис. 5.23. Группа элементов управления окажется выделенной.

Замечание

Access позволяет выделить все элементы управления, которые хотя бы частично попадают в область выделения. Такой способ выделения объектов отличается от того, который обычно используется во многих графических редакторах, в которых для выделения объекта необходимо, чтобы он целиком попал в область выделения. Этот способ выделения объектов, применяемый Access по умолчанию, можно изменить для того, чтобы выделять объект только в том случае, если он целиком попадает в область выделения. Для этого выберите команду **Сервис | Параметры** (Tools | Options), раскройте вкладку **Формы и отчеты** (Forms/Reports) и выберите переключатель **охват** (Fully Enclosed) (см. рис. 5.12).

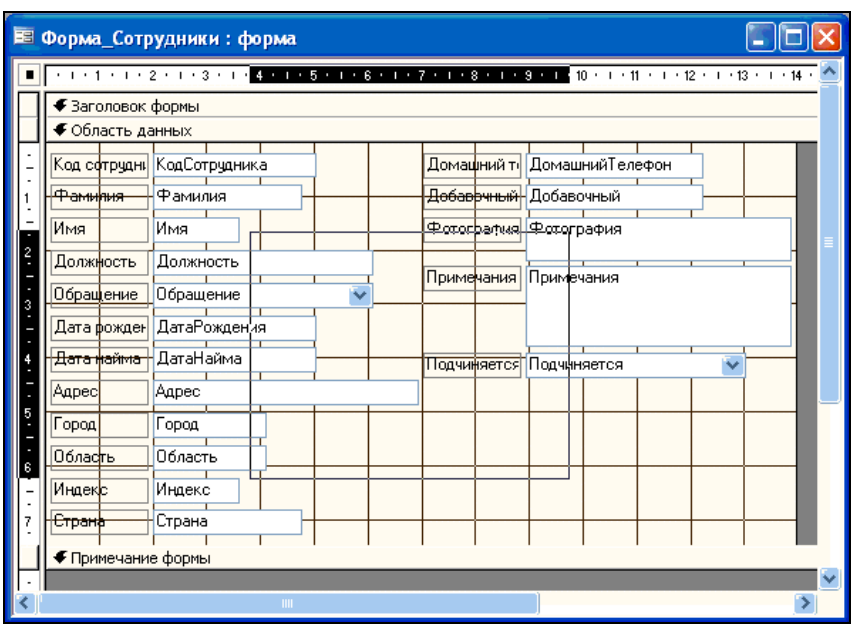


Рис. 5.23. Область выделения, заданная с помощью мыши

- Щелкните левой кнопкой мыши по одному элементу управления, чтобы выделить его, и, удерживая нажатой клавишу <Shift>, выделите остальные элементы управления.
- Чтобы отменить выделение одного из элементов управления в выделенной группе, нажмите клавишу <Shift> и, удерживая ее нажатой, щелкните левой кнопкой по этому элементу управления. Для отмены выделения всей группы элементов управления щелкните левой кнопкой мыши по любой пустой (не содержащей объектов) области формы.

Теперь можно переместить выделенную группу элементов управления, зацепив указателем мыши за границу одного из них и перетащив всю группу на новое место.

Замечание

При выделении или отмене выделения элемента управления со связанной меткой вместе с выделением элемента управления устанавливается или отменяется выделение метки.

Выравнивание одного элемента управления по сетке

В окне Конструктора форм может быть отображена сетка, состоящая из линий толщиной в один пиксел, которые проводятся через каждое деление горизонтальной и вертикальной линеек окна Конструктора форм (т. е. через сантиметр). Чтобы отобразить сетку, выберите команду **Вид | Сетка** (View | Grid). Соответствующая команда меню будет отмечена галочкой. Чтобы отменить отображение сетки, снова выберите команду **Вид | Сетка** (View | Grid). Отметка с команды меню **Сетка** (Grid) будет снята.

Если сетка видна, то с ее помощью можно выровнять элементы управления по горизонтали или по вертикали. Даже если сетка не видна, элементы управления можно "привязать к сетке", выбрав команду **Формат | Привязать к сетке** (Format | Snap to Grid). Если привязка к сетке включена, то команда меню **Привязать к сетке** (Snap to Grid) будет отмечена галочкой. Если перемещать элемент управления при включенной привязке к сетке, то левый верхний угол объекта перемещается дискретно по узлам сетки. Изменение размеров элемента управления в данном режиме также будет происходить дискретно по узлам сетки.

Замечание

Если команда привязки элементов формы к сетке включена, то для перемещения или изменения размеров элемента управления без привязки к сетке нажмите и удерживайте клавишу <Ctrl> в процессе перемещения или изменения размеров элемента управления.

Есть еще один способ выравнивания отдельного элемента управления по сетке: выделите его и выберите команду **Формат | Выровнять | по узлам сетки** (Format | Align | To Grid). В этом случае элемент управления автоматически переместится так, чтобы его левый верхний угол находился в ближайшем узле сетки.

По умолчанию сетка отображается во всех новых формах.

Расстояние между узлами сетки определяется двумя параметрами: густотой сетки по горизонтали (координата X) и по вертикали (координата Y). Густота узлов сетки

задается числом, которое означает количество равных частей, на которые будет разделен каждый сантиметр сетки в данном направлении (по вертикали или по горизонтали). Допустимой является густота узлов величиной от 1 до 64 делений на сантиметр. В случае если густота узлов сетки превышает 10 узлов на сантиметр, узлы сетки не отображаются на экране, т. к. они слишком мало отстоят друг от друга. Чтобы изменить густоту узлов сетки:

1. Выберите команду **Правка | Выделить форму** (Edit | Select Form).
2. На панели инструментов **Конструктор форм** (Form Design) нажмите кнопку **Свойства** (Properties). Появится окно свойств формы.
3. В окне свойств формы раскройте вкладку **Макет** (Format) и при помощи полосы прокрутки найдите свойства **Число делений по X** (Grid X) и **Число делений по Y** (Grid Y).
4. Измените значение свойств **Число делений по X** и **Число делений по Y** на 10.

Выравнивание и изменение размеров нескольких элементов управления

Чтобы выровнять один или несколько элементов управления по сетке или относительно друг друга, выделите выравниваемые элементы, выберите команду **Формат | Выровнять** (Format | Align), а затем выполните действия, описанные далее.

- Для выравнивания элементов управления по узлам сетки выберите команду **по узлам сетки** (To Grid).
- Для выравнивания выделенных элементов управления, расположенных в столбец, так, чтобы их левые края находились на одной вертикальной линии, выберите команду **по левому краю** (Left).
- Для выравнивания выделенных элементов управления, расположенных в столбец, так, чтобы их правые края находились на одной вертикальной линии, выберите команду **по правому краю** (Right).
- Для выравнивания выделенных элементов управления, расположенных в строку, так, чтобы их верхние границы лежали на одной линии, выберите команду **по верхнему краю** (Top).
- Для выравнивания выделенных элементов управления, расположенных в строку, по их нижним границам, выберите команду **по нижнему краю** (Bottom).

У выровненной по одной из вертикальных и горизонтальных сторон группы элементов сразу станет заметной разница в размерах, например, если одно поле немного короче другого. Если поля содержат данные разной длины, например номер телефона и адрес, то такая разница в размере обоснована. Но если поля содержат однотипную информацию одинаковой длины, то просто необходимо сделать их одинакового размера, чтобы форма выглядела аккуратно. Для выравнивания размеров нескольких элементов выделите их любым описанным ранее способом, а затем выберите одну из команд меню **Формат | Размер** (Format | Size):

- команды **по размеру данных** (To Fit) и **по узлам сетки** (To Grid), как уже говорилось в *разд. "Перемещение и изменение размеров одного элемента управления"*, позволяют автоматически изменить размеры выделенных элементов в зависимости от размера содержимого или по линиям сетки соответственно;

- **по самому высокому** (To Tallest), чтобы сделать вертикальный размер всех выделенных элементов равным размеру самого высокого (по вертикали) элемента;
- **по самому низкому** (To Shortest), чтобы сделать вертикальный размер всех выделенных элементов равным размеру самого низкого (по вертикали) элемента;
- **по самому широкому** (To Widest), чтобы сделать горизонтальный размер всех выделенных элементов равным размеру самого широкого (по горизонтали) элемента;
- **по самому узкому** (To Narrowest), чтобы сделать горизонтальный размер всех выделенных элементов равным размеру самого узкого (по горизонтали) элемента.

Чтобы настроить расстояния между несколькими вертикально расположенными (в один "столбец") элементами управления, выделите их и выберите одну из команд меню **Формат | Интервал по вертикали** (Format | Vertical Spacing):

- **Сделать равным** (Make Equal), чтобы сделать равными расстояния в данном направлении между всеми выделенными элементами;
- **Увеличить** (Increase), чтобы увеличить расстояние между элементами на размер одного деления сетки в этом направлении. Если было выделено три или более элементов, расстояния сначала автоматически выравниваются;
- **Уменьшить** (Decrease), чтобы уменьшить расстояние между элементами. Эта команда за одно действие изменяет расстояния на ту же величину, что и команда **Увеличить** (Increase).

Аналогичным образом можно настроить расстояния между несколькими горизонтально расположенными (в одну "строку") элементами управления, выделив их и воспользовавшись командами меню **Формат | Интервал по горизонтали** (Format | Horizontal Spacing).

Не пожалейте немного времени на выравнивание элементов управления. Вы увидите, что выполнить это можно быстро и просто, а ваша форма примет профессиональный вид.

Изменение взаимного расположения и группировка элементов управления

При работе с формой в режиме Конструктора, перетаскивая элементы, можно случайно поместить один элемент так, что он пересечется с другим элементом, т. е. накроет его сверху целиком или закроет какую-то его часть. Скорее всего, это нежелательный эффект, и вы захотите разъединить элементы или удалить один из них. Если один из элементов находится под другим, то говорят, что он расположен "на заднем плане", а другой элемент, соответственно, — "на переднем плане". При выделении элемента, частично накрытого другим, он временно оказывается на переднем плане.

Если элемент управления был полностью накрыт другим, то его никак не получится выделить с помощью мыши, т. к. щелчком мыши в области его расположения будет выделяться тот элемент, что находится сверху. Чтобы получить доступ к элементу, закрытому другим элементом, можно отодвинуть верхний элемент или переместить его на задний план. Для этого выделите его, а затем выберите команду **Формат | На**

задний план (Format | Send to Back). Чтобы переместить выделенный элемент на передний план, выберите команду **Формат | На передний план** (Format | Bring to Front).

Таким же образом можно перемещать на передний или задний план сразу несколько выделенных элементов. По умолчанию порядок перекрывания одного элемента другим, т. е. то, какой из элементов окажется сверху, определяется порядком создания элементов: новый элемент всегда накроет старый (созданный ранее). Этот порядок изменяется командами перемещения элементов с переднего на задний план и наоборот.

В режиме Конструктора можно объединить несколько элементов в группу, выделив их и выбрав команду **Формат | Группировать** (Format | Group). Тогда можно будет выделить всю группу одним щелчком внутри нее. При выделении всей группы вокруг нее отображается охватывающий прямоугольник с маркерами перемещения и изменения размеров. Перетаскиванием маркера изменения размера группы меняются относительные размеры сразу всех элементов в группе. Перетаскиванием за рамку или маркер перемещения группы на новое место перемещается сразу вся группа элементов. Тем не менее, в выделенной группе каждый элемент по-прежнему можно выделить и работать с ним отдельно от других элементов группы.

Чтобы отменить группировку элементов, выделите группу и выберите команду **Формат | Разгруппировать** (Format | Ungroup).

Замечание

Обращаем ваше внимание на то, что, упоминая группу элементов, чаще имеют в виду группу выделенных элементов, а не сгруппированные элементы. Хотя в большинстве случаев работа с группой элементов и сгруппированными элементами одинакова.

Копирование, вставка и удаление элементов управления

Все стандартные команды работы с Буфером обмена Windows применимы и для элементов управления. Выделенный элемент управления или группу элементов управления можно удалить или скопировать в Буфер обмена, а затем вставить в форму и надлежащим образом разместить. В Access для работы с Буфером обмена используются стандартные сочетания клавиш Windows:

- <Ctrl>+<X> — для перемещения выделенных элементов управления в Буфер обмена с удалением их из формы;
- <Ctrl>+<C> — для копирования выделенных элементов управления в Буфер обмена;
- <Ctrl>+<V> — для вставки содержимого Буфера обмена.

Традиционные сочетания клавиш <Shift>+<Delete>, <Ctrl>+<Ins> и <Shift>+<Ins> выполняют те же самые операции.

Чтобы удалить элемент управления, выделите его и нажмите клавишу <Delete>. Если случайно была удалена метка, связанная с элементом управления, то выделите другую метку, скопируйте ее в Буфер обмена, затем выделите элемент управления, с которым нужно связать метку, и вставьте ее из Буфера обмена.

Изменение надписей

Для изменения надписей или меток элементов управления можно использовать стандартные приемы редактирования. Если поместить указатель мыши в пределах границ выделенной метки или надписи и щелкнуть левой кнопкой мыши, появится текстовый курсор, используемый для вставки и удаления текста. Текст элемента управления можно выделить, удерживая нажатой клавишу <Shift> и перемещая курсор при помощи клавиш <←> и <→>. Все операции работы с Буфером обмена Windows применимы и для текста элементов управления.

Аналогичным способом, щелкнув внутри текстового поля, чтобы отобразить текстовый курсор, можно редактировать текст, отображаемый внутри поля. Это не просто текст, а имя присоединенного поля источника данных. Если ввести имя несуществующего поля источника данных или допустить ошибку при редактировании, то при переходе в режим Формы или режим Таблицы в этом текстовом поле будет отображаться сообщение об ошибке: `#Name?`. Самый простой способ исправить ошибку — удалить это поле (с помощью клавиши <Delete>) и создать новое, перетащив на форму поле источника данных, которое нужно присоединить к текстовому полю, из окна **Список полей** (Field List), как было описано в *разд. "Создание элементов управления с помощью кнопки Список полей" данной главы.*

Изменение типа элемента управления

Если при разработке формы понадобилось текстовое поле заменить раскрывающимся списком, вы можете сделать это, просто выделив данный элемент управления и выбрав команду **Формат | Преобразовать элемент в | Поле со списком** (Format | Change To | Combo Box). Так же просто можно преобразовать любой из девяти наиболее часто применяющихся типов элементов управления в подобный ему элемент управления другого типа с помощью команд меню **Формат | Преобразовать элемент в**. Девяти типам элементов соответствуют девять команд этого меню: **Поле** (Text Box), **Надпись** (Label), **Список** (List Box), **Поле со списком** (Combo Box), **Флажок** (Check Box), **Выключатель** (Toggle Box), **Переключатель** (Option Button), **Изображение** (Image) и **Кнопка** (Command Button).

Конечно, для элемента данного типа возможно преобразование в элемент не любого из названных девяти типов, а только в элемент похожего типа. Очевидно, текстовое поле не преобразовать в переключатель, зато его можно преобразовать в надпись, список или поле со списком. В меню будут доступны соответствующие команды. При преобразовании сохраняются значения общих свойств элементов обоих типов.

Оформление формы и ее элементов

Если форма создавалась "с нуля" в режиме Конструктора или с помощью Мастера без применения красивого *автоформата*, она и ее элементы окрашены в непритязательные черно-белые тона. Чтобы сделать форму более привлекательной для глаз, выделить на общем фоне какие-то важные элементы, можно изменить цвет фона формы и цвета ее элементов, выбрать оформление рамок для элементов управления, задать подходящий шрифт, его цвет и размер для текста надписей и других элементов. Параметры форматирования задаются с помощью панели инструментов **Формат**

(форма/отчет) (Formatting (Form/Report)), а также посредством окна свойств формы или ее элементов. В данном разделе мы расскажем о том, как это сделать.

Внимание!

Если форма будет печататься на принтере, не увлекайтесь ее цветовым оформлением: печать фонового рисунка и темного фона формы и ее элементов потребует значительных затрат расходных материалов, да эти изыски и ни к чему для "твердой копии" формы, поскольку на бумаге важнее сами данные, а не их оформление. Для печати лучше всего выбрать белый фон для формы и ее элементов, не очень жирные рамки для элементов и шрифт обычной толщины для текста. Если же форма имеет хорошее цветное оформление, которое важно сохранить для интерактивной работы, печатайте ее в режиме Таблицы.

Применение автоформата

В Access есть замечательное средство, позволяющее практически в одно действие оформить всю форму сразу, включая ее элементы, которое получило название *автоформат* (AutoFormat). Автоформат — это набор параметров оформления формы и ее элементов, который можно применить для форматирования формы. В составе Access есть несколько встроенных автоформатов, но пользователю также предоставляется возможность создать собственные.

Чтобы применить к форме, открытой в режиме Конструктора, один из имеющихся автоформатов:

1. Выделите всю форму, например, выбрав команду **Правка | Выделить форму** (Edit | Select Form).

Замечание

Если вы хотите применить автоформат только к отдельным элементам формы, то выделите их, а не всю форму (в предыдущих разделах мы уже рассказали о том, как выделить один или несколько элементов управления на форме).

2. Выберите команду **Формат | Автоформат** (Format | AutoFormat) или нажмите кнопку **Автоформат** (AutoFormat) на панели инструментов **Конструктор форм** (Form Design). Появится диалоговое окно **Автоформат** (AutoFormat) (рис. 5.24).

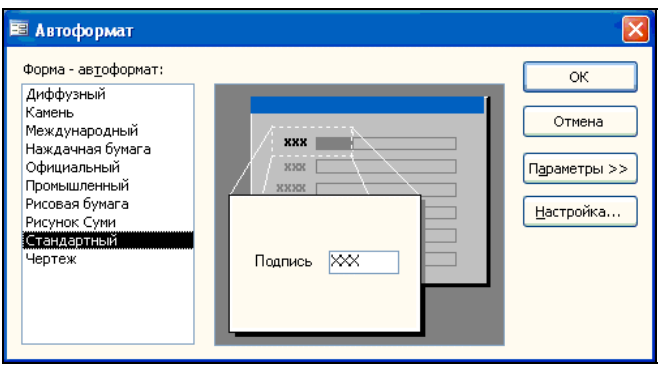


Рис. 5.24. Диалоговое окно **Автоформат**

3. Выделите в списке автоформатов тот, который нужно применить к форме. В правой половине диалогового окна отображается эскиз выбранного в списке автоформата.
4. Нажмите кнопку **OK**.

Замечание

Можно применить к форме только часть автоформата, исключив применение таких параметров автоформата: шрифт, цвет и оформление границ элементов управления. Для этого, прежде чем нажимать кнопку **OK** в диалоговом окне **Автоформат** (AutoFormat), нажмите кнопку **Параметры** (Options). В раскрывшейся нижней части диалогового окна сбросьте флажки для тех параметров формата, которые вы хотели бы оставить в текущей форме прежними: **шрифт** (Font), **цвет** (Color) или **границы** (Border).

Чтобы создать собственный автоформат, необходимо оформить в требующемся стиле любую форму, которая послужит образцом для создания автоформата:

1. Откройте форму-образец в режиме Конструктора.
2. Добавьте на эту форму элементы управления различных типов, если их на ней нет, и отформатируйте их и саму форму так, как хотите, чтобы были оформлены и другие формы с помощью создаваемого вами автоформата.
3. Когда такая форма будет готова, выберите команду **Формат | Автоформат** (Format | AutoFormat) и в появившемся диалоговом окне нажмите кнопку **Настройка** (Customize). Появится диалоговое окно **Настройка автоформата** (Customize AutoFormat).
4. Выберите переключатель **создание нового стиля на основе стиля объекта: Форма "<*имя текущей формы*>"** (Create a new AutoFormat based on the Form '<*имя текущей формы*>') и нажмите кнопку **OK**.
5. В небольшом появившемся диалоговом окне введите имя нового автоформата, например `Мой стиль`. Нажмите кнопку **OK**.
6. Нажмите кнопку **OK**, чтобы применить только что созданный автоформат к текущей форме, или кнопку **Закрыть** (Close), чтобы просто закрыть диалоговое окно **Автоформат** (AutoFormat).

Таким образом будет создан новый автоформат с именем "Мой стиль" на основе формата открытой в данный момент формы. Этот автоформат появится в списке рядом с другими автоформатами Access.

Чтобы удалить ставший ненужным автоформат, выделите его в списке в диалоговом окне **Автоформат** (AutoFormat) и нажмите кнопку **Настройка** (Customize). Выберите переключатель **удаление стиля "<*имя автоформата*>"** (Delete '<*имя автоформата*>'). Если же требуется всего лишь обновить параметры выбранного автоформата на основе формата текущей формы, выберите в этом диалоговом окне переключатель **обновление стиля "<*имя автоформата*>" с атрибутами объекта: Форма "<*имя текущей формы*>"** (Update '<*имя автоформата*>' with values from the Form '<*имя текущей формы*>') и нажмите кнопку **OK**. А затем нажмите кнопку **Закрыть** (Close), чтобы закрыть диалоговое окно **Автоформат** (AutoFormat).

Замечание

Access не запрашивает подтверждение на удаление автоформата, поэтому убедитесь в правильности выбора удаляемого автоформата перед тем, как нажать кнопку **OK**.

Изменение цвета фона формы и ее элементов

Основным параметром, определяющим внешний вид формы, является цвет ее фона. Поэтому при выборе цвета фона необходимо руководствоваться тем, для каких целей предназначена разрабатываемая форма и как в дальнейшем планируется ее использовать. Напомним, что для форм, выводимых на печать, лучше использовать менее яркие цвета, поскольку цветной фон или фон с рисунком потребует больших затрат на расходные материалы для принтера.

Цвет фона раздела формы или ее элемента определяется значением свойства **Цвет фона** (Back Color). В форме, созданной с помощью Мастера форм, фон всех ее разделов зависит от стиля оформления, выбранного при ее создании. Например, к форме `Форма_Сотрудники` был применен автоформат **Стандартный** (Standard), который задает светло-серый цвет фона.

Чтобы изменить цвет фона раздела формы или ее элемента:

1. Выделите раздел или элемент формы, цвет фона которого нужно изменить, щелкнув левой кнопкой мыши по пустой области внутри раздела (заголовка, области данных или примечания), либо по элементу формы (например, по метке, списку или полю).
2. На панели инструментов **Формат (форма/отчет)** (Formatting (Form/Report)) нажмите стрелку справа от кнопки **Цвет заливки/фона** (Fill/Back Color). Появится палитра цветов.
3. Выберите в палитре нужный цвет.

Заметим, что цвет фона раздела формы в Access устанавливается независимо от цвета фона других разделов. Поэтому, чтобы поменять цвет всех разделов, нужно задать новый цвет для каждого раздела отдельно.

Цвет фона раздела формы не влияет на цвет фона элементов управления, находящихся в нем. Для каждого элемента управления тоже можно выбрать свой цвет фона и еще цвет шрифта. Для меток элементов управления обычно задан прозрачный цвет фона, т. е. в качестве фона метки используется та часть фона раздела формы, которую она занимает. По умолчанию для текстовых полей используется белый цвет фона, поэтому цвет текстовых полей (и содержащихся в них данных) контрастен цвету фона формы.

Можно задать прозрачный цвет фона не только меткам, но и таким элементам управления, как текстовые поля и раскрывающиеся списки. Для этого выделите нужный элемент управления, а затем в списке **Цвет заливки/фона** (Fill/Back Color) выберите элемент **Прозрачный** (Transparent). Заметим, что применение прозрачного цвета фона ко всему разделу формы не разрешено. А задание прозрачного фона текстовому полю или раскрывающемуся списку приведет к такому эффекту: в режиме Формы цвет этого элемента управления будет становиться прозрачным, когда этот элемент не будет активным, т. е. когда фокус (точка ввода) будет находиться в другом элементе. Когда же точка ввода будет помещена в данное текстовое поле или

раскрывающийся список, он приобретет непрозрачный цвет фона, который был задан ему ранее. Пример такого отображения полей можно наблюдать в форме `Заказы клиентов` (Customer Orders) демонстрационной базы данных `Борей` (это поля `Название` (Company Name) и `Страна` (Country)). Дело в том, что значение "прозрачный" соответствует на самом деле не свойству элемента **Цвет фона** (Back Color), а свойству **Тип фона** (Back Style). Тип фона элемента может принимать одно из двух значений: **Обычный** (Normal) (когда потеря фокуса элементом не изменяет его цвет) или **Прозрачный** (Transparent).

Изменение фонового рисунка формы

В качестве фона формы Access позволяет задавать не только гладкие цвета, но и растровые рисунки, хранящиеся в разнообразных графических форматах, таких как WMF, GIF и BMP. В комплект поставки Access 2003 входит несколько растровых рисунков, которые хранятся в папке \Microsoft Office\Office11\Bitmaps\Styles.

Чтобы задать фоновый рисунок для формы:

1. В режиме Конструктора выделите всю форму. Для этого щелкните левой кнопкой мыши по квадратику в верхнем левом углу окна Конструктора формы, где пересекаются вертикальная и горизонтальная линейки. После выделения формы в месте пересечения линеек появится черный квадратик.
2. На панели инструментов **Конструктор форм** (Form Design) нажмите кнопку **Свойства** (Properties). Появится окно свойств формы (рис. 5.25).
3. Раскройте вкладку **Макет** (Format) в окне свойств формы и с помощью полосы прокрутки переместитесь в конец списка свойств, где находится свойство **Рисунок** (Picture).

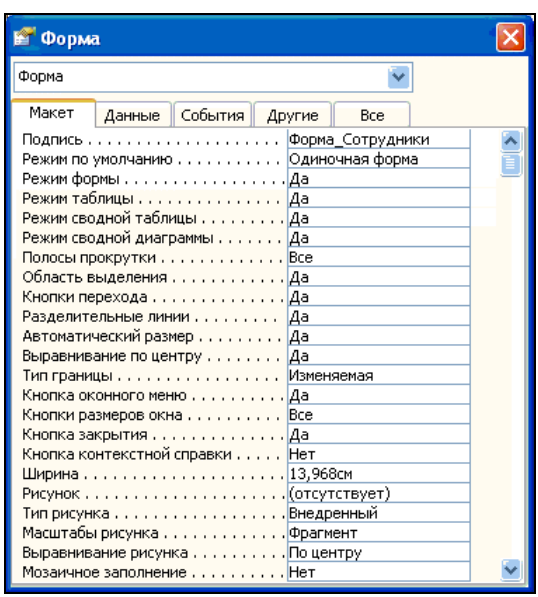


Рис. 5.25. Окно свойств формы

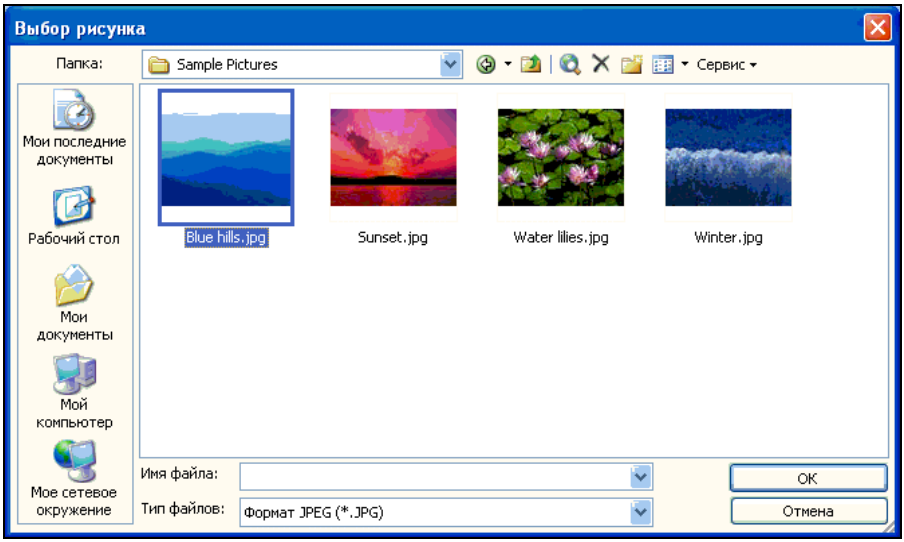


Рис. 5.26. Диалоговое окно **Выбор рисунка**

4. В поле свойства **Рисунок** (Picture) введите путь и имя файла рисунка, либо воспользуйтесь Построителем: поместите точку ввода в поле **Рисунок** (Picture), затем нажмите кнопку **Построить** (Build), которая появится справа от поля в виде маленького квадрата с изображением многоточия на нем. Появится диалоговое окно **Выбор рисунка** (Insert Picture) (рис. 5.26), с помощью которого можно найти и указать необходимый файл рисунка. Это диалоговое окно является стандартным диалоговым окном открытия файлов в Windows. После того как файл рисунка найден, выделите его название, а затем нажмите кнопку **OK**. Access отобразит путь и имя файла в поле свойства **Рисунок** (Picture).
5. Чтобы закрыть окно свойств формы, нажмите кнопку **Закрыть** (Close) вверху окна.

В отличие от цветов фона, фоновый рисунок распространяется на все разделы формы. Чтобы удалить фоновый рисунок, просто удалите содержимое текстового поля свойства **Рисунок** (Picture).

Кроме того, с помощью свойств, сгруппированных на вкладке **Макет** (Format) окна свойств формы, можно настроить отображение фонового рисунка. Свойство **Тип рисунка** (Picture Type) определяет метод, используемый Access для присоединения рисунка к форме. Выберите в раскрывающемся списке этого свойства элемент **Внедренный** (Embedded), если предполагается распространение приложения между пользователями, и форма не должна использовать внешних файлов, которые могут быть перемещены или удалены. Если же имеется много форм с одним и тем же фоновым рисунком, то в раскрывающемся списке выберите элемент **Связанный** (Linked), что позволит уменьшить использование дискового пространства путем включения фонового рисунка в форму при помощи связи с файлом рисунка.

С помощью свойства **Масштабы рисунка** (Picture Size Mode) можно определить размеры фонового рисунка. Для вывода рисунка в исходных размерах в раскрываю-

щемся списке этого свойства выберите элемент **Фрагмент** (Clip). При выборе значения **Вписать в рамку** (Stretch) рисунок растягивается или сжимается по горизонтали и по вертикали до размеров формы; при этом возможно изменение пропорций рисунка. Выбор значения **По размеру рамки** (Zoom) приведет к растяжению или сжатию рисунка с сохранением пропорций до максимально возможных размеров, при которых не происходит обрезка рисунка.

Чтобы определить положение фонового рисунка, выберите один из следующих элементов раскрывающегося списка свойства **Выравнивание рисунка** (Picture Alignment): **Сверху слева** (Top Left), **Сверху справа** (Top Right), **По центру** (Center), **Снизу слева** (Bottom Left), **Снизу справа** (Bottom Right) либо **По центру формы** (Form Center).

Замечание

Чтобы фоновый рисунок центрировался относительно формы, а не окна формы, установите для свойства **Выравнивание рисунка** (Picture Alignment) значение **По центру формы** (Form Center), а не **По центру** (Center).

Если свойство **Мозаичное заполнение** (Picture Tiling) имеет значение **Да** (Yes), то в окне формы или во всей форме (в зависимости от значения свойства **Выравнивание рисунка** (Picture Alignment)) выводится повторяющееся изображение рисунка.

После изменения значения свойства результат немедленно отображается в форме.

Форматирование элементов формы

Кроме оформления фона формы в Access можно легко отформатировать и отдельные элементы управления: изменить цвет текста, цвет, ширину и стиль границы элемента формы.

Изменить внешний вид границы можно с помощью окна свойств элемента управления. Выделите элемент управления, который необходимо изменить, щелкнув по нему левой кнопкой мыши, а затем нажмите кнопку **Свойства** (Properties) на панели инструментов **Конструктор форм** (Form Design). В открывшемся окне свойств раскройте вкладку **Макет** (Format) и с помощью полосы прокрутки найдите свойства, отвечающие за оформление текста и рамок. Выбор значений для большей части свойств рамок элементов управления осуществляется при помощи раскрывающихся списков. Для определения значения свойства, задающего цвет какого-либо параметра, необходимо ввести число, представляющее нужный цвет в системе обозначения цветов в Windows, что достаточно сложно. Поэтому для задания цвета элемента проще воспользоваться кнопкой Построителя в поле данного свойства или соответствующей кнопкой на панели инструментов **Формат (форма/отчет)** (Formatting (Form/Report)).

В табл. 5.3 приведены названия и возможные значения свойств границ элементов управления, которые можно установить в окне свойств элемента управления.

Хотя все параметры форматирования элемента управления формы можно установить в окне свойств, все же более удобным и наглядным способом настройки элемента управления является использование панели инструментов **Формат (форма/отчет)** (Formatting (Form/Report)).

Таблица 5.3. Значения свойств границ элемента управления

Свойство	Описание	Возможные значения
Тип границы (Border Style)	Определяет вид линий, образующих границу элемента управления формы	**Отсутствует** (Transparent); **Сплошная** (Solid); **Штриховая** (Dashes); **Пунктирная** (Short Dashes); **Точечная** (Dots); **Редкоточечная** (Sparse Dots); **Штрихпунктирная** (Dash Dot); **Штрихточечная** (Dash Dot Dot)
Цвет границы (Border Color)	Указывает цвет границы	Номер цвета
Ширина границы (Border Width)	Определяет ширину границы	**Сверхтонкая** (Hairline) или толщиной от 1 до 6 пунктов

Чтобы изменить цвет текста, а также цвет или ширину границы, используя кнопки этой панели инструментов, сначала выделите элемент управления, параметры которого необходимо изменить, а затем щелкните по стрелке, расположенной справа от соответствующей кнопки панели инструментов **Формат (форма/отчет)** (Formatting (Form/Report)), например кнопки **Цвет текста** (Font/Fore Color). В появившейся палитре выберите нужный цвет или ширину границы для элемента управления.

Обратите внимание на то, что изменить цвета текста и границы (свойства **Цвет текста** (Fore Color) и **Цвет линии/границы** (Line/Border Color)) можно только у элементов управления формы. При выделенном каком-либо разделе формы кнопки **Цвет текста** (Fore Color) и **Цвет линии/границы** (Line/Border Color)) панели инструментов **Формат (форма/отчет)** (Formatting (Form/Report)) не доступны. По умолчанию в качестве цвета текста и цвета границы элементов управления используется черный цвет. Выбор цвета границы элемента управления осуществляется с помощью кнопки **Цвет линии/границы** (Line/Border Color)) на панели инструментов **Формат (форма/отчет)** (Formatting (Form/Report)).

С помощью кнопки **Оформление** (Special Effects) на панели инструментов **Формат (форма/отчет)** (Formatting (Form/Report)) можно создавать специальные эффекты отображения элементов управления, например, приподнятые или утопленные текстовые поля. Чтобы определить ширину границы элементов управления, воспользуйтесь кнопкой **Толщина линии/границы** (Line/Border Width). Функции всех кнопок панели инструментов **Формат (форма/отчет)** (Formatting (Form/Report)) были описаны в табл. 5.2 этой главы.

Копирование формата элемента управления

Предположим, вы потратили какое-то время на форматирование одного элемента управления, добились желаемого эффекта, и теперь хотите также отформатировать еще один элемент. Не торопитесь повторять все выполненные действия по форматированию, просто выполните следующую элементарную процедуру:

1. Выделите элемент управления, который уже отформатирован по вашему вкусу.
2. Нажмите кнопку **Формат по образцу** (Format Painter) на панели инструментов **Конструктор форм** (Form Design).

3. Поместите указатель мыши над элементом управления, который нужно отформатировать по имеющемуся образцу. Указатель мыши примет вид инструмента копирования формата (изображение малярной кисти).
4. Щелкните левой кнопкой мыши.

В результате произойдет автоматическое копирование формата: новый элемент управления будет отформатирован по образцу предыдущего элемента управления, причем скопированы будут только те параметры форматирования, применение которых возможно для данного типа элемента управления.

Чтобы отформатировать по образцу сразу несколько элементов управления, следует после выделения элемента, имеющего необходимый формат, щелкнуть дважды по кнопке **Формат по образцу** (Format Painter) и тогда включится режим копирования формата. После этого нужно подряд щелкать по всем элементам управления, которые нужно отформатировать. Чтобы закончить эту операцию, нажмите кнопку **Формат по образцу** (Format Painter) еще раз или клавишу <Esc>.

Такая процедура в несколько раз ускорит процесс форматирования элементов управления. Есть еще один способ ускорить форматирование элементов. Чтобы все новые элементы некоторого типа, создаваемые в текущей форме, автоматически приобретали необходимые параметры форматирования, выделите элемент управления этого типа, который послужит образцом формата, и выберите команду **Формат | Задать стандартные свойства** (Format | Set Control Defaults).

Условное форматирование элементов формы

До сих пор мы говорили о статическом задании параметров форматирования элементов формы. Но Access предоставляет также инструментарий и для динамического изменения форматирования элементов управления при работе с формой в режиме Формы или режиме Таблицы в зависимости от выполнения некоторых условий. Это средство называется *формат по условию*. Чтобы применить данное средство, откройте форму в режиме Конструктора, выделите необходимый элемент управления и выберите команду **Формат | Условное форматирование** (Format | Conditional Formatting). Появится диалоговое окно **Условное форматирование** (Conditional Formatting) (рис. 5.27).

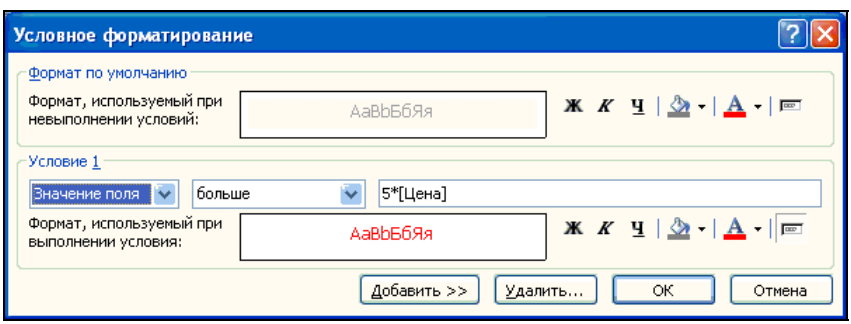


Рис. 5.27. Диалоговое окно **Условное форматирование**

Это диалоговое окно позволяет задать от одного до трех условий, от выполнения которых зависит формат элемента управления. Условия могут быть трех типов:

- значение элемента управления некоторым образом сравнивается с константой или со значением выражения;
- задается логическое выражение, истинное значение которого (**Истина** (True)) определяет, выполнилось ли условие;
- событие получения фокуса данным элементом управления.

Тип условия задается выбором соответствующего элемента в самом левом раскрывающемся списке в группе параметров условия в диалоговом окне. Таких элементов списка, соответственно, три: **Значение поля** (Field Value Is), **Выражение** (Is Expression) и **Фокус в поле** (Field Has Focus).

Операндами выражений здесь могут быть ссылки на другие поля формы в том виде, как они обычно указываются в выражениях Access, и даже вызовы функций VBA. Если условие выполнено, то элемент форматируется заданным для данного условия образом. Если ни одно из условий не выполнилось, элемент форматируется в соответствии с форматом, заданным по умолчанию, который устанавливается в самой верхней части диалогового окна.

Чтобы добавить новое условие, нажмите кнопку **Добавить** (Add) в нижней части диалогового окна. Чтобы удалить некоторые условия, нажмите кнопку **Удалить** (Delete) и установите флажки напротив условий, которые нужно удалить.

Задать параметры форматирования элемента управления в диалоговом окне **Условное форматирование** (Conditional Formatting) для случаев выполнения условий и случая форматирования по умолчанию позволяют кнопки, аналогичные кнопкам форматирования на панели инструментов **Формат (форма/отчет)** (Formatting (Form/Report)). После задания всех параметров форматирования нажмите кнопку **ОК**.

Средство форматирования по условию очень удобно применять, например, для отображения денежных сумм, т. к. легко будет заметить выделенные особым стилем цифры той суммы, что вышла за пределы допустимого интервала.

Например, откройте в режиме Конструктора форму `Подчиненная форма заказов` (Orders Subform) демонстрационной базы данных `Борей` и разверните ее на весь экран с помощью маленькой кнопки **Развернуть** (Maximize) в правом верхнем углу окна формы. Выделите поле `Отпускная цена` (ExtendedPrice) и отобразите диалоговое окно **Условное форматирование** (Conditional Formatting). В самом левом раскрывающемся списке в группе параметров **Условие 1** (Condition 1) выберите элемент **Значение поля** (Field Value Is), в следующем раскрывающемся списке выберите элемент **больше** (greater then) и в текстовом поле введите константу или выражение, например: `5*[Цена]` (или `5*UnitPrice` для английской версии). В списке цветов кнопки **Цвет текста** (Font/Fore Color) в группе параметров **Условие 1** (Condition 1) этого диалогового окна выберите красный цвет. Нажмите кнопку **ОК**.

Это значит, что если полная стоимость заказа продукта превысит пятикратную цену данного продукта, т. е. когда продукт заказан в количестве более пяти, значение поля `Отпускная цена` выделится шрифтом красного цвета. Это мы сможем наблюдать, открыв форму `Заказы` (Orders), подчиненную форму которой мы редактировали в данном примере.

Замечание

Вычисление условий может потребовать некоторого времени, поэтому отображение форматирования тех элементов, для которых условие выполнилось, может произойти с небольшой задержкой.

Использование дополнительных цветов

Стандартные палитры, отображаемые в списке кнопки настройки какого-либо цвета, предоставляют выбор, хотя и из большого, но все же ограниченного набора цветов. Может случиться так, что ни один из этих цветов не подойдет для раздела формы или элемента управления. В такой ситуации Access позволяет создавать свои собственные цвета.

1. В окне свойств элемента формы на вкладке **Макет** (Format) щелкните левой кнопкой мыши по одному из полей свойств, задающих цвет: **Цвет фона** (Back Color), **Цвет текста** (Fore Color) или **Цвет границы** (Border Color).
2. Нажмите кнопку **Построить** (Build), появляющуюся рядом с текстовым полем соответствующего свойства. Появится диалоговое окно **Цвет** (Color), в котором можно выбрать один из 48 цветов. Если среди этих цветов есть нужный, то выделите его, а затем нажмите кнопку **ОК**. При этом выбранный цвет будет принят в качестве значения соответствующего свойства, а диалоговое окно **Цвет** (Color) будет закрыто. Если все же необходимо создать свой цвет, то перейдите к третьему шагу.
3. Нажмите кнопку **Определить цвет** (Define Custom Colors). Диалоговое окно **Цвет** (Color) развернется для отображения схемы определения цвета (рис. 5.28).

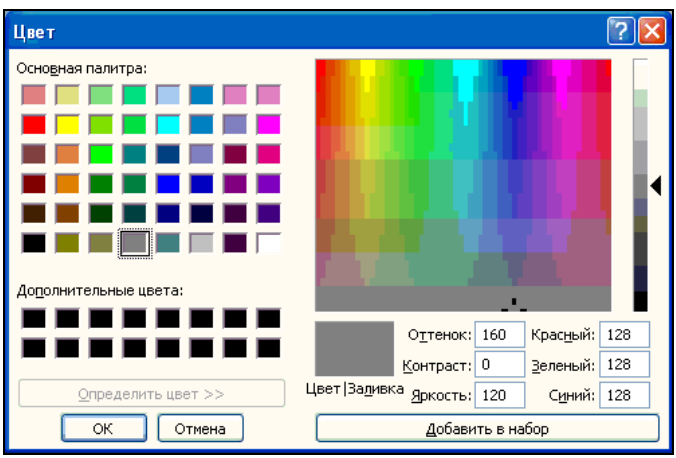


Рис. 5.28. Развернутое диалоговое окно **Цвет**

4. При помощи мыши выберите нужный цвет, щелкнув в любом месте матрицы определения цвета (большого прямоугольника, отображающего весь спектр). Затем отрегулируйте яркость, передвигая бегунок вдоль полосы, расположенной справа от матрицы определения цвета.

Если хотите точно задать цвет, введите тройку его числовых параметров: **Оттенок/Контраст/Яркость** (Hue, Saturation, Luminosity) или **Красный/Зеленый/Синий** (Red, Green, Blue). Параметр оттенок может принимать значение в интервале от 0 до 239, а параметры контрастность и яркость — в интервале от 0 до 240. Каждый из параметров **Красный/Зеленый/Синий** может принимать значение в интервале от 0 до 255.

5. Нажмите кнопку **Добавить в набор** (Add to Custom Colors). Новый цвет будет добавлен в один из 16 элементов **Дополнительные цвета** (Custom Colors). Можно было предварительно (перед выбором нового цвета) щелкнуть по одной из клеток в наборе **Дополнительные цвета** (Custom Colors), чтобы поместить новый цвет именно в эту клетку.
6. Выделите созданный цвет и нажмите кнопку **ОК**. При этом номер заданного цвета будет принят в качестве значения соответствующего свойства, а диалоговое окно **Цвет** (Color) будет закрыто.

Замечание

При создании форм, предназначенных для ввода данных, рекомендуется использовать не очень яркие цвета, которые не приводят к повышенному утомлению глаз и не мешают воспринимать данные.

Сохранение формы

Для сохранения только что созданной или отредактированной формы, открытой в режиме Конструктора, выберите команду **Файл | Сохранить** (File | Save) или нажмите кнопку **Сохранить** (Save) на панели инструментов **Конструктор форм** (Form Design). Если форма была создана и сохраняется в первый раз, появится диалоговое окно **Сохранение** (Save As) с полем для ввода имени новой формы **Имя формы** (Form Name) (рис. 5.29). Введите имя новой формы и нажмите кнопку **ОК**.

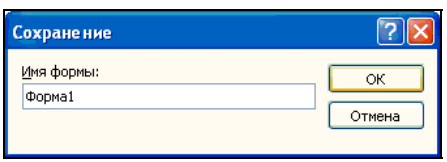


Рис. 5.29. Диалоговое окно **Сохранение**

При работе в режиме Конструктора для сохранения ранее созданной формы под тем же именем достаточно просто нажать кнопку **Сохранить** (Save) на панели инструментов **Конструктор форм** (Form Design).

Форма Access может быть сохранена в трех форматах: формы, отчета и страницы доступа к данным. Чтобы сохранить форму в одном из этих форматов:

1. Выберите команду **Файл | Сохранить как** (File | Save As). Появится диалоговое окно **Сохранение** (Save As), изображенное на рис. 5.30.
2. При необходимости в поле ввода укажите новое название сохраняемой формы.

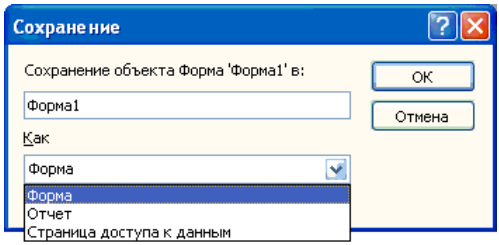


Рис. 5.30. Диалоговое окно команды **Сохранить как**

3. В раскрывающемся списке **Как** (As) выберите элемент **Форма** (Form), **Отчет** (Report) или **Страница доступа к данным** (Data Access Page).
4. Нажмите кнопку **OK**.

Форма, сохраненная как страница, помещается в список объектов **Страницы** (Pages), отображаемый в окне базы данных. В дальнейшем эту страницу можно использовать в документах Интернета. Форма, сохраненная как отчет, помещается в список объектов **Отчеты** (Reports), отображаемый в окне базы данных. Форма, сохраненная как форма, остается в списке объектов **Формы** (Forms) базы данных и в дальнейшем может использоваться по назначению, т. е. для ввода и редактирования записей таблицы или запроса.

Просмотр и ввод данных в режиме Формы

Как уже не раз отмечалось, основной целью простых форм является добавление новых записей, удаление записей или редактирование данных в одной или более таблицах, являющихся источником данных для формы. В этом разделе описаны способы добавления, редактирования и удаления записей таблицы с помощью форм.

Добавление, удаление и редактирование записей с помощью формы происходит при работе с ней в *режиме Формы* или в *режиме Таблицы*. Все же режим Формы более функционален по сравнению с режимом Таблицы. Работа с формой в режиме Таблицы ничем не отличается от работы с обычными таблицами Access *(см. гл. 2)*. А вот в режиме Формы представление и автоматическая обработка данных отличаются более широкими возможностями. В этом разделе мы расскажем об основных приемах работы в режиме Формы. Специальные возможности этого режима, включающие автоматическую обработку введенных данных с помощью процедур обработки событий и вычисляемых полей, рассмотрены в *части II* книги *(см. гл. 9)*.

Окно приложения при работе в режиме Формы

Чтобы открыть форму в режиме Формы:

- если форма закрыта, в окне базы данных дважды щелкните по ее названию;
- если форма открыта в режиме Конструктора, выберите элемент **Режим формы** (Form View) в списке кнопки **Вид** (View) на панели инструментов **Конструктор форм** (Form Design).

После перехода в режим Формы появится панель инструментов **Режим формы** (Form View) (рис. 5.31), содержащая кнопки, функции которых приведены в табл. 5.4.



Рис. 5.31. Панель инструментов **Режим формы**

Таблица 5.4. *Кнопки панели инструментов* ***Режим формы***

Кнопка	Описание	Команда меню
Вид (View)	Переход в режим Конструктора или другой режим	**Вид** | **Конструктор** (View | Form Design) и другие режимы
Сохранить (Save)	Сохранение структуры формы	**Файл** | **Сохранить** (File, Save)
Поиск файлов (Search)	Отобразить/скрыть панель поиска документов Microsoft Office	**Файл** | **Поиск файлов** (File | Search)
Печать (факс) (Print (Fax))	Печать или отправка по факсу содержимого активной формы с текущими параметрами печати без открытия диалогового окна **Печать** (Print)	
Предварительный просмотр (Print Preview)	Предварительный просмотр формы перед печатью	**Файл** | **Предварительный просмотр** (File | Print Preview)
Орфография (Spelling)	Проверка орфографии в выделенном поле	**Сервис** | **Орфография** (Tools | Spelling)
Вырезать (Cut)	Удаление выделенных объектов из формы в Буфер обмена	**Правка** | **Вырезать** (Edit | Cut)
Копировать (Copy)	Копирование выделенных объектов из формы в Буфер обмена	**Правка** | **Копировать** (Edit | Copy)
Вставить (Paste)	Вставка содержимого Буфера обмена в форму	**Правка** | **Вставить** (Edit | Paste)
Отменить ввод (Undo typing)	Отмена последнего изменения записи	**Правка** | **Отменить ввод** (Edit | Undo typing)
Добавить гиперссылку (Insert Hyperlink)	Вставка в форму гиперссылки	**Вставка** | **Гиперссылка** (Insert | Hyperlink)
Сортировка по возрастанию (Sort Ascending)	Сортировка таблицы по возрастающим значениям текущего поля	**Записи** | **Сортировка** | **Сортировка по возрастанию** (Records | Sort | Ascending)

Таблица 5.4 *(окончание)*

Кнопка	Описание	Команда меню
Сортировка по убыванию (Sort Descending)	Сортировка таблицы по убывающим значениям текущего поля	**Записи** | **Сортировка** | **Сортировка по убыванию** (Records | Sort | Descending)
Фильтр по выделенному (Filter by Selection)	Фильтрация записей по выделенному фрагменту	**Записи** | **Фильтр** | **Фильтр по выделенному** (Records | Filter | Filter by Selection)
Изменить фильтр (Filter by Form)	Фильтрация записей на основании критерия отбора введенного в поля формы	**Записи** | **Фильтр** | **Изменить фильтр** (Records | Filter | Filter by Form)
Применение фильтра (Apply Filter)	Применение фильтра. Чтобы удалить фильтр, нажмите эту кнопку еще раз	**Записи** | **Применить фильтр** (Records | Apply Filter/Sort) или **Записи** | **Удалить фильтр** (Records | Remove Filter/Sort)
Найти (Find)	Поиск вхождения заданной подстроки в выделенном поле или во всех полях таблицы. Отображает на экране диалоговое окно **Поиск** (Search)	**Правка** | **Найти** (Edit | Find)
Новая запись (New Record)	Переход на новую пустую запись в конце формы	**Правка** | **Перейти** | **Новая запись** (Edit | Go To | New Record) или **Вставка** | **Новая запись** (Insert | New Record)
Удалить запись (Delete Record)	Удаление текущей записи	**Правка** | **Удалить запись** (Edit | Delete Record)
Окно базы данных (Database Window)	Отображение окна базы данных	**Окно** | **1** (Window | 1)
Новый объект (New Object)	Отображение списка типов новых объектов базы данных, которые можно создать	
Справка: Microsoft Office Access (Microsoft Access Help)	Получение справки по Access	**Справка** | **Справка: Microsoft Office Access** (Help | Microsoft Access Help)

Кнопка **Печать (факс)** (Print (Fax)) позволяет напечатать или отправить по факсу содержимое формы, используя параметры печати по умолчанию и не открывая диалогового окна **Печать** (Print). Чтобы изменить какие-либо параметры печати, выберите команду **Файл** \| **Печать** (File \| Print) и установите нужные параметры в открывшемся диалоговом окне печати.

Чтобы проверить орфографию содержимого какого-либо поля, выделите все содержимое этого поля и нажмите кнопку **Орфография** (Spelling). Чтобы проверить орфографию во всех полях всех записей формы, нажмите кнопку **Орфография** (Spelling), предварительно ничего не выделяя.

Использование кнопки **Найти** (Find) в режиме Формы аналогично ее применению в режиме Таблицы. В появляющееся при нажатии этой кнопки диалоговое окно **Поиск и замена** (Find and Replace) вводится искомое слово, например, имя, фамилия или название улицы, а после получения результатов поиска Access отображает первую найденную запись.

Использование кнопок **Сортировка по возрастанию** (Sort Ascending), **Сортировка по убыванию** (Sort Descending), **Фильтр по выделенному** (Filter by Selection) и **Изменить фильтр** (Filter by Form) также аналогично их применению в режиме Таблицы. Заметим, что сортировка, определенная в форме, заменяет критерий сортировки запроса, используемого в качестве источника данных (в случае, если форма основана на запросе, а не непосредственно на одной или нескольких таблицах).

Добавление записи

В формах, так же как и в таблицах, предусмотрена *пустая запись*, которая предназначена для добавления новых записей в таблицу. Эта пустая запись, как и в таблице, отображается в форме после всех заполненных записей. К ней можно перейти, пролистав в форме все записи с помощью кнопок перехода по записям или с помощью специальной кнопки перехода на новую запись. После ввода данных в эту запись и ее сохранения данные автоматически попадают в таблицу.

При добавлении новой записи в таблицу с помощью формы необходимо, переместившись на пустую запись, заполнить поля этой формы в соответствии с правилами, определенными разработчиком формы. Перемещение между полями формы, используемыми для ввода и редактирования данных, происходит аналогично тому, как это делается в запросах и таблицах, за исключением того, что клавиши перемещения курсора вверх и вниз перемещают курсор между полями, а не между записями. Для сохранения введенных значений нажмите комбинацию клавиш <Shift>+<Enter>.

Замечание

Чтобы можно было добавлять новые записи в таблицу в режиме Формы, свойство **Разрешить добавление** (Allow Additions) должно иметь значение **Да** (Yes). Значение этого свойства можно изменить, используя окно свойств формы в режиме Конструктора.

Чтобы добавить новую запись в таблицу с помощью формы:

1. Откройте форму. Перейдите в режим Формы, если форма была открыта в другом режиме. Для этого нажмите кнопку **Вид** (View) панели инструментов текущего режима и выберите из списка режимов элемент **Режим формы** (Form View). В текстовых полях формы появятся значения полей первой записи таблицы или запроса, являющегося источником данных формы. Access помещает точку вставки в первое поле формы.

   Редактирование полей для формы разрешено в тех же случаях, что и для таблицы или запроса, являющегося источником данных формы, т. е. до тех пор, пока это не будет запрещено в свойстве формы **Разрешить изменение** (Allow Edits).

2. Перейдите к пустой записи, предназначенной для ввода новых данных. Для этого нажмите кнопку **Новая запись** (New Record) на панели инструментов **Режим формы** (Form View) либо маленькую кнопку **Новая запись** (New Record), расположенную в нижней части окна формы.

Замечание

Чтобы отменить создание новой записи после нажатия кнопки **Новая запись** (New Record) на панели инструментов, просто нажмите кнопку перехода к предыдущей записи для того, чтобы отменить создание новой записи.

3. При переходе на новую запись все текстовые поля и другие элементы управления в форме станут пустыми, кроме тех, значение которых определено по умолчанию. Access помещает точку вставки в первое поле формы. Для перемещения точки вставки к следующему полю нажмите клавишу <Enter> или <Tab>. Чтобы вернуться к предыдущему полю, нажмите комбинацию клавиш <Shift>+<Tab> или используйте клавиши управления курсором для перемещения между полями. Можно, конечно, переходить от поля к полю с помощью одного щелчка левой кнопкой по нужному элементу управления.
4. В поля формы введите нужные значения, используя стандартные методы редактирования. В текстовые поля введите нужные значения, а в раскрывающихся списках или полях со списками с помощью мыши выберите необходимые элементы так же, как это делается в стандартных диалоговых окнах Windows.
5. После завершения ввода данных Access сохранит их в памяти, но не добавит новую запись в таблицу. Для добавления записи в таблицу нажмите комбинацию клавиш <Shift>+<Enter> или выберите команду **Записи | Сохранить запись** (Records | Save Record) (данные сохраняются также при переходе к любой другой записи).
6. Для добавления нескольких записей повторите шаги с 2 по 5.

Чтобы запретить добавление новых записей в таблицу с помощью формы:

1. Откройте форму в режиме Конструктора.
2. Выделите всю форму.
3. На панели инструментов **Конструктор форм** (Form Design) нажмите кнопку **Свойства** (Properties).
4. В появившемся окне свойств формы (рис. 5.32) раскройте вкладку **Данные** (Data).
5. На этой вкладке в раскрывающемся списке свойства **Разрешить добавление** (Allow Additions) выберите элемент **Нет** (No).
6. Закройте окно свойств и сохраните форму.

Теперь кнопка **Новая запись** (New Records) как на панели инструментов **Режим формы** (Form View), так и внизу окна формы будет заблокирована, и добавление новых записей в таблицу с помощью формы будет невозможно.

Чтобы при добавлении новой записи какое-либо поле формы имело заранее заданное значение:

1. Откройте форму в режиме Конструктора.
2. Выделите нужное поле, щелкнув по нему левой кнопкой мыши.

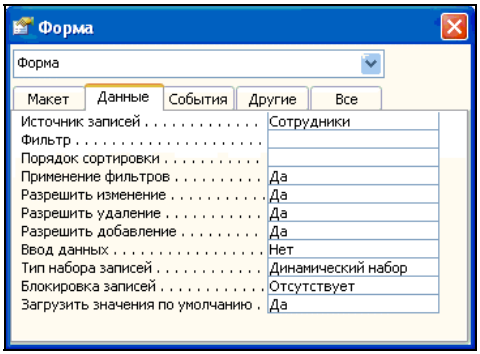


Рис. 5.32. Вкладка **Данные** диалогового окна свойств формы

3. На панели инструментов **Конструктор форм** (Form Design) нажмите кнопку **Свойства** (Properties).
4. В появившемся окне свойств поля (рис. 5.33) раскройте вкладку **Данные** (Data).

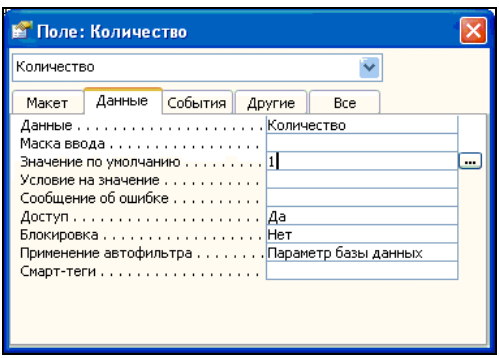


Рис. 5.33. Вкладка **Данные** диалогового окна свойств текстового поля

5. На этой вкладке в текстовое поле свойства **Значение по умолчанию** (Default Value) введите нужное значение.
6. Закройте окно свойств и сохраните форму.

Теперь в каждой новой записи выделенное вами поле будет иметь по умолчанию введенное значение. Это значение можно изменять в новой записи, если свойство **Блокировка** (Locked) этого поля имеет значение **Нет** (No). Подробнее об этом рассказано в следующем разделе.

Изменение записи

Редактирование существующих записей аналогично добавлению новых записей. Однако прежде чем приступить к редактированию записи, ее нужно найти. Для этих целей можно воспользоваться кнопками перехода по записям (навигации), расположенными в нижней части окна формы (рис. 5.34). Эти кнопки выполняют те же

функции, что и аналогичные кнопки в таблицах и запросах. В текстовом поле, находящемся между кнопками навигации, отображается порядковый номер текущей записи. Чтобы отобразить нужную запись с известным номером, просто введите номер в это поле и нажмите клавишу <Enter>. Самая левая и симметричная ей правая кнопки навигации позволяют перейти на первую или последнюю запись таблицы или запроса, являющегося источником данных для основной формы. Вторая слева и симметричная ей справа кнопки навигации позволяют переходить соответственно к предыдущей или последующей записям.

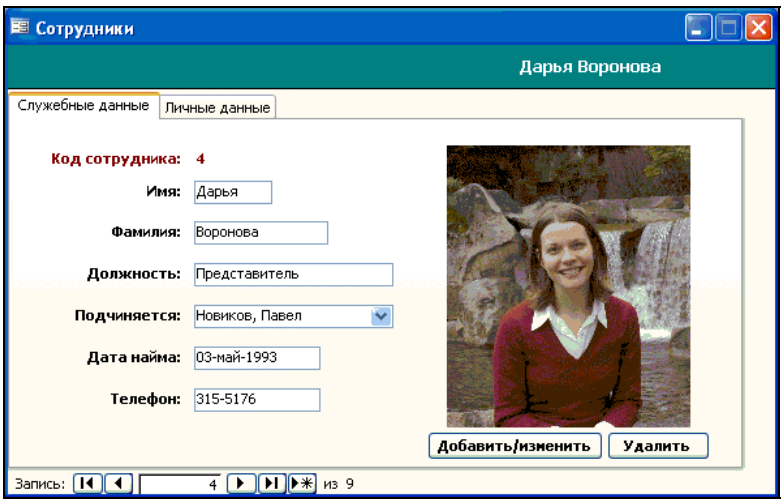


Рис. 5.34. В нижней части окна формы расположены кнопки навигации

Если вы не знаете порядкового номера нужной записи, то для ее поиска можно использовать кнопку **Найти** (Find) панели инструментов **Режим формы** (Form View). Если необходимо, чтобы записи были упорядочены по значению какого-либо поля, выделите его, а затем нажмите кнопку **Сортировка по возрастанию** (Sort Ascending) или **Сортировка по убыванию** (Sort Descending) на панели инструментов **Режим формы** (Form View).

Найдите нужную запись с помощью описанных ранее методов и приступайте к ее редактированию. Перемещение между текстовыми полями, используемыми для редактирования данных, в форме происходит аналогично тому, как это делается в запросах и таблицах, за исключением того, что клавиши перемещения курсора вверх и вниз перемещают курсор между полями, а не между записями. В текстовые поля введите нужные значения, используя стандартные методы редактирования, а в раскрывающихся списках или полях со списками с помощью мыши выберите необходимые элементы так же, как и в стандартных диалоговых окнах Windows. Для сохранения введенных значений нажмите комбинацию клавиш <Shift>+<Enter>.

Чтобы запретить редактирование записи таблицы с помощью формы:

1. Откройте форму в режиме Конструктора.
2. Выделите всю форму.

3. На панели инструментов **Конструктор форм** (Form Design) нажмите кнопку **Свойства** (Properties).
4. В появившемся окне свойств формы раскройте вкладку **Данные** (Data) (см. рис. 5.32).
5. На этой вкладке в раскрывающемся списке свойства **Разрешить изменение** (Allow Edits) выберите элемент **Нет** (No).
6. Закройте окно свойств и сохраните форму.

Теперь редактирование записей, содержащихся в таблице, с помощью формы будет невозможно. Однако если свойство формы **Разрешить добавление** (Allow Additions) имеет значение **Да** (Yes), то с помощью этой формы можно будет не только просматривать записи таблицы, но и добавлять новые.

Чтобы запретить редактирование какого-либо конкретного поля записи:

1. Откройте форму в режиме Конструктора.
2. Выделите нужное поле, щелкнув по нему левой кнопкой мыши.
3. На панели инструментов **Конструктор форм** (Form Design) нажмите кнопку **Свойства** (Properties).
4. В появившемся окне свойств поля раскройте вкладку **Данные** (Data) (см. рис. 5.33).
5. На этой вкладке в раскрывающемся списке свойства **Блокировка** (Locked) выберите элемент **Да** (Yes).
6. Закройте окно свойств и сохраните форму.

Если необходимо запретить редактирование нескольких полей записи, то повторите описанную ранее процедуру для всех полей. Теперь значения указанных вами полей нельзя будет изменить, а в новых записях нельзя будет ввести в них значение.

Замечание

Чтобы какое-либо поле формы имело заранее заданное значение, которое нельзя было бы изменить, укажите для этого поля формы нужное значение в свойстве **Значение по умолчанию** (Default Value) и установите значение свойства **Блокировка** (Locked) равным **Да** (Yes).

Отмена изменения записи

Access не внесет в таблицу изменения до тех пор, пока не будет выделена другая запись или не будет сохранена отредактированная новая запись, например, с помощью команды **Записи | Сохранить запись** (Records | Save Records) или комбинации клавиш <Shift>+<Enter>. Последнее сохранение записи можно отменить, если сразу же после сохранения записи в таблице нажать кнопку **Отменить ввод** (Undo typing) на панели инструментов **Режим формы** (Form View) или выбрать команду **Правка | Отменить ввод** (Edit | Undo typing).

Удаление записи

Для удаления текущей записи в форме, перейдите на нее с помощью кнопок навигации и нажмите кнопку **Удалить запись** (Delete Record) на панели инструментов

Режим формы (Form View) или выберите команду **Правка | Удалить запись** (Edit | Delete Record), либо щелкните по области выделения записи, находящейся слева от области данных формы (рис. 5.35), а затем нажмите клавишу <Delete>. Если текущая запись не имеет подчиненных записей в связанных таблицах или разрешено каскадное удаление записей, ее удаление возможно. В этом случае появится диалоговое окно, запрашивающее подтверждение удаления записи. Подтвердите удаление записи, нажав кнопку **Да** (Yes).

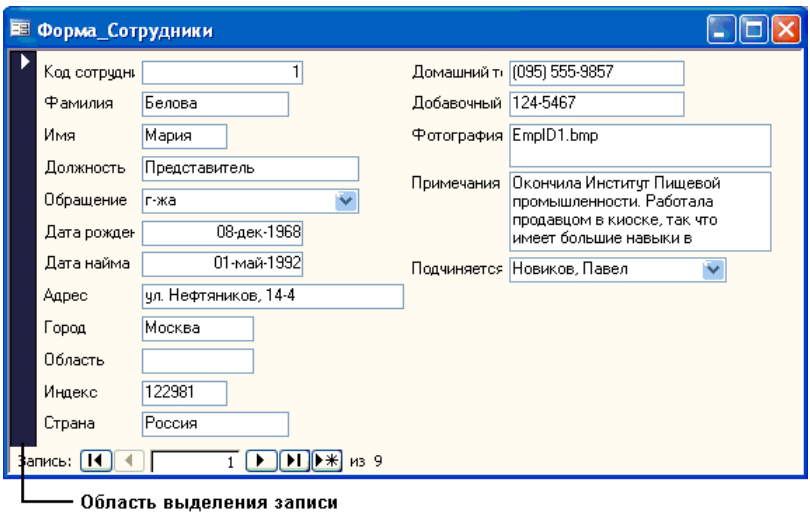


Рис. 5.35. Область выделения записи в форме

В противном случае при попытке удалить запись будет отображено другое сообщение, предупреждающее о том, что невозможно удалить запись, т. к. она содержит связанные с ней записи в других таблицах, с указанием таблицы, содержащей связанные записи. Это следствие использования средства Access, обеспечивающего автоматическую поддержку целостности данных *(см. разд. "Обеспечение целостности данных" в гл. 2)*. В данном случае, прежде чем удалить такую запись, необходимо сначала удалить все записи, связанные с ней, в других таблицах.

ГЛАВА 6



Создание и печать отчетов

Одной из основных задач создания и использования баз данных является предоставление пользователям необходимой информации на основе существующих данных. В Access для этих целей предназначены формы и отчеты. Отчеты позволяют выбрать из базы данных требуемую пользователем информацию и оформить ее в виде документов, которые можно просмотреть и напечатать. Источником данных для отчета может быть таблица или запрос. Кроме данных, полученных из таблиц, в отчете могут отображаться вычисленные по исходным данным значения, например итоговые суммы.

Отчеты и формы Access имеют много общего. Однако в отличие от форм, отчеты не предназначены для ввода и правки данных в таблицах. Они позволяют лишь просматривать и печатать данные. В отчете невозможно изменить исходные данные с помощью элементов управления, как это можно сделать с помощью форм. Хотя в отчетах можно использовать такие же элементы управления для указания состояния переключателей, флажков и списков.

Отчет, как и форма, может быть создан с помощью Мастера. Разделы отчета подобны разделам формы и включают заголовок и примечание отчета, область данных, а также верхний и нижний колонтитулы. В примечание отчета часто помещают поля с итоговыми значениями. Элементы управления могут быть добавлены в отчет с помощью Панели элементов (Toolbox), идентичной той, что используется в режиме Конструктора форм. Форматирование и группировка элементов управления в отчете выполняется аналогично форматированию и группировке элементов управления в форме. Формы могут содержать подчиненные формы, а отчеты — подчиненные отчеты. Эта глава посвящена созданию и изменению простых отчетов, не содержащих подчиненных отчетов. О создании более сложных отчетов рассказано в *гл. 10*.

Данная глава является последней в *части I* нашей книги. На ней мы заканчиваем наше знакомство с Access и описание основных приемов работы с объектами баз данных: таблицами, запросами, формами и отчетами. Этих сведений вполне достаточно для того, чтобы решать несложные задачи обработки данных, которые можно представить в виде некоторого количества связанных таблиц — базы данных. Первая часть книги показывает, что Access является удобным инструментом, который легко осваивать и использовать. Во всех главах этой части мы отсылали читателя к сле-

дующим главам книги за дополнительными сведениями, касающимися предмета рассмотрения. Следующие главы существенно расширят ваше представление о возможностях системы и покажут, что Access не только удобный, но и мощный инструмент для решения прикладных задач.

Автоматическое создание отчета на основе таблицы или запроса

Access предлагает несколько способов создания отчетов. Наиболее простым из них является использование средств автоматического создания отчета. Отчет, создаваемый автоматически на основе таблицы или запроса, называется *автоотчетом*. Access позволяет автоматически создавать отчеты двух форматов: в столбец и ленточный.

Чтобы создать автоотчет:

1. На панели объектов окна **База данных** (Database) щелкните по ярлыку **Отчеты** (Reports) и нажмите кнопку **Создать** (New). Появится диалоговое окно **Новый отчет** (New Report), представленное на рис. 6.1.

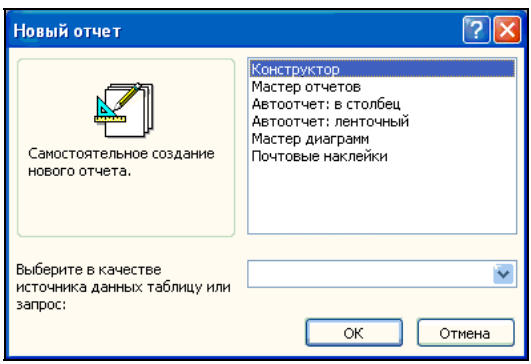


Рис. 6.1. Диалоговое окно **Новый отчет**

2. В списке диалогового окна **Новый отчет** (New Report) выделите один из элементов: **Автоотчет: в столбец** (AutoReport: Columnar) или **Автоотчет: ленточный** (AutoReport: Tabular).
3. В поле со списком, находящемся в нижней части диалогового окна **Новый отчет** (New Report), содержатся имена всех таблиц и запросов базы данных, которые могут быть использованы в качестве источника данных для отчета. Щелкните левой кнопкой мыши по кнопке со стрелкой для открытия списка, а затем выделите в списке нужный элемент.
4. Нажмите кнопку **OK**.

В результате Access автоматически создаст отчет на основе выбранного источника данных, используя ленточный формат или формат в столбец. Ленточный формат располагает поля выводимых записей в строку. Формат в столбец располагает поля

выводимых записей в столбец. Автоотчет, созданный с использованием любого из этих форматов, будет включать все поля и записи, имеющиеся в выбранном источнике данных.

Чтобы созданный отчет можно было использовать в дальнейшем, его необходимо сохранить. Для этого выберите команду **Файл | Сохранить** (File | Save). Затем в текстовое поле появившегося диалогового окна **Сохранение** (Save As) введите название нового отчета (например: `Мой отчет`) и нажмите кнопку **OK**.

Замечание

Есть еще один вариант сохранения отчета: посредством команды меню **Файл | Сохранить как** (File | Save As). С помощью нее отображается немного отличающееся диалоговое окно **Сохранение** (Save As) (рис. 6.2). Введите имя отчета и, прежде чем нажимать кнопку **OK**, убедитесь, что в раскрывающемся списке **Как** (As) этого окна выбран элемент **Отчет** (Report). Выбранный элемент определяет то, как будет сохранен новый отчет, точнее, в виде какого объекта базы данных Access. Дело в том, что в Access, начиная с предыдущей версии Access 2002, появилась возможность сохранить отчет в виде другого объекта базы данных — страницы доступа к данным *(см. гл. 11)*. Сделать это позволяет другой элемент раскрывающегося списка **Как** (As) — элемент **Страница доступа к данным** (Data Access Page).

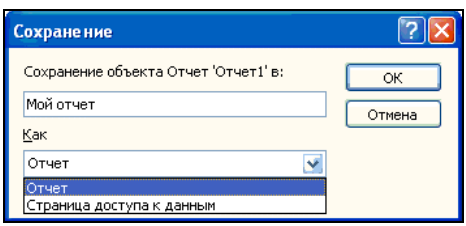


Рис. 6.2. Диалоговое окно **Сохранение**

Создание отчета с помощью Мастера

Еще одним простым и быстрым способом создания отчетов является использование Мастера. С его помощью можно создавать отчеты (от самых простых до включающих подчиненные отчеты), содержащие поля из одной или более таблиц или запросов. Использование Мастера отчетов позволяет очень быстро познакомиться со многими основными средствами разработки отчетов, предоставляемыми Access. Процесс создания отчетов более сложных, чем позволяет Мастер, описан в *гл. 10*.

Чтобы более наглядно представить процесс создания отчета с помощью Мастера, опишем процедуру создания отчета на основе сразу нескольких таблиц: `Товары` (Products), `Поставщики` (Suppliers) и `Типы` (Categories) базы данных `Борей` (Northwind). В отчете будет представлено текущее количество товаров на складе по категориям. Чтобы создать простой отчет:

1. На панели объектов окна базы данных щелкните по ярлыку **Отчеты** (Reports) и нажмите кнопку **Создать** (New). Появится диалоговое окно **Новый отчет** (New Report) (см. рис. 6.1).

2. В списке диалогового окна **Новый отчет** (New Report) выделите элемент **Мастер отчетов** (Report Wizard). В поле со списком, находящемся в нижней части диалогового окна **Новый отчет** (New Report), содержатся имена всех таблиц и запросов базы данных, которые могут быть использованы в качестве источника данных для отчета. Раскройте этот список и выделите в нем таблицу `Товары`. Нажмите кнопку **OK**. Появится первое диалоговое окно Мастера отчетов (рис. 6.3).

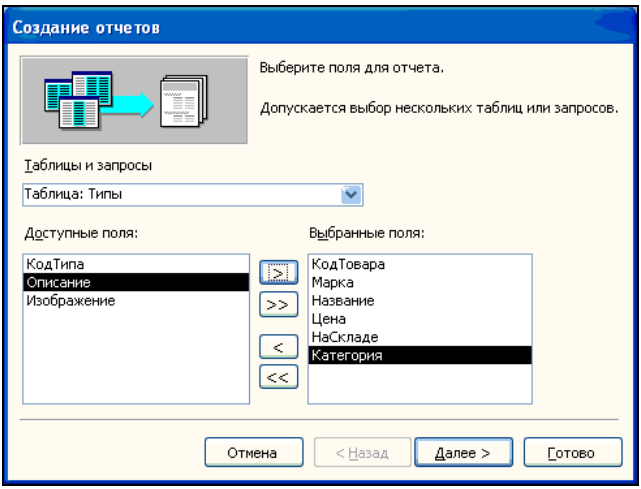


Рис. 6.3. Первое диалоговое окно Мастера отчетов

3. Прежде всего нужно определить, какие поля включить в отчет. Из таблицы `Товары` в отчете будут использоваться следующие поля: `КодТовара` (ProductID), `Марка` (ProductName), `Цена` (UnitPrice), `НаСкладе` (UnitsInStock). С помощью кнопки > (или дважды щелкнув по элементу списка) поочередно переместите эти поля из списка **Доступные поля** (Available Fields) в список **Выбранные поля** (Selected Fields). В отчете поля появляются слева направо, в соответствии с последовательностью их расположения в списке **Выбранные поля** (Selected Fields).

Замечание

Если на каком-либо шаге создания отчета при помощи Мастера отчетов была допущена ошибка или возникла необходимость изменения каких-либо параметров, то для возвращения к предыдущему шагу нажмите кнопку **Назад** (Back). Кроме того, в любой момент можно нажать кнопку **Отмена** (Cancel) для отказа от создания отчета и возвращения к окну базы данных.

4. Кроме того, чтобы видеть поставщиков товара и категории, в создаваемый отчет необходимо добавить поле `Название` (CompanyName) из таблицы `Поставщики` (Suppliers) и поле `Категория` (CategoryName) из таблицы `Типы` (Categories). Для добавления первого недостающего поля раскройте список **Таблицы и запросы** (Tables/Queries) и выберите элемент **Таблица: Поставщики** (Table: Suppliers). Имя поставщика не должно отображаться в конце отчета. Чтобы столбец `Название` выводился сразу за столбцом `Марка`, в списке **Выбранные поля** (Selected Fields)

выделите поле Марка, затем в списке **Доступные поля** (Available Fields) выделите поле Название и нажмите кнопку >. Access переместит поле Название из списка **Доступные поля** (Available Fields) и вставит его в список **Выбранные поля** (Selected Fields) после поля Марка и перед полем Цена. Аналогично добавьте в конец списка поле Категория из таблицы Типы (см. рис. 6.3).

Замечание

Можно таким способом добавлять в создаваемый отчет поля из разных таблиц, которые связаны между собой в схеме данных.

5. Чтобы перейти ко второму диалоговому окну Мастера отчетов, изображенному на рис. 6.4, нажмите кнопку **Далее**.

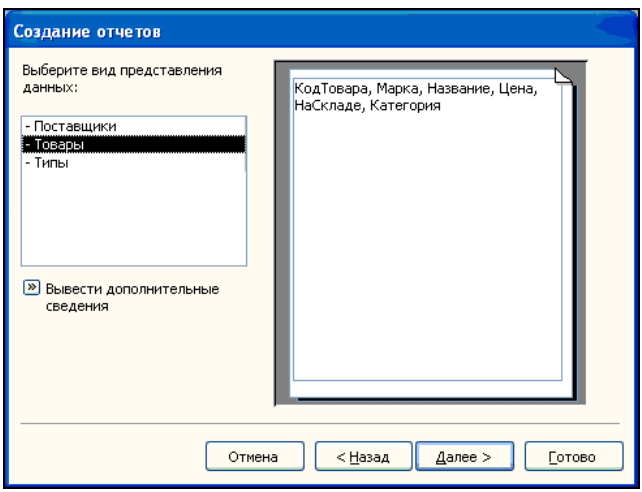


Рис. 6.4. Второе диалоговое окно Мастера отчетов

6. В этом диалоговом окне необходимо указать, каким образом будут структурированы данные в отчете. Мастер проверяет связи между таблицами и в зависимости от выбранного представления предлагает несколько вариантов группировки записи в отчете. Чтобы осуществить группировку по товарам, выделите в списке элемент **Товары** (by Products).
7. Нажмите кнопку **Вывести дополнительные сведения** (Show Me More Information) в левой части диалогового окна для отображения одного из вспомогательных диалоговых окон Мастера отчетов (рис. 6.5). Если в этом диалоговом окне нажать кнопку **Показать примеры** (Show Me Examples), то Access выведет дополнительные вспомогательные диалоговые окна, в которых используются примеры из таблиц Продажи (Sales Reps), Клиенты (Customers) и Заказы (Orders), позволяющие продемонстрировать различные способы группировки, которые Мастер отчетов может автоматически применить к создаваемому отчету. Закройте все вспомогательные диалоговые окна и вернитесь к диалоговому окну Мастера отчетов, изображенному на рис. 6.4. Нажмите кнопку **Далее** (Next) для перехода к третьему диалоговому окну Мастера отчетов.

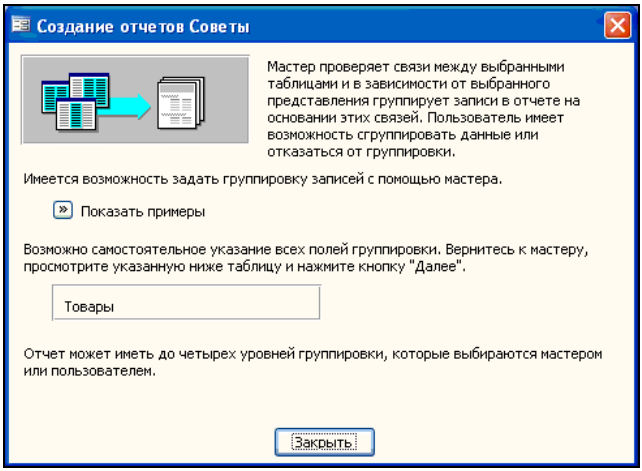


Рис. 6.5. Вспомогательное диалоговое окно Мастера отчетов

8. В третьем диалоговом окне Мастера отчетов необходимо определить дополнительные уровни группировки в отчете. Выделите в списке поле `Категория` (Category) и нажмите кнопку > (или просто дважды щелкните по этому полю в списке), чтобы установить группировку по категории товаров (рис. 6.6).

Замечание

Если потребуется отменить добавление поля группировки, нажмите кнопку < или дважды щелкните по названию этого поля (выделенному жирным шрифтом) в области диалогового окна с эскизом отчета.

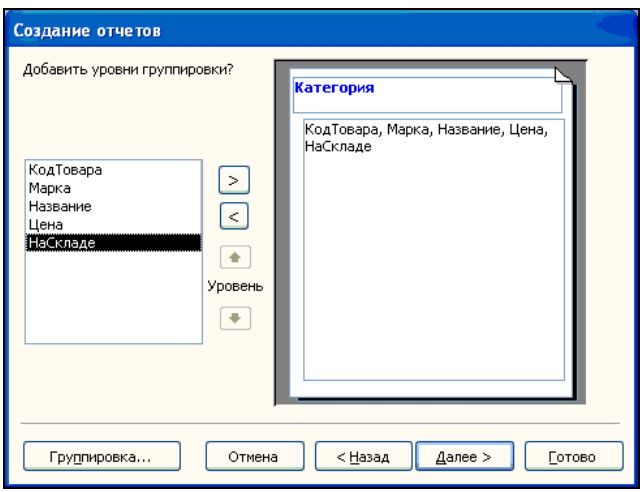


Рис. 6.6. Третье диалоговое окно Мастера отчетов

9. Мастер отчетов позволяет задать интервалы значений, по которым группируются данные. Чтобы изменить их в этом же (третьем) диалоговом окне Мастера отчетов, нажмите кнопку **Группировка** (Grouping Options). На экране появится диалоговое окно **Интервалы группировки** (Grouping Intervals), изображенное на рис. 6.7. Изменяя интервал группировки, можно влиять на способ группировки данных в отчете. Для числовых полей можно задать группировку по десяткам, сотням и прочим значениям. Для текстовых полей возможна группировка по первой букве, первым трем буквам и т. д.

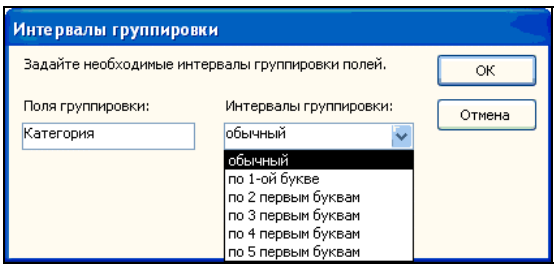


Рис. 6.7. Диалоговое окно **Интервалы группировки**

10. Для создаваемого отчета выберите из списка **Интервалы группировки** (Grouping Intervals) вариант **обычный** (Normal) и нажмите кнопку **OK**. Для перехода к четвертому диалоговому окну Мастера отчетов нажмите кнопку **Далее** (Next).

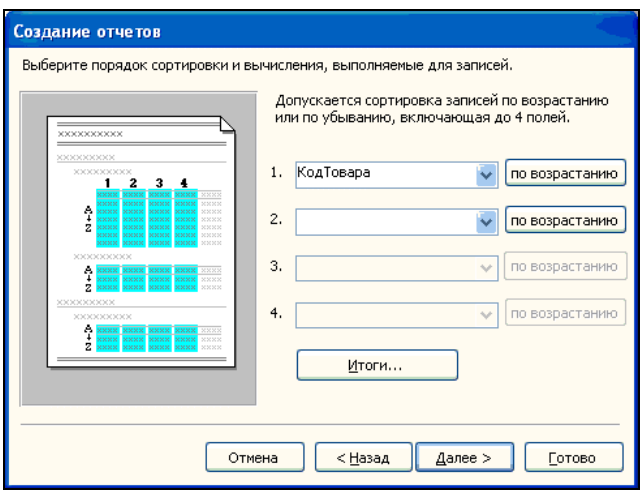


Рис. 6.8. Четвертое диалоговое окно Мастера отчетов

11. Access позволяет сортировать записи в пределах группы по значению произвольного поля, при этом сортировка проводится не более чем по четырем полям (рис. 6.8). Выберите в первом раскрывающемся списке поле `КодТовара` (ProductID). По умолчанию принят порядок сортировки по возрастанию; если

вы хотите выбрать порядок сортировки по убыванию, то нажмите кнопку справа от раскрывающегося списка. Для возврата к сортировке по возрастанию снова нажмите эту кнопку.

12. Чтобы указать для столбца отчета итоговую информацию, нажмите в этом диалоговом окне кнопку **Итоги** (Summary Options). В появившемся диалоговом окне **Итоги** (Summary Options), изображенном на рис. 6.9, можно задать варианты итоговых расчетов для некоторых столбцов. Выбор предоставляется среди всех числовых полей отчета, не являющихся счетчиками, которые Мастер отчетов перечислит в этом диалоговом окне и позволит установить для любого из них флажки **Sum**, **Avg**, **Min** и **Max**. В зависимости от установленных флажков Мастер отчетов добавит в конец отчета соответствующие итоговые поля, вычисляющие итоговую сумму, среднее значение, минимальное или максимальное значение данного поля по всем записям. Группа переключателей **Показать** (Show) позволяет выбрать, следует ли в отчете указывать только итоги или необходимо выдавать полный отчет, добавляя поля с промежуточными итогами в конце каждой группы и поля с полными итогами в конце отчета. Для создаваемого отчета установите флажки **Sum** для поля `НаСкладе` и **Avg** для поля `Цена`, переключатель **данные и итоги** (Detail and Summary) и флажок **Вычислить проценты** (Calculate Percent of Total for Sum), который позволяет выводить долю группы (в процентах) в общем итоге для всех групп (рис. 6.9). Для возврата к диалоговому окну Мастера отчетов нажмите кнопку **OK**.

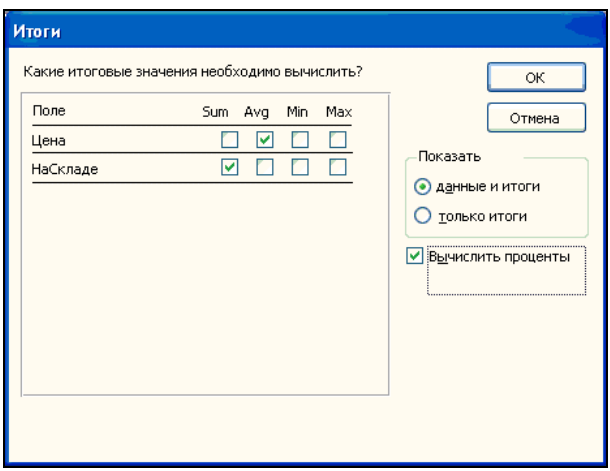


Рис. 6.9. Диалоговое окно **Итоги**

13. Нажмите кнопку **Далее** (Next), чтобы перейти к пятому диалоговому окну Мастера отчетов, изображенному на рис. 6.10. Мастер отчетов предложит на выбор шесть вариантов оформления сгруппированных данных (вариантов макета отчета). В левой части диалогового окна Мастера отображается эскиз выбранного макета. Выберите в группе **Макет** (Layout) переключатель **ступенчатый** (Stepped). По умолчанию Мастер отчетов устанавливает флажок **Настроить ширину полей для размещения на одной странице** (Adjust the field width so all fields fit on a page),

что позволяет сэкономить бумагу и сделать отчет более удобным для чтения. В группе **Ориентация** (Orientation) можно выбрать расположение страницы отчета при печати. Широким отчетам, в которых много полей отображается в одной строке, удобно задавать ориентацию **альбомная** (Landscape). Для отчета из нашего примера оставьте выбранным переключатель **книжная** (Portrait). Чтобы перейти к шестому диалоговому окну Мастера отчетов, нажмите кнопку **Далее** (Next).

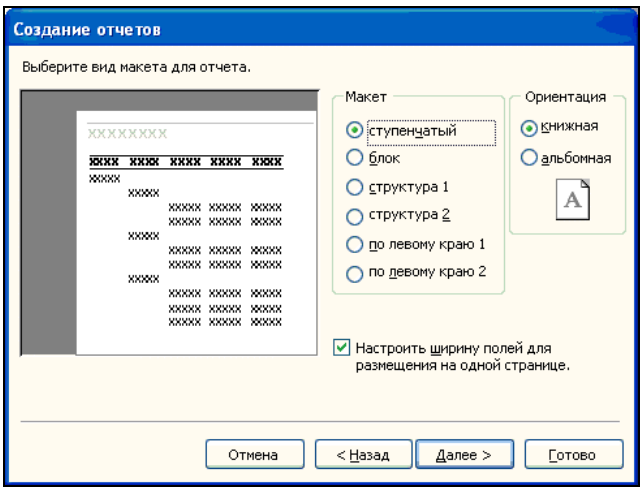


Рис. 6.10. Пятое диалоговое окно Мастера отчетов

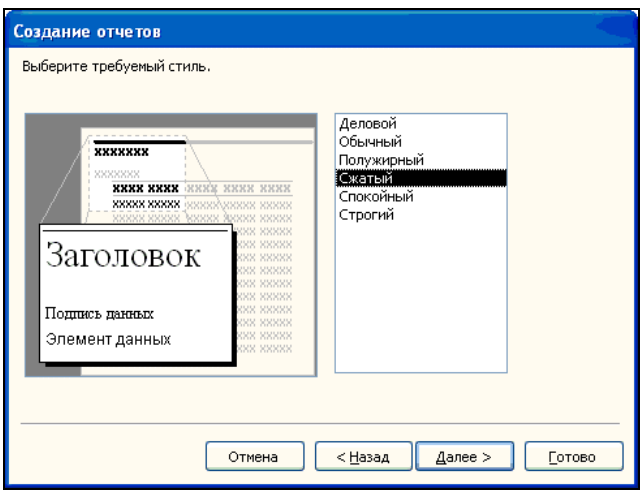


Рис. 6.11. Шестое диалоговое окно Мастера отчетов

14. В шестом диалоговом окне Мастера отчетов выберите один из предложенных стилей оформления отчета. В левой части диалогового окна Мастера отображается эскиз отчета выбранного стиля (рис. 6.11). Для создаваемого отчета

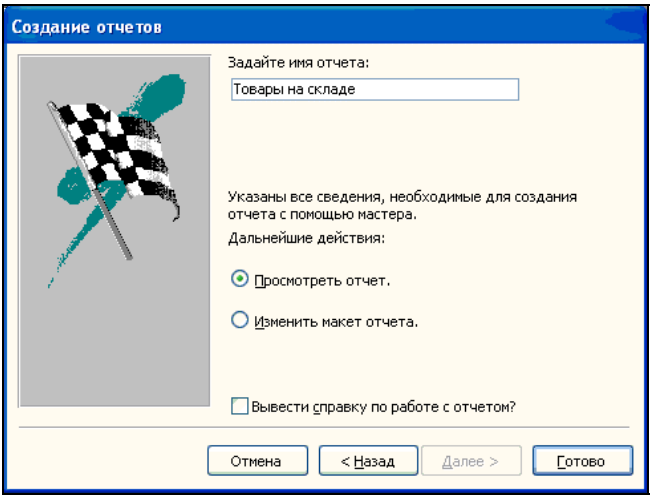


Рис. 6.12. Последнее диалоговое окно Мастера отчетов

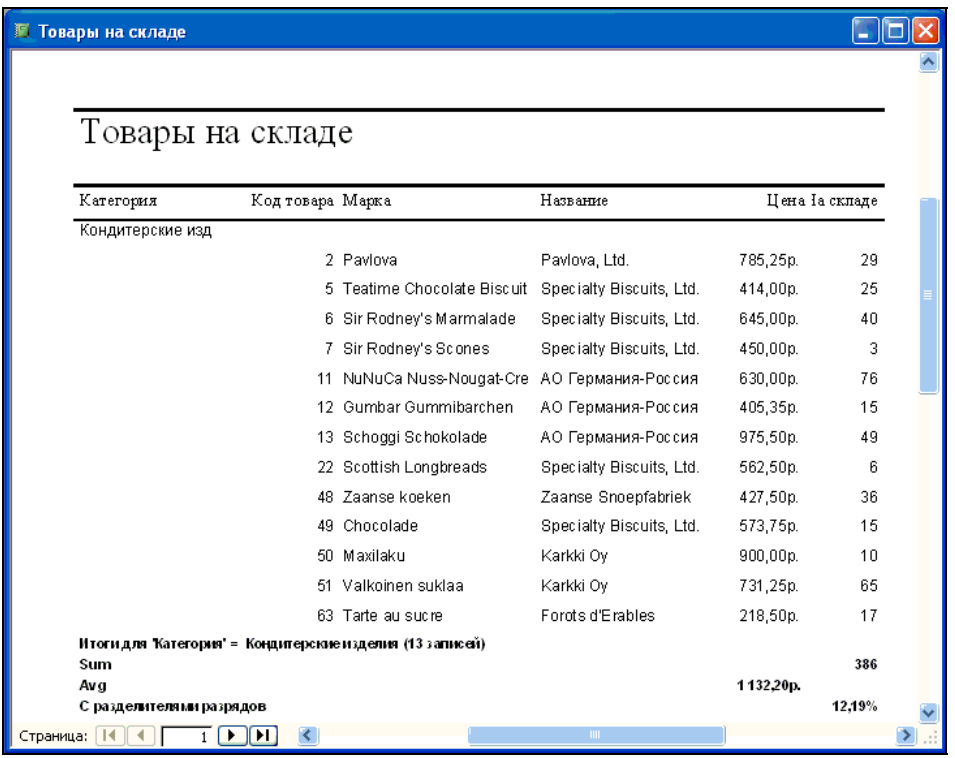


Рис. 6.13. Отчет `Товары на складе`, созданный Мастером

выберите стиль **Сжатый** (Compact), а затем нажмите кнопку **Далее** (Next) для перехода к последнему диалоговому окну Мастера отчетов.

Замечание

В окне Мастера отображается список всех доступных в данный момент стилей. Кроме предоставления шести заранее определенных стандартных стилей Access позволяет настроить любой стандартный стиль или создать свой собственный для использования при оформлении форм и отчетов, в том числе с помощью Мастера. Набор вариантов оформления для отчетов можно пополнить и применить аналогично тому, как это делается для форм *(см. разд. "Применение автоформата", гл. 5)*.

15. В последнем диалоговом окне Мастера отчетов (рис. 6.12) в качестве заголовка отчета введите: `Товары на складе`. Мастер отчетов сохранит созданный отчет под этим же именем. Выберите переключатель **Просмотреть отчет** (Preview the Report) и нажмите кнопку **Готово** (Finish) для завершения создания отчета. Мастер отчетов создаст отчет и отобразит его в режиме предварительного просмотра. Если при этом нужно получить справку по работе с отчетом, предварительно установите флажок **Вывести справку по работе с отчетом?** (Display Help on Working with the Report?).

На рис. 6.13 изображен отчет, созданный Мастером. Для просмотра всех частей отчета воспользуйтесь вертикальной и горизонтальной полосами прокрутки. Этот отчет можно использовать непосредственно в том виде, как его создал Мастер, или применить его как основу для разработки более сложного отчета.

Преобразование отчетов

Вы создаете базу данных, накапливаете в ней данные, редактируете и анализируете их, используя средства Access. В какой-то момент, когда результаты обработки данных готовы, вы представляете их в виде отчета Access — внешне выглядящего как обычный документ, но этот документ можно просмотреть лишь в базе данных Access, в которой он сохранен. Такой вариант не годится, когда результаты своей работы в виде отчета Access вы хотите передать кому-то другому, кто не работает с Access или, возможно, даже вообще не имеет дела с базами данных. Самый простой вариант — напечатать отчет на принтере, и тогда формат, в котором он был создан, не будет иметь значения. Но чаще всего отчет требуется передать именно в электронном виде, например, переслать по электронной почте или другим способом предоставить данные из отчета, чтобы в дальнейшем они могли быть использованы в других документах.

Для решения этой задачи Microsoft Access предоставляет целый ряд возможностей преобразования отчетов в другие форматы. Это формат документов RTF, с которым можно работать в Word и некоторых других текстовых редакторах, формат электронных таблиц Excel, страницы HTML, разнообразные текстовые форматы (файлы с расширением txt, csv, tab, asc) с возможностью сохранения содержимого отчета на выбор в кодировке Windows, MS-DOS или Unicode, специальный формат снимков отчетов (snapshot). Начиная с предыдущей версии Access 2002, к этим форматам добавились страницы доступа к данным и XML.

В настоящем разделе мы расскажем о том, как просто преобразовать отчет Access в любой из перечисленных форматов. И вы сможете передать отчет кому-то другому в наиболее удобном для него виде или использовать выбранный формат, чтобы поработать с отчетом в другом приложении.

Преобразование отчета в формат документов Word и Excel

Access позволяет преобразовать созданный отчет в документ Microsoft Word формата RTF. Чтобы выполнить такое преобразование:

1. Откройте отчет в любом режиме или просто выделите его в окне базы данных.
2. Выберите команду **Сервис | Связи с Office | Публикация в Microsoft Office Word** (Tools | Office Links | Publish It with MS Word).

В результате Access создаст файл формата RTF в текущем каталоге, например в папке Мои документы (My Documents), по умолчанию используя в качестве имени файла название отчета, а затем откроет его в Word.

С помощью соседней команды меню **Сервис | Связи с Office | Анализ в Microsoft Office Excel** (Tools | Office Links | Analyze It with Microsoft Excel) можно преобразовать отчет в электронную таблицу Excel. При выборе этой команды автоматически создается новая рабочая книга Excel, в которую помещается рабочий лист с представлением отчета в виде электронной таблицы. В качестве имени файла XLS используется имя отчета.

Замечание

Команды преобразования данных в формат документов Word и Excel доступны также для таблиц, запросов и форм Access. Кроме того, для таблиц и запросов доступна еще и команда **Сервис | Связи с Office | Слияние в Microsoft Office Word** (Tools | Office Links | Merge It with Microsoft Word). С помощью этой команды вызывается Мастер слияния, который позволяет на основе выделенной таблицы или запроса создать документ Word, связанный с этим источником данных, или связать данную таблицу или запрос с уже существующим документом Word. Благодаря этой связи, когда данные в таблице или запросе изменятся, соответствующим образом будут обновляться связанные поля в документе.

Преобразование отчета в другие форматы

В Access существует средство преобразования отчетов в целый ряд других форматов. Это средство называется *экспортом*. Чтобы экспортировать отчет Access в какой-нибудь другой формат, выделите отчет в окне базы данных или откройте его в любом режиме, а затем выберите команду **Файл | Экспорт** (File | Export). Появится диалоговое окно экспорта объекта базы данных, похожее на стандартное окно сохранения файла, в котором в качестве типа сохраняемого файла можно выбрать любой формат из предлагаемого списка:

- **Microsoft Office Access** (для копирования отчета в другую базу данных);
- **Microsoft Excel 5-7** (для преобразования отчета в формат ранних версий Excel);

- **Microsoft Excel 97-2003** (для преобразования отчета в формат последних версий Excel);
- **Страницы HTML** (HTML Documents) (для преобразования отчета в формат статических Web-страниц в формате HTML);
- **Текстовые документы** (Text Files) (для преобразования в один из текстовых форматов Windows или MS-DOS: TXT, CSV, TAB, ASC);
- **Формат RTF** (Reach Text Format) (для преобразования отчета в документ Word);
- **Снимок** (Snapshot Format) (для преобразования отчета в формат статических снимков);
- **Документы XML** (XML Documents) (для преобразования отчета в формат динамических Web-страниц в формате XML).

Выбрав формат в раскрывающемся списке **Тип файла** (Save as type), введите в поле **Имя файла** (File name) имя файла, который будет создан, и нажмите кнопку **Экспорт** (Export). Предварительно можно установить в этом диалоговом окне флажок **Автозагрузка** (Autostart), если вы хотите, чтобы после экспорта автоматически открылся созданный файл в приложении, в котором его можно просмотреть.

Экспортировать отчет в один из перечисленных форматов можно также с помощью макроса, включающего макрокоманду `ВывестиВФормате` (OutputTo), или используя одноименный метод `OutputTo` объекта `DoCmd` в процедуре VBA *(о применении макросов и программировании на VBA см. гл. 12 и 13)*.

Вы можете также послать отчет по электронной почте в одном из форматов экспорта прямо из Access с помощью команды меню **Файл | Отправить | Сообщение (как вложение)** (File | Send | Message (As Attachment)) или программным путем, посредством макроса, используя макрокоманду `ОтправитьОбъект` (SendObject), или с помощью процедуры VBA, используя одноименный метод `SendObject` объекта `DoCmd`.

(О преобразовании отчетов Access в различные форматы Web-страниц подробнее рассказано в разд. "Преобразование объектов базы данных в Web-страницы", гл. 11.)

Снимки отчетов

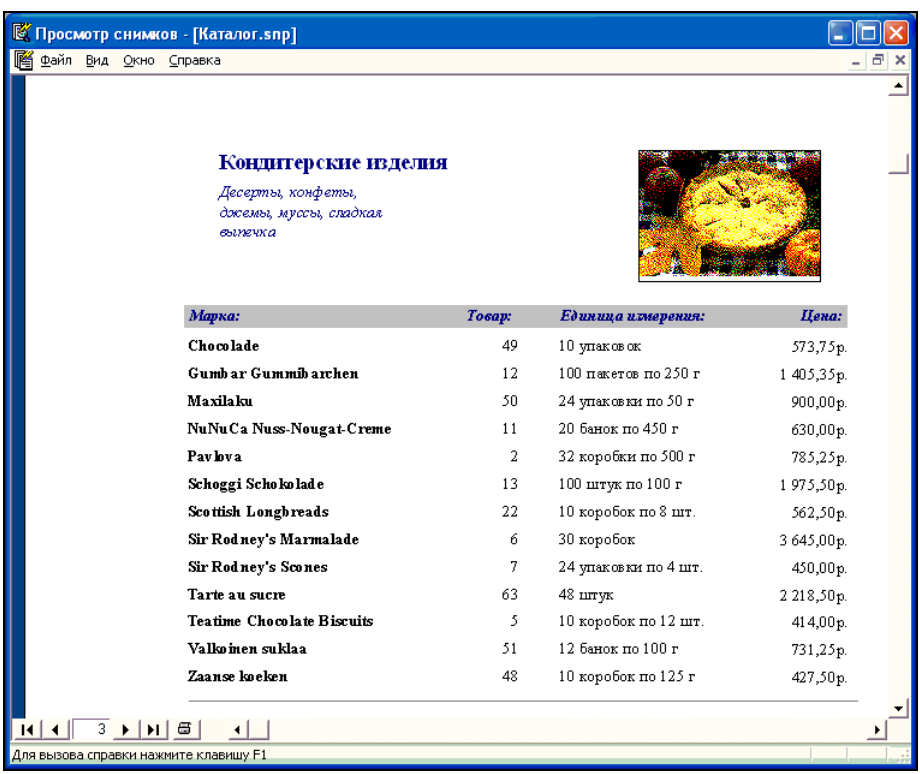
Еще в версии Access 2000 появилось дополнительное средство публикации отчетов, отличающееся от экспорта в обычные наиболее распространенные форматы документов. Это средство — преобразование отчетов Access в специально разработанный для задачи их публикации формат *снимков отчетов* (report snapshot). Основная идея этого средства — компактное представление отчетов Access в виде отдельных от базы данных файлов (снимков отчетов), которые можно просматривать, листать и печатать без использования Microsoft Access с помощью специально созданного очень небольшого приложения, названного *Просмотр снимков* (Snapshot Viewer). Эта программа совершенно независима от самого Microsoft Access и может быть установлена на компьютер без установки пакета Microsoft Office. Ее дистрибутив входит в состав пакета, а также доступен через Интернет на Web-узле Microsoft **http://www.microsoft.com**. Электронную версию отчета Access в виде файла снимка отчета можно также переслать по электронной почте или распространить другим способом передачи файлов. Те пользователи, на чьих компьютерах установлена программа Просмотр снимков, смогут просмотреть полученный в виде снимка отчет. В новой версии Access 2003 это средство сохранилось.

Для преобразования отчета Access в формат снимка требуется воспользоваться тем же средством экспорта объектов базы данных Access, упомянутом в предыдущем разделе.

Снимок отчета — файл с расширением snp — содержит копии всех страниц отчета, включая графические и другие вложенные объекты. Чтобы на основе отчета Access создать снимок:

1. В окне базы данных в списке отчетов выделите нужный отчет и выберите команду **Файл | Экспорт** (File | Export). Появится диалоговое окно **Экспорт объекта** (Export Object).
2. В раскрывающемся списке **Тип файла** (Save as type) выберите элемент **Снимок** (Snapshot Format), в раскрывающемся списке **Папка** (Save in) выберите папку, в которой будет сохранен файл, а в поле **Имя файла** (File name) введите имя файла снимка. Нажмите кнопку **Экспорт** (Export).

Созданный в результате снимок отчета будет сохранен в указанной папке, после чего, если в окне экспорта был установлен флажок **Автозагрузка** (Autostart), он будет открыт в автоматически запущенном (если ранее не было запущено) приложении Просмотр снимков (рис. 6.14).



Марка:	Товар:	Единица измерения:	Цена:
Chocolade	49	10 упаковок	573,75р.
Gumbar Gummibarchen	12	100 пакетов по 250 г	1 405,35р.
Maxilaku	50	24 упаковки по 50 г	900,00р.
NuNuCa Nuss-Nougat-Creme	11	20 банок по 450 г	630,00р.
Pavlova	2	32 коробки по 500 г	785,25р.
Schoggi Schokolade	13	100 штук по 100 г	1 975,50р.
Scottish Longbreads	22	10 коробок по 8 шт.	562,50р.
Sir Rodney's Marmalade	6	30 коробок	3 645,00р.
Sir Rodney's Scones	7	24 упаковки по 4 шт.	450,00р.
Tarte au sucre	63	48 штук	2 218,50р.
Teatime Chocolate Biscuits	5	10 коробок по 12 шт.	414,00р.
Valkoinen suklaa	51	12 банок по 100 г	731,25р.
Zaanse koeken	48	10 коробок по 125 г	427,50р.

Рис. 6.14. Снимок отчета Каталог (Catalog) из демонстрационной базы данных Борей

Замечание

Если программа Просмотр снимков на вашем компьютере не была установлена (она не устанавливается со стандартной конфигурацией Access), то в этот момент будет произведена автоматическая попытка установить ее. Поэтому, возможно, вы получите сообщение с требованием вставить в дисковод дистрибутивный диск Microsoft Office.

В нижней части окна приложения Просмотр снимков содержится панель навигации, которая позволяет листать страницы отчета при просмотре. Крайняя справа кнопка на этой панели позволяет выполнить печать отчета. Щелкнув правой кнопкой мыши по области отчета, вы получите контекстное меню, которое включает команды, дублирующие функции кнопок панели навигации и еще команду **Масштаб** (Zoom), с помощью которой вы можете регулировать масштаб отображения страниц отчета в окне программы Просмотр снимков. Это приложение позволяет одновременно просматривать несколько снимков отчета. Переключатся между снимками и управлять расположением их окон внутри приложения можно с помощью команд меню **Окно** (Window). Кроме просмотра и печати снимков программа просмотра позволяет также пересылать их по электронной почте. Для этого нужно выбрать команду меню **Файл | Отправить** (File | Send). Тогда появится окно сообщения Microsoft Outlook, в которое автоматически попадет снимок отчета в виде вложенного файла (рис. 6.15). Останется заполнить необходимые поля и отправить сообщение с помощью кнопки **Отправить** (Send). Получателю сообщения достаточно дважды щелкнуть мышью по значку вложенного файла, чтобы запустить программу Просмотр снимков и открыть в ней полученный снимок.

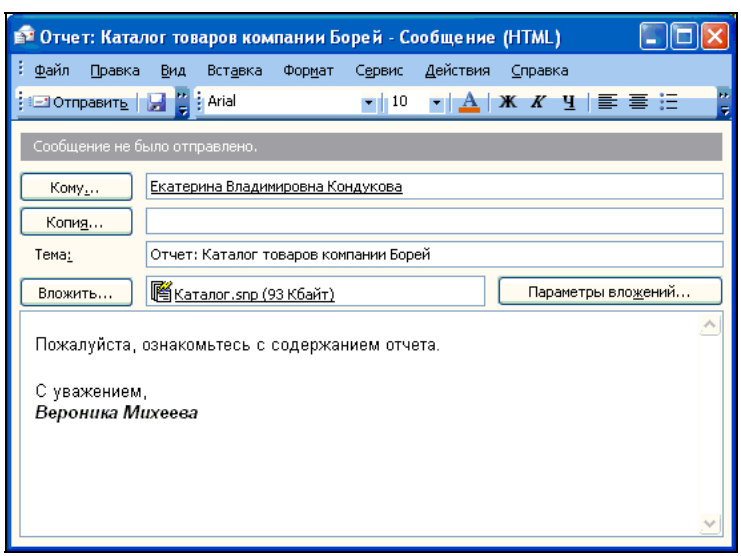


Рис. 6.15. Отправка снимка отчета по электронной почте

Создав файл снимка отчета, вы можете сразу поместить его в папку на Web-узле и, установив ссылки на него на Web-страницах, предоставить доступ к этому снимку

отчета всем пользователям корпоративной сети или Интернета с помощью программы просмотра Internet Explorer.

Для просмотра снимков отчетов можно также использовать элемент ActiveX — файл Snapview.ocx, входящий в состав дистрибутива пакета Microsoft Office 2003. Например, чтобы использовать этот ActiveX в качестве элемента управления в каком-либо объекте представления данных в Access (форме, отчете, странице доступа к данным), требуется включить ссылку на него. Для этого в редакторе VBA выберите команду **Сервис | Ссылки** (Tools | References) и в появившемся диалоговом окне установите флажок напротив элемента списка **Snapshot Viewer Control** (Элемент "Просмотр снимков").

Режимы работы с отчетами

Работа с отчетами Access может происходить в трех режимах: Просмотр образца, Предварительный просмотр и в режиме Конструктора. Переключиться из одного режима в другой можно либо с помощью команды **Образец** (Layout Preview), **Предварительный просмотр** (Print Preview) или **Конструктор** (Design View) в меню **Вид** (View) в любом режиме работы с отчетом, либо с помощью кнопки **Вид** (View) панели инструментов.

Режим Предварительного просмотра предназначен для просмотра отчета Access в том виде, в котором он будет выведен на печать. Отчет, открытый в режиме Предварительного просмотра, можно просмотреть в увеличенном или уменьшенном масштабе, отобразить на экране одновременно одну или несколько страниц, а также напечатать, предварительно определив параметры страницы. По умолчанию из окна базы данных (при двойном щелчке по отчету или нажатии кнопки **Открыть** (Preview) в окне базы данных) отчет открывается именно в этом режиме. Если отчет был открыт в другом режиме, то для перехода в режим Предварительного просмотра выберите команду **Вид | Предварительный просмотр** (View | Print Preview) или нажмите на стрелку, расположенную справа от кнопки **Вид** (View) на панели инструментов **Конструктор отчетов** (Report Design), и в открывшемся списке выберите элемент **Предварительный просмотр** (Print Preview).

Режим Просмотра образца является удобным средством проверки внешнего вида и структуры (макета) создаваемого отчета. В этом режиме в окно просмотра выводятся только те данные, которые необходимы для заполнения каждого из элементов макета отчета. Для просмотра всех данных, выводящихся в отчете, следует использовать режим Предварительного просмотра. Чтобы перейти в режим Просмотра образца, выберите команду **Вид | Образец** (View | Layout Preview) или нажмите на стрелку, расположенную справа от кнопки **Вид** (View) на панели инструментов **Конструктор отчетов** (Report Design), и в открывшемся списке выберите элемент **Образец** (Layout Preview).

В режиме Конструктора можно разрабатывать отчеты, модифицировать их структуру, изменять внешний вид, добавлять и удалять поля. Этим режимом обычно пользуются разработчики отчетов Access. Чтобы перейти в режим Конструктора, выберите команду **Вид | Конструктор** (View | Design View) или нажмите на стрелку, расположенную справа от кнопки **Вид** (View), панели инструментов, и в открывшемся

списке выберите элемент **Конструктор** (Design View). На рис. 6.16 показан отчет, открытый в режиме Конструктора. Его создание описано в *разд. "Создание отчета с помощью Мастера" этой главы.*

Замечание

Создать новый отчет можно "с нуля", начиная его разработку сразу в режиме Конструктора, не пользуясь средствами автоматического создания отчетов и Мастером отчетов Access. Для этого достаточно дважды щелкнуть по ярлыку **Создание отчета в режиме конструктора** (Create report in Design view) над списком отчетов в окне базы данных.

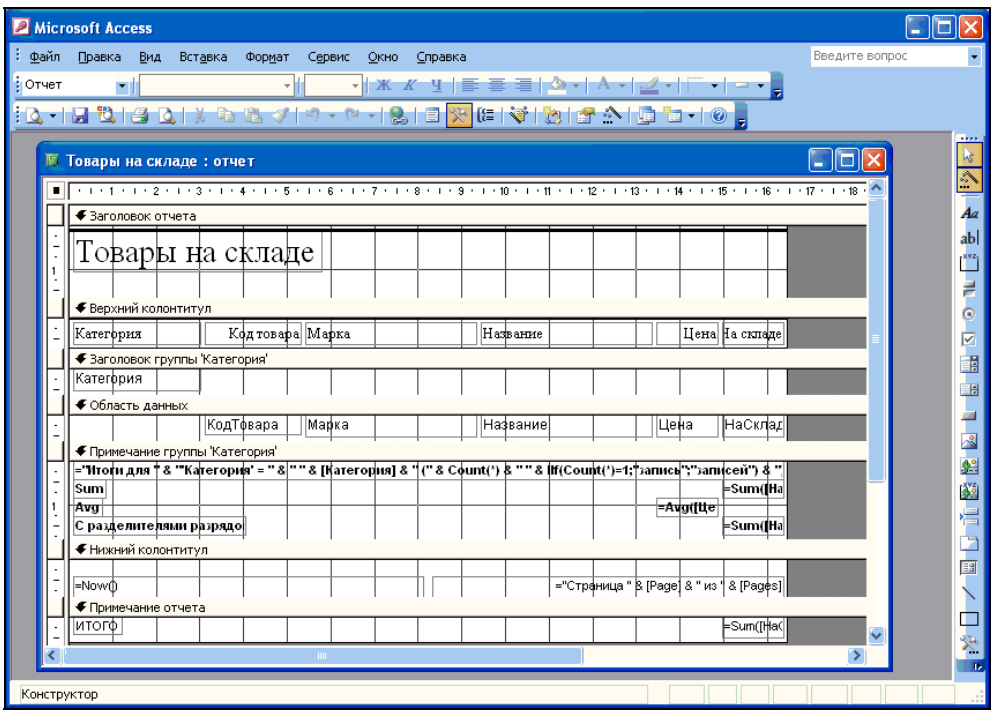


Рис. 6.16. Отчет `Товары на складе` в режиме Конструктора

Режим Предварительного просмотра

Если в окне **База данных** (Database) дважды щелкнуть левой кнопкой мыши по имени существующего отчета, то на экране появится отчет в режиме Предварительного просмотра. На рис. 6.17 изображена панель инструментов **Предварительный просмотр** (Print Preview), а в табл. 6.1 перечислены кнопки этой панели инструментов и соответствующие им команды меню в режиме Предварительного просмотра.



Рис. 6.17. Панель инструментов **Предварительный просмотр**

Таблица 6.1. *Панель инструментов* ***Предварительный просмотр***

Кнопка	Описание	Команда меню
Вид (View)	Переключение между тремя режимами работы с отчетом	**Вид** | **Конструктор** (View | Design View), **Вид** | **Предварительный просмотр** (View | Print Preview), **Вид** | **Образец** (View | Layout Preview)
Печать (факс) (Print (Fax))	Печать отчета или отправка его по факсу без отображения диалогового окна **Печать** (Print). Печать отчета выполняется с использованием текущих параметров принтера	
Масштаб (Zoom)	Обеспечивает переключение между полностраничным режимом просмотра и режимом просмотра в натуральную величину. Щелчок левой кнопкой мыши, если ее указатель принял форму увеличительного стекла, приведет к тем же результатам	**Вид** | **Масштаб** | **100%** (View | Zoom | Zoom 100%), **Вид** | **Масштаб** | **Вписать в окно** (View | Zoom | Fit To Window)
Одна страница (One Page)	Отобразить на экране одну страницу	**Вид** | **Число страниц** | **Одна страница** (View | Pages | One Page)
Две страницы (Two Pages)	Отобразить на экране две страницы	**Вид** | **Число страниц** | **Две страницы** (View | Pages | Two Pages)
Несколько страниц (Multiple Pages)	Отобразить на экране несколько страниц	**Вид** | **Число страниц** | **Четыре страницы** (View | Pages | Four Pages) или **Восемь страниц** (Eight Pages) или **Двенадцать страниц** (Twelve Pages)
Список **Масштаб** (Zoom)	Изменяет масштаб изображения от 10 до 1000 процентов, либо обеспечивает масштабирование отчета по размеру окна	**Вид** | **Масштаб** (View | Zoom)
Закрыть (Close)	Закрывает окно предварительного просмотра и обеспечивает возврат в режим Конструктора отчета или в окно **База данных** (Database)	**Файл** | **Предварительный просмотр** (File | Print Preview)

Таблица 6.1 (окончание)

Кнопка	Описание	Команда меню
Установка (Setup)	Отображение диалогового окна **Параметры страницы** (Page Setup)	**Файл	Параметры страницы** (File | Page Setup)
Связи с Office (Office Links)	Выводит раскрывающийся список команд для связи с программами Microsoft Office: **Публикация в Microsoft Office Word** (Publish It with Microsoft Word) и **Анализ в Microsoft Office Excel** (Analyze It with Microsoft Excel). (Команда **Слияние в Microsoft Office Word** (Merge It with Microsoft Word) для отчетов не доступна)	**Сервис	Связи с Office** (Tools | Office Links)
Окно базы данных (Database Window)	Открывает окно базы данных	**Окно	1** (Window | 1)
Новый объект (New Object)	Создает новый объект. Щелкните по стрелке справа от этой кнопки, появится раскрывающийся список объектов, которые можно создать	
Справка по Microsoft Office Access (Microsoft Access Help)	Вызов помощника и получение справки Access	**Справка	Справка по Microsoft Access** (Help | Microsoft Access Help)

Режим Просмотра образца

Режим Просмотра образца применяется при разработке отчета для быстрого просмотра полученной структуры отчета. В этом режиме в отчете отображается лишь часть данных, что значительно ускоряет процесс загрузки данных в отчет для просмотра.

Если отчет открыт в режиме Конструктора, то, чтобы перейти в режим Просмотра образца, на панели инструментов **Конструктор отчетов** (Report Design) нажмите стрелку справа от кнопки **Вид** (View) и в раскрывшемся списке выделите элемент **Образец** (Layout Preview). Чтобы вернуться в режим Конструктора и продолжить правку отчета, нажмите кнопку **Вид**, отображенную теперь на панели инструментов **Предварительный просмотр** (Print Preview).

Режим Конструктора

Разработку структуры отчета, изменение его внешнего вида, добавление и удаление элементов можно производить только в режиме Конструктора отчетов. Панель эле-

ментов (Toolbox), отображаемая в этом режиме, позволяет создавать элементы управления. При этом некоторые элементы управления можно создавать с помощью Мастера элементов, осуществляющего пошаговое руководство процессом создания.

К создаваемому отчету можно применить один из предлагаемых Access автоформатов, позволяющих быстро настроить внешний вид отчета. Более того, Access позволяет создавать свои собственные автоформаты и применять их к другим отчетам.

Чтобы открыть отчет в режиме Конструктора, выделите нужный отчет в окне базы данных и выберите команду **Вид | Конструктор** (View | Design View) или нажмите кнопку **Конструктор** (Design) в окне базы данных.

Структура отчета

Отчет состоит из нескольких частей, называемых *разделами*. Рассмотрим их назначение. Разделы (sections) отчета включают заголовок, примечание, область данных, а также верхний и нижний колонтитулы страниц отчета. Заголовок и примечание отчета, верхний и нижний колонтитулы страницы не являются обязательными элементами структуры отчета. Эти разделы можно добавить или удалить из отчета с помощью соответствующей команды меню **Вид** (View) в режиме Конструктора.

Раздел заголовка отчета определяет высоту области заголовка отчета, которая может содержать текст, графику и другие элементы управления, которые будут выводиться в верхней части отчета. В многостраничном отчете раздел заголовка отображается только на первой странице.

Раздел области данных предназначен для отображения записей источника данных отчета. Этот раздел может содержать элементы управления для отображения данных из каждой записи в источнике — таблице и запросе.

Раздел примечаний отчета определяет высоту области примечания отчета и добавляется вместе с разделом заголовка отчета. При печати многостраничного отчета примечание будет отображено только внизу последней страницы.

Для добавления областей заголовка и примечания отчета выберите команду **Вид | Заголовок/примечание отчета** (View | Report Header/Footer). Если отчет уже содержит эти области, то команда меню **Заголовок/примечание отчета** (Report Header/Footer) будет отмечена флажком. В этом случае команда **Вид | Заголовок/примечание отчета** (View | Report Header/Footer) позволяет удалить области заголовка и примечания из отчета.

Замечание

При удалении областей заголовка или примечания отчета, содержащих текст либо другие элементы управления, Access выводит диалоговое окно, предупреждающее об удалении содержимого заголовка и примечания.

При печати многостраничного отчета содержимое разделов верхнего и нижнего колонтитулов страниц отчета выводится на каждой странице. Чтобы добавить к отчету верхний и нижний колонтитулы, выберите команду **Вид | Колонтитулы** (View | Page Header/Footer). Если отчет уже содержит эти области, то команда меню **Колонтитулы** (Page Header/Footer) будет отмечена флажком. Для удаления областей верхнего и нижнего колонтитулов страниц отчета снова выберите команду **Вид | Колонтитулы**

(View | Page Header/Footer). Если колонтитулы содержат какие-то элементы, Access отобразит предупреждение о том, что в случае удаления колонтитулов будет удалено и их содержимое.

Поскольку отчеты, как правило, включают группировку записей и вывод промежуточных итогов, в них могут присутствовать дополнительные разделы: **Заголовок группы** (Group Header) и **Примечание группы** (Group Footer). В зависимости от количества уровней группировки в отчете может быть разное количество этих разделов.

Изменение размеров отчета и его разделов

Чтобы изменить горизонтальный размер отчета, т. е. его ширину, с помощью мыши перетащите правую границу отчета влево или вправо. При этом горизонтальная линейка, расположенная в верхней части окна Конструктора отчетов, поможет установить точную ширину отчета.

Чтобы изменить вертикальные размеры области какого-либо раздела отчета, поместите указатель мыши на нижнюю границу области так, чтобы он принял форму черты с двумя вертикальными стрелками, и перетащите его вверх или вниз. В этом случае вертикальная линейка, расположенная в левой части окна Конструктора отчетов, поможет установить точную высоту области раздела.

Вертикальный размер отчета определяется вертикальными размерами всех разделов отчета.

Чтобы настроить сразу оба размера раздела отчета — ширину и высоту — подведите указатель мыши к правому нижнему углу. Когда он примет форму четырехсторонней стрелки, нажмите левую кнопку мыши и, не отпуская ее, растяните рамку раздела до желаемых размеров.

Инструментальные средства режима Конструктора

Окно Конструктора отчетов состоит из нескольких основных элементов.

- Панель инструментов **Конструктор отчетов** (Report Design). Кнопки этой панели дублируют некоторые команды меню режима Конструктора отчетов. Функции кнопок и соответствующие им команды меню описаны в табл. 6.2.
- Панель инструментов **Формат (форма/отчет)** (Formatting (Form/Report)). Кнопки этой панели облегчают форматирование элементов управления (например, выбор цвета и стиля рамок элементов управления или параметров оформления текста).
- Панель инструментов **Панель элементов** (Toolbox). С ее помощью в отчет добавляются элементы управления (на рис. 6.16 она расположена вдоль правой вертикальной границы отчета).
- Вертикальная и горизонтальная линейки позволяют производить точную разметку отчета и его разделов.
- Крайняя правая вертикальная линия (см. рис. 6.16) позволяет определить ширину отчета — расстояние до его правой границы.
- Крайняя нижняя горизонтальная линия (см. рис. 6.16) позволяет определить высоту нижнего поля отчета.

- С помощью вертикальной и горизонтальной полос прокрутки можно просматривать части отчета, находящиеся за границами области окна Конструктора отчетов.

Панель инструментов *Конструктор отчетов*

На рис. 6.18 изображена панель инструментов **Конструктор отчетов** (Report Design), а в табл. 6.2 приведены описания кнопок этой панели инструментов и соответствующие им команды меню.



Рис. 6.18. Панель инструментов **Конструктор отчетов**

Таблица 6.2. *Панель инструментов* ***Конструктор отчетов***

Кнопка	Описание	Команда меню
Вид (View)	Переключение в другие режимы работы с отчетом. Чтобы открыть отчет в каком-либо другом режиме, нажмите стрелку справа от кнопки и в появившемся списке выделите элемент, соответствующий необходимому режиму	**Вид** | **Предварительный просмотр** (View | Print Preview), **Вид** | **Образец** (View | Layout Preview)
Сохранить (Save)	Сохранение открытого отчета	**Файл** | **Сохранить** (File | Save)
Поиск файлов (Search)	Отображение панели поиска документов Office	**Файл** | **Поиск файлов** (File | Search)
Печать (факс) (Print (Fax))	Печать или отправка по факсу отчета без открытия диалогового окна **Печать** (Print)	
Предварительный просмотр (Print Preview)	Предварительный просмотр отчета перед печатью. Отчет может быть напечатан из окна предварительного просмотра	**Файл** | **Предварительный просмотр** (File | Print Preview)
Вырезать (Cut)	Удаление выделенных объектов из отчета в Буфер обмена Windows	**Правка** | **Вырезать** (Edit | Cut)
Копировать (Copy)	Копирование выделенных объектов отчета в Буфер обмена	**Правка** | **Копировать** (Edit | Copy)
Вставить (Paste)	Вставка содержимого Буфера обмена в отчет	**Правка** | **Вставить** (Edit | Paste)
Формат по образцу (Format Painter)	Копирование параметров форматирования из одного выделенного объекта в другой	

Таблица 6.2 (окончание)

Кнопка	Описание	Команда меню
Отменить (Undo)	Отмена последнего изменения отчета	**Правка	Отменить** (Edit | Undo)
Вернуть (Redo)	Повторить отмененное действие	**Правка	Вернуть** (Edit | Redo)
Добавить гиперссылку (Insert Hyperlink)	Вставка в отчет гиперссылки	**Вставка	Гиперссылка** (Insert | Hyperlink)
Список полей (Field List)	Отображение или скрытие списка полей, содержащихся в источнике данных (запросе или таблице) отчета	**Вид	Список полей** (View | Field List)
Панель элементов (Toolbox)	Отображение или скрытие панели элементов	**Вид	Панель элементов** (View | Toolbox)
Сортировка и группировка (Sorting and Grouping)	Выводит диалоговое окно **Сортировка и группировка** (Sorting and Grouping), в котором можно задать структуру отчета и порядок отображения данных в нем	**Вид	Сортировка и группировка** (View | Sorting and Grouping)
Автоформат (AutoFormat)	Выбор заранее определенных форматов для отчетов, включая такие характеристики, как фоновый рисунок для отчета и шрифты для элементов управления	**Формат	Автоформат** (Format | AutoFormat)
Программа (Code)	Открытие модуля, содержащего процедуры обработки событий отчета	**Вид	Программа** (View | Code)
Свойства (Properties)	Открытие окон свойств выделенных объектов — отчета, раздела (например, верхнего или нижнего колонтитула), а также элементов управления	**Вид	Свойства** (View | Properties)
Построить (Build)	Вызов Построителя для выделенного элемента или свойства в отчете. Access активизирует данную кнопку только в том случае, если для выделенного элемента или свойства имеется Построитель	
Окно базы данных (Database Window)	Отображение окна базы данных	**Окно	1** (Window | 1)
Новый объект (New Object)	Создание нового объекта базы данных. Чтобы выбрать тип создаваемого объекта, нажмите стрелку справа от кнопки	
Справка: Microsoft Office Access (Microsoft Access Help)	Вызов помощника и получение справки Access 2003	**Справка	Справка: Microsoft Office Access** (Help | Microsoft Access Help)

Форматирование отчета

Как уже упоминалось ранее, одним из элементов окна Конструктора отчетов является панель инструментов **Формат (форма/отчет)** (Formatting (Form/Report)), содержащая кнопки и раскрывающиеся списки, облегчающие выбор параметров форматирования (цвета, шрифты, толщины линий и т. п.).

Функции кнопок этой панели инструментов, используемых для форматирования текста, аналогичны функциям кнопок, которые можно увидеть на панелях инструментов Microsoft Word. С помощью раскрывающегося списка **Объект** (Object) можно быстро выделить любой элемент отчета, выбрав его имя в списке. В раскрывающемся списке **Шрифт** (Font) можно выбрать любой из установленных на компьютере шрифтов. А раскрывающийся список **Размер** (Font Size) позволяет выбрать нужный размер шрифта или ввести его значение (в пунктах).

Эта же панель инструментов используется при форматировании текста элементов управления в отчете. Названия и функции кнопок, предназначенных для форматирования текста элементов управления, а также значение соответствующих свойств элемента управления подробно описаны в *разд. "Панель инструментов Формат (форма/отчет)", гл. 5.*

Кнопка **Автоформат** (AutoFormat) панели инструментов **Конструктор отчетов** (Report Design) предоставляет те же возможности по форматированию, что и для форм. Подробное описание использования автоформата, способов изменения предопределенных стилей автоформата или создания собственных стилей пользователя содержится *в разд. "Применение автоформата", гл. 5.*

Замечание

Поскольку Access хранит стили отчетов и форм по отдельности, то для отчетов придется создать отдельные стили автоформата.

Панель элементов

Панель инструментов, появляющаяся при открытии отчета в режиме Конструктора, содержащая кнопки с символическими изображениями элементов управления, называется **Панель элементов** (Toolbox). Эта та же самая панель, которая используется в режиме Конструктора форм, и, как и в случае с формой, она позволяет размещать в отчете новые элементы управления. Чтобы закрыть Панель элементов (Toolbox), нажмите кнопку **Панель элементов** (Toolbox) на панели инструментов **Конструктор отчетов** (Report Design) или кнопку **Закрыть** (Close) в правом верхнем углу Панели элементов.

Замечание

При переходе в режим Конструктора Access автоматически отображает Панель элементов. Если ее закрыть, то в следующий раз при переходе в режим Конструктора Access не будет выводить ее. В этом случае для отображения Панели элементов нажмите кнопку **Панель элементов** (Toolbox) на панели инструментов **Конструктор отчетов** (Report Design) или выберите команду **Вид | Панели инструментов** (View | Toolbars), а затем установите флажок **Панель элементов** (Toolbox).

Выделение отчета и его элементов

Прежде чем перейти к изменению свойств отчета или его элементов, необходимо научиться их выделять. Выделение объектов производится с помощью мыши.

Выделить весь отчет или его элемент можно щелчком левой кнопки мыши по определенному месту в окне Конструктора. Назовем такие места соответственно *областями выделения* всего отчета или его элемента.

Областью выделения всего отчета можно назвать квадратик, расположенный в месте пересечения вертикальной и горизонтальной линеек в верхней и левой боковой части отчета в режиме Конструктора, а еще области за границами отчета, но внутри окна Конструктора отчета, т. к. щелчком левой кнопки по этим областям выполняется выделение отчета.

Итак, чтобы выделить весь отчет, щелкните левой кнопкой мыши по области вне границ отчета, но внутри окна Конструктора отчета, либо щелкните левой кнопкой мыши по квадратику в верхнем левом углу окна Конструктора отчета, где пересекаются вертикальная и горизонтальная линейки; либо выберите команду **Правка | Выделить отчет** (Edit | Select Report).

Замечание

Другой способ выделить отчет — выбрать команду **Вид | Выделить отчет** (View | Select Report). Команда **Вид | Выделить все** (View | Select All) позволяет выделить сразу все элементы управления отчета.

Областью выделения элемента управления является все пространство внутри контура этого элемента. Щелкнув в любом месте этого пространства, вы выделяете элемент управления.

Итак, чтобы выделить элемент управления (либо элемент управления с меткой), щелкните по нему левой кнопкой мыши.

Еще одной разновидностью элементов отчета можно назвать его разделы. Областью выделения раздела отчета является любое свободное от элементов управления пространство внутри раздела или заголовок раздела, выделенный в отчете полосой серого цвета с названием заголовка.

Итак, чтобы выделить раздел заголовка отчета, примечания отчета, области данных или разделы верхнего и нижнего колонтитулов, щелкните левой кнопкой мыши по области заголовка отчета, примечания отчета, области данных, верхнего или нижнего колонтитула.

Замечание

Чтобы быстро выделить любой объект отчета или весь отчет целиком, в раскрывающемся списке **Объект** (Object) на панели инструментов **Формат (форма/отчет)** (Formatting Form/Report) выберите название нужного раздела отчета или элемента управления.

Изменение свойств отчета и его элементов

Элементами отчета являются разделы отчета и добавленные в них элементы управления. Отчет и каждый его элемент обладает целым рядом свойств, которые опреде-

ляют их внешний вид, поведение и данные, которые они отображают. Эти свойства доступны для изменения в режиме Конструктора в специальном окне свойств. Чтобы отобразить окно свойств отчета или его элемента:

1. Откройте отчет в режиме Конструктора. Для этого в окне базы данных щелкните по ярлыку **Отчеты** (Reports) и выделите в списке отчетов нужный, затем нажмите кнопку **Конструктор** (Design).
2. Выделите отчет или его элемент и нажмите кнопку **Свойства** (Properties) на панели инструментов **Конструктор отчетов** (Report Design), или выберите команду **Вид | Свойства** (View | Properties), или щелкните по области выделения всего отчета или его элемента правой кнопкой и выберите команду **Свойства** (Properties) в контекстном меню.

   Например, чтобы отобразить окно свойств отчета, щелкните по квадратику в левом верхнем углу отчета и нажмите кнопку **Свойства** (Properties) на панели инструментов **Конструктор отчетов** (Report Design).

Замечание

Чтобы быстро открыть окно свойств отчета или его элемента, дважды щелкните по области выделения отчета или его элемента.

Окно свойств отчета или какого-либо его элемента содержит пять вкладок, каждая из которых объединяет однотипные группы параметров. На последней вкладке представлены все свойства элемента или отчета.

На рис. 6.19 изображено окно свойств отчета, имеющее вкладки **Макет** (Format), **Данные** (Data), **События** (Event), **Другие** (Other) и **Все** (All).

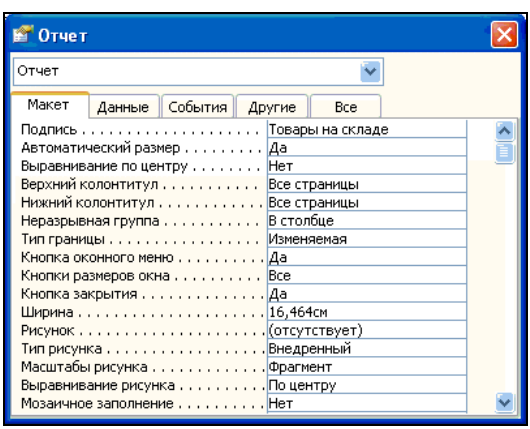


Рис. 6.19. Окно свойств отчета

Окно свойств элемента управления содержит такие же вкладки, как и окно свойств отчета, однако некоторые вкладки, например **События** (Event) и **Данные** (Data), могут оказаться пустыми. На рис. 6.20 изображено окно свойств текстового поля.

Окно свойств раздела содержит такие же вкладки, как и окно свойств отчета. На рис. 6.21 представлено окно свойств заголовка отчета.

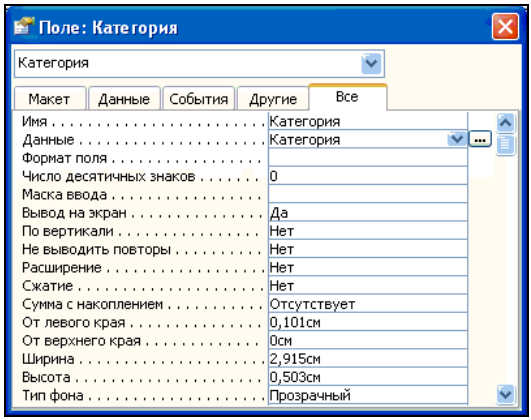


Рис. 6.20. Окно свойств текстового поля

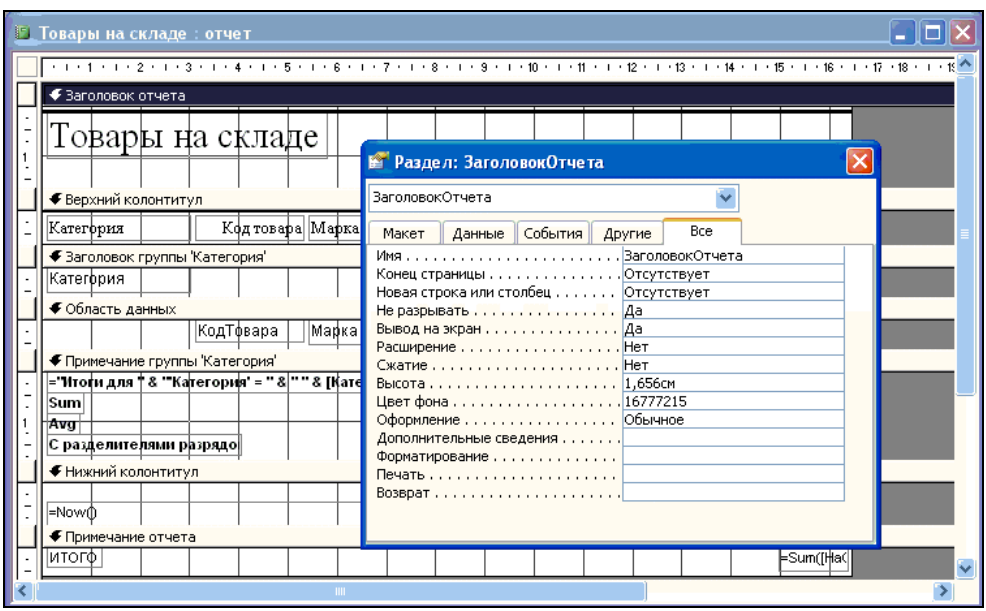


Рис. 6.21. Окно свойств заголовка отчета

Замечание

Не закрывая открытое окно свойств можно переключаться между свойствами различных элементов отчета и самого отчета, щелкая левой кнопкой мыши по соответствующим областям выделения.

На вкладке **Макет** (Format) окна свойств отчета или его элемента сгруппированы свойства, управляющие внешним видом отчета или его элемента. Например, значение свойства **Ширина** (Width) определяет ширину отчета или выделенного элемента

управления, а в поле **Подпись** (Caption) задается название, отображаемое в заголовке отчета в режиме его просмотра. На этой же вкладке располагаются поля свойств, позволяющих задать цвет, толщину и стиль рамки элемента управления, определить рисунок, используемый в качестве фона отчета, и настроить его отображение, а также многие другие параметры, отвечающие за внешний вид отчета или его элементов.

Вкладка **Данные** (Data) позволяет указать источник данных для отчета или выделенного элемента управления, а также настроить их использование.

С помощью вкладки **События** (Event) окна свойств отчета или его раздела можно определить действия, выполняемые, когда происходят определенные события, например, открытие или закрытие отчета, его активация или возникновение ошибки.

На вкладке **Другие** (Other) собраны свойства, которые нельзя отнести ни к одной из перечисленных групп свойств. Например, с помощью этой вкладки окна свойств отчета можно определить привязанные к нему файл справки и панель инструментов, способ группировки данных для отчета. Эта же вкладка в окне свойств элемента управления позволяет, например, изменить его имя или ориентацию отображаемого в нем текста.

На вкладке **Все** (All) собраны все свойства отчета или его элемента, размещенные на четырех основных вкладках окна свойств. Поскольку отчет и каждый его элемент в отдельности обладает достаточно большим количеством свойств, то список всех свойств на этой вкладке будет достаточно длинным и нахождение нужного свойства на этой вкладке может оказаться затруднительным.

Значения свойств устанавливаются либо с помощью текстовых полей, либо с помощью раскрывающихся списков, отображающихся справа от названий свойств. Например, чтобы изменить шрифт, используемый каким-либо элементом управления, щелкните левой кнопкой мыши в поле свойства **Шрифт** (Font Name), а затем щелкните по кнопке со стрелкой, появившейся на правом конце поля, и в раскрывшемся списке выберите нужный шрифт. Чтобы изменить ширину отчета, на вкладке **Макет** (Format) щелкните по полю свойства **Ширина** (Width) и введите число, соответствующее нужной ширине отчета в сантиметрах. А чтобы изменить ориентацию текста какого-либо элемента управления, на вкладке **Другие** (Other) щелкните по полю свойства **По вертикали** (Vertical), затем щелкните по кнопке со стрелкой, появившейся на правом конце поля, и в раскрывшемся списке выберите значение **Да** (Yes). Теперь текст выделенного элемента управления будет расположен по вертикали.

Некоторые свойства снабжены вспомогательным Мастером, называющимся Построителем, помогающим задать значение свойства. Кнопка вызова Построителя отображается слева от поля ввода значений свойства при перемещении в это поля курсора ввода. Чтобы воспользоваться Построителем для определения значения какого-либо свойства, щелкните по полю этого свойства, а затем нажмите кнопку с тремя точками, появившуюся справа от поля.

Кроме ручной настройки свойств форматирования, определяющих внешний вид отчета и его элементов, Access предоставляет средство автоматического форматирования, называющееся *автоформатом*. Применение автоформата позволяет очень быстро изменить параметры оформления всех элементов отчета. Это средство доступно не только для оформления отчетов, но также для форм. Использование автоформата для отчетов совершенно аналогично применению его для форм. Поэтому мы не будем повторять здесь его описание, которое уже приведено в *гл. 5*.

Создание и удаление элементов управления отчета

Access предоставляет два простых метода создания элементов управления: с помощью панели инструментов **Панель элементов** (Toolbox) и кнопки **Список полей** (Field List). Кроме того, при создании элементов управления с помощью Панели элементов можно воспользоваться Мастером, позволяющим пройти шаг за шагом процесс создания более сложных элементов управления, например, таких как списки, поля со списком, кнопки и группы элементов управления.

Создание элементов управления с помощью Панели элементов

Создание элементов управления с помощью панели элементов может происходить с использованием Мастера или без него. В данном разделе мы приведем пример создания элемента управления **Список** (List Box) этими двумя способами.

Для добавления списка в какой-либо раздел отчета без помощи Мастера:

1. Убедитесь в том, что на панели элементов не нажата кнопка **Мастера** (Control Wizards).
2. На панели элементов нажмите кнопку **Список** (List Box). Когда указатель мыши попадет в область отчета, он примет вид крестика со значком элемента управления **Список** (List Box). Центр крестика определяет позицию левого верхнего угла создаваемого элемента управления.
3. Поместите указатель мыши в виде крестика в область нужного раздела отчета. Нажмите левую кнопку мыши и, удерживая ее, переместите указатель мыши в ту точку, где должен расположиться нижний правый угол списка, растянув рамку воображаемого нового элемента.
4. При этом в строке состояния будет отображаться число строк и количество символов для текущего шрифта, которые может вместить список текущего размера.
5. При достижении элементом управления **Список** (List Box) нужных размеров отпустите левую кнопку мыши. Если список будет иметь размеры, превышающие область того раздела, к которому он добавляется, то область раздела отчета будет автоматически увеличена, чтобы поместить список целиком.
6. На некотором расстоянии слева от поля списка появится его метка. Чтобы изменить название метки, щелкните внутри нее левой кнопкой мыши. Внутри метки появится текстовый курсор. Введите нужный текст и нажмите клавишу <Enter>.

Теперь для определения свойств созданного списка нужно открыть окно его свойств и сделать необходимые изменения.

После добавления элемента управления для его перемещения и изменения размеров используются верхний угловой маркер перемещения и маркеры изменения размеров. Положение левого верхнего угла элемента определяется значениями свойств элемента управления **От левого края** (Left) и **От верхнего края** (Top). Перемещая мышью маркеры изменения размеров, вы изменяете значения свойств элемента управления **Ширина** (Width) и **Высота** (Height).

Добавление в форму большинства других типов элементов управления (выключателей, переключателей, флажков) происходит аналогично.

Если кнопка **Мастера** (Control Wizards) на Панели элементов нажата, то некоторые кнопки, например **Список** (List Box) или **Поле со списком** (Combo Box), при создании соответствующих элементов вызывают Мастер элементов.

Чтобы создать элемент управления **Список** (List Box), содержащий фиксированный набор значений с помощью Мастера:

1. На Панели элементов нажмите кнопку **Мастера** (Control Wizards), если она еще не нажата.
2. На Панели элементов нажмите кнопку **Список** (List Box). Когда указатель мыши попадет в область отчета, он примет вид крестика со значком элемента управления **Список** (List Box). Центр крестика определяет позицию левого верхнего угла создаваемого элемента управления.
3. Поместите указатель мыши в виде крестика в область нужного раздела отчета. Нажмите левую кнопку мыши и, удерживая ее, переместите указатель мыши в ту точку, где должен находиться нижний правый угол списка, растянув рамку воображаемого нового элемента. При этом в строке состояния будет отображаться число строк и количество символов для текущего шрифта, которые может вместить список текущего размера.
4. По достижении элементом управления **Список** (List Box) нужных размеров отпустите левую кнопку мыши. Появится первое диалоговое окно Мастера. В этом диалоговом окне выберите переключатель **Будет введен фиксированный набор значений** (I will type in the values that I want). Нажмите кнопку **Далее** (Next). Появится второе диалоговое окно Мастера списков (рис. 6.22).

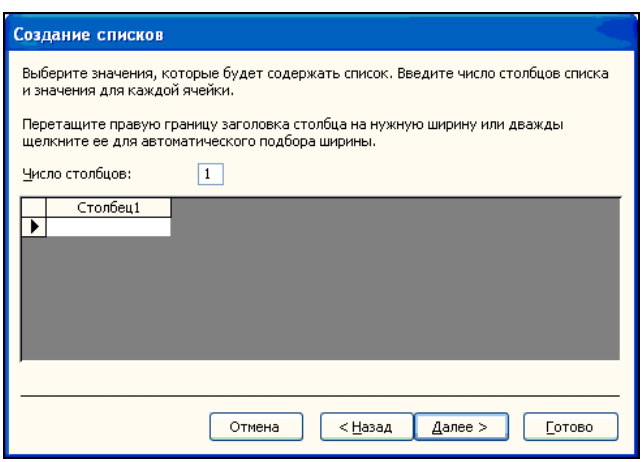


Рис. 6.22. Второе диалоговое окно Мастера списков

5. Во втором диалоговом окне Мастера списков можно указать количество столбцов создаваемого списка, а также определить их ширину. Введите число столбцов в поле ввода и, при необходимости, переместите правые границы столбцов с помощью мыши, если хотите изменить их ширину. В таблицу, изображенную внутри диалогового окна, введите желаемые значения элементов списка. Выполнив

все необходимые действия, нажмите кнопку **Далее** (Next). Появится третье диалоговое окно Мастера (рис. 6.23).

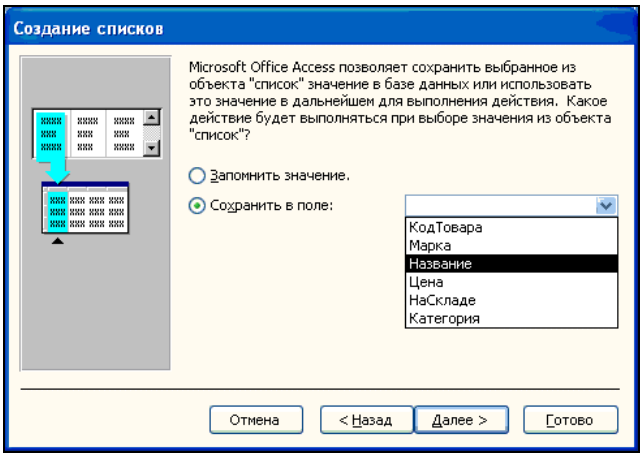


Рис. 6.23. Третье диалоговое окно Мастера списков

6. В этом диалоговом окне нужно определить, что предполагается делать со значением, выбранным в списке пользователем: запомнить его для дальнейшего использования или сохранить в каком-либо поле источника данных. Выберите нужный вариант (если требуется сохранить значение в поле источника данных, выделите элемент с его именем в раскрывающемся списке). Нажмите кнопку **Далее** (Next). Появится четвертое диалоговое окно Мастера (рис. 6.24).

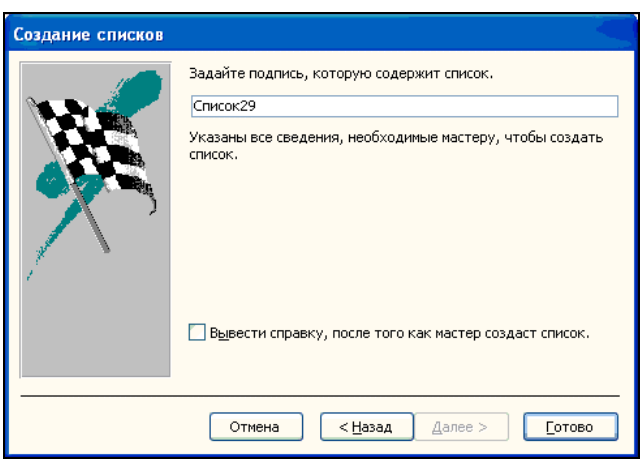


Рис. 6.24. Четвертое диалоговое окно Мастера списков

7. В этом диалоговом окне требуется указать название, которое будет использовано в качестве метки для создаваемого списка. Чтобы после создания списка полу-

чить справку по возможностям его настройки, установите флажок, расположенный внизу диалогового окна.

8. Для завершения процесса создания списка нажмите кнопку **Готово** (Finish).

Создание элементов управления с помощью кнопки *Список полей*

Не менее простым и удобным способом создания элементов управления является использование списка полей. Чтобы открыть список полей источника данных, доступных для создаваемого отчета, нажмите кнопку **Список полей** (Field List) на панели инструментов **Конструктор отчетов** (Report Design). Появится плавающее окно, содержащее поля из выбранного для формы источника данных. На рис. 6.25 изображен список полей запроса, который был использован в качестве источника данных для отчета `Товары на складе`.

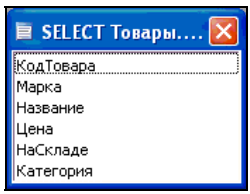


Рис. 6.25. Список полей в режиме Конструктора отчета

Чтобы создать элемент управления, связанный с каким-либо полем источника данных для создаваемого отчета (таблицы или запроса):

1. На панели инструментов **Конструктор отчетов** (Report Design) нажмите кнопку **Список полей** (Field List).
2. В открывшемся окне выделите нужное поле и с помощью мыши перетащите его в область одного из разделов отчета.
3. В отчете появится текстовое поле с меткой, содержащей название поля источника данных. Теперь его можно перемещать и форматировать стандартными методами, используемыми для настройки элементов управления.

Удаление элементов отчета

Чтобы удалить из отчета какой-либо элемент управления:

1. Выделите элемент управления, щелкнув по нему левой кнопкой мыши.
2. Нажмите клавишу <Delete> или выберите команду **Правка | Удалить** (Edit | Delete).

Чтобы удалить сразу несколько элементов управления, выделите их с помощью мыши, удерживая нажатой клавишу <Shift>, и нажмите клавишу <Delete> или выберите команду **Правка | Удалить** (Edit | Delete).

Изменение размеров и перемещение элементов управления

Access предоставляет достаточно удобный и наглядный способ настройки внешнего вида созданных элементов управления с помощью мыши. Вы можете изменять раз-

меры, перемещать, выравнивать как каждый элемент управления в отдельности, так и группы элементов управления.

Перемещение и изменение размера отдельного элемента управления

Как уже описывалось ранее, выделить отдельный элемент управления можно с помощью мыши или раскрывающегося списка **Объект** (Object) на панели инструментов **Формат (форма/отчет)** (Formatting Form/Report). При выделении элемента управления вокруг него появляется рамка с маркером перемещения (черный квадратик в левом верхнем углу) и пятью маркерами изменения размеров (рис. 6.26).

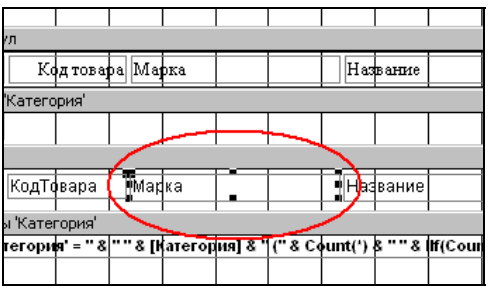


Рис. 6.26. Выделенный элемент управления `Марка` в отчете `Товары на складе` (см. рис. 6.16)

Перемещать и изменять размеры элементов управления можно различными способами.

- *Чтобы переместить элемент управления вместе с его меткой*, поместите указатель мыши в любую точку на границе выделенного элемента управления, отличную от маркеров изменения размеров и маркета перемещения. Когда указатель мыши превратится в инструмент перемещения, нажмите левую кнопку мыши и, не отпуская ее, перетащите элемент управления на новое место. При перетаскивании элемента управления его граница перемещается одновременно с указателем мыши. Когда элемент управления окажется на нужном месте, отпустите левую кнопку мыши.
- *Чтобы переместить элемент управления, не перемещая связанную с ним метку*, поместите указатель мыши над маркером перемещения, находящимся в левом верхнем углу выделенного элемента управления. Указатель мыши превратится в инструмент перемещения отдельного элемента управления. Нажмите левую кнопку мыши и, удерживая ее, перетащите элемент управления на новое место, а затем отпустите левую кнопку мыши.
- *Чтобы изменить одновременно ширину и высоту элемента управления*, перемещайте указатель мыши над одним из трех угловых маркеров изменения размера выделенного элемента управления, пока указатель мыши не превратится в наклонную двустороннюю стрелку. Нажмите левую кнопку мышки и, удерживая ее, перетащите границу элемента управления до достижения нужного размера, а затем отпустите левую кнопку мыши.

- *Чтобы изменить высоту элемента управления*, перемещайте указатель мыши над верхним или нижним маркером изменения размера выделенного элемента управления, пока указатель мыши не превратится в двустороннюю вертикальную стрелку. Нажмите левую кнопку мыши и, удерживая ее, перетащите границу элемента управления до достижения нужного размера, а затем отпустите левую кнопку мыши.
- *Чтобы изменить ширину элемента управления*, перемещайте указатель мыши над правым или левым маркером изменения размера выделенного элемента управления, пока указатель мыши не превратится в двустороннюю горизонтальную стрелку. Нажмите левую кнопку мыши и, удерживая ее, перетащите границу элемента управления до достижения нужного размера, а затем отпустите левую кнопку мыши.

Выравнивание элемента управления по сетке

В окне Конструктора отчетов может быть отображена сетка, состоящая из линий толщиной в один пиксел, которые проводятся через каждое деление горизонтальной и вертикальной линеек окна Конструктора отчетов (т. е. через сантиметр). Чтобы отобразить сетку, выберите команду **Вид | Сетка** (View | Grid). Соответствующая команда меню будет отмечена галочкой. Чтобы отменить отображение сетки, снова выберите команду **Вид | Сетка** (View | Grid). Отметка с команды меню **Сетка** (Grid) будет снята.

С помощью сетки можно выравнивать элементы управления по горизонтали или по вертикали. Элементы управления можно "привязать к сетке", выбрав команду **Формат | Привязать к сетке** (Format | Snap to Grid). Если привязка к сетке включена, то пункт меню **Привязать к сетке** (Snap to Grid) отмечен галочкой. Если перемещать элемент управления при включенной привязке к сетке, то левый верхний угол объекта перемещается дискретно по узлам сетки.

По умолчанию сетка отображается во всех новых отчетах и ее густота равна 10 узлам на сантиметр. Чтобы изменить густоту узлов сетки:

1. Выберите команду **Правка | Выделить отчет** (Edit | Select Report).
2. На панели инструментов **Конструктор форм** (Report Design) нажмите кнопку **Свойства** (Properties). Появится окно свойств отчета.
3. В окне свойств отчета раскройте вкладку **Макет** (Format) и при помощи полосы прокрутки найдите свойства **Число делений по X** (Grid X) и **Число делений по Y** (Grid Y).
4. Измените значение свойств **Число делений по X** (Grid X) и **Число делений по Y** (Grid Y).

Выделение, перемещение и изменение свойств нескольких элементов управления

Чтобы одновременно выделить несколько элементов управления, щелкните левой кнопкой мыши по одному элементу управления, чтобы выделить его, и, удерживая нажатой клавишу <Shift>, выделите остальные элементы управления, либо при помощи мыши нарисуйте область выделения вокруг группы элементов управления, и она окажется выделенной.

Замечание

Область выделения позволяет выделить все элементы управления, которые хотя бы частично попадают в нее. Однако этот способ выделения, используемый Access по умолчанию, можно изменить для того, чтобы выделять объект только в том случае, если он целиком попадает в область выделения. Для этого выберите команду **Сервис | Параметры** (Tools | Options), раскройте вкладку **Формы и отчеты** (Forms/Reports) и выберите переключатель **охват** (Fully Enclosed).

Выделив несколько элементов одновременно, можно сразу настроить их общие свойства. Для этого отобразите диалоговое окно свойств, нажав кнопку **Свойства** (Properties) на панели инструментов **Конструктор отчетов** (Report Design).

Чтобы отменить выделение одного из элементов управления в выделенной группе, нажмите клавишу <Shift> и, удерживая ее нажатой, щелкните левой кнопкой мыши по элементу управления. Для отмены выделения всей группы элементов управления щелкните левой кнопкой мыши по любой пустой (не содержащей объектов) области формы.

Переместить выделенную группу элементов управления можно, зацепив указателем мыши за границу одного из них и перетащив всю группу на новое место.

Выравнивание нескольких элементов управления

Высоту всех текстовых полей можно одновременно отрегулировать таким образом, чтобы их параметры соответствовали использующемуся шрифту. Чтобы выделить все элементы управления в отчете, выберите команду **Правка | Выделить все** (Edit | Select All), а затем команду **Формат | Размер | по размеру данных** (Format | Size | To Fit). Access автоматически установит приемлемую высоту всех текстовых полей.

Один или несколько элементов управления можно выровнять по сетке или относительно друг друга. Для этого выделите нужные элементы и выберите команду **Формат | Выровнять** (Format | Align), а затем выберите команду:

- **по узлам сетки** (To Grid), чтобы выровнять элементы управления по узлам сетки;
- **по левому краю** (Left), чтобы выровнять выделенные элементы управления, расположенные в столбец, так, чтобы их левые края находились на одной вертикальной линии;
- **по правому краю** (Right), чтобы выровнять выделенные элементы управления, расположенные в столбец, так, чтобы их правые края находились на одной вертикальной линии;
- **по верхнему краю** (Top), чтобы выровнять выделенные элементы управления, расположенные в строку по их верхней границе;
- **по нижнему краю** (Bottom), чтобы выровнять выделенные элементы управления, расположенные в строку по их нижним границам.

Для придания создаваемому отчету более привлекательного вида необходимо потратить некоторое время на выравнивание групп элементов управления по горизонтали и вертикали.

Изменение межстрочного интервала

Чтобы сделать отчет более компактным и уместить его на меньшем количестве листов, можно сократить межстрочные интервалы в разделах отчета.

Для этого:

1. Перейдите в режим Конструктора отчетов.
2. Выделите все надписи в верхнем колонтитуле и переместите полученную группу как можно ближе к верхнему краю раздела.
3. Щелкните по нижней линии верхнего колонтитула и переместите ее как можно ближе к нижнему краю надписей в этом разделе отчета.
4. Выделите все текстовые поля в разделе области данных и переместите эту группу к верхнему краю раздела.
5. Переместите нижнюю границу области данных как можно ближе к нижнему краю текстовых полей в этом разделе.
6. Нажмите кнопку **Вид** (View) на панели инструментов **Конструктор отчетов** (Report Design), чтобы проверить эффект от сделанных изменений в режиме Предварительного просмотра.

Замечание

Невозможно сделать межстрочный интервал меньше, чем высота самого высокого текстового поля или надписи, даже уменьшая значение свойства **Высота** (Height) в окне **Свойства** (Properties) раздела, поскольку Access игнорирует введенное значение и подставляет требуемое значение, вычисленное в зависимости от границы самого крайнего элемента с нижней стороны раздела.

7. Выберите команду **Файл | Сохранить** (File | Save), чтобы сохранить внесенные изменения.

Настройка расстояний между несколькими элементами

Возможно, в вашем отчете будут содержаться целые блоки текстовых полей, расположенных в строки и столбцы. Существует возможность быстро и точно изменить или выровнять горизонтальные и вертикальные расстояния между несколькими в ряд расположенными соседними элементами управления с помощью команд меню **Формат | Интервал по горизонтали** (Format | Horizontal Spacing) и **Формат | Интервал по вертикали** (Format | Vertical Spacing):

- **Сделать равным** (Make Equal) — сделать горизонтальное (вертикальное) расстояние между соседними горизонтально (вертикально) расположенными элементами одинаковым, равным среднему расстоянию между ними по этому направлению;
- **Увеличить** (Increase) — увеличить горизонтальное (вертикальное) расстояние между соседними горизонтально (вертикально) расположенными элементами на величину одного горизонтального (вертикального) деления сетки;
- **Уменьшить** (Decrease) — уменьшить горизонтальное (вертикальное) расстояние между соседними горизонтально (вертикально) расположенными элементами на величину одного горизонтального (вертикального) деления сетки.

Используйте эти команды, предварительно выделив группу элементов, расположенных в ряд.

Копирование элементов управления через буфер обмена

Выделенный элемент управления или группу элементов управления можно удалить или скопировать в буфер обмена, а затем вставить в отчет. В Access для работы с буфером обмена используются стандартные сочетания клавиш Windows: <Ctrl>+<X> или <Shift>+<Delete> — для удаления из отчета и помещения в буфер обмена, <Ctrl>+<C> или <Ctrl>+<Insert> — для копирования выделенных элементов управления в буфер обмена и <Ctrl>+<V> или <Shift>+<Insert> — для вставки содержимого буфера обмена. Чтобы безвозвратно удалить элемент управления, выделите его и нажмите клавишу <Delete>. Чтобы восстановить удаленную метку, связанную с элементом управления, выделите другую метку, скопируйте ее в буфер обмена, затем выделите элемент управления, с которым нужно связать метку, и вставьте ее из буфера обмена.

Копирование формата элемента управления

В Access существует возможность скопировать параметры форматирования из одного элемента управления отчета в другой, при этом копируются только те элементы форматирования, применение которых возможно для данного элемента управления. Чтобы скопировать параметры форматирования:

1. Выделите элемент управления, формат которого нужно скопировать.
2. На панели инструментов **Конструктор отчетов** (Report Design) нажмите кнопку **Формат по образцу** (Format Painter). Указатель мыши превратится в инструмент копирования формата. Если необходимо скопировать параметры форматирования выделенного элемента управления в несколько элементов управления, дважды щелкните левой кнопкой мыши по кнопке **Формат по образцу** (Format Painter).
3. Щелкните левой кнопкой мыши по элементу управления, в который нужно скопировать формат. Скопированы будут только те элементы форматирования, применение которых возможно для данного элемента управления.
4. Если кнопка **Формат по образцу** (Format Painter) была нажата одним щелчком, то режим копирования формата будет отключен сразу после копирования элементов форматирования в выделенный элемент управления. Если по кнопке **Формат по образцу** (Format Painter) был произведен двойной щелчок, то шаг 3 можно повторять любое число раз. Для отключения режима копирования формата нажмите кнопку **Формат по образцу** (Format Painter) еще раз или нажмите клавишу <Esc>.

Оформление страниц отчета

При создании отчета важно не только определить данные, которые должны быть представлены в отчете, но и нужным образом оформить отчет как документ: правильно разбить его на страницы, пронумеровать их, выделить наиболее важные данные. В этом разделе будут рассмотрены вопросы, касающиеся оформления страниц отчета.

Добавление разрывов и нумерации страниц

Если не все нужные данные помещаются на одной странице, Access позволяет вставить принудительный разрыв страницы, чтобы не разрывать группы данных и не нарушать структуру отчета. Для этого выполните следующее:

1. Откройте отчет в режиме Конструктора.
2. На панели элементов нажмите кнопку **Разрыв страницы** (Page Break). В тех областях отчета, где можно вставить разрыв страницы, курсор мыши будет принимать вид маленького креста, к которому прикреплена страница.
3. Поместите указатель мыши в то место, где необходимо разорвать страницу, и щелкните левой кнопкой мыши. Появится символ разрыва страницы.

Теперь вся информация, расположенная после вставленного разрыва страницы, будет выводиться на новой странице. При необходимости использования принудительных разрывов страниц и в других местах отчета повторите описанную ранее процедуру.

Вставленный разрыв страницы можно перемещать в пределах всего отчета. Для этого выделите его, щелкнув по нему левой кнопкой мыши, либо выбрав его название в списке **Объект** (Object) панели форматирования, и с помощью мыши перетащите его в новое место.

Чтобы удалить разрыв страницы, выделите его и нажмите клавишу <Delete>, либо выберите команду **Правка | Удалить** (Edit | Delete).

После того, как отчет нужным образом разбит на страницы, их нужно пронумеровать. Access предоставляет несколько возможностей нумерации многостраничных отчетов. Номера страниц могут быть двух форматов, выводиться внизу или вверху каждой страницы с различным выравниванием. Для вставки номеров страниц выполните следующее:

1. Откройте отчет в режиме Конструктора.
2. Выберите команду **Вставка | Номера страниц** (Insert | Page Numbers).
3. В появившемся диалоговом окне **Номера страниц** (Page Numbers) установите нужные параметры и нажмите кнопку **OK**.

В разделе нижнего или верхнего колонтитула (в зависимости от выбранного в окне **Номера страниц** (Page Numbers) значения) появится текстовое поле, источником данных которого будет выражение, задающее номер страницы в выбранном формате. Как и любое другое текстовое поле, номер страницы можно перемещать и форматировать, изменять его свойства и размер.

Использование в отчете линий и прямоугольников

Чтобы сделать отчет более эффектным и наглядным, Access позволяет использовать в нем линии и прямоугольники. С помощью этих элементов можно выделять заголовки и группы данных.

Чтобы добавить к отчету линию:

1. Откройте отчет в режиме Конструктора.
2. На панели элементов нажмите кнопку **Линия** (Line). В тех областях отчета, где можно вставить линию, курсор мыши будет принимать вид маленького креста, к которому прикреплена линия.

3. Щелкните левой кнопкой мыши в том месте отчета, где должна начинаться линия, и, не отпуская кнопку мыши, нарисуйте линию нужной длины, затем отпустите кнопку мыши.

Нарисованную линию можно перемещать, изменять ее размеры и свойства с помощью тех же приемов, что и для любого другого элемента управления. Чтобы удалить линию, выделите ее и нажмите клавишу <Delete>.

Совет

Если после изменения размеров горизонтальной или вертикальной линии с помощью мыши вы случайно сделали ее наклоненной на небольшой градус, который не удается исправить с помощью перетаскивания маркеров размера, установите в окне свойств нулевое значение для свойства **Высота** (Height) или **Ширина** (Width) соответственно.

Чтобы сделать линию более эффектной, ее нужно отформатировать:

1. Выделите линию, щелкнув по ней левой кнопкой мыши.
2. На панели инструментов **Конструктор отчетов** (Report Design) нажмите кнопку **Свойства** (Properties).
3. В появившемся окне свойств раскройте вкладку **Макет** (Format).
4. Измените стиль, цвет и ширину линии, установив нужные значения для свойств **Тип границы** (Border Style), **Цвет границы** (Border Color) и **Ширина границы** (Border Width).
5. Свойство **Оформление** (Special Effect) позволяет придать линии специальный эффект изображения, например, сделать ее приподнятой, утопленной, либо добавить к ней тень. Для чего просто выберите необходимый элемент в списке этого свойства.

Заключив в прямоугольник несколько элементов управления, можно выделить соответствующую группу данных:

1. Откройте отчет в режиме Конструктора.
2. На панели элементов нажмите кнопку **Прямоугольник** (Rectangle). В тех областях отчета, где можно вставить прямоугольник, курсор мыши будет принимать вид маленького креста, к которому прикреплен прямоугольник.
3. Щелкните левой кнопкой мыши в том месте отчета, где должен располагаться верхний левый угол прямоугольника, и, не отпуская кнопку мыши, перетащите указатель в нижний правый угол прямоугольника, затем отпустите кнопку мыши. Внутрь нарисованного прямоугольника должны попасть элементы управления, соответствующие той группе данных, которую нужно выделить.

Нарисованный прямоугольник можно перемещать, изменять его размеры и свойства с помощью тех же приемов, что и для любого другого элемента управления. Чтобы удалить прямоугольник, выделите его и нажмите клавишу <Delete>.

Чтобы попавшие в нарисованную прямоугольную область данные еще больше обращали на себя внимание, нужно отформатировать нарисованный прямоугольник:

1. Выделите прямоугольник, щелкнув по нему левой кнопкой мыши.
2. На панели инструментов **Конструктор отчетов** (Report Design) нажмите кнопку **Свойства** (Properties).

3. В появившемся окне свойств раскройте вкладку **Макет** (Format).
4. Чтобы изменить параметры границы прямоугольника, измените значения свойств **Тип границы** (Border Style), **Цвет границы** (Border Color) и **Ширина границы** (Border Width).
5. Чтобы прямоугольник не перекрывал данные, которые он выделяет, установите для свойства **Тип фона** (Back Style) значение **Прозрачный** (Transparent) или переместите непрозрачный, залитый цветом, прямоугольник на задний план (по отношению к элементам управления с данными) с помощью команды **Формат | На задний план** (Format | Send to Back).
6. Чтобы сделать прямоугольник более эффектным и еще больше выделить заключенные в него данные, выберите один из элементов раскрывающегося списка свойства **Оформление** (Special Effect). С помощью этого свойства прямоугольник можно сделать, например, приподнятым, утопленным, либо добавить к нему тень.

Создание в отчете вычисляемых полей

Вычисляемые поля предназначены для отображения в отчетах значений выражений на основе исходных данных. Вычисляемые поля в отчетах можно использовать для получения итоговых значений или проведения специальных вычислений, например, чтобы отобразить общую цену заказа, вычислив произведение количества товара и цены одного изделия. Рассмотрим процедуру создания вычисляемых полей на примере отчета `Товары на складе`, созданного ранее.

Чтобы добавить к отчету вычисляемое поле `Сумма`:

1. Откройте отчет `Товары на складе` в режиме Конструктора.
2. На панели элементов нажмите кнопку **Надпись** (Label) и поместите новую надпись справа от надписи `На складе` (Units in Stock) в разделе верхнего колонтитула. Введите в качестве текста надписи слово `Сумма` (Sum).
3. Чтобы создать вычисляемое поле `Сумма` (Sum), нажмите кнопку **Поле** (Text Box) на панели элементов и добавьте текстовое поле справа от текстового поля `НаСкладе` (UnitsInStock) в разделе данных. Удалите метку нового текстового поля.
4. Щелкните внутри текстового поля и введите в качестве его значения выражение: `=[НаСкладе]*[Цена]` (в английской версии: `=[UnitsInStock]*[UnitPrice]`). Откройте окно свойств поля, дважды щелкнув по нему, и на вкладке **Макет** (Format) в поле со списком свойства **Формат поля** (Format) выберите значение **Денежный** (Currency).

Замечание

Ввести выражение в текстовое поле можно несколькими способами: выделите его и нажмите комбинацию клавиш <Shift>+<F2>, чтобы войти в режим редактирования содержимого поля, или сделайте двойной щелчок по полю для отображения диалогового окна **Свойства** (Properties), а затем введите выражение как значение свойства **Данные** (Source).

5. Повторите шаги 3 и 4 для того, чтобы создать текстовое поле в разделе примечаний группы `Категория` (Category), но в качестве выражения для вычисле-

ния промежуточного значения следует ввести: `=Sum([НаСкладе]*[Цена])` (или `=Sum([UnitsInStock]*[UnitPrice])`). Нажмите кнопку **Полужирный** (Bold) на панели форматирования, чтобы установить полужирный шрифт для текста в новом поле. Откройте диалоговое окно **Свойства** (Properties) и раскройте в нем вкладку **Другие** (Other), в поле **Имя** (Name) введите имя текстового поля `Сумма` (Sum). Затем раскройте вкладку **Макет** (Format) и в поле со списком свойства **Формат поля** (Format) выберите значение **Денежный** (Currency).

6. Поскольку ширина области данных нашего отчета увеличилась, уменьшите ширину полей страницы, чтобы все поля поместились на одной странице. Для этого выберите команду **Файл | Параметры страницы** (File | Page Setup) и установите вместо ширины левого и правого полей, заданных по умолчанию, значение 10 мм *(см. разд. "Просмотр и печать отчетов" в этой главе)*.

Замечание

Благодаря новым возможностям Access 2003, это несоответствие существующей ширины полей страницы будет выявлено автоматически и отображено в виде смарт-тега, обозначающего ошибку: в левом верхнем углу отчета отобразится маленький зеленый треугольник, при щелчке на котором отобразится этот смарт-тег, позволяющий получить информацию об ошибке и исправить ее.

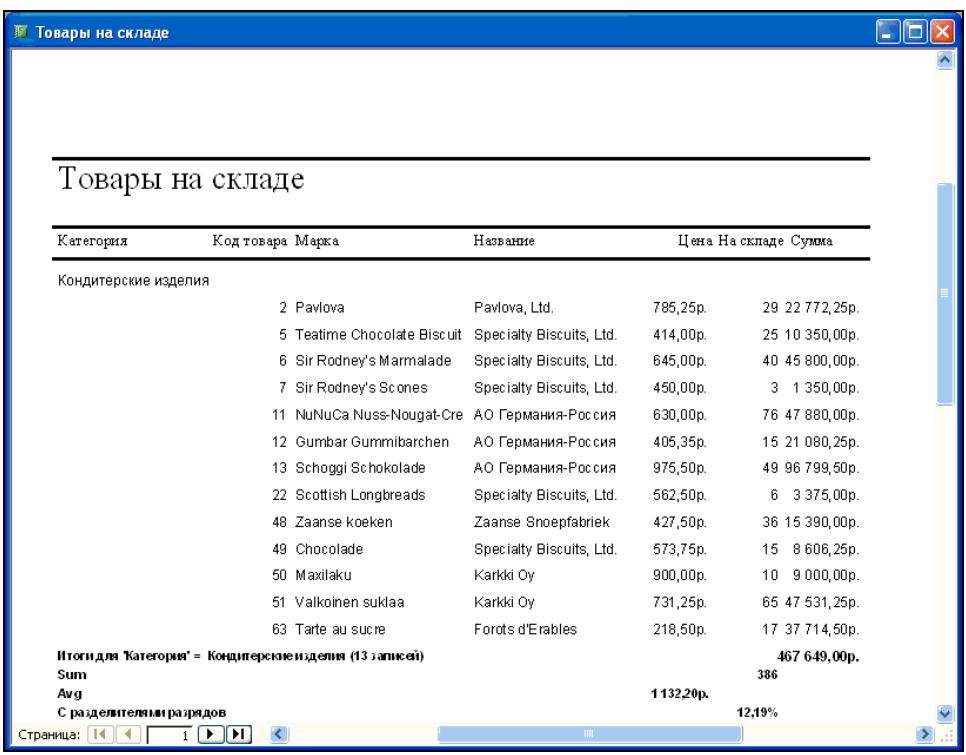


Рис. 6.27. Первая страница отчета с вычисляемыми полями

7. Нажмите кнопку **Вид** (View) на панели инструментов **Конструктор отчетов** (Report Design), чтобы оценить результаты своей работы. Отчет будет выглядеть так, как показано на рис. 6.27. К прежним данным добавился крайний правый столбец со значениями стоимости каждого товара в имеющемся количестве на складе и промежуточных итоговых сумм по стоимости товаров для каждой категории.

Замечание

Если при переходе в режим просмотра отчета появляется диалоговое окно **Введите значение параметра** (Enter Parameter Value), это означает, что в выражениях неправильно заданы названия одного или нескольких полей. Нажмите кнопку **Отмена** (Cancel) и проверьте свойства, установленные на шагах 4 и 5.

Сортировка и группировка данных

Отчеты становятся более понятными и структурированными, если данные в них организованы в группы и подгруппы. В *разд. "Создание отчета с помощью Мастера"* было показано, как задать начальные свойства группировки и сортировки данных. Однако структуру отчета можно изменить. Для этого используется диалоговое окно **Сортировка и группировка** (Sorting and Grouping), изображенное на рис. 6.28.

Чтобы открыть диалоговое окно **Сортировка и группировка** (Sorting and Grouping), откройте отчет в режиме Конструктора и нажмите кнопку **Сортировка и группировка** (Sorting and Grouping) на панели инструментов **Конструктор отчетов** (Report Design).

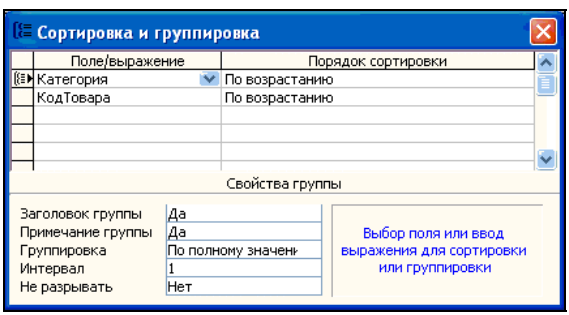


Рис. 6.28. Диалоговое окно **Сортировка и группировка**

В этом диалоговом окне можно определить поля или выражения, по которым Access будет группировать данные. Кроме того, сгруппированные данные можно сортировать по возрастанию или по убыванию. Символ **Сортировка и группировка** (Sorting and Grouping) на кнопке выделения в левом сером столбце этого диалогового окна указывает на то, что данное поле или выражение (находящееся справа от значка) используется для группировки записей. В данном примере поле `Категория` (CategoryName) используется для группировки строк отчета, а сортировка строк в пределах группы выполняется по полю `КодТовара` (ProductID).

Access позволяет группировать данные двумя способами: по категориям, при этом каждой категории должно соответствовать уникальное значение, и по диапазону

значений, который может быть как числовым, так и алфавитным. При этом для группировки данных можно использовать не только значения полей, но и выражения.

Замечание

Уровней группировки может быть несколько, т. е. можно создавать подгруппы внутри группы записей. Примером отчета с несколькими уровнями группировки, одним из которых является группировка по значению выражения в качестве категории, является отчет `Продажи по сотрудникам и странам` (Employee Sales by Country) — один из отчетов базы данных `Борей`. В нем используются группы (страны) и подгруппы (имена сотрудников). Более подробную информацию о выполнении группировки в отчетах вы можете найти также в *разд. "Сортировка и группировка данных", гл. 10.*

Применение шаблона отчетов

Как правило, отчеты, подготавливаемые в одной базе данных, должны быть выполнены в одном стиле. Потратив достаточно много времени на создание и форматирование одного отчета, не хотелось бы терять еще столько же времени на разработку остальных. В качестве решения этой проблемы Access предлагает использовать созданный отчет в качестве шаблона для всех новых отчетов.

Чтобы создать отчет, который послужит шаблоном для всех последующих отчетов:

1. В окне **База данных** (Database) щелкните по ярлыку **Отчеты** (Reports) и нажмите кнопку **Создать** (New).
2. В появившемся диалоговом окне **Новый отчет** (New Report) выделите в списке способов создания элемент **Конструктор** (Design View) и нажмите кнопку **OK**. Откроется окно Конструктора отчетов.
3. Если необходимо, добавьте к отчету разделы верхнего и нижнего колонтитулов, раздел примечания и раздел заголовка отчета. С помощью мыши установите необходимую ширину отчета и высоту его разделов.
4. Выделите весь отчет, щелкнув левой кнопкой мыши в левом верхнем углу в месте пересечения линеек, либо в списке **Объект** (Object) панели форматирования, выбрав элемент **Отчет** (Report).
5. На панели инструментов **Конструктор отчетов** (Report Design) нажмите кнопку **Свойства** (Properties).
6. В появившемся окне свойств отчета раскройте вкладку **Макет** (Format). Определите на этой вкладке те значения свойств, которые должны иметь все создаваемые отчеты. Например, можно указать файл рисунка, который будет использоваться в качестве фона отчетов, определить параметры отображения рисунка, установить точную ширину отчета в сантиметрах, задать частоту сетки, по которой выравниваются объекты отчета.

Замечание

Для контроля вносимых изменений используйте режим Предварительного просмотра. Чтобы перейти в этот режим, на панели инструментов **Конструктор отчета** (Report Design) нажмите кнопку **Вид** (View), на которой отображен элемент списка режимов **Предварительный просмотр** (Print Preview).

7. Закройте окно свойств и сохраните отчет: выберите команду **Файл | Сохранить** (File | Save) и в появившемся диалоговом окне введите название отчета, например `Образец`. Закройте отчет, щелкнув по кнопке с изображением крестика в правом верхнем углу отчета.
8. Выберите команду **Сервис | Параметры** (Tools | Options). В появившемся диалоговом окне **Параметры** (Options) раскройте вкладку **Формы и отчеты** (Forms/Reports).
9. В текстовом поле **Шаблон отчета** (Report template) укажите название созданного отчета (в нашем примере `Образец`) и нажмите кнопку **ОК**.

Теперь при создании отчета в режиме Конструктора в качестве шаблона будет использоваться указанный отчет, т. е. по умолчанию свойства создаваемого отчета будут иметь значения, определенные в шаблоне.

Замечание

Чтобы вернуть стандартное форматирование новых отчетов, в поле **Шаблон отчета** (Report template) введите название `Normal`.

Просмотр и печать отчетов

Созданный отчет можно отобразить на экране и напечатать. Access позволяет просмотреть отчет перед печатью и установить подходящие поля страницы. При просмотре отчета в режиме одной полной страницы (масштаб 100%) Access отображает отчет так, как он будет выглядеть при печати.

Просмотреть отчет перед печатью можно в режиме Просмотра образца или в режиме Предварительного просмотра. Переход из одного режима в другой описан в *разд. "Режимы работы с отчетами"*.

Чтобы изменить поля страницы отчета:

1. В режиме Предварительного просмотра отчета выберите команду **Файл | Параметры страницы** (File | Page Setup). Откроется диалоговое окно **Параметры страницы** (Page Setup), изображенное на рис. 6.29. Это диалоговое окно похоже на диалоговые окна **Параметры страницы** (Page Setup) в других приложениях Windows.

Замечание

Эту же команду можно использовать и в режиме Просмотра образца отчета, и в режиме Конструктора, после выделения отчета в списке отчетов в окне базы данных.

2. Раскройте вкладку **Поля** (Margins) и уменьшите поля сверху, снизу, слева и справа, чтобы увеличить количество информации, размещаемое на одной странице. Установка флажка **Печатать только данные** (Print Data Only) позволяет распечатать только данные из отчета, при этом графическое оформление элементов управления, верхний и нижний колонтитулы, а также разделы заголовка и примечаний отчета печататься не будут.

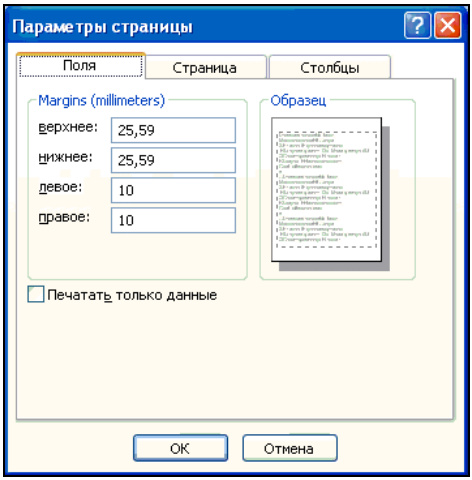


Рис. 6.29. Диалоговое окно **Параметры страницы**

Замечание

Учтите, что установленные в диалоговом окне **Параметры страницы** (Page Setup) значения размеров полей страницы будут использоваться только в данном отчете, поскольку для каждого отчета задается свой набор значений полей. При сохранении отчета Access сохраняет информацию о полях страницы. Для того чтобы установить значения полей страницы, которые будут применяться по умолчанию для всех отчетов базы данных, нужно использовать вкладку **Общие** (General) диалогового окна **Параметры** (Options) *(см. также "Настройка пользовательского интерфейса", гл. 14).*

3. На вкладке **Страница** (Page) диалогового окна **Параметры страницы** (Page Setup) можно определить ориентацию страницы, тип бумаги и принтер, на котором нужно напечатать отчет. После установки всех параметров нажмите кнопку **ОК**.

Чтобы при просмотре отчета увеличить какую-либо его часть, поместите над ней указатель мыши в виде увеличительного стекла и щелкните левой кнопкой мыши, либо нажмите кнопку **Масштаб** (Zoom) на панели инструментов **Предварительный просмотр** (Print Preview). Можно также выбрать в раскрывающемся списке **Масштаб** (Zoom) той же панели инструментов нужный процент увеличения.

Чтобы напечатать отчет, нажмите кнопку **Печать (факс)** (Print (Fax)) на панели инструментов **Предварительный просмотр** (Print Preview). Access напечатает отчет, используя текущие настройки принтера. Если вы хотите поменять принтер, ориентацию страницы или значение какого-либо другого параметра принтера, то выберите команду **Файл | Печать** (File | Print). На экране появится стандартное диалоговое окно **Печать** (Print) для принтера, заданного в Windows в качестве принтера по умолчанию (рис. 6.30).

С помощью диалогового окна **Печать** (Print) можно распечатать либо весь отчет целиком, либо его часть, вывести отчет в файл для последующей печати и указать ко-

личество печатаемых копий. Нажав кнопку **Свойства** (Properties), можно изменить значения параметров, относящихся к выбранному принтеру. Для печати отчета нажмите кнопку **OK**.

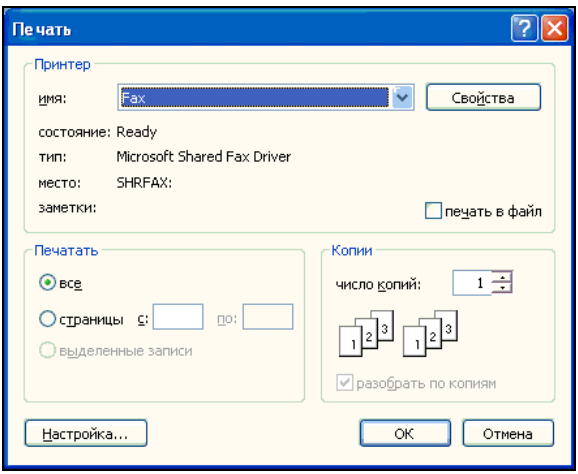


Рис. 6.30. Диалоговое окно **Печать**



ЧАСТЬ II

Профессиональная работа с Microsoft Access 2003

Глава 7. Новые возможности Microsoft Access 2003

Глава 8. Редактирование и анализ данных с помощью запросов

Глава 9. Создание составных форм

Глава 10. Подготовка составных отчетов

Глава 11. Публикация данных в Интернете и корпоративной сети

Глава 12. Программирование с помощью макросов

Глава 13. Программирование на языке VBA

Глава 14. Настройка пользовательского интерфейса

Глава 15. Интеграция Access 2003 с компонентами Office 2003

ГЛАВА 7



Новые возможности Microsoft Access 2003

Приложение Microsoft Access 2003 входит в комплект Microsoft Office 2003. Оно может использоваться отдельно от других приложений комплекта для разработки удобных и функциональных приложений баз данных, но эффективность от применения Access совместно с другими приложениями Microsoft Office многократно возрастает и помогает в решении комплексных задач информационного бизнеса. Поэтому прежде чем перейти к рассмотрению новых возможностей Microsoft Access 2003, поговорим о новшествах всего комплекта офисных приложений в целом. Первый раздел данной главы посвящен этому вопросу.

Затем, в следующих разделах главы, мы рассмотрим, что нового появилось именно в Microsoft Access 2003 по сравнению с предыдущей версией Microsoft Access 2002. Основными новшествами Access 2003 являются:

- новый дизайн пользовательского интерфейса;
- появление системы защиты от вирусов, распространяющихся через макросы в базах данных;
- введение поддержки технологии Smart Tag (смарт-тег), ранее доступной лишь в других приложениях офис;
- существенное расширение поддержки формата XML;
- появление новых средств совместной работы — интеграция Access с SharePoint;
- расширение функциональных возможностей по разработке и сопровождению баз данных Microsoft Access.

Новые возможности комплекта приложений Microsoft Office 2003

В настоящее время информационные технологии проникают во все сферы деятельности человека. Программные средства для работы с информацией используются шире, чем когда-либо раньше. Компания Microsoft проанализировала отклики пользователей своих приложений, изучила потребности людей, работающих с информа-

цией, их привычки, типичные задачи и процедуры, которые они выполняют изо дня в день, чтобы понять, как в действительности они используют приложения в своем бизнесе. Проектируя новую версию комплекта офисных приложений, Microsoft сконцентрировалась на решении общих проблем.

- Огромный объем накопленной информации затрудняет ее использование, анализ и поиск необходимых сведений.
- Сотрудничество и обмен информацией требуют больших усилий и, как правило, осложнены для людей, находящихся территориально в разных местах. До сегодняшнего дня было тяжело организовать сотрудничество без разворачивания в данном месте специальной инфраструктуры.
- Разрозненные островки данных, образовавшиеся вследствие многолетнего накопления, преобразования и слияния информации, сводят на нет возможность извлечь полезную информацию и превращают корпоративную информационную систему в "силосную яму".
- Бизнес-процессы недостаточно интегрированы в существующие информационные системы.
- Используемые программные средства должны быть продуктивными и простыми в использовании.
- Необходима интеграция между приложениями и разнообразными устройствами.

Следующее поколение популярных офисных приложений от Microsoft обеспечивает пользователя всеми необходимыми средствами организации бизнеса на современном уровне. Использование новой версии Microsoft Office 2003 позволяет преодолеть указанные трудности и увеличить продуктивность работы как отдельного пользователя, так и организации в целом благодаря целому ряду качественных усовершенствований на уровне всего комплекта приложений.

- Применение Microsoft Office 2003 открывает новые возможности для упорядочивания больших объемов разнородной информации, хранящейся в разных источниках.
- Приложения Microsoft Office 2003, интегрированные со службами Microsoft Windows SharePoint, предоставляют пользователю мощные средства совместной работы, доступные через привычный и удобный интерфейс. А интегрированное средство обмена мгновенными сообщениями между виртуально присутствующими собеседниками обеспечивает постоянную коммуникацию пользователей между собой.
- Microsoft Office 2003 является первым многофункциональным комплектом настольных приложений, включающим полноценную поддержку индустриального стандарта универсального описания информации XML и определяемых пользователем схем данных. Это открывает легкий путь к соединению различных информационных систем и предоставлению хранящихся в них данных пользователю.
- С помощью приложений Microsoft Office 2003 можно реализовать простые и эффективные процессы для выполнения ежедневных бизнес-функций, таких как оформление заказов, счетов, пресс-релизов, отчетов, презентаций, составление писем клиентам и т. п. В данные процессы могут быть вовлечены сотрудники

разных отделов и они могут совместно работать над одними и теми же документами. При этом исчезает надобность в передаче документов от сотрудника к сотруднику и обратно.

- Новые средства и технологии, предлагаемые Microsoft Office 2003, такие как Панели задач и смарт-теги (Smart Tags), делают работу с проектами еще проще и удобнее, особенно при работе в команде.
- Microsoft Office 2003 поддерживает подключение мини-компьютеров по инфракрасному порту, что позволяет, например, синхронизировать электронную почту на мобильных устройствах, а также интегрироваться с широким спектром разнообразных электронных устройств.

Подводя итог, можно сказать, что новая версия комплекта Microsoft Office 2003 не просто предоставляет новые функциональные средства, а обеспечивает более широкие возможности по управлению информацией, повышает эффективность бизнес-процессов, коммуникации и сотрудничества на уровне всей организации, а также увеличивает персональную продуктивность отдельного пользователя.

Новые требования к системе

Установка Microsoft Office 2003 предъявляет новые требования к системе.

- Персональный компьютер на базе микропроцессора Pentium 133 МГц или выше; рекомендуется использовать Pentium III.
- Операционная система Microsoft Windows 2000 с установленным пакетом обновлений Service Pack 3, либо операционная система Microsoft Windows XP или выше.
- Программа Microsoft Internet Explorer 5.01 или выше, либо Netscape Navigator не ниже версии 6.2.

Если ваш персональный компьютер удовлетворяет перечисленным требованиям, вы можете выполнить установку Microsoft Office 2003, а затем зарегистрировать вашу копию в Microsoft через Интернет или по телефону с помощью Мастера активации офисных приложений.

Новый дизайн пользовательского интерфейса

Графический интерфейс всего комплекта новых офисных приложений выполнен в стиле оформления Windows XP. Теперь панели и окна офисных приложений снабжены полноцветными трехмерными элементами и органично сочетаются с внешним видом новейшей операционной системы от Microsoft (к сожалению, на черно-белых иллюстрациях в книге этого не видно).

Внешний вид рабочей области Microsoft Access 2003 представлен на рис. 7.1. Сравните его с внешним видом предыдущей версии Microsoft Access 2002 (рис. 7.2).

Обратите внимание на новый внешний вид Панели задач, отображенной в правой части экрана (см. рис. 7.1).

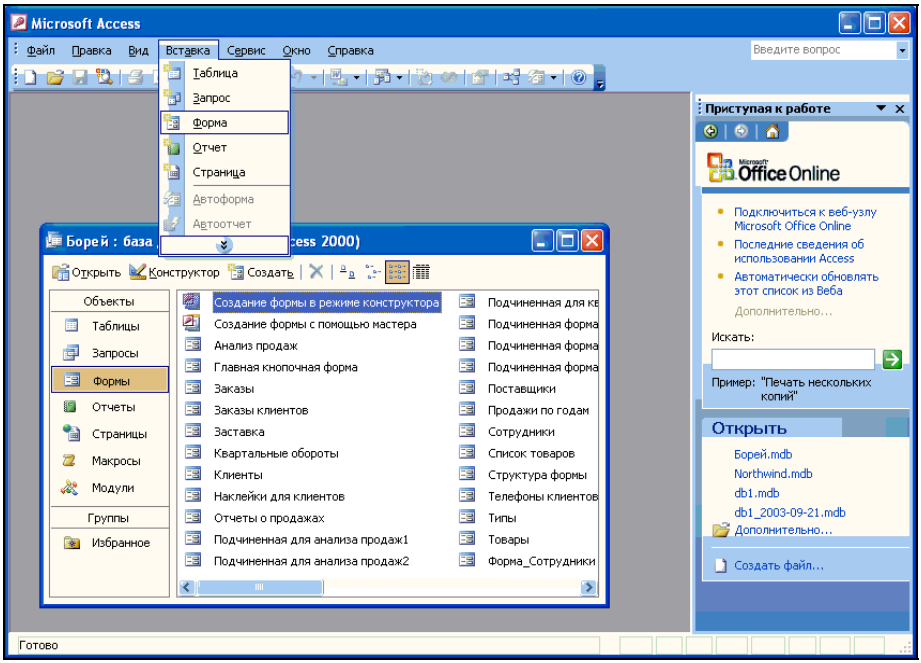


Рис. 7.1. Внешний вид рабочей области Microsoft Access 2003

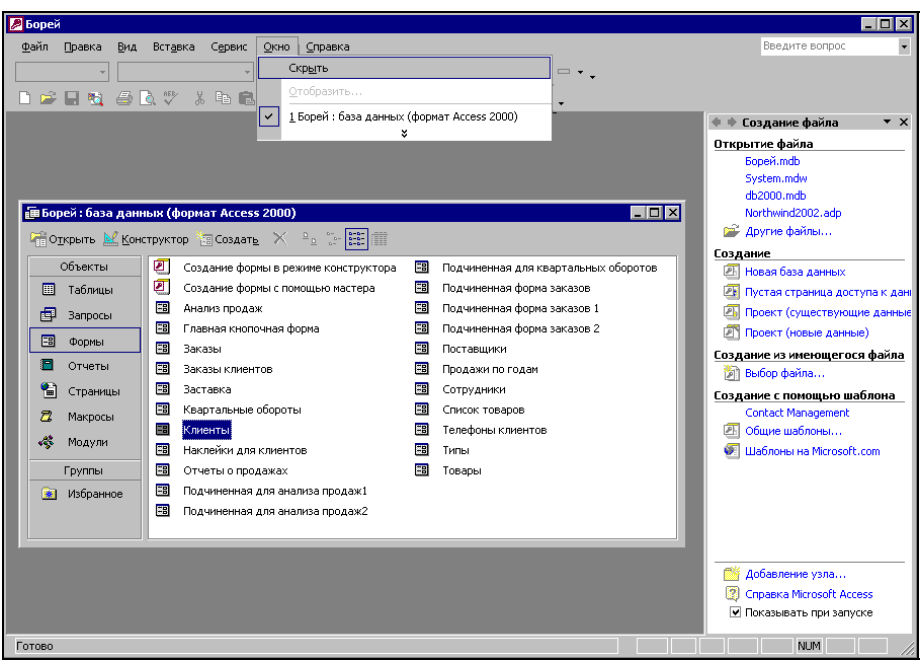


Рис. 7.2. Внешний вид рабочей области Microsoft Access 2002

В этой части рабочей области Access 2003 может отображаться одна из следующих панелей:

- **Приступая к работе** (Getting Started);
- **Справка Access** (Microsoft Access Help);
- **Поиск** (Search);
- **Результаты поиска** (Search Results);
- **Поиск файлов** (File Search);
- **Буфер обмена** (Clipboard);
- **Зависимости объектов** (Object Dependencies);
- **Создание файла** (New File).

Каждая из этих панелей предназначена для решения одной из типичных задач работы в среде Microsoft Access. Некоторые из этих панелей (**Создание файла** (New File), **Поиск файлов** (File Search) и **Буфер обмена** (Clipboard)) появились еще в предыдущей версии Access 2002, но и они несколько изменились внешне в новой версии Access 2003.

Переход между панелями осуществляется с помощью кнопок со стрелками, расположенных в левом верхнем углу Панели задач, или посредством меню, раскрывающегося щелчком по названию текущей панели вверху Панели задач. Ширину панели можно очень легко регулировать с помощью мыши.

Панель задач можно скрыть, если она мешает, щелкнув по крестику в ее правом верхнем углу. Снова отобразить или скрыть Панель задач можно щелкнув правой кнопкой по обычной панели инструментов и в раскрывшемся меню со списком названий панелей инструментов выбрать пункт **Область задач** (Task Pane).

Панель **Приступая к работе** (см. рис. 7.1) появляется на экране при запуске Access 2003 автоматически. К ней всегда можно быстро перейти, даже если открыта другая панель, просто щелкнув по кнопке **Домашняя страница** (Home). С помощью этой панели можно быстро выполнить следующие основные команды:

- открыть одну из ранее открывавшихся в Access баз данных и проектов;
- открыть любую базу данных или проект Access, или другой файл;
- перейти к панели **Создание файла** (New File).

Панель **Создание файла** (New File) (рис. 7.3) можно отобразить также командой меню **Файл | Создать** (File | New). С помощью этой панели можно выполнить команды:

- создать пустую базу данных или надстройку Access;
- создать пустую страницу доступа к данным;
- создать проект Access на основе новых или существующих данных на сервере SQL;
- создать базу данных, проект Access или страницу доступа к данным на основе существующего файла или шаблона, или с помощью мастера.

Еще одна полезная панель для работы с файлами отображается с помощью кнопки **Поиск файлов** (File Search) на панели инструментов **База данных** (Database). Ее

можно открыть командой меню **Файл | Поиск файлов** (File | File Search). Эта панель предназначена для быстрого поиска баз данных Access, документов и прочих файлов, созданных в приложениях Office. Данная панель предоставляет пользователю два режима поиска: **Обычный поиск файлов** (Basic File Search) и **Расширенный поиск файлов** (Advanced File Search) (рис. 7.4). Переключиться между режимами поиска очень просто с помощью гиперссылки в нижней части панели.

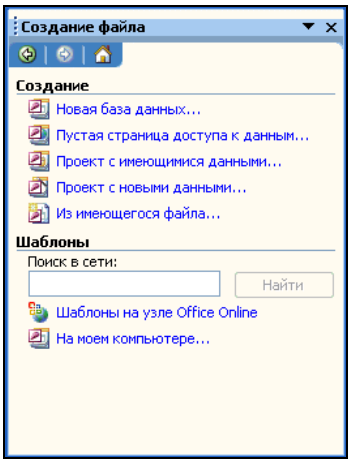


Рис. 7.3. Панель **Создание файла**

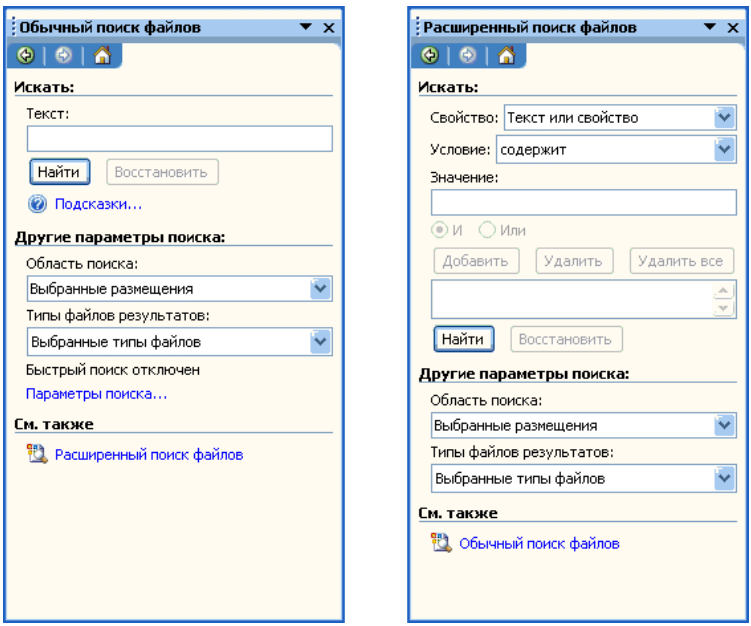


Рис. 7.4. Панель **Поиск файлов** в обычном (слева) и расширенном (справа) режиме

В режиме обычного поиска можно найти любой документ Microsoft Office, Web-страницу или элемент Outlook, в том числе содержащий указанный текст, в любой заданной папке или на всех ресурсах компьютера, включая папки Outlook и папки Web. Пока выполняется процесс поиска, вы можете продолжать работу. Найденные файлы отображаются на отдельной панели **Результаты поиска** в виде списка, каждый элемент которого снабжается контекстным меню, с помощью которого с найденным файлом можно выполнить то или иное действие в зависимости от его типа (рис. 7.5).

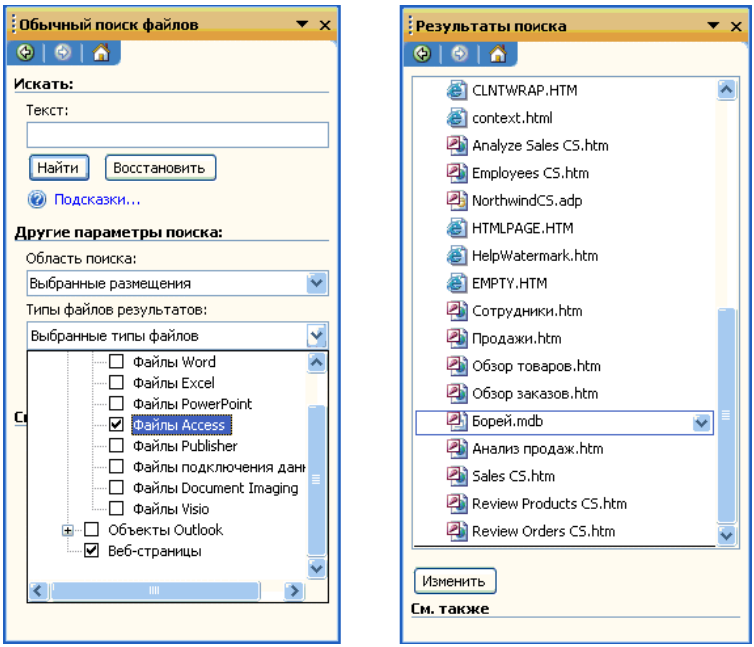


Рис. 7.5. Условия (слева) и результат (справа) поиска

В режиме расширенного поиска можно искать документы Office, Web-страницы и элементы Outlook с заданными условиями на их свойства, например: имя файла, дату создания, дату модификации, имя автора и пр. Например, чтобы найти все файлы, содержащие в имени автора имя Veronica, на панели расширенного поиска в списке **Свойство** (Property) выделите элемент **Автор** (Author), в списке **Условие** (Condition) — элемент **содержит** (includes), в поле **Значение** (Value) введите имя `Veronica` и нажмите кнопку **Добавить** (Add). В списке условий появится элемент **Автор содержит Veronica** (File name includes Veronica). Теперь можно, например, таким же образом ввести еще одно или несколько дополнительных условий на различные свойства файла, предварительно выбрав переключатель **И** (And) или **Или** (Or). После этого нужно нажать кнопку **Найти** (Go) и список найденных файлов отобразится на панели.

Панель **Буфер обмена** (Clipboard) отображается с помощью команды **Правка | Буфер обмена Office** (Edit | Office Clipboard) или нажатием два раза подряд комбинации

клавиш <Ctrl>+<C>. С помощью этой панели удобно работать с содержимым буфера обмена, в который можно скопировать и одновременно хранить до 24 элементов. Эти элементы можно вставлять в открытую базу данных в произвольном порядке и любом количестве. Их также можно удалить из Буфера обмена с помощью контекстного меню элемента на панели (рис. 7.6).

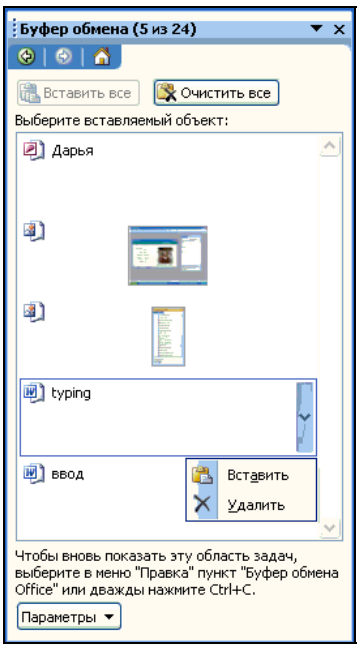


Рис. 7.6. Панель **Буфер обмена**

Панель **Зависимости объектов** (Object Dependencies) — совершенно новая панель в Access. Она появилась только в версии Access 2003. Ее назначение описано в *разд. "Новые функциональные возможности по разработке и сопровождению баз данных" этой главы.* А использование панели **Справка Access** (Microsoft Access Help) описано в следующем разделе.

Новая справочная система

При современном уровне развития технологий огромные возможности по поиску интересующей информации предоставляет Интернет. В связи с этим в новой версии Microsoft Office 2003 концепция предоставления пользователю справочной информации была существенно изменена. Теперь справочная система офисных приложений представляет собой Web-портал, открывающий доступ не только к локальной справочной информации, но и к огромному объему ресурсов с постоянно обновляемой информацией на Web-узле компании Microsoft. Новый принцип организации интерфейса справочной системы приложений Microsoft Office 2003 является естественным шагом в развитии системы получения справочной информации.

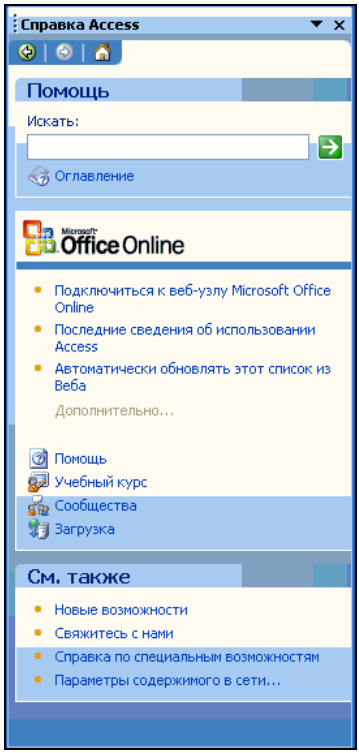


Рис. 7.7. Справочная система Microsoft Access

Главное окно справочной системы теперь отображается на Панели задач (рис. 7.7). Чтобы открыть окно справочной системы Microsoft Access 2003, выберите команду **Справка | Справка: Microsoft Office Access** (Help | Microsoft Access Help). На Панели задач откроется страница **Справка Access** (Microsoft Access Help).

На этой странице щелчком по ссылке **Оглавление** (Table of Contents) открывается страница, содержащая иерархию ссылок на разделы справочной системы. При этом многие разделы загружаются с сайта Microsoft через Интернет. Чтобы найти информацию, связанную с некоторым ключевым словом, введите это слово или несколько слов в поле **Искать** (Search) на странице **Справка Access** и нажмите маленькую кнопку рядом с полем, чтобы начать поиск.

Кроме того, полезную информацию можно получить также на другой странице Панели задач — **Приступая к работе** (Getting Started) (см. рис. 7.1). Здесь есть несколько ссылок на полезные ресурсы Интернета, а также поле **Искать** (Search) для поиска по ключевым словам.

Новый формат файлов баз данных

В новой версии Microsoft Office 2003 можно открывать файлы, созданные в предыдущих версиях: Office XP, Office 2000 и Office 97. Это справедливо и для Access 2003.

Кроме того, файлы, созданные в Microsoft Office 2003, могут быть открыты в любой из перечисленных предыдущих версий. Однако для Access такая обратная совместимость не поддерживается.

По умолчанию в Microsoft Access 2003 для создания баз данных используется формат Access 2000. Можно потребовать использовать новый формат для создания баз данных, установив соответствующий параметр в окне настройки параметров базы данных (команда **Сервис | Параметры** (Tools, Options), вкладка **Другие** (Advanced)). Однако если вы выберите новейший формат Access 2003, имейте в виду, что такую базу данных нельзя будет использовать в Access 2000.

Существующую базу данных можно преобразовать в любой из доступных форматов Access: Access 97, Access 2000 и обновленный формат Access 2002—2003. Для этого используйте соответствующие команды в меню **Сервис | Служебные программы | Преобразовать базу данных** (Tools | Database Utilities | Convert Database).

Не забывайте, что некоторые новые возможности, доступные в последней версии Microsoft Office 2003, могут быть недоступны или выглядеть по-другому в предыдущих версиях Microsoft Office.

Новые средства защиты от макровирусов

К сожалению, опасность пострадать от атаки вирусов постоянно возрастает. Непрерывно разрабатываются все новые и новые вирусы. Недавно большое количество компьютеров во всем мире пострадало от нового семейства вирусов типа W32\Blaster, ориентированных на нарушение работоспособности программного обеспечения компании Microsoft. Сообщения об этом представлены на Web-узле Microsoft. Даже если разрушительная сила вируса направлена против производителя программного обеспечения, как в случае с вирусом Blaster, в первую очередь страдают конечные пользователи, теряющие рабочее время на восстановление утраченных данных и испорченной операционной системы. Последние новости и информацию о том, как защитить свой персональный компьютер от неблагоприятного вмешательства, периодически публикуются в Интернете. Последнюю информацию, касающуюся безопасности продуктов Microsoft, смотрите на Web-узле компании: **http://www.microsoft.com/**.

Теперь и в Microsoft Access предусмотрена защита от вирусов, которые могут распространяться через автоматически запускаемые макросы в файлах баз данных. Теперь, как и в Microsoft Word, в Microsoft Access при открытии базы данных, содержащей исполняемый код (будь то макросы или модули VBA), появляется предупреждающее сообщение с возможностью отключить макросы, чтобы обеспечить защиту от возможного запуска вирусов (рис. 7.8).

Появление этого окна зависит от установленного уровня системы безопасности Microsoft Access 2003. Выбрать один из трех уровней защиты (высокий, средний или низкий) позволяет диалоговое окно, представленное на рис. 7.9, отображаемое командой меню **Сервис | Макрос | Безопасность** (Tools | Macro | Security).

Совет

Если команда **Безопасность** (Security) в меню **Макрос** (Macro) отсутствует, чтобы ее отобразить воспользуйтесь средством настройки меню и панелей инструментов

Access. Для этого щелкните правой кнопкой по любой панели инструментов и в контекстном меню выберите команду **Настройка** (Customize). В появившемся диалоговом окне на вкладке **Команды** (Commands) в списке **Категории** (Categories) выделите элемент **Сервис** (Tools). В появившемся списке **Команды** (Commands) найдите элемент Безопасность и перетащите его с помощью мыши в меню **Сервис | Макрос** (Tools | Macro). Закройте диалоговое окно **Настройка** (Customize).

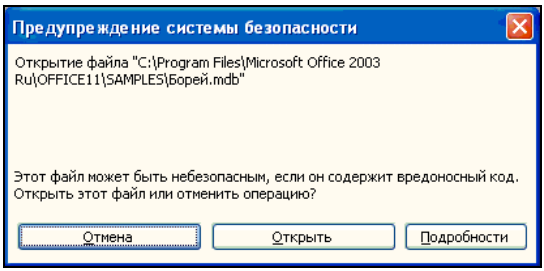


Рис. 7.8. Предупреждение о наличии потенциально опасного программного кода в базе данных

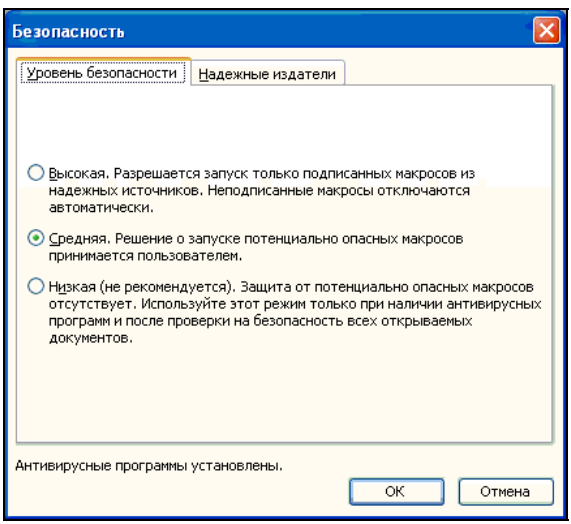


Рис. 7.9. Диалоговое окно установки уровня системы безопасности

В диалоговом окне **Безопасность** (Security) на вкладке **Надежные издатели** (Trusted Publishers) отображается список поставщиков баз данных, которым можно доверять. Список надежных издателей можно пополнить во время открытия файла или загрузки надстройки с подписанным макросом.

Поддержка технологии Smart Tag

Технология Smart Tag (или просто смарт-теги) уже существовала в предыдущей версии пакета Microsoft Office XP. Пользователям и разработчикам приложений Word и

Excel она уже знакома. В новой версии Microsoft Office 2003 эта технология распространена и на другие приложения: Access, PowerPoint и Outlook.

Способ применения и пользовательский интерфейс в виде специальных автоматически отображаемых элементов (смарт-тегов) одинаков во всех приложениях семейства Microsoft Office, поддерживающих эту технологию. Функциональность смарт-тегов заключается в автоматическом распознавании предопределенных строк текста в процессе их ввода (например, имен, адресов и т. п.) и сопоставлении им соответствующих действий, которые пользователь может захотеть выполнить над текстом сразу же после его ввода. Пользователи Microsoft Office и поставщики программного обеспечения могут разрабатывать собственные смарт-теги, специализированные для их собственных задач. Например, с помощью специально разработанного смарт-тега можно автоматически отображать сведения о медицинской страховке по введенному идентификатору пациента или отбирать из базы данных сведения о клиенте при вводе его имени и т. п.

Разработчики приложений Microsoft Access 2003 могут присоединить смарт-теги к любым полям данных. Любой элемент управления на форме, отчете или странице доступа к данным теперь имеет дополнительное свойство для указания присоединенного смарт-тега. Поля таблиц также снабжены этим свойством. Поля данных в запросах принимают значение этого свойства из соответствующих им полей таблиц или запросов, являющихся для них источником данных.

Присоединение смарт-тегов к полям в формах не требует программирования и обеспечивает форму новыми функциональными возможностями. Таким образом, работа в Access становится похожей на работу с документом в других офисных приложениях.

Возможна настройка смарт-тегов на автоматическое выполнение действий, таких как преобразование номеров заказов в гиперссылки на базу данных продаж, добавление соответствующим образом отформатированной подписи к документу, подстановка полных имен людей или названий компаний, ранее упомянутых в статье, по введенному индексу.

Для разработки и распространения смарт-тегов легче всего воспользоваться новой версией пакета средств разработчика SDK для Office 2003.

Поддержка XML и схем данных

Благодаря поддержке XML в Microsoft Office 2003 офисные приложения могут взаимодействовать практически с любыми системами. Достаточно, чтобы эти системы воспринимали данные в формате XML. А количество и разнообразие таких систем постоянно возрастает, поскольку этот формат является общепринятым в индустрии программного обеспечения стандартом представления информации любого рода. Этот универсальный формат предназначен для представления, хранения и обмена данными любой структуры и сложности.

Поддержка XML в Microsoft Access 2003 расширена по сравнению с предыдущей версией. Теперь можно получать данные XML сразу из нескольких таблиц в базе данных. Импорт и экспорт данных в формате XML стал еще более гибким. Можно импортировать данные из файла XML, построенного на любой схеме, непосредственно в существующую структуру данных базы данных Access 2003. Для такого пре-

образования может использоваться и специально разработанный файл в формате XSLT. У операции экспорта данных из Access в формат XML тоже появился новый параметр, позволяющий включить применение файла XSLT в процессе преобразования. Таким образом, в результате экспорта можно получить файл XML со схемой, отличной от генерируемой в Access по умолчанию. Эти усовершенствования позволяют легко обмениваться информацией пользователям одной или нескольких организаций. Операция экспорта предоставляет еще несколько новых возможностей. Теперь можно экспортировать не все данные, а лишь текущую запись. Кроме того, в экспортируемые данные можно включить данные из связанных таблиц, в том числе из списков выбора значения поля.

(Более подробно операции импорта и экспорта данных формата XML в Access 2003 рассмотрены в гл. 3 и 11.)

Поддержка XML, технология Smart Tag и интегрированные Панели задач делают Microsoft Office платформой для разработки комплексных решений и бизнес-процессов на базе знакомого интерфейса офисных приложений. Появление нового приложения Microsoft InfoPath в семействе приложений Microsoft Office 2003 открывает новые перспективы в создании бизнес-решений на базе XML, в частности, в комбинации с Microsoft Access 2003.

(Более подробно новые возможности по использованию формата XML в Access 2003 описаны в гл. 11.)

Сотрудничество и обмен информацией

Обеспечение возможностей более тесного сотрудничества при работе над офисными документами и приложениями является ключевой целью нового продукта Microsoft Office 2003. Интеграция новой версии Office с Microsoft SharePoint открывает новые пути объединения сотрудничающих людей, находящихся в различных точках земного шара. Благодаря такому объединению, например, становится возможным получение экспертной оценки для любого проекта, ускоряется процесс согласования и рецензирования документов.

Новые возможности, способствующие сотрудничеству, включают более тесную интеграцию Office 2003 со службами SharePoint, улучшенные средства совместной работы через серверы командной работы (Team Sites), через совместные рабочие области документов (Document Workspaces), посредством Панелей задач SharePoint в документах Microsoft Office и через рабочие области для проведения телеконференций (Meeting Workspaces).

В частности, интеграция со службами SharePoint реализована и в Microsoft Access 2003. Теперь пользователи могут установить связь между таблицами, созданными в базе данных Access 2003, и списками данных на Web-узле SharePoint с полноценным доступом для чтения и записи информации. Благодаря этому становится возможным использовать службы SharePoint в качестве некоего нового типа сервера баз данных для пользователей Access, которые хотят сотрудничать через Web с помощью привычного интерфейса настольных приложений, или для тех, кто хочет подключить данные SharePoint к другим отслеживающим модификации документов приложениям, уже разработанным в Access.

Новые функциональные возможности по разработке и сопровождению баз данных

Новая версия Microsoft Access 2003 содержит ряд усовершенствований, расширяющих функциональные возможности, помогающие в разработке и сопровождении баз данных. Этот, на первый взгляд казалось бы, небольшой набор незначительных улучшений на практике существенно облегчает задачу модификации базы данных.

Резервное копирование баз данных

Появилась функция резервного копирования базы данных. Теперь, работая с базой данных, можно непосредственно в среде Access создать ее резервную копию. Для этого нужно выбрать новую команду меню **Сервис | Служебные программы | Резервная копия базы данных** (Tools | Database Utilities | Back Up Database). Появится диалоговое окно, в котором следует указать путь для сохранения резервной копии (это может быть локальный диск или любой другой ресурс в сети). Имя резервной копии будет задано автоматически: к имени базы данных прибавится суффикс, содержащий текущую дату. Перед выполнением данной команды Access автоматически проверяет, что никто из других пользователей не использует эту базу данных. В процессе резервного копирования Access также выполняет операции сжатия и восстановления базы данных, чтобы гарантировать целостность ее резервной копии.

Средства автозамены в виде Smart Tag

Теперь активизация любых средств автозамены приводит к появлению интерфейса смарт-тега (Smart Tag). Поскольку изменения, автоматически выполняемые над текстом, не всегда нужны пользователю, интерфейс соответствующего смарт-тега предоставляет команды **Изменить обратно** (Undo), **Отключить автозамену** (Stop) и **Параметры автозамены** (Control AutoCorrect Options), дающие пользователю возможность самому управлять результатом операции, чтобы, например, вернуться к первоначальному варианту введенного текста.

Теперь пользователь может контролировать результат выполнения следующих операций, относящихся к функциям автозамены:

- исправление двух заглавных букв в начале слова;
- преобразование в заглавную букву первой буквы предложения;
- преобразование в заглавную букву первой буквы в названии дня недели по-английски;
- автоматическая замена текста при вводе в соответствии с предопределенными шаблонами.

Операция автозамены выполняется для всех текстовых полей и полей со списком в режиме Формы (Form View) и для всех полей в режиме Таблицы (Datasheet View).

Например, при вводе в поле таблицы слова с двумя первыми заглавными буквами выполняется автоматическое исправление регистра второй буквы и отображается смарт-тег, предлагающий по желанию отменить исправление (рис. 7.10).

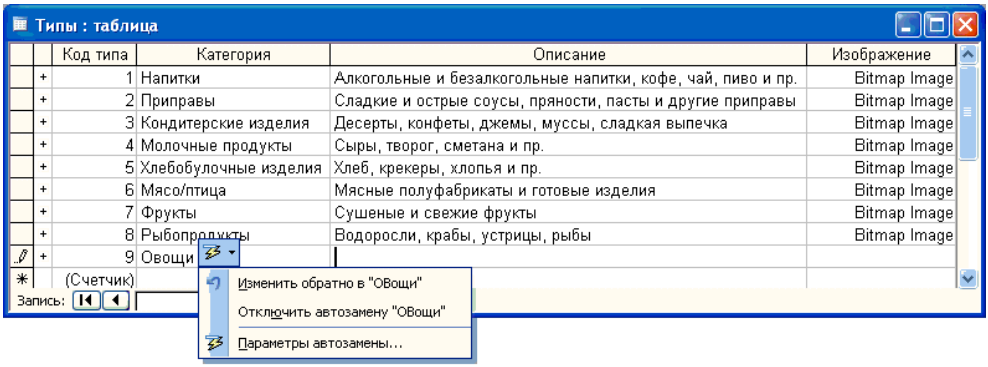


Рис. 7.10. Смарт-тег для поля таблицы

Отслеживание зависимостей объектов

В результате удаления или переименования объекта базы данных легко потерять его связь с другими, зависимыми от него объектами. В Access 2003 есть новое средство отображения зависимых объектов, которое позволяет легко найти все объекты, связанные с текущим объектом базы данных. С помощью этой информации упрощается модификация существующего приложения и сокращается время, необходимое на его отладку.

Замечание

Это средство доступно только для баз данных ядра Jet, т. е. для проектов Access, связанных с данными на SQL Server, эта возможность не предусмотрена.

Переименование объекта базы данных или поля таблицы ведет к автоматическому изменению ссылок на него в свойствах других объектов, чтобы сохранить их работоспособность.

Чтобы найти в базе данных все зависимые объекты:

1. В окне базы данных щелкните правой кнопкой по объекту, для которого требуется найти все зависимые от него объекты (это может быть таблица, запрос, форма или отчет).
2. В контекстном меню выберите команду **Зависимости объектов** (Object Dependencies).

Совет

Эта же команда для выделенного объекта доступна в меню **Вид** (View).

3. На Панели задач Access отобразится панель **Зависимости объектов** (Object Dependencies) с иерархическим списком всех объектов базы данных, связанных с указанным объектом (рис. 7.11).

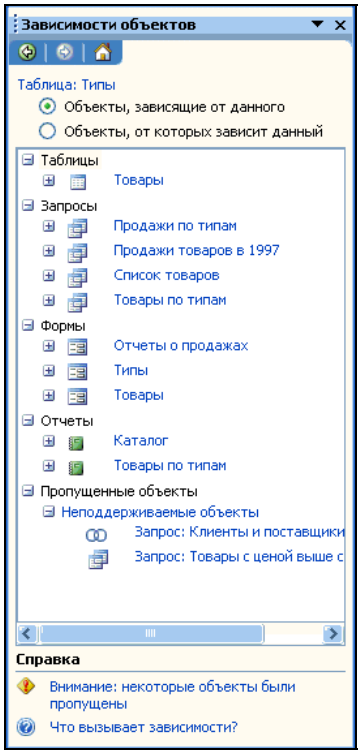


Рис. 7.11. Панель **Зависимости объектов**

Настройка шрифта в окне редактирования текста запроса

Важным усовершенствованием для пользователей Access является новая возможность настраивать размер и начертание шрифта в окне редактирования текста запроса в режиме SQL. Просьбы об этой возможности поступали от клиентов Microsoft еще с первых версий Access. Наконец, она была реализована и теперь доступна через команду меню **Сервис | Параметры** (Tools | Options). Настройка параметров шрифта для окна текста запросов выполняется на вкладке **Таблицы и запросы** (Tables/Queries) диалогового окна **Параметры** (Options).

Автоматическое исправление ошибок

В предыдущей версии Microsoft Office XP, в Excel 2002, была представлена реализация идеи использования интерфейса автозамены для выделения общих ошибок, случающихся при вводе данных в электронную таблицу, и предоставления пользователю возможности выбрать способ их исправления. Новая версия Access 2003 тоже включает подобную возможность, позволяющую выявлять наиболее общие ошибки в объектах базы данных и предлагающую пользователю выбрать вариант возможных

исправлений. Новое средство отображает соответствующий смарт-тег при обнаружении какой-либо из следующих ошибок:

- метка, не связанная с элементом управления;
- новая метка, не связанная с элементом управления;
- метка, не связанная с элементом управления, с не уникальной клавишей быстрого доступа (только в формах);
- не уникальная клавиша быстрого доступа (только в формах);
- неверно заданная клавиша быстрого доступа (только в формах);
- неверный источник данных для элемента управления (может быть предложено добавить знак "=" для исправления этой ошибки);
- циклическая ссылка;
- не уникальное значение элемента в группе;
- неверное значение параметра сортировки или группировки (только в отчетах);
- ширина отчета больше размера печатаемой страницы.

Преобразование связанных таблиц в локальные

Иногда для ускорения работы отчетов, построенных на связанных таблицах или запросах, требуется преобразовать связанные таблицы в локальные. Раньше пользователям Access приходилось для этого создавать сложные запросы на создание таблиц. Теперь пользователь может просто выделить ярлык связанной таблицы, скопировать ее в буфер обмена (например, нажав комбинацию клавиш <Ctrl>+<C>) и выполнить операцию вставки обратно в эту базу данных (например, нажав комбинацию клавиш <Ctrl>+<V>). Появится диалоговое окно, позволяющее выбрать способ копирования, в том числе и с преобразованием копируемой связанной таблицы в локальную.

Автоматическое распространение измененных свойств поля

Это новое средство позволяет изменить свойство поля базы данных в одном месте и автоматически распространить это изменение на все связанные с данным полем элементы управления. Любой разработчик баз данных Access высоко оценит пользу от этого средства и то, как просто стало распространять изменения свойств полей, сделанные в таблицах, на все связанные формы и отчеты. Раньше разработчикам приходилось вручную приводить в соответствие свойства каждого элемента управления, связанного с измененным полем. Существовала большая вероятность забыть о каких-то из таких элементов. Новая возможность существенно снижает подобный риск, а также предельно сокращает время, необходимое на выполнение такого рода задачи.

Если пользователь изменит свойство поля в таблице в режиме Конструктора (Table Design), автоматически отобразится смарт-тег **Параметры обновления свойства** (Property Update Options), предоставляющий возможность выбрать команду **Обновить *<имя свойства>* везде, где используется *<имя поля>*** (Update *<имя свойства>* every-

where <*имя поля*> is used). Например, на рис. 7.12 представлен смарт-тег, появившийся после изменения свойства **Формат** (Format) у поля `ДеньРождения` (BirthDay) в таблице `Сотрудники` (Employees) базы данных `Борей` (Northwind).

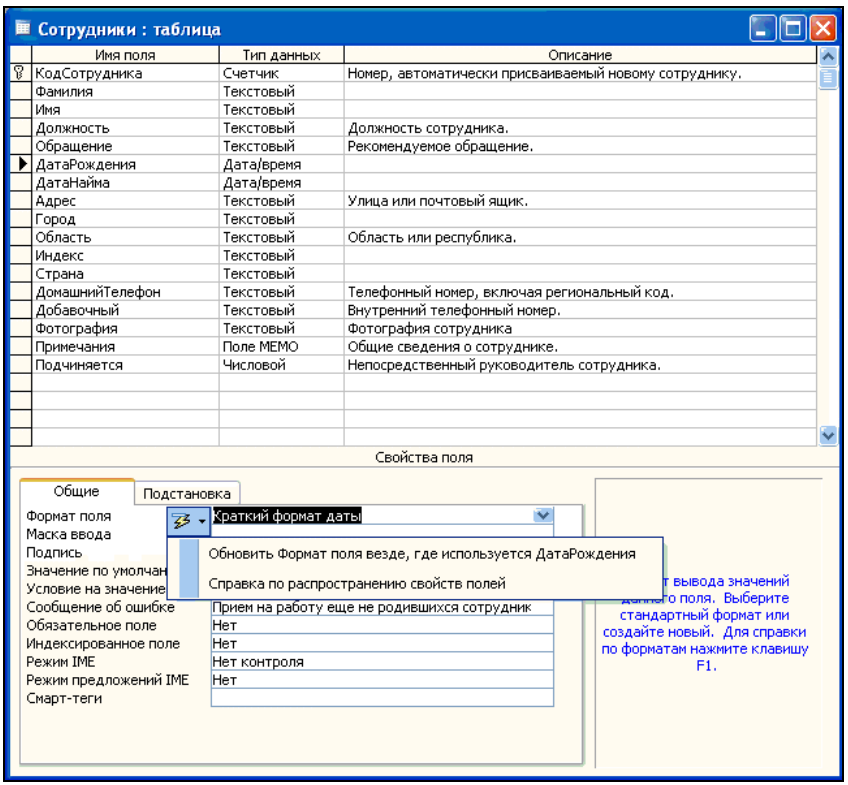


Рис. 7.12. Команда автоматического распространения изменения, сделанного в свойстве поля таблицы

ГЛАВА 8



Редактирование и анализ данных с помощью запросов

В *части I* книги мы познакомили вас с одним из важнейших объектов базы данных Access — запросами. При этом рассматривался только один вид запросов — запросы на выборку данных из таблиц. Microsoft Access, однако, позволяет не только выбирать нужные данные, но и изменять их с помощью специальных запросов. Такие запросы называются *запросами на изменение* (Action Query). Существует четыре типа запросов на изменение.

- *Запросы на создание таблицы* (Make-Table Query) создают таблицы на основе данных, содержащихся в результирующем множестве запроса. Чаще всего этот тип используется для экспорта информации в другие приложения. Кроме того, запросы на создание таблицы удобно применять для копирования таблиц в другую базу данных. В некоторых случаях с их помощью можно повысить быстродействие форм и отчетов, базирующихся на одном сложном запросе.
- *Запросы на добавление записей* (Append Query) позволяют добавить в таблицу создаваемые запросом записи.
- *Запросы на обновление* (Update Query) изменяют значения полей в записях таблицы, отобранных с помощью запроса.
- *Запросы на удаление* (Delete Query) удаляют из таблицы записи, соответствующие результирующему множеству запроса.

Если таблицы содержат большое количество данных, важно не только извлечь нужные данные, но и как-то обработать их, т. е. представить в виде, удобном для анализа. Для этого в Access существует специальный вид запросов — перекрестные запросы, а также более мощные средства — сводные таблицы и сводные диаграммы.

Самым простым способом обработки результирующих данных запроса является группировка и вычисление различных итоговых значений. Это можно сделать прямо в запросе на выборку, определив поля для группировки и используя статистические функции.

И наконец, для создания запросов можно использовать специальный язык — Jet SQL, т. е. диалект языка структурированных запросов, который используется ядром базы данных Access — процессором обработки данных Jet.

Таким образом, в этой главе описывается:

- использование запросов для обновления данных, добавления и удаления записей в таблицах;
- перекрестные запросы;
- создание таблиц с помощью запросов;
- сводные таблицы и диаграммы;
- типичные конструкции SQL при формировании различных типов запросов.

Создание таблиц с помощью запроса

С помощью этого вида запроса можно выбрать нужные данные из уже существующей таблицы посредством обычного запроса на выборку, а затем поместить их в новую таблицу, структура которой определяется структурой записей результирующего множества запроса. Таким образом, процесс создания таблицы с помощью запроса состоит из трех шагов:

1. Создать запрос на выборку.
2. Преобразовать запрос на выборку в запрос на изменение, задав место размещения новой таблицы.
3. Выполнить запрос на изменение, тем самым поместив отобранные записи в новую таблицу.

Созданию различных запросов на выборку мы уделили достаточно внимания в *гл. 4 "Отбор и сортировка записей с помощью запросов"*. Оставшиеся два этапа описаны далее на примере запроса на выборку `Холодные клиенты`, созданного в базе данных `Борей` при описании внешнего соединения в *гл. 4*.

1. Откройте запрос `Холодные клиенты` в режиме Конструктора, выделив его имя в списке запросов в окне базы данных и нажав кнопку **Конструктор** (Design).
2. Добавьте в бланке запроса критерий выборки `Is Null` в столбце `КодЗаказа` (OrderID).
3. Сбросьте флажок **Вывод на экран** (Output) для этого поля.

Замечание

Это нужно сделать, чтобы поле не попало в новую таблицу. Поле `КодЗаказа` (OrderID) в таблице `Заказы` (Orders) определено как Счетчик (AutoNumber) и не может иметь значение `Null`. Это свойство наследуется новой таблицей, поэтому если флажок не сбросить, то при выполнении запроса будет выдаваться сообщение об ошибке.

4. Выполните запрос и убедитесь, что отобраны нужные записи.

Преобразование запроса на выборку в запрос на изменение

Чтобы создать таблицу из результатов запроса, использовавшегося для отбора записей в связанных таблицах, его нужно преобразовать из запроса на выборку в запрос на изменение.

Для этого выполните следующее:

1. Выберите команду меню **Запрос | Создание таблицы** (Query | Make Table) или щелкните по стрелке на кнопке **Тип запроса** (Query Type) на панели инструментов **Конструктор запросов** (Query Design) и выберите из списка тип **Создание таблицы** (Make-Table Query). Появится диалоговое окно **Создание таблицы** (Make Table), которое показано на рис. 8.1.

Замечание

Меню **Запрос** (Query) доступно только в режиме Конструктора запроса.

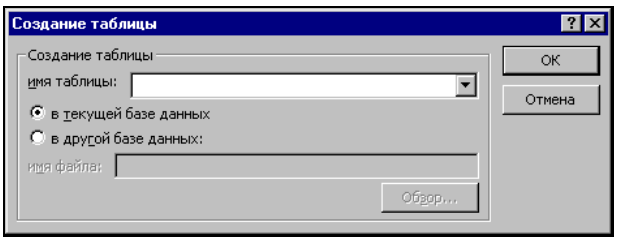


Рис. 8.1. Диалоговое окно **Создание таблицы**

2. В текстовом поле **имя таблицы** (Table Name) введите описательное имя для новой таблицы, например `Список холодных клиентов`.

Внимание!

В базе данных Access не должно быть таблицы и запроса с одинаковыми именами.

В этом диалоговом окне можно определить также, где должна быть создана таблица — в текущей базе данных (переключатель **в текущей базе данных** (Current Database) выбран по умолчанию) или в другой базе данных. В последнем случае нужно выбрать переключатель **в другой базе данных** (Another Database) и ввести в поле **имя файла** (File Name) имя этой базы данных.

3. Нажмите кнопку **OK**. Запрос на выборку преобразуется в запрос на создание таблицы.
4. Нажмите стрелку на кнопке **Вид** (View) панели инструментов и выберите из списка элемент **Режим SQL** (SQL View). Вы увидите инструкцию SQL, которая формируется автоматически, когда вы в графическом режиме создаете запрос. Она отличается от обычной инструкции `SELECT` тем, что содержит еще одно предложение `INTO`.
5. Сохраните и закройте запрос. Теперь в окне базы данных ему соответствует другой значок, который сопровождается восклицательным знаком (рис. 8.2), это означает, что запрос является запросом на изменение.

Выполнение запроса на изменение

Выполнение запроса на изменение приведет к появлению новой таблицы с записями, которые будут идентичны результирующему набору записей того запроса на выборку, который был создан на первом этапе.

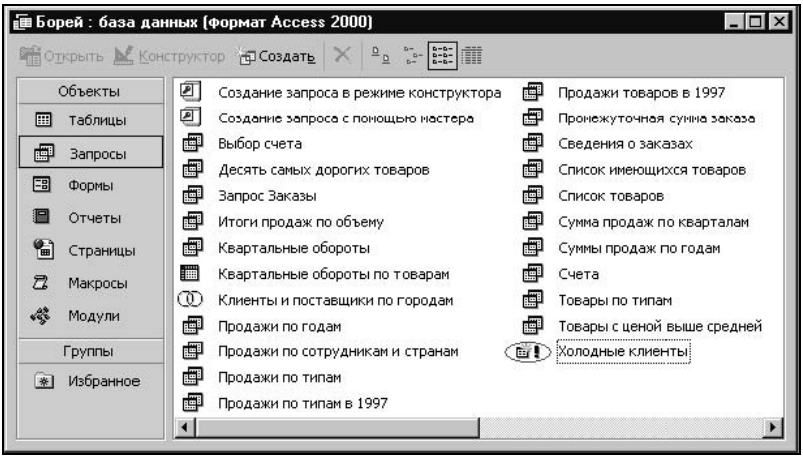


Рис. 8.2. Запрос на изменение в окне базы данных

1. Запустите запрос двойным щелчком мыши по его имени в окне базы данных. При этом он не отображает результирующий набор записей в режиме таблицы, как это делает запрос на выборку, а выполняет нужное действие. Перед его выполнением появляется сообщение, которое предупреждает, что в новую таблицу будут внесены изменения (хотя таблица еще не создана).

Внимание!

Одно из важных правил во время работы с любыми запросами на изменение — нужно обязательно создавать резервную копию таблицы, в которую вносятся изменения. Дело в том, что исправления, внесенные запросами на изменение, необратимы, к тому же часто нелегко обнаружить записи, измененные ошибочным запросом.

2. Подтвердите выполнение операции, нажав кнопку **Да** (Yes). Появится второе сообщение, показывающее число записей, которые будут помещены в новую таблицу в результате выполнения запроса (рис. 8.3).

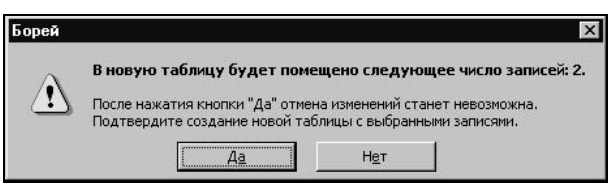


Рис. 8.3. Предупреждающее сообщение о числе новых записей

3. Нажмите кнопку **Да** (Yes). Поскольку запрос запускается впервые, будет создана новая таблица `Список холодных клиентов`.
4. Щелкните по ярлыку **Таблицы** (Tables) в окне базы данных и убедитесь, что в списке появилась новая таблица (рис. 8.4).

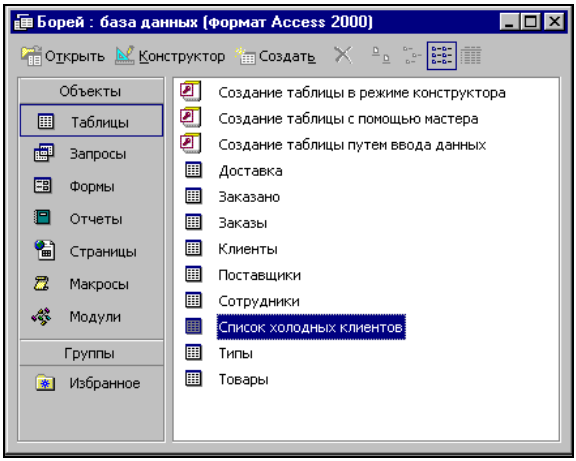


Рис. 8.4. Новая таблица в окне базы данных

5. Дважды щелкните мышью по значку этой таблицы, чтобы ее открыть. Она содержит те же данные, что и запрос на выборку (рис. 8.5).



Рис. 8.5. Таблица, созданная при помощи запроса

Запрос на создание таблицы может использоваться многократно для создания временных таблиц, при этом каждый раз будет создаваться новая таблица и удаляться старая.

Создание запросов на добавление данных

Запрос на добавление записей (Append Query) используется обычно для того, чтобы скопировать данные из одной таблицы в другую. Так же, как и запрос на создание таблицы, он строится на основе запроса на выборку, т. к. сначала нужно отобрать те данные, которые в виде записей будут добавлены в другую таблицу.

Поэтому сначала создается запрос на выборку данных, в который включаются те поля, которые должны составлять добавляемые записи, а также поля, в которых устанавливаются критерии отбора записей.

В качестве примера мы рассмотрим запрос, который будет выбирать из базы данных все заказы за 1996 год и переносить их в другую таблицу, так чтобы таблица `Заказы` (Orders) содержала только актуальные данные. (Демонстрационная база `Борей` со-

держит заказы только за 1996, 1997, 1998 годы, поэтому не удивляйтесь, что в примере используется такой "старый" год.) Для этого сначала создадим таблицу, которая будет хранить устаревшие данные:

1. Раскройте список таблиц в окне базы данных и выделите таблицу `Заказы` (Orders).
2. Скопируйте таблицу в буфер обмена, нажав комбинацию клавиш <Ctrl>+<C>.
3. Вставьте таблицу из буфера обмена, нажав комбинацию клавиш <Ctrl>+<V>. Появится диалоговое окно **Вставка таблицы** (Paste Table As).
4. В группе **Параметры вставки** (Paste Options) выберите переключатель **Только структура** (Structure Only).
5. В поле **имя таблицы** (Table Name) введите строку `Заказы (архив)`. Нажмите кнопку **OK** или клавишу <Enter>.

В списке появляется новая таблица, пока пустая, т. к. мы скопировали только структуру таблицы `Заказы` (Orders). Теперь создадим запрос на добавление, который позволит перенести в новую таблицу данные из таблицы `Заказы` (Orders).

1. Раскройте список таблиц в окне базы данных и выделите в нем таблицу `Заказы`.
2. Щелкните левой кнопкой мыши по стрелке на кнопке **Новый объект** (New Object) панели инструментов **База данных** (Database) и выберите из раскрывшегося списка элемент **Запрос** (Query). В диалоговом окне **Новый запрос** (New Query) выберите значение **Конструктор** (Design View). Появляется окно Конструктора запроса с таблицей `Заказы` на верхней панели.
3. Сначала создайте запрос на выборку нужных записей. Перетащите из списка полей таблицы `Заказы` все поля в строку бланка запроса **Поле** (Field).
4. В строке **Условие отбора** (Criteria) столбца `ДатаРазмещения` введите выражение `Between #01.01.96# And #31.12.96#`.
5. Запустите запрос на выборку, чтобы проверить, правильно ли отобраны записи. Оказались отобранными 152 записи.
6. Выберите команду **Запрос | Добавление** (Query | Append) или щелкните левой кнопкой мыши по стрелке на кнопке **Тип запроса** (Query Type) панели инструментов и выберите из списка элемент **Добавление** (Append Query). Появляется диалоговое окно **Добавление** (Append), аналогичное диалоговому окну **Создание таблицы** (Make Table) (рис. 8.6).

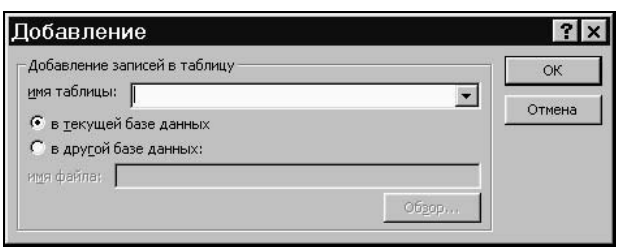


Рис. 8.6. Диалоговое окно **Добавление**

7. В поле **имя таблицы** (Table Name) необходимо выбрать из списка имя таблицы, в которую будут добавляться записи. В данном случае по умолчанию в нем указана таблица `Заказы (архив)`, которую мы только что создали. Нажмите кнопку **OK**. (Если бы таблица, в которую нужно добавить записи находилась в другой базе данных, нужно было бы выбрать переключатель **в другой базе данных** (Another Database) и затем в поле **имя файла** (File Name) указать имя файла MDB, который содержит требуемую таблицу.)
8. В бланке запроса на добавление появляется дополнительная строка **Добавление** (Append To), содержащая названия полей таблицы, в которую добавляются записи (рис. 8.7). В данном случае названия полей запроса и названия полей таблицы совпадают, поэтому по умолчанию вся эта строка заполнена. Если имена полей не совпадают, необходимо выбрать из раскрывающихся списков в строке **Добавление** (Append To) имена поля результирующей таблицы, соответствующие полям в исходной таблице.

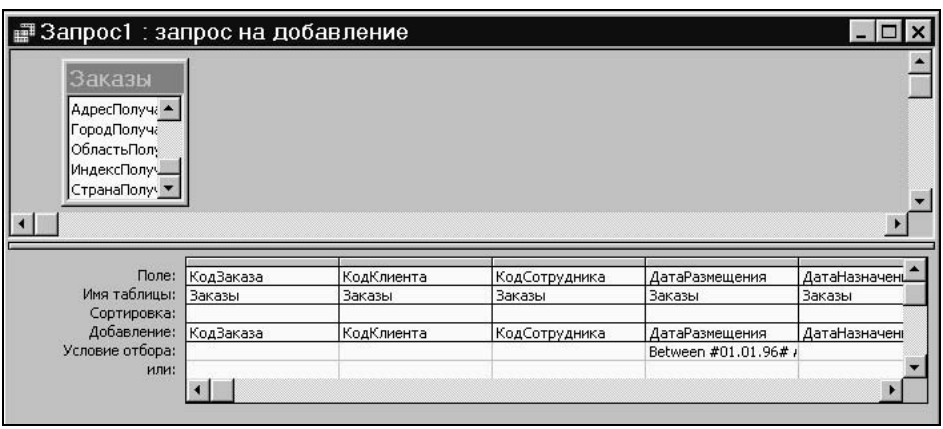


Рис. 8.7. Бланк запроса на добавление

9. Нажмите кнопку **Запуск** (Run), чтобы выполнить запрос. Прежде чем новые записи будут добавлены в таблицу, Access выдает сообщение о числе записей, которые предполагается добавить. Нужно нажать кнопку **Да** (Yes), и только после этого записи будут добавлены. Если нажать кнопку **Нет** (No), вставка записей будет отменена.
10. Сохраните запрос в базе данных, присвоив ему имя `Копирование заказов`. Обратите внимание на значок в списке запросов, соответствующий запросу на добавление. Как и в любом запросе на изменение, он содержит восклицательный знак, но отличается от значка запроса на создание таблицы.

Если записи добавляются в таблицу, которая уже не пуста, то наиболее часто встречающейся ошибкой при выполнении этого запроса является попытка вставить записи, у которых значение первичного ключа совпадает с ключами уже имеющихся в ней записей. Такие записи вставлены не будут, появится только сообщение об их количестве.

Создание запросов на обновление таблиц

Запросы на обновление используются главным образом для того, чтобы внести изменения сразу в большое количество записей с помощью одного запроса. Классическим примером такого запроса является изменение цены в таблице `Товары` (Products) на некоторый постоянный коэффициент, например можно увеличить цену одной из категорий товара на 20%. Чтобы создать такой запрос, выполните следующее:

1. Раскройте список таблиц, щелкнув мышью по ярлыку **Таблицы** (Tables) окна базы данных.
2. Выделите таблицу `Товары` (Products). Щелкните левой кнопкой мыши по стрелке на кнопке **Новый объект** (New Object) на панели инструментов и выберите из списка значение **Запрос** (Query). Появится окно Конструктора запросов с таблицей `Товары` (Products) на верхней панели. Перенесите в бланк запроса поля `Цена` (UnitPrice) и `КодТипа` (CategoryID).
3. Введите условие отбора записей: например, в столбец `КодТипа` (CategoryID) введите значение `1`.
4. Выполните запрос, чтобы убедиться, что отбираются все записи, содержащие напитки.
5. Теперь изменим запрос, превратив его в запрос на обновление. Для этого выполните команду меню **Запрос | Обновление** (Query | Update). Изменяется заголовок запроса и появляется дополнительная строка **Обновление** (Update To). При этом исчезают строки **Сортировка** (Sort) и **Вывод на экран** (Show) (рис. 8.8).

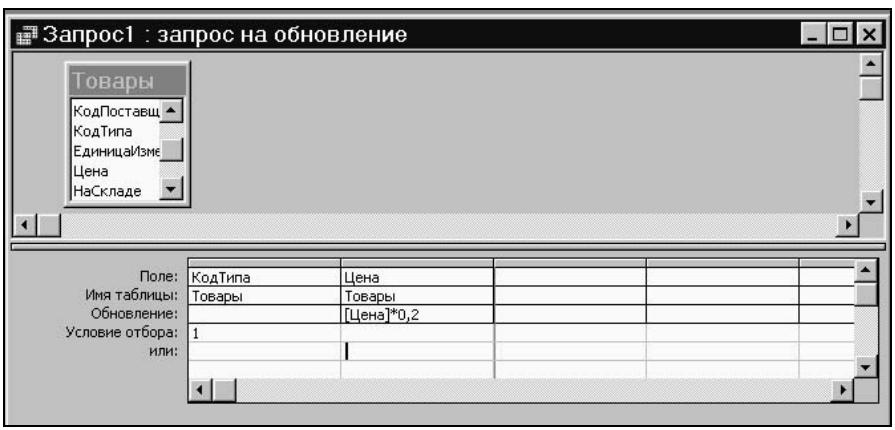


Рис. 8.8. Запрос на обновление записей

6. Теперь нужно в строку **Обновление** (Update To) ввести выражение, по которому будет вычисляться новая цена: `[Цена]*0,2`. В других случаях можно вводить константу, например, если нужно поменять дату во многих записях на текущую.
7. Выполните запрос. Для этого нажмите кнопку **Запуск** (Run) на панели инструментов. Так же, как и при добавлении записей в таблицу, Access выдает сообще-

ние о количестве обновляемых записей и запрашивает подтверждение на обновление. Вы можете подтвердить обновление записей или отвергнуть.

Совет

Прежде чем выполнять запрос на обновление, щелкните левой кнопкой мыши по стрелке на кнопке **Вид** (View) и выберите **Режим таблицы** (Datasheet View). Вы увидите все записи, которые будут обновлены, хотя данные в обновляемых полях будут еще старые. И только убедившись, что отобраны нужные записи, можно выполнять запрос.

В приведенном примере изменялись значения в поле, которое не является первичным ключом в таблице Товары (Products). Особый случай возникает, когда требуется обновить значение первичного ключа в таблице. Если эта таблица связана отношением "один-ко-многим" с другими таблицами, то при изменении первичного ключа записи должны одновременно измениться значения внешних ключей во всех связанных записях подчиненных таблиц. Access обеспечивает выполнение такого изменения автоматически, т. к. поддерживает каскадное обновление записей. При определении связи между таблицами можно установить флажок **каскадное обновление связанных полей** (Cascade Update Related Fields) *(см. разд. "Поддержка целостности данных" гл. 2)*.

Если такой флажок установлен, то при изменении значения первичного ключа в главной таблице Access автоматически выполняет специальный запрос, с помощью которого обновляются внешние ключи всех связанных записей в подчиненной таблице.

Убедиться в том, как работает такой запрос, можно на примере таблиц Сотрудники (Employee) и Заказы (Orders). Для этого нам придется сначала сделать копии этих таблиц и установить связь между ними:

1. Раскройте список таблиц в окне базы данных и выделите таблицу Сотрудники.
2. Скопируйте таблицу в буфер обмена, нажав комбинацию клавиш <Ctrl>+<C>.
3. Вставьте таблицу из буфера обмена, нажав комбинацию клавиш <Ctrl>+<V>. Появится диалоговое окно **Вставка таблицы** (Paste Table As).
4. В поле **Имя таблицы** (Table Name) введите строку Сотрудники (копия). Оставьте предлагаемое по умолчанию значение переключателя в группе **Параметры вставки** (Paste Options). Нажмите кнопку **ОК** или клавишу <Enter>. В списке появится новая таблица.
5. Выполните шаги 1—4 для таблицы Заказы (Orders), создав таблицу Заказы (копия).
6. Для выполнения примера нам придется немного изменить таблицу Заказы (копия). Поле КодСотрудника (EmployeeID) в таблице Заказы (Orders) является полем подстановки, т. е. хотя в этом поле содержатся коды сотрудников, при отображении таблицы в этом поле показываются фамилия и имя сотрудника. Мы можем временно удалить поле подстановки, чтобы видеть результаты каскадного обновления данного поля. Для этого достаточно открыть таблицу Заказы (копия) в режиме Конструктора, выделить строку КодСотрудника (EmployeeID), раскрыть в панели **Свойства поля** (Field Properties) вкладку **Подстановка** (Lookup) и уста-

новить значение свойства **Тип источника строк** (Row Source Type) равным **Поле** (Text box). Сохраните изменение.

7. Еще одно изменение потребуется внести в таблицу `Сотрудники (копия)`. Поле `КодСотрудника` (EmployeeID) в ней имеет тип **Счетчик** (AutoNumber), поэтому Access не позволит изменить его значение. Откройте эту таблицу в режиме Конструктора и замените тип данных для поля `КодСотрудника` (EmployeeID) на **Числовой** (Number).
8. Теперь нужно установить связи между новыми таблицами. Нажмите кнопку **Схема данных** (Relationships) на панели инструментов для вывода окна **Схема данных** (Relationships).
9. Нажмите кнопку **Очистить макет** (Clear Layout), чтобы очистить окно схемы данных. Подтвердите свое намерение в окне сообщения, нажав кнопку **Да** (Yes).
10. Добавьте таблицы `Сотрудники (копия)` и `Заказы (копия)` с помощью диалогового окна **Добавление таблицы** (Show Table). Для этого нажмите соответствующую кнопку на панели инструментов.
11. Чтобы установить связь "один-ко-многим" между таблицами, перетащите поле `КодСотрудника` (EmployeeID) таблицы `Сотрудники (копия)` в соответствующее поле таблицы `Заказы (копия)`. Появится диалоговое окно **Изменение связей** (Edit Relationship).
12. В этом окне необходимо установить флажки **Обеспечение целостности данных** (Enforce Referential Integrity) и **каскадное обновление связанных полей** (Cascade Update Related Fields) (рис. 8.9). Нажмите кнопку **OK**.
13. Закройте окно **Схема данных** (Relationships).

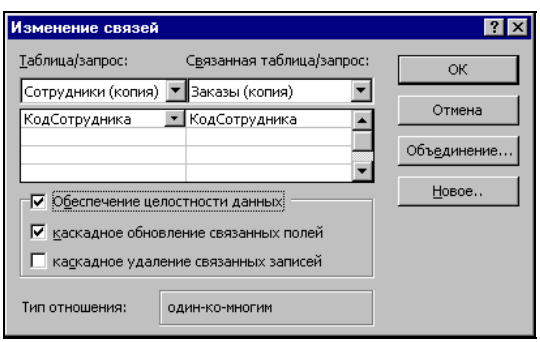
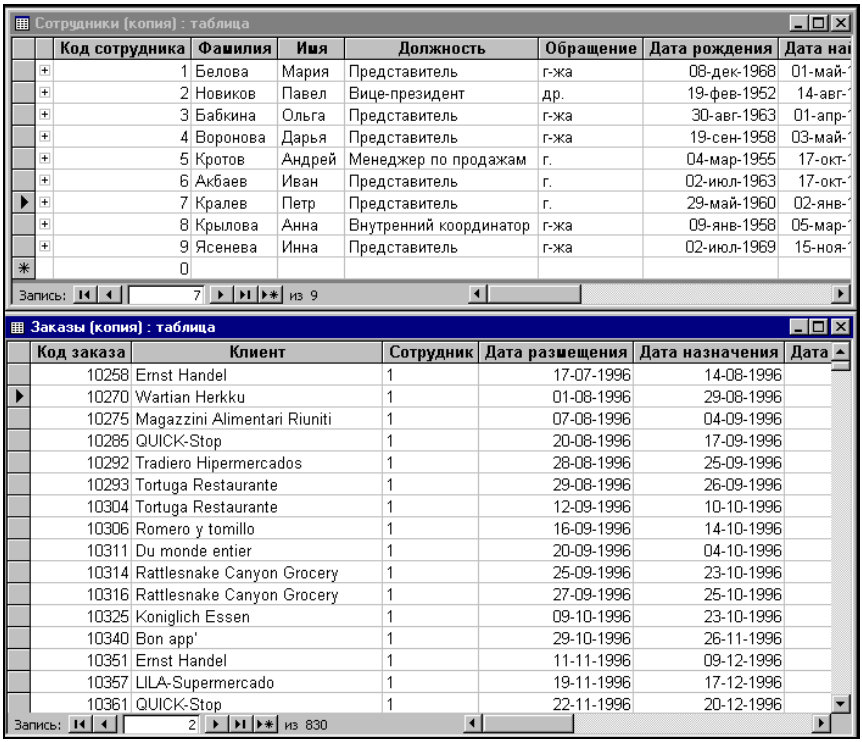


Рис. 8.9. Диалоговое окно **Изменение связей**

Для выполнения каскадного обновления сделайте следующее:

1. Откройте таблицы `Сотрудники (копия)` и `Заказы (копия)`, разместите их на экране так, чтобы были видны обе таблицы (рис. 8.10).
2. Отсортируйте таблицу `Заказы (копия)` по столбцу `Сотрудник` (Employee). Для этого выделите данный столбец, щелкнув левой кнопкой мыши по его заголовку, и нажмите кнопку **Сортировка по возрастанию** (Sort Ascending) на панели инструментов.



Сотрудники (копия) : таблица

Код сотрудника	Фамилия	Имя	Должность	Обращение	Дата рождения	Дата на
1	Белова	Мария	Представитель	г-жа	08-дек-1968	01-май-
2	Новиков	Павел	Вице-президент	др.	19-фев-1952	14-авг-
3	Бабкина	Ольга	Представитель	г-жа	30-авг-1963	01-апр-
4	Воронова	Дарья	Представитель	г-жа	19-сен-1958	03-май-
5	Кротов	Андрей	Менеджер по продажам	г.	04-мар-1955	17-окт-
6	Акбаев	Иван	Представитель	г.	02-июл-1963	17-окт-
7	Кралев	Петр	Представитель	г.	29-май-1960	02-янв-
8	Крылова	Анна	Внутренний координатор	г-жа	09-янв-1958	05-мар-
9	Ясенева	Инна	Представитель	г-жа	02-июл-1969	15-ноя-
0						

Запись: 7 из 9

Заказы (копия) : таблица

Код заказа	Клиент	Сотрудник	Дата размещения	Дата назначения	Дата
10258	Ernst Handel	1	17-07-1996	14-08-1996	
10270	Wartian Herkku	1	01-08-1996	29-08-1996	
10275	Magazzini Alimentari Riuniti	1	07-08-1996	04-09-1996	
10285	QUICK-Stop	1	20-08-1996	17-09-1996	
10292	Tradiero Hipermercados	1	28-08-1996	25-09-1996	
10293	Tortuga Restaurante	1	29-08-1996	26-09-1996	
10304	Tortuga Restaurante	1	12-09-1996	10-10-1996	
10306	Romero y tomillo	1	16-09-1996	14-10-1996	
10311	Du monde entier	1	20-09-1996	04-10-1996	
10314	Rattlesnake Canyon Grocery	1	25-09-1996	23-10-1996	
10316	Rattlesnake Canyon Grocery	1	27-09-1996	25-10-1996	
10325	Koniglich Essen	1	09-10-1996	23-10-1996	
10340	Bon app'	1	29-10-1996	26-11-1996	
10351	Ernst Handel	1	11-11-1996	09-12-1996	
10357	LILA-Supermercado	1	19-11-1996	17-12-1996	
10361	QUICK-Stop	1	22-11-1996	20-12-1996	

Запись: 2 из 830

Рис. 8.10. Каскадное обновление записей

3. Измените значение в столбце `Код сотрудника` (EmployeeID) первой записи таблицы `Сотрудники (копия)`, введя число `10`, и нажмите клавишу <↓>, чтобы перейти к следующей записи. Изменение кода сразу же будет отображено в столбце `Сотрудник` (Employee) таблицы `Заказы (копия)`.

Создание запросов на удаление записей

Запросы на удаление записей позволяют отобрать требуемые записи и удалить их за один прием. Принцип создания такого запроса аналогичен созданию запроса на обновление, однако, удаляя записи из связанных таблиц, необходимо помнить о том, что при этом не должна нарушаться целостность данных.

Если две таблицы связаны отношением "один-ко-многим", нельзя удалять записи из таблицы "один", если в таблице "многие" присутствуют соответствующие им записи. Сначала должны быть удалены записи в таблице "многие" и только потом — соответствующие им записи в таблице "один". Для того чтобы упростить этот процесс, Access позволяет при определении связей между таблицами установить флажок **каскадное удаление связанных записей** (Cascade Delete Related Records). Такой флажок установлен, например, для связи таблиц `Заказы` (Orders) и `Заказано` (Order Details). Действительно, если требуется удалить из базы данных какой-то заказ, должны быть

удалены не только запись об этом заказе в таблице Заказы (Orders), но и все позиции данного заказа в таблице Заказано (Order Details).

Для создания запроса на удаления мы воспользуемся уже имеющимся запросом на добавление записей Копирование заказов.

1. Откройте этот запрос в режиме Конструктора.
2. Чтобы преобразовать запрос на добавление в запрос на удаление записей, выберите команду **Запрос | Удаление** (Query | Delete Query). В бланке запроса появится строка **Удаление** (Delete) (рис. 8.11). В первом столбце строки **Удаление** (Delete) показывается значение **Из** (From), которое указывает, что будут удаляться записи из таблицы Заказы (Orders). Во втором столбце строки **Удаление** (Delete) отображается значение **Условие** (Where), что указывает на использование этого столбца для определения критерия отбора удаляемых записей. В строке **Условие отбора** (Criteria) показывается выражение, которое было в запросе на добавление.

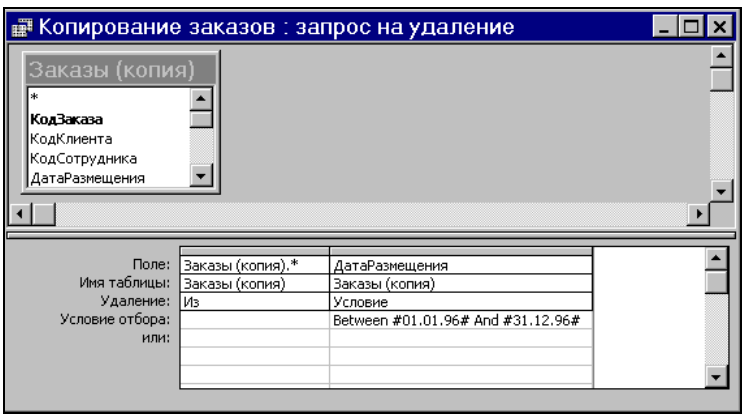


Рис. 8.11. Запрос на удаление записей из таблицы Заказы

3. Нажмите кнопку **Запуск** (Run). Появится сообщение с приглашением подтвердить удаление записей и информация о количестве записей, которые будут удалены. Их будет столько же, сколько было отобрано в соответствующем запросе на добавление. В этот момент еще можно отменить удаление, для чего достаточно нажать кнопку **Нет** (No) в окне сообщения. Нажмите кнопку **Да** (Yes), подтверждая удаление.
4. Будут удалены все заказы за 1996 год из таблицы Заказы (Orders) и все позиции заказов из таблицы Заказано (Order Details). Вы можете убедиться в этом, если откроете таблицу Заказано (Order Details).
5. Сохраните запрос, дав ему имя Удаление заказов.

Однако такое каскадное удаление записей не всегда возможно. Например, для рассмотренной нами в предыдущем разделе связи таблиц Сотрудники (Employees) и Заказы (Orders) нельзя устанавливать флажок **каскадное удаление связанных записей** (Cascade Delete Related Records), т. к. удаление записи о сотруднике из таблицы Сотрудники (Employees) не должно повлечь за собой удаление всех его заказов. Если вы попытаетесь удалить какую-либо запись о сотруднике и в таблице Заказы (Orders)

окажутся заказы, принятые этим сотрудником, Access выдаст сообщение об ошибке (рис. 8.12).

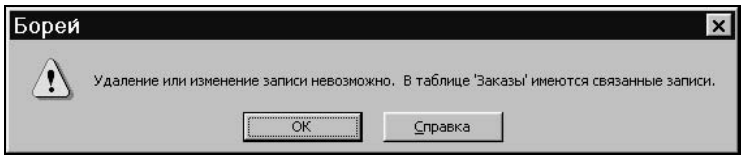


Рис. 8.12. Сообщение о невозможности удаления записей

Замечание

Для того чтобы выполнялась автоматическая проверка целостности данных, не забудьте для всех связей в базе данных установить флажок **Обеспечение целостности данных** (Enforce Referential Integrity).

Анализ данных с помощью запросов

Еще одной функцией запросов Access является анализ данных, которые распределены по разным таблицам. Анализ данных может выполняться с помощью:

- перекрестных запросов, которые позволяют выполнить некоторую обработку таблиц с целью получения сводных данных;
- статистических функций, которые позволяют выполнять различного рода расчеты;
- Мастеров, которые позволяют отыскать в таблице повторяющиеся записи и записи, для которых отсутствуют подчиненные записи в связанной таблице;
- сводных таблиц и сводных диаграмм — аналогов соответствующих средств Excel.

Перекрестные запросы

Перекрестные запросы — это запросы, в которых происходит статистическая обработка данных, результаты которой выводятся в виде таблицы, очень похожей на сводную таблицу Excel. Перекрестные запросы обладают следующими достоинствами:

- возможностью обработки значительного объема данных и вывода их в формате, который очень хорошо подходит для автоматического создания графиков и диаграмм;
- простотой и скоростью создания сложных запросов с несколькими уровнями детализации.

Однако они имеют и недостатки — например, нельзя сортировать таблицу результатов по значениям, содержащимся в столбцах, т. к. в подавляющем большинстве случаев одновременное упорядочивание данных в столбцах по всем строкам невозможно. При этом вы можете задать сортировку по возрастанию или по убыванию для заголовков строк.

Перекрестные запросы удобны для представления данных в виде таблицы, но т. к. такая таблица часто имеет разное количество столбцов, на них довольно сложно строить отчеты. Создание отчета на базе перекрестного запроса будет рассматриваться в *гл. 13*.

В качестве примера сформируем два перекрестных запроса к базе данных Борей для вывода ежемесячных продаж товара (по его коду) и ежеквартальных продаж товара (по его типу). Подобные таблицы, содержащие итоги, которые подводятся через определенные промежутки времени, например через каждый месяц или квартал, обычно используются в качестве данных для графиков.

В описываемой далее пошаговой процедуре создания перекрестного запроса показано, как можно пользоваться выражениями для создания новых вычисляемых полей в запросах.

Итак, мы будем создавать перекрестный запрос, в котором в строках выводятся товары, а в столбцах — соответствующие им ежемесячные объемы продаж. Для этого выполните следующее:

1. Создайте новый запрос в режиме Конструктора и добавьте в него таблицы Товары (Products), Заказы (Orders) и Заказано (Order Details).
2. Перетащите поля КодТовара (ProductID) и Марка (ProductName) таблицы Товары (Products), а затем поле ДатаРазмещения (OrderedDate) таблицы Заказы (Orders) в первые три столбца бланка запроса.
3. Выберите команду меню **Запрос | Перекрестный** (Query | Crosstab Query). Заголовок окна запроса **Запрос1:на выборку** (Query1:Select Query) изменится на **Запрос1:перекрестный запрос** (Query1:Crosstab Query). Кроме того, в бланк запроса будут добавлены строки **Перекрестная таблица** (Crosstab) и **Групповая операция** (Total), в последнюю во всех столбцах автоматически вводится операция **Группировка** (Group By).
4. Выберите в списке ячейки **Перекрестная таблица** (Crosstab) столбца КодТовара (ProductID) значение **Заголовки строк** (Row Heading). Выполните то же самое для столбца Марка (ProductName). Эти столбцы будут заголовками строк перекрестной таблицы.
5. Выберите в списке ячейки **Групповая операция** (Total) столбца ДатаРазмещения (OrderedDate) значение **Условие** (Where). В ячейке **Условие отбора** (Criteria) этого столбца введите выражение `<=#31.12.1997# And >=#01.01.1997#` для вывода в перекрестной таблице данных за 1997 год.

Замечание

Можно вводить только последние две цифры года, первые две цифры будут подставляться автоматически.

6. Установите курсор в ячейку **Поле** (Field) следующего (пустого) столбца и введите выражение:

```
Объем продаж: Sum([Количество]*[Заказано].[Цена])
```

Это будет поле, значение которого вычисляется с помощью описанного выражения, а название поля — Объем продаж. В выражении используются поля из таб-

лиц, которые включены в запрос, однако обратите внимание, что сами поля в запрос не включены. Это означает, что при создании вычисляемого поля в выражение можно включать ссылки не только на поля самого запроса, но и на поля, которые не включаются в результат запроса. Важно, чтобы они были в исходных таблицах. При ссылке на поле `Цена` (Price) в выражении мы указали еще имя таблицы `Заказано` (Order Details), а при ссылке на поле `Количество` не указывали. Указать имя таблицы пришлось потому, что поле с именем `Цена` (Price) присутствует и в таблице `Товары` (Products), и в таблице `Заказано` (Order Details). Если не указать в выражении имени таблицы, Access не сможет определить, из какой таблицы брать значения, поэтому при выполнении запроса выдаст сообщение об ошибке, как это представлено на рис. 8.13.

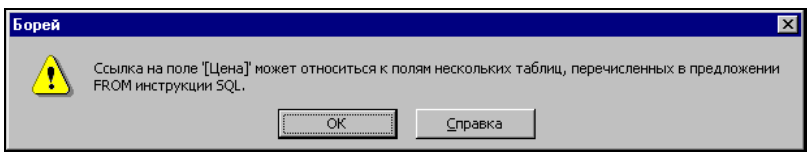


Рис. 8.13. Сообщение об ошибке в выражении вычисляемого поля

7. Выберите в ячейке **Групповая операция** (Total) того же столбца значение **Выражение** (Expression), а затем в ячейке **Перекрестная таблица** (Crosstab) — **Значение** (Value). В столбце `Объем продаж` вычисляется общий объем заказов на каждый товар, который будет подставляться в ячейки результирующей таблицы запроса.
8. Установите курсор в ячейку **Поле** (Field) следующего (пустого) столбца и введите выражение `Format([ДатаРазмещения];"mmm")`. Функция `Format()` в данном случае будет возвращать три первые буквы месяца от значения даты в столбце `ДатаРазмещения` (OrderedDate). С помощью этой функции мы создали еще одно

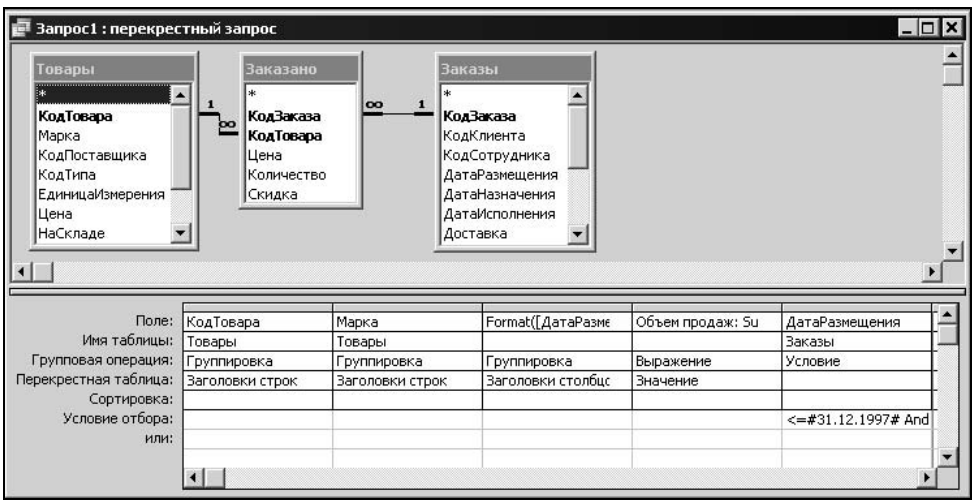


Рис. 8.14. Перекрестный запрос в режиме Конструктора

вычисляемое поле в запросе, причем после окончания ввода этого выражения Access автоматически создает имя этого поля `Выражение1` (Expr1). Так происходит всякий раз, когда мы не определяем имя вычисляемого поля. Выберите из списка в ячейке **Перекрестная таблица** (Crosstab) этого столбца значение **Заголовки столбцов** (Column Heading) (рис. 8.14). Это означает, что данный столбец запроса будет преобразован в процессе выполнения запроса в строку, содержащую заголовки столбцов результирующей таблицы.

9. Нажмите кнопку **Запуск** (Run). Появится результирующее множество записей, представленное на рис. 8.15.

Стоит обратить внимание, что выведенная перекрестная таблица обладает одним недостатком: расположение столбцов определяется алфавитным порядком их заголовков, а не хронологической очередностью. Избежать этой неприятности можно. Для этого необходимо задать порядок сортировки заголовков столбцов. В следующем разделе мы покажем, как это сделать.

Запрос1 : перекрестный запрос

Код товара	Марка	авг	апр	дек	июл	июн	май
1	Genen Shouyu	7 200р.	7 200р.		5 220р.	1 800р.	1 440р
2	Pavlova	19 000р.	2 280р.	6 150р.	1 900р.		
3	Alice Mutton			2 000р.	1 400р.		6 000р
4	Carnarvon Tigers	7 700р.	11 000р.		4 400р.		22 000р
5	Teatime Chocolate Biscu	3 203р.					
6	Sir Rodney's Marmalade	17 500р.					
7	Sir Rodney's Scones	10 500р.	15 000р.	1 400р.	16 500р.		3 000р
8	Gustaf's Knackebrod		13 600р.	2 000р.			
9	Tunnbrod		15 520р.				
10	Guarana Fantastica	5 580р.	4 960р.	6 120р.	24 800р.	1 550р.	620р
11	NuNuCa Nuss-Nougat-Cr	2 100р.		7 850р.	2 940р.	10 500р.	19 530р
12	Gumbar Gummibarchen	13 680р.					17 100р
13	Schoggi Schokolade	780р.	600р.	1 200р.	4 080р.		1 080р

Запись: 1 из 77

Рис. 8.15. Результирующее множество перекрестного запроса

Использование фиксированных заголовков столбцов в перекрестных запросах

Для того чтобы в построенном запросе вывести столбцы в порядке календарной очередности месяцев, нужно задать фиксированные заголовки столбцов.

1. Переключитесь в режим Конструктора и выведите на экран свойства запроса. Для этого нажмите кнопку **Свойства** (Properties) на панели инструментов и щелкните левой кнопкой мыши на свободном поле панели, в которой отображаются таблицы запроса. В диалоговом окне свойств запроса выводится свойство **Заголовки столбцов** (Column Headings), которое имеется только у перекрестных запросов.

2. Введите краткие названия каждого месяца года в поле свойства **Заголовки столбцов** (Column Headings) (рис. 8.16). Необходимо воспроизвести названия так, как они возвращаются функцией `Format()` и выглядят при стандартном отображении заголовков столбцов (см. рис. 8.15), в противном случае данные для неправильно заданных месяцев не отображаются. Названия требуется заключать в кавычки и

разделять символом точки с запятой. Пробелы не нужны. По завершении ввода всех 12 месяцев закройте окно свойств.

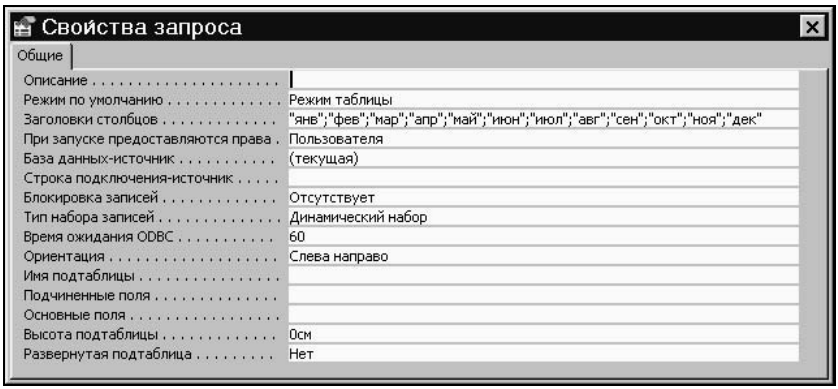


Рис. 8.16. Задание значения свойства **Заголовки столбцов**

3. Перейдите в режим просмотра SQL (рис. 8.17). Обратите внимание на использование выражений в вычисляемых полях и критерии запроса, а также на использование ключевых слов `TRANSFORM` и `PIVOT`.

   Инструкция SQL содержит операцию `TRANSFORM`, в которой определяются данные, содержащиеся в таблице. В операции `PIVOT` задаются заголовки столбцов. После ключевого слова `IN` указываются фиксированные названия столбцов, если они есть.

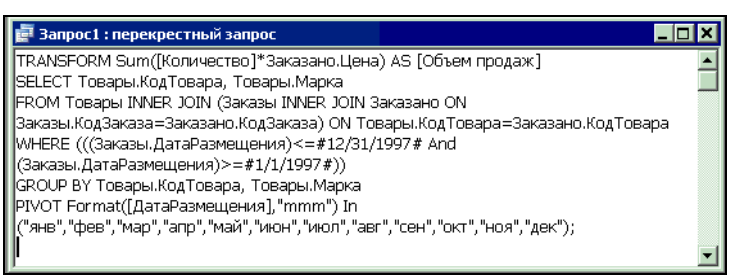


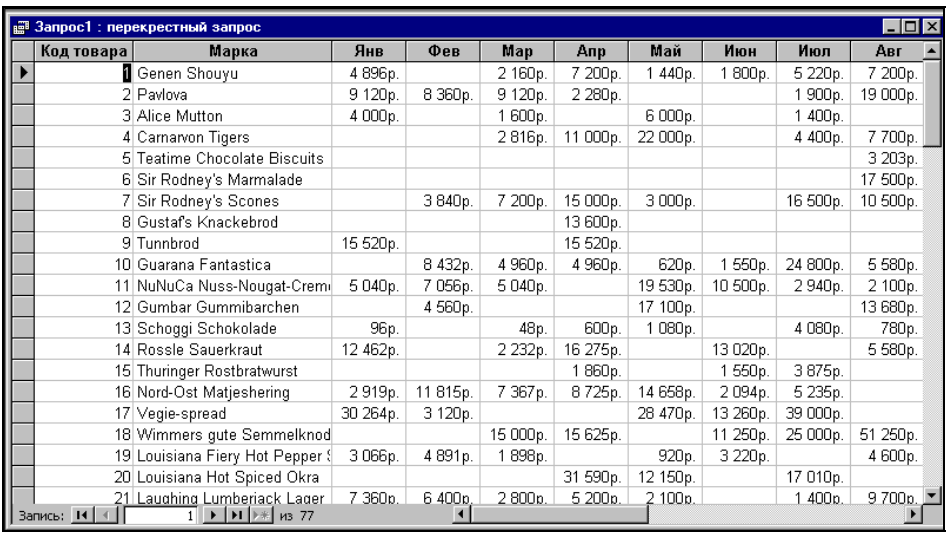
Рис. 8.17. Перекрестный запрос на SQL

Замечание

В ANSI SQL слова `TRANSFORM` и `PIVOT` не являются зарезервированными, а ключевое слово `IN` интерпретируется по-разному в Access и ANSI SQL. Обратите также внимание на формат даты в инструкции SQL: сначала пишется номер месяца, а потом число месяца, разделителем является косая черта, и все выражение содержится между значками "#". Именно в таком формате должны быть всегда написаны даты в предложении SQL, хотя, как вы видели, в бланке запроса можно писать даты в привычном нам европейском формате. Нужное преобразование Конструктор выполняет автоматически.

4. Нажмите кнопку **Запуск** (Run) для вывода результатов запроса. Теперь заголовки столбцов перекрестной таблицы соответствуют календарной последовательности (рис. 8.18).

5. Выберите команду **Файл | Сохранить как** (File | Save As) и сохраните запрос под именем `Ежемесячная выручка от продаж в 1997 году по типам`.



Запрос1 : перекрестный запрос

Код товара	Марка	Янв	Фев	Мар	Апр	Май	Июн	Июл	Авг
1	Genen Shouyu	4 896р.		2 160р.	7 200р.	1 440р.	1 800р.	5 220р.	7 200р.
2	Pavlova	9 120р.	8 360р.	9 120р.	2 280р.			1 900р.	19 000р.
3	Alice Mutton	4 000р.		1 600р.		6 000р.		1 400р.	
4	Carnarvon Tigers			2 816р.	11 000р.	22 000р.		4 400р.	7 700р.
5	Teatime Chocolate Biscuits								3 203р.
6	Sir Rodney's Marmalade								17 500р.
7	Sir Rodney's Scones		3 840р.	7 200р.	15 000р.	3 000р.		16 500р.	10 500р.
8	Gustaf's Knackebrod				13 600р.				
9	Tunnbrod	15 520р.			15 520р.				
10	Guarana Fantastica		8 432р.	4 960р.	4 960р.	620р.	1 550р.	24 800р.	5 580р.
11	NuNuCa Nuss-Nougat-Crem	5 040р.	7 056р.	5 040р.		19 530р.	10 500р.	2 940р.	2 100р.
12	Gumbar Gummibarchen		4 560р.			17 100р.			13 680р.
13	Schoggi Schokolade	96р.		48р.	600р.	1 080р.		4 080р.	780р.
14	Rossle Sauerkraut	12 462р.		2 232р.	16 275р.		13 020р.		5 580р.
15	Thuringer Rostbratwurst				1 860р.		1 550р.	3 875р.	
16	Nord-Ost Matjeshering	2 919р.	11 815р.	7 367р.	8 725р.	14 658р.	2 094р.	5 235р.	
17	Vegie-spread	30 264р.	3 120р.			28 470р.	13 260р.	39 000р.	
18	Wimmers gute Semmelknod			15 000р.	15 625р.		11 250р.	25 000р.	51 250р.
19	Louisiana Fiery Hot Pepper S	3 066р.	4 891р.	1 898р.		920р.	3 220р.		4 600р.
20	Louisiana Hot Spiced Okra				31 590р.	12 150р.		17 010р.	
21	Laughing Lumberjack Lager	7 360р.	6 400р.	2 800р.	5 200р.	2 100р.		1 400р.	9 700р.

Запись: 1 из 77

Рис. 8.18. Перекрестная таблица с фиксированными заголовками столбцов

Изменение уровня детализации в запросе

Чтобы создать удобное для анализа представление или наглядный график, можно сократить число строк и столбцов в результатах построенного запроса. С этой целью создадим перекрестный запрос, который выводит ежеквартальную выручку от продаж по типам товаров:

1. Добавьте в уже созданный запрос таблицу `Типы` (Categories).

2. Замените столбцы `КодТовара` (ProductID) и `Марка` (ProductName) в запросе столбцами `КодТипа` (CategoryID) и `Категория` (CategoryName) таблицы `Типы` (Categories). Для этого сначала удалите лишние столбцы, а затем перенесите нужные столбцы в бланк заказа. В ячейках **Перекрестная таблица** (Crosstab) обоих столбцов выберите значение **Заголовки строк** (Row Heading) (рис. 8.19).

3. Измените выражение в столбце `Выражение1` (Exp1):

```
Format([ДатаРазмещения];"""Квартал ""q"),
```

которое определяет заголовки столбцов `Квартал 1`, `Квартал 2`, `Квартал 3`, `Квартал 4`. Несколько кавычек требуется для указания того, что слово `Квартал` и пробел рассматриваются как строка, а `q` — как формат.

4. Откройте бланк свойств и очистите содержимое свойства **Заголовки столбцов** (Column Headings), а потом закройте бланк. Если не удалить фиксированные заголовки, то в перекрестной таблице не будет данных. Чтобы ускорить работу запроса, можно задать в поле свойства **Заголовки столбцов** (Column Headings) четыре фиксированных названия:

```
"Квартал 1", "Квартал 2", "Квартал 3", "Квартал 4".
```

5. Нажмите кнопку **Запуск** (Run) для вывода результатов запроса (рис. 8.20).

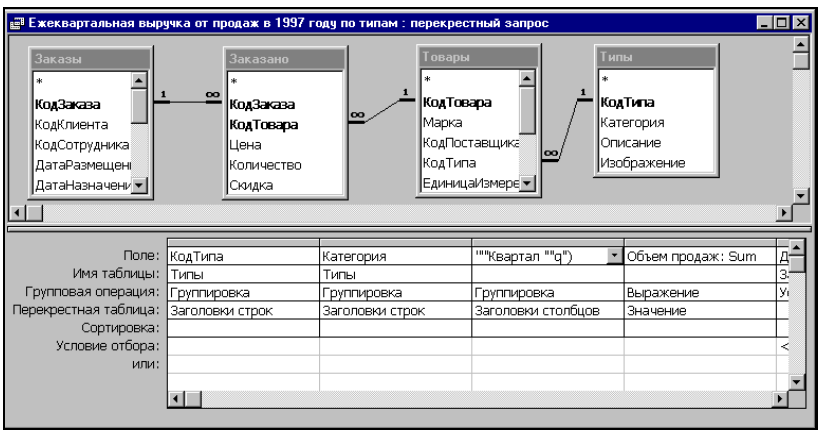


Рис. 8.19. Запрос для вывода выручки от продаж по типам товаров

Ежеквартальная выручка от продаж в 1997 году по типам : перекрестный запрос

Код типа	Категория	Квартал 1	Квартал 2	Квартал 3	Квартал 4
1	Напитки	219 910р.	226 066р.	303 398р.	397 057р.
2	Приправы	168 283р.	236 849р.	267 278р.	358 358р.
3	Кондитерские изделия	150 020р.	162 787р.	217 526р.	290 195р.
4	Молочные продукты	127 509р.	197 542р.	184 162р.	138 813р.
5	Хлебобулочные изделия	80 880р.	128 562р.	171 264р.	191 874р.
6	Мясо/птица	144 565р.	176 199р.	132 001р.	182 699р.
7	Фрукты	86 090р.	80 813р.	74 913р.	101 934р.
8	Рыбопродукты	501 542р.	342 389р.	313 334р.	310 041р.

Запись: 1 из 8

Рис. 8.20. Перекрестная таблица с итогами по кварталам

Использование статистических функций для выполнения расчетов

Статистические функции используются в запросах главным образом для вычисления всевозможных итоговых значений, например для числового поля таблицы можно вычислить среднее значение или сумму значений для всех или отобранных записей, можно посчитать количество записей, возвращаемых запросом.

Статистические функции, которые можно использовать в запросах Access, входят в состав Access SQL (Jet SQL). Описание этих функций, а также типов полей, к которым они могут быть применены, приведено в табл. 8.1.

Таблица 8.1. *Статистические функции Access SQL*

Функция	Типы полей	Описание
`Avg()`	Все типы полей, исключая **Текстовый** (Text), **Поле Memo** (Memo) и **Поле объекта OLE** (OLE Object)	Вычисляет арифметическое среднее набора чисел, содержащихся в указанном поле запроса
`Count()`	Все типы полей	Вычисляет количество непустых записей, возвращаемых запросом
`First()`	Все типы полей	Возвращает значение поля из первой записи результирующего набора
`Last()`	Все типы полей	Возвращает значение поля из последней записи результирующего набора
`Max()`	Все типы полей, исключая **Текстовый** (Text), **Поле Memo** (Memo) и **Поле объекта OLE** (OLE Object)	Возвращает максимальное значение из набора, содержащегося в указанном поле
`Min()`	Все типы полей, исключая **Текстовый** (Text), **Поле Memo** (Memo) и **Поле объекта OLE** (OLE Object)	Возвращает минимальное значение из набора, содержащегося в указанном поле
`StDev()`	Все типы полей, исключая **Текстовый** (Text), **Поле Memo** (Memo) и **Поле объекта OLE** (OLE Object)	Возвращает значение среднеквадратичного отклонения, вычисляемого по набору значений, содержащихся в указанном поле
`Sum()`	Все типы полей, исключая **Текстовый** (Text), **Поле Memo** (Memo) и **Поле объекта OLE** (OLE Object)	Возвращают сумму набора значений, содержащихся в заданном поле
`Var()`	Все типы полей, исключая **Текстовый** (Text), **Поле Memo** (Memo) и **Поле объекта OLE** (OLE Object)	Возвращает значение дисперсии, вычисляемой по набору значений, содержащихся в указанном поле

Перечисленных функций обычно достаточно для решения большинства задач. Тем не менее, при необходимости пользователь может самостоятельно написать функцию на языке VBA, которая будет выполнять сложные статистические, научные или инженерные расчеты над данными, и использовать эту функцию в запросах.

Замечание

Все статистические функции Access SQL поддерживаются стандартом ANSI SQL, но не все из них поддерживаются другими СУБД. Например, не все серверные базы данных поддерживают функции `StDev()` и `Var()`. Поэтому, если вы планируете в

дальнейшем перенести свою базу данных на сервер, прежде чем использовать эти функции, нужно убедиться, поддерживаются ли они сервером. Иначе потом придется изменять все объекты, в которых используются эти функции.

Использование статистических функций для расчета итоговых значений тесно связано с применением групповых операций в запросе. Групповые операции позволяют задать группы, для которых выполняются вычисления. Далее мы приведем примеры таких расчетов, как на всем множестве записей, так и на подмножествах, выбранных по условиям запроса.

Вычисления на всем диапазоне записей

Первый запрос, который мы предлагаем построить, должен ответить на вопрос: "Сколько заказов и на какую сумму разместил каждый клиент?" Для этого нам придется построить многотабличный запрос. Воспользуемся уже существующим в базе данных `Борей` запросом `Сведения о заказах` (Oreder Details Extended), а также таблицами `Клиенты` (Customers) и `Заказы` (Orders):

1. Создайте новый запрос и добавьте в него таблицы `Клиенты` (Customers), `Заказы` (Orders) и запрос `Сведения о заказах` (Oreder Details Extended).
2. Перетащите в бланк заказа поле `Название` (CompanyName) из таблицы `Клиенты` (Customers), поле `КодЗаказа` (OrderID) из таблицы `Заказы` (Orders), а затем поле `Отпускная цена` (ExtendedPrice) из запроса `Сведения о заказах` (Oreder Details Extended).
3. Нажмите кнопку **Групповые операции** (Totals) на панели инструментов. В бланк запроса добавляется строка **Групповая операция** (Total), содержащая по умолчанию в каждой ячейке операцию **Группировка** (Group By).
4. Теперь нужно в строке **Групповая операция** (Total) задать необходимые статистические функции. Для поля `Название` (CompanyName) оставьте значение **Группировка** (Group By), для поля `КодЗаказа` (OrderID) выберите из списка функцию `Count`, а для поля `Отпускная цена` — `Sum` (рис. 8.21).

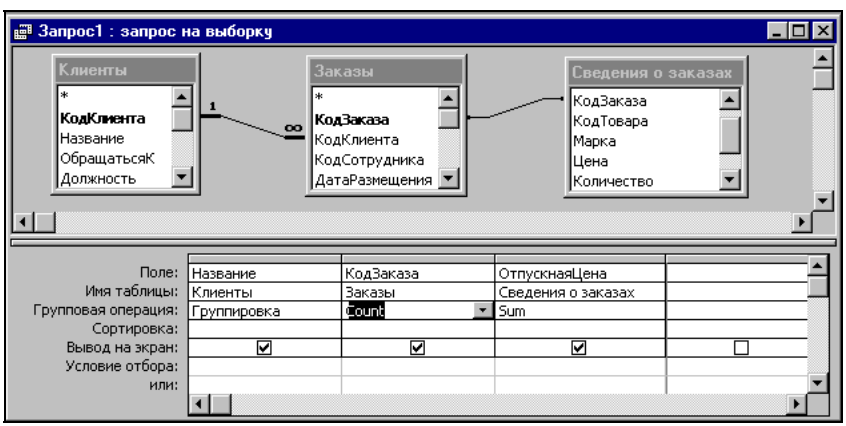
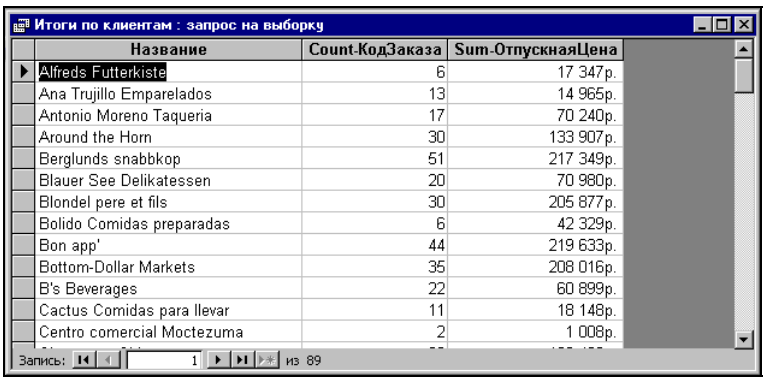


Рис. 8.21. Использование статистических функций в запросе

5. Нажмите кнопку **Запуск** (Run) на панели инструментов, чтобы посмотреть результаты запроса. Вы получите таблицу, которая содержит список клиентов, и для каждого клиента указывается количество заказов, которое он сделал, и общая сумма этих заказов (рис. 8.22).
6. Закройте запрос, сохранив его под именем `Итоги по клиентам`.



Итоги по клиентам : запрос на выборку

Название	Count-КодЗаказа	Sum-ОтпускнаяЦена
Alfreds Futterkiste	6	17 347р.
Ana Trujillo Emparelados	13	14 965р.
Antonio Moreno Taqueria	17	70 240р.
Around the Horn	30	133 907р.
Berglunds snabbkop	51	217 349р.
Blauer See Delikatessen	20	70 980р.
Blondel pere et fils	30	205 877р.
Bolido Comidas preparadas	6	42 329р.
Bon app'	44	219 633р.
Bottom-Dollar Markets	35	208 016р.
B's Beverages	22	60 899р.
Cactus Comidas para llevar	11	18 148р.
Centro comercial Moctezuma	2	1 008р.

Запись: 1 из 89

Рис. 8.22. Запрос `Итоги по клиентам`

Вычисления на выделенных записях таблицы

В предыдущем примере расчеты производились над всеми имеющимися в таблице `Заказы` (Orders) записями. Однако часто требуется выполнить вычисления над определенным набором данных и, основываясь на них, составить статистику. Для этого нужно добавить в запрос условия для выборки нужного набора записей. Например, мы можем изменить предыдущий запрос таким образом, чтобы получить данные не по всем клиентам, а только по тем, которые расположены в Германии. Для этого выполните следующее:

1. Откройте запрос `Итоги по клиентам`, который мы построили в предыдущем разделе, в режиме Конструктора.
2. Перетащите поле `Страна` (Country) из таблицы `Клиенты` (Customers) в бланк запроса. В строке **Групповая операция** (Total) для этого поля появляется элемент **Группировка** (Group by).
3. Замените элемент **Группировка** (Group by) на элемент **Условие** (Where), выбрав его из списка (рис. 8.23). При этом автоматически сбрасывается флажок (Show) в данном столбце. Это означает, что в результат запроса поле `Страна` (Country) не попадет. Если вы попытаетесь установить этот флажок, то при выполнении запроса будет выведено сообщение об ошибке (рис. 8.24). Действительно, если мы хотим, чтобы поле `Страна` (Country) отобразилось в результате запроса, его нужно добавить в бланк запроса второй раз, при этом в строке **Групповая операция** (Total) данного столбца нужно оставить элемент **Группировка** (Group by) и установить флажок **Вывод на экран** (Show).
4. Введите слово `Германия` в ячейку **Условие отбора** (Criteria) первого столбца `Страна` (того, который не выводится на экран).

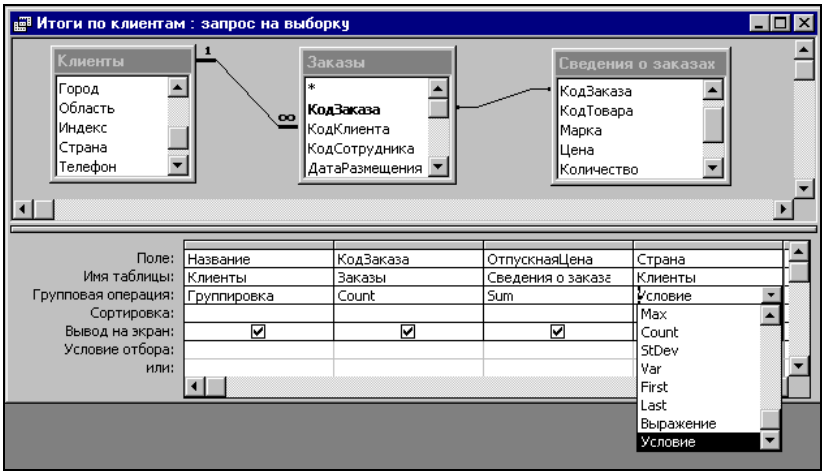


Рис. 8.23. Задание условия отбора записей в запросе `Итоги по клиентам`

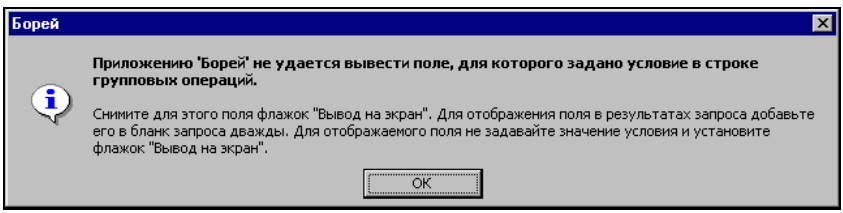


Рис. 8.24. Сообщение об ошибке при попытке отобразить поле, для которого задано условие

5. Выберите команду **Вид | Режим SQL** (View | SQL View) или щелкните по стрелке на кнопке **Вид** (View) и выберите значение **Режим SQL** (SQL View) для вывода инструкции SQL. Она будет выглядеть следующим образом:

```
SELECT Клиенты.Название, Count(Заказы.КодЗаказа) AS [CountOfКодЗаказа],
Sum([Сведения о заказах].ОтпускнаяЦена) AS [SumOfОтпускнаяЦена],
Клиенты.Страна, Клиенты.Страна
FROM Клиенты INNER JOIN ([Сведения о заказах] INNER JOIN Заказы
ON [Сведения о заказах].КодЗаказа = Заказы.КодЗаказа) ON
Клиенты.КодКлиента = Заказы.КодКлиента
WHERE (((Клиенты.Страна)="Германия"))
GROUP BY Клиенты.Название, Клиенты.Страна;
```

   В ней используются две статистические функции — `Count ()` и `Sum()` — и предложение `WHERE`, содержащее условие отбора записей.

6. Измените это предложение, заменив названия полей, в которых ведутся расчеты:

```
AS [Число заказов] вместо AS [CountOfКодЗаказа],
AS [Общая сумма] вместо AS [SumOfОтпускнаяЦена].
```

7. Нажмите на панели инструментов кнопку **Запуск** (Run) для вывода результата (рис. 8.25).



Рис. 8.25. Результаты запроса `Итоги по клиентам в Германии`

Статистические функции можно использовать также в вычисляемых полях запроса. Например, для создания запроса `Итоги по клиентам` мы могли бы вместо запроса `Сведения о заказах` использовать прямо таблицу `Заказано` (Order Details). На рис. 8.26 показано вычисляемое поле `Стоимость`. В выражении для этого поля включена функция `Sum()`:

```
Стоимость: Sum(CCur([Заказано].[Цена]*[Количество]*(1-[Скидка])/100)*100)
```

Тогда в строке **Групповая операция** (Total) для этого столбца должен быть выбран элемент **Выражение** (Expression). При использовании группировки по полю `КодЗаказа` (OrderID), поле `Стоимость` будет содержать общую стоимость заказа.

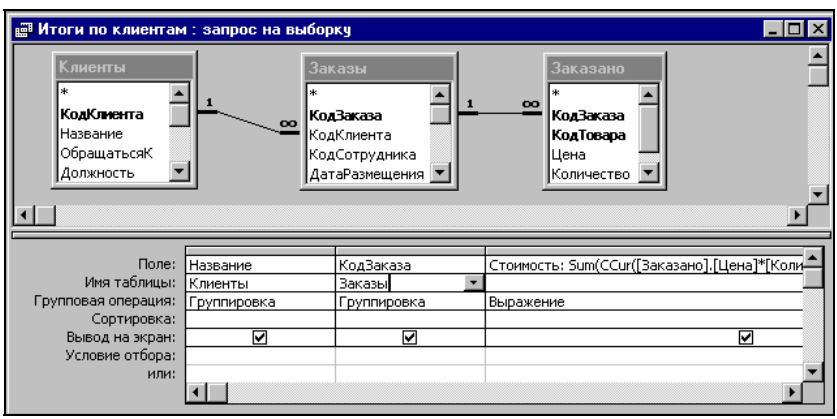


Рис. 8.26. Использование статистической функции в выражении для вычисляемого поля

Можно создать и другое вычисляемое поле:

```
Стоимость: CCur([Заказано].[Цена]*[Количество]*(1-[Скидка])/100)*100,
```

а функцию `Sum()` использовать в строке **Групповая операция** (Total). Оба варианта будут правильны.

Можно изменить этот запрос таким образом, чтобы посчитать общее число клиентов, заказов и общую сумму всех заказов. В этом случае результатом запроса будет одна строка, содержащая итоговые значения по всем полям. К сожалению, нельзя построить запрос, результатом которого была бы таблица, содержащая как детальные, так и итоговые данные, как это обычно бывает в таблицах Excel. Такое представление данных можно организовать только в сводных таблицах.

Внимание!

Статистические функции при вычислениях не обрабатывают записи, которые имеют значение `Null` в том поле, к которому они применяются. Поэтому нужно быть осторожным при использовании функции `Count()`. Она будет считать только те записи, которые не содержат пустых значений. Если нужно сосчитать все записи в результирующем наборе или группе, используйте функцию `Nz(<имя поля>)`, которая преобразует пустое значение в 0, и только потом суммируйте.

Отбор повторяющихся записей и записей, не имеющих подчиненных

Типичной операцией, которую приходится выполнять в базе данных, является поиск дубликатов, т. е. повторяющихся записей в таблице. Например, можно выяснить, не встречается ли в таблице `Клиенты` (Customers) одна и та же фирма несколько раз. При этом необходимо сначала определить, какие записи считать дубликатами. В данном случае дубликатами мы будем считать организации с одинаковыми названиями и адресами. Можно создать запрос, который найдет все такие записи. Для этого проще всего воспользоваться соответствующим Мастером:

1. Раскройте список запросов в окне базы данных и нажмите кнопку **Создать** (New). В диалоговом окне **Новый запрос** (New Query) выберите из списка элемент **Повторяющиеся записи** (Find Duplicates Query Wizard). Появится диалоговое окно Мастера (рис. 8.27).

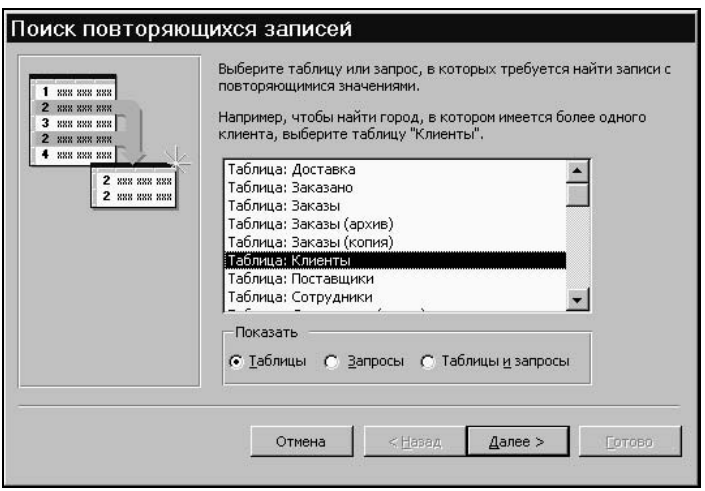


Рис. 8.27. Диалоговое окно Мастера поиска повторяющихся записей

2. Выберите из списка таблицу Клиенты (Customers) (по умолчанию в окне отображается именно список таблиц, но можно установить флажок и получить список запросов или таблиц и запросов вместе).
3. Следуйте указаниям Мастера (они подробно описаны в диалоговых окнах), и в результате вы получите запрос, который в режиме Конструктора может выглядеть так, как представлено на рис. 8.28.
4. Данный запрос имеет сложное выражение в строке **Условие отбора** (Criteria) поля Название (CompanyName):

```
In (SELECT [Название] FROM [Клиенты] As Tmp GROUP BY
[Название],[Адрес] HAVING Count(*)>1 And [Адрес] =
[Клиенты].[Адрес])
```

Это выражение включает еще один запрос. Такой запрос называется *подчиненным* запросом *(подробную информацию о подчиненных запросах см. в разд. "Создание подчиненных запросов" данной главы)*. Чтобы посмотреть, как полностью выглядит инструкция SQL, соответствующая этому запросу, нужно перейти в режим SQL.

Совет

Мы рекомендуем при работе с запросами в режиме Конструктора чаще открывать окно **Режим SQL** (SQL View). Изучая, как Access создает инструкции SQL, можно постепенно выучить этот язык. Правда, помните, что в данном случае мы имеем дело с диалектом стандартного языка SQL — Access (Jet) SQL. Далее мы расскажем, чем он отличается от стандартного языка SQL *(см. разд. "Сравнение ANSI и Jet SQL" данной главы)*.

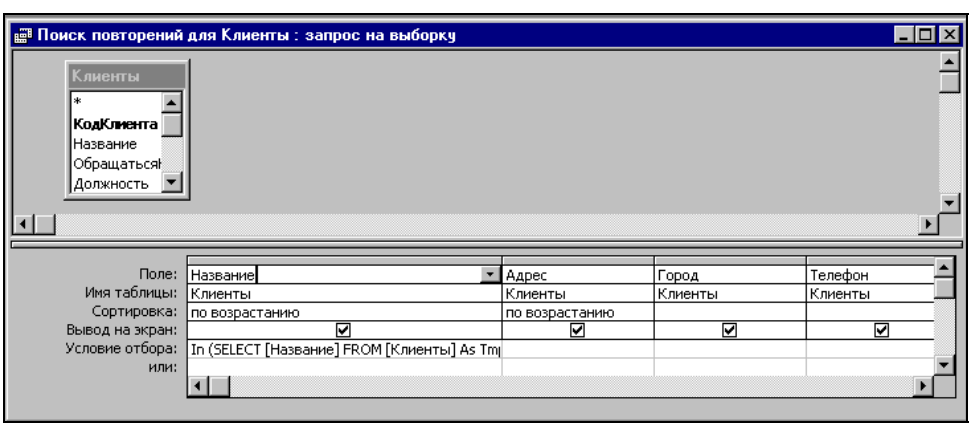


Рис. 8.28. Запрос, отбирающий повторяющиеся записи в таблице Клиенты

Такой поиск разумно выполнять также в таблицах, которые экспортированы из другой среды, например Excel. Эти таблицы могут содержать дублирующие записи, которые нужно найти и удалить. Для удаления дубликатов можно воспользоваться процедурой, которая описана в справке Access в разделах **Создание баз данных и объектов и работа с ними**, **Объекты базы данных**, **Работа с запросами**, **Создание за-**

просов, Работа с запросами на изменение, **Удаление повторяющихся записей из таблицы** (Creating and Working with Databases and Objects, Database Objects, Queries, Creating Queries, Action Queries, Delete duplicate records from a table).

Еще один Мастер позволяет найти все записи в главной таблице, которые не имеют связанных записей в подчиненной таблице (обратной ситуации быть не должно, если вы правильно установили флажки, обеспечивающие целостность данных) *(см. также разд. "Поддержка целостности данных" гл. 2)*.

Чтобы создать запрос, который покажет всех клиентов, не имеющих заказов, выполните следующее:

1. Раскройте список запросов в окне базы данных и нажмите кнопку **Создать** (New). В диалоговом окне **Новый запрос** (New Query) выберите из списка элемент **Записи без подчиненных** (Find Unmatched Query Wizard). Появится диалоговое окно **Поиск записей, не имеющих подчиненных** (Find Unmatched Query Wizard), аналогичное окну **Поиск повторяющихся записей** (см. рис. 8.27).
2. Выберите из списка таблицу `Клиенты` (Customers).
3. Следуйте указаниям Мастера, в результате чего вы получите запрос, который в режиме Конструктора может выглядеть так, как представлено на рис. 8.29.

В бланке запроса в столбце `КодКлиента` (CustomerID) задано условие — `Is Null`. Это означает, что выбираются все клиенты, код которых отсутствует в таблице `Заказы` (Orders). Вы можете использовать это условие всякий раз, когда нужно отыскать записи, у которых одно из полей пусто.

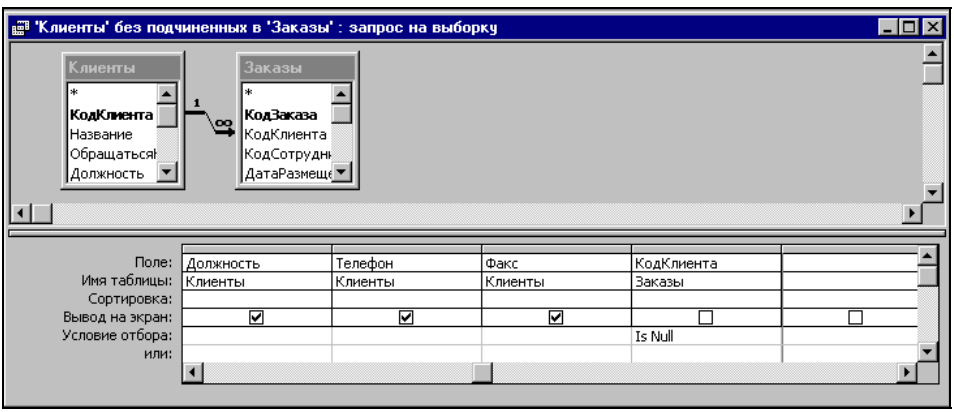


Рис. 8.29. Запрос для поиска записей, не имеющих подчиненных записей

Сводные таблицы

Сводная таблица — это еще одно удобное средство для анализа данных. Они позволяют превратить обычную таблицу или результирующее множество запроса, содержащее большое число записей и непригодное для анализа, в компактную таблицу, включающую только итоговые данные. Причем, в отличие от перекрестных запросов, структура сводной таблицы легко трансформируется, позволяя просматривать

данные в различных разрезах и с различной степенью детализации, что и требуется для анализа. Кроме того, данные, представленные в табличной форме, могут быть легко преобразованы в графическую форму с помощью сводных диаграмм, что делает их еще более наглядными.

Замечание

Режим **Сводная таблица** (Pivot Table) может быть применен и к таблице, и к запросу, и к форме. Но, на наш взгляд, он наиболее применим именно к запросу, который обычно объединяет данные из нескольких таблиц и эти данные нужно иметь возможность представить в удобном для анализа виде. Поэтому работу со сводными таблицами и диаграммами мы описываем в этой главе. Но вы можете создавать их конечно не только для запросов.

Создание сводной таблицы

Для того чтобы продемонстрировать возможности сводных таблиц, рассмотрим запрос `Счета` (Invoices). Раскройте список запросов и откройте этот запрос в режиме Таблицы. Вы увидите таблицу, которая содержит свыше двух тысяч записей и более двух десятков полей. Теперь попробуем представить этот запрос в виде сводной таблицы.

1. Щелкните по стрелке на кнопке **Вид** (View) панели инструментов и выберите из списка элемент **Сводная таблица** (PivotTable View). На экране появится макет будущей таблицы, который состоит из четырех областей. Каждая область имеет надпись, которая показывает, какие данные должны в ней помещаться (рис. 8.30).
2. Чтобы удобнее было создавать таблицу, необходимо отобразить на экране список полей запроса. Этот список показывается в специальном окне **Список полей сводной таблицы** (PivotTable Field List) (рис. 8.30, справа). Если он не виден на вашем экране, нажмите кнопку **Поля** (Field List) на панели инструментов **Сводные таблицы** (Pivot Tables). Теперь требуется просто перетащить мышью поля в соответствующие области таблицы.
3. Найдите в списке поле `Страна` (Country) и переместите его в верхнюю область таблицы. Она называется *областью фильтра*, после чего вы сможете фильтровать данные в таблице по странам клиентов. В сводной таблице появляется еще одно измерение — глубина, — благодаря которому вы можете просматривать и анализировать данные по странам. Каждый срез таблицы будет отображать данные по выбранной стране.
4. Выделите в списке полей поле `Марка` (ProductName) и перетащите его в крайнюю левую область. Эта область называется *областью строк* и будет содержать список всех товаров, на которые были выписаны счета. Заметьте, что этот список по умолчанию упорядочивается по наименованию товара и последней строкой является строка `Общие итоги` (Grand Total), содержащая общий итог по всем строкам таблицы.
5. Чтобы определить столбцы таблицы, переместите поле `Дата размещения по месяцам` (OrderDate By Month) в область столбцов, которая имеет надпись **Перетащите сюда поля столбцов** (Drop Column Fields Here). В таблице появятся три столбца, которые будут содержать данные по годам — это самый верхний уровень группировки, — и один итоговой столбец `Общие итоги` (Grand Total).

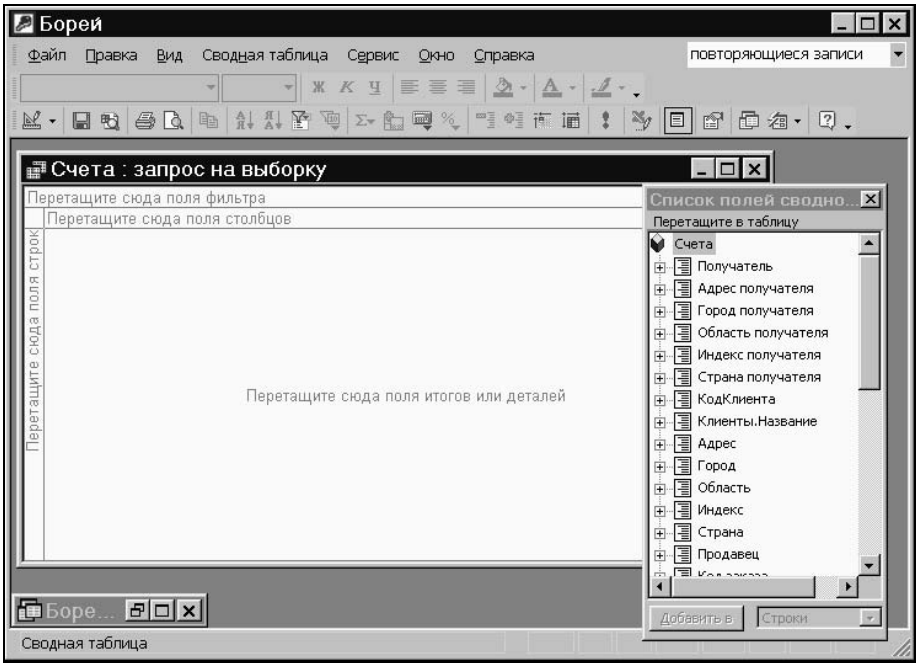


Рис. 8.30. Макет сводной таблицы

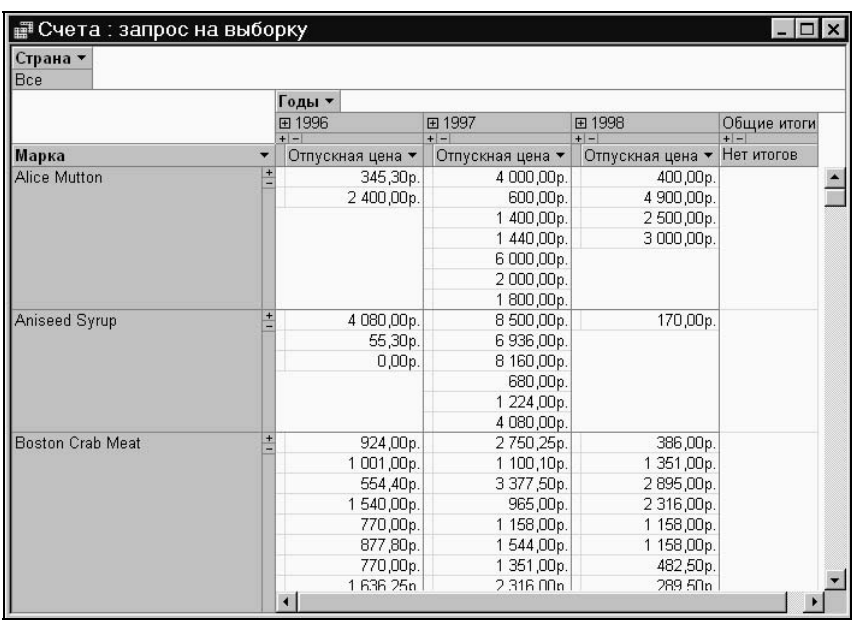


Рис. 8.31. Запрос `Счета` в режиме сводной таблицы

6. Мы определили заголовки строк и столбцов таблицы, теперь остается определить, что будет отображаться в самой таблице. Переместите в самую большую область таблицы поле `Отпускная цена` (ExtendedPrice). Таблица приобретет вид, представленный на рис. 8.31. По умолчанию в ней отображаются все данные по всем товарам и странам.

Когда таблица отображается в режиме Сводная таблица, в верхней части окна приложения видна специальная панель инструментов **Сводные таблицы** (Pivot Tables). Эта панель инструментов представлена на рис. 8.32.



Рис. 8.32. Панель инструментов **Сводные таблицы**

Краткое описание кнопок этой панели приведено в табл. 8.2 (как и в других случаях, в таблице представлены те кнопки, которые используются только в данном режиме). В дальнейших разделах этой главы мы познакомимся с большинством функций, связанных с этими кнопками.

Таблица 8.2. *Панель инструментов* ***Сводные таблицы***

Кнопка	Описание	Команда меню
Автофильтр (AutoFilter)	Устанавливает и сбрасывает фильтры	**Сводная таблица | Автофильтр** (PivotTable | AutoFilter)
Показать верхние и нижние элементы (Show Top/Bottom Items)	Устанавливает фильтр, отображая заданное число первых или последних строк в отсортированной таблице	**Сводная таблица | Показать верхние и нижние элементы** (PivotTable | Show Top/Bottom Items)
Автовычисления (AutoCalc)	Позволяет вычислить промежуточные итоги по каждой группе записей	**Сводная таблица | Автовычисления** (PivotTable | AutoCalc)
Итоги (Subtotal)	Позволяет скрывать или отображать итоговые столбцы или строки	**Сводная таблица | Итоги** (PivotTable | Subtotal)
Вычисляемые итоги и поля (Calculated Total and Fields)	Позволяет задать формулы для расчета как специальных полей, так и итоговых данных	**Сводная таблица | Вычисляемые итоги и поля** (PivotTable | Calculated Total and Fields)
Отобразить как (Show As)	Позволяет отображать итоговые значения в процентах относительно общего итога по столбцу, строке или родительскому элементу на оси столбца или строки	**Сводная таблица | Отобразить как** (PivotTable | Show As)

Таблица 8.2 *(окончание)*

Кнопка	Описание	Команда меню
Свернуть (Collapse)	Уменьшает на один шаг уровень детализации отображаемых данных в выделенной области таблицы	**Сводная таблица** | **Свернуть** (PivotTable | Collapse)
Развернуть (Expand)	Увеличивает на один шаг уровень детализации отображаемых данных в выделенной области таблицы	**Сводная таблица** | **Развернуть** (PivotTable | Expand)
Скрыть подробности (Hide Details)	Позволяет скрыть детальные данные и оставить только подытоги и итоги	**Сводная таблица** | **Скрыть подробности** (PivotTable | Hide Details)
Подробности (Show Details)	Отображает все данные в строках или столбцах таблицы	**Сводная таблица** | **Подробности** (PivotTable | Show Details)
Обновить (Refresh)	Обновляет сводную таблицу, выполняя повторный запрос к базе данных	**Сводная таблица** | **Обновить** (PivotTable | Refresh)
Экспорт в Microsoft Office Excel (Export to Microsoft Office Excel)	Обеспечивает экспорт сводной таблицы в интерактивный сводный отчет Microsoft Excel	**Сводная таблица** | **Экспорт в Microsoft Excel** (PivotTable | Export to Microsoft Excel)
Список полей (Field List)	Отображает диалоговое окно со списком полей таблицы	**Вид** | **Список полей** (View | Field List)

Счета : запрос на выборку

Страна: Все

Годы

Марка	1996 Сумма "Отпускная цена"	1997 Сумма "Отпускная цена"	1998 Сумма "Отпускная цена"	Общие итоги Сумма "Отпускная цена"
Alice Mutton	2 745,30р.	17 240,00р.	10 800,00р.	30 785,30р.
Aniseed Syrup	4 135,30р.	29 580,00р.	170,00р.	33 885,30р.
Boston Crab Meat	8 073,45р.	50 400,52р.	29 157,48р.	87 631,45р.
Camembert Pierrot	1 311,00р.	92 445,00р.	52 611,00р.	146 367,00р.
Carnarvon Tigers	19 049,70р.	63 398,80р.	15 015,00р.	97 463,50р.
Chai	32 053,00р.	80 564,75р.	106 961,92р.	219 579,67р.
Chang	24 738,00р.	94 586,84р.	20 229,04р.	139 553,88р.
Chartreuse verte	27 818,30р.	98 147,30р.	53 176,00р.	179 141,60р.
Chef Anton's Cajun	420,00р.	9 100,00р.	14 448,00р.	23 968,00р.
Chef Anton's Gumb	12 940,00р.	41 575,00р.	32 625,00р.	87 140,00р.
Chocolade	19 200,00р.	31 286,00р.	41 960,00р.	92 446,00р.
Cote de Blaye	35 582,40р.	44 757,00р.	42 606,00р.	122 945,40р.
Escargots de Bourg	90 354,00р.	357 753,00р.	263 450,00р.	711 557,00р.
Filo Mix	51 037,60р.	106 619,30р.	48 084,80р.	205 741,70р.
Flotemysost	7 756,00р.	16 926,00р.	12 362,00р.	37 044,00р.
Geitost	10 024,00р.	21 070,00р.	32 410,00р.	63 504,00р.
Genen Shouyu	16 056,00р.	48 870,00р.	62 955,00р.	127 881,00р.
Gnocchi di nonna A	20 631,00р.	21 567,00р.	34 417,50р.	76 615,50р.
Gorgonzola Telino				

Рис. 8.33. Отображение итоговых данных в режиме сводной таблицы

Определим итоговые столбцы:

1. Щелкните левой кнопкой мыши на заголовке столбца Отпускная цена (ExtendedPrice) — все столбцы (кроме итогового) окажутся выделенными. Теперь щелкните по кнопке **Автовычисления** (AutoCalc) на панели инструментов и выберите из списка имя функции — Sum. Появляется дополнительная строка для каждого товара, содержащая итоговое значение по каждому товару за год, а в итоговом столбце будет отображена итоговая цифра продаж каждого товара по всем годам.
2. Теперь можно скрыть детальные данные, оставив только итоговые. Нажмите кнопку **Скрыть подробности** (Hide Details) на панели инструментов. Окончательный вид таблицы представлен на рис. 8.33.

Фильтрация данных сводной таблицы

Сейчас в таблице отображаются данные о продажах по всем товарам и всем странам. Она напоминает результат перекрестного запроса, но далее вы убедитесь, как легко можно ею манипулировать, отображая разные данные. Например, она может быть очень легко изменена, когда требуется посмотреть выборочно данные по отдельной стране или странам и некоторым товарам, т. е. по тем полям, которые отображаются в области фильтра, области строк и области столбцов.

Справа от названия каждого из этих полей видна небольшая стрелка вниз. Щелкните по этой стрелке в поле Страна (Country) в области фильтра. Появится небольшое окно со списком значений поля, которые присутствуют в исходных данных. По умолчанию у каждой страны установлен флажок (рис. 8.34).

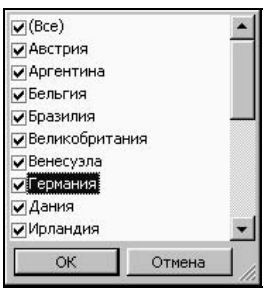


Рис. 8.34. Выпадающий список значений поля

Сбросьте флажок у первой строки — **Все** (All), щелкнув по нему мышью. Все флажки одновременно сбросятся. Затем установите его только у одной страны, например Бельгия, и нажмите кнопку **ОК**. Теперь ваша таблица отображает данные только по Бельгии. Под названием поля Страна (Country) отображается название выбранной страны.

Если нужно отобразить данные сразу для нескольких стран (но не всех), просто установите в списке значений флажки у этих стран. Данные в таблице опять изменятся, только, к сожалению, не видно, какие страны участвуют в выборке — под названием поля в области фильтра указывается просто **несколько элементов** (Multiple Items). Но если вы просто подведете курсор к этой надписи, то всплывает окно со списком выбранных стран.

При желании таблицу можно настроить таким образом, чтобы она не позволяла делать множественный выбор в полях фильтра (слово "полях" мы употребили не случайно, т. к. в этой области может быть несколько полей):

1. Нажмите кнопку **Свойства** (Properties) на панели инструментов. При этом область фильтра с полем `Страна` (Country) должна быть выделенной. Появится диалоговое окно **Свойства** (Properties), которое отображает свойства поля `Страна` (Country). Оно содержит несколько вкладок.
2. Раскройте вкладку **Фильтр и группировка** (Filter and Group) и сбросьте флажок **Разрешить выделять несколько элементов в области фильтра** (Allow selecting multiple items when in filter area) (рис. 8.35).

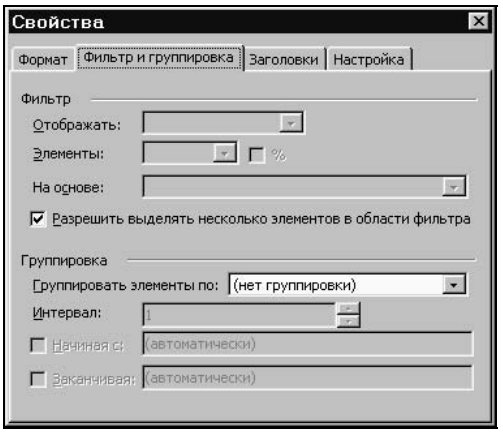


Рис. 8.35. Диалоговое окно **Свойства**

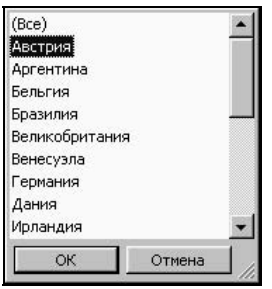


Рис. 8.36. Окно значений поля с отключением множественного выбора

3. Раскройте список значений поля `Страна` (Country). Теперь в списке значений полей отсутствуют флажки, и вы можете выбрать только одну страну (рис. 8.36).

Теперь рассмотрим, как можно фильтровать отображаемые в таблице данные по значениям поля `Марка` (ProductName), которое находится в области строк.

Раскройте список значений поля, щелкнув по стрелке справа от названия поля. Сбросьте флажок **Все** (All) и установите флажки напротив нужных товаров. Нажмите кнопку **OK**. Таблица стала намного короче, а стрелочка в названии столбца стала синей, означая фильтр (рис. 8.37).

Чтобы сбросить установленный фильтр, нажмите кнопку **Автофильтр** (AutoFilter) на панели инструментов. Обратите внимание, что при этом сбрасываются все фильтры, в том числе установленные в области фильтра. Последний установленный фильтр сохраняется, и если вы нажмете эту кнопку еще раз, то снова увидите отфильтрованные данные.

Можно установить фильтр и другими способами. Например, можно отобразить только те товары, которые принесли больше всего или меньше всего дохода. Проще всего для этого воспользоваться специальной кнопкой на панели инструментов.



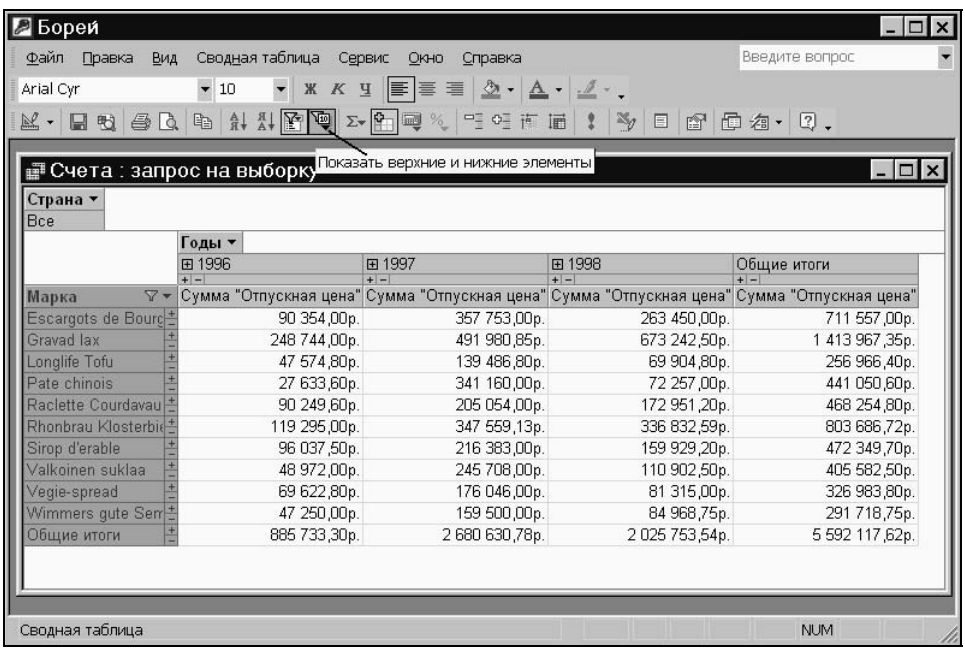
Счета : запрос на выборку

Страна				
Австрия				
	Годы			
	⊞ 1996	⊞ 1997	⊞ 1998	Общие итоги
Марка	Отпускная цена	Отпускная цена	Отпускная цена	Сумма "Отпускная цена"
Alice Mutton		1 800,00р.	2 500,00р.	4 300,00р.
		1 800,00р.	2 500,00р.	
Chai	8 379,00р.		61 180,00р.	69 559,00р.
Общие итоги	8 379,00р.	1 800,00р.	63 680,00р.	73 859,00р.

Рис. 8.37. Сводная таблица с фильтром по товарам

1. Сначала сбросьте все установленные фильтры, т. к. в противном случае новый фильтр будет накладываться на уже отфильтрованные данные.
2. Выделите столбец `Марка` (ProductName), щелкнув по его заголовку.
3. Нажмите кнопку **Показать верхние и нижние элементы** (Show Top/Bottom Items) на панели инструментов и выберите из списка элемент **Показать только верхний элемент** (Show Only the Top) и затем 10, как изображено на рис. 8.38. В таблице будет показано 10 строк, содержащих максимальное значение в поле `Общие итоги` (Grand Total).

Чтобы убрать этот фильтр, достаточно нажать ту же кнопку еще раз и выбрать элемент **Показать все** (Show All).



Счета : запрос на выборку

Страна				
Все				
	Годы			
	⊞ 1996	⊞ 1997	⊞ 1998	Общие итоги
Марка	Сумма "Отпускная цена"	Сумма "Отпускная цена"	Сумма "Отпускная цена"	Сумма "Отпускная цена"
Escargots de Bourg	90 354,00р.	357 753,00р.	263 450,00р.	711 557,00р.
Gravad lax	248 744,00р.	491 980,85р.	673 242,50р.	1 413 967,35р.
Longlife Tofu	47 574,80р.	139 486,80р.	69 904,80р.	256 966,40р.
Pate chinois	27 633,60р.	341 160,00р.	72 257,00р.	441 050,60р.
Raclette Courdavau	90 249,60р.	205 054,00р.	172 951,20р.	468 254,80р.
Rhonbrau Klosterbie	119 295,00р.	347 559,13р.	336 832,59р.	803 686,72р.
Sirop d'erable	96 037,50р.	216 383,00р.	159 929,20р.	472 349,70р.
Valkoinen suklaa	48 972,00р.	245 708,00р.	110 902,50р.	405 582,50р.
Vegie-spread	69 622,80р.	176 046,00р.	81 315,00р.	326 983,80р.
Wimmers gute Sem	47 250,00р.	159 500,00р.	84 968,75р.	291 718,75р.
Общие итоги	885 733,30р.	2 680 630,78р.	2 025 753,54р.	5 592 117,62р.

Рис. 8.38. Товары — лидеры продаж

Кнопка **Показать верхние и нижние элементы** (Show Top/Bottom Items) позволяет фильтровать данные, основываясь на значениях в области детальных и итоговых данных.

Фильтр можно устанавливать более гибко, определяя значения в окне **Свойства** (Properties).

1. Выделите область строк, щелкнув мышью по имени поля `Марка` (ProductName), затем раскройте вкладку **Фильтр и группировка** (Filter and Group) окна **Свойства** (Properties). Верхняя часть этой вкладки содержит поля, относящиеся к фильтрации данных в таблице, а нижняя — к группировке.
2. По умолчанию в таблице отображаются **все значения** (All items). Выберите из списка в поле **Отображать** (Display the) значение **первые** (Top). В поле **Элементы** (Items) можно указать, какое количество значений выводить, а в поле **На основе** (Based on) — на основе какого поля отбирать записи. В списке предлагается на выбор два варианта: либо по значениям в поле `Сумма "Отпускная цена"` (Sum of ExtendedPrice), либо по порядку в исходном запросе. В первом случае мы получим тот же результат, что и в предыдущем примере, а во втором будут выведены строки с первыми по алфавиту названиями товаров, как показано на рис. 8.39.

Счета : запрос на выборку

Страна: Все

	Годы			
	⊞ 1996	⊞ 1997	⊞ 1998	Общие итоги
Марка	Сумма "Отпускная цена"	Сумма "Отпускная цена"	Сумма "Отпускная цена"	Сумма "Отпускная цена"
Alice Mutton	2 745,30р.	17 240,00р.	10 800,00р.	30 785,30р.
Aniseed Syrup	4 135,30р.	29 580,00р.	170,00р.	33 885,30р.
Boston Crab Meat	8 073,45р.	50 400,52р.	29 157,48р.	87 631,45р.
Camembert Pierrot	1 311,00р.	92 445,00р.	52 611,00р.	146 367,00р.
Carnarvon Tigers	19 049,70р.	63 398,80р.	15 015,00р.	97 463,50р.
Chai	32 053,00р.	80 564,75р.	106 961,92р.	219 579,67р.
Chang	24 738,00р.	94 586,84р.	20 229,04р.	139 553,88р.
Chartreuse verte	27 818,30р.	98 147,30р.	53 176,00р.	179 141,60р.
Chef Anton's Cajun	420,00р.	9 100,00р.	14 448,00р.	23 968,00р.
Chef Anton's Gumb	12 940,00р.	41 575,00р.	32 625,00р.	87 140,00р.
Общие итоги	133 284,05р.	577 038,21р.	335 193,44р.	1 045 515,70р.

Рис. 8.39. Фильтр для первых элементов сводной таблицы

Изменение сводной таблицы

Сводные таблицы предназначены только для чтения, данные в них нельзя изменять. Но зато пользователь может очень легко трансформировать сводную таблицу: добавлять новые строки и столбцы, менять их местами, изменять названия полей и уровень детализации отображаемых данных.

В каждой области таблицы может размещаться не одно, а несколько полей данных. Чтобы добавить неиспользованные поля запроса в сводную таблицу, просто перетащите их мышью в нужную область.

Например, можно добавить в область фильтра поле `Клиенты.Название` (CompanyName), что позволит фильтровать данные не только по странам, но и по клиен-

там. Перетащите поле Клиенты.Название (CompanyName) из списка полей в область фильтра и поместите его рядом с полем Страна (Country). Устанавливая флажки против нужных клиентов, вы сможете получать сводные данные по счетам для каждого клиента.

Чтобы удалить поле строки, столбца или фильтра из таблицы, перетащите его мышью за пределы сводной таблицы (при этом рядом с курсором появится изображение креста) и отпустите. Поле исчезнет из таблицы.

Вы можете легко поменять местами поля из области фильтра и из области столбцов или строки поменять местами со столбцами. Переместите поле Клиенты.Название (CompanyName) в область столбцов, а поле Годы (Year) — в область фильтра. Теперь в столбцах таблицы будут отображаться данные по продажам для каждого клиента (рис. 8.40), а, используя поле Дата размещения по месяцам (Order Date By Month), вы сможете фильтровать эти данные.

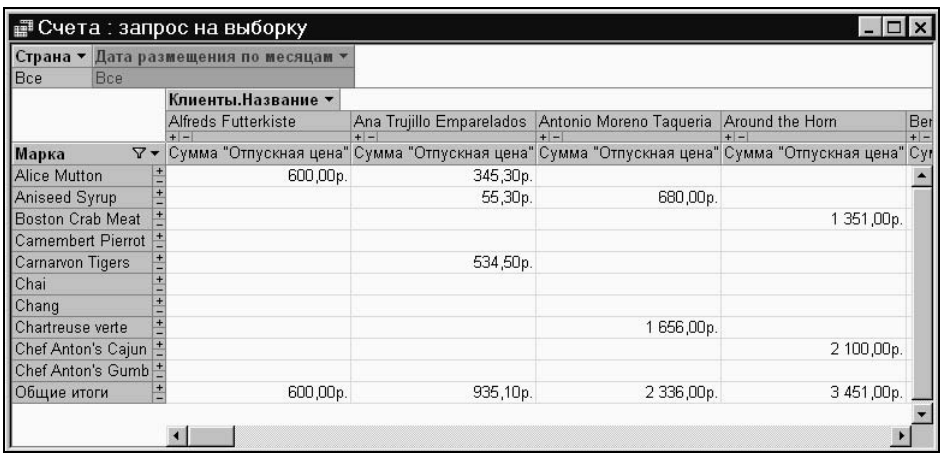


Рис. 8.40. Отображение в сводной таблице данных по клиентам

Совет

При переносе полей обращайте внимание на форму указателя мыши. Когда он попадает в одну из областей таблицы — строк, столбцов, данных или фильтра — его форма меняется, и по ней можно определить, когда следует отпускать кнопку мыши.

Сортировка в сводной таблице

В отличие от аналогичных таблиц в перекрестном запросе, данные в сводной таблице легко сортировать. Для этого используются стандартные кнопки **Сортировать по возрастанию** (Sort Ascending) и **Сортировать по убыванию** (Sort Descending) на панели инструментов.

Нужно выделить поле, по которому требуется отсортировать данные, и нажать соответствующую кнопку на панели инструментов. Если выделить поле в области строк, то данные будут отсортированы в порядке возрастания или убывания значений этого

поля (в нашем примере названия товаров будут отсортированы в алфавитном порядке). Если выделить поле в области столбцов, то столбцы в таблице будут отсортированы в порядке возрастания значений в заголовках столбцов (в данном примере, клиенты будут выведены в алфавитном порядке). Если выделить область детальных и итоговых данных таблицы, то она будет отсортирована в порядке возрастания (или убывания) значений в столбце `Общие итоги` (Grand Total).

Изменение уровня детализации

Вернемся к первоначальному виду таблицы, когда в столбцах отображались данные по годам (см. рис. 8.33). Рассмотрим, как можно изменить уровень детализации данных, выводимых в сводной таблице.

Обратите внимание, что в наименованиях столбцов справа виден маркер развертывания (значок плюса в квадратике). Это означает, что в данный момент отображаются данные самого верхнего уровня детализации.

1. Щелкните по этому маркеру в столбце, соответствующем 1997 году. Появится еще одна строка в заголовке, и вместо одного столбца отобразятся пять: четыре соответствуют кварталам, а последний столбец представляет собой итог за год (рис. 8.41). Поскольку крайний правый столбец таблицы содержит общий итог, по всем годам, то данные в этом столбце будут представлять собой промежуточный итог **Итоги** (Total). Маркер у цифр года изменился: появился значок минуса в квадратике. Его можно использовать потом для того, чтобы снова "свернуть" данные. Справа от поля `Годы` (Year) появилось еще одно поле — `Кварталы` (Quarters).

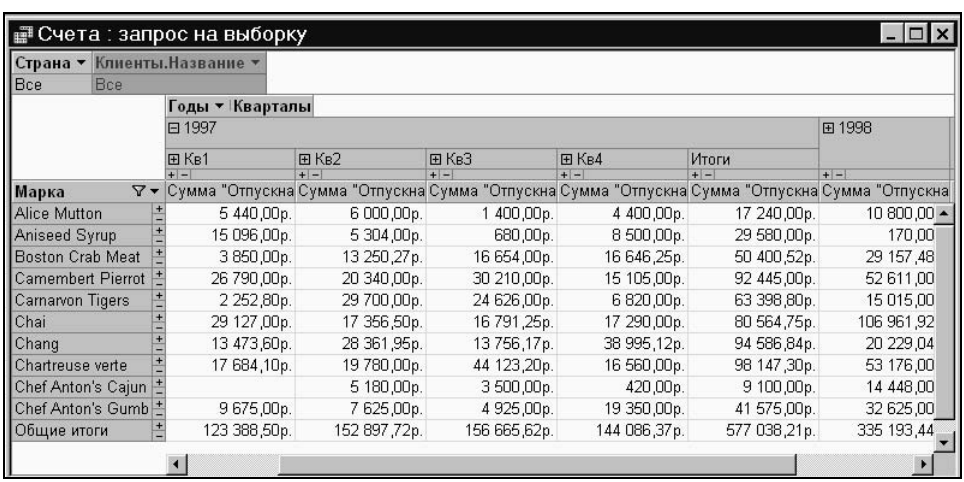


Рис. 8.41. Детализация данных по столбцам сводной таблицы

2. Слева в заголовке кварталов также виден маркер развертывания. Щелкнув по нему, можно дальше развернуть данные и просматривать их по месяцам. Но если вы хотите развернуть данные сразу по всем кварталам, воспользуйтесь специальной кнопкой на панели инструментов. Выделите поле **Кварталы** (Quarters) и на-

жмите кнопку **Развернуть** (Expand) на панели инструментов. Число столбцов в таблице увеличится и для каждого месяца будет отдельный столбец. Кроме того, будут отображаться промежуточные итоги по кварталам и годам.

Замечание

Обратите внимание, что, выполняя все эти операции, мы не создавали никаких вычисляемых полей. Так Access умеет работать с данными типа **Дата/Время** (Date/Time).

3. Если вы хотите теперь снова свернуть таблицу и отобразить только верхний уровень иерархии, выделите поле `Годы` (Year), щелкнув по его названию, и нажмите кнопку **Свернуть** (Collapse) на панели инструментов. Таблица вновь приобретет первоначальный вид.

Для того чтобы отображать или скрывать детальные данные, можно использовать также маленькие кнопочки с изображением знаков плюса и минуса, находящиеся у каждого значения поля, как в строках, так и в столбцах.

1. Подведите указатель мыши к кнопке со знаком плюс в строке с названием первого товара. Появится подсказка **Показать/скрыть детали** (Show/Hide Details). Нажмите эту кнопку. Строка расширится, и в ней будут выведены как детальные данные по всем счетам, в которых участвовал этот товар, так и итоговые данные за год (рис. 8.42). Это как раз тот режим, который не удается получить, если воспользоваться группировкой в обычных запросах на выборку.
2. Чтобы снова скрыть детальные данные, нажмите кнопку со знаком минуса.
3. Чтобы отобразить детальные данные сразу по всем строкам, выделите весь столбец с наименованиями товаров и нажмите кнопку **Подробности** (Show Details) на панели инструментов.
4. Чтобы снова скрыть детальные данные по всем строкам, нажмите кнопку **Скрыть подробности** (Hide Details) на панели инструментов.

Аналогично отображаются и скрываются данные и для столбцов таблицы.

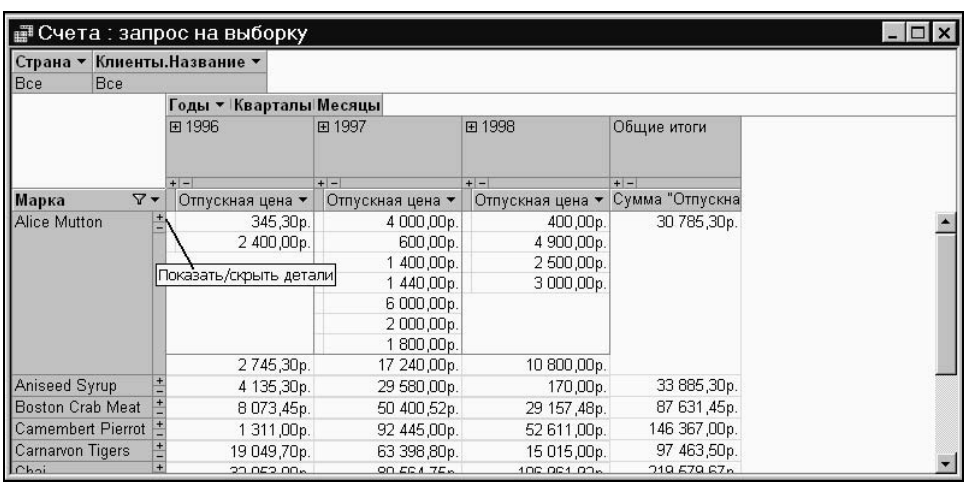
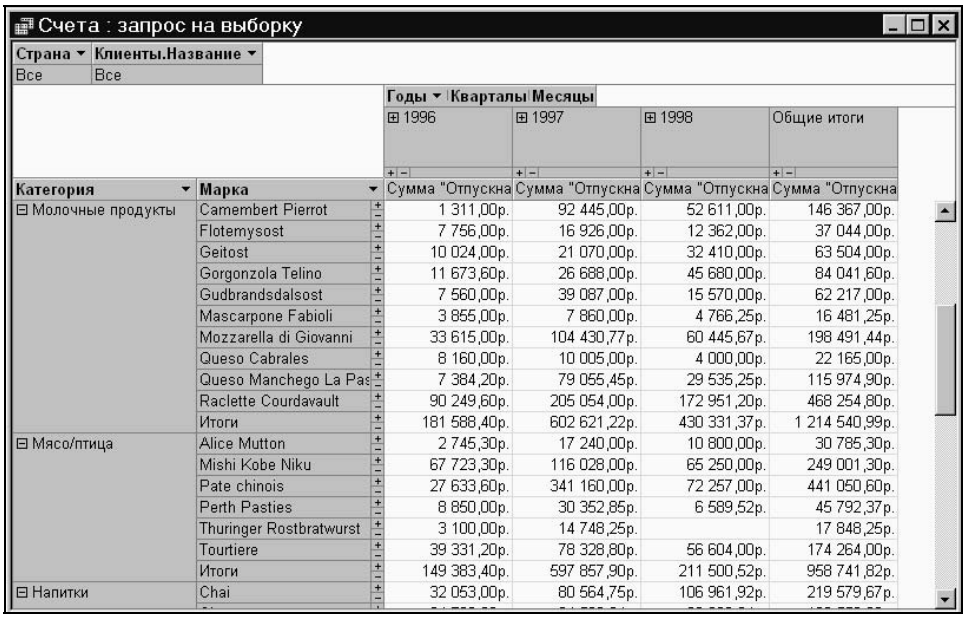


Рис. 8.42. Детализация данных по строкам сводной таблицы

Детализация данных может быть изменена также при добавлении новых полей в область строк или столбцов. Например, можно добавить в область строк поле Категория (Category) и сгруппировать все товары по категориям.

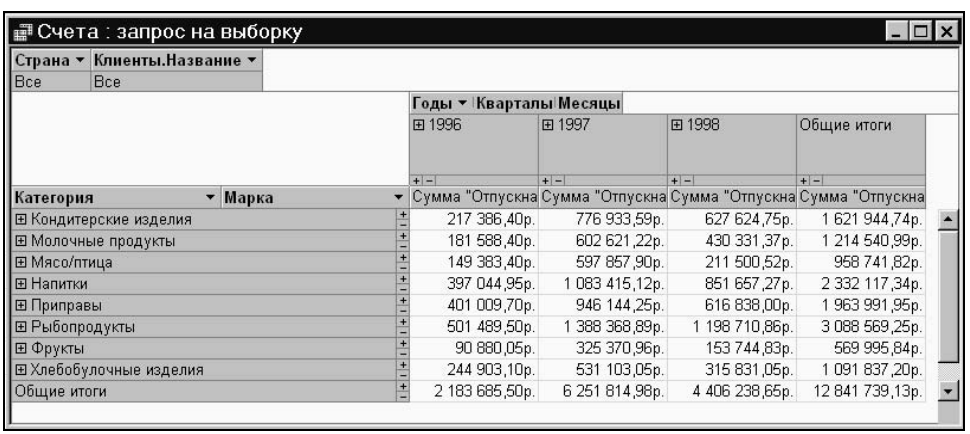
Поле Категория (Category) отсутствует в исходном запросе Счета (Invoices), поэтому сначала нужно перейти в режим Конструктора, добавить к запросу таблицу Типы (Categories), затем добавить в бланк запроса поле Категория (CategoryName) из этой таблицы и снова вернуться в режим сводной таблицы. В списке полей появится новое поле.


Счета : запрос на выборку

Страна	Клиенты.Название
Все	Все

		Годы	Кварталы	Месяцы	
		⊞ 1996	⊞ 1997	⊞ 1998	Общие итоги
Категория	Марка	Сумма "Отпускна	Сумма "Отпускна	Сумма "Отпускна	Сумма "Отпускна
⊟ Молочные продукты	Camembert Pierrot	1 311,00р.	92 445,00р.	52 611,00р.	146 367,00р.
	Flotemysost	7 756,00р.	16 926,00р.	12 362,00р.	37 044,00р.
	Geitost	10 024,00р.	21 070,00р.	32 410,00р.	63 504,00р.
	Gorgonzola Telino	11 673,60р.	26 688,00р.	45 680,00р.	84 041,60р.
	Gudbrandsdalsost	7 560,00р.	39 087,00р.	15 570,00р.	62 217,00р.
	Mascarpone Fabioli	3 855,00р.	7 860,00р.	4 766,25р.	16 481,25р.
	Mozzarella di Giovanni	33 615,00р.	104 430,77р.	60 445,67р.	198 491,44р.
	Queso Cabrales	8 160,00р.	10 005,00р.	4 000,00р.	22 165,00р.
	Queso Manchego La Pas	7 384,20р.	79 055,45р.	29 535,25р.	115 974,90р.
	Raclette Courdavault	90 249,60р.	205 054,00р.	172 951,20р.	468 254,80р.
	Итоги	181 588,40р.	602 621,22р.	430 331,37р.	1 214 540,99р.
⊟ Мясо/птица	Alice Mutton	2 745,30р.	17 240,00р.	10 800,00р.	30 785,30р.
	Mishi Kobe Niku	67 723,30р.	116 028,00р.	65 250,00р.	249 001,30р.
	Pate chinois	27 633,60р.	341 160,00р.	72 257,00р.	441 050,60р.
	Perth Pasties	8 850,00р.	30 352,85р.	6 589,52р.	45 792,37р.
	Thuringer Rostbratwurst	3 100,00р.	14 748,25р.		17 848,25р.
	Tourtiere	39 331,20р.	78 328,80р.	56 604,00р.	174 264,00р.
	Итоги	149 383,40р.	597 857,90р.	211 500,52р.	958 741,82р.
⊟ Напитки	Chai	32 053,00р.	80 564,75р.	106 961,92р.	219 579,67р.

Рис. 8.43. Добавление поля в область строк


Счета : запрос на выборку

Страна	Клиенты.Название
Все	Все

		Годы	Кварталы	Месяцы	
		⊞ 1996	⊞ 1997	⊞ 1998	Общие итоги
Категория	Марка	Сумма "Отпускна	Сумма "Отпускна	Сумма "Отпускна	Сумма "Отпускна
⊞ Кондитерские изделия		217 386,40р.	776 933,59р.	627 624,75р.	1 621 944,74р.
⊞ Молочные продукты		181 588,40р.	602 621,22р.	430 331,37р.	1 214 540,99р.
⊞ Мясо/птица		149 383,40р.	597 857,90р.	211 500,52р.	958 741,82р.
⊞ Напитки		397 044,95р.	1 083 415,12р.	851 657,27р.	2 332 117,34р.
⊞ Приправы		401 009,70р.	946 144,25р.	616 838,00р.	1 963 991,95р.
⊞ Рыбопродукты		501 489,50р.	1 388 368,89р.	1 198 710,86р.	3 088 569,25р.
⊞ Фрукты		90 880,05р.	325 370,96р.	153 744,83р.	569 995,84р.
⊞ Хлебобулочные изделия		244 903,10р.	531 103,05р.	315 831,05р.	1 091 837,20р.
Общие итоги		2 183 685,50р.	6 251 814,98р.	4 406 238,65р.	12 841 739,13р.

Рис. 8.44. Отображение товаров, сгруппированных по категориям

Далее необходимо осуществить следующие действия:

1. Перетащите поле `Категория` (CategoryName) из списка полей в область строк, поместив его перед полем `Марка` (ProductName). Сводная таблица приобретет вид, показанный на рис. 8.43. Теперь в левой части таблицы — два поля, причем все товары автоматически сгруппировались по категориям. Последняя строка в каждой категории представляет собой промежуточный итог продаж по категории.
2. Выделите поле `Категория` (CategoryName) и нажмите кнопку **Свернуть** (Collapse) на панели инструментов. Таблица быстро перестроится и будет иметь вид, показанный на рис. 8.44. Маркеры развертывания справа указывают на то, что представлен верхний уровень детализации, и данные могут быть уточнены.

Общие и промежуточные итоги

Как вы уже убедились, при создании сводной таблицы Access автоматически добавляет общие итоги для строк и столбцов. Кроме того, если в области строки или столбца содержится больше одного поля, то выводятся и промежуточные итоги. Вы научились скрывать детальные данные, отображая в таблице только итоговые. Но можно скрывать при желании и итоговые данные, оставляя в таблице только детальные. Кроме того, для расчета итоговых значений можно использовать несколько функций, а не только суммирование, как это было показано в приведенном ранее примере. И наконец, для расчета итоговых значений можно создавать и добавлять в таблицу вычисляемые поля. Об этих возможностях и будет рассказано в данном разделе.

Чтобы скрыть общий итог по строкам, необходимо выделить крайнее левое поле в области строк (в нашем примере `Категория` (CategoryName)) и нажать кнопку **Итоги** (Subtotal) на панели инструментов. Аналогично, чтобы скрыть общий итог по всем столбцам, нужно выделить самое левое поле в области столбцов и нажать ту же кнопку. На рис. 8.45 показана таблица нашего примера со скрытыми значениями общих итогов.

Чтобы снова отобразить общий итог, опять выделите нужное поле и нажмите эту кнопку второй раз.



Рис. 8.45. Сводная таблица со скрытыми общими итогами

Если вы выделите любое другое поле в области строк или столбцов и нажмете кнопку **Итоги** (Subtotal), будут скрыты промежуточные итоги для этого поля.

Для создания итоговых значений мы использовали команду **Автовычисления** (AutoCalc). Эта команда в зависимости от типа данных в поле, по которому подводятся итоги, сама выбирает подходящую функцию. Например, если поле числового типа, обычно используется функция `Sum()`, если текстового — `Count()`. Вообще говоря, для расчета итоговых значений можно использовать почти те же статистические функции, что и при группировке в запросах на выборку. Список этих функций приведен в табл. 8.1 *(см. разд. "Использование статистических функций для выполнения расчетов" в этой главе)*. В данном случае отсутствуют только функции `First()` и `Last()`.

Например, если сводная таблица содержит данные с оценками по всем предметам учеников в школьном классе, то итоговым значением, скорее всего, будет средний балл, для вычисления которого можно использовать функцию `Avg()`. В одной сводной таблице может быть представлено несколько типов итоговых значений. Например, в нашей таблице можно в качестве итоговых значений отображать не только общие суммы, вырученные за проданные товары, но и количество выписанных счетов. Для того чтобы добавить еще одно итоговое поле, выделите поле `Марка` (ProductName) в области строк и нажмите кнопку **Автовычисления** (AutoCalc). Выберите из списка функцию **Число** (Count). Через несколько секунд в сводной таблице для каждого года будет отображено два столбца:

- **Сумма "Отпускная цена"** (Sum of ExpendingPrice);
- **Количество значений "Марка"** (Count of ProductName).

Но второй столбец с итоговыми значениями может быть отображен и по-другому — в виде строки. Для этого выполните следующее:

1. Выделите столбец `Количество значений "Марка"` (Count of ProductName) (одновременно будут выделены все столбцы данного типа).
2. Откройте диалоговое окно **Свойства** (Properties) и раскройте вкладку **Отчет** (Report).
3. В группе **Отображать итоги в** (Display total as) установите переключатель **заголовках строк** (Row headings). Таблица будет трансформирована и примет вид, представленный на рис. 8.46.

Для отображения итоговых значений могут быть использованы также вычисляемые поля. Для добавления вычисляемого поля нажмите кнопку **Вычисляемые итоги и поля** (Calculated Total and Fields) на панели инструментов и выберите из раскрывшегося меню значение **Создание вычисляемого итога** (Create Calculated Total). В области данных появятся дополнительные строки или столбцы, и будет открыта вкладка **Вычисление** (Calculated) диалогового окна **Свойства** (Properties) (рис. 8.47).

Вы должны ввести имя нового поля в поле **Имя** (Name), а формулу для расчета значения — в большое поле ниже. Если для составления выражения нужно сослаться на какое-то поле, выберите имя этого поля в списке в нижней части вкладки и нажмите кнопку **Добавить ссылку на** (Insert Reference to). Когда выражение будет готово, нажмите кнопку **Изменить** (Change).

Аналогично можно добавить вычисляемое поле и в детальные данные таблицы, хотя в этом случае его можно создать и в исходном запросе.

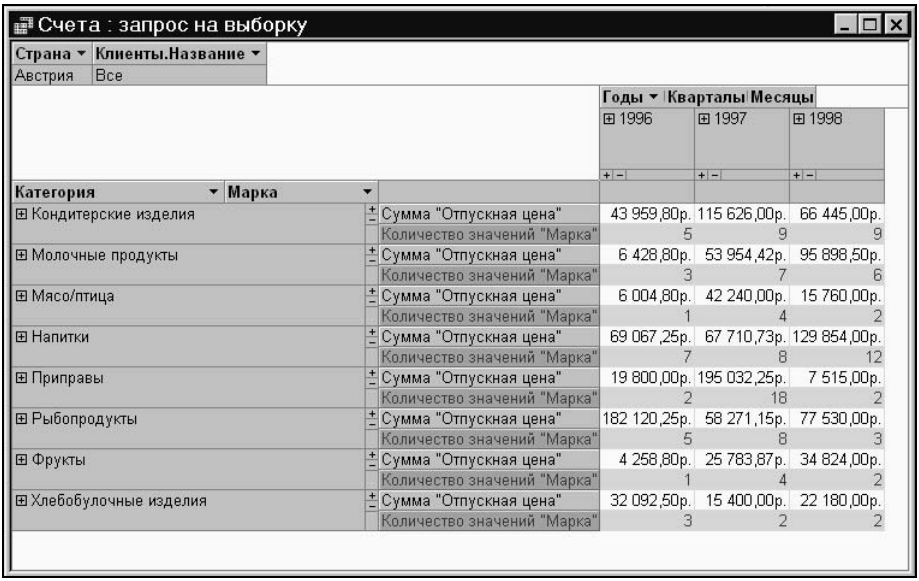


Рис. 8.46. Сводная таблица с несколькими итоговыми полями

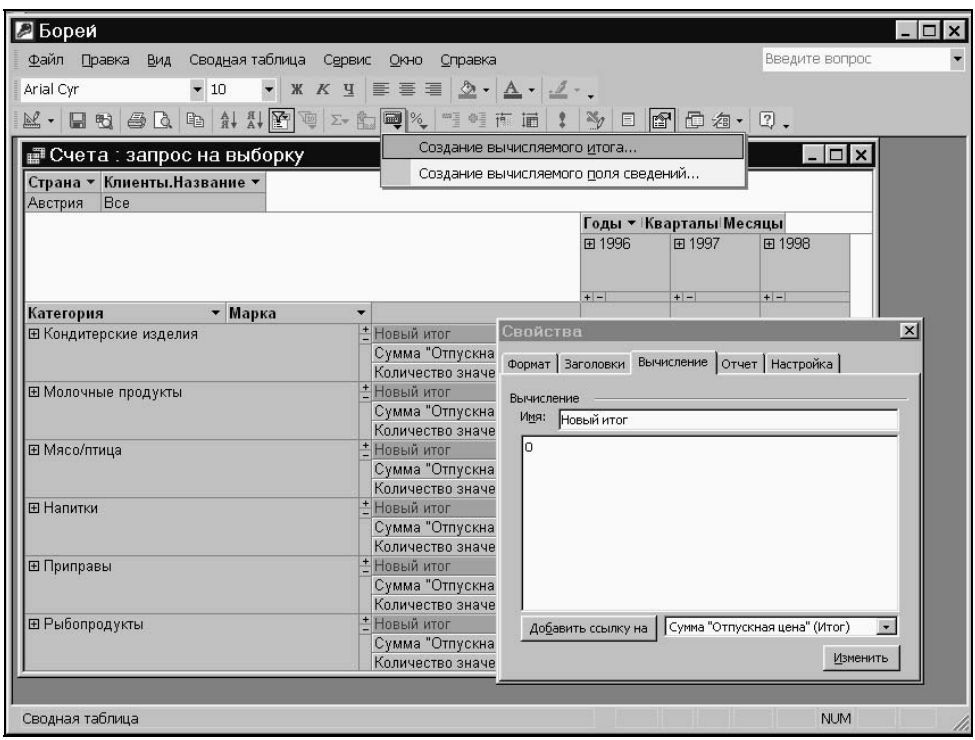


Рис. 8.47. Создание вычисляемого поля

Форматирование сводной таблицы

Для форматирования полей сводной таблицы используется диалоговое окно **Свойства** (Properties). Можно задать формат для полей области фильтра, строк, столбцов и детальных данных. Для этого сначала нужно выделить поле, а затем раскрыть вкладку **Формат** (Format) диалогового окна свойств. Эта вкладка показана на рис. 8.48.

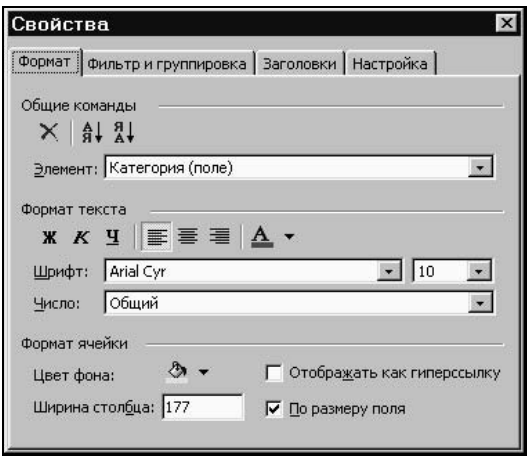


Рис. 8.48. Диалоговое окно **Свойства** сводной таблицы, вкладка **Формат**

Из рис. 8.48 видно, что для поля можно определить:

- тип, размер и цвет используемого шрифта;
- способ выравнивания данных в поле;
- цвет фона;
- ширину столбца.

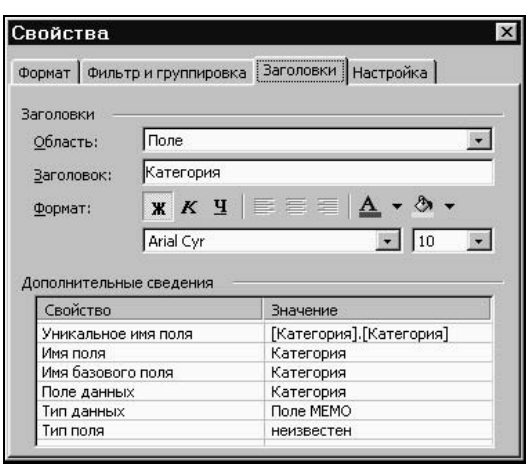


Рис. 8.49. Диалоговое окно **Свойства** сводной таблицы, вкладка **Заголовки**

Если в области данных отображаются числовые поля, даты или время, то можно задать формат отображения данных в этих полях — либо выбрать из списка стандартный формат, либо задать нужный формат с помощью маски. Используйте для этого поле **Число** (Number).

Кроме того, можно задать формат заголовков полей сводной таблицы, для чего предназначена вкладка **Заголовки** (Caption) (рис. 8.49).

В поле **Заголовок** (Caption) можно задать текст, который будет отображен в названии поля, и затем формат этого текста.

Сводные диаграммы

Для того чтобы отображение данных было еще более наглядным, можно использовать сводные диаграммы. Сводная диаграмма строится автоматически на основе уже созданной сводной таблицы, но может также строиться и самостоятельно, на основе исходной таблицы или запроса. Когда диаграмма построена, ее можно перестраивать, перетаскивая поля, аналогично тому, как это делается в сводных таблицах.

Создание сводной диаграммы

Создать диаграмму на основе сводной таблицы очень просто. Переключитесь в режим **Сводная диаграмма** (PivotChart View), и на экране появится сводная диаграмма, примерно такая, как на рис. 8.50.

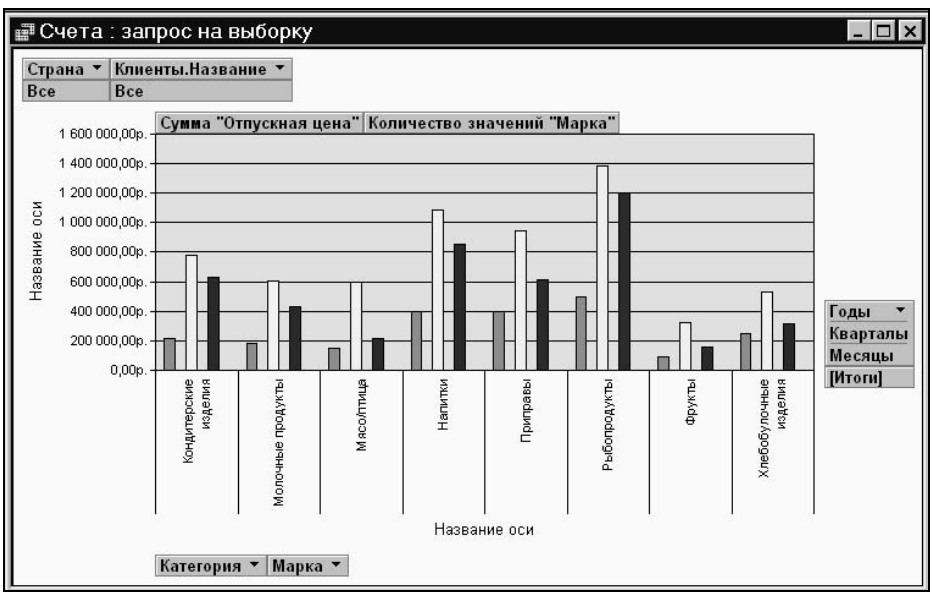


Рис. 8.50. Сводная диаграмма, построенная на основе сводной таблицы

Как и сводная таблица, она имеет поле фильтра — `Страна` (Country), которое отображается в левом верхнем углу экрана, поля строк и столбцов, которые здесь ото-

бражаются справа и снизу. Эта сводная диаграмма тесно связана с таблицей. Если вы переключитесь в режим сводной таблицы и измените ее структуру, это изменение будет отображено и на сводной диаграмме, и наоборот, если сейчас изменить структуру сводной диаграммы, то это изменение появится и на сводной таблице, когда вы вновь переключитесь в тот режим.

Но можно создать диаграмму и непосредственно на базе запроса или таблицы. Причем одновременно будет создаваться и сводная таблица. Сводная таблица и сводная диаграмма — это две формы представления одних и тех же данных.

В качестве примера предлагается построить сводную диаграмму для запроса `Продажи по сотрудникам и странам` (Employee Sales by Country).

1. Откройте этот запрос в режиме Конструктора.
2. Запрос имеет два параметра: [**Начальная дата**] и [**Конечная дата**], которые используются для фильтрации данных. Для сводной диаграммы эти параметры не нужны, поэтому сначала удалите выражение из строки **Условие отбора** (Criteria), затем откройте диалоговое окно **Параметры** (Query Parameters) *(см. разд. "Запросы с параметрами" гл. 4)* и удалите оба параметра.

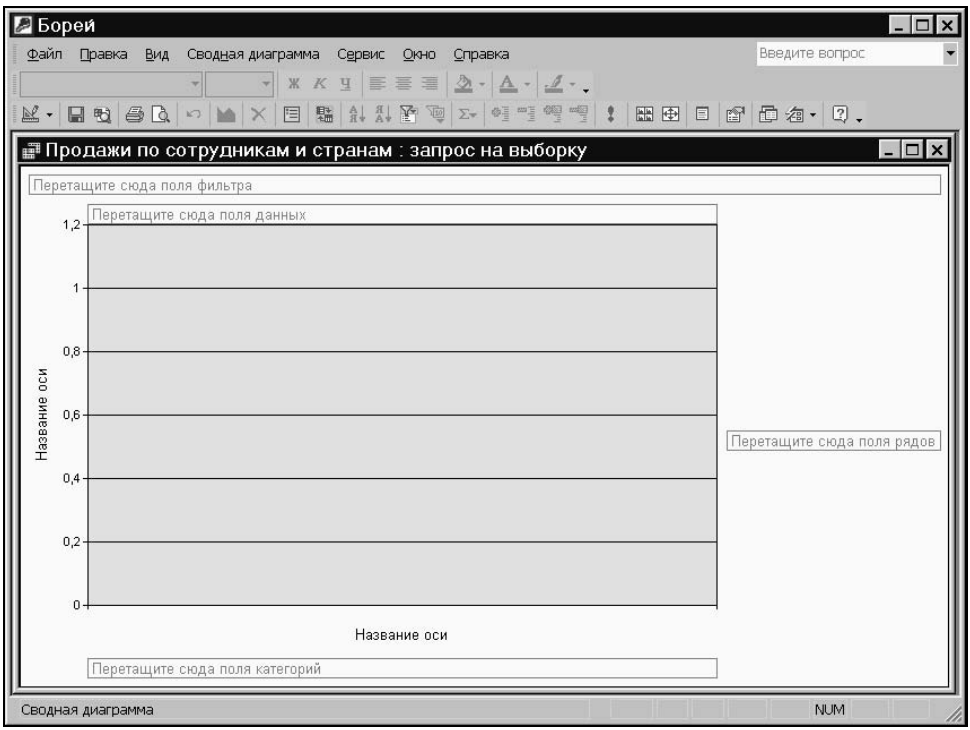


Рис. 8.51. Макет сводной диаграммы

3. Щелкните по стрелке на кнопке **Вид** (View) панели инструментов и выберите из меню пункт **Сводная диаграмма** (PivotChart View). Появится окно, основную часть которого занимает область отображения диаграммы (рис. 8.51), ограничен-

ная осями координат и размеченная линиями сетки. Кроме этого, видны область фильтра, которая играет ту же роль и расположена так же, как и в сводной таблице, область категорий и область рядов, которые соответствуют строкам и столбцам сводной таблицы. В область категорий переносятся поля, значения которых должны откладываться по оси X (горизонтальной), а в область рядов — поля, каждое значение которых соответствует одной серии точек или столбцов на диаграмме (в зависимости от типа диаграммы). Эти поля соответствуют полям столбцов на сводной диаграмме. В область данных помещаются поля, значения которых будут отображаться по оси Y (вертикальной) диаграммы.

4. Перетащите из списка полей в область фильтра поле `Страна` (Country), в область категорий — поля `Фамилия` (Last Name) и `Имя` (First Name), в область рядов — поле `Дата исполнения по месяцам` (Shipped Date By Month). Следите, как будет меняться область диаграммы.

Замечание

Если вы не видите диалоговое окно со списком полей, щелкните по кнопке **Список полей** (Fields List) панели инструментов.

5. Перенесите поле `СуммаПродаж` (Sale Amount) в область данных — диаграмма готова. Нажмите кнопку **Добавить легенду** (Show Legend), чтобы отобразить легенду, после чего вы получите диаграмму, представленную на рис. 8.52.

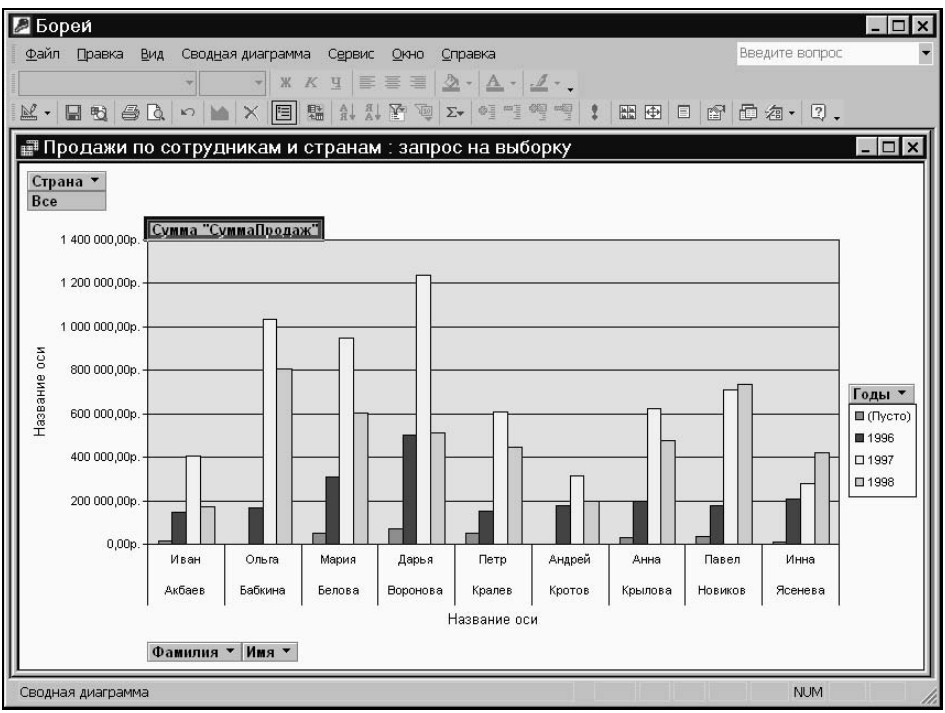


Рис. 8.52. Сводная диаграмма `Продажи по сотрудникам и странам`

6. Можно еще ввести надписи у осей диаграммы. Щелкните по надписи **Название оси** (Axis Title) под осью X. Выведите на экран окно **Свойства** (Properties) и раскройте вкладку **Формат** (Format). Введите в поле **Заголовок** (Caption) `Сотрудники`. Аналогично введите надпись `Объем продаж для оси Y`.

Примечание

Как уже говорилось ранее, одновременно со сводной диаграммой создается и сводная таблица.

7. Щелкните по стрелке на кнопке **Вид** (View) и переключитесь в режим сводной таблицы. Вы увидите сводную таблицу, показанную на рис. 8.53.

Продажи по сотрудникам и странам : запрос на выборку

Страна: Все

Годы

Фамилия	Имя	родаж"	⊞ 1996 Сумма "СуммаПродаж"	⊞ 1997 Сумма "СуммаПродаж"	⊞ 1998 Сумма "СуммаПродаж"	Общие итоги Сумма "СуммаПродаж"
⊟ Акбаев	Иван	55,00р.	145 196,85р.	408 263,72р.	171 815,72р.	739 131,29р.
	Итоги	55,00р.	145 196,85р.	408 263,72р.	171 815,72р.	739 131,29р.
⊟ Бабкина	Ольга		166 743,60р.	1 037 190,91р.	808 697,91р.	2 012 632,42р.
	Итоги		166 743,60р.	1 037 190,91р.	808 697,91р.	2 012 632,42р.
⊟ Белова	Мария	02,21р.	308 617,65р.	950 370,27р.	605 652,26р.	1 912 942,39р.
	Итоги	02,21р.	308 617,65р.	950 370,27р.	605 652,26р.	1 912 942,39р.
⊟ Воронова	Дарья	71,50р.	499 843,40р.	1 237 775,67р.	511 630,13р.	2 320 520,70р.
	Итоги	71,50р.	499 843,40р.	1 237 775,67р.	511 630,13р.	2 320 520,70р.
⊟ Кралев	Петр	39,85р.	153 256,70р.	606 405,59р.	445 598,95р.	1 254 751,09р.
	Итоги	39,85р.	153 256,70р.	606 405,59р.	445 598,95р.	1 254 751,09р.
⊟ Кротов	Андрей		179 461,70р.	314 331,87р.	196 918,94р.	690 712,51р.
	Итоги		179 461,70р.	314 331,87р.	196 918,94р.	690 712,51р.
⊟ Крылова	Анна	96,00р.	198 113,00р.	624 188,34р.	477 279,43р.	1 329 776,77р.
	Итоги	96,00р.	198 113,00р.	624 188,34р.	477 279,43р.	1 329 776,77р.
⊟ Новиков	Павел	79,75р.	178 114,60р.	711 681,40р.	737 901,80р.	1 665 377,55р.
	Итоги	79,75р.	178 114,60р.	711 681,40р.	737 901,80р.	1 665 377,55р.
⊟ Ясенева	Инна	80,00р.	206 663,90р.	279 223,92р.	421 426,59р.	915 894,41р.
	Итоги	80,00р.	206 663,90р.	279 223,92р.	421 426,59р.	915 894,41р.
Общие итоги		74,31р.	2 036 011,40р.	6 169 431,69р.	4 376 921,73р.	12 841 739,13р.

Рис. 8.53. Сводная таблица `Продажи по сотрудникам и странам`

Изменение вида диаграммы

Так же, как и сводная таблица, сводная диаграмма является очень гибкой, и ее можно легко трансформировать, добавляя и удаляя поля, перемещая их из одной области в другую. Кроме того, можно изменить тип диаграммы, например, превратить ее из гистограммы в график (рис. 8.54). В этом случае каждая ломаная линия соответствует серии столбцов на гистограмме. Каждая линия выделяется своим цветом и видом точек. Легенда описывает, какому значению поля `Годы` (Year) соответствует каждая линия.

Для того чтобы изменить тип диаграммы, щелкните мышью в любом месте области диаграммы и убедитесь, что эта область выделена. Тогда становится доступна кнопка **Тип диаграммы** (Chart Type) на панели инструментов (рис. 8.55). Нажмите эту



Рис. 8.54. Диаграмма в виде графика

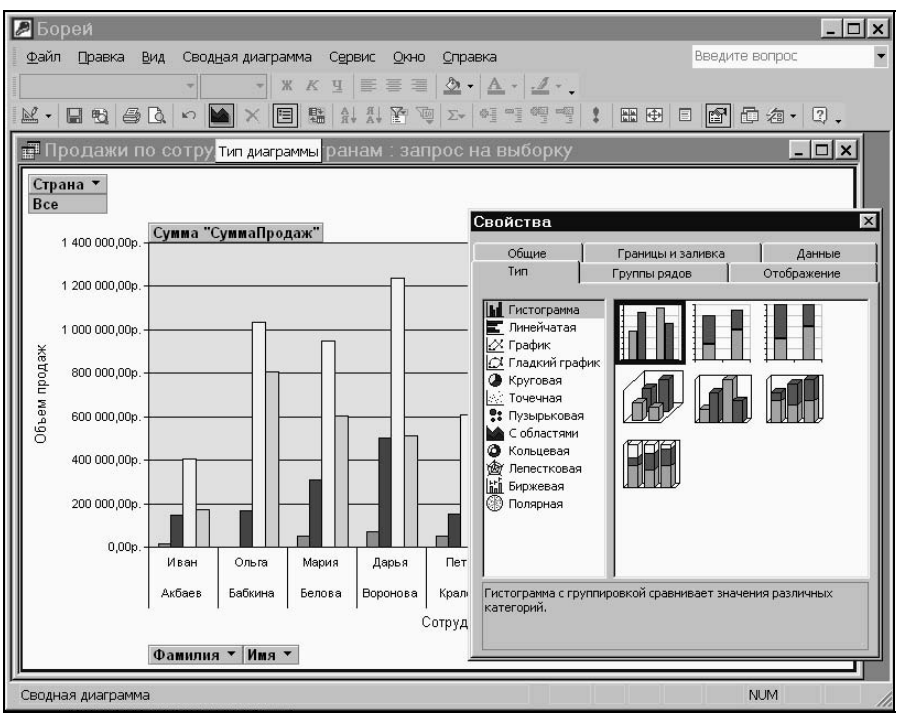


Рис. 8.55. Изменение типа сводной диаграммы

кнопку. В диалоговом окне **Свойства** (Properties) раскроется вкладка **Тип** (Type). На ней представлены все типы диаграмм, которые могут быть построены. Выберите нужный тип из списка, а затем подтип справа.

Можно очень легко поменять местами поля рядов и категорий. Для этого достаточно нажать кнопку **По строке или по столбцу** (By Row/By Column) на панели инструментов. И диаграмма мгновенно трансформируется (рис. 8.56). Остальные кнопки панели инструментов, такие как **Свернуть** (Collapse), **Развернуть** (Expand), **Автовычисления** (AutoCalc), работают так же, как и в сводной таблице.

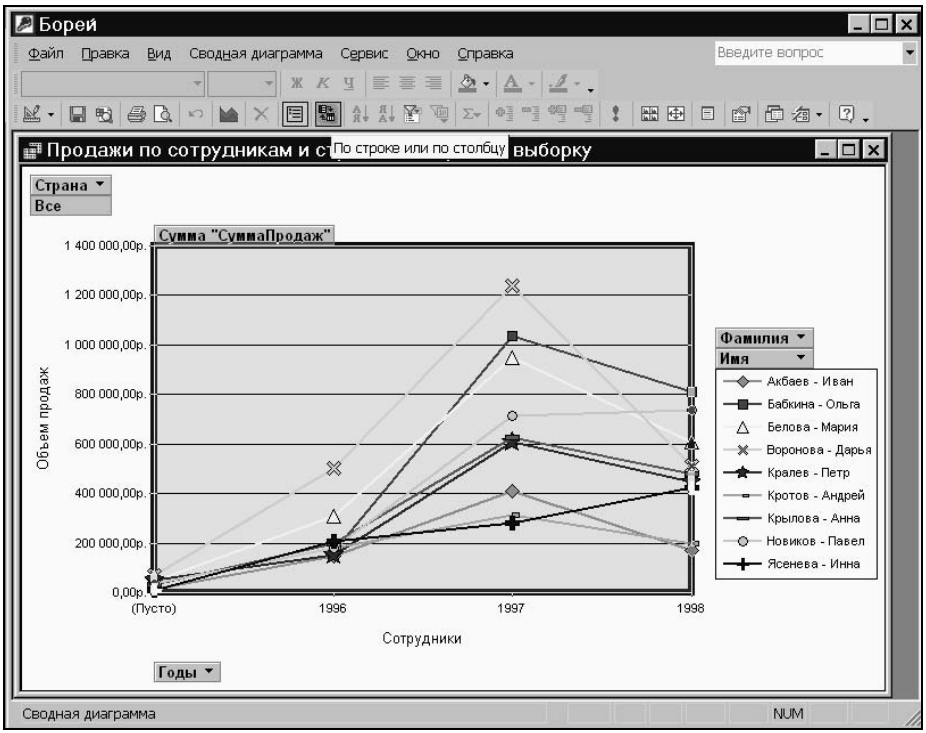


Рис. 8.56. Трансформация сводной диаграммы

Можно поворачивать диаграмму по часовой или против часовой стрелки. Для этого, выделив область диаграммы, необходимо открыть вкладку **Общие** (General) диалогового окна **Свойства** (Properties) и использовать четыре кнопки: **Отразить справа налево** (Flip Horizontal), **Отразить сверху вниз** (Flip Vertical), **Повернуть на 90° по часовой стрелке** (Rotate Clockwise) и **Повернуть на 90° против часовой стрелки** (Rotate CounterClockwise).

Форматирование элементов сводной диаграммы

Форматирование элементов сводной диаграммы выполняется с помощью различных вкладок диалогового окна **Свойства** (Properties).

- Для любых надписей на диаграмме можно определить размер, цвет и тип шрифта, место размещения.
- Для поля диаграммы можно определить цвет и текстуру заполнения.
- Для линии сетки и границ можно указать цвет, тип, а можно их просто убрать.
- Для столбиков или линий диаграммы можно определить цвет и тип.

Мы не будем здесь описывать подробно все возможности форматирования диаграммы (их очень много), поскольку вы сами сможете легко понять, как изменяются параметры в окне **Свойства** (Properties), и получать диаграммы, удовлетворяющие самого требовательного начальника.

Настройка свойств полей в запросе и свойств самого запроса

По умолчанию поля в запросе наследуют все свойства, которые определены для поля соответствующей таблицы. Однако можно изменить значение этих свойств, и данные изменения будут влиять только на поля в запросе и не будут влиять на поля в исходной таблице.

К вычисляемому полю в запросе применимы только свойства, заданные в запросе, т. к. данные, отображаемые в вычисляемом поле, не хранятся в исходной таблице. Например, если значение в поле вычисляется путем перемножения чисел, которые не являются целыми, следует установить свойство **Формат** (Format) для этого поля. Можно также применить функцию `Format (<имя поля>, <формат>)` или подходящую функцию преобразования типов данных, чтобы получить отображение данных в столбце в требуемом формате.

Существует несколько способов вывода на экран диалогового окна, содержащего свойства запроса или свойства поля в запросе. Сначала нужно щелкнуть левой кнопкой мыши в нужном столбце или на свободном поле в верхней панели запроса (для вывода свойств самого запроса), затем нажать кнопку **Свойства** (Properties) на панели инструментов или выполнить команду **Вид | Свойства** (View | Properties). Появится диалоговое окно, содержащее соответствующий список свойств. Кроме того, можно щелкнуть правой кнопкой мыши в столбце бланка запроса и выбрать в контекстном меню команду **Свойства** (Properties). Самый быстрый способ вывода окна свойств запроса — это дважды щелкнуть левой кнопкой мыши по свободной области верхней части окна запроса.

В окне свойств поля можно задать значения следующих свойств:

- **Описание** (Description) — позволяет задать текст, который выводится в строке состояния, когда курсор помещается в это поле в результирующем множестве запроса;
- **Формат поля** (Format) — позволяет задать формат отображения данных в результирующем множестве запроса;
- **Маска ввода** (Input Mask) — позволяет указать формат для ввода данных, если результирующее множество запроса является обновляемым. Для создания маски ввода, соответствующей типу данных поля, можно нажать кнопку Построителя справа от поля свойства, чтобы запустить Мастер по созданию масок ввода;

- **Подпись** (Caption) — позволяет изменить заголовок столбца результирующего множества запроса. Это свойство мы уже рассматривали в *разд. "Создание и изменение запроса с помощью Конструктора" гл. 4*;
- **Смарт-теги** (Smart Tags) — позволяет определить смарт-теги для данного поля.

Каждое из перечисленных свойств соответствует соглашениям о свойствах полей таблиц. Присваивание значения свойству **Подпись** (Caption) эквивалентно указанию префикса в строке **Поле** (Field) в бланке запроса для этого поля. Значение свойства **Маска ввода** (Input Mask) не обязательно соответствует значению свойства **Формат поля** (Format). Например, поле с типом данных **Дата/Время** (Date/Time) может иметь средний формат даты для вывода, а в маске ввода можно задать короткий формат.

С любым полем запроса можно связать смарт-тег: либо стандартный, либо специально созданный. Для этого нужно нажать кнопку Построителя справа от поля свойства. Появится диалоговое окно **Смарт-теги** (Smart Tags), в котором отображается список доступных смарт-тегов. По умолчанию включаются два смарт-тега: **Дата** (Date) и **Имя** (Person Name) (рис. 8.57).

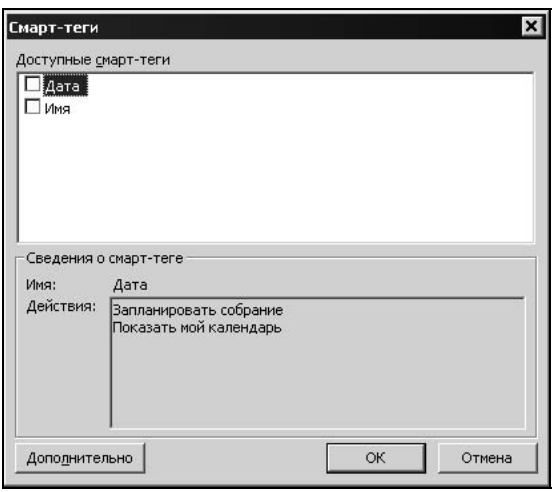


Рис. 8.57. Диалоговое окно **Смарт-теги**

В нижней части окна показываются действия, которые связаны с выбранным смарт-тегом. Нажав кнопку **Дополнительно** (More Smart Tags), можно скачать дополнительные смарт-теги с Web-сайта Microsoft.

В окне свойств запроса (рис. 8.58) можно задать значения следующих свойств:

- **Описание** (Description) — используется, чтобы задать текст, который будет выводиться в строке состояния при отображении результирующего множества запроса.
- **Режим по умолчанию** (Default View) — позволяет указать тот вид, в котором будут представлены результаты запроса при его открытии: таблицы, сводная таблица,

сводная диаграмма. То есть теперь вы можете открыть запрос прямо в режиме сводной таблицы или диаграммы.

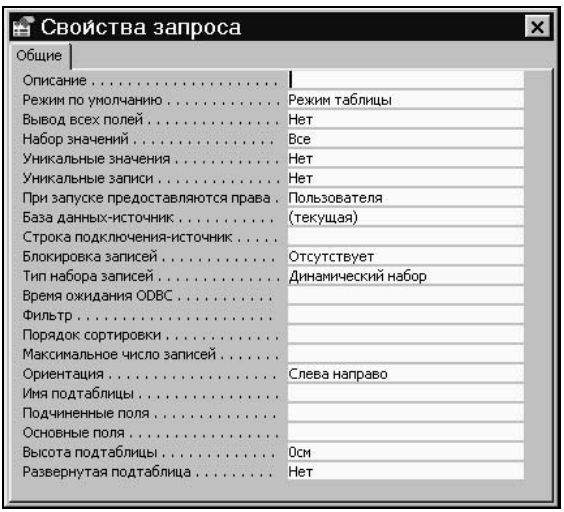


Рис. 8.58. Диалоговое окно **Свойства запроса**

- **Вывод всех полей** (Output All Fields) — позволяет включить в результирующий набор все поля из всех базовых таблиц запроса. Следует отметить, что при выводе большого количества полей выполнение запроса происходит медленнее, особенно если в результирующем множестве содержится большое количество записей.
- **Набор значений** (Top Values) — позволяет указать, какое количество строк из результирующего набора нужно отобразить на экране. По умолчанию это свойство имеет значение **Все** (All), однако, если запрос содержит большое количество записей, чтобы ускорить вывод, можно указать число, которое задает либо количество записей, либо процент записей. В этом случае Access будет выводить информацию на экран сразу, как только найдет заданное количество строк.
- **Уникальные значения** (Unicue Values) и **Уникальные записи** (Unique records) — позволяют не включать в результирующий набор повторяющиеся записи. Эти свойства мы рассмотрим подробнее в *разд. "Зарезервированные слова DISTINCTROW и DISTINCT" данной главы (см. также разд. "Изменение данных в результирующем множестве запроса" гл. 4).*
- **При запуске предоставляются права** (Run Permissions) — позволяет ограничить доступ к базовым таблицам других пользователей. Это свойство используется в том случае, если база данных работает в многопользовательском режиме и защищена на уровне пользователей *(см. разд. "Администрирование баз данных Access, защищенных на уровне пользователей" гл. 20)*. Оно позволяет дать пользователю дополнительные права по сравнению с теми, что назначены ему при защите базы данных. Для этого текущему свойству должно быть присвоено значение **Владельца** (Owner's). По умолчанию оно имеет значение **Пользователя** (User's), при этом пользователи не имеют никаких дополнительных прав.

- **База данных-источник** (Source Database) и **Строка подключения-источник** (Source Connect Str) — используются для доступа к внешней базе данных, содержат имя и тип внешней базы данных.
- **Блокировка записей** (Record Locks) — также используется в многопользовательском режиме и позволяет установить разные типы блокировки. По умолчанию имеет значение **Отсутствует** (No Locks). Это означает, что записи не блокируются в процессе их редактирования пользователем. Если данное свойство имеет значение **Изменяемой записи** (Edited Records), то когда пользователь редактирует запись, она блокируется и не может быть изменена другим пользователем. Значение **Всех** записей (All Records) позволяет блокировать все записи, отобранные запросом, до тех пор, пока пользователь не закроет запрос. *(Подробно о многопользовательском режиме работы и типах блокировок в Access см. разд. "Организация совместного доступа к данным и объектам" гл. 16.)*
- **Тип набора записей** (RecordsetType) — позволяет определить тип результирующего набора записей. Это свойство может иметь три значения:
  - **Динамический набор** (Dynaset) — если запрос базируется на одной таблице или таблицах, связанных отношением "один-к-одному", то все поля запроса могут редактироваться, а если таблицы в запросе связаны отношением "один-ко-многим", то редактироваться данные могут только в таблице на стороне "один";
  - **Динамический набор (несогл.)** (Dynaset (Inconsistent updates)) — все поля запроса могут редактироваться;
  - **Статический набор** (Snapshot) — ни одно поле запроса не может редактироваться.
- **Время ожидания ODBC** (ODBC Timeout) — указывает число секунд, в течение которых Access будет повторять попытки выполнения запроса к базе данных, подключенной к текущей базе посредством драйверов ODBC. Если по истечении заданного времени ответа от сервера базы данных не поступает, выдается сообщение об ошибке.
- **Фильтр** (Filter) — позволяет задать фильтр, который можно применить к результирующему набору запроса. Значением свойства является строковое выражение, которое создается в соответствии с правилами составления предложения `WHERE`. Этот фильтр сохраняется вместе с запросом.
- **Порядок сортировки** (OrderBy) — позволяет задать порядок сортировки записей в результирующем наборе. Значением этого свойства является строковое выражение, содержащее имена полей, перечисленные через запятую.
- **Максимальное число записей** (Max Records) — определяет максимальное количество записей, которое возвращается запросом из внешней базы данных.
- **Ориентация** (Orientation) — позволяет выводить поля запроса как в обычном, так и в обратном порядке. По умолчанию задается тот порядок следования полей, который определен в запросе: **Слева направо** (Left-to-Right).

Последние пять свойств позволяют задать свойства подчиненной таблицы, если в запросе отображаются записи как главной, так и подчиненной таблиц: имя таблицы, связывающие поля в главной и подчиненной таблицах, высота таблицы, и будет ли она по умолчанию отображаться или же будет скрыта.

Создание запросов SQL

Все запросы, которые мы рассматривали до сих пор, создавались либо с помощью Мастера, либо с помощью Конструктора запросов. Конструктор запросов представляет собой графический инструмент для создания *запросов по образцу* (Query By Example, QBE). Однако на самом деле любой запрос хранится в базе данных в формате *языка структурированных запросов* (Structured Query Language, SQL). Основное достоинство этого языка состоит в том, что он является стандартом для большинства реляционных СУБД. SQL имеет унифицированный набор инструкций, которые можно использовать во всех СУБД, поддерживающих этот язык. Действующим на данный момент стандартом языка SQL является принятая Американским национальным институтом стандартов (American National Standards Institute, ANSI) версия SQL-92. Фирмы-разработчики СУБД при реализации языка SQL могут вносить в него расширения, но обязаны реализовать базовый набор команд ASNSI SQL.

Процессор обработки данных Jet, который является составной частью Access, выполняет инструкции Access SQL (Jet SQL), который отличается от ANSI SQL существенно (как правило, настольные СУБД, совместимые со стандартом SQL, реализуют не все инструкции ANSI SQL).

Замечание

В дальнейшем для обозначения используемого в Access диалекта языка SQL мы будем применять термин Jet SQL. Это будет правильнее, т. к. процессор обработки данных используется не только в приложениях, созданных в среде Access, но и в приложениях, разработанных с помощью Microsoft Visual Basic.

В данном разделе мы собираемся дать обзор используемого в Access языка SQL и показать, как можно создать запросы, которые невозможно создать с помощью Конструктора запросов.

Сравнение ANSI и Jet SQL

Язык Jet SQL почти соответствует стандарту ANSI SQL-89. В реализацию языка SQL для Microsoft Jet 4.*x* (используемого, начиная с версии Microsoft Access 2000) внесены несколько расширений, которые приближают его к стандарту ANSI SQL-92 и Transact-SQL — диалекту языка SQL для Microsoft SQL Server. Для того чтобы обеспечить совместимость с предыдущими версиями Microsoft Jet, эти расширения можно использовать только в специальном режиме — ANSI SQL-92.

Замечание

Режим ANSI SQL-92 доступен только при использовании программы Microsoft OLE DB Provider для Jet.

Основные различия языков Jet SQL и ANSI SQL состоят в следующем:

- они имеют разные наборы зарезервированных слов и типов данных;
- разные правила применимы к оператору `Between...And`, используемому для определения условий выборки записей;
- подстановочные знаки ANSI и Microsoft Jet, которые используются в операторе `Like`, взаимно исключают друг друга;

- язык Jet SQL обычно предоставляет пользователю большую свободу, например разрешается группировка и сортировка по выражениям;
- язык Jet SQL позволяет использовать более сложные выражения.

По умолчанию в базах данных Access используется язык Jet SQL, причем как в формате Access 2000, так и в формате Access 2002-2003. Для использования ANSI SQL нужно установить соответствующий параметр в окне **Параметры** (Options).

1. Откройте окно **Параметры** (Options).
2. Раскройте вкладку **Таблицы и запросы** (Tables/Queries).
3. Установите один из флажков в группе **Синтаксис для SQL Server (ANSI 92)** (SQL Server Compatible Syntax) и нажмите кнопку **Применить** (Apply) (рис. 8.59).

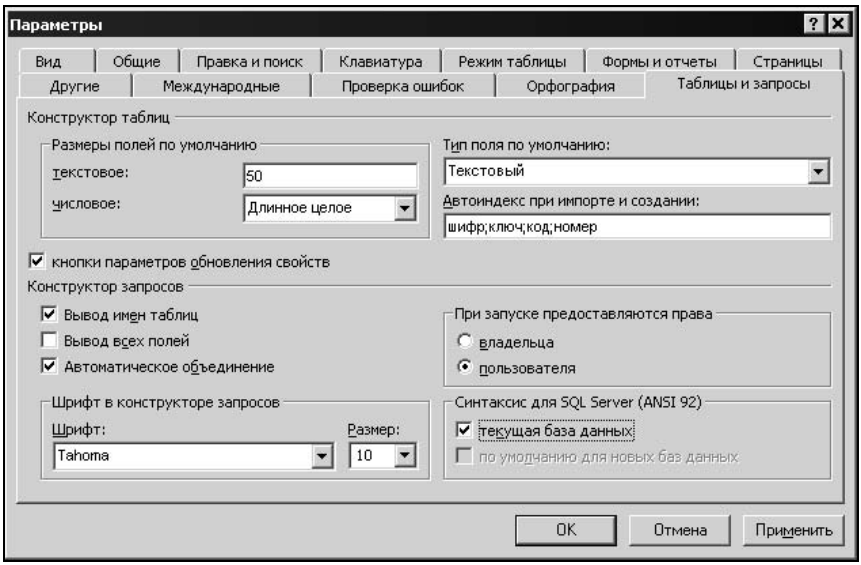


Рис. 8.59. Диалоговое окно **Параметры**, вкладка **Таблицы и запросы**

Если текущая база данных имеет формат Access 2000, то доступен только один флажок — **текущая база данных** (This database). Использовать этот флажок лучше, когда в базе данных нет еще запросов. В противном случае после применения нового параметра все запросы придется протестировать и возможно часть из них переписать, т. к. они могут работать неправильно.

Если база данных имеет формат Access 2002-2003, то доступен еще один флажок — **по умолчанию для новых баз данных** (Default for new databases). При установке этого флажка во всех вновь создаваемых базах данных будет использоваться новый синтаксис языка SQL.

Совет

Применение синтаксиса ANSI SQL-92 рекомендуется в том случае, если предполагается перевод приложения в дальнейшем на SQL Server, чтобы избежать сущест-

венной переработки запросов. В остальных случаях вполне можно обойтись и стандартным для Access Jet SQL. Избегайте смены режима запросов SQL в существующем приложении.

Зарезервированные слова Jet SQL

Приведенные далее таблицы предназначены для сравнения зарезервированных слов ANSI SQL и Jet SQL.

В табл. 8.3 перечислены зарезервированные слова Jet SQL, которые идентичны зарезервированным словам ANSI SQL. Знаком звездочки помечены слова, которые доступны только в режиме ANSI SQL-92.

Таблица 8.3. *Зарезервированные слова Jet SQL*

ADD	COMMIT*	FETCH*	MAX	ROLLBACK*
ALL	CONSTRAINT	FROM	MIN	SELECT
ALTER	COUNT	FOREIGN	NOT	SET
ANY	CREATE	GRANT*	NULL	SOME
ALIAS	CREATE VIEW*	HAVING	ON	TRANSACTION*
AS	CURRENT*	IN	OR	UNION
ASC	CURSOR*	INDEX	ORDER	UNIQUE
AUTHORAZATION*	DECLARE*	INNER	OUTER	UPDATE
AVG	DELETE	INSERT	PARAMETERS	VALUE
BEGIN*	DESC	INTO	PRIMARY	VALUES
BETWEEN	DISALLOW	IS	PRIVILEGES*	WHERE
BY	DISTINCT	JOIN	PROCEDURE	WORK*
CHECK*	DROP	KEY	REFERENCES	
CLOSE*	DROP VIEW*	LEFT	REVOKE*	
COLUMN	EXISTS	LIKE	RIGHT	

Зарезервированные слова, обозначающие типы данных, не включены в табл. 8.3, т. к. соответствие типов данных ANSI SQL и Jet SQL приводится далее, в табл. 8.7. Большинство операторов сравнения в ANSI SQL и Jet SQL совпадают: =, <, <=, > и =>. Исключение составляет оператор неравенства. Оператору неравенства != в ANSI SQL соответствует оператор <> в Jet SQL.

Как и в ANSI SQL, зарезервированное слово Jet SQL `IN` может быть использовано для задания списка значений в предложении `WHERE` или списка, созданного подчиненным запросом. Оператор `IN` также может использоваться для идентификации таблицы в другой базе данных.

Хотя в последней версии Jet SQL появились новые инструкции, связанные с обработкой транзакций (раньше они реализовывались с помощью процедур VBA), тем не менее, существует отличие в механизме выполнения транзакции от рекомендуемого ANSI SQL:

- автоматический запуск транзакции невозможен. Чтобы начать транзакцию, ее необходимо явно запустить с помощью инструкции `BEGIN TRANSACTION`;
- допускается пять уровней вложения транзакций. Чтобы запустить вложенную транзакцию, воспользуйтесь инструкцией `BEGIN TRANSACTION` в контексте существующей транзакции;
- для присоединенных (связанных) таблиц транзакции не поддерживаются.

В обычном режиме ряд инструкций ANSI SQL не поддерживаются, однако их можно реализовать другими средствами Access: меню, кнопками, диалоговыми окнами. В табл. 8.4 приведен список инструкций ANSI SQL и эквивалентные им средства Access.

Таблица 8.4. *Зарезервированные слова ANSI SQL, не поддерживаемые Access SQL*

Зарезервированное слово	Эквивалент в Access
`AUTHORIZATION`	Диалоговое окно прав доступа
`BEGIN`	Метод Access VBA `BeginTrans`
`CHECK`	Свойство **Условие на значение** поля таблицы
`CLOSE`	Кнопка системного меню **Закрыть**
`COMMIT`	Метод Access VBA `CommitTrans`
`CREATE VIEW`	Режим конструктора запросов и фильтры
`CURRENT`	Запрос в режиме таблицы, области выделения записи
`CURSOR`	Запрос в режиме таблицы
`DECLARE`	Запрос в режиме таблицы (курсор поддерживается автоматически)
`DROP VIEW`	Режим Конструктора запросов
`FETCH`	Поля в форме или отчете
`GRANT`, `PRIVILEGES`, `REVOKE`	Диалоговое окно прав доступа
`ROLLBACK`	Метод Access VBA `RollbackTrans`
`TRANSACTION`	Методы транзакций в Access VBA
`VALUES`	Значения, введенные в таблицы или формы
`WORK`	Метод Access VBA `BeginTrans`

Функции и операторы Access, используемые вместо ключевых слов ANSI SQL

Часть зарезервированных слов ANSI SQL заменяется в Jet SQL операторами или функциями. В табл. 8.5 перечислены функции и операторы Jet SQL, которые соответствуют зарезервированным словам ANSI SQL.

Таблица 8.5. *Функции и операторы Access, используемые вместо зарезервированных слов ANSI SQL*

Access	ANSI SQL	Access	ANSI SQL
And	AND	Max()	MAX()
Avg()	AVG()	Min()	MIN()
Between	BETWEEN	Not	NOT
Count()	COUNT	Null	NULL
Is	IS	Or	OR
Like	LIKE	Sum()	SUM

Функция Access `IsNull()`, возвращающая значение `True(-1)`, если ее аргумент равен пустому значению, и `False(0)`, в обратном случае, не имеет эквивалента в ANSI SQL и не является заменой модификаторов `Is Null` или `Is Not Null` в предложении `WHERE`. Кроме того, Jet SQL не поддерживает статистические функции по подмножеству, аргументами которых является выражение с параметром `DISTINCT`, например `AVG (DISTINCT <имя_поля>)`. В таких случаях требуется использовать параметр `DISTINCTROW` в инструкции `SELECT`. Выражения, в которых используются операторы, такие как, например, `And` или `Or`, заключаются в инструкциях Jet SQL в скобки.

Оператор `Between...And`, который имеет синтаксис

`<выражение> [NOT] Between <значение_1> And <значение_2>`

по-разному трактуется в языках Jet SQL и ANSI SQL. В языке Jet SQL `<значение_1>` может превышать `<значение_2>`; в ANSI SQL `<значение_1>` должно быть меньше или равно `<значение_2>`.

Зарезервированные слова Jet SQL и функции, не входящие в ANSI SQL

Существует несколько зарезервированных слов Jet SQL, которые не имеют аналогов в ANSI SQL:

- `DISTINCTROW` — позволяет исключить из результирующего множества запроса повторяющиеся строки;
- `PIVOT` — используется в перекрестных запросах;
- `TRANSFORM` — создание перекрестного запроса.

Кроме того, Jet SQL позволяет использовать четыре статические функции по подмножеству, не включенные в ANSI SQL (табл. 8.6).

Таблица 8.6. *Дополнительные статические функции Access SQL*

Функция	Назначение
StdDev()	Смещенное значение среднеквадратичного отклонения для выборки
StdDevP()	Несмещенное значение среднеквадратичного отклонения для выборки

Таблица 8.6 (окончание)

Функция	Назначение
`Var()`	Значение смещенной дисперсии для выборки
`VarP()`	Значение несмещенной дисперсии для выборки

Эквивалентные типы данных Jet SQL и ANSI SQL

В табл. 8.7 перечислены типы данных ANSI SQL, эквивалентные им типы данных языка Jet SQL и допустимые синонимы.

Таблица 8.7. Эквивалентные типы данных Jet SQL и ANSI SQL

Типы данных ANSI SQL	Типы данных Jet SQL	Синоним	Примечание
`BIT`, `BIT VARYING`	`BINARY`	`VARBINARY`, `BINARY VARYING` `BIT VARYING`	Не является стандартным типом данных Access
Не поддерживается	`BIT`	`BOOLEAN`, `LOGICAL`, `LOGICAL1`, `YESNO`	В Access эквивалентно типу данных **Логический**
Не поддерживается	`TINYINT`	`INTEGER1`, `BYTE`	Целое число длиной 1 байт
Не поддерживается	`COUNTER`	`AUTOINCREMENT`	
Не поддерживается	`MONEY`	`CURRENCY`	В Access эквивалентно типу данных **Денежный**
`DATE`, `TIME`, `TIMESTAMP`	`DATETIME`	`DATE`, `TIME`	В Access эквивалентно типу данных **Дата/Время**
Не поддерживается	`UNIQUEIDENTIFIER`	`GUID`	
`DECIMAL`	`DECIMAL`	`NUMERIC`, `DEC`	
`REAL`	`REAL`	`SINGLE`, `FLOAT4`, `IEEESINGLE`	Тип данных **Числовой** (с плавающей точкой, 4 байт)
`DOUBLE PRECISION`, `FLOAT`	`FLOAT`	`DOUBLE`, `FLOAT8`, `IEEEDOUBLE`, `NUMBER`	Тип данных **Числовой** (с плавающей точкой, 8 байт)
`SMALLINT`	`SMALLINT`	`SHORT`, `INTEGER2`	Тип данных **Числовой** (2 байта)

Таблица 8.7 *(окончание)*

Типы данных ANSI SQL	Типы данных Jet SQL	Синоним	Примечание
INTEGER	INTEGER	LONG, INT, INTEGER4	Тип данных **Числовой** (длинное целое)
INTERVAL	Не поддерживается		
Не поддерживается	IMAGE	LONGBINARY, GENERAL, OLEOBJECT	Тип данных поля объекта OLE
Не поддерживается	TEXT	LONGTEXT, LONGCHAR, MEMO, NOTE, NTEXT	Тип данных **Поле MEMO**
CHARACTER, CHARACTER VARYING, NATIONAL CHARACTER, NATIONAL CHARACTER VARYING	CHAR	TEXT(n), ALPHANUMERIC, CHARACTER, STRING, VARCHAR, CHARACTER VARYING, NCHAR, NATIONAL CHARACTER, NATIONAL CHAR, NATIONAL CHARACTER VARYING, NATIONAL CHAR VARYING	Тип данных **Текстовый**

Реализация SQL процессора Jet 4.*x* имеет существенные отличия в типах данных от предыдущей версии Jet SQL, а именно:

- тип TIMESTAMP теперь не является синонимом типа DATETIME;
- тип NUMERIC теперь используется как синоним типа DECIMAL, а не FLOAT или DOUBLE, как это было раньше;
- поля типа LONGTEXT и CHAR всегда хранятся в формате представления знаков Unicode, который эквивалентен такому типу данных ANSI SQL, как NATIONAL CHAR;
- если имя типа данных TEXT используется без указания необязательной длины, то создается поле типа LONGTEXT. В результате появляется возможность применять инструкции CREATE TABLE для создания типов данных, совместимых с Microsoft SQL Server;
- если имя типа данных TEXT используется с указанием необязательной длины, например TEXT(25), то тип данных соответствующего поля будет эквивалентен типу CHAR. В результате сохраняется обратная совместимость с большинством приложений, использующих процессор Microsoft Jet, а также согласованность типа данных TEXT (без указания длины) с Microsoft SQL Server;
- тип данных BIT стандарта ANSI SQL не соответствует типу данных BIT языка Jet SQL. Он совпадает с типом данных BINARY. Для типа данных BIT языка Jet SQL нет эквивалентного типа в ANSI SQL.

Подстановочные знаки при сравнении строк

В данном случае речь идет о трех подстановочных знаках, приведенных в табл. 8.8.

Таблица 8.8. *Соответствие подстановочных знаков*

Jet SQL	ANSI SQL	Примечание
?	_ (подчеркивание)	Любой одиночный символ
*	%	Произвольное количество символов
#	не имеет эквивалента	Любая цифра от 0 до 9
`[список_знаков]`	не имеет эквивалента	Любой одиночный знак, входящий в `список_знаков`
`[!список_знаков]`	не имеет эквивалента	Любой одиночный знак, не входящий в `список_знаков`

Язык SQL Microsoft Jet поддерживает использование в операторе `Like` как *подстановочных знаков* ANSI SQL (_ и #), так и подстановочных знаков ядра Microsoft Jet (? и *). Однако подстановочные знаки ANSI SQL доступны только в режиме ANSI SQL-92, в противном случае они воспринимаются просто как строковые константы. И наоборот, в режиме ANSI SQL-92 подстановочные знаки Jet SQL трактуются как строковые константы. Таким образом, нельзя в одном запросе смешивать эти знаки.

Далее перечислены важные правила установления совпадения оператором `Like`.

- Восклицательный знак (!) в начале `списка_знаков` означает, что совпадение наступит, если в выражении будет найден любой знак, отсутствующий в списке. Восклицательный знак, использованный вне квадратных скобок, соответствует самому себе.
- Знак дефиса (-) можно использовать в начале (после восклицательного знака, если он есть) или в конце `списка_знаков` для установления соответствия с самим собой. В любом другом месте знак дефиса означает диапазон символов ANSI.
- Если указан диапазон знаков, его границы должны следовать в возрастающем порядке (А—Я или 0—100). [А—Я] является правильным образцом, а [Я—А] — неправильным.
- Квадратные скобки ([ ]) игнорируются; они рассматриваются как пустая строка ("").

Зарезервированные слова *DISTINCTROW* и *DISTINCT*

Зарезервированные слова `DISTINCTROW` и `DISTINCT` в инструкции `SELECT` Jet SQL позволяют исключить из результирующего множества повторяющиеся строки. Разница между этими словами состоит в том, как при их использовании определяются повторяющиеся записи.

Если в инструкции `SELECT` используется слово `DISTINCTROW`, то для сравнения записей применяются все поля исходной таблицы, независимо от того, какие из этих полей включены в запрос.

Если в инструкцию SELECT входит слово DISTINCT, то для сравнения записей используются данные в строках результирующего множества запроса, т. е. только те поля, которые включены в запрос.

Замечание

Модификаторы могут использоваться не только в инструкции SELECT, но также и в запросах на добавление записей и на создание таблицы.

Для того чтобы понять, как влияет включение этих слов в запрос на его результат, построим запрос, который будет показывать, какие товары приобрел каждый клиент.

1. Создайте новый запрос в базе данных Борей. Для этого раскройте список запросов окна базы данных и нажмите кнопку **Создать** (New Query). В списке диалогового окна **Новый запрос** (New Query) выделите значение **Конструктор** (Design View) и нажмите кнопку **ОК**.
2. Добавьте в запрос таблицы Клиенты (Customers), Заказы (Orders), Заказано (Orders Details) и Товары (Products). Access автоматически создает необходимые связи между таблицами.
3. Перетащите поле Название (CompanyName) из списка полей таблицы Клиенты (Customers) в первый столбец бланка запроса. Выделите ячейку **Сортировка** (Sort) и выберите значение **По возрастанию** (Ascending).
4. Перетащите поле Марка (ProductName) из списка полей таблицы Товары (Products) во второй столбец бланка запроса. Задайте в этом столбце также сортировку по возрастанию.
5. Нажмите на панели инструментов кнопку **Запуск** (Run Query) для выполнения запроса.

Вы должны получить результирующее множество, которое содержит 2 169 записей. При этом те клиенты, которые не сделали ни одного заказа, в результирующее множество не попадут, однако клиенты, которые заказывали один и тот же товар несколько раз, могут попасть в таблицу несколько раз. Теперь добавим в инструкцию SQL SELECT слово DISTINCTROW:

1. Выберите команду меню **Вид | Режим SQL** (View | SQL View) или щелкните по стрелке на кнопке **Вид** (View) на панели управления и выберите элемент **Режим SQL** (SQL View). Откроется диалоговое окно, в котором выведена инструкция SQL, соответствующая построенному запросу.
2. Введите слово DISTINCTROW сразу после слова SELECT. Запрос должен выглядеть так, как показано на рис. 8.60.

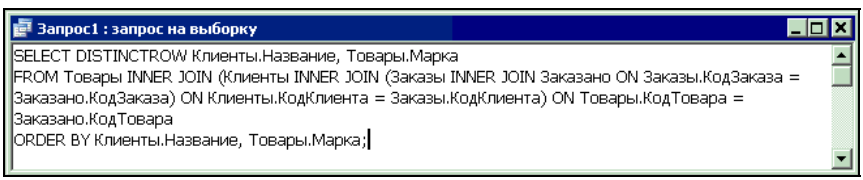


Рис. 8.60. Инструкция SQL с модификатором DISTINCTROW

3. Нажмите кнопку **Запуск** (Run). Результирующее множество запроса будет включать 1695 записей.
4. Снова выполните команду **Вид | Режим SQL** (View | SQL View) и замените в инструкции SQL слово `DISTINCTRTOW` словом `DISTINCT`.
5. Нажмите кнопку **Запуск** (Run). Результирующее множество запроса содержит те же 1695 строк, что и при использовании ключевого слова `DISTINCTROW`.

В данном случае разницы между употреблением этих модификаторов нет (в других случаях может получиться иной результат). Однако при попытке редактировать данные в таблице вы убедитесь, что при использовании модификатора `DISTINCT` результирующий набор записей получается необновляемым, а модификатор `DISTINCTROW` допускает изменение данных в результирующем наборе.

Вместо того чтобы вносить эти модификаторы вручную в режиме SQL, можно просто установить соответствующие свойства запроса. Чтобы открыть диалоговое окно свойств запроса, щелкните правой кнопкой мыши по свободному полю в верхней панели окна Конструктора запроса и выберите из контекстного меню команду **Свойства** (Properties). Появляется диалоговое окно, представленное на рис. 8.58.

Два свойства в этом окне определяют использование модификаторов `DISTINCTROW` и `DISTINCT`: **Уникальные значения** (Unique Values) и **Уникальные записи** (Unique Rows). Свойство **Уникальные значения** (Unique Values) соответствует модификатору `DISTINCT`, а свойство **Уникальные записи** (Unique Rows) — модификатору `DISTINCTROW`. Если вы попробуете установить эти значения, то увидите, что Access не позволит установить значения **Да** (Yes) для обоих свойств. Если одно из них имеет значение **Да** (Yes), то для второго автоматически устанавливается значение **Нет** (No). Если оба свойства имеют значение **Нет** (No), запрос будет включать в результирующий набор все записи.

Замечание

Свойство **Уникальные записи** (Unique Rows) имеет смысл только для запроса, который строится по нескольким таблицам. Если запрос имеет только одну таблицу, значение этого свойства игнорируется.

Использование разделителей и символов SQL

Кроме используемых в выражениях операторов сравнения разделителями в SQL являются запятая, точка, точка с запятой и двоеточие. Описанные далее разделители используются с небольшими отличиями в ANSI SQL и в Jet SQL. Так в Jet SQL:

- запятые используются для разделения членов списков, например имен полей: `Имя, Фамилия, Отчество, [Год рождения}, Адрес, Город, Индекс`;
- для задания имен полей, которые содержат недопустимые символы (например, пробелы), используются квадратные скобки: `[Дата размещения]`;
- если в запрос включены поля нескольких таблиц, то используется полное имя поля, которое состоит из двух частей: имени таблицы и имени поля, между которыми ставится разделитель — точка: `Заказы.КодЗаказа`;
- строки в ANSI SQL заключаются в одиночные кавычки (`'`). Строки в Jet SQL можно заключать как в одиночные, так и в двойные кавычки (`"`). При использо-

вании инструкций SQL в процедурах VBA рекомендуется использовать одиночные кавычки;

- в конце инструкции Jet SQL обязательно ставится точка с запятой;
- в инструкциях Jet SQL нельзя использовать переменные.

В инструкциях SQL, разбитых на несколько строк, часто используются отступы. Отступы указывают на продолжение предыдущей строки или на фрагмент, связанный с расположенным на предыдущей строке ключевым словом.

Использование инструкций SQL для создания запросов

Есть несколько типов запросов, которые невозможно создать с помощью Конструктора запросов. Их можно создать, вводя соответствующую инструкцию в окне SQL. Для этого нужно:

1. Открыть список запросов и нажать кнопку **Создать** (New).
2. Закрыть диалоговое окно **Добавление таблицы** (Show Table), не добавляя таблиц.
3. Выбрать команду **Вид | Режим SQL** (View | SQL View) для вывода окна SQL.
4. Ввести в окне инструкцию SQL, используя для ввода новой строки комбинацию клавиш <Ctrl>+<Enter>.
5. Нажать на панели инструментов кнопку **Запуск** (Run), чтобы вывести результирующее множество записей.

Далее рассматриваются запросы, которые могут быть введены только в режиме SQL.

Создание запросов на объединение записей (*Union*)

Этот тип запроса позволяет объединить в одном результирующем наборе результаты нескольких запросов, таблиц и инструкций `SELECT`.

Синтаксис этого запроса следующий:

```
[TABLE] запрос_1 UNION [ALL] [TABLE] запрос_2 [UNION [ALL] [TABLE] запрос_n
[ ... ]],
```

где `запрос_1`, `запрос_2`, ... `запрос_n` — инструкция `SELECT`, имя сохраненного запроса или имя сохраненной таблицы, перед которым стоит зарезервированное слово `TABLE`.

В каждом аргументе `запрос_i` допускается использование предложения `GROUP BY` или `HAVING` для группировки возвращаемых данных. В конец последнего аргумента `запрос_n` можно включить предложение `ORDER BY`, чтобы отсортировать возвращенные данные. Таким образом, группировка записей может выполняться в каждом из объединяемых запросов или таблиц, а сортировка выполняется только в результирующем множестве записей запроса на объединение. В качестве поля для сортировки нужно указывать имена полей из первого запроса или таблицы, т. к. именно эти имена будут использоваться в качестве имен полей в результирующем множестве запроса.

По умолчанию записи, повторяющиеся в объединяемых таблицах и запросах, не включаются в результирующее множество запроса. Использование предиката ALL в запросе позволяет включить все записи. Кроме того, такие запросы выполняются быстрее.

Все запросы, включенные в операцию UNION, должны отбирать одинаковое число полей; при этом типы данных и размеры полей не обязаны совпадать. Если число полей не совпадает, выводится сообщение об ошибке.

Для создания запроса на объединение существует специальная команда меню **Запрос | Запрос SQL | Запрос на объединение** (Query | SQL Specific | Union), которая доступна в режиме Конструктора запросов. При выполнении этой команды открывается пустое окно **Режим SQL** (SQL View), в которое нужно ввести инструкцию.

На рис. 8.61 показана инструкция SQL, содержащая операцию UNION, для запроса Клиенты и поставщики по городам (Customers and Suppliers by City), который содержится в базе данных Борей. Чтобы открыть окно SQL, откройте список запросов в окне базы данных, выделите запрос Клиенты и поставщики по городам (Customers and Suppliers by City) и нажмите кнопку **Конструктор** (Design). При этом кнопка и команда меню **Вид | Конструктор** (View | Design View) становятся недоступными.

Совет

При работе в режиме SQL можно изменить используемый шрифт (на наш взгляд он слишком мелкий). Для этого нужно открыть окно **Параметры** (Options), вкладку **Таблицы и запросы** (Tables/Queries) и установить **Шрифт в конструкторе запросов** (Query design font).

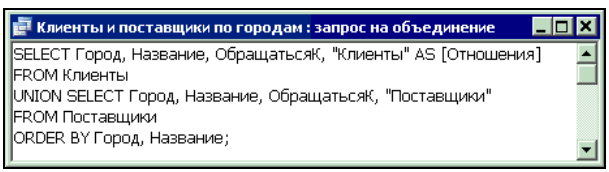


Рис. 8.61. Запрос на объединение



Рис. 8.62. Результирующее множество запроса Клиенты и поставщики по городам

Результат выполнения запроса на объединение `Клиенты и поставщики по городам` (Customers and Suppliers by City) приведен на рис. 8.62.

Создание запроса к серверу

Еще одним типом запроса SQL является запрос к серверу. Он обрабатывается не процессором Jet, как все остальные запросы, а непосредственно передается на сервер базы данных, к которому выполняется обращение, например Microsoft SQL Server. Главной особенностью этого запроса является то, что он должен использовать синтаксис языка SQL сервера базы данных (для Microsoft SQL Server — это Transact-SQL). Создается подобный запрос аналогично запросу на объединение с помощью команды меню **Запрос | Запрос SQL | К серверу** (Query | SQL Specific | Pass-Through). Запросы к серверу используются в приложениях, имеющих архитектуру "клиент-сервер".

Управляющие запросы

Управляющие запросы являются третьим типом запросов, которые создаются в окне **Режим SQL** (SQL View) и используют инструкции Jet SQL. Такие запросы позволяют работать с таблицами и индексами: создавать, изменять и удалять таблицы, создавать индексы.

Инструкции SQL, которые при этом используются, относятся к подмножеству SQL, называющемуся DDL (Data Definition Language) — язык определения данных. Набор этих инструкций в Jet SQL существенно сокращен по сравнению со стандартом ANSI SQL, т. к. многие операции можно выполнить с помощью других средств (команд меню или окна базы данных).

В обычном режиме можно использовать следующие инструкции SQL:

- `CREATE TABLE` — создает таблицу;
- `CONSTRAINT` — создает уникальный индекс, в том числе по первичному ключу, а также применяется для установления отношений между таблицами;
- `ALTER TABLE` — изменяет структуру таблицы, созданной с помощью инструкции `CREATE TABLE`;
- `DROP` — удаляет таблицу из базы данных или удаляет индексы в таблице;
- `CREATE INDEX` — создает индекс для поля или группы полей.

В режиме ANSI SQL-92 есть дополнительные инструкции:

- `CREATE PROCEDURE` — создает хранимую процедуру;
- `CREATE VIEW` — создает новое представление;
- `CREATE USER` или `GROUP` — создает одного или нескольких новых пользователей или группу;
- `ADD USER` — добавляет одного или нескольких существующих пользователей к существующей группе;
- `DROP USER` или `GROUP` — удаляет одного или нескольких существующих пользователей или группы, либо исключает одного или нескольких существующих пользователей из существующей группы;

- ALTER USER или DATABASE — изменяет пароль существующего пользователя или базы данных;
- GRANT — предоставляет конкретные привилегии существующему пользователю или группе;
- REVOKE — отменяет конкретные привилегии существующего пользователя или группы.

Замечание

Запросы данного типа обычно используются в программах на языке Visual Basic для приложений (VBA) тогда, когда нужно выполнить соответствующие операции программно. В противном случае для этого гораздо удобнее и быстрее использовать средства пользовательского интерфейса Access.

В рамках данной книги мы не описываем подробно синтаксис всех инструкций SQL. Заинтересованному читателю рекомендуется обратиться к разделу **Справочник Microsoft Jet SQL** (Microsoft Jet SQL Reference) в оглавлении **Справки Access** (Access Help).

Создание подчиненных запросов

Инструкции Jet SQL, такие как SELECT, SELECT...INTO, INSERT...INTO, DELETE или UPDATE, позволяют для вычисления предиката в предложении WHERE использовать другой запрос. Этот запрос называется *подчиненным запросом*.

Подчиненный запрос включается в главный запрос одним из следующих способов:

- сравнение [ANY | ALL | SOME] (инструкцияSQL);
- выражение [NOT] IN (инструкцияSQL);
- [NOT] EXISTS (инструкцияSQL).

Где:

- сравнение — выражение и оператор сравнения, который сравнивает это выражение с результатами подчиненного запроса;
- выражение — выражение, для которого проводится поиск в результирующем наборе записей подчиненного запроса;
- инструкцияSQL — инструкция SELECT, которая представляет подчиненный запрос.

Предикаты ANY или SOME являются синонимами и используются, если в главном запросе нужно выбрать записи, удовлетворяющие сравнению со всеми записями, выбранными в подчиненном запросе. Далее приведен пример запроса, который выбирает из таблицы Товары (Products) все товары, цена которых не ниже, чем цена товаров у конкурентов:

```
SELECT * FROM Товары
WHERE Товары.Цена > ANY
(SELECT ТоварыКонкурентов.Цена FROM ТоварыКонкурентов)
```

Предикат ALL используется для выбора в главном запросе только тех записей, которые удовлетворяют сравнению со всеми записями, выбранными в подчиненном за-

просе. В следующем примере выбираются все заказы, сделанные в 1998 году, стоимость которых ниже стоимости любого заказа, размещенного в 1997 году:

```
SELECT DISTINCTROW Заказы.КодЗаказа
FROM Заказы
WHERE Year(Заказы.ДатаРазмещения) = 1998
      AND Заказы.СуммаЗаказа < ALL
      (SELECT Заказы.СуммаЗаказа FROM Заказы
       WHERE Year(Заказы.ДатаРазмещения) = 1997);
```

Предикат `IN` используется для выбора в главном запросе только тех записей, которые содержат значения, совпадающие с одним из значений, выбранных подчиненным запросом. Например, чтобы выбрать клиентов, которые разместили заказы в январе 1998 года, можно написать следующий запрос:

```
SELECT * FROM Клиенты
WHERE КодКлиента IN
(SELECT КодКлиента FROM Заказы
WHERE ДатаРазмещения BETWEEN #1/1/98# AND #31/1/98#;
```

И наоборот, предикат `NOT IN` используется для выбора в главном запросе только тех записей, которые содержат значения, не совпадающие ни с одним из значений, отобранных подчиненным запросом.

Предикат `EXISTS` используется в логическом выражении для определения того, должен ли подчиненный запрос возвращать какие-либо записи. Например, чтобы выбрать всех поставщиков для товаров в некотором заказе, можно использовать следующую инструкцию SQL:

```
SELECT DISTINCTROW Поставщики.Название
FROM Поставщики
WHERE Exists
(SELECT Заказано.КодТовара
FROM Заказано
WHERE КодЗаказа = 121 AND Заказано.КодТовара = Поставщики.КодТовара);
```

В подчиненном запросе можно использовать псевдонимы таблиц для ссылки на таблицы, перечисленные в предложении `FROM`, расположенном вне подчиненного запроса. В следующем примере выбираются фамилии и имена сотрудников, чья зарплата равна или больше средней зарплаты сотрудников, имеющих ту же должность. В предыдущем примере можно присвоить таблице `Сотрудники` (Emploees) псевдоним `T1`, и тогда запрос будет выглядеть следующим образом:

```
SELECT DISTINCTROW Поставщики.Название
FROM Поставщики As П1
WHERE Exists
(SELECT Заказано.КодТовара
FROM Заказано
WHERE КодЗаказа = 121 AND Заказано.КодТовара = П1.КодТовара);
```

Кроме приведенных примеров, когда подчиненные запросы используются в предложении `WHERE`, их можно применить в предложении `HAVING` вместо выражения для

группировки записей, в списке полей после слова SELECT и даже в предложении FROM вместо имени таблицы, причем в последнем случае можно использовать синонимы таблиц, как в инструкциях Transact-SQL.

Некоторые подчиненные запросы можно использовать в перекрестных запросах как предикаты в предложении WHERE. Подчиненные запросы, которые используются для вывода результатов (в списке SELECT), нельзя применять в перекрестных запросах.

В отличие от запросов на объединение, подчиненный запрос можно создать в режиме Конструктора. В строке **Условия отбора** (Criteria) в качестве условия отбора следует указать, например, зарезервированное слово IN, а затем ввести инструкцию SELECT. А можно сначала создать в режиме Конструктора подчиненный запрос, переключиться в режим SQL, скопировать получившуюся инструкцию SQL в буфер обмена, а затем создать главный запрос и скопировать в строку **Условия отбора** (Criteria) подчиненный запрос из буфера.

ГЛАВА 9



Создание составных форм

Формы являются основным объектом, который обеспечивает пользователю удобный доступ к данным, поэтому им уделено большое внимание в настоящей книге. В первой части мы показали несколько способов создания форм и детально описали основной инструмент их разработки — Конструктор. Вы убедились, какое многообразие возможностей предоставляет этот инструмент для построения удобных и красивых форм. Однако в *гл. 5* мы рассмотрели лишь самые основные вопросы, касающиеся работы с формами и элементами управления. Здесь мы поговорим об этом более подробно, и обсудим более сложные вопросы, касающиеся создания и оформления составных форм — многотабличных и многостраничных, и рассмотрим применение элементов управления всех типов. Материал этой главы является продолжением главы 5, поэтому рекомендуется ознакомиться с ней, прежде чем перейти к дальнейшему изучению. Часть материала, касающегося внедрения рисунков и других объектов в форму, вынесена в главу 15. Мы будем еще раз возвращаться к формам в главах 12 и 13 при рассмотрении программирования в формах и отчетах с помощью макросов и процедур VBA.

Использование Панели элементов

Панель элементов (Toolbox) является основным инструментом для добавления элементов управления в форму в режиме Конструктора. Внешний вид Панели элементов (Toolbox) представлен на рис. 9.1.

Отобразить Панель элементов (Toolbox) на экране можно тремя способами.

- Выберите команду **Вид | Панель элементов** (View | Toolbox).
- Нажмите кнопку **Панель элементов** (Toolbox) на панели инструментов **Конструктор форм** (Form Design).

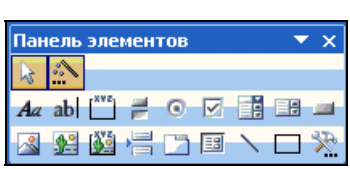


Рис. 9.1. Панель элементов

- Щелкните правой кнопкой мыши по любой панели инструментов и выберите в контекстном меню команду **Панель элементов** (Toolbox).

Наиболее часто применяемые элементы управления расположены прямо на панели. Остальные элементы отображаются в виде дополнительного списка, если нажать кнопку **Другие элементы** (More Controls) в правом нижнем углу Панели элементов (Toolbox).

В верхней части Панели элементов расположены две специальные кнопки: **Выбор объектов** (Select Objects) и **Мастера** (Control Wizards). Кнопка **Выбор объектов** (Select Objects) в нажатом состоянии означает, что в данный момент включен режим выделения элементов, т. е. реакцией на щелчок мыши в форме будет выделение соответствующего элемента управления или другого объекта, попавшего в указанную область. Этот режим включен по умолчанию. Однако если нажать другую кнопку на Панели элементов (Toolbox), предназначенную для создания элемента управления, кнопка выбора объектов автоматически отжимается. Ее нужно снова нажать, если вы передумали создавать элемент и хотите вернуться в режим выделения объектов. Нажатая кнопка **Мастера** (Control Wizards) означает, что при размещении элемента управления на форме запустится соответствующий Мастер. При отжатой кнопке Мастера не запускаются, и элементы требуется настраивать вручную. Мастера элементов помогают создавать элементы управления, требующие определенной настройки, такие как поля со списком, командные кнопки, группы элементов выбора. Практически для каждого элемента управления существует свой Мастер. Мастера элементов удобно использовать как на начальном этапе работы с Access для обучения, так и в дальнейшей работе, поскольку они автоматизируют рутинную операцию стандартной настройки элемента, представляя ее в виде удобного диалога.

Кроме Мастеров элементов, работу по настройке элементов облегчают специальные Мастера, помогающие в создании нетривиальных значений свойств элемента, их называют Построителями. Например, Построитель запросов (Query Builder) помогает создать инструкцию SQL, которая определяет источник строк для списков и полей со списками. Построитель выражений (Expression Builder) помогает создавать синтаксически правильные выражения для вычисляемых элементов управления.

Основные кнопки, расположенные на Панели элементов (Toolbox) предназначены для создания элементов управления разных типов. Название типа элемента управления отображается в виде всплывающей подсказки при наведении указателя мыши на соответствующую кнопку Панели элементов (Toolbox).

Панель элементов (Toolbox) обычно отображается как "плавающая" (см. рис. 9.1), так что ее можно свободно перемещать по экрану, находя для нее удобное место. Перемещая с помощью мыши рамку Панели элементов (Toolbox), можно изменить ее размеры или закрепить у границы окна, например, как это показано на рис. 9.2.

Встроенные элементы управления

В виде кнопок непосредственно на Панели элементов (Toolbox) размещаются лишь встроенные элементы управления, в отличие от внедряемых объектов и элементов управления ActiveX, которые могут быть добавлены в приложение. *(См. также гл. 15 "Интеграция Access 2003 с компонентами Office 2003".)*

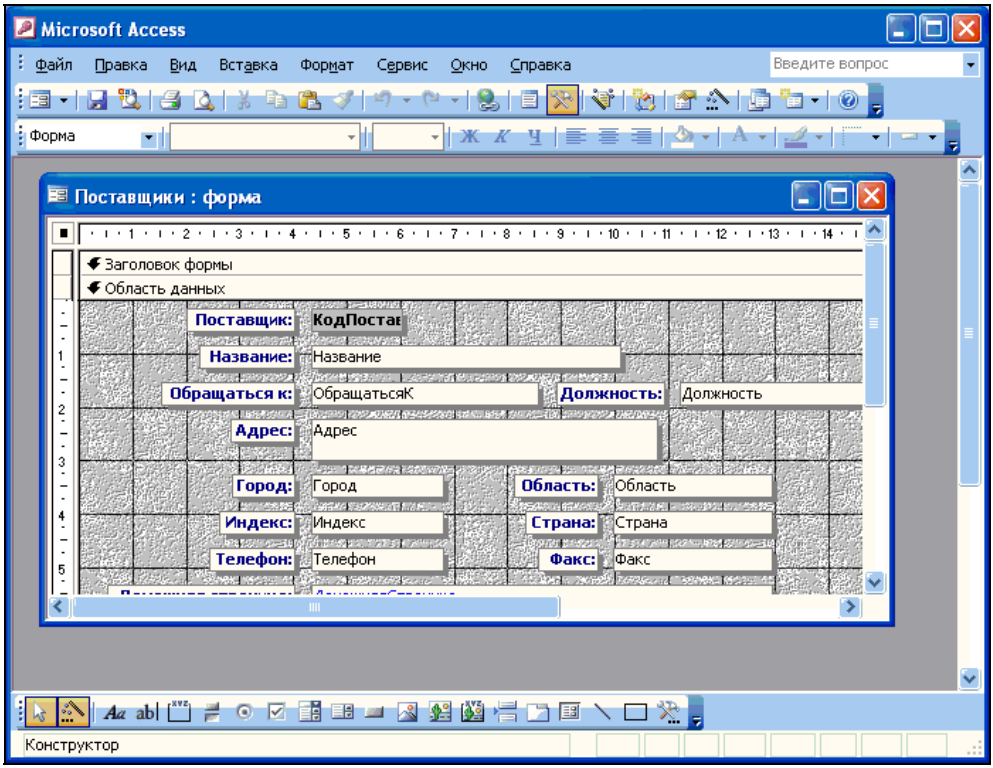


Рис. 9.2. Закрепленная Панель элементов (внизу окна)

Разновидности элементов управления по типу содержимого

В Access существует три разновидности элементов управления, в зависимости от типа содержимого в них, т. е. от способа заполнения их данными:

- присоединенные;
- свободные;
- вычисляемые.

К какому из перечисленных видов относится данный элемент управления, определяется тем, как задан для него источник данных.

Присоединенные элементы управления связаны с полями базовой таблицы, т. е. той таблицы, которая является источником данных для формы. Если источником данных является запрос, то присоединенные элементы управления могут связываться с полями в разных таблицах. В присоединенном элементе отображаются данные, которые содержатся в связанном с ним поле таблицы, и при изменении этих данных пользователем соответствующим образом обновляется и значение в поле таблицы. В присоединенных элементах можно отображать все доступные в Access типы данных, в том числе объекты OLE и гиперссылки.

Свободные элементы управления не связаны с таблицами. Они предназначены либо для ввода информации, которая не предназначена для непосредственного редактирования данных в источнике, а используется в других целях (обычно макросами или программами VBA), либо для отображения объектов OLE, которые хранятся в самих формах. Свободными элементами являются также все элементы, не связанные с какими-либо данными, а предназначенные лишь для улучшения визуального восприятия форм, такие как: линии, прямоугольники, рисунки.

Вычисляемые элементы управления — это такие элементы, значения которых вычисляются на основе значений других элементов. В качестве источника данных для этих элементов используются выражения и функции.

Элемент управления *Надпись*

Элемент **Надпись** (Label) используется для размещения в форме текста: заголовков полей, заголовка формы, различных поясняющих надписей. Надписи бывают двух типов:

- присоединенные;
- свободные.

Присоединенные надписи — это те надписи, что не существуют самостоятельно, а связаны с другими элементами управления и служат их заголовками, т. к. других видимых на экране заголовков у этих элементов нет. К подобным элементам относятся, например, текстовые поля, поля со списками, флажки, переключатели и др. Такого типа надписи автоматически создаются при размещении соответствующего элемента управления на форме и присоединяются к нему. При отображении формы в режиме Таблицы присоединенные надписи становятся заголовками столбцов.

Надписи, которые создаются с помощью кнопки **Надпись** (Label) на Панели элементов (Toolbox), являются свободными, они не связываются ни с какими элементами управления. Эти надписи служат для оформления формы или для помощи пользователю, однако они не отображаются на экране, если форма выводится в режиме Таблицы.

Свободную надпись можно при необходимости присоединить к элементу управления:

1. Выделите элемент управления **Надпись** (Label), который нужно присоединить к другому элементу.
2. Нажмите кнопку **Вырезать** (Cut) на панели инструментов (или нажмите комбинацию клавиш <Ctrl>+<X>).
3. Выделите элемент управления, к которому нужно присоединить надпись.
4. Нажмите кнопку **Вставить** (Paste) на панели инструментов (или нажмите комбинацию клавиш <Ctrl>+<V>).

Надпись окажется присоединенной к элементу управления.

Текст надписи определяется свойством **Подпись** (Caption), его можно изменить в диалоговом окне свойств надписи или непосредственно, щелкнув по выделенной надписи на форме. Текст надписей, как свободных, так и присоединенных, можно изменять динамически при работе приложения с помощью макросов или программ VBA. Можно также выводить новые надписи в форме или скрывать существующие

при выполнении определенных условий. Максимальная длина текста надписи — 2048 символов. *(См. также гл. 13 "Программирование на языке VBA".)*

Кроме обычного текста, надпись (не присоединенная к другому элементу) может также содержать гиперссылку. В этом случае свойство **Адрес гиперссылки** (Hyperlink Address) содержит адрес документа, на который ссылается гиперссылка, в формате URL или UNC. Указать дополнительно элемент внутри данного документа, чтобы именно на него ссылалась гиперссылка, позволяет свойство **Дополнительный адрес** (Hyperlink SubAddress). Это может быть закладка в документе, идентификатор поименованного тега, ссылка на лист и интервал ячеек в рабочей книге Excel или другой адресуемый элемент в зависимости от типа документа. Например, чтобы гиперссылка указывала на 10-й слайд презентации PowerPoint, нужно задать следующие значения свойств:

- адрес гиперссылки: `C:\Мои документы\Доклад.ppt`;
- дополнительный адрес: `10`.

А чтобы гиперссылка указывала на ячейку или интервал ячеек на листе электронной таблицы Excel, дополнительный адрес должен иметь вид: `ИмяЛиста!ИмяИнтервала`, например: `Лист1!A2`, `Лист1!A2:B5` или `Лист1!Итого`.

Создать гиперссылку удобнее с помощью команды **Вставка | Гиперссылка** (Insert | Hyperlink).

Кроме надписей, гиперссылку могут содержать и элементы управления **Кнопка** (Command Button). Для этого у кнопок есть такие же свойства, задающие адрес гиперссылки.

Замечание

Надписи с гиперссылками могут быть помещены и в отчет, но при открытии отчета в Access не будут срабатывать. Однако такие гиперссылки будут работать в отчете, преобразованном в формат Word, Excel или HTML.

Элемент управления *Поле*

Элемент управления **Поле** (Text Box) наиболее часто используется, поскольку он служит для ввода и отображения данных. Обычно значение в поле вводится пользователем, однако можно задать его программно, присваивая значение свойству `Text` этого элемента. С помощью этого же свойства можно программно узнать текущее значение в поле. Длина текста, вводимого в текстовое поле, ограничена 65 535 символами.

Как и многие другие элементы управления, текстовые поля могут быть трех типов:

- присоединенными;
- свободными;
- вычисляемыми.

Мы уже описывали способы создания присоединенных и свободных текстовых полей. *(См. также разд. "Создание и удаление элементов управления", гл. 5.)*

О том, как создавать вычисляемые поля, рассказано в *разд. "Вычисления в формах" данной главы.*

Элементы управления выбора: *Выключатель*, *Переключатель*, *Флажок* и *Группа*

О предоставлении пользователю возможности выбирать значение из предложенного набора вариантов с помощью элементов управления выбора уже говорилось в *гл. 5*. Рассмотрим эти элементы управления более подробно. Три типа элементов: выключатели, переключатели и флажки — выполняют одинаковые функции и используются для представления значения логического типа данных. Они могут принимать значения `0` (Нет) или `-1` (Да). Различаются они внешним видом (рис. 9.3).

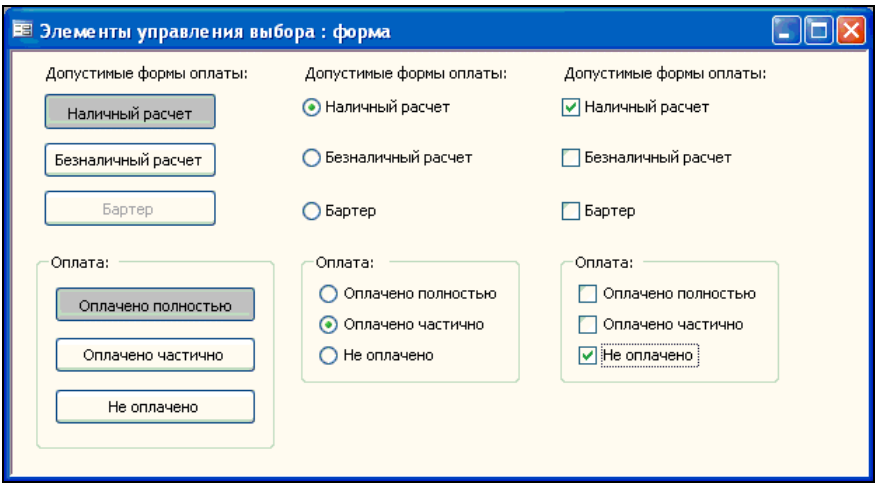


Рис. 9.3. Элементы управления выбора

В верхней строчке изображены элементы, значения которых равны **Да** (Yes), в следующей строчке — эти же элементы, когда они имеют значение **Нет** (No). В третьей строчке изображены переключатель и флажок, когда их значение не определено (неопределенное значение элемента Null отображается затемненным оформлением элемента — лучше всего это видно у элемента типа Выключатель). Выключатель имеет значение **Да** (Yes), когда он включен, и **Нет** (No), когда выключен. Переключатель имеет значение **Да** (Yes), когда он выбран, и **Нет** (No), когда не выбран. Флажок имеет значение **Да** (Yes), когда он установлен, и **Нет** (No), когда сброшен. Чтобы разрешить пользователю устанавливать третий вариант состояния — неопределенное, установите для свойства **Тройное состояние** (Triple State) значение **Да** (Yes). Если элементы выбора используются для отображения логических данных, хранящихся в поле таблицы, нужно определить для них значения по умолчанию, т. к. в противном случае в новой записи значения этих элементов не будут определены. Значение по умолчанию задается как значение одноименного свойства **Значение по умолчанию** (Default Value) либо в окне свойств элемента, либо с помощью процедуры VBA, либо посредством макроса. Всякий раз, когда пользователь щелкает мышью по элементу выбора, его значение изменяется на противоположное.

Поскольку эти элементы предназначены в первую очередь для улучшения визуального восприятия данных, то предусмотрено несколько вариантов их оформления.

Для элементов управления **Флажок** (CheckBox) и **Надпись** (Label) эти варианты представлены на рис. 9.4, остальные элементы выбора имеют те же варианты оформления.

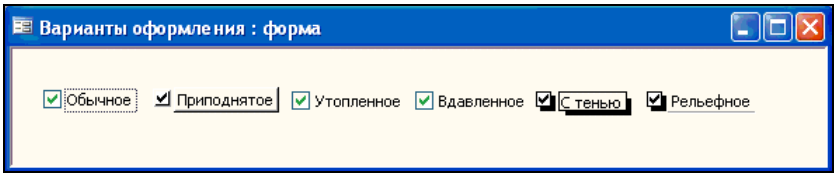


Рис. 9.4. Варианты оформления элементов выбора

Элемент управления **Группа** (Option Group) используется для того, чтобы дать возможность выбрать одно значение из небольшого набора вариантов. Пример групп элементов приведен на рис. 9.3. Если элементы выбора объединены с помощью элемента управления **Группа** (Option Group), то только один из этих элементов может быть выбран пользователем, т. е. получить значение **Да** (Yes).

Если требуется задать несколько независимых признаков, например тип оплаты и способ вывоза товара, то нужно создать несколько независимых элементов выбора или групп.

В одну группу можно объединить элементы выбора даже разных типов, но к полю таблицы присоединяется значение самой группы, а не составляющих ее элементов. Значением группы является уникальное значение, сопоставленное выбранному элементу (обычно это его порядковый номер). Чтобы привязать группу к источнику данных, установите значение свойства **Данные** (ControlSource) элемента **Группа** (Option Group). Чтобы выделить элемент **Группа** (Option Group), а не элемент управления в группе, нужно щелкнуть по рамке группы. Тогда в диалоговом окне **Свойства** (Properties) отобразятся свойства группы (рис. 9.5).

Здесь же можно указать значение по умолчанию для группы, чтобы указать, какой элемент в группе должен быть выбран по умолчанию.

Элементы управления в группе не должны присоединяться к полям таблицы (в окне свойств для сгруппированных элементов даже не отображается свойство **Данные** (ControlSource)). Зато для каждого из элементов управления в группе нужно задать то уникальное значение, о котором мы упомянули, чтобы определить значение самой группы, когда данный элемент будет выбран. Делается это с помощью свойства **Значение параметра** (Option Value), которое должно быть числовым и обязательно уникальным для каждого из элементов управления в группе (рис. 9.6). В приведенном примере заданы следующие значения свойства **Значение параметра** (Option Value): `1` — Оплачено полностью, `2` — Оплачено частично, `3` — Не оплачено.

Когда пользователь выбирает один из переключателей в группе, значение элемента управления **Группа** (Option Group) становится равным значению параметра выбранного переключателя. Это значение сохранится в поле таблицы, которое указано в качестве источника данных для группы.

Для размещения элемента управления **Группа** (Option Group) на форме можно воспользоваться соответствующим Мастером. Для демонстрации работы Мастера рассмотрим процедуру создания группы, аналогичной представленной на рис. 9.5.

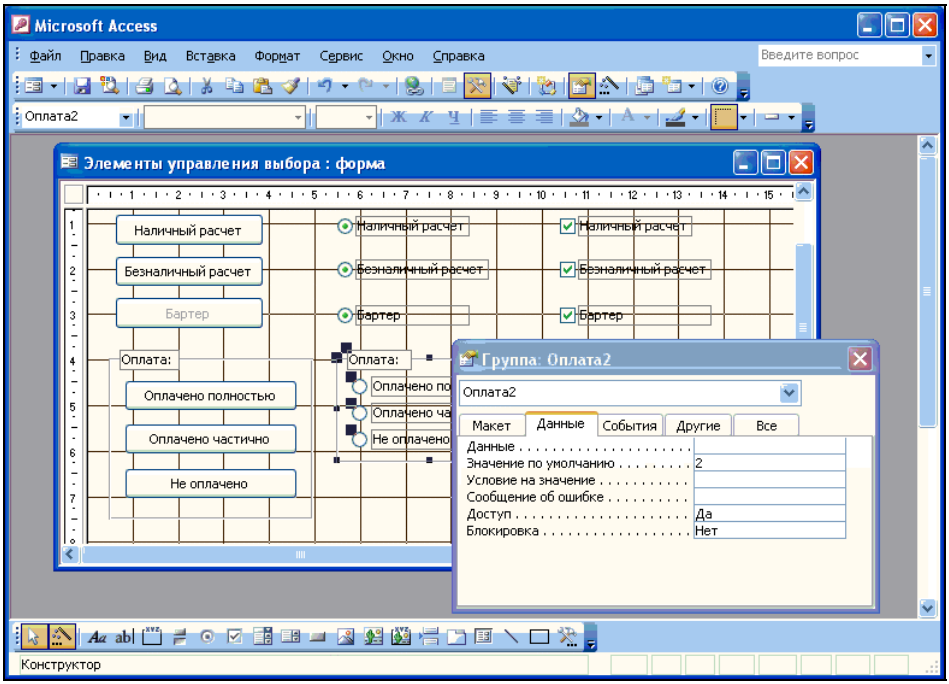


Рис. 9.5. Свойства элемента управления **Группа**

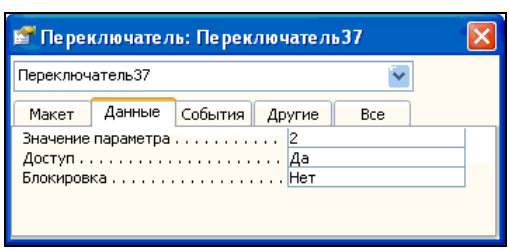


Рис. 9.6. Установка значения параметра переключателя

Создайте новую форму, открыв ее в режиме Конструктора. Затем выполните следующие действия:

1. Выберите на Панели элементов (Toolbox) элемент управления **Группа** (Option Group) и перетащите его на форму (кнопка **Мастера** (Control Wizards) при этом должна быть нажата). Запустится Мастер группы и появится окно, представленное на рис. 9.7.
2. В этом окне нужно ввести текст надписей, которые будут размещаться справа от флажков (выключателей или переключателей). Вы можете заполнить эту таблицу так, как это показано на рис. 9.7. Нажмите кнопку **Далее** (Next). Появится следующее диалоговое окно (рис. 9.8).

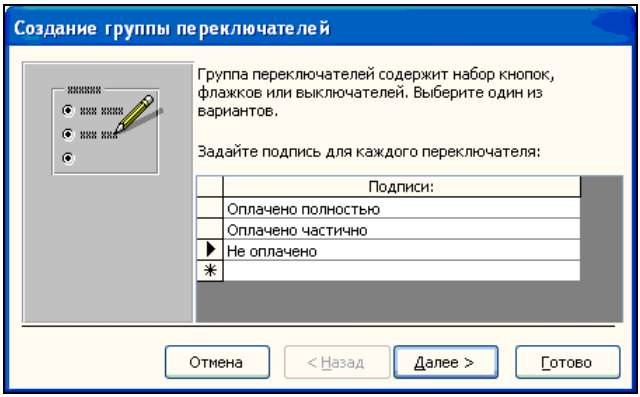


Рис. 9.7. Первое диалоговое окно Мастера группы

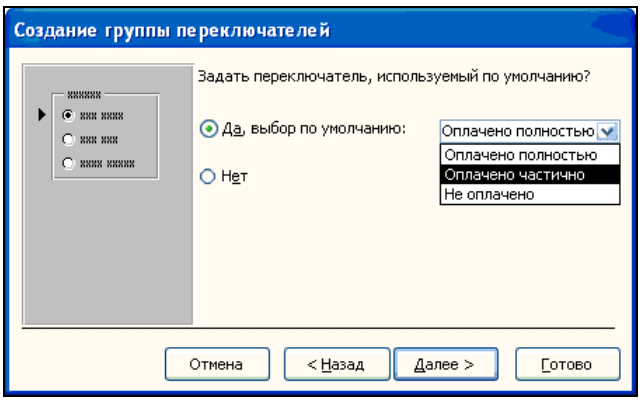


Рис. 9.8. Второе диалоговое окно Мастера группы

3. Теперь необходимо выбрать тот флажок (выключатель, переключатель), который будет установлен по умолчанию, или указать, что не нужно определять значение по умолчанию. Выберите из списка значение `Не оплачено` и нажмите кнопку **Далее** (Next).
4. В следующем диалоговом окне Мастера (рис. 9.9) нужно определить значения для каждого параметра. По умолчанию Мастер предлагает возрастающую последовательность чисел: `1`, `2`, `3`. Можете оставить эти значения и просто нажать кнопку **Далее** (Next).
5. Если данная форма связана с каким-либо источником данных, т. е. возможность создаваемую группу привязать к одному из полей в этом источнике данных. В этом случае появится диалоговое окно, представленное на рис. 9.10. В противном случае отобразится окно, изображенное на рис. 9.11, и нужно будет перейти сразу к шагу 6. В диалоговом окне, показанном на рис. 9.10, следует выбрать, будет ли создаваемая группа связана с полем источника данных или останется свободным элементом — например, чтобы значение группы было использовано в выражениях в других элементах формы, в макросах или процедурах VBA.

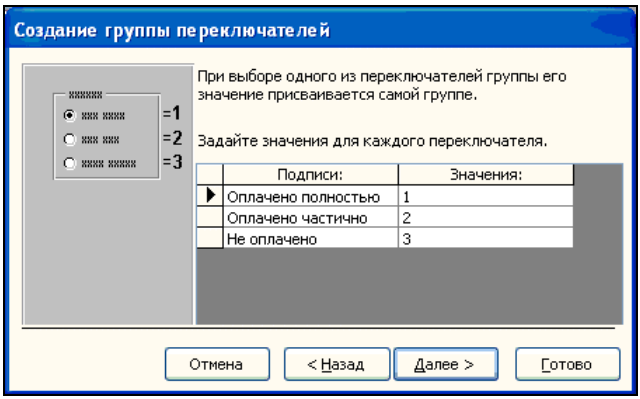


Рис. 9.9. Третье диалоговое окно Мастера группы

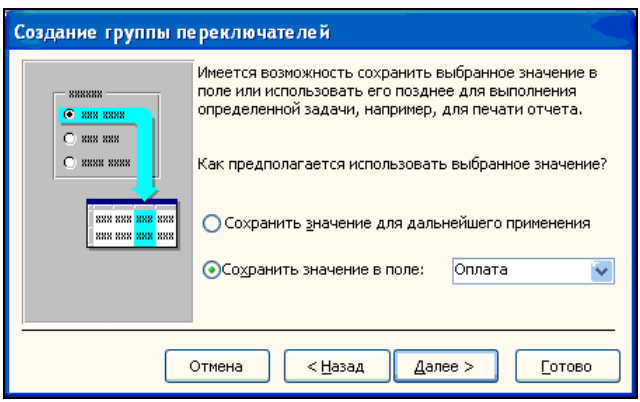


Рис. 9.10. Четвертое диалоговое окно Мастера группы

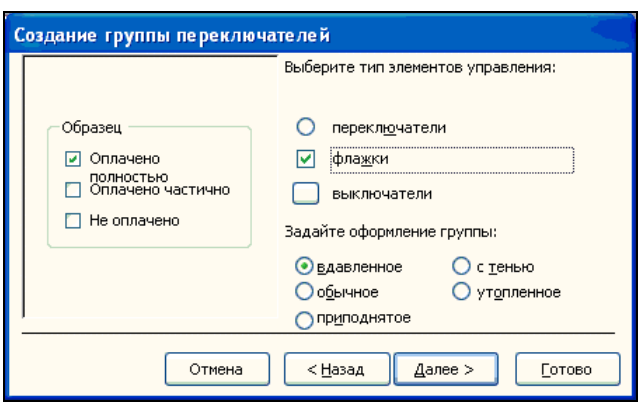


Рис. 9.11. Пятое диалоговое окно Мастера группы

Чтобы связать группу с полем из источника данных, выберите переключатель **Сохранить значение в поле** (Store the value in this field) и выберите имя поля в раскрывающемся списке. Если же нужно оставить группу свободной, оставьте выбранным переключатель **Сохранить значение для дальнейшего применения** (Save the value for later use). Нажмите кнопку **Далее** (Next).

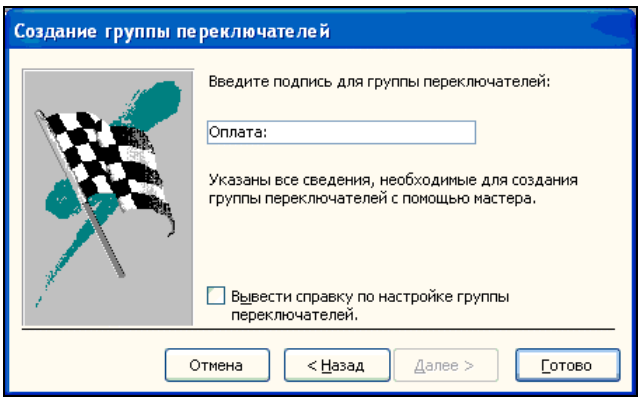


Рис. 9.12. Последнее диалоговое окно Мастера группы

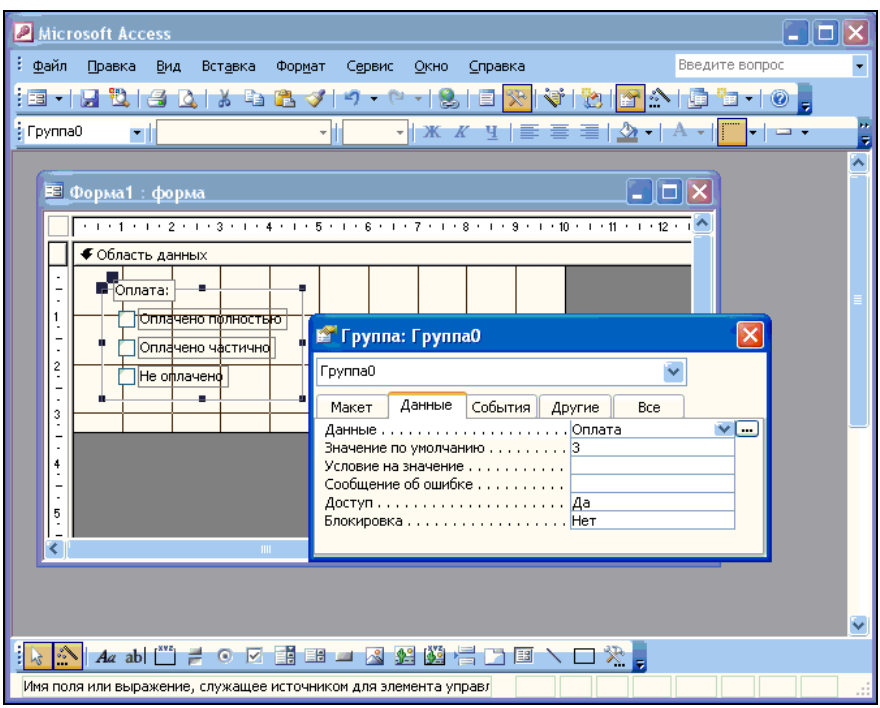


Рис. 9.13. Группа флажков, созданная с помощью Мастера

6. В следующем окне предлагается выбрать тип элементов управления в группе (флажки, выключатели или переключатели) и вариант оформления группы (один из пяти вариантов, описанных ранее). Выберите тот тип и вариант, которые вам понравятся. Обратите внимание, что слева Мастер показывает вам образец выбранного вами варианта (см. рис. 9.11).
7. В последнем диалоговом окне нужно только определить текст надписи, которая будет служить названием группы. В нашем примере требуется ввести название `Оплата` (рис. 9.12). Теперь нажмите кнопку **Готово** (Finish), и вы увидите результат работы Мастера. У вас должна получиться группа флажков, аналогичная представленной на рис. 9.13.
8. Дальше вы можете реорганизовать эту группу, как вам нравится: например, вытянуть все флажки в одну строку, перенести надписи под флажки и т. д.

Элемент управления *Список*

Элемент управления **Список** (ListBox) используется для представления на экране поля, возможные значения которого ограничиваются списком, причем списком не длинным. Это связано с тем, что такой список занимает много места на экране (рис. 9.14). И хотя можно использовать полосы прокрутки, чтобы посмотреть значения, которые не помещаются в выделенную область, очевидно, что для отображения длинных списков этот элемент управления не очень подходит.

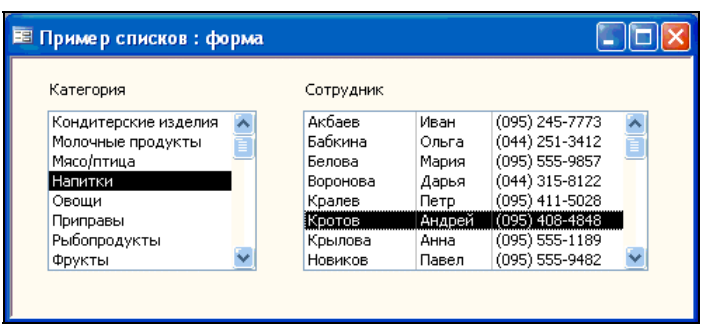


Рис. 9.14. Примеры элемента управления **Список**

Список может состоять из одного или нескольких столбцов, однако при выборе элемента списка в связанное с ним поле таблицы передается только одно значение, например, в приведенных на рис. 9.14 списках это `КодТипа` (CategoryID) и `КодСотрудника` (EmployeeID), которые в этих элементах управления даже не отображаются. Внешний вид списка и его содержимое определяются его свойствами, которые можно задать вручную, а можно при размещении списка в форме воспользоваться Мастером, который, запросив нужную информацию, определит бóльшую часть свойств автоматически. Создание списка с помощью Мастера уже демонстрировалось в *части I* книги. Мастер списков очень похож также на Мастера, с помощью которого создается элемент **Поле со списком**. *(См. разд. "Создание и удаление элементов управления", гл. 5.)*

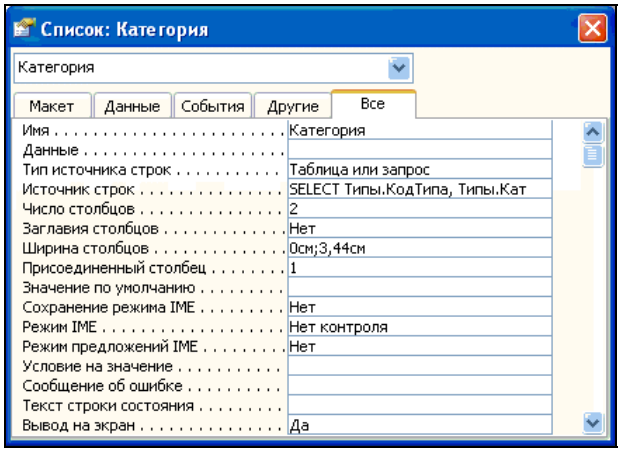


Рис. 9.15. Свойства элемента управления **Список**

Для того чтобы список правильно работал, нужно корректно определить его важнейшие свойства — это первые восемь свойств на рис. 9.15. Соответственно, если он работает неправильно, именно эти свойства нужно проверять в первую очередь.

- **Имя** (Name) — название данного элемента управления. Нужно дать ему осмысленное название, т. к. оно используется во всех ссылках на этот элемент управления в выражениях, программах и макросах.
- **Данные** (Control Source) — название поля в таблице, которое будет содержать выбранное в списке значение.
- Следующие два свойства: **Тип источника строк** (Row Source Type) и **Источник строк** (Row Source) — определяют источник данных для элементов списка. В приведенном примере в качестве источника данных используется таблица `Типы` (Categories), поэтому в свойстве **Тип источника строк** (Row Source Type) выбрано значение **Таблица или запрос** (Table/Query), а в качестве значения свойства **Источник строк** (Row Source) указана инструкция SQL, которая выбирает нужные данные из таблицы (поля `КодТипа` (CategoryID) и `Категория` (CategoryName)):

```
SELECT Типы.КодТипа, Типы.Категория FROM Типы;
```

Для английской версии Access:

```
SELECT Categories.CategoryID, Categories.CategoryName FROM Categories;
```

Чтобы элементы в данном списке были отсортированы в алфавитном порядке по полю `Категория` (CategoryName), в конец инструкции SQL нужно добавить предложение:

```
ORDER BY Типы.Категория (ORDER BY Categories.CategoryName).
```

Замечание

Значением этого свойства может быть также название таблицы или запроса. Список, базирующийся на таблице или сохраненном запросе, будет работать быстрее, чем список, у которого источник строк задан в виде инструкции SQL.

- Свойство **Число столбцов** (Column Count) имеет значение `2`, т. е. в списке будет два столбца (выбираются два поля), однако свойство **Ширина столбцов** (Column Width) определяет ширину первого столбца, равной `0`, т. е. он не будет отображаться на экране.
- Зато именно первый столбец является присоединенным к полю таблицы, т. е. выбранное значение `КодТипа` (CategoryID) будет записываться в базовую таблицу формы. Номер присоединенного столбца указывается в свойстве **Присоединенный столбец** (Bound Column).
- Свойство **Заглавия столбцов** (Column Heads) в нашем примере имеет значение **Нет** (No), поэтому заголовки отображаться в списке не будут.

Кроме рассмотренного, существуют еще три типа источников строк для элемента управления **Список** (ListBox):

- *Список значений.* Если количество строк в списке не велико и постоянно, то можно все значения задать прямо в окне свойств элемента. В этом случае в свойстве **Тип источника строк** (Row Source Type) выбирается значение **Список значений** (Value List), а значения элементов списка перечисляются через точку с запятой без пробелов в свойстве **Источник строк** (Row Source). Например:

  ```
  понедельник;вторник;среда;четверг;пятница;суббота;воскресенье
  ```

  Если элементы списка содержат пробелы, включите их в кавычки. Чтобы все дни недели отобразились в одном столбце списка, установите значение свойства **Число столбцов** (Column Count) равным `1`, а свойства **Ширина столбцов** (Column Width) равным ненулевому значению, например `2,5см` (2,5cm).

  Если нужно вывести два столбца значений, то в свойстве **Источник строк** (Row Source) в перечислении значений задаются последовательно пары значений, число столбцов устанавливается равным `2`, а в свойстве **Ширина столбцов** (Column Width) задается пара значений, например: `2,5см;3см` (2,5cm;3cm).

- *Список полей.* Этот тип источника строк (значение **Список полей** (Filed List) свойства **Тип источника строк** (Row Source Type)) предназначен для вывода в списке имен полей таблицы или запроса. Имя таблицы или запроса задается в свойстве **Источник строк** (Row Source). Таким образом в списке можно отобразить имена полей таблиц, чтобы предоставить возможность их выбора.
- *Определяемая пользователем функция.* Наконец, источником строк для списка может быть функция, написанная на языке VBA, которая заполнит список значениями. В этом случае свойство **Тип источника строк** (Row Source Type) нужно установить, как это показано на рис. 9.16, т. е. указать имя функции без знака равенства (=) и без скобок, а свойство **Источник строк** (Row Source) должно остаться пустым. Этот вариант полезен, когда требуется заполнить список вычисленными значениями, возможно даже на основе данных, которых нет в базе (например, когда нужно вычислить определенные даты, начиная с текущей), или когда содержимое списка должно динамически меняться в зависимости от содержимого других элементов управления (например, если для выбранной категории товаров в одном списке, в другом должен отображаться набор товаров именно этой категории).

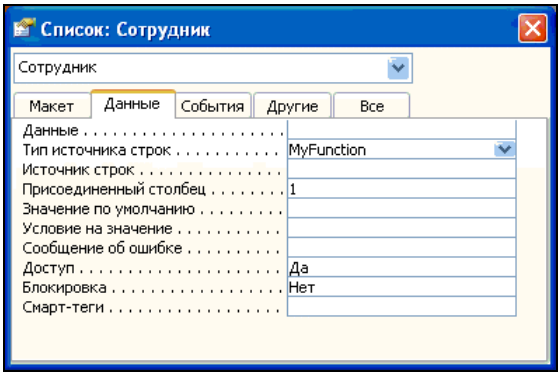


Рис. 9.16. Использование функции в качестве источника строк

Несколько слов о третьем типе источника строк списка. Настоящая функция должна иметь такой список аргументов:

```
Function Имя_функции(fld As Control, id As Variant, _
     row As Variant, col As Variant, code As Variant) As Variant
```

Данная функция вызывается Access каждый раз, когда требуется заполнение списка элементами. Например, это происходит автоматически, когда форма, содержащая список, открывается в первый раз. Вызвать обновление содержимого списка в какой-то другой момент можно программно, с помощью метода обновления содержимого элемента управления `Require`. Каждый раз при обновлении содержимого списка эта функция вызывается неоднократно: с различным "кодом действия". Поясним значения ее аргументов и возвращаемого значения.

- `code` — "код действия" определяет, какое действие должна выполнить функция при данном вызове. Это могут быть следующие действия: инициализация списка, его открытие, получение количества строк, количества столбцов, ширины столбцов, самих элементов списка и завершение работы. Всем этим действиям соответствуют зарезервированные константы VBA.
- `fld` — ссылка на элемент управления **Список** или **Поле со списком**, который заполняется элементами, с помощью данной функции.
- `id` — уникальный идентификатор заполняемого элемента управления. Этот аргумент полезен, когда такая функция используется для заполнения нескольких элементов управления, чтобы их различить.
- `row` — текущий номер строки для заполняемого элемента списка (начиная с 0).
- `col` — текущий номер столбца для заполняемого элемента списка (начиная с 0).
- Возвращаемое значение и его тип зависит от текущего "кода действия", переданного в функцию. Например, для кода `acLBGetValue` (получить данные) это должно быть строковое значение элемента списка в заданной строке `row` и столбце `col`.

Все значения аргументов формируются и передаются в функцию самим Access. Разработчику остается их обработать.

Совет

О создании функции, выполняющей роль источника данных для списка или поля со списком, можно получить подробную информацию в справочной системе Access 2003, в разделе: "Microsoft Access Visual Basic Reference (Справочник по Visual Basic для Microsoft Access), Properties (Свойства), R, RowSourceType Property (Свойство RowSourceType)".

Свойство **Присоединенный столбец** (Bound Column), который, как уже было сказано, определяет тот столбец списка, значение которого при выборе становится значением элемента управления, может иметь значение 0. В этом случае значением элемента **Список** (List Box) станет индекс (номер) выбранного элемента в списке.

Элемент управления **Список** (List Box) интересен еще тем, что он позволяет делать множественный выбор в списке, т. е. в нем можно выбрать одновременно несколько элементов, чего не позволяет делать элемент **Поле со списком** (ComboBox). Такая возможность требуется довольно часто, когда задаются критерии отбора записей. Например, в базе данных объектов недвижимости надо найти интересующие клиента квартиры. А его интересуют квартиры в определенных районах города. Можно, конечно, выбирать в списке по очереди сначала один район, чтобы выбрать все квартиры в нем, затем другой район и т. д. Однако можно использовать для указания интересующих районов список, в котором выбираются сразу все интересующие районы, а потом формируется соответствующий запрос к базе данных.

Определить, как пользователь будет выбирать значения из списка, можно с помощью свойства **Несвязное выделение** (Multi Select). Его можно найти на вкладке **Другие** (Other) окна свойств. По умолчанию это свойство имеет значение **Отсутствует** (None). В этом случае разрешается выбор только одного элемента из списка. Но это свойство может иметь значения **Простой** (Simple) и **Со связным выбором** (Extended). В первом случае разрешается множественный выбор просто щелчком мыши по нужным элементам, а во втором список работает так, как почти все списки в Windows, т. е. можно использовать клавиши <Shift> и <Ctrl> для выделения смежных и несмежных элементов. Результаты выбора хранятся в специальном массиве и могут быть использованы в программах на VBA.

Элемент управления *Поле со списком*

Элемент управления **Поле со списком** (ComboBox) во многом аналогичен элементу управления **Список** (ListBox). Применяется он в формах для той же самой цели, т. е. когда требуется, чтобы значение поля в таблице не вводилось вручную, а выбиралось из известного списка значений. Как правило, этими списками являются различного рода справочники. Отличий между данными типами элементов управления три:

1. **Поле со списком** (ComboBox) позволяет не только выбирать значение из списка, но и вводить его прямо в поле ввода.
2. **Поле со списком** (ComboBox) не накладывает таких ограничений на длину списка, как **Список** (ListBox), т. к. в нем отображается только текущее значение, а остальные значения выводятся, когда пользователь щелкает мышью по стрелке вниз с правой стороны поля. Соответственно, элемент **Поле со списком** (ComboBox) занимает на форме меньше места, чем **Список** (ListBox).
3. Поле со списком позволяет выбрать только один элемент из списка.

Основные свойства для элемента управления **Поле со списком** (ComboBox) такие же, как и у элемента управления **Список** (ListBox), однако есть дополнительные свойства, которые определяют формат вывода данных в текстовую часть поля: **Формат поля** (Format), **Число десятичных знаков** (Decimal Places), **Маска ввода** (Input Mask) (рис. 9.17).

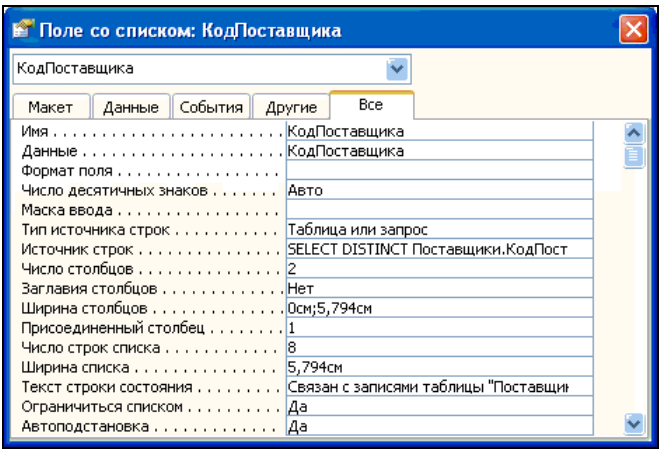


Рис. 9.17. Свойства элемента **Поле со списком**

Еще два дополнительных свойства определяют особенности работы поля со списком:

1. **Ограничиться списком** (Limit To List). Это свойство может иметь значение **Да** (Yes), и тогда значение, которое вводится в поле, должно обязательно совпадать с одним из элементов списка, или **Нет** (No), и тогда список может дополняться значениями, вводимыми в текстовую часть поля со списком. Если значение этого поля **Да** (Yes), а вводимое значение не совпадает ни с одним из элементов списка, Access выдает сообщение об ошибке. Все ситуации, возникающие при вводе нового значения в поле, могут быть обработаны с помощью макроса или процедуры VBA, например, чтобы расширить список значений. *(См. также разд. "Применение макросов", гл. 12.)*
2. **Автоподстановка** (Auto Expand). Это свойство обычно имеет значение **Да** (Yes), что обеспечивает удобный выбор значения из списка. Так как списки в данном случае могут быть длинными, например список клиентов, прокручивать их в поисках нужного значения неудобно. Однако можно просто вводить это значение в текстовую часть поля. При этом Access автоматически подставит в поле то значение из списка, у которого первые буквы совпадают с введенными.

Обычно элемент управления **Поле со списком** (ComboBox) используется для тех полей базовой таблицы, которые служат внешними ключами, т. е. содержат значения, соответствующие первичным ключам записей в какой-то другой таблице. В таблице такие поля обычно определяют как поля подстановки, т. е. при отображении этих полей отображается не то значение, которое содержится в данном поле, а соответствующее значение из связанной таблицы (ведь ключом часто является ничем не примечательный код). *(См. также разд. "Создание таблиц", гл. 2.)*

В этом случае Мастер форм при создании формы автоматически создаст для таких полей поля со списком и задаст их свойства.

Совет

Если вы в режиме Конструктора хотите создать элемент управления **Поле со списком** (ComboBox) для поля таблицы, являющегося полем подстановки, то лучше делать это путем перетаскивания поля из окна **Список полей** (Field List), т. к. в этом случае все свойства элемента будут заданы автоматически.

Элемент управления *Кнопка*

Элемент управления **Кнопка** (Command Button) является очень важным элементом формы, поскольку именно с кнопками связаны различные действия, выполняемые пользователем в приложении (сохранение введенных данных, вызов другой формы, вывод на печать документа и т. д.). Обычно кнопка выбирается, когда пользователь щелкает по ней мышью. Однако можно выбрать (нажать, т. к. визуально это именно так и выглядит) кнопку, используя только клавиатуру. Для этого нужно поместить на нее фокус, т. е. сделать активной, нажимая клавишу <Tab>, а затем нажать клавишу <Пробел> или <Enter>. Для этой же цели можно назначить кнопке специальную комбинацию клавиш.

Для создания кнопки очень удобно использовать Мастер кнопок, поскольку он выполнит за вас большую работу и даже напишет программу, которая будет выполнять действия, для которых эта кнопка предназначена. Для примера создадим кнопку, которая будет открывать новую форму:

1. Создайте в режиме Конструктора пустую форму.
2. Выберите на Панели элементов (Toolbox) элемент управления **Кнопка** (Command Button) (кнопка **Мастера** (Control Wizards) на Панели элементов (Toolbox) должна быть нажата).
3. В появившемся диалоговом окне вы увидите два списка: левый список содержит категории действий, а правый сами действия. Выберите в списке **Категории** (Categories) значение **Работа с формой** (Form Operations) (рис. 9.18).
4. В правом списке при этом появится список действий, относящихся к этой категории. Выберите значение **Открыть форму** (Open Form) и нажмите кнопку **Далее** (Next).
5. В следующем окне требуется выбрать форму, которую нужно открывать с помощью кнопки. Выберите одну из форм, например, `Заказы` (Orders) (рис. 9.19).
6. В третьем диалоговом окне нужно выбрать, все ли записи будут отображаться в открываемой форме или нужно отобрать записи, связанные с записями в данной форме (рис. 9.20). Поскольку наша форма не содержит пока ни одного поля, то мы не сможем задать критерии отбора, поэтому выберите второй переключатель **Открыть форму и показать все записи** (Open the form and show all the records) и нажмите кнопку **Далее** (Next).
7. В следующем окне предлагается выбрать рисунок, который вы хотите поместить на кнопку или задать надпись (рис. 9.21). Чтобы увидеть список всех готовых рисунков, нужно установить флажок **Показать все рисунки** (Show All Pictures).

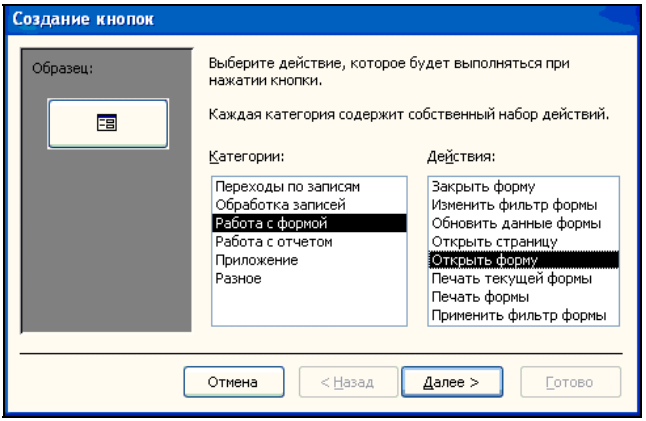


Рис. 9.18. Первое диалоговое окно Мастера кнопок

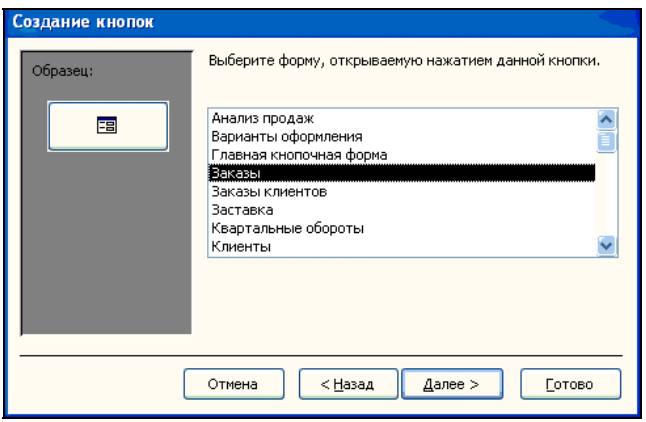


Рис. 9.19. Второе диалоговое окно Мастера кнопок

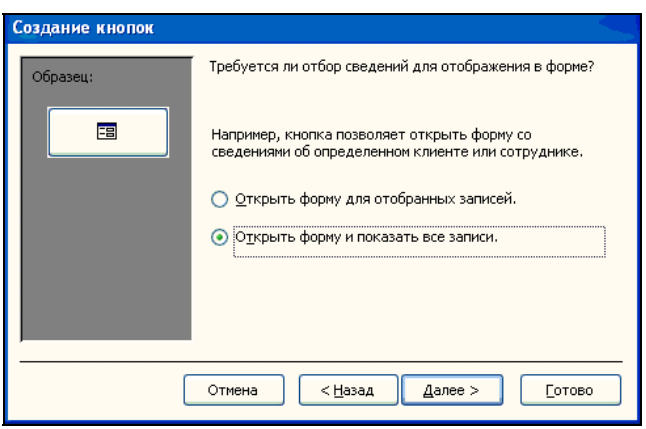


Рис. 9.20. Третье диалоговое окно Мастера кнопок

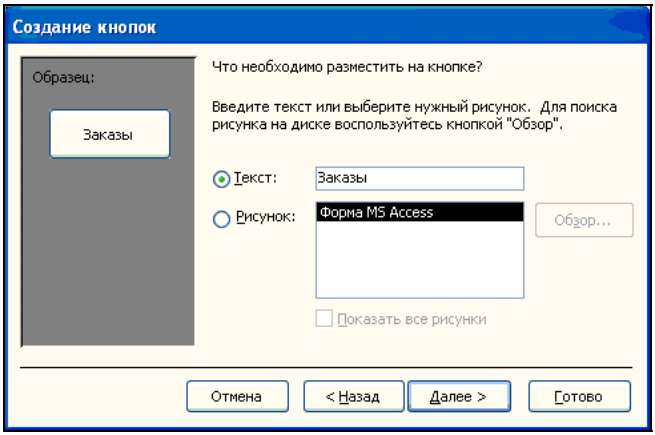


Рис. 9.21. Четвертое диалоговое окно Мастера кнопок

Чтобы отобразить надпись на кнопке, выберите переключатель **Текст** (Text) и введите текст надписи, например, совпадающий с именем формы: `Заказы`. Нажмите кнопку **Далее** (Next).

8. В последнем окне Мастера кнопок требуется ввести имя кнопки. Оно может не совпадать с надписью на кнопке, но должно быть коротким и понятным. В данном случае можно оставить тот же текст, что и в надписи: `Заказы` (рис. 9.22). Нажмите кнопку **Готово** (Finish).

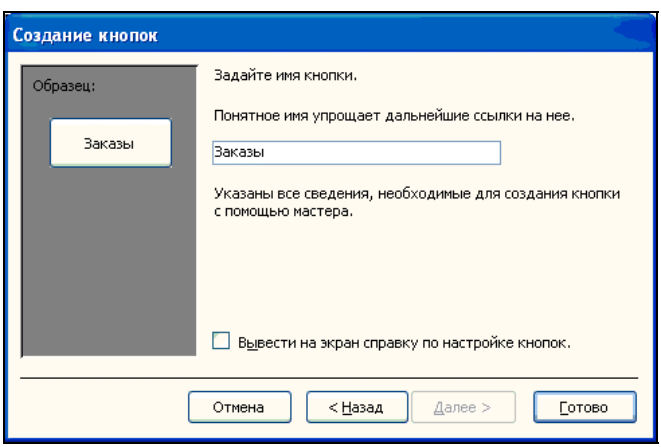


Рис. 9.22. Пятое диалоговое окно Мастера кнопок

На форме появилась кнопка. Если вы потом решите все-таки поместить на кнопку какую-нибудь картинку, это можно сделать, установив свойство **Рисунок** (Picture). Для этого можно использовать Построитель, который предлагает либо выбрать одну из стандартных картинок, либо использовать графический файл, например, созданный в графическом редакторе Paint в Windows (рис. 9.23).

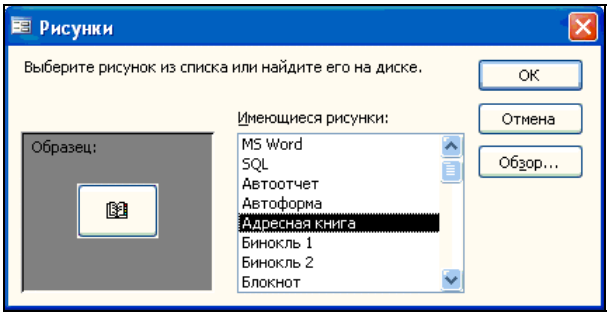


Рис. 9.23. Размещение рисунка на кнопке

Можно назначить элементу управления **Кнопка** (Command Button) комбинацию клавиш. Это бывает оправдано, например, если пользователь работает с клавиатурой, и ему неудобно для нажатия кнопки переключаться на мышь, а также если в форме много элементов и для активизации кнопки приходится много раз нажимать клавишу <Tab>. Чтобы назначить кнопке комбинацию клавиш, в свойстве **Подпись** (Caption), которое определяет надпись на кнопке, выбирают одну букву (она не обязательно должна быть первой) и добавляют перед ней знак амперсанд (&) (рис. 9.24). Выбранная буква в надписи на кнопке будет подчеркнута. Тогда для того, чтобы выбрать кнопку, достаточно просто нажать комбинацию клавиш <Alt>+<*выбранная буква*>, в нашем примере — это <Alt>+<з>.

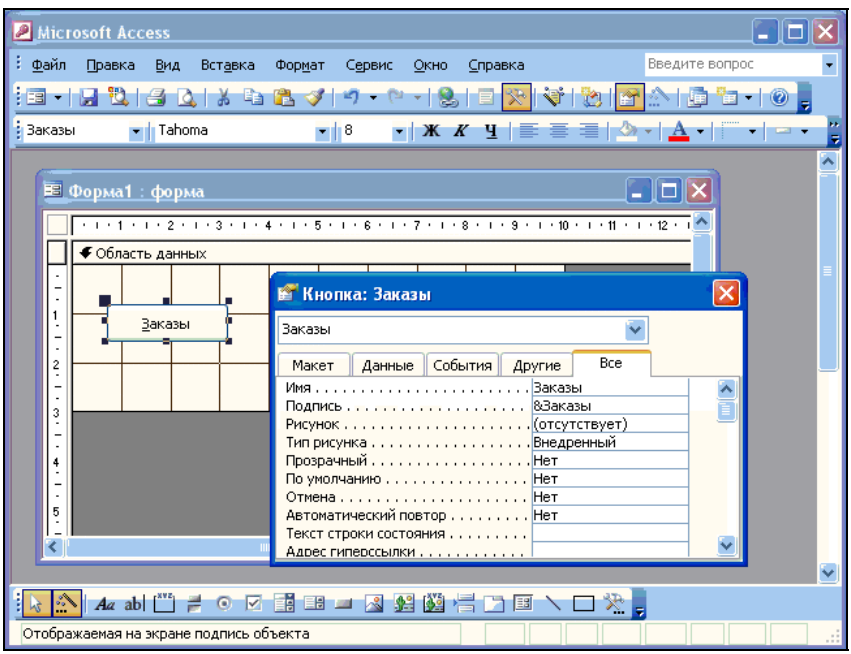


Рис. 9.24. Назначение кнопке комбинации клавиш

Для любой формы можно создать две специальные кнопки.

- Одна будет "нажиматься" всякий раз, когда пользователь нажмет клавишу <Enter> и независимо от того, на каком элементе формы установлен фокус. Такая кнопка определяет *действие по умолчанию* и для нее свойству **По умолчанию** (Default) присваивают значение **Да** (Yes).
- Вторая кнопка "нажимается" всякий раз, когда пользователь нажмет клавишу <Esc> и опять независимо от того, на каком элементе формы установлен фокус. Такая кнопка играет роль *отмены по умолчанию*. У нее должно быть установлено значение **Да** (Yes) для свойства **Отмена** (Cancel).

Элементы управления *Свободная рамка объекта* и *Присоединенная рамка объекта*

Элементы управления **Свободная рамка объекта** (Unbound Object Frame) и **Присоединенная рамка объекта** (Bound Object Frame) используются для того, чтобы вставить в форму объекты, созданные в других приложениях, например документ Word, таблицу Excel или рисунок. Они различаются тем, будут ли объекты, вставленные в эти рамки, храниться в таблицах Access, или нет.

Элемент управления **Свободная рамка объекта** (Unbound Object Frame) содержит объект, который хранится либо в самой форме, либо во внешнем файле, и независимо от того, какая запись отображается в форме, объект будет отображаться один и тот же. Этот объект может быть изменен прямо из формы с помощью того приложения, в котором он был создан.

Элемент управления **Присоединенная рамка объекта** (Bound Object Frame) содержит объект, который хранится в таблице, в поле, которое имеет тип **Поле объекта OLE** (OLE Object). Поэтому присоединенная рамка объекта имеет в качестве источника данных поле в базовой таблице. Типичный пример такого элемента управления — фотографии людей, изделий и т. д.

Подробно создание и использование этих элементов управления рассмотрено в *гл. 15*.

Элемент управления *Рисунок*

Элемент управления **Рисунок** (Image) очень похож по своему назначению и свойствам на элемент **Свободная рамка объекта** (Unbound Object Frame). Он предназначен для вставки в форму графических файлов, но требует меньше системных ресурсов и работает быстрее, чем элемент управления **Свободная рамка объекта** (Unbound Object Frame). Форматы графических файлов, которые поддерживаются Access, разнообразны — это растровые рисунки, метафайлы, значки и др.

Подробно создание и использование этого элемента управления рассмотрено в *гл 15*.

Элемент управления *Разрыв страницы*

Элемент управления **Разрыв страницы** (Page Break) используется в многостраничных формах для того, чтобы определить место конца одной и начала новой страницы. На форме этот элемент отображается в виде пяти горизонтально расположенных точек и имеет всего четыре свойства: **Имя** (Name), координаты относительно верх-

него левого угла формы — **От левого края** (Left), **От верхнего края** (Top) и **Дополнительные сведения** (Tag) (рис. 9.25).

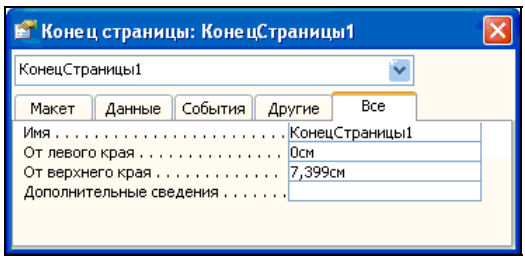


Рис. 9.25. Свойства элемента **Разрыв страницы**

Для переключения между страницами в такой форме используются клавиши <PgUp> и <PgDn>.

(См. также разд."Создание многостраничных форм" в этой главе).

Элемент управления *Набор вкладок*

Элемент управления **Набор вкладок** (Tab Control) также используется для создания многостраничных форм. Он позволяет придать формам такой же вид, как у диалоговых окон свойств в Windows. В этом элементе может быть от одной до нескольких вкладок. Переключение между вкладками формы выполняется щелчком левой кнопкой мыши по ярлыку вкладки. Вкладки могут иметь два варианта оформления: ярлыки и кнопки (рис. 9.26).

В таких формах Access на ярлыках могут размещаться не только надписи, но и рисунки, а также надписи и рисунки вместе. Если требуется большое количество вкладок, можно отобразить их в несколько рядов. О создании форм, содержащих несколько вкладок, рассказано далее *в этой главе, в разд. "Создание многостраничных форм"*.

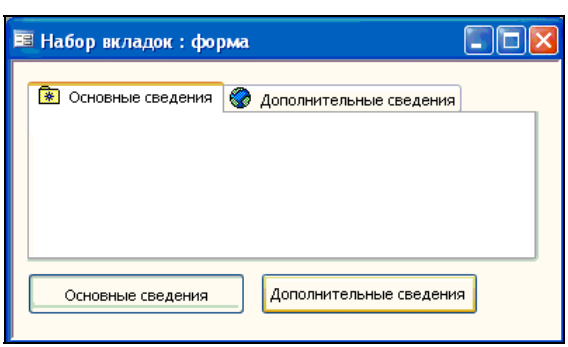


Рис. 9.26. Элемент управления **Набор вкладок**

Элемент управления *Подчиненная форма/отчет*

Элемент управления **Подчиненная форма/отчет** (Subform/Subreport) используется для отображения в форме другой формы. *(См. также разд. "Создание многотабличных форм" в этой главе.)*

Элементы управления *Линия* и *Прямоугольник*

Элементы управления **Линия** (Line) и **Прямоугольник** (Rectangle) используются для оформления внешнего вида форм, чаще всего для выделения группы логически связанных элементов управления.

Для управления цветом и стилем линии, а также окантовки прямоугольника используются свойства **Цвет границы** (Border Color), **Тип границы** (Border Style), **Ширина границы** (Border Width) и **Оформление** (Special Effect). Для прямоугольника можно также задать свойства **Тип фона** (Back Style) и **Цвет фона** (Back Color).

Существует восемь вариантов типа линии и границ прямоугольника: **Отсутствует**, **Сплошная**, **Штриховая**, **Пунктирная**, **Точечная**, **Редкоточечная**, **Штрихпунктирная** и **Штрихточечная** (Transparent, Solid, Dashes, Short Dashes, Dots, Sparse Dots, Dash Dot, Dash Dot Dot). Для линии и границ прямоугольника определено шесть вариантов оформления: **обычное**, **приподнятое**, **утопленное**, **вдавленное**, **с тенью**, **рельефное** (Flat, Raised, Sunken, Etched, Shadowed, Chiseled).

Цвет границы и цвет фона задаются с помощью Построителя, который позволяет выбрать один из 64 предложенных цветов или определить собственный цвет.

На рис. 9.27 представлены различные варианты оформления прямоугольников и линий. Свойство **Тип фона** (Back Style) для прямоугольника имеет два значения:

- **прозрачный** — цвет фона прямоугольника совпадает с цветом фона раздела формы и все элементы, лежащие под ним, видны;
- **нормальный** — можно задать цвет фона, но тогда элементы управления, которые выделяет этот прямоугольник, должны быть размещены поверх него (т. е. на переднем плане — команда **Формат | На передний план** (Format | Bring to Front)).



Рис. 9.27. Различные варианты оформления прямоугольника и линии

Преобразование одного элемента управления в другой

Иногда в процессе разработки формы требуется изменить тип элемента управления для отображения какого-либо поля. Такая ситуация может возникнуть, например, из-за ошибки в начальном выборе типа элемента. Чаще всего требуется преобразовать текстовое поле в поле со списком или изменить тип элемента выбора, заменив флажок переключателем или выключатель переключателем. Конструктор форм позволяет легко это сделать:

1. Выделите элемент управления, тип которого нужно изменить, на форме, щелкнув по нему правой кнопкой мыши.
2. В появившемся контекстном меню выберите команду **Преобразовать элемент в** (Change To).
3. Справа раскроется меню, в котором будут доступны только те типы элементов, в которые может быть преобразован выбранный элемент управления (рис. 9.28). Например, для текстового поля это может быть поле со списком, список или надпись.

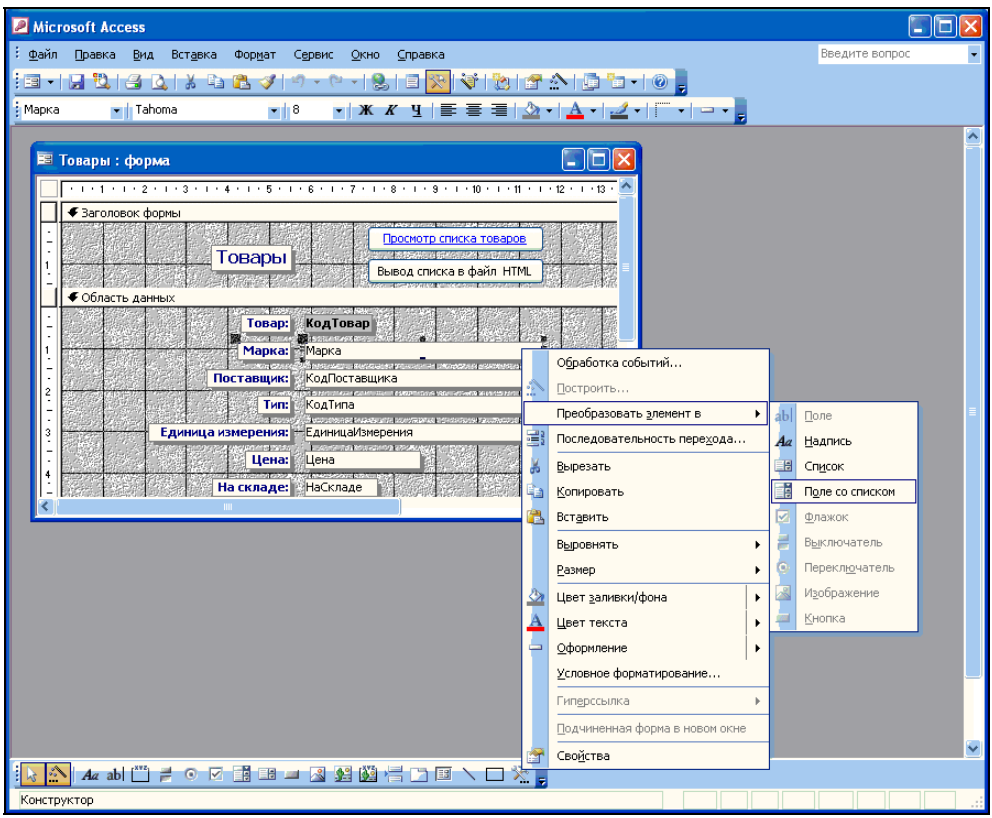


Рис. 9.28. Преобразование одного элемента управления в другой

4. Выберите в меню нужный тип элемента управления. Элемент немедленно будет преобразован.

Замечание

Вместо контекстного меню для выполнения операции можно использовать команду меню **Формат** | **Преобразовать элемент в** (Format | Change To).

Создание многотабличных форм

К многотабличным формам относят формы, построенные на основе запроса, объединяющего несколько таблиц, а также формы с внедренными в них другими формами. Создание многотабличных форм на основе запроса рассматривалось ранее. *(См. гл. 5 "Создание форм для ввода данных".)*

В данном разделе мы остановимся на создании сложных форм, включающих подчиненные формы.

Подчиненной формой называется форма, которая встраивается в другую форму. При этом форма, которая включает подчиненную форму, называется *основной* (или главной) *формой*.

Обычно такие формы применяются для отображения данных из связанных таблиц, например, категории товаров и товары, организации и контактные лица в этих организациях и т. д. Однако это бывает не всегда. Иногда главная форма содержит свободные элементы управления и различные кнопки, которые влияют на представление данных в подчиненной форме. Типичным примером такой формы является форма для отбора записей в таблице. Основная форма содержит поля, в которых задаются критерии отбора, и кнопки **Фильтр** или **Найти** и **Отмена фильтра** или **Показать все** (рис. 9.29).

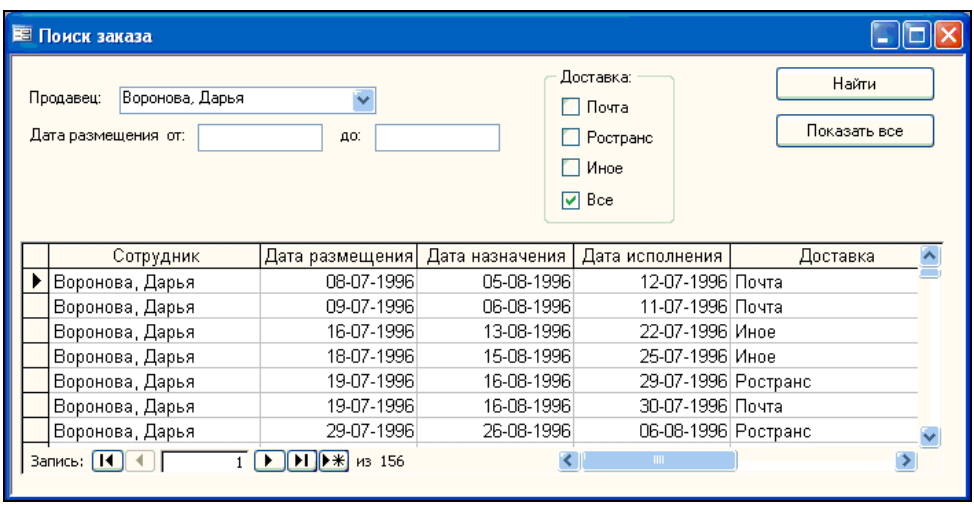


Рис. 9.29. Пример формы, содержащей подчиненную форму

Применение встроенных подчиненных форм обеспечивает более компактное представление на экране данных из нескольких таблиц, чем использование разных форм для каждой таблицы.

Создавать такие формы можно разными способами. Рассмотрим сначала способы создания главной и подчиненной форм для двух связанных таблиц.

Создание главной и подчиненной форм с помощью Мастера автоформ

В Access, начиная с версии Access 2000, существует возможность автоматического отслеживания связей между таблицами. Поэтому простые формы для связанных таблиц можно построить с помощью Мастера автоформ. Он сразу найдет все таблицы, которые связаны с главной отношением "один-ко-многим", и создаст для каждой такой таблицы подчиненную форму. В качестве примера давайте создадим новую форму `Клиенты`, в которой должны быть отражены не только данные о клиенте, но и о его заказах.

Сначала откройте окно **Схема данных** (Relationship) и посмотрите, как связаны интересующие нас таблицы (рис. 9.30). Таблица `Клиенты` (Customers) связана с таблицей `Заказы` (Orders) отношением "один-ко-многим". В свою очередь таблица `Заказы` (Orders) связана с таблицей `Заказано` (Order Details) тоже отношением "один-ко-многим".

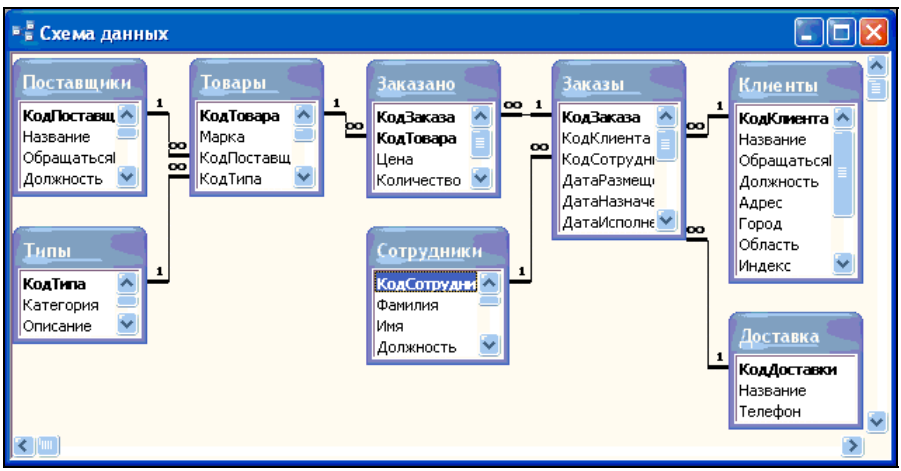


Рис. 9.30. Диалоговое окно **Схема данных** приложения `Борей`

Теперь в окне базы данных на странице **Таблицы** (Table) выделите таблицу `Клиенты` (Customers) и нажмите стрелку вниз на кнопке **Новый объект** (New Object) на панели инструментов. В открывшемся списке выберите объект **Автоформа** (AutoForm).

Подождите, пока Мастер автоформ закончит свою работу. В результате вы должны увидеть на экране форму, которая представлена на рис. 9.31.

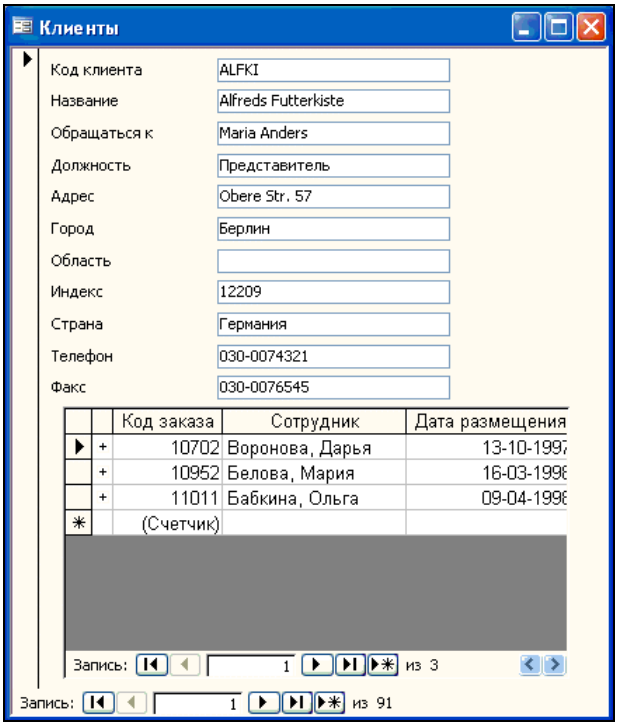


Рис. 9.31. Составная форма, созданная Мастером автоформ

Чтобы открыть вторую подчиненную форму (список подчиненных записей следующего уровня), просто щелкните мышью по квадратику с крестом слева от выбранной записи в подчиненной форме. В данной примере в главной форме отображается одна запись о клиенте, а в подчиненной форме — только те записи из таблицы `Заказы` (Orders), которые связаны с текущей записью в главной форме, т. е. только заказы текущего клиента. В списке подчиненных записей для выбранного заказа отображается список заказанных продуктов.

Уже из этой формы видно, что Access обеспечивает вложенность подчиненных форм, причем максимальное количество уровней вложенности достаточно велико: 7. С другой стороны, если таблица связана отношениями "один-ко-многим" с несколькими таблицами, то в одну главную форму можно встроить любое количество подчиненных форм.

Такое иерархическое представление связанных данных удобно не только для отображения, но и для ввода данных.

Например, нужно оформить заказ клиенту, которого еще нет в базе данных. Значит, требуется добавить запись о новом клиенте в таблицу. Для этого перейдите в режим ввода записи в главной форме, щелкнув мышью по кнопке **Новая запись** (New Record) внизу формы или на панели инструментов. Введите данные о клиенте, а затем переместите фокус в подчиненную форму для оформления заказа. Как только вы переместили фокус в подчиненную форму, запись о клиенте из главной формы

сохранится. Теперь подчиненная запись о заказе будет корректно добавлена в таблицу Заказы (Orders), гарантируя сохранение ссылочной целостности.

Аналогично будет сохранена запись о заказе, прежде чем начнут вводиться данные о заказанных продуктах во вложенной подчиненной форме. Записи в подчиненной форме сохраняются при переходе к каждой следующей строке.

Использование Мастера форм для создания главной и подчиненной форм

Составную форму с подчиненными формами можно создать также с помощью Мастера форм. Это почти так же просто, как использовать средство создания автоформ, но при этом есть возможность поместить в формы не все поля, а только те, что требуется.

Рассмотрим создание такой формы для таблиц Сотрудники (Employees) и Заказы (Orders). Снова посмотрим на схему данных (см. рис. 9.30) и убедимся, что таблица Сотрудники (Employees) связана с таблицей Заказы (Orders) отношением "один-ко-многим".

1. Нажмите стрелку справа от кнопки **Новый объект** (New Object) на панели инструментов и выберите в списке команду **Форма** (Form), или нажмите кнопку **Создать** (New) на странице **Формы** (Forms) окна **База данных** (Database).
2. В диалоговом окне **Новая форма** (New Form) выделите элемент списка **Мастер форм** (Form Wizard), выберите в поле со списком таблицу для главной формы — Сотрудники (Employees) — и нажмите кнопку **ОК**. *(См. также разд. "Создание формы с помощью Мастера", гл. 5.)*

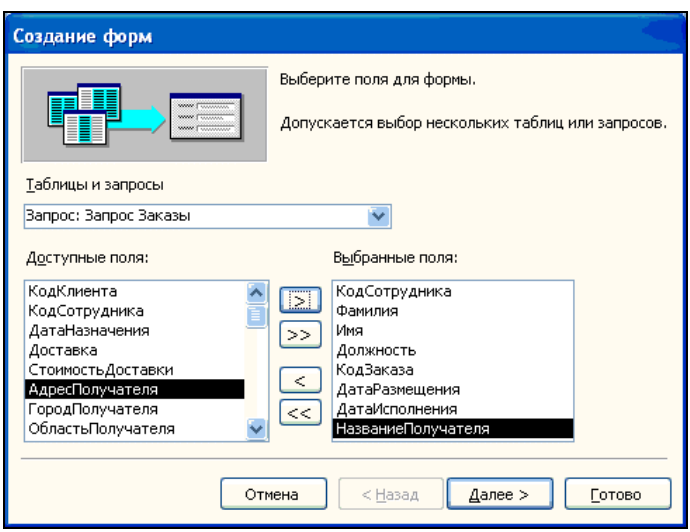


Рис. 9.32. Первое диалоговое окно Мастера форм

3. В первом диалоговом окне Мастера форм (рис. 9.32) в списке **Доступные поля** (Available Fields) выберите те поля, которые вы хотите включить в главную фор-

му. Перенесите их в список **Выбранные поля** (Selected Fields) с помощью кнопки >. Затем в этом же окне в поле со списком **Таблицы и запросы** (Tables/Queries) выберите запрос `Запрос Заказы` (Orders Qry), который содержит данные из двух таблиц: `Заказы` (Orders) и `Клиенты` (Customers). В списке **Доступные поля** (Available Fields) из этого запроса выберите поля, которые нужно включить в подчиненную форму, и так же добавьте их в список выбранных полей. Нажмите кнопку **Далее** (Next).

4. В следующем окне Мастера (рис. 9.33) предлагается выбрать, какая из двух таблиц будет главной (по умолчанию устанавливается первая таблица, поля из которой мы выбирали). Переключатель, определяющий вид представления иерархии форм, по умолчанию установлен в значение **Подчиненные формы** (Form with subform(s)). Это то, что нам нужно. Нажимаем кнопку **Далее** (Next).

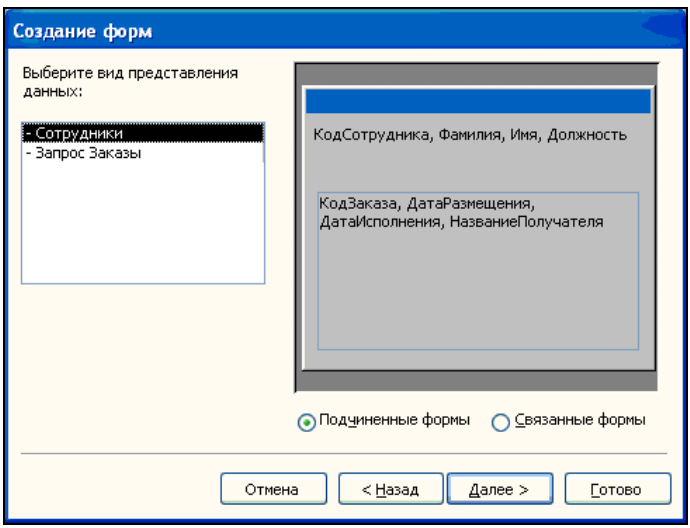


Рис. 9.33. Второе диалоговое окно Мастера форм

5. В остальных диалоговых окнах нужно выбрать вид подчиненной формы (ленточный или табличный), стиль и задать имена для главной и подчиненной форм. После нажатия кнопки **Готово** (Finish) создается форма, содержащая одну подчиненную форму, которая выглядит примерно так, как показано на рис. 9.34.

Теперь можно открыть эту форму в режиме Конструктора и настроить ее нужным образом.

Добавление подчиненной формы в главную форму

Подчиненная форма является элементом управления и, как и все остальные элементы, может быть добавлена в главную форму с помощью Мастера подчиненных форм. При этом ее можно создать заранее, и тогда мастеру нужно только указать имя добавляемой формы. Если подчиненная форма еще не создана, Мастер поможет ее создать.

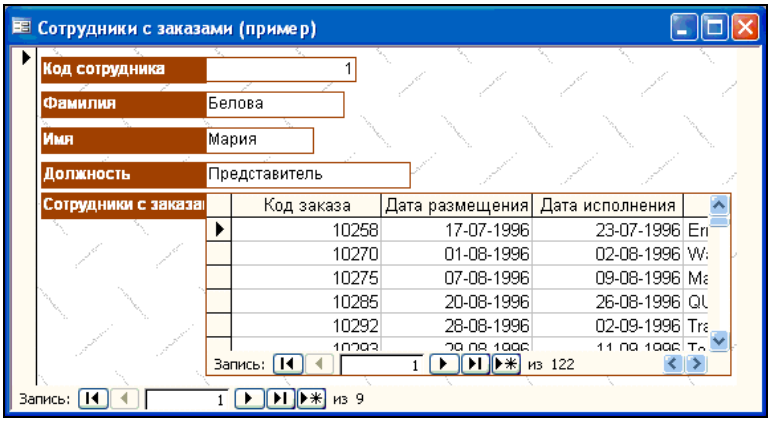


Рис. 9.34. Составная форма, созданная Мастером форм

В качестве примера добавим в форму `Поставщики` (Suppliers) подчиненную форму, которая будет содержать список товаров текущего поставщика (в исходной версии базы данных `Борей` товары поставщика отображаются в отдельной связанной форме при нажатии кнопки **Просмотр товаров** (Review Products)):

1. Скопируйте существующую форму `Поставщики` (Suppliers) (выделите ее в окне базы данных, нажмите комбинацию клавиш <Ctrl>+<C>, а затем <Ctrl>+<V>). В качестве имени копии введите `Поставщики (пример)`.
2. Откройте форму `Поставщики (пример)` в режиме Конструктора и отобразите на экране Панель элементов, если ее не видно.
3. При нажатой кнопке **Мастера** (Control Wizards) перетащите с Панели элементов в область данных формы элемент управления **Подчиненная форма/отчет** (Subform/Subreport), поместив ее под всеми остальными элементами. Перед этим можно увеличить высоту области данных, но это не обязательно, т. к. она увеличится автоматически после вставки нового элемента.
4. В первом диалоговом окне Мастера выберите переключатель **Имеющиеся таблицы и запросы** (Use existing Tables and Queries) и нажмите кнопку **Далее** (Next).
5. Во втором диалоговом окне в раскрывающемся списке **Таблицы и запросы** (Tables/Queries) выберите таблицу `Товары` (Products). В списке **Доступные поля** (Available Fields) выберите поля, включите их в форму кнопкой > (рис. 9.35) и нажмите кнопку **Далее** (Next).
6. В третьем диалоговом окне нужно определить поля, по которым связываются главная и подчиненная формы (рис. 9.36). Можно выбрать их из списка или задать самостоятельно. Так как таблицы `Поставщики` (Suppliers) и `Товары` (Products) связаны, в списке отображена нужная связь. Выберите ее и нажмите кнопку **Далее** (Next).
7. В последнем диалоговом окне нужно задать имя подчиненной формы. По умолчанию предлагается имя `подчиненная форма Товары` (Products subform). Если не хотите использовать имя, заданное по умолчанию, введите свое, например: `Поставщики подчиненная (пример)`, и нажмите кнопку **Готово**.

Результат работы Мастера представлен на рис. 9.37.

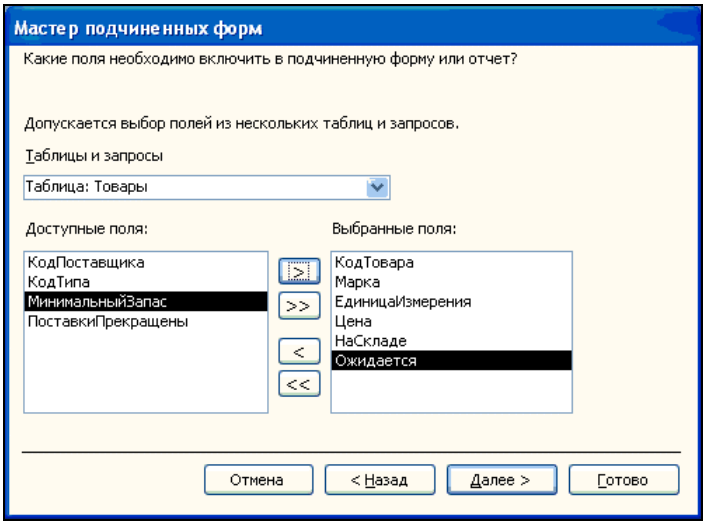


Рис. 9.35. Второе диалоговое окно Мастера подчиненных форм

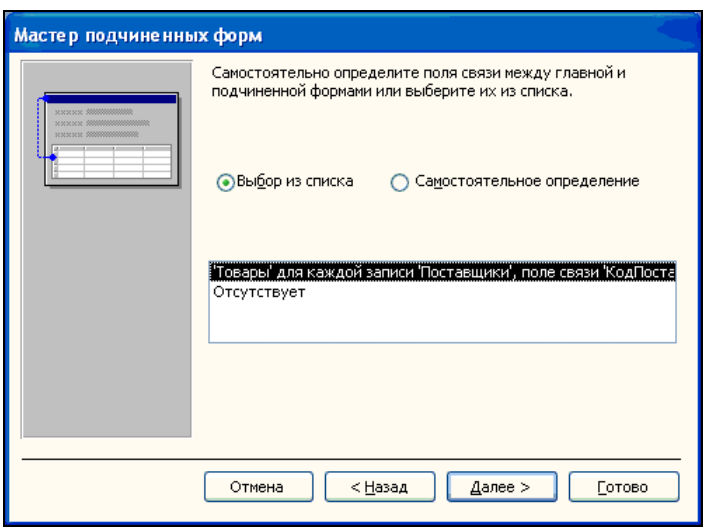


Рис. 9.36. Третье диалоговое окно Мастера подчиненных форм

Мастер подчиненных форм удобен тогда, когда подчиненная форма еще не создана. Если подчиненная форма уже существует, то можно просто разместить рядом окно базы данных с раскрытой страницей **Формы** (Forms) и главную форму, открытую в режиме Конструктора, и просто перетащить мышью нужную форму из окна базы данных в главную форму. При этом она становится подчиненной формой, и автоматически устанавливается связь между ними.

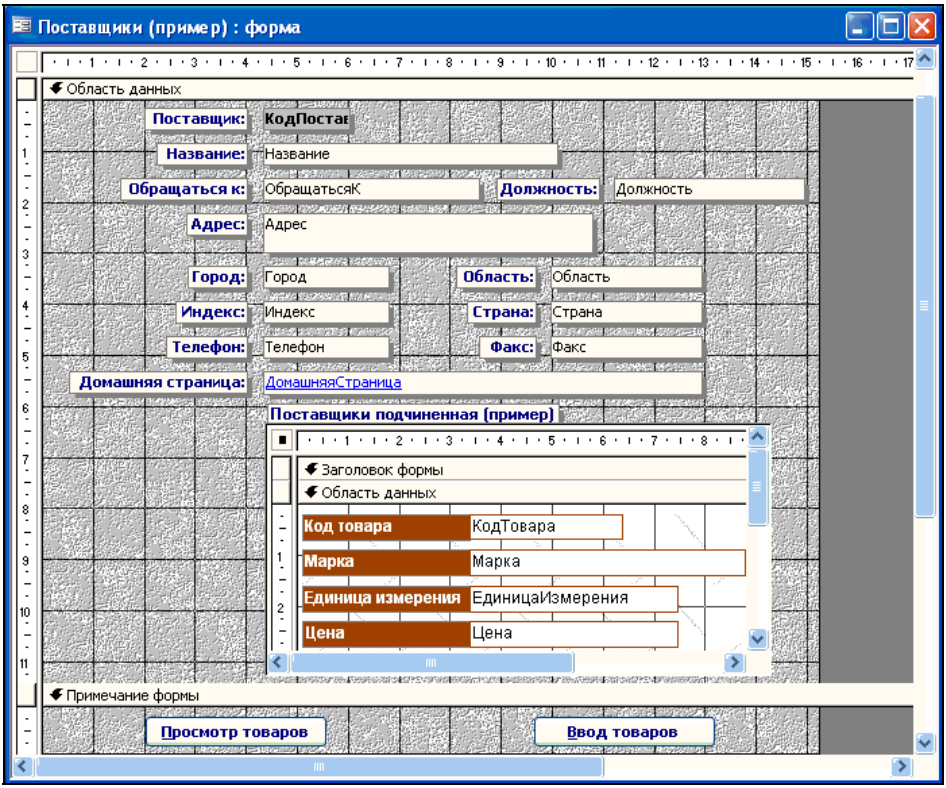


Рис. 9.37. Подчиненная форма `Товары` внутри формы `Поставщики`

Связывание главной и подчиненной форм

Связывание главной и подчиненной форм осуществляется с помощью свойств **Основные поля** (Link Master Fields) и **Подчиненные поля** (Link Child Fields) элемента управления **Подчиненная форма/отчет** (Subform/Subreport). Как мы уже сказали, Мастера форм автоматически заполняют эти свойства, основываясь на связях между таблицами. Если связи не заданы, то связующими полями считаются поля в таблицах, которые имеют одинаковые имена и совместимые типы данных. При этом связующее поле в базовой таблице для главной формы должно быть ключевым.

Если при разработке иерархической формы данные в главной и подчиненной формах окажутся не синхронизированными, следует проверить значение свойств **Основные поля** (Link Master Fields) и **Подчиненные поля** (Link Child Fields), а также ввести или откорректировать их значения:

1. Откройте главную форму в режиме Конструктора.
2. Выделите элемент управления **Подчиненная форма/отчет** (Subform/Subreport) (проще всего сделать это, описав с помощью мыши прямоугольник вокруг или пересекая его) и откройте окно **Свойства** (Properties) для этого элемента.

3. В ячейку свойства **Подчиненные поля** (Link Child Fields) следует ввести имя связующего поля из таблицы или запроса, на котором базируется подчиненная форма.
4. В ячейку свойства **Основные поля** (Link Master Fields) ввести имя связующего поля из таблицы или запроса, на котором базируется главная форма.
5. Если вы не помните имена полей в базовых таблицах или запросах, можно воспользоваться Построителем для связывания полей. Нажмите кнопку Построителя справа от любого из упомянутых свойств, и на экране появится диалоговое окно (рис. 9.38).

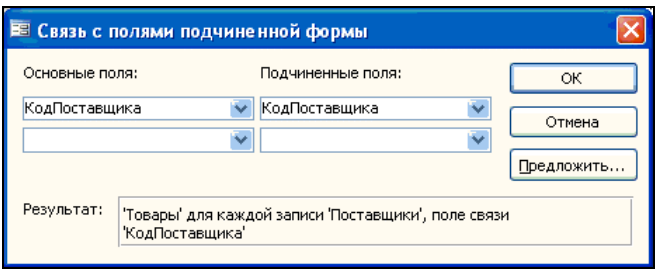


Рис. 9.38. Определение связей между главной и подчиненной формами

В левой части окна расположены раскрывающиеся списки для выбора связующих полей в главной форме, а справа — для выбора связующих полей в подчиненной форме. Связь может задаваться с помощью нескольких полей. Это может быть, если ключевое поле в главной таблице составное. Построитель позволяет задать максимум три поля и, как правило, этого хватает. В противном случае нужно просто вписать недостающие поля в значения свойств, разделив их точкой с запятой. Важно только, чтобы поля в каждом из данных свойств перечислялись в одинаковом порядке.

В предложенных списках содержатся все поля базовых таблиц или запросов. Это означает, что связующие поля не обязательно должны присутствовать в формах, они обязаны быть в источнике данных формы.

Синхронизация данных в элементах формы

В этом разделе приведены некоторые типичные примеры, иллюстрирующие отображение в форме данных в соответствии со значениями некоторых ее элементов. Вообще говоря, подобные приемы можно применить для синхронизаций данных как в простых, так и в составных формах, содержащих подчиненные формы. Примеры подобраны, исходя из наиболее часто встречающихся вопросов у наших читателей. Мы надеемся, что изучение этого раздела, а также применение комбинаций описанных приемов помогут читателям решить основные задачи при проектировании интерактивных форм.

Переход на определенную запись в форме

Типичной задачей является организация более удобного чем последовательный перебор перехода по записям в форме посредством задания значений определенных полей записи, для поиска по ним. Например, в форме с информацией для большого количества сотрудников хотелось бы быстро перейти к карточке сотрудника с известной фамилией. Это можно сделать с помощью специального списка для выбора фамилии.

В *гл. 5* мы упомянули о том, что при создании списка/поля со списком с помощью Мастера в форме, привязанной к источнику данных, Access предлагает сделать его средством перехода к нужной записи в форме. Рассмотрим пример создания такого списка:

1. Скопируйте форму `Сотрудники` (Employees) базы данных `Борей` (Northwind) и сохраните копию с именем `Сотрудники (пример)`.
2. Откройте созданную форму в режиме Конструктора.
3. Убедитесь, что на панели инструментов **Панели элементов** (Toolbox) нажата кнопка **Мастера** (Control Wizards). Нажмите кнопку **Поле со списком** (Combo Box) и поместите новый элемент в разделе заголовка формы, начертив мышью прямоугольник нужного размера.
4. Появится первое диалоговое окно Мастера списков. Выберите переключатель **Поиск записи в форме на основе значения, которое содержит поле со списком** (Find a record on my form based on the value I selected in my combo box) и нажмите кнопку **Далее** (Next).
5. В следующем диалоговом окне Мастера выберите поля, которые образуют столбцы элементов списка: `КодСотрудника` (EmployeeID), `Фамилия` (LastName) и `Имя` (FirstName). Нажмите кнопку **Далее** (Next).
6. В следующем диалоговом окне предлагается настроить ширину столбцов списка. Столбец с ключевым полем `КодСотрудника` (EmployeeID) рекомендуется сделать скрытым (установите флажок **Скрыть ключевой столбец** (Hide key column)). Оставьте предложенные параметры и нажмите кнопку **Далее** (Next).
7. В последнем диалоговом окне Мастера требуется ввести текст метки, поясняющей назначение списка. Введите текст `Фамилия:` и нажмите кнопку **Готово** (Finish).

Форма готова. Откройте ее в режиме Формы и убедитесь, что при выборе желаемых фамилии и имени в созданном поле со списком осуществляется переход к информации об этом сотруднике. Такое поле со списком удобно и тем, что значение в нем можно не только выбрать, но и ввести первые буквы фамилии, тогда необходимый элемент будет выбран автоматически. Останется нажать клавишу <Enter>, чтобы отобразить информацию о выбранном сотруднике.

Действие созданного поля со списком обеспечивается процедурой обработки события **После обновления** (After Update), сгенерированной Мастером. *(См. также разд. "Создание процедур обработки событий", гл. 13.)*

Динамическая фильтрация данных

Предположим, у нас есть форма, содержащая подчиненную форму, в которой отображается много записей, и хотелось бы интерактивно задавать в главной форме

какое-нибудь условие для отбора интересующих записей в подчиненной форме, чтобы упростить их визуальный анализ.

Это несложно сделать с помощью применения фильтра для источника данных формы (в данном случае — подчиненной). Фильтр можно применить двумя способами:

- с помощью интерфейса Access (соответствующих кнопок на панели инструментов **Режим формы** (Form View), команд стандартного меню и контекстного меню поля формы);
- с помощью изменения значений свойств формы.

Первый способ применения фильтра в форме во многом аналогичен подобным действиям в режиме Таблицы, которые были рассмотрены в *разд. "Фильтрация данных" гл. 2*. Мы рассмотрим второй способ.

В этом случае применяются свойства формы **Фильтр** (Filter) и **Применение фильтров** (Filter On). Свойство **Фильтр** (Filter) содержит строковое значение — предложение `WHERE` на языке SQL (без ключевого слова `WHERE`). Свойство **Применение фильтров** (Filter On) определяет, применен ли фильтр, указанный в свойстве **Фильтр** (Filter), к форме. В отличие от свойства **Применение фильтров** (Filter On), свойство **Фильтр** (Filter) можно задать и в режиме Конструктора в окне свойств формы, чтобы фильтр применился в момент загрузки формы. Но нам нужно изменять фильтр динамически — в процессе работы с формой в режиме Формы. Мы реализуем это с помощью процедуры VBA.

И хоть данная глава не предназначена для описания программирования на VBA, мы приведем пример соответствующей процедуры VBA, поскольку он обеспечивает несложное решение довольно типичной задачи.

Рассмотрим решение этой проблемы на примере формы `Типы` (Categories), содержащей подчиненную форму `Список товаров` (Product List). Организуем отбор товаров в подчиненной форме в зависимости от наличия их поставок:

1. Скопируйте форму `Типы` (Categories) базы данных `Борей` (Northwind) и сохраните копию с именем `Типы (пример)`. Откройте созданную форму в режиме Конструктора.
2. Создайте в главной форме элемент **Поле со списком** (ComboBox), содержащее 2 столбца с фиксированным набором значений *(см. гл. 5)*. Метку списка назовите `Условие:`. Свойства списка должны иметь значения, описанные в табл. 9.1.

Таблица 9.1. *Свойства фильтрующего элемента*

Свойство	Значение
Тип источника строк (Row Source Type)	**Список значений** (Value List)
Количество столбцов (Column Count)	`2`
Источник строк (Row Source)	`0;"Поставки выполняются";-1; "Поставки прекращены";2;"Все товары"`
Присоединенный столбец (Bound Column)	`1`

Таблица 9.1 *(окончание)*

Свойство	Значение
Ширина столбцов (Column Widths)	`0cm;2,5cm`
Данные (ControlSource)	оставить пустым
Имя (Name)	`Отбор`

Замечание

Обратите внимание на значения созданного элемента в свойстве **Источник строк** — `0`, `-1` и `2`. Первые два значения выбраны в соответствии с возможными вариантами значений поля `ПоставкиПрекращены` (Discontinued) таблицы `Товары` (Products), которая является источником данных для подчиненной формы в данном примере. По этому полю и будет проводиться фильтрация данных в подчиненной форме.

3. В окне свойств созданного списка раскройте вкладку **События** (Events) и нажмите кнопку Построителя слева от свойства **После обновления** (After Update).
4. В появившемся диалоговом окне выделите элемент списка **Программы** (Code Builder) и нажмите кнопку **ОК**.
5. Откроется редактор VBA, отображающий сгенерированный заголовок процедуры обработки события. Вставьте в процедуру следующий код:

```
Private Sub Отбор_AfterUpdate()
    If Me!Отбор = 2 Then
       [Список товаров].Form.Filter = ""
       [Список товаров].Form.FilterOn = False
    Else
       [Список товаров].Form.Filter = _
           "[ПоставкиПрекращены] = " & Me!Отбор
       [Список товаров].Form.FilterOn = True
    End If
End Sub
```

В английской версии `Борей` вместо русских названий подчиненной формы `[Список товаров]` и поля `[ПоставкиПрекращены]` используйте их английские аналоги: `[Product List]` и `[Discontinued]`.

6. Закройте редактор VBA. Сохраните форму.

Откройте форму `Типы (пример)` в режиме Формы. Попробуйте выбирать в созданном списке разные значения и переходить от записи к записи. Вы увидите, что в соответствии с выбранным в списке значением в подчиненной форме отображаются только товары с выполняющимися поставками, только товары с прекращенными поставками или все товары.

Автоматическая подстановка значения в поле ввода

Часто для заполнения некоторых полей таблиц требуется вычислить выражение на основе данных в других полях этой таблицы и, возможно, связанных с ней таблиц.

Чтобы при вводе данных не вычислять это значение каждый раз самостоятельно, требуется создать форму для ввода данных, выполняющую необходимые вычисления автоматически. Это позволит застраховаться от ошибок при вычислениях и сократит время ввода данных.

Представим себе такой пример (отвлечемся от базы данных Борей). Предположим в нашей базе есть таблица Товары (Products) со списком товаров, где для каждого товара указаны его название и цена (таблица содержит поля НаименованиеТовара (ProductName) и Цена (UnitPrice), первое поле является ключевым). Есть другая таблица Заказы (Orders) со списком заказов на эти товары, содержащая информацию о типе и количестве заказанного товара (поля НаименованиеТовара (ProductName) и Количество (Quantity)), а также о его стоимости (поле Сумма (OrderPrice)). Таблица Заказы (Orders) связана с таблицей Товары (Products) по полю НаименованиеТовара (ProductName) отношением "один-ко-многим". Требуется создать форму для добавления новых заказов так, чтобы при выборе необходимого товара из предложенного списка и после ввода заказанного количества в поле Сумма (OrderPrice) автоматически подставлялось значение стоимости всего заказа для данного товара.

Для решения поставленной задачи требуется выполнить следующие действия (реализация этого примера — небольшая база данных Заказы.mdb — есть на компакт-диске, прилагаемом к книге):

1. Создайте запрос, содержащий поле Цена (UnitPrice) из таблицы Товары (Products) и все поля из таблицы Заказы (Orders). Сохраните его с именем Заказано (Ordered).
2. Создайте форму Заказы (Orders) на основе запроса Заказано.
3. Поместите в форму элемент управления **Список**, связанный с полем НаименованиеТовара (ProductName) источника данных формы.

Замечание

Это можно выполнить с помощью окна **Список полей** (Filed List) *(см. гл. 5)*, предварительно указав в режиме Конструктора таблицы Заказы (Orders) для поля НаименованиеТовара на вкладке **Подстановка** (Lookup) значение **Поле со списком** (Combo Box) для свойства **Тип элемента управления** (Display Control) и имя таблицы Товары (Products) в качестве значения свойства **Источник строк** (Row Source).

4. Поместите в форму также текстовые поля, связанные с полями Количество (Quantity) и Сумма (OrderPrice) источника данных формы.
5. Для поля Количество (Quantity) в окне свойств раскройте вкладку **События** (Events). Для события **После обновления** (After Update) создайте процедуру обработки события *(см. разд. "Динамическая фильтрация данных" этой главы)* со следующим содержанием:

```
Private Sub Количество_AfterUpdate()
    Сумма = Цена * Количество
End Sub
```

Таким образом, после ввода значения в поле Количество (Quantity) в поле Сумма (OrderPrice) будет автоматически подставляться вычисленная стоимость заказа. А поскольку поле Сумма (OrderPrice) является присоединенным к одноименному

полю в таблице Заказы (Orders), вычисленное значение автоматически попадет в таблицу.

Совет

Не всегда вычисляемые значения требуется сохранять в самих таблицах. Чаще всего достаточно только отображать результат вычисления в вычисляемом поле формы или запроса для информирования пользователя. Ведь это значение можно вычислить в любой момент на основе данных, хранящихся в таблицах, и не потребуется выполнять проверку на правильность сохраненного значения при изменении значений полей, использованных в его вычислении, в других формах или непосредственно в таблице.

(См. также разд. "Создание вычисляемых полей" этой главы и разд. "Вычисления на выделенных записях таблицы" гл. 8.)

Создание многостраничных форм

Многостраничные формы используются в том случае, если данных, связанных с объектом, так много, что их невозможно разместить в форме, ограниченной размерами экрана.

Для создания многостраничных форм используются элементы управления **Разрыв страницы** (Page Break) и **Набор вкладок** (Tab Control). Более распространенным является второй способ создания многостраничных форм — с помощью элемента управления **Набор вкладок** (Tab Control). Для того чтобы посмотреть, как выглядит такая форма и как она устроена, вы можете открыть в режиме Конструктора форму Сотрудники (Employees) — пример многостраничной формы с вкладками в демонстрационном приложении Борей. Форма Сотрудники (Employees) содержит элемент управления **Набор вкладок** (Tab Control), состоящий из двух вкладок. Переключение между ними выполняется с помощью щелчка мыши по ярлыку вкладки.

Создание многостраничных форм с элементом *Разрыв страницы*

Рассмотрим процедуру создания формы, подобной форме Сотрудники (Employees) в приложении Борей (Northwind), но состоящую не из набора вкладок, а из страниц, разделенных элементом управления **Разрыв страницы** (Page Break).

1. На панели инструментов **База данных** (Database) нажмите стрелку справа от кнопки **Новый объект** (New object) и выберите элемент **Форма** (Form).
2. В появившемся диалоговом окне выделите элемент списка **Автоформа: в столбец** (AutoForm: Columnar) и в раскрывающемся списке выберите таблицу Сотрудники (Employees). Нажмите кнопку **ОК**.
3. Автоматически будет создана форма, содержащая все поля из таблицы Сотрудники (Employees). Откройте ее в режиме Конструктора (например, выбрав команду **Вид | Конструктор** (View | Design View)).
4. Высота формы будет около 7,5 см. Растяните нижний край области данных формы так, чтобы высота области данных стала в два раза больше — 15 см.

Внимание!

При проектировании многостраничной формы нужно заранее спланировать, как разместить информацию на ее страницах. Обычно логически связанные данные группируются и разносятся по страницам формы. Однако при использовании элемента управления **Разрыв страницы** (Page Break) важно, чтобы все страницы были одинаковой высоты, т. к. в противном случае при переходе со страницы на страницу не всегда происходит точное переключение, может оказаться, что в окне видна только часть нужной страницы.

5. Пользуясь линейкой разметки формы, слева от нее, заметьте деление напротив отметки 7,5 см и перетащите туда с Панели элементов (Toolbox) элемент управления **Разрыв страницы** (Page Break). Этот элемент управления всегда располагается у левой границы формы и определяет конец текущей страницы.
6. Мысленно разделите все поля на форме на две группы: те, что относятся к служебным данным о сотруднике, и те, что относятся к его личным данным, как они разделены в форме `Сотрудники` (Employees). Перетащите все поля, относящиеся к личной информации в нижнюю часть формы — ниже элемента **Разрыв страницы** (Page Break). Эта часть формы будет служить второй страницей.
7. Чтобы сделать форму более наглядной, добавьте сразу под элементом **Разрыв страницы** (Page Break) надпись с текстом `Личные данные`. А также растяните нижнюю границу раздела **Заголовок формы** (Form Header) на высоту 1 см и добавьте в этот раздел вычисляемое поле (без метки), содержащее выражение *(см. разд. "Создание вычисляемых полей" этой главы)*:

   `=[Имя] & " " & [Фамилия]` (или `=[FirstName] & " " & [LastName]`).

   Отформатируйте добавленные элементы так, чтобы их выделить.
8. В многостраничных формах обычно не размещают полос прокрутки, это излишне, поэтому установите для свойства **Полосы прокрутки** (Scroll Bars) значение **Отсутствуют** (Neither) в окне свойств формы. Мы практически закончили работу над содержимым формы, должно получиться примерно то, что показано на рис. 9.39. Для наглядности на рисунке мы обвели элемент **Разрыв страницы** (Page Break).
9. Перемещая рамку окна формы, установите высоту окна так, чтобы самый нижний край был около отметки 5,5 см на линейке разметки (так, чтобы в режиме Формы высота окна формы была примерно равна высоте заголовка формы плюс высота одной страницы — это 1 см + 7,5 см).
10. С помощью кнопки **Сохранить** (Save) на панели инструментов сохраните созданную форму с именем `Сотрудники (пример с переводом страниц)`.
11. Откройте форму в режиме Формы — команда **Вид | Режим формы** (View | Form View).

Созданная форма состоит из двух страниц. На первой странице размещены служебные данные о сотруднике, а на второй странице — личные данные. Переключение между страницами, как обычно, выполняется с помощью клавиш <PageUp> и <PageDown>. Но кроме этого, для удобства переключения можно добавить на каждую страницу специальные кнопки: **Служебные данные** (Company Info) и **Личные**

данные (Personal Info), с реакцией на событие **Нажатие кнопки** (Click), вызывающей макрокоманду переключения на другую страницу: `GoToPage` с заданным номером страницы (в нашем примере — `1` или `2`) или инструкцию `DoCmd.GoToPage` в процедуре VBA. *(См. также "Создание процедур обработки событий", гл. 13.)*

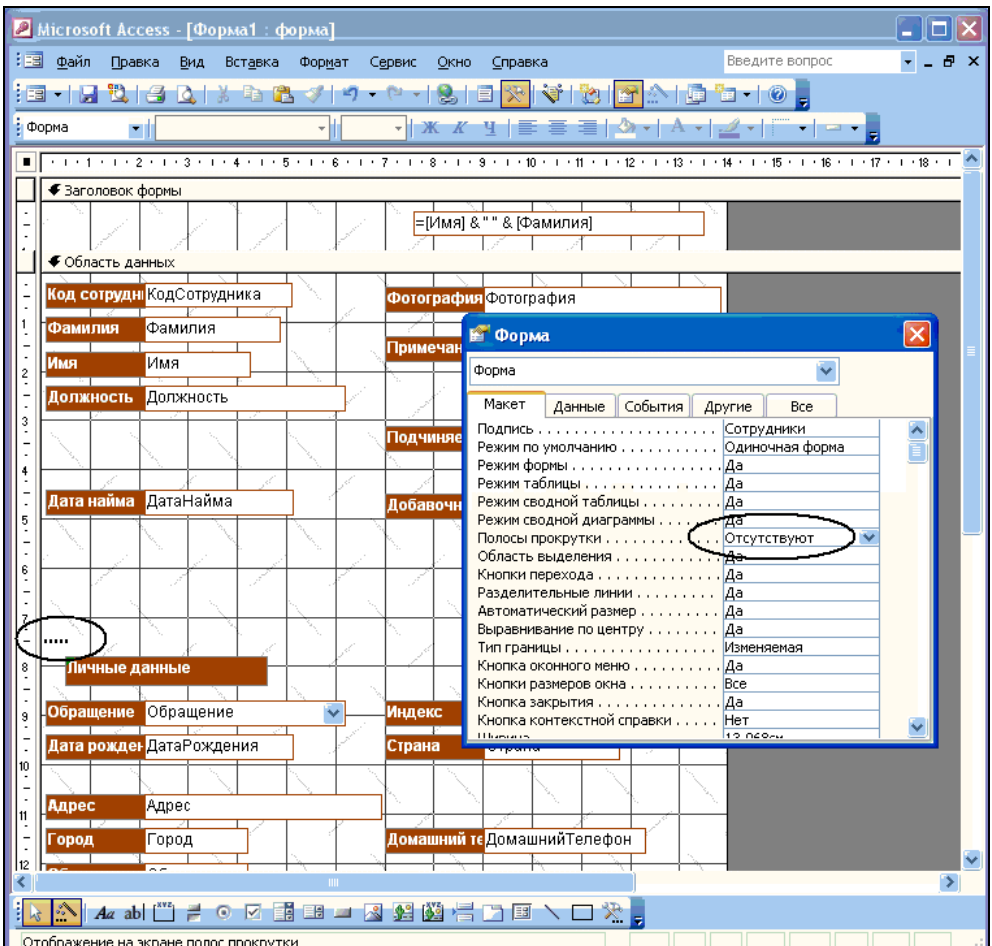


Рис. 9.39. Использование элемента **Разрыв страницы**

Обратите внимание на поле, которое расположено в заголовке формы. Это вычисляемое поле, значение которого меняется для каждой записи в форме. Оно содержит имя и фамилию сотрудника. Это поле предназначено для того, чтобы при переходе на вторую страницу видеть имя и фамилию сотрудника, тогда пользователь всегда будет знать, с какой записью таблицы он работает.

Чтобы придать созданной форме законченный вид, нужно будет еще поработать с ее оформлением и, возможно, преобразовать типы некоторых элементов, присоеди-

ненных к полям источника данных. *(См. также разд. "Преобразование одного элемента управления в другой" этой главы и разд. "Оформление формы и ее элементов" гл. 5.)*

Создание многостраничной формы с вкладками

А теперь рассмотрим процедуру создания многостраничной формы с набором вкладок:

1. Нажмите кнопку **Новый объект** (New object) на панели инструментов **База данных** (Database) и выберите объект **Форма** (Form).
2. В диалоговом окне выделите элемент списка **Конструктор** (Design View) и нажмите кнопку **ОК**.
3. Выведите на экран Панель элементов (Toolbox), если она не видна, и нажмите на ней кнопку **Вкладка** (Tab Control). Этот элемент управления не имеет Мастера, поэтому состояние кнопки **Мастера** (Control Wizard) не имеет значения.
4. В области данных формы очертите прямоугольную область достаточно большую, чтобы на ней можно было разместить несколько элементов управления. Access создаст в форме элемент управления **Набор вкладок** (Tab Control), состоящий из двух страниц.
5. Если окно **Свойства** (Properties) в данный момент открыто, то оно показывает свойства элемента управления **Набор вкладок** (Tab Control). Однако наборы свойств существуют и отдельно для каждой страницы. Чтобы их просмотреть, щелкните по ярлыку любой вкладки. Теперь в окне свойств отображаются свойства соответствующей страницы (рис. 9.40).
6. Измените заголовок первой страницы, введя в ячейку свойства **Подпись** (Caption) слово `Клиенты`. Для второй страницы введите значение `Квартальные обороты`. Обратите внимание, как ширина вкладки автоматически меняется в зависимости от длины текста заголовка.
7. В окне свойств формы установите в качестве значения свойства **Источник записей** (Record Source) таблицу `Клиенты` (Customers). Свойству **Полосы прокрутки** (Scroll Bars) присвойте значение **Отсутствуют** (Neither), свойству **Область выделения** (Record Selectors) — значение **Нет** (No). В поле свойства **Подпись** (Caption) введите название формы, например `Сведения о клиентах`. Закройте окно свойств.
8. Раскройте первую вкладку — **Клиенты**. Теперь на свободное поле страницы можно размещать другие элементы управления так же, как обычно в форме. Разместите на ней элементы управления, присоединенные к полям таблицы `Клиенты` (Customers). Например, с помощью окна **Список полей** (Filed List).
9. Раскройте вторую вкладку — **Квартальные обороты**. Перенесите на нее форму `Подчиненная для квартальных оборотов` (Quarterly Orders Subform) из окна базы данных с помощью мыши. Эта форма станет подчиненной формой в создаваемой нами форме. Настройте размеры элемента управления **Подчиненная форма/отчет** (Subform/Subreport) таким образом, чтобы поместилась вся необходимая информация.
10. Форма готова. Сохраните ее с именем `Сведения о клиентах (пример)`.

Откройте созданную форму в режиме Формы. Вы должны получить результат, похожий на тот, что представлен на рис. 9.41 и 9.42.

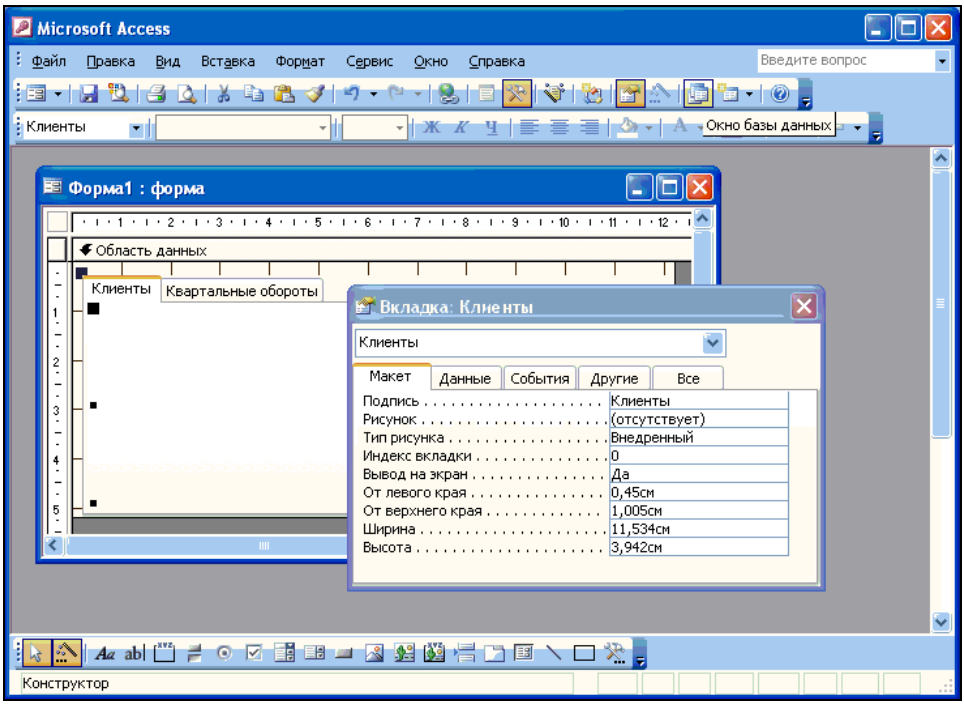


Рис. 9.40. Форма с набором вкладок в режиме Конструктора

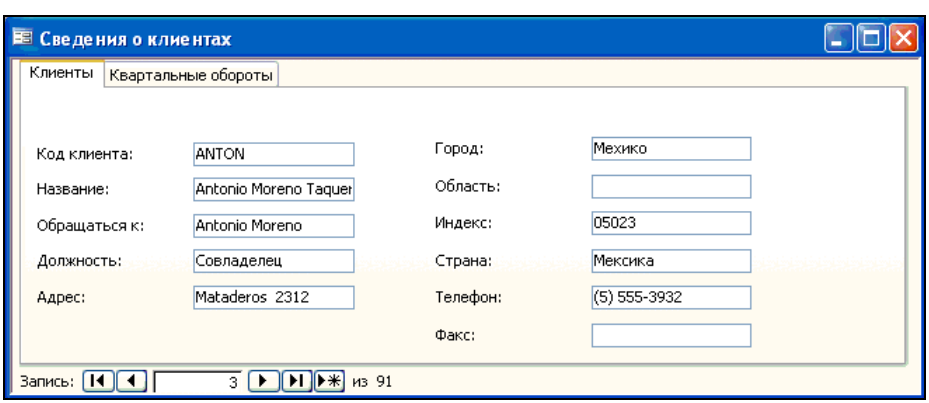


Рис. 9.41. Первая вкладка формы `Сведения о клиентах` (пример)

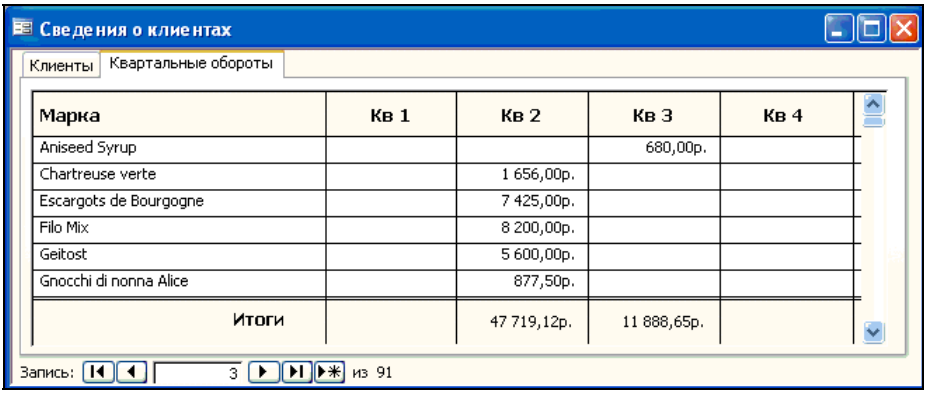


Рис. 9.42. Вторая вкладка формы `Сведения о клиентах` (пример)

Добавление и удаление страниц в наборе вкладок

Если для размещения всей информации недостаточно двух страниц, то можно легко добавить новые страницы к набору.

1. Раскройте любую вкладку, щелкнув по ее ярлыку.
2. Щелкните по ней правой кнопкой мыши и выберите в контекстном меню команду **Добавить вкладку** (Insert Page) или команду **Вставка | Вкладка** (Insert | Page).
3. Новая страница будет вставлена в конец набора вкладок.

Для того чтобы удалить лишнюю страницу, выполните следующее:

1. Раскройте вкладку, которую нужно удалить, щелкнув по ее ярлыку.
2. Щелкните по ней правой кнопкой мыши и в контекстном меню выберите команду **Удалить вкладку** (Delete Page) или команду **Правка | Удалить** (Edit | Delete).

Изменение порядка следования страниц

Иногда требуется изменить порядок следования страниц в наборе вкладок, например, когда новую вкладку, вставленную в конец набора вкладок, нужно поместить перед другими вкладками:

Чтобы изменить порядок взаимного расположения вкладок:

1. Щелкните правой кнопкой по любой вкладке и в контекстном меню выберите команду **Последовательность вкладок** (Page Order).
2. В появившемся диалоговом окне **Порядок страниц** (Page Order) (рис. 9.43) установите необходимый порядок следования вкладок в форме: выделите в списке название вкладки, которую нужно переместить, и нажимайте кнопки **Вверх** (Move Up) и **Вниз** (Move Down), пока не переместите ее на нужную позицию.
3. Повторите эти действия для каждой вкладки, которую нужно переместить, и нажмите кнопку **ОК**.

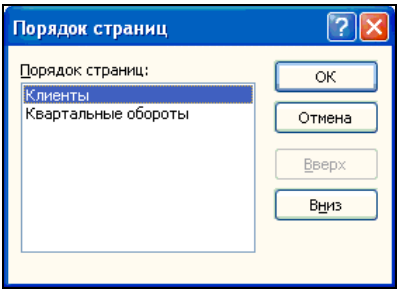


Рис. 9.43. Диалоговое окно **Порядок страниц**

Свойства элемента управления *Набор вкладок*

Наиболее важные свойства элемента управления **Набор вкладок** (Tab Control) приведены в табл. 9.2.

Таблица 9.2. *Свойства элемента управления* ***Набор вкладок***

Свойство	Описание	К какому элементу относится
Подпись (Caption)	Определяет текст, который отображается на ярлыке вкладки. Если это свойство не задано, то отображаться будет значение свойства **Имя** (Name)	Элемент управления **Вкладка** (Page) — страница элемента управления **Набор вкладок** (Tab Control)
Стиль (Style)	Определяет представление ярлыков (переключателей) вкладок: **Ярлычки**, **Кнопки**, **Нет** (Tabs, Buttons, None). По умолчанию устанавливается стиль **Ярлычки** (Tabs)	Элемент управления **Набор вкладок** (Tab Control)
Несколько строк (Multi Row)	Определяет, можно ли выводить ярлыки вкладок в несколько рядов. По умолчанию устанавливается значение **Нет** (No)	Элемент управления **Набор вкладок** (Tab Control)
Рисунок (Picture)	Определяет рисунок, который отображается на ярлыке страницы. Можно использовать встроенные значки Access или любой растровый рисунок — файл BMP	Элемент управления **Вкладка** — страница элемента управления **Набор вкладок** (Tab Control)
Высота ярлычка (Tab Fixed Height)	Высота ярлыков вкладок, указанная в определенных с помощью Панели управления Windows единицах измерения. Когда значение этого свойства 0, высота каждого ярлыка вкладки будет соответствовать его содержимому	Элемент управления **Набор вкладок** (Tab Control)

Таблица 9.2 (окончание)

Свойство	Описание	К какому элементу относится
Ширина ярлычка (Tab Fixed Width)	Ширина ярлыков вкладок, указанная в определенных с помощью Панели управления Windows единицах измерения. Когда значение этого свойства 0, ширина каждого ярлыка вкладки будет соответствовать его содержимому	Элемент управления **Набор вкладок** (Tab Control)

Как видно из этой таблицы, перечисленные свойства относятся либо к отдельной странице, либо к элементу управления в целом. Поэтому для их установки нужно выделять либо отдельную страницу (вкладку), либо весь элемент управления **Набор вкладок** (Tab Control). Самый простой способ выделения всего элемента управления — щелкнуть по свободной области справа от ярлыков.

Всплывающие формы и окна диалога

Вы, наверное, обратили внимание, что в Access, да и не только в нем, существуют такие окна, которые размещаются на экране поверх всех остальных окон, даже если в данный момент активным является другое окно. Примером такого окна является окно **Свойства** (Properties) в Конструкторе форм. Вы тоже можете в своем приложении создавать формы такого типа. Они называются *всплывающими окнами*.

Например, если **Главная кнопочная форма**, из которой можно выполнить основные действия в приложении, небольшая и не закрывает большинство других форм, ее можно сделать всплывающей, чтобы она была доступна в любое время. На рис. 9.44 представлена такая форма, которая находится поверх окна базы данных, хотя оно является активным.

Чтобы форма обладала таким свойством, необходимо присвоить значение **Да** (Yes) свойству **Всплывающее окно** (Pop Up). Это свойство находится на вкладке **Другие** (Other) окна свойств формы (рис. 9.45).

Обычную форму можно также превратить в модальное диалоговое окно, если свойству **Модальное окно** (Modal) этой формы задать значение **Да** (Yes). *Модальная форма* отличается тем, что пока она открыта, вы не можете переместить фокус на другой объект — форму, меню, кнопку панели инструментов и пр., т. е. другие объекты становятся недоступными, пока не будет закончена работа с данной формой, и она не будет закрыта.

Модальные формы обычно используются для создания специальных диалоговых окон, которые запрашивают у пользователя некоторую информацию. Для того чтобы выполнять другие задачи, пользователь должен ввести эту информацию и закрыть форму. Обычно в таком окне свойства **Кнопка оконного меню** (Control Box), **Кнопки размеров окна** (Min Max Buttons) и **Кнопка закрытия** (Close Button) устанавливают таким образом, чтобы эти кнопки не отображались в форме. Зато в самой форме создают две кнопки: **OK** и **Отмена** (Cancel). Кнопка **OK** позволяет выполнить даль-

нейшие действия и закрывает форму. Кнопка **Отмена** (Cancel) закрывает форму и выполняет все действия по прерыванию выполняемой операции. Для этого создают программы VBA или макросы, которые связывают с событием **Нажатие кнопки** (Click) каждой из этих кнопок. *(См. также разд. "Программирование на языке VBA", гл. 13.)*

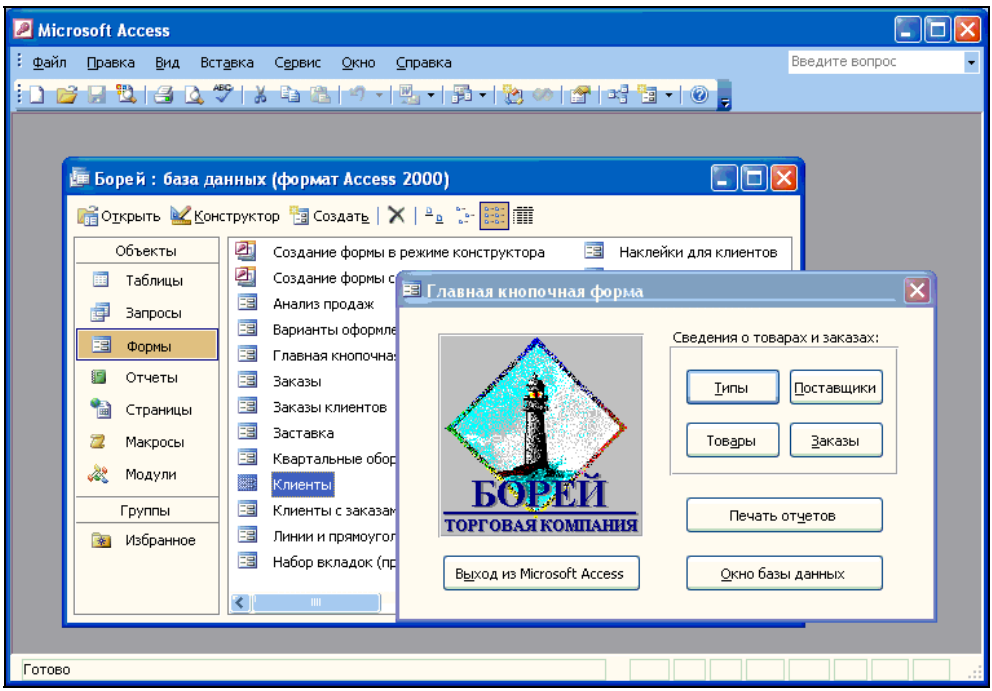


Рис. 9.44. Всплывающая **Главная кнопочная форма**

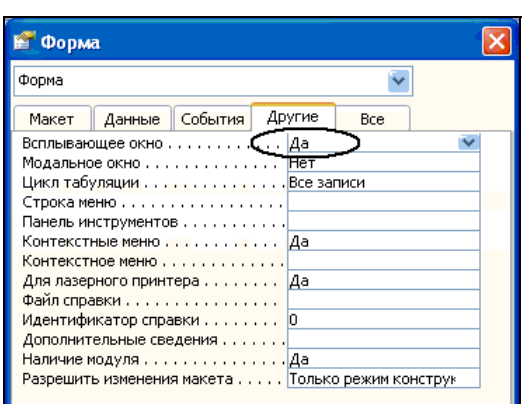


Рис. 9.45. Установка значения свойства **Всплывающее окно**

Обычно для всплывающих или модальных окон изменяют значение свойства **Тип границы** (Border Style), которое по умолчанию имеет значение **Изменяемая** (Sizable). Для всплывающей формы обычно устанавливают для этого свойства значение **Тонкая** (Thin). При этом всплывающую форму можно перемещать по экрану, но нельзя изменить ее размеры. Для модальной формы обычно устанавливают для свойства **Тип границы** значение **Окна диалога** (Dialog). При этом автоматически отключаются кнопки размеров окна (но кнопку закрытия вам придется отключить вручную, если это нужно), поэтому она не может быть развернута на весь экран, свернута в значок и просто изменена в размерах.

Примером модального диалогового окна является форма `Продажи по годам` (Sales by Year Dialog) в приложении `Борей` (Northwind) (рис. 9.46).

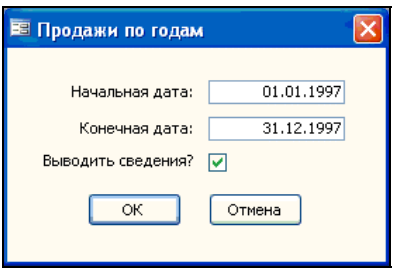


Рис. 9.46. Модальное диалоговое окно в приложении `Борей`

Вычисления в формах

Вычисления в формах используются:

- при создании вычисляемых полей, т. е. таких элементов управления, значения которых вычисляются на основе значений других элементов управления или встроенных функций;
- для вычисления значения по умолчанию элементов управления формы;
- в выражениях, с помощью которых задаются некоторые свойства элементов управления, например: условие на значение, условия для отбора записей в форме и ряд других.

Правила составления выражений в Access мы уже описывали *(см. разд. "Использование выражений в запросах", гл. 4)*. Поэтому здесь мы остановимся в первую очередь на вычисляемых полях формы и опишем, где они нужны и как их создать.

Создание вычисляемых полей

Простейшим вычисляемым полем является поле, в котором должна выводиться текущая дата. Чтобы создать такое поле, выполните следующее:

1. Создайте свободный элемент управления типа **Поле** (Text Box) в нужном месте формы.

2. Прямо в поле введите выражение `=Date()`. Знак равенства обязателен.
3. Откройте окно **Свойства** (Properties) для этого элемента и задайте для свойства **Формат поля** (Format) желаемый формат даты, например **Длинный формат даты** (Long Date).
4. Установите длину поля таким образом, чтобы в нем помещалось нужное количество символов.
5. Измените название метки этого текстового поля. Созданное поле будет выглядеть так, как показано на рис. 9.47.

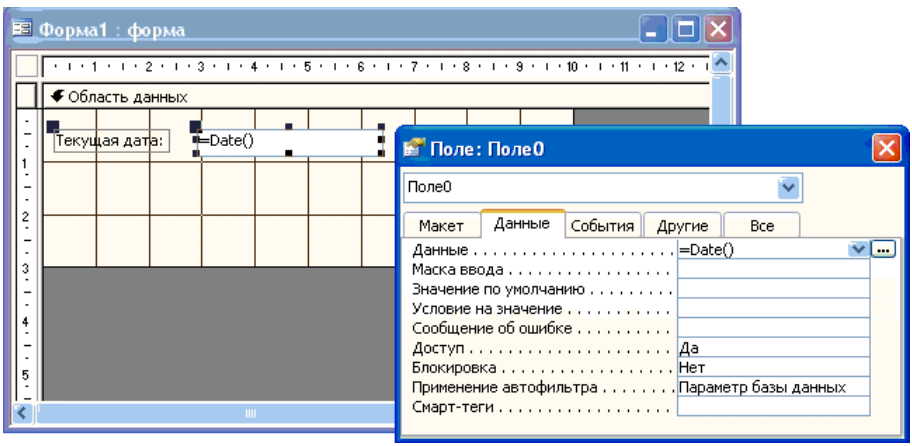


Рис. 9.47. Вычисляемое поле

Обычно для создания вычисляемых полей выбирают именно элемент управления **Текстовое поле**, хотя это и не обязательно. Допускается использовать для этого любые элементы управления, имеющие свойство **Данные** (Control Source).

Именно в свойстве **Данные** (Control Source) должно быть задано выражение, которое вычисляет значение данного элемента. Ввод выражения допускается прямо в элемент управления, как мы это сделали в примере. Однако, если выражение достаточно длинное, его не удобно вводить прямо в поле. Можно ввести выражение в ячейку свойства **Данные** (Control Source) в окне **Свойства** (Properties), а если выражение не помещается в ячейке свойства, можно нажать комбинацию клавиш <Shiftl>+<F2> и открыть окно **Область ввода** (Zoom). Еще можно воспользоваться Построителем выражений.

Вычисляемые поля создаются чаще всего в табличных или ленточных формах для отображения столбца с какими-нибудь вычисленными значениями. Например, стоимость товара рассчитывается как `Цена * Количество`. Для отображения таких расчетных данных используют два способа.

- Первый способ основан на том, что форма строится на базе запроса, в который включают вычисляемые столбцы. В форме создается текстовое поле, у которого в качестве источника данных указывается вычисляемый столбец запроса. Поле блокируется, чтобы пользователь не мог изменить данные в нем, ведь эти данные вычисляются в процессе выполнения запроса.

- Во втором случае форма может строиться прямо на базе таблицы. В ней создается текстовое поле, в котором в качестве источника данных указывается выражение, например: `= [Цена за единицу] * Количество`. При этом в выражении могут указываться не обязательно те поля, которые включены в форму, а любые поля базовой таблицы.

Замечание

При ссылках на поля в выражениях обязательно использование квадратных скобок, если название поля содержит пробелы, в остальных случаях можно обходиться без них.

Первый способ в большинстве случаев предпочтительнее, т. к. подобный запрос может быть использован не только в одной форме, но и в отчетах или других формах. Примером такой организации расчетов может служить форма `Подчиненная форма заказов` (Orders Subform) базы данных `Борей`. Она базируется на запросе `Сведения о заказах` (Order Details Extended), в котором создано вычисляемое поле `ОтпускнаяЦена` (ExtendedPrice), содержащее формулу:

```
ОтпускнаяЦена:
CCur([Заказано].[Цена]*[Количество]*(1-[Скидка])/100)*100
```

В примере английской версии Access:

```
ExtendedPrice:
CCur([Order Details].UnitPrice*[Quantity]*(1-[Discount])/100)*100
```

Свойство **Данные** (Control Source) элемента управления **ОтпускнаяЦена** (ExtendedPrice) в форме имеет значение `ОтпускнаяЦена` (ExtendedPrice) — имя вычисляемого поля запроса.

Замечание

В приведенной ранее формуле `CCur` — это функция, которая преобразует значение выражения в тип **Денежный** (Currency).

Создание вычисляемых полей для отображения итоговых значений

Можно создать вычисляемые поля в форме, которые будут содержать итоговые значения для нескольких записей. Например, требуется рассчитать сумму счета или заказа по каждой позиции счета.

Хотя такое вычисляемое поле уже есть в базе данных `Борей`, рассмотрим в качестве примера процедуру его создания:

1. Создать в области примечаний формы `Подчиненная форма заказов` (Orders Subform), содержащей позиции заказа, вычисляемое текстовое поле. В качестве значения этого поля задать итоговую сумму: `=Sum([ОтпускнаяЦена])` (или `=Sum([ExtendedPrice])`). То есть это поле рассчитывается как сумма значений поля `ОтпускнаяЦена` (ExtendedPrice) для всех позиций заказа (рис. 9.48).
2. Так как это поле не отображается, когда форма выводится на экран в режиме Таблицы, нужно в главной форме — `Заказы` (Orders) — создать еще одно вычис-

ляемое поле, для которого в качестве источника данных указать итоговое поле подчиненной формы (рис. 9.49):

```
=[Подчиненная форма заказов].[Form]![ПромежуточнаяСумма]
```

или:

```
=[Orders Subform].Form![Order Subtotal]
```

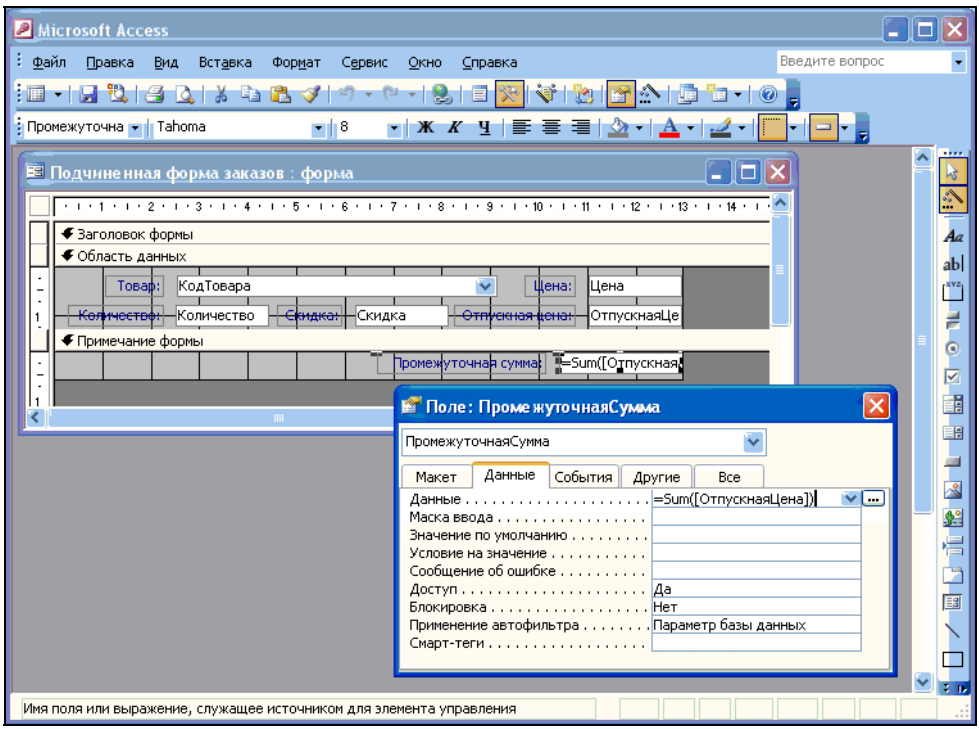


Рис. 9.48. Поле для вычисления итогового значения

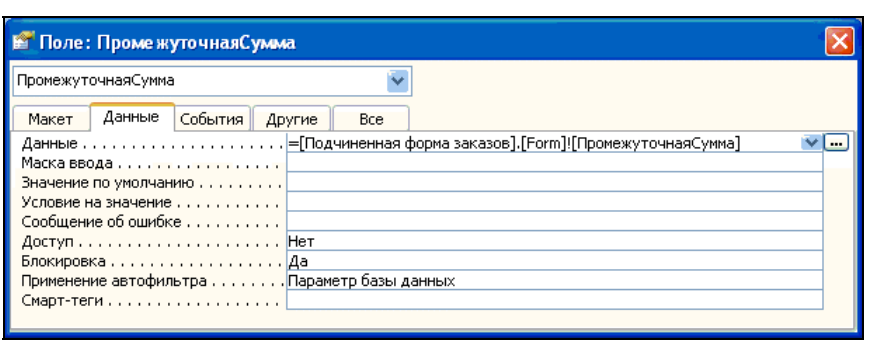


Рис. 9.49. Вычисляемое поле в главной форме `Заказы`

Особенностью создания итогового поля в подчиненной форме является то, что в функции Sum нельзя использовать имя вычисляемого элемента управления, можно применять только имя вычисляемого поля в базовом запросе. Если же суммируемые значения вычисляются прямо в форме, то в формуле для итогового поля придется повторить выражение для расчета этих значений. Например, если бы элемент управления ОтпускнаяЦена (ExtendedPrice) рассчитывался по той же формуле в форме, т. е. в качестве источника данных для него стояло выражение:

```
= CCur([Заказано].[Цена]*[Количество]*(1-[Скидка])/100)*100,
```

то в свойстве **Данные** (Control Source) для элемента управления **ПромежуточнаяСумма** (Order Subtotal) пришлось бы ввести выражение

```
= Sum(CCur([Заказано].[Цена]*[Количество]*(1-[Скидка]) /100)*100).
```

Это еще один аргумент в пользу того, что лучше такие вычисления выполнять в базовом запросе.

Замечание

Имена вычисляемых элементов управления нельзя использовать не только в функции Sum, но и в других статистических функциях, как агрегатных, так и в функциях по подмножеству.

Использование статистических функций в формах

При использовании статистических функций не обрабатываются записи, которые содержат пустое значение (Null) в указанных полях. Например, выражение Sum([В наличии]+[Принято]-[Отпущено]-[Заказано]) будет возвращать значение Null, если хотя бы одно из полей, участвующих в выражении, содержит пустое значение. Чтобы предотвратить такие ошибки в вычислениях, следует использовать функцию Nz, которая преобразует пустое значение в нулевое. Значит, описанное ранее выражение должно выглядеть так:

```
Sum(Nz([В наличии])+Nz([Принято])-Nz([Отпущено])-Nz([Заказано]))
```

В табл. 9.3 приведены наиболее часто используемые в формах статистические функции *(см. также разд. "Функции", гл. 4)*.

Таблица 9.3. *Статистические функции*

Функция	Описание
Sum	Сумма значений поля
Avg	Среднее значение поля
Min	Минимальное значение поля
Max	Максимальное значение поля
Count	Количество значений поля
DSum	Сумма значений в указанном подмножестве записей

Таблица 9.3 *(окончание)*

Функция	Описание
DAvg	Среднее значение в указанном подмножестве записей
DCount	Количество записей в подмножестве
DLookup	Значение конкретного поля в указанном подмножестве записей
DMin, DMax	Минимальное и максимальное значения поля в указанном подмножестве записей

В таблице приведены и обычные статистические функции, и статистические функции по подмножеству. В вычисляемых элементах управления форм могут использоваться и те и другие. При этом статистические функции по подмножеству используются, как правило, если требуется ограничить набор обрабатываемых записей, в противном случае применяют простые статистические функции. У статистических функций по подмножеству задаются три аргумента: выражение, домен и критерий отбора. Выражение, в данном случае, может состоять из имени поля в домене, а также может быть комбинацией операций над полями в домене, константами, именами элементов управления (в форме), вызовами функций. Домен — это таблица, запрос или инструкция SQL, т. е. набор записей, являющийся источником данных для заданного выражения. Критерий — необязательный параметр, задающий условие отбора записей в домене, в виде предложения `WHERE` на языке SQL (но без самого ключевого слова `WHERE`). Все три параметра задаются в виде строкового значения. Результатом функции является обработка значений заданного выражения для всех выбранных по заданному критерию записей в домене.

Далее приводятся примеры использования статистических функций.

Функция *DLookup*

Обычно используется для вывода в форме значения поля из таблицы, не входящей в базовый запрос. Например, в форме `Заказы` (Orders) можно отобразить дополнительные поля из таблицы `Клиенты` (Customers), которые не были включены в базовый запрос. Например, чтобы отобразить значение поля `Телефон` (Phone), в качестве значения свойства **Данные** (Control Source) вычисляемого элемента управления нужно ввести следующее выражение:

```
=DLookup("[Телефон]"; "[Клиенты]";
"[КодКлиента] = '" & Forms![Заказы]![КодКлиента] & "'")
```

Для английской версии Access:

```
=DLookUp("[Phone]";"[Customers]";
"[CustomerID] = '" & Forms![Orders]![CustomerID] & "'")
```

Критерием для выбора записи из таблицы `Клиенты` (Customers) является поле `КодКлиента` (CustomerID) в форме `Заказы` (Orders).

Замечание

В данном случае выбор записей идет по полю текстового типа `КодКлиента` (CustomerID), поэтому условие на его значение должно задаваться в виде строки,

заключенной в кавычки или апострофы. Поэтому необходима конкатенация заданного в условии значения с двух сторон с апострофом или кавычкой. Включение апострофа в строку не отличается от включения обычной буквы или цифры, а чтобы включить в строку символ кавычки, ее придется удвоить, чтобы она отличалась от кавычек, ограничивающих само строковое значение. Таким образом, если строка состоит из одной кавычки, она будет выглядеть так: """". В нашем примере для задания условия на текстовое значение мы использовали апострофы.

Функции *DCount* и *DSum*

Предположим, в форме Заказы (Orders) требуется создать вычисляемые поля для отображения количества заказов, сделанных клиентом, и общей суммы заказов. Для этого нужно создать два вычисляемых поля, а в качестве значений свойства **Данные** (Control Source) ввести следующие выражения:

```
=DCount("[КодЗаказа]"; "[Заказы]";
"[КодКлиента] = '" & Forms![Заказы]![КодКлиента] & "'")
=DSum("[ОтпускнаяЦена]"; "[Счета]";
"[КодКлиента] = '" & Forms![Заказы]![КодКлиента] & "'")
```

Для английской версии Access, соответственно:

```
=DCount("[OrderID]";"[Orders]";
"[CustomerID] = '" & Forms![Orders]![CustomerID] & "'")
=DSum("[ExtendedPrice]";"[Invoices]";
"[CustomerID] = '" & Forms![Orders]![CustomerID] & "'")
```

Функция *DMax*

Часто требуется организовать счетчик в одном из полей формы, например, номер заказа должен при вводе каждой новой записи увеличиваться на 1. Стандартный тип поля **Счетчик** (AutoNumber) не всегда подходит для этих целей, т. к. значение в поле такого типа нельзя вводить вручную. Если должна допускаться ручная корректировка номера заказа, удобнее для хранения номера заказа добавить в таблицу обычное числовое поле, а в форму поместить присоединенное к нему вычисляемое поле, у которого в свойстве **Значение по умолчанию** (Default Value) задано выражение:

```
=DMax("[КодЗаказа]"; "[Заказы]") + 1
```

Для английской версии:

```
=DMax("[OrderID]";"[Orders]")+1
```

Рекомендации по оформлению форм

В последнем разделе главы предлагаем вашему вниманию ряд рекомендаций по тому, как должны проектироваться формы, чтобы они выглядели профессионально. Возможно, наши советы покажутся вам тривиальными, однако опыт авторов в консультировании пользователей Access показывает, что даже простые и, казалось бы, очевидные правила на практике далеко не всегда выполняются. Если вы имеете большой опыт проектирования пользовательского интерфейса, можно смело пропустить этот раздел и перейти к изучению другого важнейшего типа объектов Access — отчетов.

Размещение элементов управления в форме

Если в форме требуется разместить много элементов управления — большое количество текстовых полей, полей со списком, флажков и кнопок, следует уделить внимание структуре формы. Форма состоит из нескольких разделов, в каждый из которых можно помещать необходимые элементы управления. Разделами формы являются заголовок и примечание формы, верхний и нижний колонтитул и область данных *(см. разд. "Структура формы", гл. 5)*.

В области заголовка обычно размещается не только заголовок формы, но заголовки полей, если это форма ленточного типа, некоторые вспомогательные поля, например, для ввода критериев поиска записей и т. д. В области данных размещаются все поля, отображающие данные из базовой таблицы или запроса. Область примечания можно использовать для размещения командных кнопок или вывода информационных сообщений для пользователя. Каждая из этих областей может быть выделена своим цветом. Если элементов очень много, следует сделать форму многостраничной.

Нужно учесть, что при печати формы область заголовка будет присутствовать только на первой странице, а область примечания — только на последней.

Размещение элементов управления в области данных

Элементы управления в области данных формы должны размещаться в определенной последовательности, при определении которой рекомендуется принимать во внимание следующие факторы:

- частота использования;
- важность;
- привычная последовательность (город после страны, улица после города и т. д.);
- схожесть с соответствующим документом.

Связанные элементы управления следует группировать. Чаще всего группировка выполняется по функциональному признаку. Группы элементов должны иметь название и быть выделенными (например, рамкой).

Внешний вид элементов управления

Предлагаем несколько рекомендаций для оформления внешнего вида элементов управления.

- Поля для ввода данных должны быть выровнены и иметь одинаковую высоту. Обычно надписи выравниваются по правому краю, а текстовые поля — по левому. Расстояния между ними должны быть одинаковыми и достаточными, чтобы обеспечить визуальное отделение полей друг от друга (почаще пользуйтесь командами **Выровнять** (Align), **Размер** (Size), **Интервал по горизонтали** (Horizontal Spacing) и **Интервал по вертикали** (Vertical Spacing) меню **Формат** (Format)).
- Пользователь должен четко видеть разницу между различными категориями представленной информации, например, между изменяемыми полями и неизменяемыми, обязательными для заполнения и необязательными.

- Все элементы, отображающие данные, должны быть подписаны, если пользователю не очевидно их назначение.
- Надпись должна отделяться от другой информации и располагаться слева от элемента, к которому она привязана, на расстоянии не менее одного пробела.
- Разные поля не должны иметь одинаковые надписи.
- Одинаковые поля в разных формах должны иметь одинаковые надписи.
- Там, где это возможно, нужно предлагать пользователю выбор варианта вводимого значения. Это обеспечивается такими элементами, как список, поле со списком, группа.
- Курсор должен быть всегда хорошо видимым на фоне элементов.

Формулировка надписей и инструкций

Форма, предназначенная для ввода данных, должна включать инструкции для пользователя с описанием возможных действий. Обычно инструкции требуются, когда пользователь выбирает (активизирует) элемент управления в форме. Приведем несколько общих правил для написания инструкций.

- Для описания действий нужно использовать известную пользователю терминологию.
- Используйте единый стиль во всех формах приложения.
- Если выбраны несколько действий, нужно сообщать, в каком порядке они будут выполнены.
- Информация должна быть краткой, но достаточной, чтобы пользователь смог понять, что от него требуется или что он может сделать.

Навигация в форме

Одинаковые действия во всех формах должны выполняться единообразно, например, все формы должны использовать одни и те же наборы клавиш для перемещения курсора по элементам управления. К счастью, это условие выполнить просто, т. к. Access позволяет задать наборы клавиш в виде параметров, которые действуют во всех формах приложения. Вам нужно только выбрать один из предлагаемых вариантов в диалоговом окне **Параметры** (Options), которое появляется при выборе команды **Сервис | Параметры** (Tools | Options). *(См. гл. 14 "Настройка пользовательского интерфейса".)*

Например, для перехода к следующему полю можно использовать клавиши <Tab> и <↓>, для перемещения к предыдущему полю — клавиши <Shift>+<Tab> и <↑>, для перемещения между записями — кнопки навигации в нижней части формы или клавишу <Enter>.

Для этого нужно установить значения переключателей на вкладке **Клавиатура** (Keyboard) диалогового окна **Параметры** (Options) следующим образом:

- **Переход при нажатии Enter** (Move after enter) — **следующее поле** (Next field);
- **Переход по клавише со стрелкой** (Arrow key behavior) — **следующий знак** (Next character);

- **Поведение при входе в поле** (Behavior entering field) — **перейти в начало поля** (Go to start of field).

В многостраничных формах для перелистывания страниц используются по умолчанию клавиши <PageUp> и <PageDown>. Тем не менее рекомендуется для этих целей создать специальные кнопки, которые будут использоваться, если пользователь работает с мышью.

При открытии формы курсор, как правило, должен позиционироваться в поле, расположенном в левом верхнем углу. Настройте соответствующим образом последовательность перехода по элементам формы с помощью команды **Вид | Последовательность перехода** (View | Tab Order).

В любой форме для ввода данных пользователь должен иметь возможность:

- закрывать форму, сохраняя сделанные изменения;
- закрывать форму без сохранения (отменять изменения).

И хотя, строго говоря, для выполнения этих действий можно обойтись уже имеющимися средствами, т. к. Access автоматически сохраняет запись при переходе к новой записи или закрытии формы, а отменить изменения можно с помощью клавиши <Esc>, мы все-таки рекомендуем создать специальные кнопки **Сохранение** и **Отмена**, т. к. такой вариант является более очевидным для пользователя.

Использование цвета

Цвет обычно привлекает взгляд, поэтому его следует использовать для того, чтобы обратить внимание пользователя к определенным областям экрана, например, чтобы показать отношения или группировки, поскольку элементы одного цвета воспринимаются людьми как связанные. Цвет можно использовать также для выделения полей, обязательных для заполнения. При выборе цветовых решений следует придерживаться определенных принципов.

- Избегать использования ярких цветов на больших областях, поскольку они имеют тенденцию оставлять изображение на сетчатке глаза.
- Для фона должен использоваться неброский цвет (бледно-голубой, серый).
- Цвета белый, серый, черный должны использоваться для текстовых объектов, которые будут просматриваться в течение длительного времени.
- Цвета, выбранные для активного режима, или цвета подсветки должны обеспечивать максимальный контраст с цветом фона. Например, с черным фоном, использование красного и синего должно быть минимизировано — белый, желтый, зеленый и сиреневый обеспечивают лучший контраст с черным фоном.
- Одновременно должно быть видно не более пяти разных цветов, а лучше всего использовать два или три цвета, т. к. использование красочного экрана может первоначально привлечь больше внимания, но вызывает большее напряжение при работе в течение длительного времени.
- Цвет не должен быть единственным индикатором свойства или функции. Вместе с цветом можно использовать другие визуальные эффекты, например курсив, подчеркивание и т. п.

- Красный цвет является наиболее значимым, его резервируют для важных предупреждений и используют редко.
- Красный цвет не должен отображаться на зеленом фоне, как и зеленый цвет на красном, поскольку эта цветовая комбинация может вызывать раздражение глаз.
- Синий не должен использоваться для текстовых символов, линий или границ, за исключением случая, когда объект является "исчезающим", т. е. сливающимся с цветом фона. Синий цвет плохо виден и делает элементы размытыми и стертыми.
- Насыщенные красные и синие цвета не должны использоваться в смежных областях, поскольку это сочетание стимулирует нарушения в виде резкого эффекта глубины на изображаемом пространстве.
- Принятые цветовые решения необходимо строго соблюдать во всех формах приложения.

Оформление в едином стиле

Когда вы определитесь, какой стиль оформления подходит для форм вашего приложения, советуем выдерживать его при оформлении всех или большинства форм в нем. Удобнее всего сделать это, создав на основе такого оформления собственный автоформат или шаблон для оформления всех форм в приложении. Об автоформате и шаблоне оформления для форм уже говорилось в *гл. 5*. Такие средства оформления доступны и для форм, и для отчетов. Их применение очень похоже в обоих случаях.

(См. разд. "Применение автоформата" и "Параметры работы с формами" гл. 5, а также разд. "Применение шаблона отчетов" гл. 6.)

Использование в формах новых возможностей Access 2003

В *гл. 7* было рассказано о том, какие новые возможности появились в Access по сравнению с предыдущей версией. Некоторые из них, в частности, позволяют усовершенствовать разработку и использование форм. Например, для ввода данных в форму пользователь может использовать теперь не только клавиатуру и мышь, но и диктовать текст собственным голосом (на китайском (упрощенное письмо), английском (США) и японском языках). Это обеспечивается специальным средством распознавания речи, доступным в любом приложении Microsoft Office. А также с помощью голосовых команд можно выбирать элементы меню, панелей инструментов, диалоговых окон и областей задач (только для американского варианта английского языка). Средство распознавания речи появилось еще в Access 2002, но в новой версии оно усовершенствовано. Подробное описание установки и применения этой возможности можно прочитать в справочной системе Access 2003.

Стоит обратить внимание еще на одно полезное нововведение: смарт-теги (Smart Teg). Это совершенно новое средство для Access — оно появилось только в версии Access 2003, хотя в других офисных приложениях подобная возможность существо-

вала и раньше. Смарт-теги можно присоединить к некоторым типам элементов управления, таким как **Поле** (Field) и **Надпись** (Label). С помощью смарт-тегов, например, можно контролировать ввод текста в поля формы, и в случае совпадения введенного текста с ожидаемым значением, автоматически выполнить обработку данной информации. Для этого следует задать смарт-тег в качестве значения свойства **Смарт-теги** (Smart Tag) соответствующего элемента управления. Создать новый смарт-тег поможет Построитель. Вызвать Построитель смарт-тегов можно, нажав маленькую кнопку "**...**" напротив поля свойства **Смарт-теги** (Smart Tag) в окне свойств элемента управления.

ГЛАВА 10



Подготовка составных отчетов

Одной из важнейших функций, которые обычно предоставляют настольные СУБД, является возможность создания различных отчетов — документов, содержащих данные из таблиц базы данных. Отличительной особенностью Microsoft Access является наличие очень мощного генератора отчетов, который позволяет создавать отчеты высокого качества. Наличие Мастеров, генерирующих автоматически или при помощи диалога с пользователем самые разнообразные отчеты, значительно облегчает трудоемкий процесс их создания. В *гл. 6* мы уже рассматривали основные вопросы, касающиеся создания отчетов в Access. В этой главе мы поговорим о них более подробно, ориентируясь на профессиональное использование средств Access. В частности, покажем, как с помощью Мастера или в режиме Конструктора создавать достаточно сложные по структуре отчеты, включающие подчиненные отчеты или имеющие несколько уровней группировки записей. А также обсудим здесь многие другие вопросы о работе с отчетами, не вошедшие в *гл. 6*. Мы не будем затрагивать здесь тему программирования на VBA или использования макросов при создании отчетов, оставив ее для рассмотрения в последующих главах *(гл. 12 и 13)*, и покажем, как много можно сделать, применяя только интерактивные средства, предоставляемые Access. В качестве примеров будут использованы уже готовые отчеты, которые содержатся в учебной базе данных `Борей` (Northwind), или отчеты, созданные на основе таблиц этой базы данных.

Приложение Microsoft Access имеет существенные отличия и преимущества по сравнению с другими наиболее известными СУБД. Это главная черта продуктов Microsoft — удобство в использовании и одновременно мощность продукта — в сочетании с возможностью построения комплексных решений на базе современных технологий, а именно: интеграция баз данных с SQL Server, с другими приложениями Office и публикация интерактивных данных в Web.

В этой главе рассмотрены следующие вопросы:

- создание отчетов, включающих подчиненные отчеты;
- создание отчетов на базе перекрестных запросов;
- печать отчета в несколько колонок — в стиле газетных полос;
- способы сортировки и группировки записей в отчете;

- вычисления в отчетах;
- фильтрация данных в отчете.

Создание составных отчетов

Под составными отчетами в данном случае понимаются отчеты, имеющие сложную структуру: включающие подчиненные отчеты, построенные на базе перекрестных запросов, выводимые на печать в виде нескольких колонок, как это делается, например, в газетах. Далее будут описаны такие отчеты, их назначение и способы создания.

Применение подчиненных отчетов

По аналогии с составными формами, включающими подчиненные формы, можно предположить, что составные отчеты используются также и для печати данных из таблиц, связанных отношением "один-ко-многим". Однако это не совсем так. В большинстве случаев для отображения данных из связанных таблиц не требуется внедрять подчиненные отчеты *(пример такого отчета см. в разд. "Создание отчета с помощью Мастера" гл. 6)*.

Например, если нужно создать отчет по товарам с разбивкой по категориям товаров, то это можно сделать двумя способами:

1. В основном отчете выводятся данные о категории товаров, а в подчиненном — о товарах, входящих в категории. И это будет аналогично созданию формы `Категории товаров`, в которую включается подчиненная форма, отображающая данные из таблицы товаров.
2. То же самое может быть выполнено без использования подчиненного отчета с помощью группировки товаров по категориям. Отчет в этом случае должен строиться на базе запроса, включающего данные из таблиц `Категории` и `Товары`.

Тем не менее бывают ситуации, когда использование подчиненного отчета необходимо или предпочтительно. Например:

- Главный отчет представляет собой контейнер, в который включаются один или более подчиненных отчетов из несвязанных таблиц. В этом случае главный отчет может вообще не быть связанным с таблицей или запросом и содержать только надписи и вычисляемые элементы.
- Иногда итоговые данные требуется представить в документе в виде отдельной таблицы наряду с детальными данными. В этом случае просто группировкой записей с итоговыми строками не обойтись, нужно ввести подчиненный отчет, который выведет итоговые данные в требуемом виде.
- Группировка в отчетах Access является очень мощным механизмом, который позволяет получить самые разные способы обработки данных отчета. Однако только при использовании подчиненных отчетов можно создать группы, которые содержат две и более подгруппы данных.
- Главный отчет может содержать, во-первых, несколько подчиненных отчетов и, во-вторых, два уровня вложенности подчиненных отчетов. И, кроме того, на лю-

бом уровне вложенности наряду с подчиненным отчетом можно вставить в главный отчет подчиненную форму.

Основное достоинство подчиненных отчетов состоит в том, что с их помощью можно разместить на странице отчета самую разнообразную информацию в удобном структурированном виде, чего невозможно добиться, если просто создать сложный запрос, включив в него данные из многих таблиц.

Примеры использования подчиненных отчетов можно найти в списке отчетов учебной базы данных Борей (Northwind).

Например, на рис. 10.1 представлен отчет Продажи по типам (Sales by Category), который содержит один подчиненный отчет и внедренную диаграмму.



Рис. 10.1. Отчет, содержащий подчиненный отчет

Откройте отчет Продажи по типам (Sales by Category) в режиме Конструктора. Вы увидите, что подчиненный отчет размещен в разделе заголовка группы Категория

(CategoryName) (рис. 10.2). Оба отчета, и главный и подчиненный, базируются на одном и том же запросе Продажи по типам (Sales by Category), но в главном отчете выводятся данные о категориях — имя категории, а в подчиненном — о товарах, входящих в эту категорию (наименование товара и его цена).

Выделите элемент управления **Подчиненная форма/отчет** (Subform/subreport) и посмотрите свойства этого элемента. Вы увидите, что главный и подчиненный отчеты связываются по полю Категория (CategoryName). Поскольку и в формах, и в отчетах используется один и тот же элемент управления **Подчиненная форма/отчет**, свойства для связанных полей те же, что и в форме *(см. гл. 9)*.

Обратите внимание, что в макете подчиненного отчета нет элемента управления, связанного с полем Категория, однако, как и при связывании форм, в качестве полей для связи можно использовать поля базового запроса.

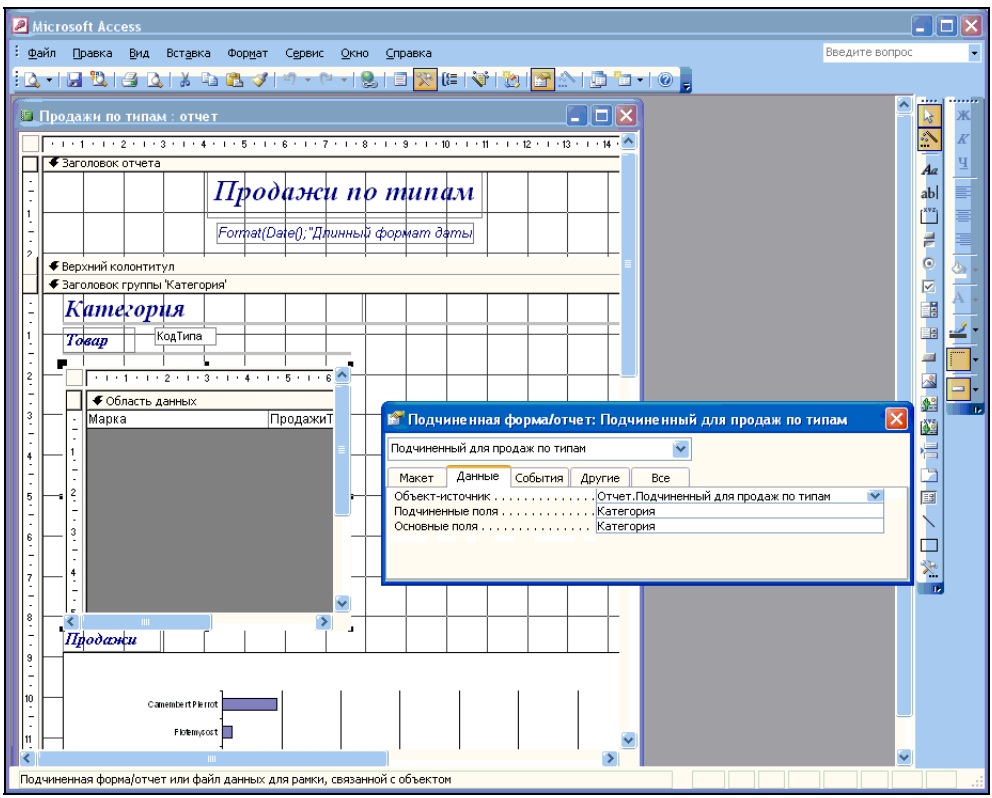


Рис. 10.2. Отчет Продажи по типам в режиме Конструктора

Отчет Продажи по годам (Sales by Year) является примером отчета, в котором подчиненный отчет используется для вывода итоговых данных о продажах по кварталам года (рис. 10.3).

Оба отчета, основной и подчиненный, базируются на запросе Продажи по годам (Sales by Year), который позволяет выбрать все заказы за указанный период времени

(проверяется поле ДатаИсполнения (ShippedDate). Запрос включает вычисляемое поле Год (Year), которое рассчитывается на основе даты заказа и используется при группировке в отчете, и поле, содержащее итоговую сумму каждого заказа ПромежуточнаяСумма (Subtotal).

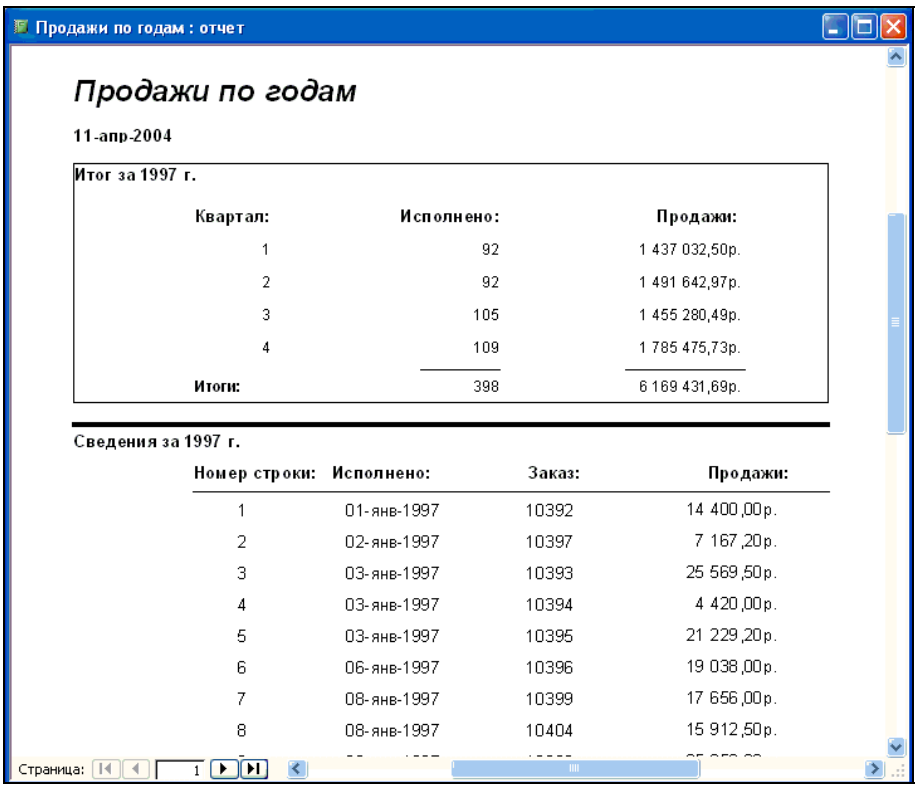


Рис. 10.3. Использование подчиненного отчета для вывода итогов

Подчиненный отчет Подчиненный для продаж по годам (Sales by Year Subreport) содержит несколько вычисляемых элементов:

- ГодНадпись (YearLabel), который печатается в верхнем левом углу, вычисляется с помощью выражения:

  ="Итог за " & [Год] & " г." (или: =[Year] & " Summary")

- Квартал (Quarter), который в качестве источника данных содержит выражение:

  =Format([ДатаИсполнения];"q") (или: =Format([ShippedDate];"q"))

- ПосчетПромежуточнойСуммы (CountSubtotal) — количество исполненных заказов, которое вычисляется с помощью статистической функции Count:

  =Count([КодЗаказа]) (или: =Count([OrderID]))

- КвПромежуточнаяСумма (QtrSubtotal) — итоговая сумма за квартал, которая вычисляется с помощью выражения:

  `=Sum([ПромежуточнаяСумма])` (или: `=Sum([Subtotal])`)

Группировка записей выполняется по году и по кварталу. Обратите внимание, что раздел **Область данных** (Detail) отчета пуст, т. е. в отчет выводятся только итоговые значения (рис. 10.4).

Детальные данные, т. е. все записи из базового запроса, включены в главный отчет.

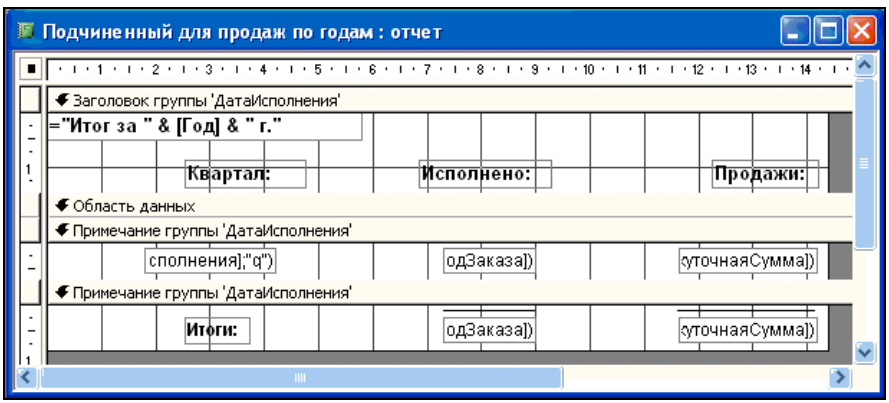


Рис. 10.4. Отчет Подчиненный для продаж по годам в режиме Конструктора

Создание подчиненных отчетов

Подчиненные отчеты могут создаваться двумя способами:

1. Путем создания подчиненного отчета в существующем отчете, для чего можно применить Мастера подчиненной формы/отчета.
2. Путем внедрения уже существующего отчета (или формы) в другой отчет (в этом случае внедряемый отчет становится подчиненным).

Для того чтобы продемонстрировать, как создаются составные отчеты, мы построим отчет, показывающий квартальные обороты клиентов. Сначала нужно создать запрос, который будет базовым для главного отчета. Этот запрос будет иметь параметр `[Отчетный год:]` и отбираться будут все клиенты, которые имеют заказы в указанном году (рис. 10.5).

Этот запрос очень просто создать на основе имеющегося в базе данных Борей подобного запроса Квартальные обороты (Quarterly Orders):

1. Выделите этот запрос в окне базы данных и скопируйте его, используя команды копирования и вставки **Правка | Копировать** (Edit | Copy) и **Правка | Вставить** (Edit | Paste). Дайте ему название Квартальные обороты клиентов (пример).
2. Откройте запрос в режиме Конструктора.

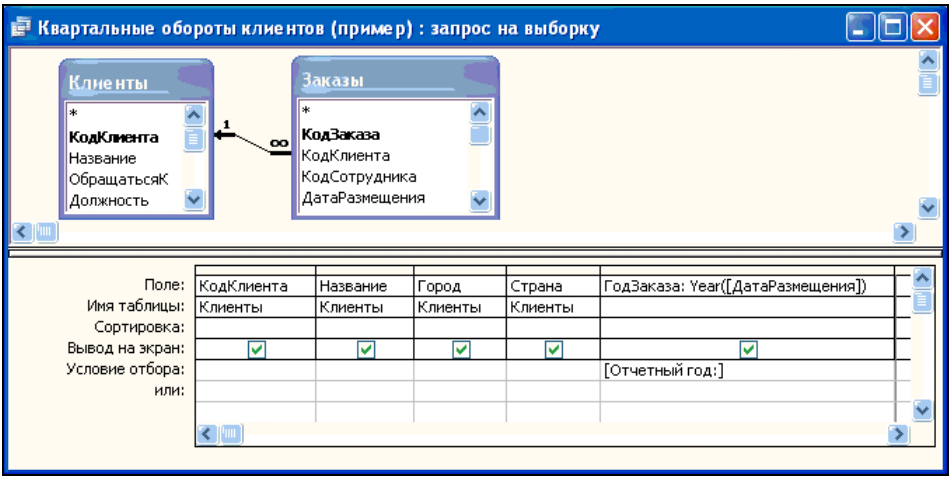


Рис. 10.5. Базовый запрос для отчета `Квартальные обороты клиентов`

3. Удалите последнее поле `ДатаРазмещения` (OrderDate) из таблицы полей внизу окна запроса и вместо него вставьте вычисляемое поле:

   ```
   ГодЗаказа: Year([ДатаРазмещения])
   ```

   (для английской версии используйте английское имя поля `OrderDate`, а в качестве имени вычисляемого поля можете задать `OrderYear`), а в строку **Условие отбора** (Criteria) введите имя параметра запроса `[Отчетный год:]`. Сохраните и закройте запрос.

Теперь создадим отчет на основе только что созданного запроса. Для этого можно воспользоваться Мастером отчетов (Report Wizard) *(см. гл. 6)*.

Выберите для помещения в отчет поля `Название` (CompanyName), `Город` (City) и `Страна` (Country), отсортируйте записи по полю `Название`, выберите тип отчета **в столбец** (Column) и один из предложенных стилей отчета. В результате вы получите отчет примерно такой, как изображен на рис. 10.6.

Для того чтобы построить подчиненный отчет, содержащий квартальные обороты клиентов, воспользуемся уже готовым запросом `Квартальные обороты по товарам` (Quarterly Orders by Product), слегка его изменив. А именно, нужно удалить из запроса последнее поле `ДатаРазмещения` (OrderDate), для которого задано лишнее в нашем примере условие выборки. Тогда запрос будет выглядеть так, как это представлено на рис. 10.7. Сохраните его с именем `Квартальные обороты клиентов по товарам (пример)`.

Для создания подчиненного отчета воспользуемся Мастером подчиненной формы/отчета (Subform/Subreport Wizard). Для этого:

1. Откройте отчет `Квартальные обороты клиентов (пример)` в режиме Конструктора.
2. Выберите элемент управления **Подчиненная форма/отчет** (Subform/Subreport) на Панели элементов (Toolbox) и разместите его в разделе **Область данных** (Detail)

ниже полей, содержащих данные о клиенте. (Кнопка **Мастера** (Control Wizards) должна быть нажата).

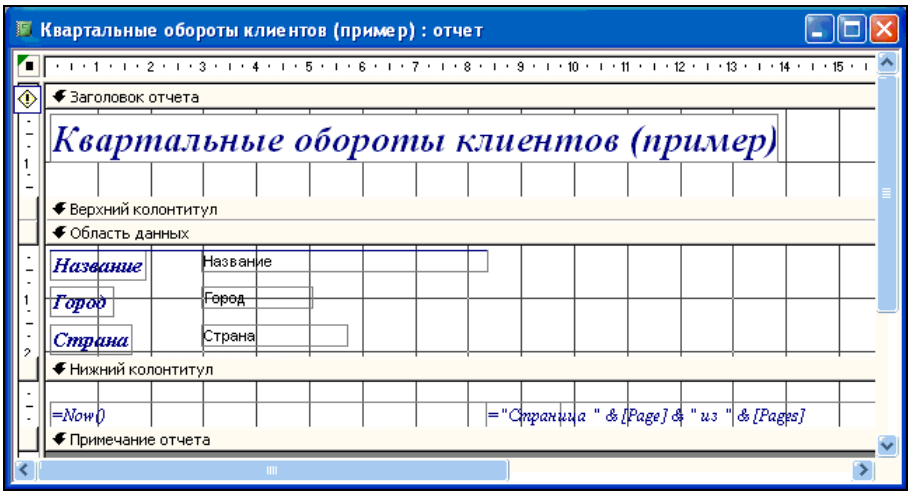


Рис. 10.6. Главный отчет составного отчета

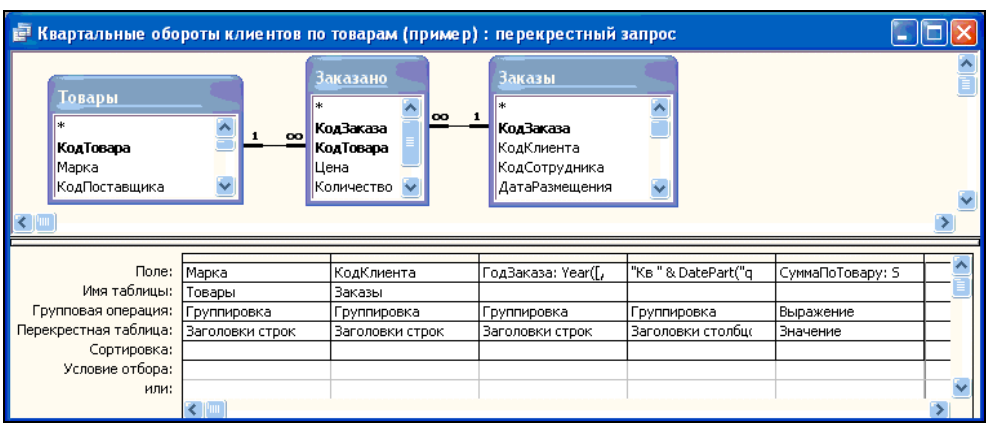


Рис. 10.7. Базовый запрос для подчиненного отчета

3. В первом диалоговом окне Мастера выберите переключатель **Имеющиеся таблицы и запросы** (Use existing Tables and Queries) (рис. 10.8). Нажмите кнопку **Далее** (Next).

4. Во втором диалоговом окне нужно определить, какие поля требуется включить в подчиненный отчет (рис. 10.9). При этом поля в общем случае можно выбирать из нескольких таблиц и запросов. В данном случае нужно в раскрывающемся списке **Таблицы и запросы** (Tables/Query) выбрать запрос `Квартальные обороты клиентов по товарам (пример)` (который был создан для этого примера), затем

нажать кнопку с двумя стрелками, чтобы включить в подчиненный отчет все поля запроса, и нажать кнопку **Далее** (Next).

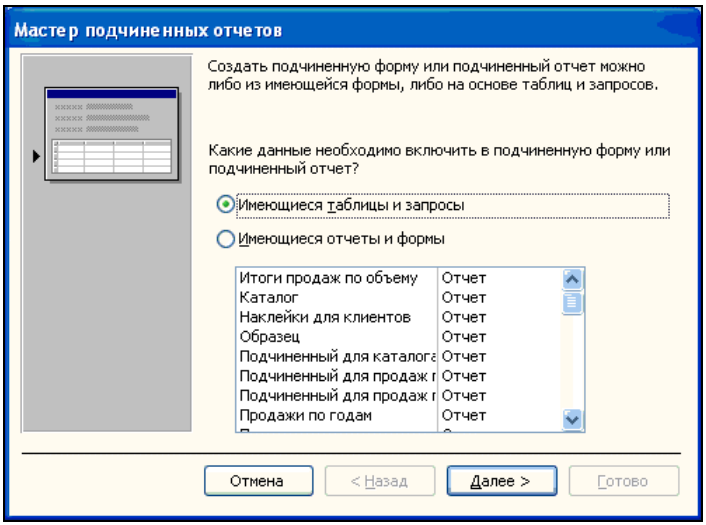


Рис. 10.8. Первое диалоговое окно Мастера подчиненных отчетов

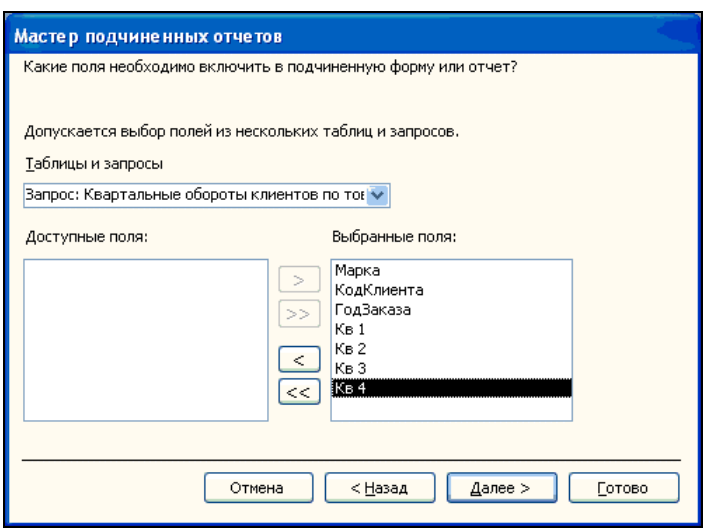


Рис. 10.9. Второе диалоговое окно Мастера подчиненных отчетов

5. В следующем диалоговом окне необходимо определить связи между главным и подчиненными отчетами, если эта связь существует, или выбрать из списка значение **Отсутствует** (None), если вы создаете независимый подчиненный отчет. В данном случае можно выбрать связь в отображенном списке, но нам предло-

женные варианты не подходят, поэтому выберите переключатель **Самостоятельное определение** (Define my own), чтобы задать эту связь самостоятельно.

Замечание

Так как поля в связываемых источниках данных имеют одинаковые типы и названия, Мастер может автоматически связать отчеты по этим полям. Именно такие связи будут предложены в списке возможных вариантов связывания отчетов.

6. В диалоговом окне появятся две пары раскрывающихся списков (рис. 10.10). В списках слева отображаются поля главного отчета, а в списках справа — связанные с ними поля подчиненного отчета. Связь может быть задана по одному или нескольким полям. В данном случае требуется задать связь по двум полям, чтобы связать источники данных главного и подчиненного отчетов одновременно и по коду клиента, и по отчетному году. Для этого выберите в раскрывающихся списках слева элементы: `КодКлиента` (CustomerID) и `ГодЗаказа` (OrderYear), и соответствующие (с такими же именами) элементы в раскрывающихся списках напротив (справа), как представлено на рис. 10.10.

Замечание

То, что между главным и подчиненным отчетами установлена связь по полям `КодКлиента` и `ГодЗаказа`, означает, что каждая запись в главном отчете будет связана с набором записей из подчиненного отчета с соответствующими значениями в связанных полях. А поскольку значение года заказа в главном отчете определяется значением параметра запроса `[Отчетный год:]`, то в связанном описанным способом подчиненном отчете будут отобраны только те записи, у которых значение поля `ГодЗаказа` совпадет со значением такого же поля в главном отчете, а значит, и с заданным пользователем значением параметра `[Отчетный год:]`.

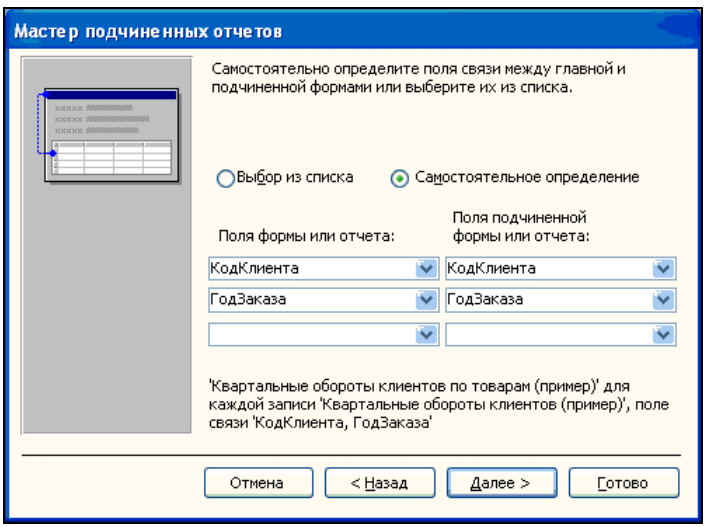


Рис. 10.10. Третье диалоговое окно Мастера подчиненных отчетов

7. Определив связь между отчетами, нажмите кнопку **Далее** (Next).
8. В последнем диалоговом окне требуется задать имя подчиненного отчета. Введите название `Подчиненный отчет Квартальные обороты (пример)` и нажмите кнопку **Готово** (Finish).

В результате в главном отчете будет создано поле **Подчиненный отчет** (Subreport). Вам потребуется только настроить высоту и ширину этого поля, удалить его метку, и вы получите отчет, аналогичный изображенному на рис. 10.11.

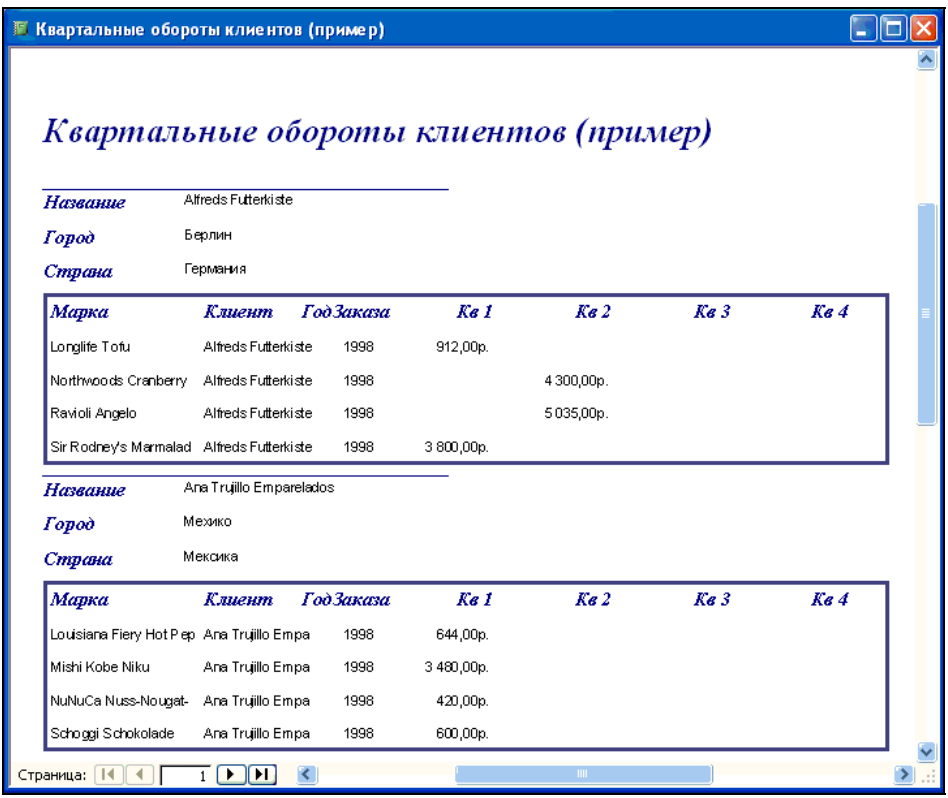


Рис. 10.11. Отчет `Квартальные обороты клиентов (пример)`

В качестве подчиненного отчета может быть использована и форма. Например, отчет, аналогичный только что описанному, можно получить с помощью уже имеющейся в базе данных `Борей` формы `Подчиненная для квартальных оборотов` (Quarterly Orders Subform). Для этого:

1. Скопируйте данную форму, задав ей новое имя `Подчиненная для квартальных оборотов (пример)`.
2. Откройте полученную копию в режиме Конструктора и замените источник данных на запрос `Квартальные обороты клиентов по товарам (пример)`, созданный для предыдущего примера. Сохраните и закройте форму.

Замечание

Источник данных для формы доступен в режиме Конструктора в окне свойств формы как свойство **Источник записей** (Record Source) на вкладке **Данные** (Data).

3. Затем скопируйте созданный ранее отчет `Квартальные обороты клиентов (пример)`, задав ему новое имя `Квартальные обороты клиентов 2 (пример)`, и откройте его в режиме Конструктора.
4. Удалите подчиненный отчет и вместо него просто перенесите с помощью мыши созданную форму из окна базы данных в область данных отчета. Автоматически будет создан элемент **Подчиненная форма/отчет** (Subform/Subreport) с именем `Подчиненная для квартальных оборотов`.
5. Теперь настройте высоту и ширину созданного элемента и в свойствах подчиненной формы на вкладке **Данные** (Data) определите связующие поля. Для этого нажмите кнопку Построителя справа от свойства **Подчиненные поля** (Link Child Fields) и задайте в появившемся диалоговом окне две пары одноименных связанных полей: `КодКлиента` (CustomerID) и `ГодЗаказа` (OrderDate), таким же образом, как это было описано в предыдущем примере. Указанные поля запишутся в качестве значений свойств **Подчиненные поля** (Link Child Fields) и **Основные поля** (Link Master Fields).
6. В результате вы получите отчет, аналогичный представленному на рис. 10.12.

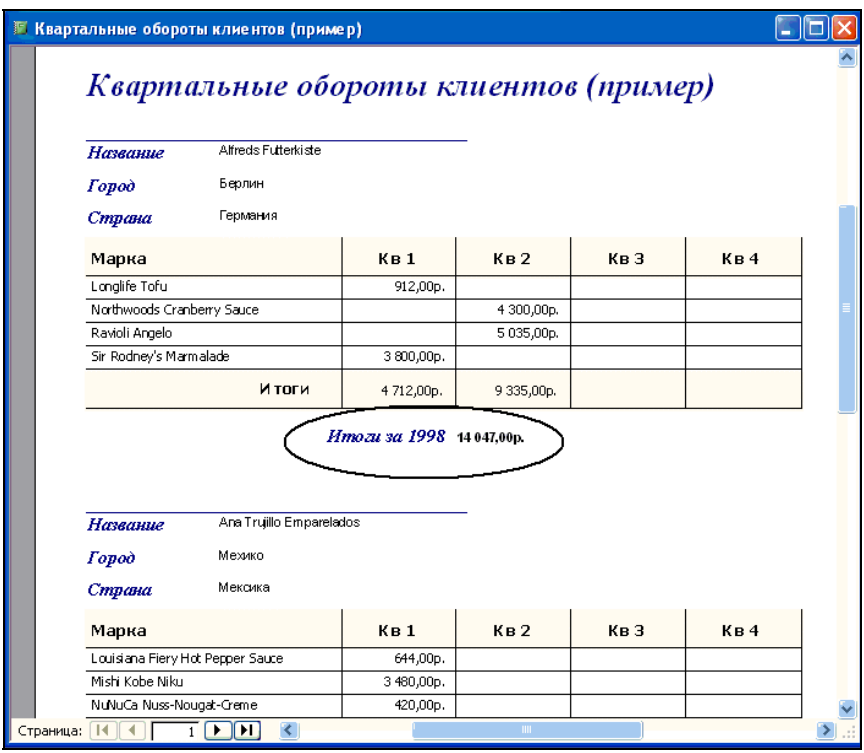


Рис. 10.12. Составной отчет, включающий подчиненную форму

Для полноты картины добавим в главный отчет два вычисляемых поля, которые будут показывать, на какую сумму клиент купил товара за год. Оба поля представляют собой текстовые поля, у которых свойство **Данные** (Control Source) задано выражениями. Первое поле формирует надпись с номером отчетного года. Для вычисления значения этого поля используется следующее выражение:

```
="Итоги за " & [Reports]![Квартальные обороты клиентов 2
(пример)]![Подчиненная для квартальных оборотов].[Report]![ГодЗаказа]
```

(для английской версии используйте название поля `OrderYear`). Второе поле содержит сумму всех заказов клиента за год и вычисляется с помощью следующего выражения:

```
=[Подчиненная для квартальных оборотов].Form![Итого]
```

(для английской версии используйте название поля `Total`), т. е. отображает итоговое поле `Итого` (Total) из подчиненной формы `Подчиненная для квартальных оборотов`. Отформатируйте созданное поле для отображения в нем денежной суммы в том же виде, как отображаются денежные суммы в подчиненной форме, скопировав значения свойств **Формат поля** (Format) и **Число десятичных знаков** (Decimal Places).

Перекрестные отчеты

Перекрестные отчеты строятся на основе перекрестных запросов и содержат итоговые данные *(см. также гл. 8)*.

В предыдущем разделе мы использовали один из таких перекрестных запросов для создания подчиненного отчета `Квартальные обороты клиентов по товарам (пример)`. Аналогично создается и обычный отчет на базе перекрестного запроса. Пример такого отчета приведен на рис. 10.13. Он создан на базе того же самого запроса `Квартальные обороты клиентов по товарам (пример)`, но отображает данные в другом разрезе: проданные товары группируются по наименованию, затем по клиентам. Создать такой отчет не сложно:

1. Создайте новый отчет в режиме Конструктора и выберите для него в качестве источника данных запрос `Квартальные обороты клиентов по товарам (пример)`.
2. Разместите надписи, содержащие заголовки столбцов отчета, в разделе **Верхний колонтитул** (Page Header), а все текстовые поля со значениями — в области данных. (Еще проще создать такой отчет с помощью автоотчета: ленточный).
3. Определите в окне **Сортировка и группировка** (Sorting and Grouping) сортировку по полям `Марка` (ProductName) и `КодКлиента` (CustomerID). Для поля `Марка` отобразите заголовок группировки, выбрав значение **Да** (Yes) в раскрывающемся списке **Заголовок группы** (Group Header). Закройте окно.
4. Перенесите текстовое поле `Марка` в отображенный заголовок группы.

Замечание

В данном запросе, когда он был создан для примера с подчиненным отчетом, не было задано параметра для выбора записей только для указанного отчетного года, т. к. записи выбирались на основании связанных данных из главного отчета. Поэтому, чтобы в создаваемом теперь отчете отображались только записи за указанный

год, определите выражение с параметром для фильтрации данных в новом отчете: `ГодЗаказа = [Отчетный год:]` (для английской версии используйте английское имя поля `OrderYear`) *(как задать фильтр, рассказано в разд. "Фильтрация данных в отчете" этой главы)*.

Отчет готов. Такой отчет называется *статическим* перекрестным отчетом, т. к. хотя его источник данных — перекрестный запрос, число столбцов в нем всегда одинаково, заголовки столбцов являются постоянными. Он создается так же, как и обычный отчет, на базе готового запроса.

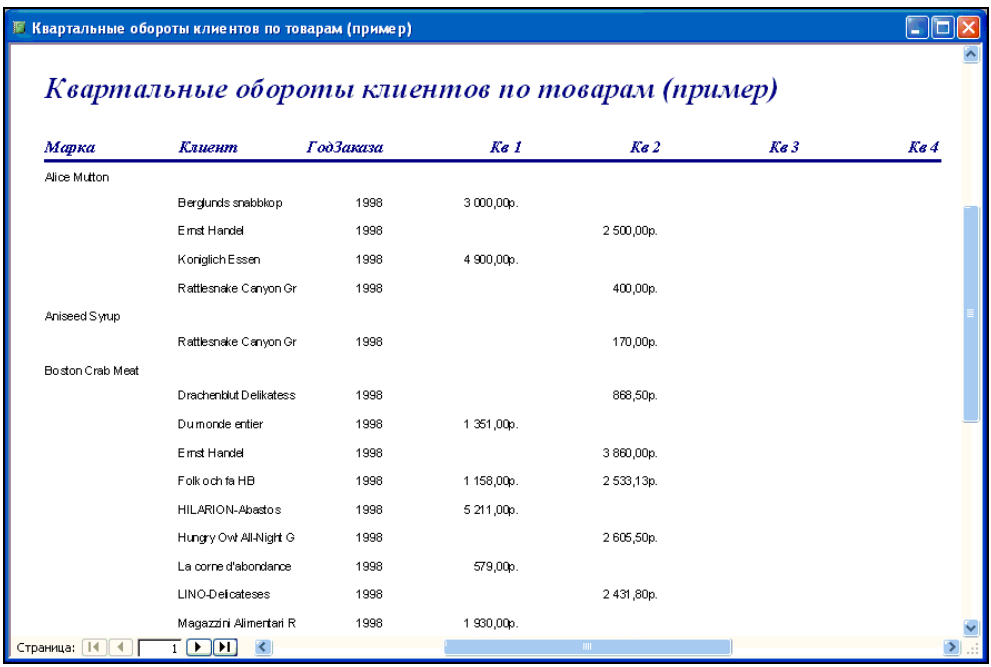


Рис. 10.13. Статический перекрестный отчет

Более сложной задачей является создание *динамического* перекрестного отчета. Такой отчет возникает, если базовый перекрестный запрос может иметь разное количество и разные заголовки столбцов. Пример такого запроса приведен на рис. 10.14. Этот отчет показывает выработку сотрудников фирмы по месяцам года. Такой отчет может создаваться ежеквартально, и каждый раз число столбцов в нем будет увеличиваться, пока в конце года не станет равным 14 (12 месяцев плюс итоговый столбец и столбец с фамилиями сотрудников).

Еще одним примером динамического перекрестного отчета является отчет, в котором показывается, какие товары вы заказали поставщикам, но еще не получили. Такой отчет может представлять собой таблицу, в которой по вертикали отображаются товары, а по горизонтали поставщики, которым эти товары заказаны. Его можно построить, если число поставщиков ограничено и не очень велико.



Фамилия	январь	февраль	март	апрель	май	Итого
Акбаев	55677	21459	56065	38615	0	171816
Бабкина	236387	261236	120353	160787	29936	808698
Белова	83941	72349	107220	294370	47773	605652
Воронова	62872	251571	87509	103762	5916	511630
Кралев	72540	63171	61864	213283	34741	445598
Кротов	90998	79801	24020	2100	0	196919
Крылова	107925	27352	135031	161471	45500	477279
Новиков	69237	170365	129224	362359	6718	737902
Ясенева	56938	204183	54011	92275	14020	421426
Итого:	836515	1151487	775294	1429020	184602	4376920

Рис. 10.14. Динамический перекрестный отчет

Для создания динамических отчетов не обойтись без программирования. В режиме Конструктора отчет о выработке сотрудников выглядит так, как это представлено на рис. 10.15, т. е. он содержит несвязанные с данными надписи и текстовые поля. Их привязка к результатам запроса выполняется программным путем в процессе выполнения отчета. Позже мы покажем, как создать программы для такого отчета *(см. разд. "Создание перекрестного отчета с изменяющимся числом столбцов" гл. 13)*.

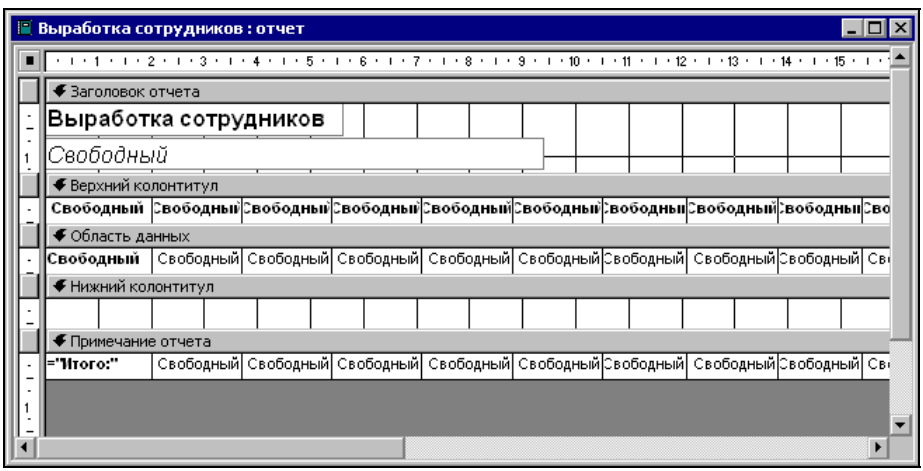


Рис. 10.15. Динамический перекрестный отчет в режиме Конструктора

Печать отчета в несколько колонок

Иногда требуется отобразить данные на листе бумаги в несколько колонок. Это может потребоваться для компактного представления данных в отчете. Пример такого отчета приведен на рис. 10.16 — это отчет `Товары по типам` (Products by Category) из базы данных `Борей` (Northwind).

В нем содержатся данные о количестве товаров на складе. Данные о товарах группируются по категориям товаров и отображаются в три колонки.

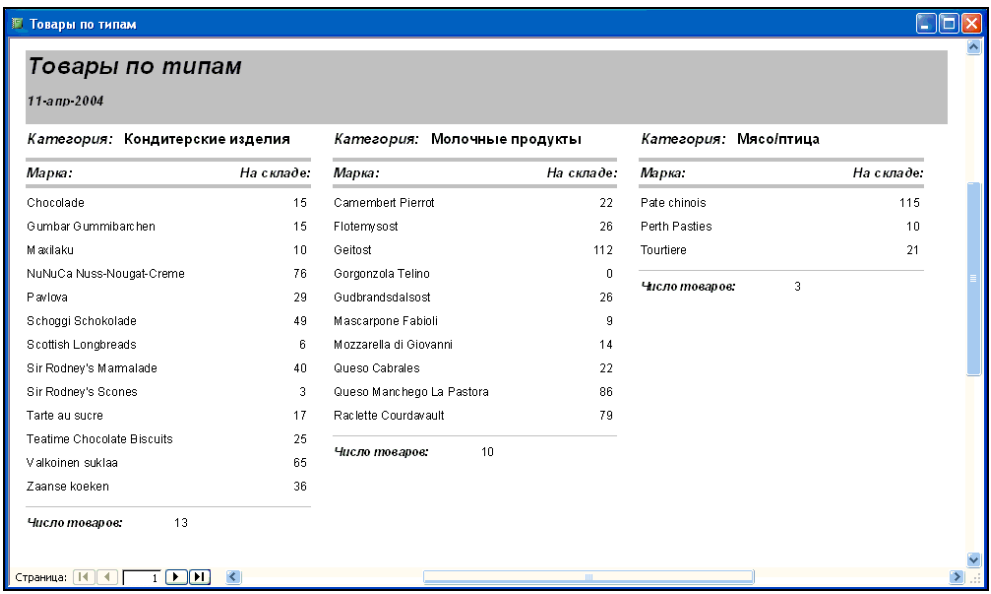


Рис. 10.16. Отчет в три колонки

Для того чтобы получить такой отчет, нужно задать определенным образом параметры печати отчета. Эти параметры задаются в диалоговом окне **Параметры страницы** (Page Setup) (рис. 10.17). Для того чтобы вывести это окно, в режиме Конструктора отчета выберите команду меню **Файл | Параметры страницы** (File | Page Setup). На вкладке **Поля** (Margins) определите ширину полей страницы. Затем раскройте вкладку **Столбцы** (Columns).

Группа полей **Параметры сетки** (Grid Setting) позволяет определить количество колонок на листе (поле **число столбцов** (Number of Columns)) и расстояние между колонками (поле **столбцов** (Column Spacing)). Поле **интервал строк** (Row Spacing) используется при печати наклеек на конверты.

Группа полей **Размер столбца** (Column Size) позволяет определить ширину колонки. Если установить флажок **по размеру данных** (Same As Detail), то поля **ширина** (Width) и **высота** (Height) будут установлены равными ширине и высоте раздела **Область данных** (Detail) отчета. Если этот флажок сброшен, то значение поля **ширина** (Width) будет определять ширину колонки (поле **высота** (Height) используется для определения размера наклеек).

Переключатели в группе **Макет столбца** (Column Layout) определяют ориентацию колонок. В данном случае выбран переключатель **сверху вниз** (Down, then Across), который обеспечивает печать данных в виде газетных полос.

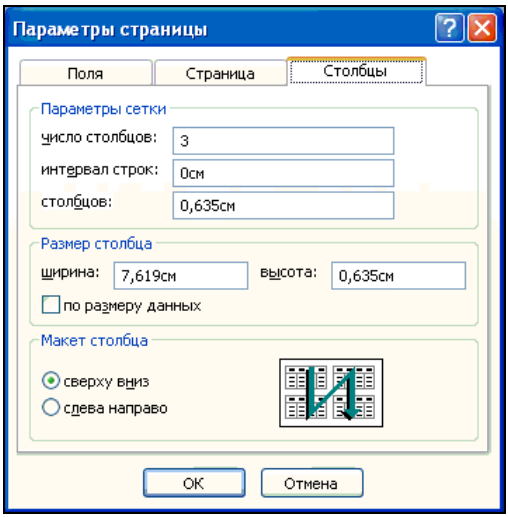


Рис. 10.17. Диалоговое окно **Параметры страницы**

Сортировка и группировка данных

В отличие от форм отчеты содержат, как правило, большое количество записей, которые для удобства просмотра и анализа должны быть отсортированы и объединены в группы. Рассмотрим возможности, которые предоставляет Access для группировки и сортировки записей в отчетах. Для этого создадим отчет `Заказы клиентов (пример)`, в котором сгруппируем все заказы по клиентам.

Создание базового запроса

Для этого сначала создадим базовый запрос. Объединим в этом запросе таблицы `Клиенты` (Customers) и `Заказы` (Orders) и запрос `Сведения о заказах` (Order Details Extended), как это показано на рис. 10.18.

Включите в запрос следующие поля:

- из таблицы `Клиенты` (Customers) — `Название` (CompanyName), `Страна` (Country), `Город` (City);
- из таблицы `Заказы` (Orders) — `КодЗаказа` (OrderID), `ДатаРазмещения` (OrderDate), `ДатаНазначения` (RequiredDate), `ДатаИсполнения` (ShippedDate);
- из запроса `Сведения о заказах` (Order Details Extended) — `Марка` (ProductName), `Цена` (UnitPrice), `Количество` (Quantity), `Скидка` (Discount), `ОтпускнаяЦена` (ExtendedPrice).

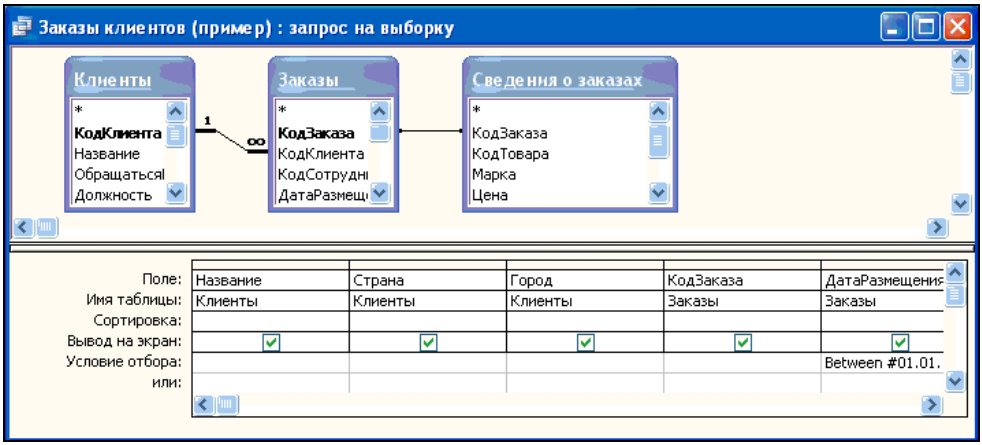


Рис. 10.18. Базовый запрос для отчета `Заказы клиентов (пример)`

Ограничим количество записей в выборке, введя для поля `ДатаРазмещения` (OrderDate) **Условие отбора** (Criteria): `Between #01.01.98# And #31.12.98#`. Сохраните созданный запрос с именем `Заказы клиентов (пример)`.

Теперь создайте новый отчет на базе запроса `Заказы клиентов (пример)`. Для этого выделите этот запрос в списке запросов окна базы данных и нажмите стрелку на кнопке **Новый объект** (New Object) на панели инструментов. Выберите из раскрывающегося списка значение **Отчет** (Report) и затем в диалоговом окне **Новый отчет** (New Report) выберите значение **Конструктор** (Design View).

Определение условий сортировки и группировки

Теперь определим структуру отчета, задавая условия группировки и сортировки записей. Эти условия задаются в диалоговом окне **Сортировка и группировка** (Sorting and Grouping), которое появляется при нажатии кнопки с таким же названием на панели инструментов.

Чтобы создать группу, нужно задать поле отчета, по которому будут группироваться записи, и установить значения свойств **Заголовок группы** (Group Header) и **Примечание группы** (Group Footer) равными **Да** (Yes). Первый уровень группировки в нашем примере будет определяться полем `Название` (CompanyName) клиента. Выберите это поле из раскрывающегося списка в первой строке окна **Сортировка и группировка** (Sorting and Grouping). Поле справа в этой строке, определяющее порядок сортировки, будет по умолчанию иметь значение **По возрастанию** (Ascending). Установите значения свойств **Заголовок группы** (Group Header) и **Примечание группы** (Group Footer) равными **Да** (Yes). После этого в области выделения первой строки (в окне) появится символ группировки, а в отчете появятся два раздела: **Заголовок группы 'Название'** (CompanyName Header) и **Примечание группы 'Название'** (CompanyName Footer).

Второй уровень группировки будет организован по полю `КодЗаказа` (OrderID). Заполните аналогичным образом его свойства в окне **Сортировка и группировка** (Sorting and Grouping).

Кроме того, что мы задали два уровня группировки для нашего отчета, можно еще определить порядок сортировки в пределах самой внутренней группы. В третьей строке окна **Сортировка и группировка** (Sorting and Grouping) выберите поле `Марка` (ProductName), чтобы отсортировать товары в заказе по алфавиту. После этого отчет примет вид, показанный на рис. 10.19.

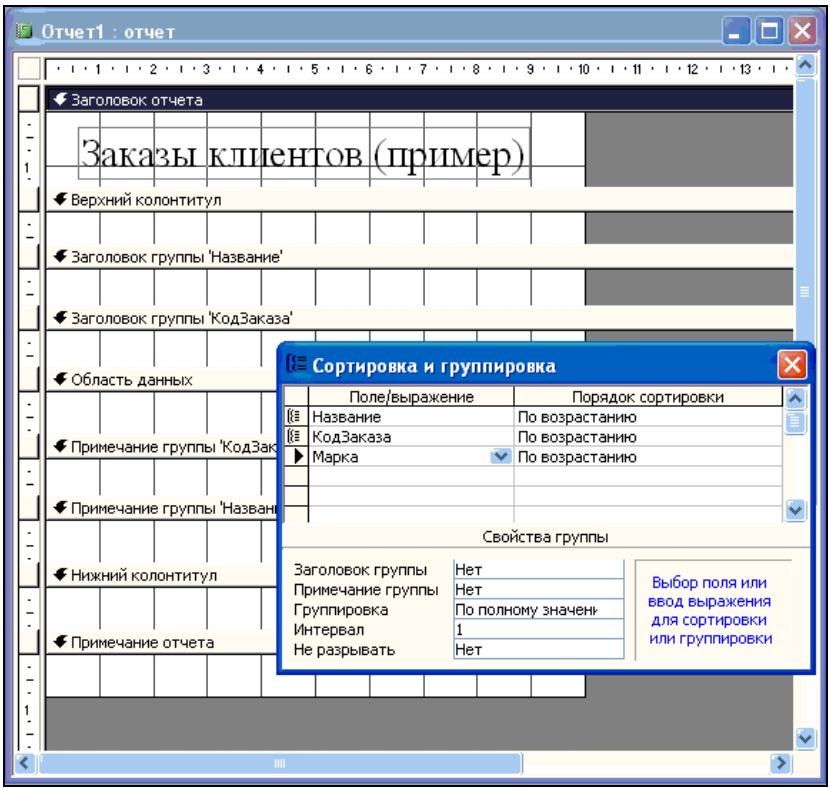


Рис. 10.19. Структура отчета, а также условия сортировки и группировки

Уровень группировки по полю `КодЗаказа` оказался вложенным. И таких уровней вложенности может быть до 10.

Для каждой группы кроме свойств **Заголовок группы** (Group Header) и **Примечание группы** (Group Footer), которые позволяют вставлять в отчет дополнительные разделы, можно задать еще три свойства: **Группировка** (Group On), **Интервал** (Group Interval) и **Не разрывать** (Keep Together).

Свойства **Группировка** (Group On) и **Интервал** (Group Interval) связаны между собой и в совокупности определяют, каким образом мы хотим группировать данные. Значения, которые может принимать свойство **Группировка** (Group On), зависят от типа данных группирующего поля. Обычно данные группируются по каждому значению поля, т. е. каждому значению поля должна соответствовать отдельная группа. Тогда свойство **Группировка** (Group On) должно иметь значение **По полному значению**

(Each Value), а значение свойства **Интервал** (Group Interval) должно быть равным 1. В нашем примере для поля `Название` установите эти свойства именно так.

При группировке по полю с текстовым содержимым допускается начинать новую группу всякий раз, когда изменится первый или несколько первых символов значения поля. В этом случае свойство **Группировка** (Group On) должно иметь значение **По первым знакам** (Prefix Characters), а свойство **Интервал** (Group Interval) будет определять, сколько первых символов должно использоваться для образования новой группы. Если это значение равно 1, то для каждой буквы алфавита будет создана отдельная группа — это позволяет создавать отчеты типа предметного указателя в книгах. Примером такого отчета является отчет `Список товаров` (Alphabetical List of Products) в базе данных `Борей` (Northwind).

Для числового поля `КодЗаказа` (OrderID) мы оставляем значение свойства **Группировка** (Group On) равным **По полному значению** (Each Value), как это установлено по умолчанию. Значение свойства **Интервал** (Group Interval) останется равным `1`, т. к. нам нужно создать отдельную группу для каждого заказа. Однако свойство **Группировка** (Group On) позволяет группировать числовые данные и по диапазонам значений. В этом случае нужно установить значение этого свойства равным **Интервал** (Interval), а значение свойства **Интервал** (Group Interval) равным числу, которое определяет диапазон группировки. Например, если числовое поле содержит возраст человека, то значение `5` свойства **Интервал** (Group Interval) будет определять диапазоны возрастов от 0 до 4, от 5 до 9, от 10 до 14 и т. д.

Для поля типа Дата/Время свойство **Группировка** (Group On), кроме значения по умолчанию **По полному значению** (Each Value), может принимать следующие значения: **По годам** (Year), **По кварталам** (Qtr), **По месяцам** (Month), **По неделям** (Week), **По дням** (Day), **По часам** (Hour), **По минутам** (Minute). Значение свойства **Интервал** (Group Interval) определяет число выбранных единиц измерения в диапазоне. Например, если значение свойства **Группировка** (Group On) установлено равным **По месяцам**, а **Интервал** (Group Interval) имеет значение `1`, то данные будут группироваться помесячно.

Свойство **Не разрывать** (Keep Together) позволяет управлять тем, как группа будет размещаться на странице. Если группы небольшие, то лучше всего установить значение этого свойства равным **Вся группа** (Whole Group). Тогда при печати отчета вся группа, т. е. заголовок группы, все записи из области данных и примечание группы будут размещаться на одной странице (если вся группа не помещается на текущую страницу, она продолжится со следующей страницы), а каждая следующая группа будет начинаться с новой страницы (если только она не помещается целиком на текущей странице). Если группа может содержать столько данных, что они не поместятся на одной странице, то лучше установить значение этого свойства равным **С первыми данными** (With First Detail). В этом случае заголовок группы размещается на текущей странице, только если на ней помещается также первая строка данных, в противном случае вся группа переносится на следующую страницу.

Свойство **Не разрывать** (Keep Together) для группы связано с аналогичным свойством раздела отчета. Эти свойства должны быть согласованы. Например, если в группе установлено значение свойства **Не разрывать** (Keep Together) равным **Вся группа** (Whole Group), то значения свойства **Не разрывать** (Keep Together) соответствующих разделов отчета должны иметь значения равными **Да** (Yes). В противном случае

приоритет имеет свойство, заданное в окне **Сортировка и группировка** (Sorting and Grouping).

Сам отчет также имеет свойство **Неразрывная группа** (Grp Keep Together), которое применяется для отчетов, печатающихся в несколько колонок. Оно действует только для групп, у которых значение свойства **Не разрывать** (Keep Together) установлено равным **Вся группа** (Whole Group) или **С первыми данными** (With First Detail). Если же установлено значение **Нет** (No), то свойство **Неразрывная группа** (Grp Keep Together) игнорируется. Данное свойство отчета может иметь значения **На странице** (Per Page) и **В столбце** (Per Column). В первом случае группа печатается без разрыва на странице, а во втором — без разрыва в колонке (полосе) отчета.

Замечание

В ячейки **Поле/выражение** (Field/Expression) окна **Сортировка и группировка** (Sorting and Grouping) можно вводить не только названия полей, но и выражения, т. е. группировка может выполняться по значению выражения. Ввод выражения начинается со знака равенства (=), так же как в свойствах форм и отчетов.

Определение содержимого отчета

После того как мы создали структуру отчета, можно начать помещать поля в разделы отчета.

Выберите команду **Файл | Параметры страницы** (File | Page Setup) и установите в диалоговом окне поля отчета так: верхнее, нижнее и левое — размером 2,5 см, правое — 1 см (в полях диалогового окна эти значения задаются в миллиметрах). Затем закройте окно **Параметры страницы** (Page Setup) и перетащите правую границу области данных вправо, установив ширину отчета 17 см.

В разделе **Заголовок отчета** (Report Header) создайте надпись `Заказы клиентов (пример)`. Установите шрифт надписи **Times New Roman Cyr** размером 21 пункт. Над надписью добавьте горизонтальную линию шириной 2 пункта от левого до правого края отчета. Высота раздела заголовка отчета в нашем примере составляет 1,6 см.

Замечание

Если раздел **Заголовок отчета** (Report Header) отсутствует в окне Конструктора, выберите команду **Вид | Заголовок/примечание отчета** (View | Report Header /Footer).

Раздел **Верхний колонтитул** (Page Header) оставляем пустым и устанавливаем его высоту равной `0`.

Оформим раздел заголовка группы `Название` (CompanyName Header). Откройте список полей и перетащите из него в раздел заголовка группы следующие поля: `Название` (CompanyName), `Страна` (Country) и `Город` (City). Удалите присоединенные к полям надписи и установите для всех полей шрифт Arial Cyr размером 14 пунктов и прозрачный фон. Выделите область этого раздела, разместив элемент **Прямоугольник** (Rectangle), имеющий цвет фона серый и ширину границы 2 пункта, позади текстовых полей (рис. 10.20). Сократите высоту раздела так, чтобы его нижний край совпадал с нижней границей прямоугольника.

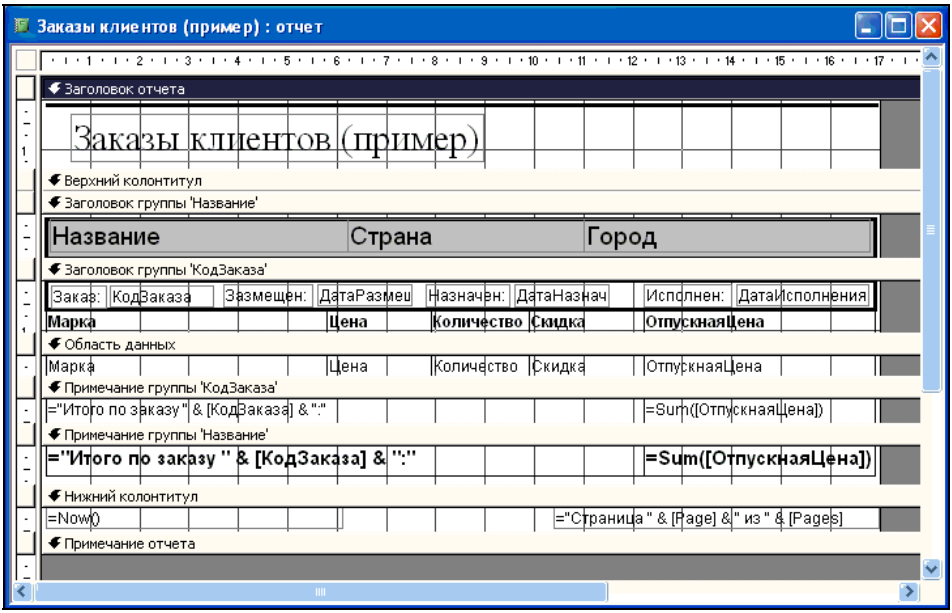


Рис. 10.20. Отчет `Заказы клиентов (пример)` в режиме Конструктора

Прежде чем размещать остальные поля, определим некоторые стандартные свойства этих полей (т. е. свойства, которые будут задаваться по умолчанию для каждого нового создаваемого в отчете текстового поля). Для этого нажмите кнопку **Поле** (Text Box) на Панели элементов (Toolbox) и нажмите кнопку **Свойства** (Properties) на панели инструментов **Конструктор отчетов** (Report Design). Откроется окно **"Поле" — стандартные свойства** (Default Text Box). Задайте следующие значения свойств: **Шрифт** (Font Name) — **Arial Cyr**, **Размер шрифта** (Font Size) — `9`, **Насыщенность** (Font Weight) — **обычный** (Normal), **Добавление подписи** (Auto Label) — **Да** (Yes), **Подписи с двоеточием** (Add Colon) — **Нет** (No). Закройте окно свойств поля. Затем нажмите кнопку **Надпись** (Label) и установите такие же значения для свойств, определяющих шрифт надписи.

Перетащите из списка полей в раздел заголовка группы `КодЗаказа` (OrderID Header) следующие поля: `КодЗаказа` (OrderID), `ДатаРазмещения` (OrderDate), `ДатаНазначения` (RequiredDate) и `ДатаИсполнения` (ShippedDate). Измените надписи для полей так, как это показано на рис. 10.20. Выделите эти поля, разместив позади них прямоугольник, имеющий ширину границы 2 пункта и тип фона **Прозрачный** (Transparent).

Измените стандартное значение свойства **Тип границы** (Border Style) для надписей и текстовых полей, присвоив им значение **Сплошная** (Solid), т. к. все остальные надписи и поля будут окружены рамками.

Далее перетащите в Область данных (Detail) следующие поля: `Марка` (ProductName), `Цена` (UnitPrice), `Количество` (Quantity), `Скидка` (Discount) и `ОтпускнаяЦена` (ExtendedPrice). Теперь нужно переместить присоединенные к ним надписи в область заголовка группы `КодЗаказа`. Для этого выделите все присоединенные надписи для этих полей и удалите их в буфер обмена (например, с помощью комбинации

клавиш <Ctrl>+<X>). Затем выделите раздел **Заголовок группы 'КодЗаказа'** и вставьте в него надписи из буфера обмена (<Ctrl>+<V>). Расположите эти надписи под прямоугольником так, чтобы нижняя граница прямоугольника и верхняя граница рамки надписей совпали. Измените для всех надписей свойство шрифта **Насыщенность** (Font Weight) на **жирный** (Bold) и распределите надписи по ширине отчета так, как это показано на рис. 10.20. Рамки надписей должны плотно примыкать друг к другу. Подтяните нижний край раздела заголовка группы так, чтобы он совпал с нижней границей надписей.

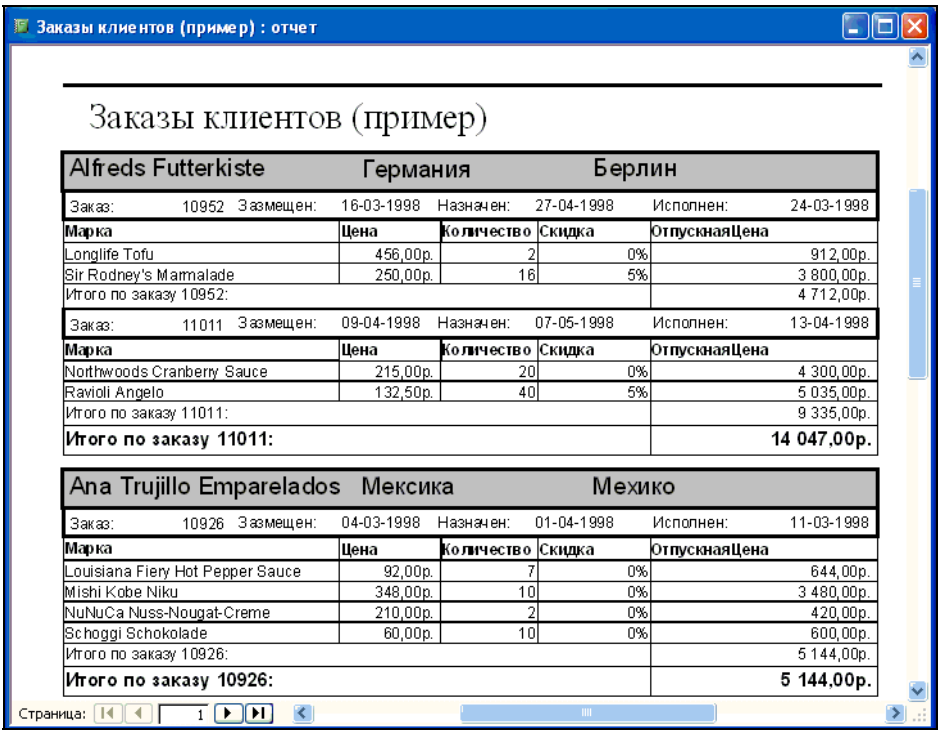


Рис. 10.21. Отчет `Заказы клиентов (пример)` в режиме Предварительного просмотра

Разместите поля в области данных так, чтобы они располагались вдоль верхнего края раздела точно под соответствующими надписями. Рамки полей, так же как и рамки надписей, должны плотно примыкать друг к другу. Подтяните нижний край области данных так, чтобы он совпал с нижней границей полей.

В разделе примечания группы `КодЗаказа` (OrderID Footer) разместите два вычисляемых текстовых поля. Одно из них будет содержать текст, который определяется выражением:

```
="Итого по заказу " & [КодЗаказа] & ":"
```

(для английской версии используйте имя поля `OrderID`). Второе поле должно содержать итоговую сумму заказа и вычисляться с помощью выражения:

```
=Sum([ОтпускнаяЦена])
```

(для английской версии используйте имя поля `ExtendedPrice`). У этих полей в нашем примере отсутствуют присоединенные надписи (первое поле играет роль надписи для второго). Поэтому перед их размещением можно соответствующим образом изменить стандартное свойство **Добавление подписи** (Auto Label) для текстового поля. Эти поля также имеют рамки, ширина которых должна быть установлена так, чтобы они примыкали друг к другу у левой границы поля `ОтпускнаяЦена`, находящегося справа в области данных.

В разделе примечания группы `Название` (CompanyName Footer) выведем итоги продаж по клиенту в целом (по всем его заказам). Эти поля формируются аналогично итоговым полям в группе `КодЗаказа`, т. е. вычисляются согласно выражениям:

- для первого поля: `="Итого по клиенту " & [Марка] & ":"`
- для второго поля: `=Sum([ОтпускнаяЦена])`

(для английской версии используйте имена полей `ProductName` и `ExtendedPrice`, соответственно). Чтобы выделить значения этих полей, увеличьте размер шрифта до 11 пунктов и установите насыщенность шрифта — **жирный** (Bold).

Для созданных полей с итоговыми суммами задайте для свойства **Формат поля** (Format) значение **Денежный** (Currency).

В итоге вы должны получить отчет, который будет выглядеть при печати так, как представлено на рис. 10.21.

Настройка печати групп отчета

Кроме тех свойств групп отчета, которые обсуждались ранее, существуют еще несколько свойств разделов отчета, которые влияют на представление групп в отчетах. Теперь мы опишем эти свойства.

Расположение групп в отчетах, имеющих несколько колонок

При печати отчета в несколько колонок (полос) на любом уровне группировки можно указать, чтобы каждая группа этого уровня всегда начиналась в новой колонке. Примером такого отчета является отчет `Товары по типам` (Products by Category) демонстрационной базы данных `Борей`.

Для регулировки расположения групп на странице такого отчета используется свойство **Новая строка или столбец** (New Row or Col) раздела заголовка или примечания группы отчета. Установив это свойство равным **До раздела** (Before Section) для заголовка группы, можно заставить Access печатать каждую группу этого уровня группировки в новой колонке или новой полосе. Если значение свойства **Новая строка или столбец** (New Row or Col) установить равным **После раздела** (After Section), то данный раздел отчета печатается в текущей колонке или полосе, а следующий начинается в новой. И наконец, если значение этого свойства для заголовка группы установить равным **До и после раздела** (Before & After), то заголовок группы будет печататься в отдельной колонке или полосе, т. к. следующий за ним раздел данных тоже начнет печататься в новой колонке или полосе.

Повторение заголовка группы на каждой странице или колонке

Если группа занимает при печати больше одной страницы или переносится на следующую страницу, можно заставить Access печатать заголовок группы на каждой

странице. Для этого используется свойство **Повторение раздела** (Repeat Section) заголовка группы. По умолчанию это свойство имеет значение **Нет** (No).

Скрытие повторяющихся данных в отчете

При составлении простых отчетов, содержащих отсортированные данные, могут возникать ситуации, когда в нескольких строках отчета в одном поле повторяются одни и те же данные, например, наименование клиента, адрес и пр. Очевидно, что отчет будет выглядеть лучше, если данные в поле не будут повторяться, а будут печататься каждый раз только в первой строке. Каждое поле в области данных отчета имеет свойство **Не выводить повторы** (Hide Dublicates). Если установить это свойство равным **Да** (Yes) для поля, содержащего повторяющие данные, то данные повторяться не будут. Для сравнения на рис. 10.22 и 10.23 приводится один и тот же отчет `Список заказов (пример)` с разными значениями свойства **Не выводить повторы** (Hide Dublicates) для поля `Название` (CompanyName).

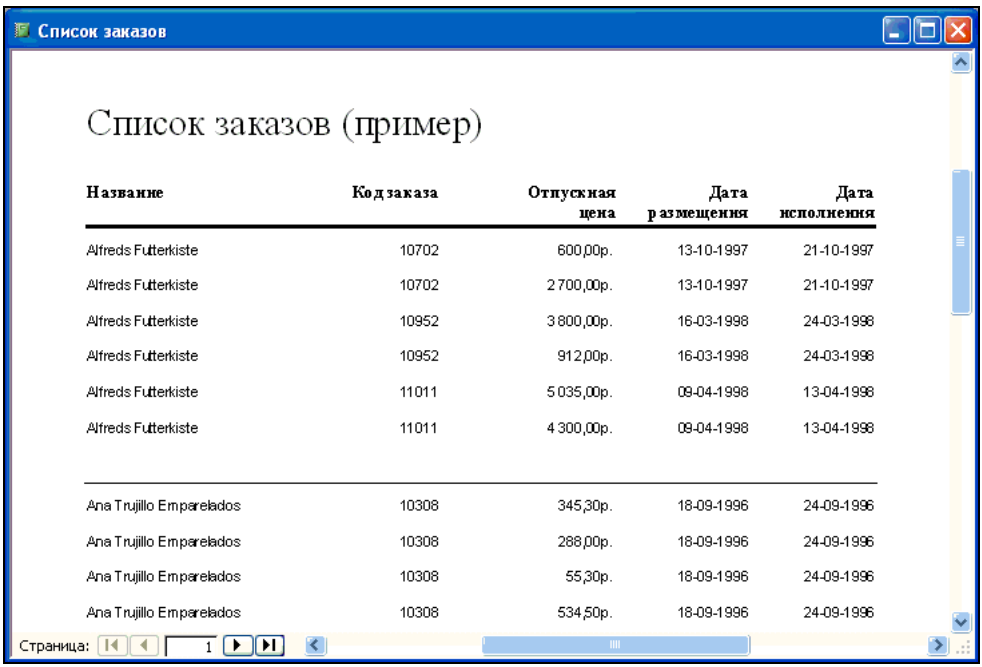


Рис. 10.22. Свойство **Не выводить повторы** поля `Название` равно **Нет**

Если в отчете используется группировка, то не всегда группирующее поле выносится в отдельный раздел (заголовок или примечание группы). Иногда его оставляют в области данных отчета, вынося итоговые значения в раздел примечания группы. В этом случае также используется свойство **Не выводить повторы** (Hide Dublicates) для группирующего поля, чтобы не выводить повторяющиеся значения. Отчет при этом будет выглядеть так, как это показано на рис. 10.24.

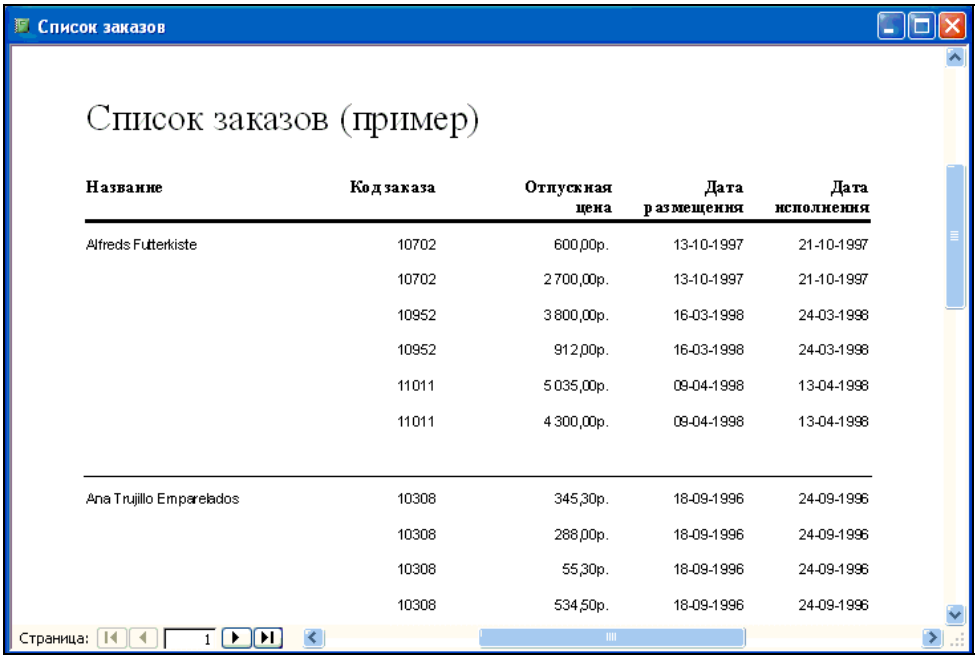


Рис. 10.23. Свойство **Не выводить повторы** поля `Название` равно **Да**

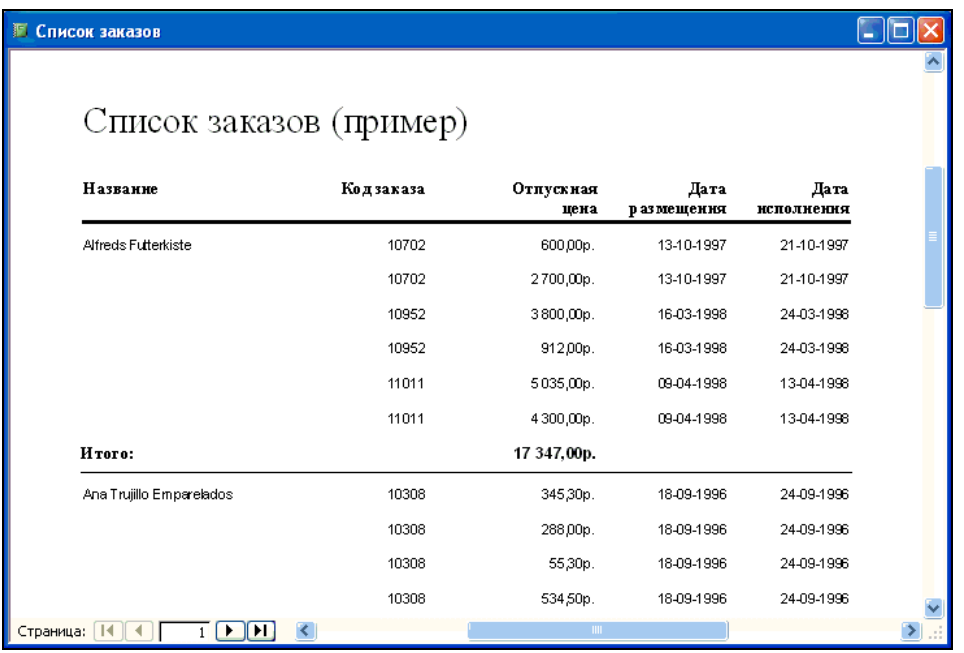


Рис. 10.24. Помещение группирующего поля в область данных отчета

Проведение вычислений

В *гл. 6* и в предыдущих разделах данной главы мы создавали вычисляемые поля в отчетах главным образом для вывода итоговых значений. Сейчас мы покажем еще несколько типов вычисляемых полей, которые применяются для решения типовых задач.

Создание полей, содержащих сумму с накоплением

Часто в отчетах требуется создать поле, в котором суммируются значения из всех предыдущих записей. Такое поле представляет собой специальный вид итогового значения. Для того чтобы текстовое поле накапливало данные из предыдущих записей, нужно установить для него значение свойства **Сумма с накоплением** (Running Sum) равным либо значению **Для группы** (Over Group), либо **Для всего** (Over All).

Значение **Для группы** (Over Group) используется, когда нужно просуммировать значение в пределах одной группы, в следующей группе это значение сбрасывается в 0 и суммирование начинается вновь.

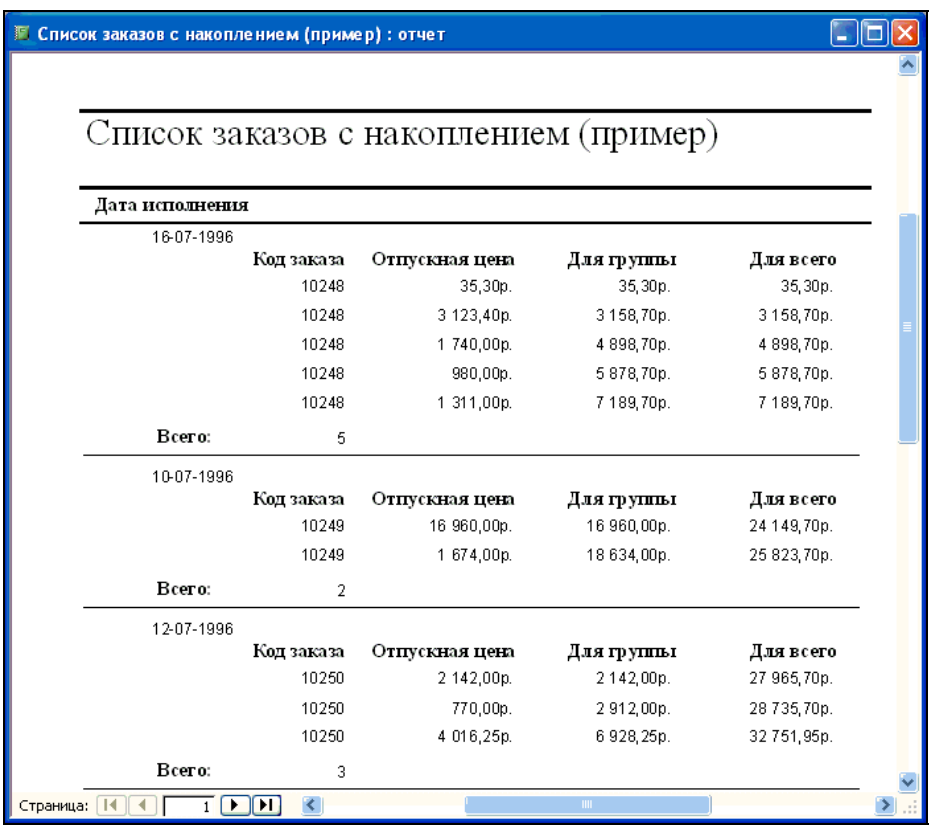


Рис. 10.25. Использование полей с накоплением суммы

Значение **Для всего** (Over All) используется либо в том случае, когда в отчете нет группировки, либо когда нужно получить итоговое значение по всем группам.

Такое накапливающее поле может быть размещено либо в области данных, и тогда оно будет суммировать данные в каждой записи, либо в области заголовка или примечания группы, тогда оно будет накапливать итоговые значения для всех групп одного уровня. Пример отчета, содержащего поля с накоплением, приведен на рис. 10.25. Первое вычисляемое поле (с названием `Для группы`) в этом отчете содержит выражение: `=[ОтпускнаяЦена]` (для английской версии `=[ExtendedPrice]`) и имеет свойство **Сумма с накоплением** (Running Sum), равное значению **Для группы** (Over Group), второе (с названием `Для всего`) содержит такое же выражение, но имеет значение данного свойства, равное значению **Для всего** (Over All).

Вычисление процентов

Иногда в отчете требуется вывести проценты, которые составляют конкретные значения в общей или частичной итоговой сумме. В отчетах Access можно создавать такие поля. Для этого в области данных создается вычисляемое поле, значение которого определяется выражением типа: `=[Стоимость]/[Общая стоимость]`, т. е. нужно меньшее значение разделить на большее. Свойство **Формат** (Format) для этого поля нужно установить равным **Процентный** (Percent).

Если требуется вычислить процент частичной (групповой) суммы от общей суммы или суммы более высокого уровня группировки, нужно создать аналогичное поле в разделе заголовка или примечания соответствующей группы.

Access обеспечивает возможность таких расчетов за счет того, что отчет формируется в два прохода. При первом проходе вычисляются поля в области данных, итоговые значения для групп и для отчета в целом и определяется длина отчета. Во время второго прохода вычисляются все выражения, которые ссылаются на итоги, рассчитанные во время первого прохода.

Пример использования полей, содержащих процент от итогового значения, вы можете найти в отчете `Продажи по сотрудникам и странам` (Employee Sales by Country) базы данных `Борей` (Northwind).

Нумерация записей и подсчет количества записей в отчете

Свойство **Сумма с накоплением** (Running Sum), которое мы рассматривали ранее, может быть использовано еще одним несколько неожиданным способом. Оно используется для нумерации записей (строк) в отчете. Если форма отчета требует, чтобы строки отчета были пронумерованы, то для этого придется создать вычисляемое поле, свойство **Данные** (Control Source) которого определяется выражением: `=1`, а свойство **Сумма с накоплением** (Running Sum) нужно установить равным **Для всего** (Over All) или **Для группы** (Over Group), если требуется нумерация записей в пределах группы. Тогда при печати отчета значение этого поля будет увеличиваться на 1 для каждой новой записи. Пример нумерации записей описанным образом вы можете увидеть в отчете `Продажи по годам` (Sales by Year) базы данных `Борей` (Northwind).

Иногда в отчете требуется сосчитать количество записей, содержащихся в каждой группе, например, количество заказов у каждого клиента или количество счетов, выписанных сотрудником отдела продаж. Такое значение, так же как и итоговая сумма, обычно выводится в примечании группы. Для того чтобы его отобразить, нужно в области данных создать вычисляемое поле, которое будет содержать номер строки (как это описано ранее), и сделать это поле невидимым, т. е. свойству **Вывод на экран** (Visible) для него присвоить значение **Нет** (No). Затем в области примечания группы нужно создать вычисляемое поле, свойство **Данные** (Control Source) которого должно ссылаться на это невидимое поле. Пример таких полей приведен на рис. 10.26.

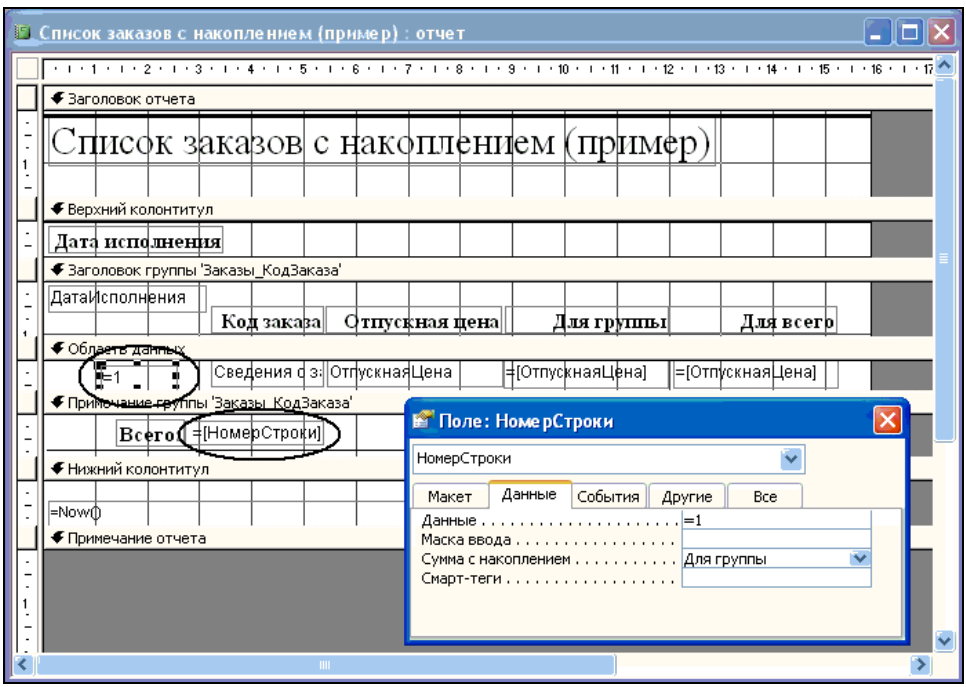


Рис. 10.26. Подсчет числа записей в группе

Если требуется подсчитать общее количество записей в отчете, то тогда просто создается одно вычисляемое поле в разделе заголовка или примечания отчета, у которого свойство **Данные** (Control Source) устанавливается равным выражению `=Count(*)`.

Объединение текстовых данных и печать условий отбора записей в отчете

В отчетах чаще, чем в формах, применяются вычисляемые поля, которые представляют собой комбинацию данных из различных полей таблиц или базовых запросов. Типичным примером такого объединения является поле, содержащее фамилию и

инициалы человека, хотя в базе данных часто фамилия, имя и отчество хранятся в раздельных полях. Второй, не менее распространенный случай, когда в одном столбце отчета требуется вывести адрес, который обычно тоже хранится в нескольких отдельных полях таблицы. Часто в одном столбце отчета объединяют данные из нескольких полей таблицы в целях уменьшения ширины отчета. Примеры создания текстовых выражений такого типа приводились ранее *(см. гл. 4, 6)*.

Еще один типичный случай объединения текстовых полей использовался нами в отчете `Заказы клиентов` (см. рис. 10.21). Это поля, которые выглядят в отчете как надписи (**Итого по заказу <*номер*>:**, **Итого по клиенту <*имя*>:**), хотя на самом деле представляют собой комбинацию текстовых строк и значений полей базового запроса.

Иногда в отчетах нужно вывести условия отбора записей. Для этого также применяют вычисляемые поля, содержащие комбинацию текста и параметров отбора. Пример такого выражения показан на рис. 10.27: в заголовке отчета создано поле, которое вычисляется с помощью выражения, содержащего ссылки на параметры базового запроса `[Начальная дата]` и `[Конечная дата]`. В этом же отчете используются и другие вычисляемые поля, объединяющие текстовые данные из разных полей.

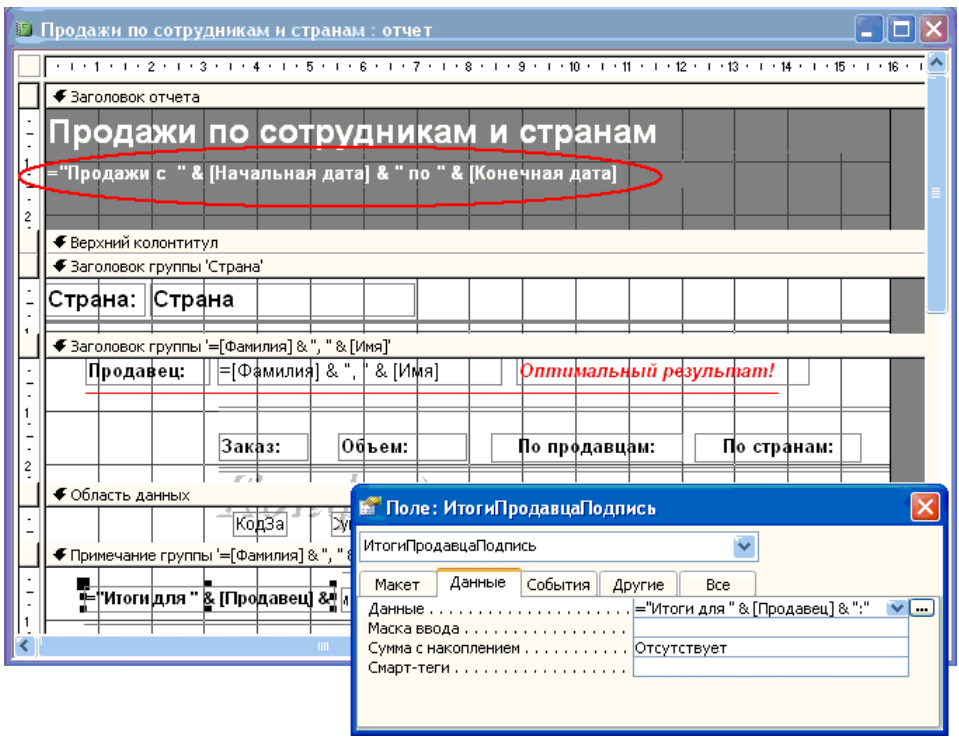


Рис. 10.27. Объединение текстовых строк и полей запроса в отчетах

Аналогично можно написать выражение, ссылающееся на значения полей открытой формы, которая часто используется для запроса критериев отбора записей. Приме-

ром такой формы, представляющей собой диалоговое окно, может быть форма Продажи по годам (Sales by Year Dialog), изображенная на рис. 10.28.

Для того чтобы в заголовке отчета написать интервал дат, заданных в форме Продажи по годам, нужно для источника данных вычисляемого поля использовать выражение:

```
="Продажи с  " & Forms![Продажи по годам]![НачальнаяДата] & "
по " & Forms![Продажи по годам]![КонечнаяДата]
```

для английской версии:

```
="Продажи с  " & Forms![Sales by Year Dialog]![BeginningDate] & " по " &
Forms![Sales by Year Dialog]![EndingDate]
```

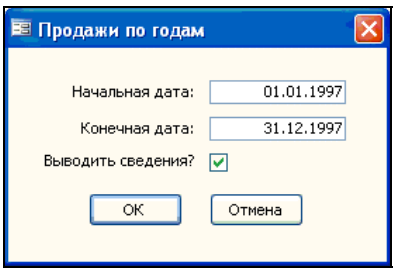


Рис. 10.28. Диалоговое окно для ввода критериев отбора записей

Замечание

Чтобы значения таких выражений были правильными, требуется, чтобы форма, на которую они ссылаются (в данном случае, Продажи по годам), была открыта в момент загрузки отчета. В противном случае в этих вычисляемых полях будет отображен код ошибки.

Совет

Если захотите поэкспериментировать с формой Продажи по годам, советуем вам временно изменить свойство **Модальное окно** (Modal) этой формы на **Нет** (No), чтобы иметь возможность, в то время как форма открыта, переключиться на отчет, содержащий вычисляемые поля.

Фильтрация данных в отчете

Записи в отчетах, так же как и в формах, могут фильтроваться. Для этого существует свойство отчета **Фильтр** (Filter). Значением этого свойства является логическое выражение WHERE инструкции SELECT. Применение фильтра состоит в том, что, когда заданное в качестве фильтра логическое выражение для конкретной записи принимает значение Истина (True), эта запись отображается, в противном случае — нет. Пример такого выражения представлен на рис. 10.29 (задан интервал дат, в котором должно лежать значение поля ДатаИсполнения (ShippedDate)). Для того чтобы фильтр, введенный в диалоговом окне свойств отчета, был применен к записям в

отчете, нужно еще установить значение свойства **Фильтр включен** (Filter On) равным **Да** (Yes). В противном случае он сохраняется, загружается при выполнении отчета, но не применяется. Свойство можно установить либо в диалоговом окне свойств, как это показано на рис. 10.29, либо в макросе или процедуре VBA.

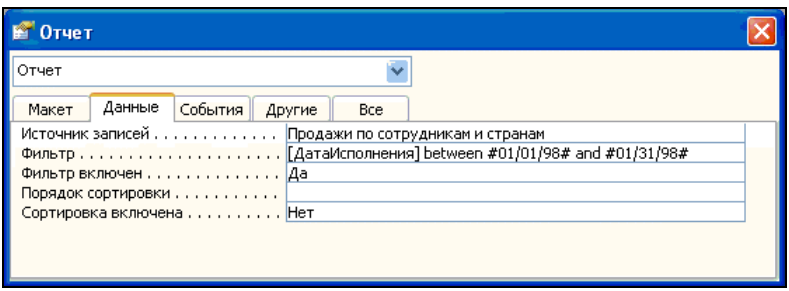


Рис. 10.29. Свойства **Фильтр** и **Фильтр включен** в окне свойств отчета

ГЛАВА 11



Публикация данных в Интернете и корпоративной сети

В последние несколько лет мы стали свидетелями революционного развития информационных технологий. Широкое распространение всемирной сети Интернет, унификация средств представления и передачи данных открыли удивительные возможности для доступа к информации. Вот лишь несколько примеров. Уже сейчас через Интернет можно узнать последние новости, сводку погоды, проанализировать предлагаемые услуги и ассортимент товаров в магазинах, сделать заказ. Ученые со всего мира могут объединить накопленные в ходе исследований данные и, обработав их, получить новые результаты. В связи с такими возможностями очень важно интегрировать современные системы управления базами данных со всемирной сетью Интернет. Поэтому компания Microsoft расширила функциональные средства своего популярного продукта по работе с базами данных Access технологиями публикации в Интернете. Об их применении и пойдет речь в этой главе.

Интеграция Access со всемирной сетью Интернет

Удобный интерфейс "Всемирной паутины" (World Wide Web) связывает большое количество ресурсов, имеющихся в сети Интернет. Используя World Wide Web, можно перемещаться между тысячами компьютерных узлов, системными приложениями, файлами и документами. Простота перемещения между документами и возможность читать их, используя любую компьютерную систему, позволили внедрить технологию Web в организациях. Многие организации разрабатывают собственные сети на основе технологий Интернет (интранет), чтобы размещать в них внутреннюю информацию, предназначенную только для сотрудников.

Рассмотрим, какие функциональные средства появились в Access для интеграции с Интернетом.

Сначала в версии Microsoft Office 2000 были объединены две мощные информационные технологии, определяющие новую модель работы с компьютером. Первая основана на том, что информация может быть размещена где угодно — на локальном жестком диске, в локальной или корпоративной сети или в глобальной сети

Интернет. Другая — на том, что пользователи реально работают не с приложениями, а непосредственно с документами и содержащейся в них информацией. В результате можно выбрать один из двух возможных подходов к работе.

- Работа преимущественно с приложениями Office с эпизодическими обращениями в интранет или Интернет за необходимой Web-страницей, документом, надстройкой для приложения или дополнительной информацией о программе.
- Работа преимущественно внутри Internet Explorer, использование его в качестве единственной среды, в которой можно просматривать и редактировать любой документ, расположенный на вашем жестком диске, в сети компании или в Интернете.

Тогда корпорация Microsoft включила в Access 2000 средства публикации объектов базы данных в виде статических или динамических Web-страниц. Во многих объектах базы данных (например, таблицах, формах, кнопках на формах и т. п.) стало возможным использовать гиперссылки для перехода к другим документам. В Access 2000 появилась возможность создания интерактивных Web-страниц для просмотра, ввода и анализа данных в виде нового типа объектов базы данных — страниц доступа к данным.

В следующей версии Microsoft Office 2002 был сделан еще шаг в направлении интеграции с Интернетом — введение формата XML как общего формата для представления документов всех приложений семейства Microsoft Office. В сочетании с удобными средствами публикации данных в виде Web-страниц и набором Web-компонентов, входящим в комплект поставки Microsoft Office, Access 2002 стал мощным инструментом для создания информационных систем, способных предоставить удобный доступ к информации практически из любой точки земного шара.

В новой версии Access 2003 расширена поддержка формата XML. А в составе семейства приложений Microsoft Office 2003 появился новый продукт, предназначенный специально для работы с данными в формате XML — InfoPath. В частности, InfoPath позволяет подключаться к разнообразным источникам данных, в том числе к Access 2003. Этой теме посвящен отдельный раздел данной главы.

Основные сведения об Интернете

Прежде чем приступить к подробному рассмотрению возможностей Access, связанных с публикацией данных, мы объясним значение некоторых терминов и понятий, касающихся сети Интернет, а также кратко рассмотрим возможности программы просмотра Web-страниц Internet Explorer. *(Значение терминов, упомянутых, но не объясненных в этой главе, можно найти в Приложении 1.)*

Интернет и *интранет* — это сети, в которых информация легко доступна, даже при использовании различных типов компьютеров с различными возможностями. Вместо применения архаических команд работы с сетями можно просматривать страницы информации, просто вводя адрес страницы. Этот адрес известен как *унифицированный указатель ресурсов* (Uniform Resource Locator), или *URL*. В зависимости от того, каким образом необходимо получить доступ к документу (через локальный диск, локальную сеть, Web-узел или файловый архив), URL (даже для одного и того же документа) может выглядеть по-разному. URL состоит из двух частей: специфи-

катора протокола для доступа к данному ресурсу и спецификатора расположения самого ресурса. Например:

- **file://c:\sales\sales.htm** — файл на локальном компьютере;
- **file://brig\sales\sales.htm** — файл на компьютере в локальной сети;
- **http://brig/sales/sales.htm** — файл на Web-сервере в сети интранет;
- **http://brig.boreas.ru/sales/sales.htm** — файл на удаленном Web-сервере в сети Интернет;
- **ftp://brig.boreas.ru/sales/sales.htm** — файл на удаленном FTP-сервере в сети Интернет.

Если конкретный файл в URL не указан, то открывается Web-страница, установленная по умолчанию для данного Web-сервера.

Термин *Web-сервер* (Web-узел) может трактоваться несколькими способами. С одной стороны, это набор документов, связанных гиперссылками (при этом у Web-сервера имеется основная страница, через которую за один или несколько шагов доступны все остальные), с другой стороны, термин Web-сервер может означать компьютер, на котором размещен набор документов, доступный через локальную или глобальную сеть. Наконец, последнее значение этого термина — программное обеспечение, предназначенное для доступа к набору документов через локальную или глобальную сеть. Везде в данной главе, где это особо не оговорено, мы будем использовать первое значение термина Web-сервер.

Web-страница (или страница Интернет, или документ в формате HTML) — это текстовый документ, содержащий специальные команды. При открытии Web-страницы в простом текстовом редакторе (например, Блокнот) вы увидите именно эти команды. Однако будучи открытой с помощью программы просмотра Интернета, такой как Internet Explorer, Mosaic или Netscape, Web-страница может отображать текст, графику, гиперссылки на другие документы, а также элементы управления. Секрет состоит в том, что программа просмотра Web-страниц содержит интерпретатор команд языка HTML, содержащихся в файле Web-страницы.

Язык HTML (Hypertext Markup Language) является системой разметки документов для их дальнейшей публикации в сети World Wide Web. Документы, подготовленные в формате HTML, включают рисунки со ссылками и команды форматирования. Для просмотра этих документов используется средство просмотра Web-страниц (например, программа Internet Explorer).

Гиперссылка — это текст, выделенный цветом или подчеркиванием, либо графическое изображение, при щелчке по которому осуществляется переход к файлу, определенному месту в файле, странице HTML в World Wide Web или странице HTML во внутренней сети (интранет). Гиперссылки могут также указывать на Gopher, Telnet, группы новостей (newsgroup) и узлы FTP. При переходе между страницами с помощью гиперссылок создается и сохраняется хронология просмотра всех страниц. Средства просмотра Web-страниц, подобные Internet Explorer, имеют на панелях инструментов кнопки перемещения, которые позволяют двигаться вперед или назад от одной просмотренной страницы к другой.

Публикация — это процесс вывода таблиц, форм и отчетов в статическом или динамическом формате HTML с последующей установкой всех связанных файлов в виде

приложений World Wide Web на один из серверов Web, например, Microsoft Internet Information Server или Microsoft Personal Web Server.

Программа просмотра Интернета

В ответ на растущую популярность Интернета корпорация Microsoft быстро разработала удобное и надежное средство навигации по ресурсам Интернета — Internet Explorer. В последних версиях своей операционной системы Windows компания Microsoft продвигает это приложение, предлагая использовать его даже для работы с документами на локальном компьютере, вместо приложения Проводник, чтобы работа с локальными и удаленными ресурсами выглядела для пользователя единообразно — с помощью универсального навигационного приложения. При помощи Internet Explorer можно просматривать не только страницы Интернета, но и работать с документами Word, рабочими листами Excel, презентациями PowerPoint вне зависимости от того, был ли сохранен документ в виде Web-страницы или в стандартном для приложения формате. При открытии документов, сохраненных в стандартном для создавшего их приложения формате, в окне Internet Explorer появляются меню и панели инструментов соответствующего приложения, позволяющие редактировать документ прямо в Internet Explorer. Это стало возможным благодаря технологии ActiveX.

Internet Explorer — это не только программа просмотра Web-страниц. В состав Internet Explorer (начиная с версии 5.0) входит множество дополнительных приложений и компонентов. В их числе:

- клиент электронной почты Outlook Express;
- программа для проведения телеконференций Microsoft NetMeeting;
- редактор Web-страниц FrontPage Express;
- универсальный проигрыватель мультимедийных файлов Media Player.

Мы не будем описывать в этой главе использование Internet Explorer и других приложений для работы с Интернетом. Во-первых, большинство читателей данной книги (тем более тех, кто читает эту главу) уже достаточно хорошо знакомо с Internet Explorer и сопутствующими приложениями. Во-вторых, у всех приложений, входящих в состав Internet Explorer, имеется справочная система, в которой можно быстро найти ответ на любой возникший у вас вопрос.

О примерах этой главы

Для иллюстрации создания и настройки разнообразных Web-документов в этой главе, как и в других, мы использовали демонстрационную базу данных `Борей` (Northwind), входящую в состав Microsoft Access.

Приведенные в этой главе примеры достаточно просты, однако они затрагивают практически все аспекты создания и использования Web-документов и специальных объектов: страниц доступа к данным в Access. Наша цель состояла в том, чтобы как можно шире обозреть имеющиеся в Access возможности. Вдумчивый читатель может самостоятельно глубже изучить заинтересовавшие его вопросы, пользуясь справочной системой Access и специальной справочной литературой.

Использование гиперссылок в объектах базы данных

В таблицы, формы и страницы доступа к данным Access можно добавлять гиперссылки для перехода к объектам той же самой или другой базы данных Access, к документам, созданным в других приложениях Access, к файлам, расположенным в Интернете или локальной сети организации.

Замечание

Гиперссылки можно добавлять и в отчеты Access, однако они будут работать только после преобразования отчета в формат Word, Excel или HTML.

В форме, отчете или на странице доступа к данным можно создать надпись или рисунок и присоединить к ним гиперссылку. В формах и на страницах доступа к данным можно присоединять гиперссылку к кнопкам. Тогда при нажатии на такую кнопку осуществляется переход к просмотру документа, на который указывает гиперссылка. Например, в базе данных `Борей` (Northwind) форма `Товары` (Products) содержит кнопку с гиперссылкой для перехода к просмотру отчета `Список товаров` (Alphabetical List of Products).

Мы рассмотрим использование гиперссылок в полях таблиц Access. Для остальных объектов, поддерживающих работу с гиперссылками, процедуры будут практически аналогичными.

Добавить гиперссылку можно только в то поле таблицы, для которого в режиме Конструктора таблицы указан тип **Гиперссылка** (Hyperlink). Например, такое поле есть в таблице `Поставщики` (Suppliers) в базе данных `Борей` (Northwind).

Чтобы добавить гиперссылку в поле таблицы или запроса:

1. Откройте таблицу или запрос в режиме Таблицы.
2. Поместите точку вставки в поле, имеющем тип **Гиперссылка** (Hyperlink) и содержащем пустое значение (в нашем примере это поле `Домашняя страница` (Home Page) в таблице `Поставщики` (Suppliers)).
3. Выберите команду **Вставка | Гиперссылка** (Insert | Hyperlink) или нажмите кнопку **Добавить гиперссылку** (Insert Hyperlink) на панели инструментов **Таблица в режиме таблицы** (Table Datasheet), или нажмите комбинацию клавиш <Ctrl>+<K>. Появится диалоговое окно **Добавление гиперссылки** (Insert Hyperlink) (рис. 11.1).
4. В поле **Текст** (Text to display) введите текст, который должна отображать гиперссылка, а в поле **Адрес** (Address) введите путь или URL файла, на который должна указывать гиперссылка.

   Гиперссылка может указывать на файл, находящийся на локальном диске или в сети (локальной или глобальной), на еще не существующую страницу, а также на адрес электронной почты. Используйте панели с ярлыками для быстрого поиска и ввода адресов документов, с которыми вы уже работали, а также в том случае, если вы не знаете правильного формата адреса объекта, на который должна указывать гиперссылка.

Для поиска файла на локальном диске можно использовать кнопку **Поиск файла** (Browse for File) на панели инструментов в диалоговом окне (см. рис. 11.1). Эта кнопка открывает диалоговое окно **Связать с файлом** (Link to File), в котором необходимо указать нужный файл и нажать кнопку **OK**.

Для поиска Web-страницы, на которую должна указывать гиперссылка, нажмите кнопку **Интернет** (Browse the Web). Запустится программа Internet Explorer. Откройте в Internet Explorer интересующий вас документ и закройте окно Internet Explorer. Адрес документа будет автоматически введен в поле **Адрес**.

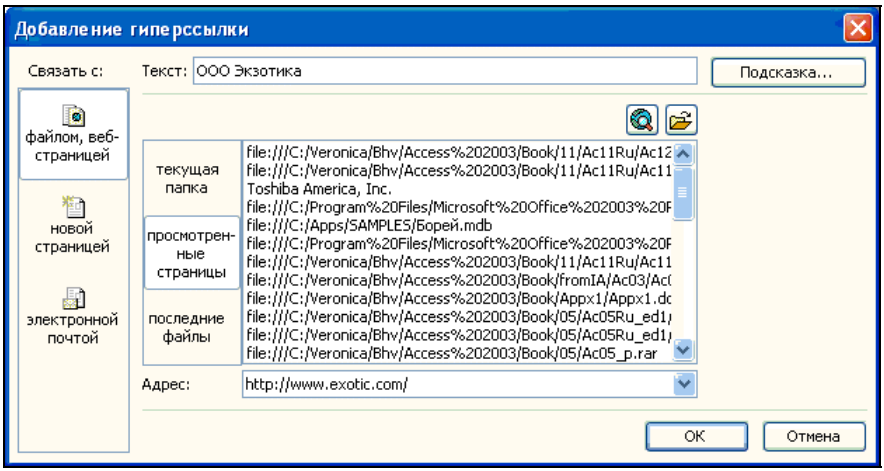


Рис. 11.1. Диалоговое окно **Добавление гиперссылки**

5. Чтобы при наведении указателя мыши на гиперссылку автоматически отображалась всплывающая подсказка с пояснительным текстом, нажмите кнопку **Подсказка** (ScreenTip). Появится диалоговое окно **Подсказка для гиперссылки** (Set Hyperlink ScreenTip). Введите текст подсказки и нажмите кнопку **OK**.
6. Нажмите кнопку **OK**. Гиперссылка будет вставлена в поле таблицы Access (рис. 11.2).

Чтобы добавить гиперссылку на кнопку в форме:

1. Откройте форму в режиме Конструктора.
2. С помощью Панели элементов (Toolbox) создайте новую кнопку на форме или выделите существующую.
3. Откройте окно свойств кнопки. На вкладке **Все** (All) задайте значение свойства **Адрес гиперссылки** (Hyperlink Address). Для этого нажмите кнопку Построителя.
4. Появится диалоговое окно **Добавление гиперссылки** (Insert Hyperlink), аналогичное представленному на рис. 11.1. Но в данном случае в этом окне в группе **Связать с** (Link to) появится кнопка **объектом базы данных** (Object in This Database), дающая возможность указать на объект базы данных. А еще в случае указания ссылки на Web-страницу или файл будет доступна кнопка **Закладка** (Bookmark). Нажмите эту кнопку, чтобы ссылка указывала не на весь документ, а на опреде-

ленную его часть. Появится диалоговое окно **Выбор места в документе** (Select Place in Document). Его вид зависит от типа документа, адрес которого указан в поле **Адрес** (Address). Закладкой может являться диапазон ячеек Excel, закладка в документе Word, а также объект базы данных Access. На рис. 11.3 показано окно **Выбор места в документе** (Select Place in Document) для базы данных Access. Выберите закладку и нажмите кнопку **OK**.

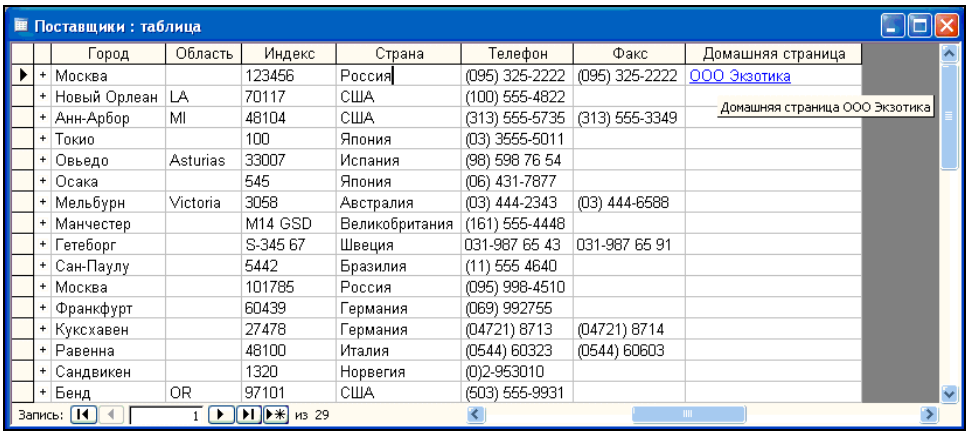


Рис. 11.2. Гиперссылка в поле таблицы Access

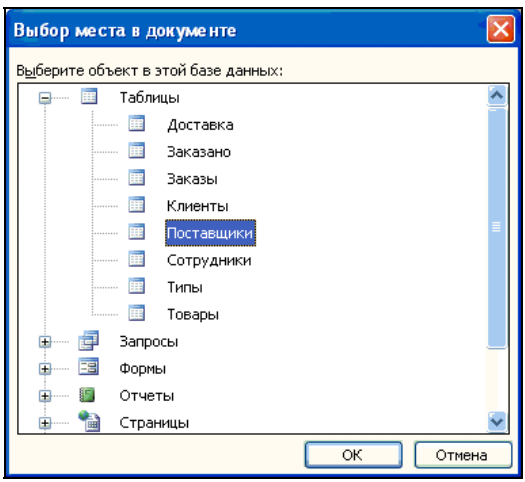


Рис. 11.3. Диалоговое окно **Выбор места в документе**

Замечание

Еще один способ указать отдельную часть документа в качестве гиперссылки (т. е. относительный адрес объекта) — в окне свойств кнопки задать значение свойства **Дополнительный адрес** (Hyperlink SubAddress).

Теперь давайте обсудим операции, которые применимы к гиперссылкам. Для этого щелкните правой кнопкой мыши по гиперссылке в таблице Access и в появившемся контекстном меню выберите команду **Гиперссылка** (Hyperlink). Раскроется меню, содержащее список команд, применимых к гиперссылкам. Описание этих команд приведено в табл. 11.1.

Таблица 11.1. *Команды для работы с гиперссылками*

Команда	Описание
Изменить гиперссылку (Edit Hyperlink)	Используется для добавления или изменения существующей гиперссылки. Если гиперссылка еще не была добавлена, то при вызове этой команды открывается диалоговое окно **Добавление гиперссылки** (см. рис. 11.1). Если же гиперссылка уже была добавлена, то появится точно такое же диалоговое окно, но называться оно будет **Изменение гиперссылки** (Change Hyperlink). Процедура изменения гиперссылки ничем не отличается от приведенной ранее процедуры добавления гиперссылки
Открыть гиперссылку (Open Hyperlink)	Открывает документ, на который указывает гиперссылка в программе просмотра, используемой по умолчанию, или в приложении, использовавшемся для создания документа. Если для гиперссылки была указана закладка, то при открытии документа автоматически выделяется диапазон, помеченный этой закладкой. Выбор команды **Открыть гиперссылку** (Open Hyperlink) эквивалентен щелчку по гиперссылке в режиме Таблицы
Открыть в новом окне (Open in New Window)	Аналогично предыдущей команде, только документ открывается в новом окне
Копировать гиперссылку (Copy Hyperlink)	Используется для копирования гиперссылки в Буфер обмена Windows
Поместить в папку "Избранное" (Add to Favorites)	Добавляет ярлык к документу, на который указывает гиперссылка, в папку "Избранное"
Отображать текст (Display Text)	В поле этой команды меню можно указать текст, отображаемый гиперссылкой
Удалить гиперссылку (Remove Hyperlink)	Используется для удаления гиперссылки

При создании гиперссылки можно указать путь или адрес ресурса как в абсолютном, так и в относительном формате. Мы рекомендуем везде, где это возможно, использовать относительные гиперссылки, поскольку в таком случае при перемещении группы папок, содержащих связанные между собой файлы, все гиперссылки останутся корректными и не потребуют дополнительных изменений. В относительном пути используется следующее обозначение для папки уровнем выше:

..\

Относительная ссылка на файл в этой же папке:

FileName.doc

Относительная ссылка на файл, находящийся в папке уровнем выше:

..\FileName.doc

Относительная ссылка на файл в папке Docs внутри текущей папки:

Docs\FileName.doc

Для настройки относительных гиперссылок можно установить базовый адрес:

1. Откройте базу данных, для которой необходимо установить базовый адрес гиперссылок.
2. Выберите команду **Файл | Свойства базы данных** (File | Database Properties).
3. Раскройте вкладку **Документ** (Summary).
4. В поле **База гиперссылки** (Hyperlink base) введите путь, относительно которого будут отсчитываться все гиперссылки в базе данных. Например: C:\MyProject.

Панель инструментов *Веб-узел*

Панель инструментов **Веб-узел** (Web) (рис. 11.4) используется для управления перемещением по Web из любого приложения Office. Кнопки на панели инструментов **Веб-узел**, описанные в табл. 11.2, производят те же действия, что и соответствующие кнопки на панели инструментов Internet Explorer или какой-либо другой программы просмотра.



Рис. 11.4. Панель инструментов **Веб-узел**

Таблица 11.2. *Кнопки панели инструментов* ***Веб-узел***

Кнопка	Описание
Назад (Back)	Переход на предыдущую Web-страницу или документ в хронологии переходов по гиперссылкам
Далее (Forward)	Переход на следующую Web-страницу или документ в хронологии переходов по гиперссылкам
Остановить (Stop)	Остановка загрузки текущей Web-страницы или документа
Обновить (Refresh)	Повторная загрузка текущей Web-страницы или документа
Начальная страница (Start Page)	Переход к начальной странице

Таблица 11.2 *(окончание)*

Кнопка	Описание
Найти в Интернете (Search the Web)	Переход на страницу поиска, где можно вводить ключевые слова для поиска
Меню **Избранное** (Favorites)	Список избранных Web-страниц или документов
Меню **Переход** (Go)	Меню навигации и установки начальной и поисковой страниц
Отображать только веб-панель (Show Only Web Toolbar)	Переключение в режим отображения из встроенных панелей инструментов только панели **Веб-узел** и обратно
Адрес (Address)	Строка для ввода URL и раскрывающийся список с использованными ранее URL

Используйте панель инструментов **Веб-узел** при необходимости перехода по гиперссылкам в процессе работы с Access. Эта панель инструментов дает возможность перемещаться вперед или назад по посещенным страницам или переходить к любому уже загружавшемуся ранее документу.

Совет

Если вам часто приходится открывать одни и те же документы или страницы, можно сэкономить время, добавляя документ Office или адрес Web-страницы в меню **Избранное** (Favorites). Чтобы перейти к документу, выберите ссылку на него в меню **Избранное** (Favorites).

Преобразование объектов базы данных в Web-страницы

В Access существует целых три варианта преобразования объекта базы данных в Web-страницы для публикации его в Интернете или корпоративной сети:

- в формат статических или динамических страниц HTML;
- в страницы доступа к данным (начиная с версии Access 2002);
- в универсальный формат XML (начиная с версии Access 2002).

У каждого из форматов есть свои достоинства, и выбор формата зависит от конкретной задачи, которую должна решить публикация Web-страницы. Коротко остановимся на соответствующих характеристиках форматов.

Статические страницы HTML можно легко и быстро сгенерировать из объекта базы данных, они не требуют подключения к источнику данных, содержат в себе всю необходимую информацию для отображения и не требуют никакой дополнительной настройки при публикации на Web-сервере в сети Интернет или интранет, в общей папке на файл-сервере в локальной сети, на FTP-сервере, при пересылке по элек-

тронной почте. Однако такие страницы содержат лишь те данные, которые существовали в базе данных на момент публикации, и, конечно, эти данные доступны только для просмотра, не для редактирования.

Чтобы предоставить пользователям Web актуальную информацию, достаточно один раз создать страницу в формате динамической, генерируемой сервером страницы HTML, и каждый раз после изменений в источнике новые данные попадут в нее автоматически. Такие страницы могут быть отображены любой программой просмотра Интернета. Однако редактировать данные с помощью такой страницы нельзя.

Страницы доступа к данным представляют собой полноценный интерактивный интерфейс к данным в базе. Формат динамических страниц HTML, генерируемых сервером (ASP и HTX/IDC), позволяет получать актуальные данные, но редактировать их нельзя. Страницы доступа к данным позволяют делать и то и другое. И еще с их помощью можно добавлять, удалять, сортировать и группировать записи. Каждый раз после изменений в базе данных новые данные автоматически попадут на страницу доступа к данным. А если отредактировать данные на странице доступа к данным, изменения автоматически попадут в присоединенную базу данных.

Универсальный формат представления данных XML позволяет получить данные в стандартизированном виде, не зависящем от приложения, в котором они были созданы, в данном случае — от Access. С такими данными может работать любое приложение, которое воспринимает этот универсальный формат.

В Access предусмотрено два пути преобразования объектов: экспорт и сохранение в виде другого объекта базы данных.

Таблица 11.3. *Форматы для преобразования объектов баз данных Access*

Объект	Доступные типы объектов при сохранении объекта	Доступные форматы Web при экспорте объекта
Таблица базы данных или проекта Access	Таблица, форма, отчет, страница доступа к данным	Страницы HTML (*.html; *.htm) Страницы Microsoft ASP (*.asp) Microsoft IIS 1-2 (*.htx; *.idc) Документы XML (*.xml)
Запрос базы данных или Представление проекта Access	Запрос, форма, отчет, страница доступа к данным	Страницы HTML (*.html; *.htm) Страницы Microsoft ASP (*.asp) Microsoft IIS 1-2 (*.htx; *.idc) Документы XML (*.xml)
Сохраненная процедура проекта Access	Сохраненная процедура, форма, отчет, страница доступа к данным	Страницы HTML (*.html; *.htm) Страницы Microsoft ASP (*.asp) Microsoft IIS 1-2 (*.htx; *.idc) Документы XML (*.xml)
Функция проекта Access	Функция, форма, отчет	Страницы HTML (*.html; *.htm) Страницы Microsoft ASP (*.asp) Microsoft IIS 1-2 (*.htx; *.idc) Документы XML (*.xml)

Таблица 11.3 *(окончание)*

Объект	Доступные типы объектов при сохранении объекта	Доступные форматы Web при экспорте объекта
Форма	Форма, отчет, страница доступа к данным	Страницы HTML (*.html; *.htm) Страницы Microsoft ASP (*.asp) Microsoft IIS 1-2 (*.htx; *.idc) Документы XML (*.xml)
Отчет	Отчет, страница доступа к данным	Страницы HTML (*.html; *.htm) Документы XML (*.xml)
Страница доступа к данным	Страница доступа к данным	Страница доступа к данным Microsoft
Макрос	Макрос, модуль	Нет
Модуль	Модуль	Нет

Из табл. 11.3 видим, что все объекты баз данных Access, кроме макросов, модулей и еще функций проектов Access, можно сохранить в виде страницы доступа к данным и преобразовать в разнообразные форматы Web-страниц. Функции проектов Access нельзя сохранить в виде страницы доступа к данным, но можно экспортировать в любой из поддерживаемых форматов Web-страниц.

Внимание!

Если в результате экспорта требуется получить Web-страницу в формате ASP (в т. ч. для документов XML с представлением в виде ASP) или HTX/IDC, обязательно указывайте имена всех файлов и папок латинскими буквами, т. к. Web-сервер поддерживает только латинские имена генерируемых страниц, готовые документы помещайте на Web-сервер и открывайте его, пользуясь доступом к этому серверу через HTTP. Если сохраненный документ содержал русский текст, а кодировка, в которой он был сохранен, — это UTF или Unicode, при просмотре страницы в Internet Explorer может потребоваться выбрать кириллическую кодировку Windows с помощью команды **Вид | Кодировка | Кириллица (Windows)** (View | Encoding | Cyrillic (Windows)).

Преобразование в формат HTML

Перед тем как приступить к описанию процедур преобразования объектов базы данных в формат HTML, необходимо определить, какой формат файла HTML следует использовать. Выбрать оптимальный формат можно на основе того, какие задачи должно решать разрабатываемое приложение. Статический формат HTML используется, если данные изменяются редко, и приложение Web не использует формы. Динамический формат применяется, если изменение данных происходит часто, а данные базы данных Microsoft Access должны извлекаться приложением Web с помощью форм. В этом разделе мы остановимся на технических моментах, относящихся к статическому и динамическому форматам HTML. При первом прочтении оставшуюся часть этого раздела можно пропустить и перейти к следующему разделу.

Статические страницы HTML можно создать на основе таблиц, запросов, форм и отчетов. Полученные в результате файлы HTML являются снимком данных, полученным на время публикации файлов. При изменении данных экспортированные ранее файлы необходимо обновить вручную (повторить операцию экспорта).

Динамический формат HTML позволяет произвести операцию преобразования один раз. После этого все обновления данных происходят автоматически. Access 2002 поддерживает преобразование объектов в два динамических формата HTML — IDC/HTX и ASP (Active Server Pages).

При преобразовании объектов в формат IDC/HTX вместо файлов с расширением html создаются файлы с расширением htx (шаблон отображения данных) и idc (команды для извлечения данных из базы). HTX-файл является файлом в формате HTML и содержит команды и инструкции по форматированию, а вместо данных содержит шаблоны, указывающие место вставки значений из запроса в файле с расширением idc. IDC-файл содержит запрос в форме инструкции SQL и данные, используемые Internet Information Server для связи с источником данных ODBC, в этом конкретном случае — с базой данных Microsoft Access. Данные о связи включают название источника данных, а также имя пользователя и пароль, если для открытия базы данных используется защита на уровне пользователей. При обращении к объекту базы данных, опубликованному в описываемом формате, Web-сервер (Internet Information Server) открывает базу данных Microsoft Access (с помощью драйвера Microsoft Access и данных в IDC-файле), запускает запрос из файла IDC для доступа к данным. После выполнения запроса полученный результат объединяется с данными из файла HTX в один файл с расширением html, который затем отправляется программе просмотра, где отображается в виде Web-страницы.

При преобразовании объектов в динамический формат ASP вместо файлов с расширением html создается файл с расширением asp, который содержит команды HTML, один или несколько запросов в форме инструкций SQL, а также инструкции форматирования данных и программы на языке VBScript. Кроме этого, файл с расширением asp содержит сведения о соединении ODBC, используемые для связи с источником данных. Сведения о связи включают имя источника данных, а также имя пользователя и пароль, если для входа в базу данных предусмотрена защита на уровне пользователей. Принцип работы Active Server Pages похож на описанный ранее, однако за счет множества дополнительных возможностей и более надежного выполнения технология Active Server Pages на сегодняшний день стала стандартом де-факто для динамических Web-страниц.

Замечание

Средства поддержки Active Server Pages включены в состав Microsoft Internet Information Server версии 3.0 и выше.

В следующих разделах мы подробно рассмотрим процедуры публикации объектов базы данных в статический или динамический формат HTML.

Преобразование в статический формат HTML

Преобразование объекта базы данных в формат HTML или XML производится с помощью одной и той же команды **Экспорт** (Export), о которой уже говорилось

в *гл. 6, разд. "Преобразование отчета в другие форматы"*. Однако, в данном случае, кроме выбора формата экспорта требуется предпринять еще несколько шагов, поэтому мы остановимся на этом варианте преобразования объектов и рассмотрим его поподробнее.

Чтобы преобразовать объект базы данных в статический формат HTML:

1. В окне **База данных** (Database) выделите щелчком левой кнопки мыши нужный объект и выберите команду **Файл | Экспорт** (File | Export), или щелкните по объекту правой кнопкой мыши и в появившемся контекстном меню выберите команду **Экспорт** (Export). Появится диалоговое окно **Экспорт объекта** (Export To) (рис. 11.5).
2. Раскройте список **Тип файла** (Save as type) и выделите элемент **Документы HTML** (HTML Documents).

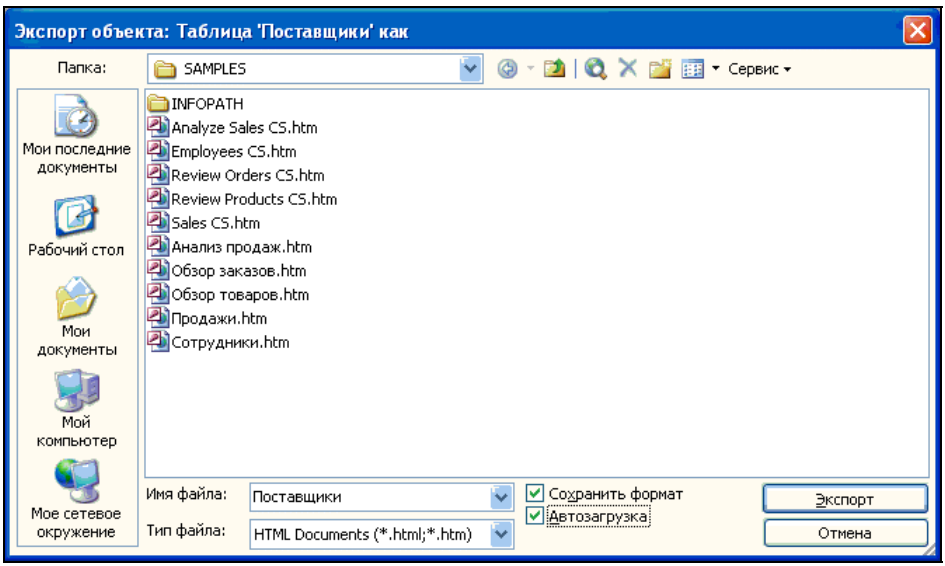


Рис. 11.5. Диалоговое окно **Экспорт объекта**

3. Выберите диск или папку, в которую необходимо сохранить экспортированный объект, и в поле **Имя файла** (File name) введите имя файла HTML.

Замечание

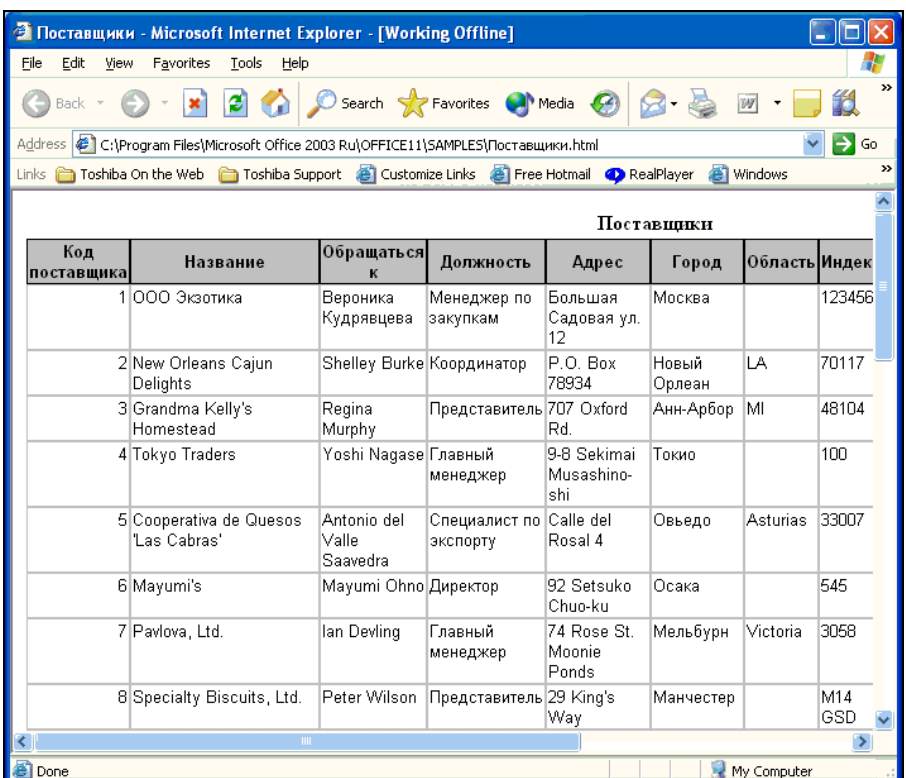
Если вы собираетесь опубликовать Web-страницу на Web-сервере, то в поле **Имя файла** необходимо указать англоязычное имя файла. Поскольку протокол, используемый для доступа к Web-страницам (HTTP — Hypertext Transfer Protocol) поддерживает только имена файлов, набранные латинскими символами.

4. Установите флажки **Сохранить формат** (Save formatted), если вы экспортируете таблицу и хотите сохранить ее в формате, близком к ее внешнему виду в режиме Таблицы, и **Автозагрузка** (Autostart), если нужно, чтобы созданная в результате

экспорта Web-страница сразу же была открыта в программе просмотра Интернета.

5. Нажмите кнопку **Экспорт** (Export).
6. Если на шаге 4 был вручную или автоматически установлен флажок **Сохранить формат**, то на экране появится диалоговое окно **Параметры вывода в формате HTML** (HTML Output Options), в котором Access предложит выбрать шаблон документа HTML, который будет использован при оформлении новых страниц. Если требуется использовать шаблон, нажмите кнопку **Обзор** (Browse) и выделите существующий файл HTML. Если шаблон не нужен, убедитесь, что флажок напротив имени шаблона снят, а поле ввода оставьте пустым. В этом случае для Web-страницы будет использовано стандартное оформление. Выберите переключатель, соответствующий кодировке, в которой будут созданы страницы (можно выбрать кодировку по умолчанию или **Юникод** (Unicode)), и нажмите кнопку **ОК**.

Результат экспорта таблицы `Поставщики` (Suppliers) базы данных `Борей` (Northwind) в формат HTML показан на рис. 11.6.



Код поставщика	Название	Обращаться к	Должность	Адрес	Город	Область	Индек
1	ООО Экзотика	Вероника Кудрявцева	Менеджер по закупкам	Большая Садовая ул. 12	Москва		123456
2	New Orleans Cajun Delights	Shelley Burke	Координатор	P.O. Box 78934	Новый Орлеан	LA	70117
3	Grandma Kelly's Homestead	Regina Murphy	Представитель	707 Oxford Rd.	Анн-Арбор	MI	48104
4	Tokyo Traders	Yoshi Nagase	Главный менеджер	9-8 Sekimai Musashino-shi	Токио		100
5	Cooperativa de Quesos 'Las Cabras'	Antonio del Valle Saavedra	Специалист по экспорту	Calle del Rosal 4	Овьедо	Asturias	33007
6	Mayumi's	Mayumi Ohno	Директор	92 Setsuko Chuo-ku	Осака		545
7	Pavlova, Ltd.	Ian Devling	Главный менеджер	74 Rose St. Moonie Ponds	Мельбурн	Victoria	3058
8	Specialty Biscuits, Ltd.	Peter Wilson	Представитель	29 King's Way	Манчестер		M14 GSD

Рис. 11.6. Таблица `Поставщики`, экспортированная в формат HTML

Web-страница создается на основе набора записей для объекта в режиме Таблицы, включая значения его свойств **Порядок сортировки** (Order By) и **Фильтр** (Filter), и

выглядит как таблица HTML с ячейками, заполненными данными, извлеченными из выбранного объекта базы данных. Если экспортируется запрос с параметрами, то сначала происходит поиск значений параметров, а затем экспорт результатов.

Значения большинства полей (за исключением полей объектов OLE и гиперссылок) отображаются в виде строк и имеют тот же формат, что и в таблице, включая параметры свойств **Формат поля** (Format) или **Маска ввода** (InputMask). Поля с типом данных гиперссылки отображаются в виде ссылок HTML с помощью тегов `<A HREF>`. По умолчанию все неформатированные типы данных, за исключением текстовых и MEMO, сохраняются с выравниванием по правому краю. Текстовые поля и поля MEMO по умолчанию сохраняются с выравниванием по левому краю.

Если размеры таблицы достаточно велики, то ее вывод на экран и отображение через программу просмотра может занять значительное время. В этом случае рекомендуется сократить размер таблицы, разделив ее на несколько таблиц меньшего размера с помощью условия отбора, например по полю даты, или использовать отчет или форму для просмотра данных.

Помимо экспорта, Access позволяет выполнять и преобразование данных в обратном направлении — импорт из формата HTML в базу данных *(см. разд. "Импорт и присоединение документов HTML", гл. 3)*.

Преобразование в динамический формат HTML

Чтобы преобразовать объект базы данных в динамический формат HTML:

1. В окне **База данных** (Database) выделите щелчком левой кнопки мыши нужный объект и выберите команду **Файл | Экспорт** (File | Export), или щелкните по объекту правой кнопкой мыши и в появившемся контекстном меню выберите команду **Экспорт** (Export). Появится диалоговое окно **Экспорт объекта** (Export To) (см. рис. 11.5).
2. В поле **Тип файла** (Save as type) выделите элемент **Microsoft IIS 1-2** или **Страницы Microsoft ASP** (Microsoft Active Server Pages), в зависимости от требуемого динамического формата HTML.
3. Выберите диск или папку, в которой необходимо сохранить экспортируемый объект. В поле **Имя файла** (File name) введите имя файла (латинскими буквами).
4. Нажмите кнопку **Экспорт** (Export).
5. Если вы используете формат Active Server Pages, то появится диалоговое окно **Настройка вывода файлов ASP** (Microsoft Active Server Pages Output Options). В этом окне необходимо указать название источника данных ODBC, который будет использован на Web-сервере для доступа к данным, а также ввести адрес сервера, на котором будет размещен файл ASP. Например, если файл ASP должен быть опубликован в папке \Sales на сервере \\Brig, необходимо ввести `http://Brig/Sales/`). Кроме того, может потребоваться указать имя пользователя и пароль для доступа к базе данных (рис. 11.7). После завершения настройки параметров файлов ASP нажмите кнопку **OK**.

Замечание

Как и в случае экспорта объекта базы данных в статический формат HTML, можно указать файл шаблона, который будет использоваться при форматировании Web-страницы.

Если же вы используете формат Microsoft IIS 1-2, то появится диалоговое окно **Настройка вывода файлов HTX и IDC** (HTX/IDC Output Options), в нем обязательно нужно указать имя источника данных, который Web-сервер должен использовать для доступа к объекту базы данных (рис. 11.8). В этом же окне можно указать имя пользователя и пароль для доступа к данным, а также шаблон HTML, который должен использоваться для форматирования Web-страницы. После завершения настройки параметров файлов IDC/HTX нажмите кнопку **OK**.

6. Объект будет преобразован в заданный динамический формат HTML.

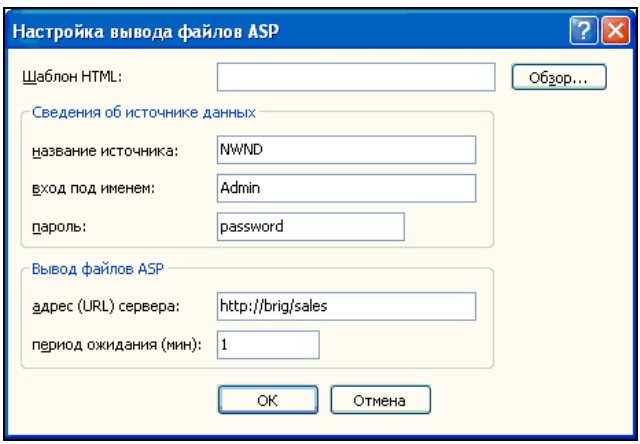


Рис. 11.7. Диалоговое окно **Настройка вывода файлов ASP**

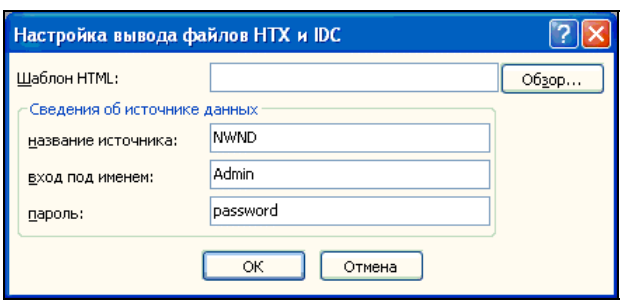


Рис. 11.8. Диалоговое окно **Настройка вывода файлов HTX и IDC**

Замечание

Для отображения динамических Web-страниц в программе просмотра необходимо опубликовать их на Web-сервере и обращаться к динамическим страницам только через него. В файлах динамических Web-страниц нет полезных данных, которые можно было бы отобразить в программе просмотра. Эти данные может сформировать только Web-сервер при обращении к динамической Web-странице.

Если объект в режиме Таблицы содержит запрос с параметрами, то Microsoft Access имитирует диалоговое окно **Введите значение параметра** (Enter Parameter Value).

Для этого создается дополнительная Web-страница, содержащая текстовое поле для ввода значения параметра и кнопку для запуска запроса. Страница параметров HTML должна быть выведена на экран до страницы таблицы. Во время экспорта Microsoft Access выполняет запрос и выводит на экран окно **Введите значение параметра**, но в нем достаточно просто нажать кнопку **OK**, поскольку любой ввод данных будет проигнорирован.

Преобразование в формат XML

Рассмотрим процедуру преобразования объекта базы данных в формат XML:

1. В окне базы данных выделите необходимый объект щелчком левой кнопки мыши и выберите команду **Файл | Экспорт** (File | Export), или щелкните по объекту правой кнопкой и выберите в контекстном меню команду **Экспорт** (Export).
2. Появится диалоговое окно **Экспорт объекта** (Export To). В раскрывающемся списке **Типы** (Save as type) выделите элемент **Документы XML** (XML Documents). В поле **Имя файла** (File name) введите название документа HTML и нажмите кнопку **Экспорт** (Export).
3. Появится диалоговое окно **Экспорт XML** (Export XML) (рис. 11.9).

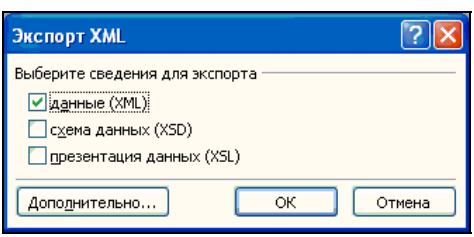


Рис. 11.9. Диалоговое окно **Экспорт XML**

4. Это диалоговое окно содержит три флажка для трех составляющих документа XML: данных, схемы данных и представления данных *(см. разд. "Формат XML в Access. Общие сведения" гл. 3)*. Установите флажки для тех частей документа XML, которые необходимо получить в результате экспорта. Например, если установить одновременно флажки **данные (XML)** (Data (XML)) и **схема данных (XSD)** (Schema of the data (XSD)), то схема данных будет экспортирована и сохранена отдельно от файла данных XML, в файле XSD. Для таблиц можно экспортировать только данные или их структуру или данные и структуру вместе и не экспортировать представление таблицы в виде Web-страницы: флажок **презентация данных (XSL)** (Presentation of your data (XSL)) можно снять.

Замечание

Флажок **данные (XML)**, следует сбросить лишь в том случае, если вы хотите экспортировать только новое представление объекта базы данных, а данные были экспортированы в файл XML раньше. Иначе при открытии полученной Web-страницы будет выдано сообщение об ошибке подключения к источнику данных, а сама страница окажется практически пустой.

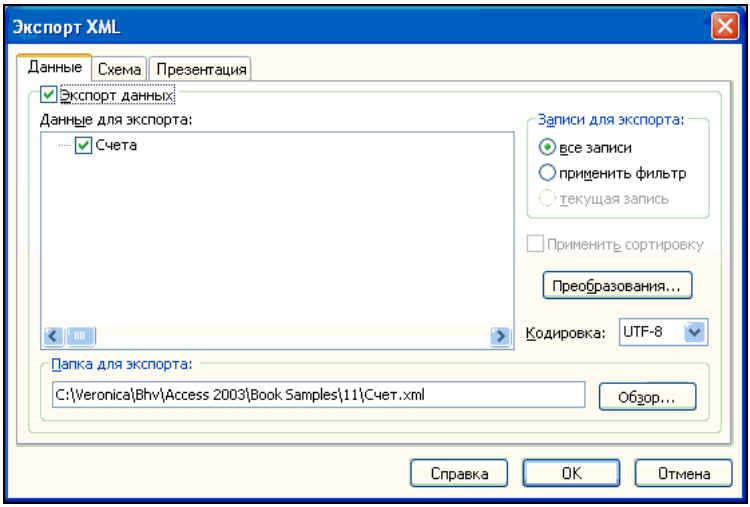


Рис. 11.10. Диалоговое окно **Экспорт XML** в развернутом виде

5. Чтобы установить дополнительные параметры преобразования, нажмите кнопку **Дополнительно** (Advanced). Диалоговое окно развернется (рис. 11.10).
6. Расширенное диалоговое окно содержит три вкладки: по одной для каждого из трех составляющих документа XML: данных, схемы данных и представления данных. На каждой из вкладок можно установить флажок, позволяющий экспортировать соответствующую часть документа XML, указать путь и имя файла, в котором будет сохранена эта часть документа, и задать некоторые дополнительные параметры:
   - На вкладке **Данные** (Data) с помощью раскрывающегося списка **Кодировка** (Encoding) можно выбрать кодировку для сохранения содержимого страницы. Можно задать фильтр для экспорта только выбранных записей. Кнопка **Преобразования** (Transforms) открывает диалоговое окно, позволяющее задать список преобразований (файлов XSL или XSLT), для автоматического применения сразу после экспорта.
   - На вкладке **Схема** (Schema) с помощью переключателей можно выбрать, экспортировать ли структуру данных в отдельный файл XSD или внедрить ее в файл XML вместе с данными. А также указать, следует ли экспортировать сведения об индексах и ключе.
   - На вкладке **Презентация** (Presentation) с помощью переключателей можно выбрать формат представления Web-страницы: обрабатываемый на стороне клиента HTML или на стороне сервера ASP; если экспортируемый объект содержит рисунки, можно указать имя папки, в которой они расположены, если их нужно включить в результирующий документ. На этой же вкладке есть поле для ввода имени файла XSL с описанием оформления страницы. Если указанный файл не существует, Access создаст файл в формате XSL с описанием внешнего представления документа (разработчик может использовать его в дальнейшем как основу для разработки файла в формате XSLT для преобразования документа XML в другой формат).

Совет

Если потребуется дополнительная информация о представлении данных в формате XML, нажмите кнопку **Справка** (Help). Информацию о разработке и преобразовании документов XML можно найти на Web-узле Microsoft для разработчиков: **http://msdn.microsoft.com/**.

7. Нажмите кнопку **OK**.
8. Если предложенные параметры преобразования не были изменены, в результате будут созданы три файла: XML — с данными, XSL — со стилями и HTM — с представлением преобразованного объекта. Чтобы открыть полученную Web-страницу, требуется открыть в Internet Explorer файл HTM или ASP в зависимости от того, какой формат представления был выбран.

К сожалению, подчиненные формы и отчеты, а также связанные рисунки не будут добавлены в документ XML, полученный из объекта базы данных. Однако простые объекты с внедренными рисунками, экспортированные в формат XML, будут

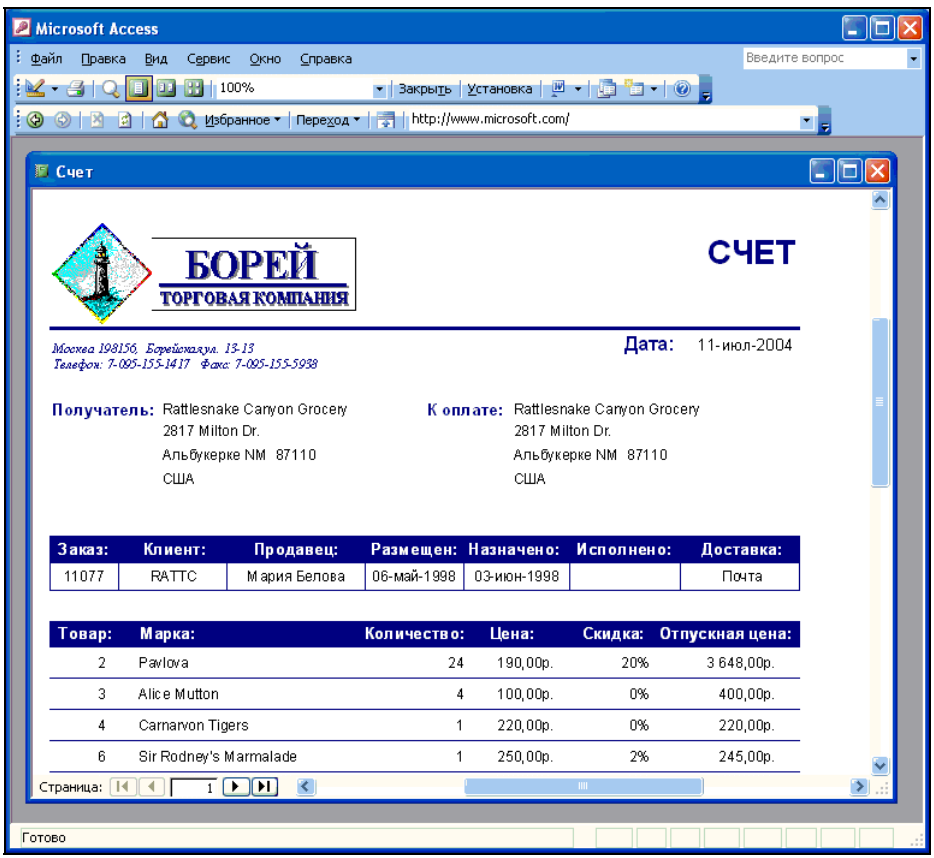


Рис. 11.11. Отчет `Счет`, открытый в базе данных `Борей`

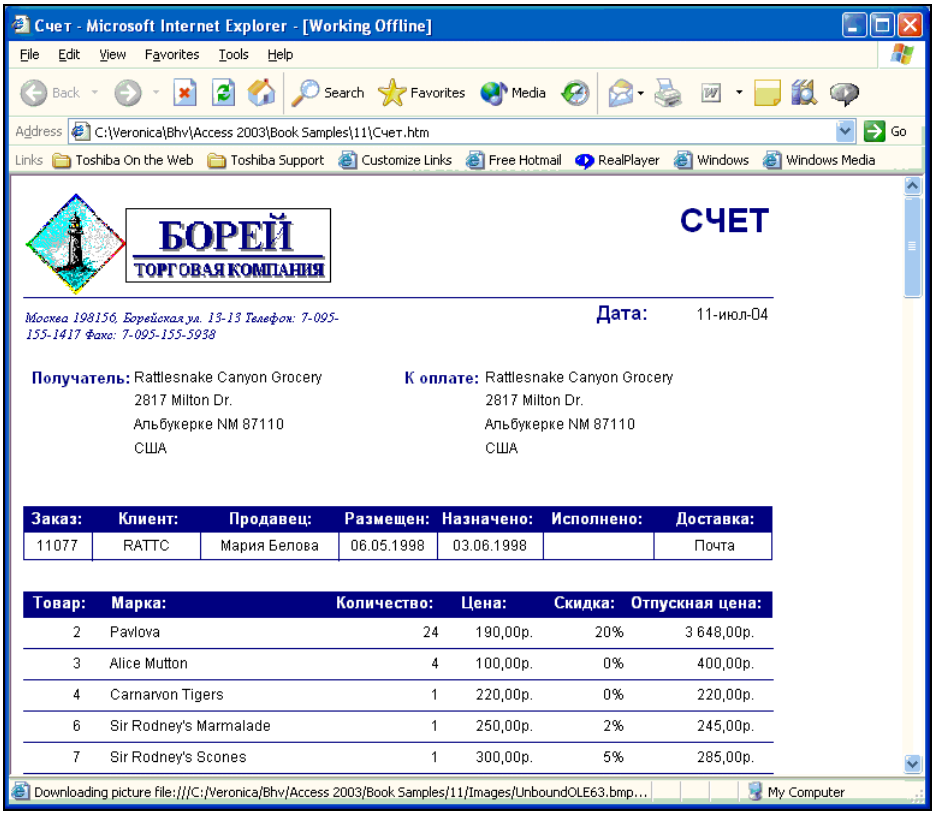


Рис. 11.12. Отчет `Счет`, данные которого преобразованы в формат XML, открытый в Internet Explorer

выглядеть практически точно так же, как исходные в Access (при условии, что имена внедренных рисунков и папки, в которую они помещаются, написаны латинскими буквами). Вы можете в этом убедиться, сравнив, к примеру, исходный и преобразованный в формат XML отчет `Счет` (Invoice) из базы данных `Борей` (Northwind) (рис. 11.11 и 11.12).

Если данных в источнике данных выбранного объекта много, загрузка Web-страницы в программу просмотра Интернета может занять существенное время. Для ускорения процесса загрузки можно экспортировать лишь часть данных, задав фильтр.

Замечание

Поскольку сами данные, содержащиеся в таком документе Web, полностью хранятся в файле XML, нет нужды в доступе к базе данных, из которой изначально были взяты эти данные (как, например, и в случае преобразования отчета в формат XML). Это означает, что для обеспечения доступа к такому документу достаточно поместить только упомянутый набор файлов, содержащих описание документа, на сервер

Web. С другой стороны, это означает, что изменение данных в таком документе возможно лишь с помощью приложений, имеющих доступ к файлу XML и способных воспринять этот формат.

Преобразование в страницу доступа к данным

Третий вариант преобразования объектов базы данных — в формат страниц доступа к данным — доступен с помощью команды **Сохранить как** (Save As). Чтобы выполнить такое преобразование, нужно выделить объект в окне базы данных или открыть его в любом режиме и выбрать эту команду в меню **Файл** (File) или контекстном меню объекта. Появится диалоговое окно, в котором потребуется указать имя новой страницы и выбрать в списке предложенных форматов элемент **Страница доступа к данным** (Data Access Page), по умолчанию будет выбран элемент с исходным типом объекта.

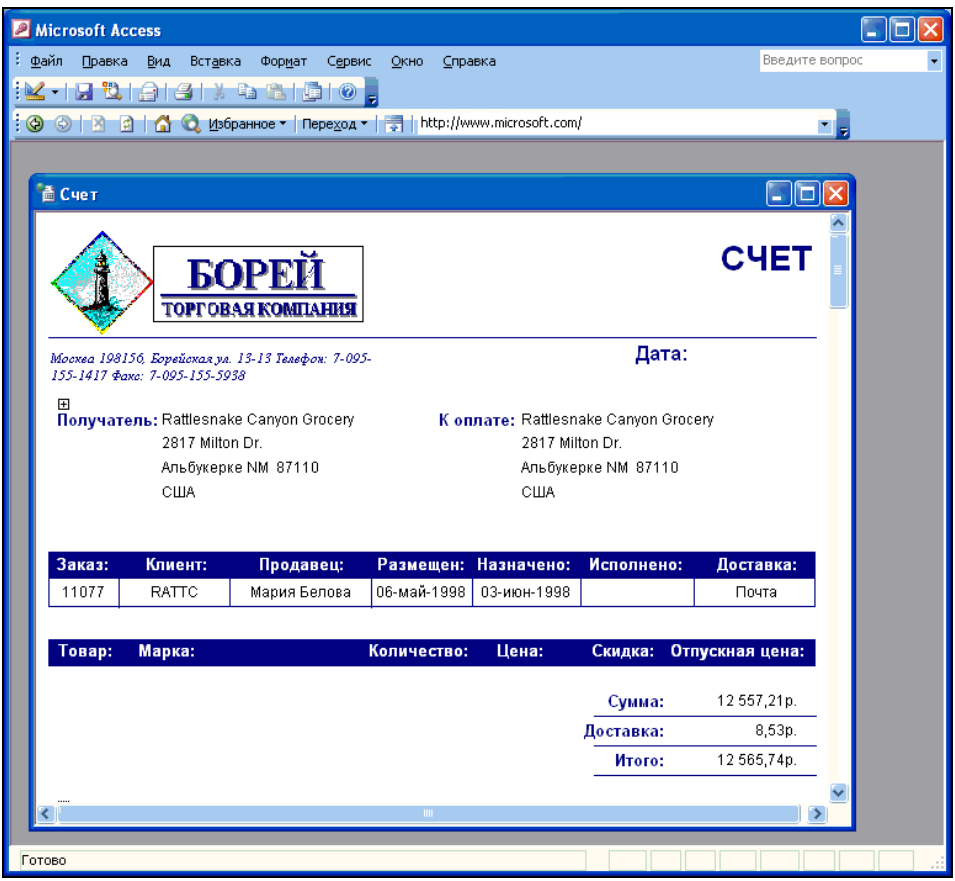


Рис. 11.13. Отчет `Счет`, преобразованный в страницу доступа к данным

Замечание

Страницы доступа к данным также можно сохранить в формате HTML приложений — файлов с расширением hta (HTML Application). Этот формат позволяет использовать технологию динамического HTML — DHTML (Dynamic HTML) "вне программы просмотра" для создания самостоятельных приложений. С помощью HTA можно разрабатывать приложения как обычные Web-страницы в формате DHTML с применением сценариев, с другой стороны, это позволяет включать в разрабатываемое приложение то содержимое, которое изначально разрабатывалось для Web. Подробнее о формате HTA можно узнать на Web-узле Microsoft MSDN.

Внешний вид полученной таким способом страницы доступа к данным практически не будет отличаться от внешнего вида исходного объекта. К тому же на страницах доступа к данным могут появиться дополнительные элементы управления, позволяющие работать с записями. Например, на странице доступа к данным, полученной из отчета `Счет` (Invoice) базы данных `Борей` (Northwind), есть элементы управления, позволяющие переходить между записями и сворачивать или разворачивать уровни группировки (рис. 11.13).

Вообще говоря, интерфейс базы данных обычно включает более сложные формы и отчеты: многостраничные, с подчиненными формами и отчетами, со связанными объектами. К сожалению, точных аналогов многостраничных форм, подчиненных форм и отчетов и связанных объектов в страницах доступа к данным Access не существует, однако с помощью программирования и использования элементов ActiveX можно реализовать все необходимое для работы с данными. Кроме того, программирование позволяет снабдить страницу доступа к данным процедурами реакций на события, такие как нажатие кнопки, получение страницей фокуса и многие другие. То есть страницы обладают всеми необходимыми возможностями, которые предоставляют формы и отчеты Access.

Страницы доступа к данным

Страницы доступа к данным (Data Access Pages) — это специальный тип Web-страниц, предоставляющий пользователям Web интерфейс форм и отчетов Access для доступа к данным из базы данных Microsoft Access. Страницы доступа к данным представляют собой текстовые файлы с исходным кодом, соответствующим расширенному стандарту HTML — XML (Extended Markup Language). Страницы доступа к данным поддерживают возможности DHTML.

Внимание!

Страницы доступа к данным поддерживаются программой Internet Explorer версии 4.0 или более поздней. Полная поддержка всех возможностей реализована в версии 5.0.

Страницы доступа к данным появились как новый тип объектов базы данных в Access 2000. В отличие от других объектов, они не хранятся в файле базы данных, а хранятся в виде отдельных файлов HTML. В базе данных содержатся только ярлыки, указывающие на эти файлы. Список ярлыков страниц доступа к данным отображается в окне базы данных с помощью щелчка по ярлыку **Страницы** (Pages) на панели объектов в окне базы данных. Работать со страницей доступа к данным можно либо

открыв ее в программе просмотра Интернета, такой как Internet Explorer, либо открыв ее в приложении Access (начиная с версии Access 2000). В последнем случае страница доступа к данным может быть открыта в двух режимах: в *режиме Просмотра* и в *режиме Конструктора*. В режиме Конструктора можно изменять содержимое страницы: ее элементы, текст и оформление, а также настраивать свойства страницы и ее элементов. Программирование страницы доступа к данным осуществляется в Редакторе сценариев. Открыть Редактор сценариев можно выделив ярлык страницы в окне базы данных и нажав кнопку **Редактор сценариев** (Microsoft Script Editor) на панели инструментов, или выбрав команду **Сервис | Макрос | Редактор сценариев** (Tools | Macro | Microsoft Script Editor) в стандартном меню Access.

Чтобы открыть страницу в *режиме Просмотра*, используйте кнопку **Открыть** (Open) в окне базы данных или элемент списка **Просмотр страницы** (Page View) у кнопки **Вид** (View) на панели инструментов **Макет страницы** (Page View), или команду меню **Вид | Просмотр страницы** (View | Page View).

Чтобы открыть страницу в *режиме Конструктора*, используйте кнопку **Конструктор** (Design) в окне базы данных или элемент списка **Конструктор** (Design View) у кнопки **Вид** (View) на панели инструментов **Макет страницы** (Page View), или команду меню **Вид | Конструктор** (View | Design View).

Возможности страницы доступа к данным зависят от того, для чего она предназначена. Существует три основных варианта.

- *Составление отчетов в интерактивном режиме*. Страницы доступа к данным этого типа часто используются для объединения и группировки сведений, хранящихся в базе данных, а также для публикации сводок данных. Например, страница может публиковать сводки по продажам для каждого региона, в котором проводятся операции. С помощью маркеров развертывания можно перейти от общей сводки данных, такой как список всех регионов и общий объем продаж по ним, к подробностям по продажам отдельных товаров в каждом регионе. Страница может также содержать кнопки панели инструментов, позволяющие сортировать и фильтровать данные, однако редактировать данные на страницах этого типа невозможно.
- *Ввод данных*. Этот тип страниц доступа к данным используется для просмотра, добавления и редактирования записей или только для добавления записей.
- *Анализ данных*. Страницы доступа к данным этого типа могут включать сводные списки (аналог сводных таблиц Excel), которые позволяют реорганизовывать данные для их анализа различными способами. Страница может содержать диаграммы для анализа тенденций, закономерностей и выполнения сравнений между данными в базе данных или электронные таблицы, в которые можно вводить и редактировать данные, а также использовать формулы для выполнения некоторых вычислений, выполняемых в Microsoft Excel.

Страница доступа к данным непосредственно связана с базой данных. При просмотре в Internet Explorer отображается копия страницы. Поэтому любой отбор данных, сортировка и другие изменения способа отображения данных, включая изменения в сводном списке или электронной таблице, влияют только на копию страницы. Однако изменения в самих данных — изменение значений, добавление или удаление данных — сохраняются в базе данных и впоследствии доступны всем, кто просматривает страницу.

Создание страниц доступа к данным

Для создания страницы доступа к данным существует несколько способов. Можно создать ее "с нуля" в режиме Конструктора, а можно воспользоваться Мастером. С помощью Мастера даже новичок может быстро создать страницы доступа к данным и познакомиться с ними. Кроме того, в Access есть средство для автоматического создания страниц (автостраниц), подобное средствам создания автоформ и автоотчетов. Можно также создать страницу доступа к данным из других объектов базы данных, например, из таблицы, запроса, формы или отчета, путем преобразования *(см. разд. "Преобразование в страницу доступа к данным" этой главы)*.

Автоматическое создание на основе таблицы или запроса

Начнем с простого примера — создание страницы для просмотра и редактирования личной карточки сотрудника. В приложении `Борей` (Northwind) уже есть подобная страница `Сотрудники` (Employees), но она не позволяет добавлять запись для нового сотрудника. Мы рассмотрим, как создать такую страницу. Проще всего воспользоваться средством автоматического создания страницы, когда требуется минимальное участие пользователя:

1. В базе данных `Борей` (Northwind) щелкните по ярлыку **Страницы** (Pages).
2. На панели инструментов базы данных нажмите кнопку **Создать** (New). Появится диалоговое окно **Новая страница доступа к данным** (New Data Access Page) (рис. 11.14).
3. Выделите элемент **Автостраница: в столбец** (AutoPage: Columnar). В раскрывающемся списке выберите имя таблицы `Сотрудники` (Employees), которая будет использована в качестве источника данных для страницы, и нажмите кнопку **OK**.

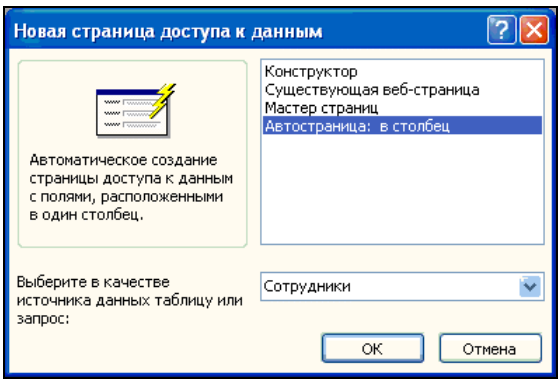


Рис. 11.14. Диалоговое окно **Новая страница доступа к данным**

Замечание

Обратите внимание, в диалоговом окне, представленном на рис. 11.14, отображен список элементов, соответствующих всем стандартным способам создания страниц

доступа к данным: в режиме Конструктора; из существующей страницы Web; с помощью Мастера страниц; с помощью средства автоматического создания страниц (так называемых автостраниц). Чтобы создать страницу доступа к данным одним из этих способов, выделите соответствующий элемент в этом списке.

По окончании работы мастера на экране отобразится созданная страница доступа к данным в режиме Просмотра (рис. 11.15).

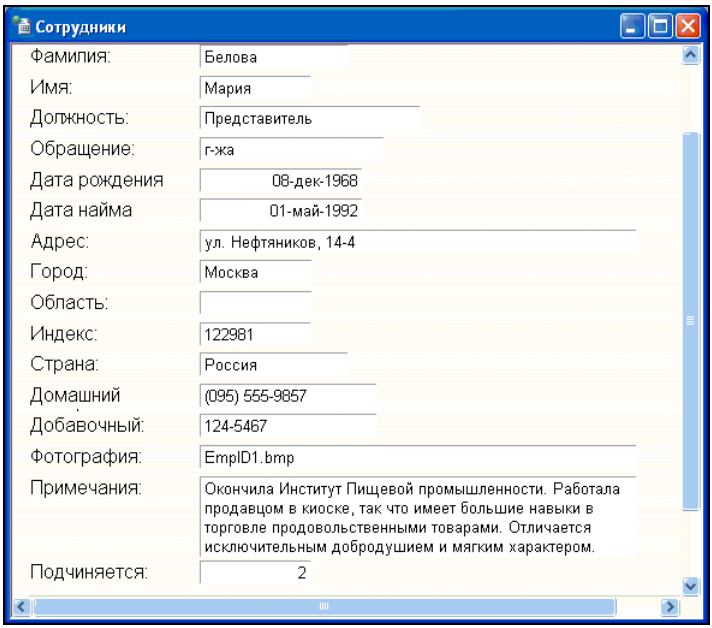


Рис. 11.15. Автостраница, созданная на основе таблицы `Сотрудники`

Если страница содержит большое количество полей, так что не все из них могут поместиться на экране, используйте полосы прокрутки для просмотра всех частей страницы. Внизу столбца с полями имеется панель перехода по записям.

Чтобы сохранить страницу, выполните одно из следующих действий:

- нажмите кнопку **Сохранить** (Save) на панели инструментов **Макет страницы** (Page View);
- выберите команду **Файл | Сохранить** (File | Save);
- закройте страницу доступа к данным, щелкнув по кнопке с изображением крестика в правом верхнем углу окна страницы, и подтвердите ее сохранение, нажав кнопку **Да** (Yes).

В любом случае после этих действий появится диалоговое окно для выбора папки и имени файла для сохранения страницы в формате HTML. Введите имя файла `Личная карточка.htm`. Нажмите кнопку **Сохранить** (Save). Возможно, отобразится предупреждающее диалоговое окно Access о том, что для подключения страницы к источнику данных (базе данных) задан абсолютный путь (рис. 11.16). Чтобы подклю-

чение работало в сети, требуется указать путь к источнику данных в формате UNC. (Как это сделать, описано в *разд. "Подключение к источнику данных" этой главы.*) Установите флажок, чтобы это окно больше не отображалось. Нажмите кнопку **OK**.

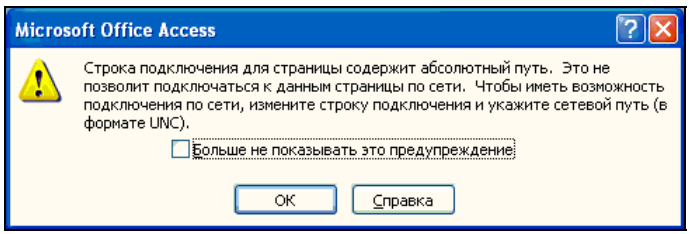


Рис. 11.16. Предупреждение об абсолютном пути в строке подключения

Страница будет сохранена в файле с заданным именем, а в базе данных появится одноименный ярлык для доступа к ней (рис. 11.17).

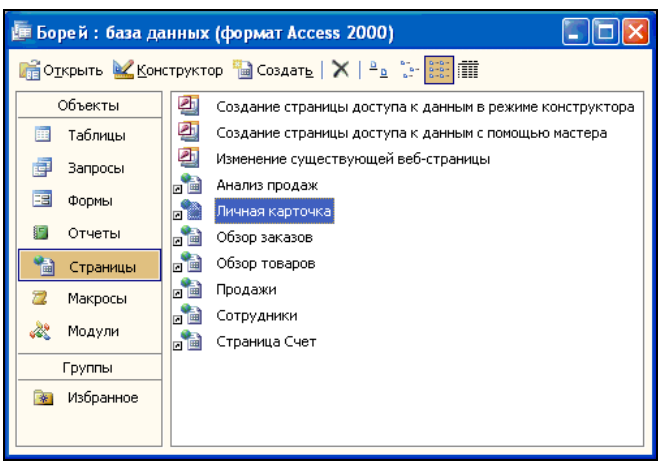


Рис. 11.17. Ярлык для доступа к странице **Личная карточка**

Создание с помощью Мастера

Описанное в предыдущем разделе средство создания автостраниц позволяет быстро создать макет страницы на основе единственной таблицы или запроса в качестве источника данных. Однако созданные таким способом страницы достаточно примитивны и, как правило, требуют дальнейшего усовершенствования в режиме Конструктора. Эти страницы можно сравнить с самыми простыми ленточными формами и отчетами или автоформами и автоотчетами в столбец.

Более широкие возможности по автоматическому созданию страниц предоставляет Мастер страниц доступа к данным. С его помощью можно создать страницу на основе полей из нескольких таблиц или запросов, использовать группировку и сортировку записей.

Например, чтобы создать страницу доступа к данным, аналогичную отчету Продажи по сотрудникам и странам (Sales by Employees and Countries), воспользуемся Мастером:

1. В базе данных Борей (Northwind) щелкните по ярлыку **Страницы** (Pages).
2. Дважды щелкните по ярлыку **Создание страницы доступа к данным с помощью мастера** (Create data access page by using wizard).
3. Появится первое диалоговое окно Мастера. На этом шаге требуется выбрать источник данных для страницы: одно или несколько полей из одной или нескольких таблиц или запросов в базе данных (рис. 11.18).

   Для нашего примера источником данных служит следующий набор полей:

   - поля Страна (Country) и Фамилия (LastName) из таблицы Сотрудники (Employees);
   - поля ДатаИсполнения (ShippedDate) и КодЗаказа (OrderID) из таблицы Заказы (Orders);
   - поле ПромежуточнаяСумма (Subtotal) из запроса Промежуточная сумма заказа (Order Subtotals).

   Чтобы добавить эти поля на страницу, сначала выберите в раскрывающемся списке элемент **Таблица: Сотрудники** (Table: Employees). Затем выделяйте в списке **Доступные поля** (Available Fields) по одному полю из полей Страна и Фамилия и переносите их с помощью кнопки > в список **Выбранные поля** (Selected Fields). Далее аналогичным образом выберите необходимые поля из таблицы Заказы и запроса Промежуточная сумма заказа. Нажмите кнопку **Далее** (Next).

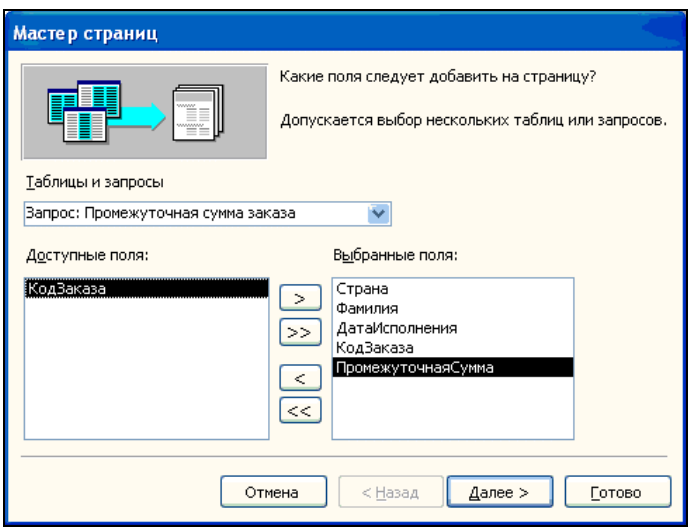


Рис. 11.18. Первое диалоговое окно Мастера страниц доступа к данным

4. Появится диалоговое окно для добавления уровней группировки (рис. 11.19). Если на странице доступа к данным используется группировка, правка сгруппиро-

ванных данных с ее помощью будет невозможна, т. к. такие данные становятся доступны только для чтения. Чтобы задать группировку данных по странам, выделите элемент Страна, соответствующий полю Страна таблицы Сотрудники и нажмите кнопку >. Чтобы задать группировку данных по сотрудникам внутри групп данных о продажах по странам, необходимо добавить второй уровень группировки по элементу Фамилия. Добавленные уровни группировки отобразятся на эскизе в правой части диалогового окна (см. рис. 11.19). Нажмите кнопку **Далее** (Next).

Замечание

Чтобы поменять порядок следования уровней группировки, используйте кнопки с изображением стрелок вверх и вниз, предварительно щелкнув на эскизе по полю группировки, которое надо переместить (выбранное таким образом поле группировки выделяется жирным шрифтом). Чтобы задать интервал значений группируемых полей, нажмите кнопку **Группировка** (Grouping Options) внизу диалогового окна.

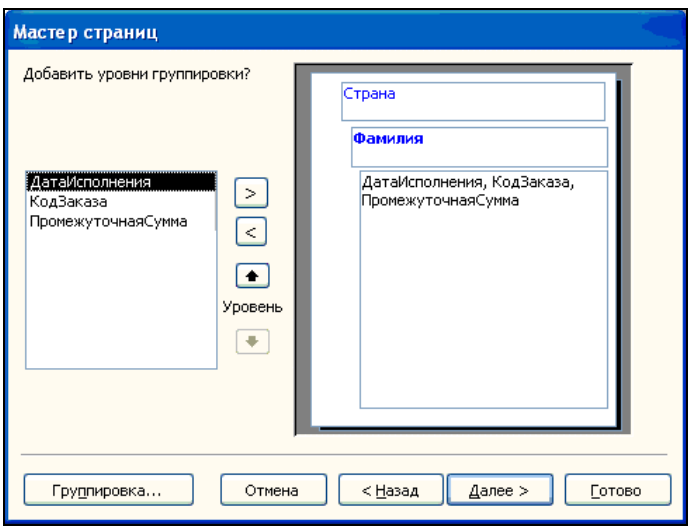


Рис. 11.19. Второе диалоговое окно Мастера страниц доступа к данным

5. Появится следующее диалоговое окно Мастера, позволяющее задать порядок сортировки. Это диалоговое окно, как и предыдущее, похоже на соответствующее диалоговое окно Мастера форм или отчетов, поэтому вы уже знаете, как им пользоваться. Как и Мастера форм и отчетов, Мастер страниц доступа к данным позволяет использовать до четырех вложенных уровней сортировки в одном из двух направлений: по возрастанию и по убыванию. В нашем примере мы используем сортировку по возрастанию по полю ДатаИсполнения и сортировку по убыванию по полю ПромежуточнаяСумма (рис. 11.20). Нажмите кнопку **Далее** (Next).
6. Появится последнее диалоговое окно Мастера, в котором требуется ввести название страницы доступа к данным (это название заголовка и ярлыка страницы в базе данных) и выбрать способ отображения созданной страницы. Введите в поле

ввода название Продажи по сотрудникам и странам и выберите переключатель **Открыть страницу** (Open the page) (рис. 11.21). Нажмите кнопку **Готово** (Finish).

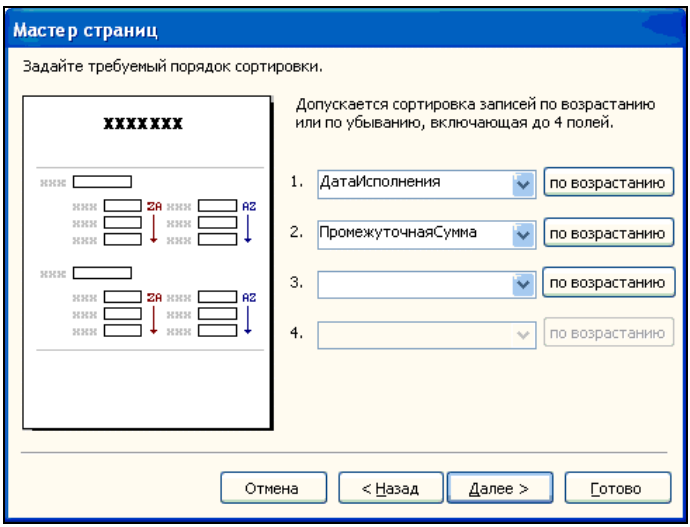


Рис. 11.20. Третье диалоговое окно Мастера страниц доступа к данным

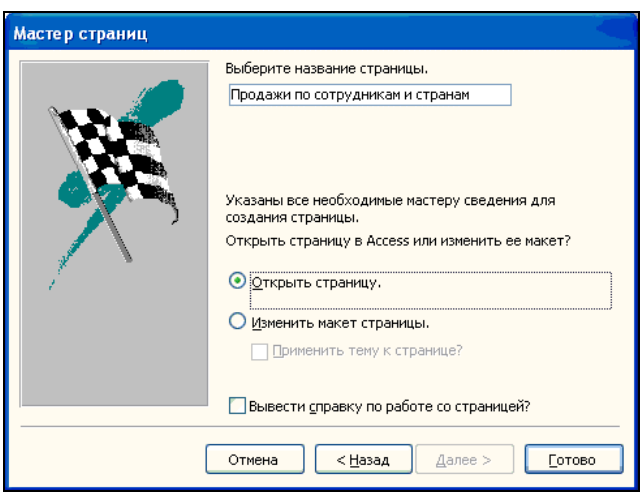


Рис. 11.21. Последнее диалоговое окно Мастера страниц доступа к данным

7. На этом сбор сведений о странице завершается и начинается ее автоматическое формирование. После небольшой паузы на экране появится созданная Мастером страница в режиме Просмотра (рис. 11.22). Чтобы отобразить список сотрудников для данной страны и список заказов, обработанных этими сотрудниками, используйте кнопки развертывания структуры. С помощью панелей перемеще-

ния по записям в группе можно выбрать интересующую вас страну, сотрудника или заказ.

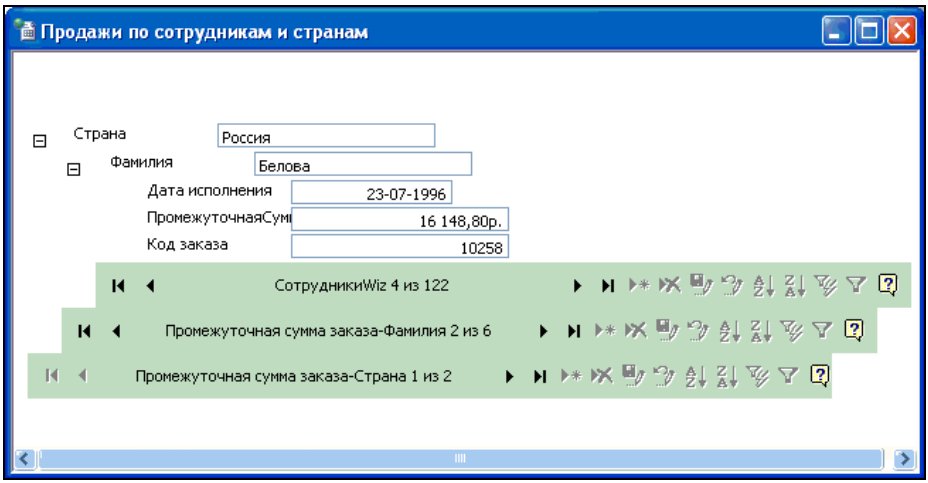


Рис. 11.22. Страница **Продажи по сотрудникам и странам**, созданная Мастером

8. Нажмите кнопку **Сохранить** (Save) на панели инструментов **Макет страницы** (Page View). В появившемся диалоговом окне укажите папку и имя файла для сохранения страницы доступа к данным (например: `Продажи по сотрудникам и странам.htm`) и нажмите кнопку **Сохранить** (Save).

Итак, минимальными усилиями мы получили страницу доступа к данным, предоставляющую пользователям Web возможность просмотреть информацию о продажах. Благодаря группировке записей на странице поиск необходимой информации осуществляется быстро и удобно:

1. С помощью *кнопок перехода по записям* во внешней группе (самая нижняя на странице панель кнопок) следует выбрать необходимую страну, отобразить группу сотрудников из этой страны, щелкнув по *маркеру развертывания группы* (кнопка с изображением плюса).
2. Затем нужно с помощью кнопок перехода средней группы выбрать сотрудника в данной стране и отобразить группу обработанных сотрудником заказов, щелкнув по маркеру развертывания группы.
3. И наконец, выбрать интересующий заказ с помощью кнопок перехода по записям во внутренней группе данных.

Таким образом можно выяснить, в каких странах работает предприятие, какие сотрудники там заняты и какие заказы ими обработаны. Практически то же самое могут сделать пользователи Access, работающие с отчетом `Продажи по сотрудникам и странам` (Sales by Employees and Countries).

Однако эта страница недостаточно удобна и еще не готова к публикации. Она обладает целым рядом недостатков.

- Для выбранной страны нужно сразу отобразить список всех (или, по крайней мере, нескольких) работающих в ней сотрудников.
- В группе с информацией о заказах следует отобразить информацию сразу о нескольких заказах в виде таблицы.
- Требуется изменить оформление и надписи на странице.
- Не хватает вычисляемых полей с итоговыми суммами по заказам.
- И наконец, следует изменить способ отображения данных. В представленном варианте данные отображаются в текстовых полях, которые выглядят как поля для ввода данных и дают иллюзию возможности их правки. Во-первых, правка данных в страницах с группировкой запрещена. Во-вторых, правка данных здесь не нужна, поскольку назначение этой страницы подразумевает лишь поиск информации.

Подключение существующей Web-страницы к базе данных

В отличие от форм, отчетов и других объектов базы данных, страницы доступа к данным не хранятся в файле базы данных или проекта Access (это файлы с расширением mdb, adp, mde или ade), а хранятся в отдельных файлах (страницах HTML, HTA или ASP). В файле Access сохраняются только ярлыки, ссылающиеся на эти файлы.

Ярлык страницы доступа к данным можно удалить. Для этого нужно его выделить в окне базы данных и нажать клавишу <Delete>. Например, удалите ярлык страницы `Личная карточка` из базы данных `Борей` (Northwind). Появится диалоговое окно (рис. 11.23), позволяющее выбрать вариант удаления ярлыка страницы: вместе с файлами страницы (кнопка **Удалить ссылку и файлы** (Delete Link and Files)) или без них (кнопка **Удалить только ссылку** (Delete Link Only)). Нажмите кнопку **Удалить только ссылку**.

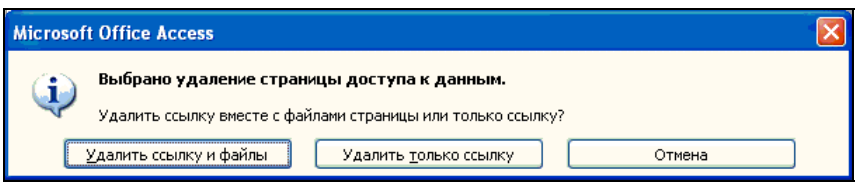


Рис. 11.23. Выбор варианта удаления страницы доступа к данным

Ярлык в базе данных исчезнет, но файл Личная карточка.htm останется на диске нетронутым.

Ярлык страницы в базе данных можно переименовать. Для этого просто щелкните по нему, когда он выделен.

Если вы случайно удалили ярлык к странице доступа к данным, или хотите подключить страницу доступа к данным к другой базе данных, или хотите подключить обычную HTML-страницу для ее изменения в режиме Конструктора страниц досту-

па к данным, откройте базу данных, щелкните по ярлыку **Страницы** (Pages) и выполните одно из действий:

- дважды щелкните по ярлыку **Изменение существующей Web-страницы** (Edit Web page that already exists);
- нажмите кнопку **Создать** (New) в окне базы данных, затем в появившемся диалоговом окне выберите элемент **Существующая Web-страница** (Existing Web page).

После этого появится диалоговое окно **Поиск веб-страницы** (Locate Web Page), такое же, как стандартное окно открытия файла. Выберите папку и укажите файл Web-страницы с расширением htm или html или файл ярлыка Web-страницы с расширением url, нажмите кнопку **Открыть** (Open).

Замечание

Чтобы открыть страницу только для чтения в базе данных Access, у этой кнопки раскройте список и выберите элемент **Открыть для чтения** (Open Read-Only).

Есть еще одна особенность работы с ярлыками страниц доступа к данным. Если в базе данных есть такой ярлык, а страница доступа к данным была перемещена или удалена, при попытке открыть страницу двойным щелчком по ярлыку будет отображено диалоговое окно (рис. 11.24) с просьбой указать новое местоположение страницы. Нажмите кнопку **Обновить ссылку** (Update Link) и укажите папку и файл страницы.

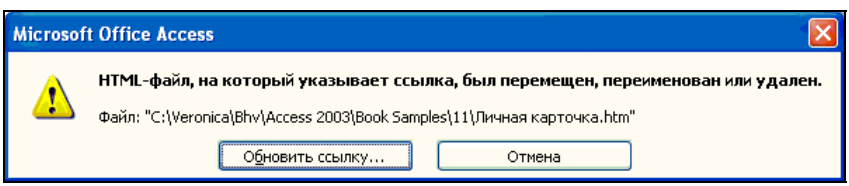


Рис. 11.24. Диалоговое окно с просьбой указать новое местоположение страницы

С другой стороны, может быть перемещена или удалена база данных, к которой подключена страница доступа к данным. В этом случае в момент открытия страницы доступа к данным в Access в режиме Просмотра отобразится диалоговое окно для выбора источника данных (см. рис. 11.27). Если источник данных будет указан неверно, в результате будет отображена страница, не содержащая никаких данных, а при попытке обновить соединение отобразится сообщение об ошибке (рис. 11.25).

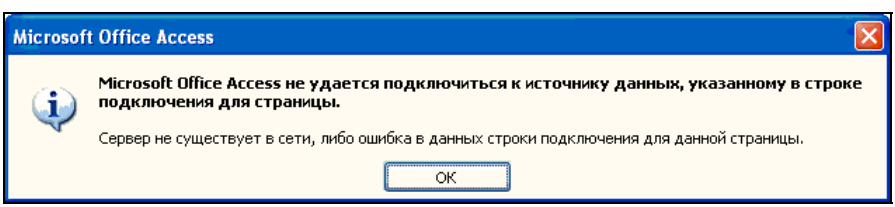


Рис. 11.25. Сообщение об ошибке при подключении к источнику данных

В этой ситуации необходимо исправить параметры подключения, о чем рассказывается в следующем разделе.

Подключение к источнику данных

Настройка свойств подключения к источнику данных для страницы доступа к данным выполняется с помощью специального диалогового окна. Это может потребоваться, если нужно изменить источник данных, к которому подключена страница, или если источник данных требуется переместить в другую папку, например, чтобы предоставить к нему общий доступ в сети или опубликовать его вместе со страницей, переместив их на Web-сервер организации.

Чтобы отобразить диалоговое окно свойств подключения:

1. В режиме Конструктора страницы доступа к данным откройте Список полей (кнопка **Список полей** (Field List) на панели инструментов **Макет страницы** (Page View)).
2. Нажмите кнопку **Свойства подключения для страницы** (Page connection properties), которая находится вверху в окне **Список полей** (Field List).

Появится диалоговое окно, позволяющее настроить параметры подключения к источнику данных для этой страницы (рис. 11.26).

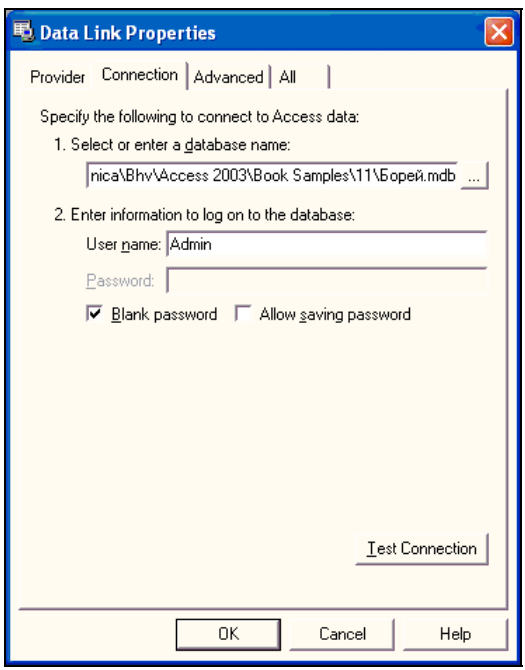


Рис. 11.26. Диалоговое окно свойств связи с данными

На вкладке **Connection** (Подключение) в соответствующем поле ввода задайте правильное имя и местоположение файла с данными (можно использовать для этого

кнопку "...", отображающую стандартное окно открытия файла). Проверьте возможность подключения, нажав кнопку **Test Connection** (Проверить подключение). Если все в порядке, нажмите кнопку **OK** и еще раз **OK**, чтобы закрыть это окно. Перед закрытием страницы доступа к данным сохраните ее, чтобы сохранить установленные параметры подключения.

Замечания

Сведения о подключении страницы сохраняются в ее свойстве **ConnectionString** (Строка подключения). Его значение можно изменять непосредственно в окне свойств страницы доступа к данным (см. рис. 11.35), а не только в окне свойств подключения.

При создании страницы доступа к данным или подключении Web-страницы, созданной не в Microsoft Access, когда уже имеется открытая база данных Microsoft Access или проект Microsoft Access, страница автоматически подключается к открытой базе данных, а сведения о подключении сохраняются в свойстве **ConnectionString** страницы.

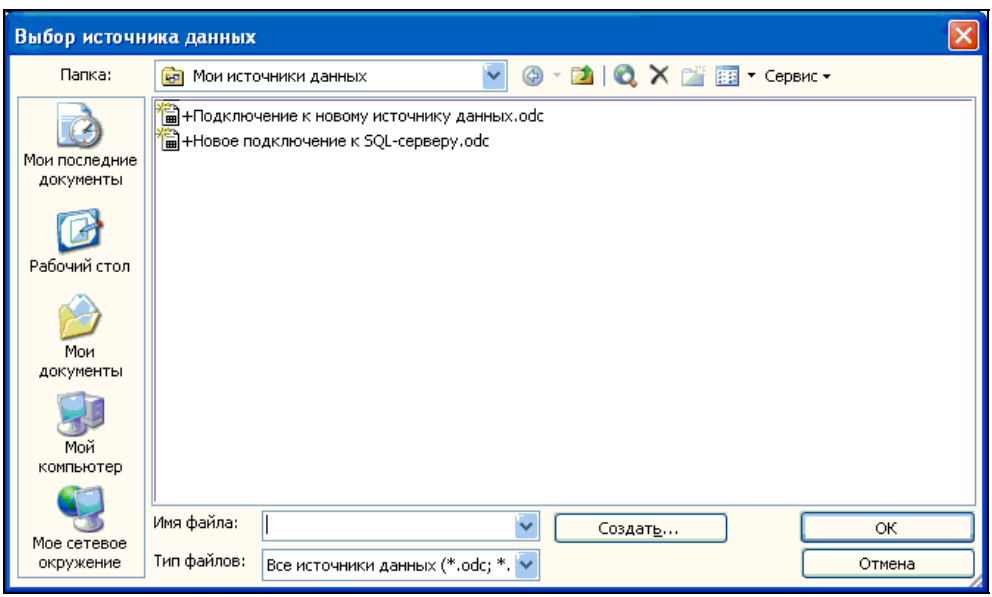


Рис. 11.27. Диалоговое окно **Выбор источника данных**

Параметры подключения к источнику данных могут быть заданы и в файле подключения, указанном в свойстве **ConnectionFile** (Файл подключения) в окне свойств страницы. Чтобы создать такой файл, можно воспользоваться Построителем. Для этого нажмите кнопку "..." справа от поля свойства **ConnectionFile**. Появится диалоговое окно, представленное на рис. 11.27. Выберите папку, укажите существующий файл подключения или создайте его с помощью Мастера подключения данных (Data Connection Wizard), который вызывается с помощью кнопки **Создать** (New Source). Файл подключения содержит строку подключения. Ее можно отредактиро-

вать, открыв этот файл в текстовом редакторе, например Блокноте Windows, и найдя тег `<odc:ConnectionString>`.

Разработка страницы доступа к данным в режиме Конструктора

Страницу доступа к данным можно создать самостоятельно, без помощи Мастеров, в режиме Конструктора. Кроме того, в режиме Конструктора можно отредактировать существующую страницу доступа к данным, созданную ранее любым способом.

В режиме Конструктора можно выполнить следующие действия со страницей доступа к данным:

- добавить или удалить элементы управления;
- изменить свойства элементов управления;
- изменить свойства страницы доступа к данным;
- изменить оформление страницы: фон, цвет и выравнивание текста, начертание и размер шрифтов;
- добавить или удалить графические элементы, изменить их размер;
- добавить, удалить или изменить таблицы;
- разместить на странице гиперссылки, назначить их тексту или рисункам.

То есть для работы со страницами доступа к данным Access предоставляет все стандартные средства, которые обеспечивают обычные редакторы страниц Web, включая возможность программирования на языках сценариев *(см. разд. "Программирование страниц доступа к данным" этой главы)*.

Чтобы создать или отредактировать страницу доступа к данным в режиме Конструктора в окне базы данных, нажмите кнопку **Страницы** (Pages) и выполните одно из действий:

- дважды щелкните по ярлыку **Создать страницу доступа к данным в режиме конструктора** (Create data access page in Design view);
- нажмите кнопку **Создать** (New) в окне базы данных и в появившемся диалоговом окне выберите элемент списка **Конструктор** (Design View). Нажмите кнопку **ОК**;
- выделите ярлык существующей страницы и нажмите кнопку **Конструктор** (Design) в окне базы данных. Страница будет открыта в режиме Конструктора.

При создании страницы может отобразиться окно с предупреждением, что новые страницы будут сохранены в новом формате Access и могут быть доступны только в режиме Просмотра в предыдущей версии Access 2000 при условии, если на компьютере установлены Web-компоненты Microsoft Office XP. Нажмите кнопку **ОК**.

Структура страницы доступа к данным

Страница доступа к данным имеет одну общую область для размещения всех данных — *тело страницы*. В тело страницы могут быть добавлены другие элементы, связанные и несвязанные с данными, а также разделы. При создании страницы в режиме Конструктора в нее обычно добавляется специальный элемент **НесвязанныйРаздел** (SectionUnbound), в котором могут быть размещены другие элементы.

Если страница содержит данные, то каждая группа данных представляется одним или несколькими разделами: заголовок группы, примечание группы, область подписей элементов и область панели кнопок перехода.

(См. разд. "Сортировка и группировка данных" этой главы.)

Элементы управления

Подобно формам и отчетам Access страница доступа к данным может содержать большое количество различных *элементов управления* (control objects). Набор элементов может варьироваться в зависимости от цели создания страницы. Наиболее часто используемые элементы управления приведены в табл. 11.4.

В режиме Конструктора страницы доступа к данным доступна панель инструментов **Панель элементов** (Toolbox), содержащая кнопки, позволяющие парой щелчков мышью создать на странице любые элементы управления. Работа с элементами управления на странице полностью аналогична работе с элементами управления на форме или отчете.

Таблица 11.4. *Элементы управления, использующиеся на страницах доступа к данным*

Элемент управления	Описание
Текстовое поле или поле ввода (Text Box)	Отображают существующие данные из базы данных или, если разрешено, принимают вводимые пользователем новые данные для хранения в базе данных
Связанный текст (Bound Span)	Текстовое поле, отображающее текст в формате HTML, который нельзя редактировать. Этот элемент может быть привязан к полю в таблице источника данных. В этот элемент данные загружаются быстрее, чем в текстовое поле, поэтому они более эффективны на страницах, предназначенных только для отображения данных
Надпись (Label)	Текстовая надпись не может быть связана с источником данных, но может быть привязана к другому элементу управления, как часть графического элемента
Группа переключателей (Option Group), переключатель (Option Button), флажок (Check Box)	Элементы управления, аналогичные подобным элементам в формах Access. Эти элементы можно привязать к источнику данных
Список (List Box), раскрывающийся список (Dropdown List))	Элементы управления, аналогичные подобным элементам в формах Access. Эти элементы можно привязать к источнику данных. Элементы списка могут формироваться из подключенной таблицы или из фиксированного набора значений. Список может содержать несколько столбцов значений
Кнопка (Command Button)	Кнопка, подобная кнопкам в формах. Для кнопки можно определить процедуры обработки массы разнообразных событий. Так можно управлять поведением страницы. Кроме того, кнопку тоже можно привязать к источнику данных

Таблица 11.4 *(окончание)*

Элемент управления	Описание
Маркер развертывания группы (Expand)	Элемент, позволяющий отобразить/скрыть содержимое группы данных
Панель перехода по записям (Record Navigation)	Позволяет быстро выполнять переходы между записями либо добавлять, удалять, сохранять, отменять изменения, сортировать или фильтровать записи. На странице доступа с группами каждая группа может иметь собственную панель перехода по записям, расположенную в нижней части развернутой группы. Если панель на странице не включает в себя некоторые из кнопок, показанных на следующем рисунке, или включает кнопки, не перечисленные здесь, значит она была специально настроена разработчиком страницы

Еще один элемент управления — *несвязанный раздел* (SectionUnbound) — почти всегда присутствует на странице. Он автоматически добавляется на страницу доступа к данным при ее создании. К нему привязывается источник данных — набор записей, сформированный из таблицы или запроса. На странице может быть только один несвязанный раздел. Чтобы связать страницу с несколькими источниками данных, необходимо добавлять на нее специальные элементы ActiveX. Чтобы добавить на страницу несвязанный раздел, если его нет, выберите команду **Вставка | Несвязанный раздел** (Insert | Unbound Section). Привязанный источник данных указывается в свойстве **RecordSource** (Источник записей) этого раздела.

Кроме перечисленных элементов управления на странице могут быть рисунки, гиперссылки и прочие элементы.

Замечание

Строго говоря, список элементов управления, которые можно использовать на страницах доступа к данным, не ограничен, поскольку можно применять любые элементы управления ActiveX, число которых растет с каждым днем.

Добавление, удаление и перемещение элементов управления

Для добавления новых полей на страницу доступа к данным в режиме Конструктора предназначено окно **Список полей** (Field List), представленное на рис. 11.28. Если оно скрыто, его можно отобразить с помощью команды меню **Вид | Список полей** (View | Field List) или кнопки **Список полей** (Field List) на панели инструментов **Макет страницы** (Page Design).

В окне **Список полей** отображаются списки всех таблиц и запросов открытой базы данных. Еще одно окно — **Структура данных** (Data Outline) (см. рис. 11.28) — имеет похожий вид, но отображает структуру данных, добавленных на страницу: уровни группировки и набор записей, что является источником данных для открытой страницы доступа к данным. Это окно можно отобразить командой меню **Вид | Структура данных** (View | Data Outline) или кнопкой **Структура данных** (Data Outline) на панели инструментов **Макет страницы** (Page Design). В этих окнах для каждой таблицы, запроса или набора записей можно отобразить список полей. Списки таблиц,

запросов и их полей представлены в виде иерархии, подобной иерархии папок и файлов, отображающейся в окне приложения Проводник Windows. Каждый уровень иерархии можно раскрыть с помощью значка "+" или свернуть с помощью значка "–", отображенного рядом с элементом верхнего уровня иерархии.

1. В диалоговом окне **Список полей** (Field List) раскройте список таблиц, щелкнув по значку "+" напротив надписи **Таблицы** (Tables).
2. Раскройте список полей таблицы `Сотрудники` (Employees), щелкнув по значку "+" напротив имени этой таблицы.
3. С помощью полосы прокрутки переместитесь по списку полей и найдите поле, которое нужно добавить на страницу, выделите его, щелкнув левой кнопкой мыши.
4. Нажмите кнопку **Добавить на страницу** (Add To Page) или перетащите поле в нужное место на странице с помощью мыши.

На странице появится текстовое поле, привязанное к выбранному полю, которое становится для этого элемента управления источником данных.

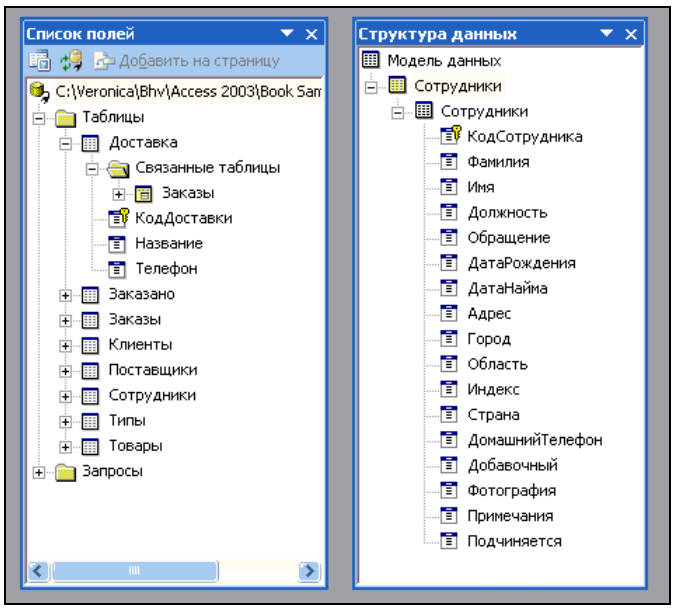


Рис. 11.28. Окна **Список полей** и **Структура данных**

Кроме полей на страницу доступа к данным можно добавлять другие элементы управления, однако если новый элемент создается не автоматически, то в этом случае источник данных придется назначить полю вручную в окне свойств.

Полученная нами страница `Личная карточка`, описанная в *разд. "Автоматическое создание на основе таблицы или запроса" этой главы*, имеет поля `Обращение` (TitleOfCourtesy) и `Подчиняется` (ReportsTo), которые отображаются как текстовые поля, а должны отображаться как поля подстановки, т. е. в виде списков с возмож-

ными значениями. Необходимо исправить этот недостаток. Для чего создадим данные списки в режиме Конструктора заново и воспользуемся Мастером создания полей подстановки.

Замечание

Чтобы в режиме Конструктора отображались диалоговые окна Построителей и Мастеров, упрощающих работу, необходимо оставить нажатой кнопку-переключатель **Мастера** (Control Wizards) на панели инструментов **Панель элементов** (Toolbox).

Сначала нужно удалить старые элементы управления:

1. Откройте страницу `Личная карточка` в режиме Конструктора.
2. Проверьте, что кнопка-переключатель **Выбор объектов** (Select Objects), находящаяся на панели инструментов **Панель элементов**, нажата.
3. Выделите на странице текстовое поле `Обращение` (TitleOfCourtesy) (для этого щелкните по нему левой кнопкой мыши).
4. Нажмите клавишу <Delete>.

Точно так же удалите со страницы текстовое поле `Подчиняется` (ReportsTo).

Чтобы добавить на страницу раскрывающийся список `Обращение`:

1. На панели инструментов **Панель элементов** нажмите кнопку **Раскрывающийся список** (Dropdown List).
2. Выберите на странице доступа к данным свободное место для размещения нового элемента управления, щелкните там левой кнопкой мыши и, не отпуская кнопку мыши, растяните прямоугольный контур будущих очертаний элемента до желаемых размеров.
3. Появится диалоговое окно **Создание полей со списком** (Combo Box Wizard), предлагающее выбрать вариант заполнения списка данными: из таблицы или запроса, либо фиксированным набором значений. В нашем примере нужно выбрать переключатель **Будет введен фиксированный набор значений** (I will type in the values that I want). Нажмите кнопку **Далее** (Next).
4. Появится еще одно диалоговое окно Мастера списков. Введите список значений (рис. 11.29) и нажмите кнопку **Далее** (Next).
5. В следующем диалоговом окне требуется выбрать столбец, значения которого будут привязаны к значению списка. Выберите первый столбец **Столбец1** (Col1) и нажмите кнопку **Далее** (Next).
6. Появится последнее диалоговое окно Мастера. Введите в текстовое поле строку `Обращение`: для создания надписи, которая будет отображаться рядом с полем со списком, и нажмите кнопку **Готово** (Finish).

На странице появится изображение поля со списком и надписью около него.

Замечание

Процедура добавления списков на основе данных из таблиц/запросов аналогична описанной процедуре. Разница в том, что, добавив на страницу элемент управления *список* или *раскрывающийся список*, в первом окне Мастера создания списка нужно

выбрать вариант заполнения списка из таблицы или запроса и на следующем шаге указать имя этого источника данных. Помните об ограничении: в источнике элементов списка на странице должно быть не более двух столбцов.

Созданные списки необходимо привязать к соответствующим полям в источнике данных страницы, чтобы в списках отображалось значение для текущей записи. Сделать это можно в окне свойств элемента *(см. разд. "Изменение свойств страницы доступа к данным и ее элементов" этой главы)*.

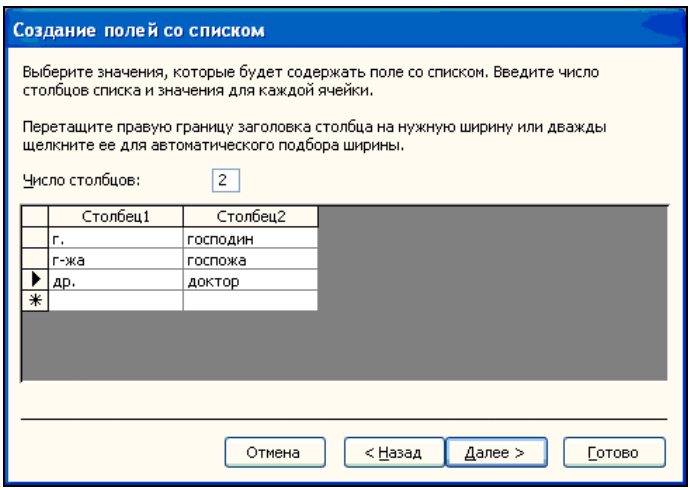


Рис. 11.29. Второе диалоговое окно Мастера создания списков

Еще проще добавить на страницу список `Подчиняется`, для этого просто перетащите поле `Подчиняется` (ReportsTo) из окна списка полей на страницу. В результате на странице будет автоматически создан раскрывающийся список, содержащий в качестве источника элементов списка сохраненный запрос. Этот запрос автоматически копируется из исходного поля подстановки таблицы `Сотрудники` (Employees) (рис. 11.30).

Если вас не устраивает расположение созданного элемента управления, выделите его, затем подведите указатель мыши к его контуру, так, чтобы указатель мыши принял форму крестика с четырьмя стрелками на концах или изображения руки и, удерживая нажатой левую кнопку мыши, переместите элемент управления в нужное место на странице.

Изменить размеры элемента управления можно, растянув с помощью левой кнопки мыши маркеры изменения размеров, отображающиеся в виде маленьких белых квадратиков по периметру выделенного элемента управления.

Если при перемещении элемента вы случайно закрыли один элемент другим, можно поместить верхний элемент на задний план, чтобы отобразить заслоненный элемент:

1. Выделите элемент управления, щелкнув по нему.
2. Выберите команду **Формат | На задний план** (Format | Send to Back).

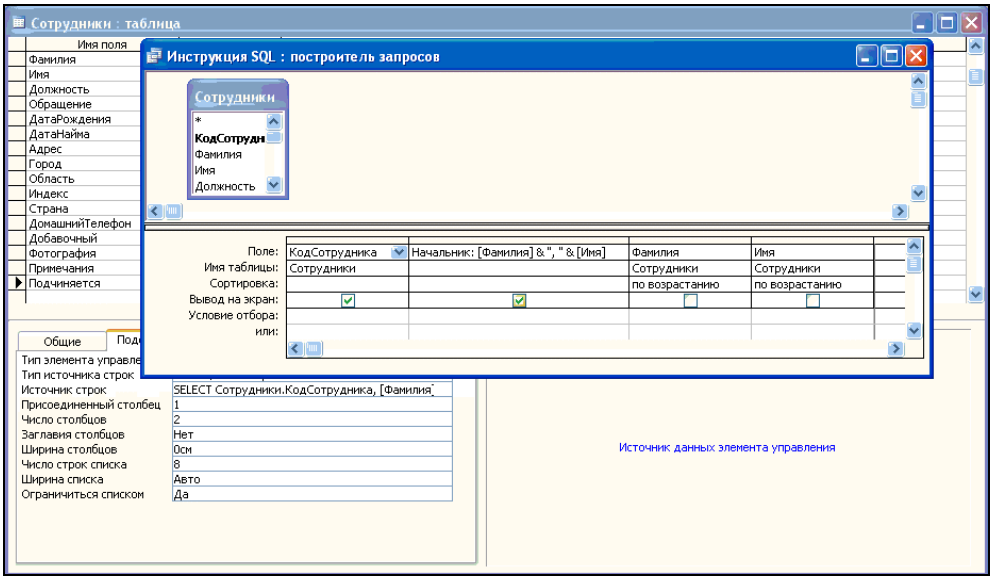


Рис. 11.30. Запрос — источник данных для заполнения списка `Подчиняется`

Если на странице не хватает возможности для перехода по записям в источнике данных для просмотра и правки записей, добавьте на страницу специальный элемент управления *кнопки перехода*, отображающийся в виде панели. Этот элемент управления включает не только специальные кнопки для перехода по записям, но и кнопки для сортировки и фильтрации записей, а также сохранения и отмены изменений для текущей записи, добавления и удаления записи. Чтобы добавить этот элемент управления, воспользуйтесь кнопкой **Кнопки перехода** (Record Navigation) на панели инструментов **Панель элементов**. Размещение и изменение размеров этого элемента управления выполняется так же, как и для других элементов управления.

С помощью элемента **Кнопки перехода** можно также добавлять новые записи. Для этого откройте страницу в режиме Просмотра, нажмите кнопку **Новый** (New) на панели кнопок перехода, затем заполните поля и нажмите кнопку **Сохранить** (Save) на этой же панели (рис. 11.31).

Если вашей странице не нужны все кнопки, изображенные на панели кнопок перехода, вы можете создать собственные элементы управления, с помощью которых будут выполняться необходимые действия. Эти действия придется запрограммировать и назначить их в качестве обработчиков событий элементов. Например, в базе данных `Борей` (Northwind) на странице `Сотрудники` (Employees) для перехода по записям используются текстовые надписи, щелчок по которым вызывает необходимое действие. Если в качестве таких элементов использовать кнопки, то для их создания можно обойтись без программирования. Необходимую процедуру создаст Мастер:

1. Откройте страницу в режиме Конструктора.
2. Нажмите кнопку **Кнопка** (Command Button) на панели инструментов **Панель элементов**.

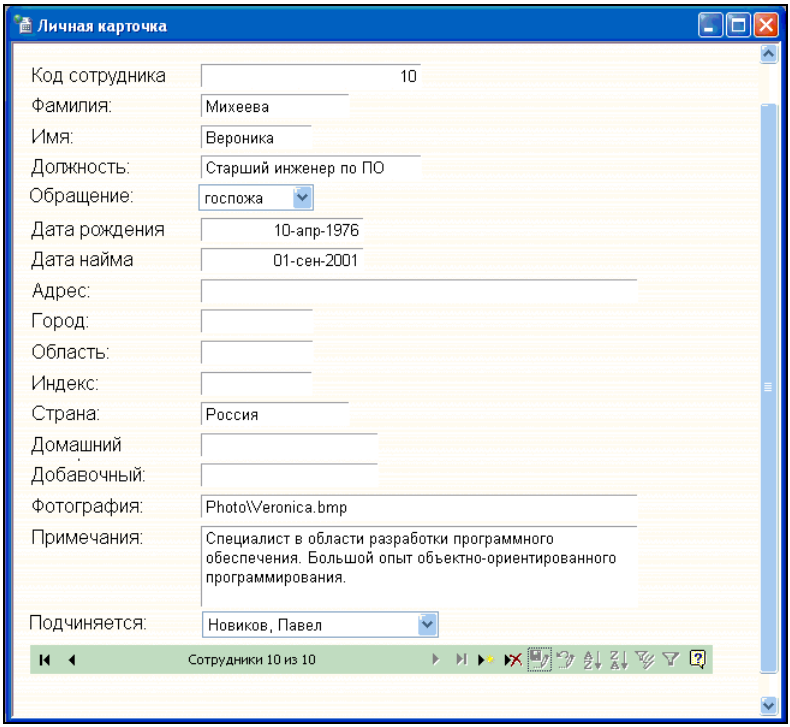


Рис. 11.31. Новая запись на странице **Личная карточка**

3. На свободном участке страницы доступа к данным щелкните левой кнопкой мыши и, не отпуская ее, растяните контур кнопки до желаемых размеров.
4. Отпустите кнопку мыши. Появится диалоговое окно Мастера создания кнопок.
5. В списке категорий выделите элемент **Обработка записей** (Record Operations), а в списке действий — необходимое действие, например элемент **Сохранить запись** (Save Record), чтобы создать кнопку для сохранения измененной записи. Нажмите кнопку **Далее** (Next).
6. Чтобы отобразить на кнопке не рисунок, а текстовую надпись, выберите переключатель **Текст** (Text) и введите текст: `Сохранить`. Нажмите кнопку **Далее**.
7. На последнем шаге Мастера требуется указать имя кнопки для ссылки на нее. Введите имя `Сохранить`. Нажмите кнопку **Готово** (Finish).

Таким образом на странице появится работающая кнопка **Сохранить**.

Сортировка и группировка данных

На страницах доступа к данным возможна группировка записей, как и в отчетах Access. Группы объединяют данные в наборы. Например, записи заказов на странице доступа к данным могут быть сгруппированы по странам. Название каждой страны — это заголовок группы. В зависимости от того, как разработана страница, на ней либо отображаются все группы (обычно со скрытием входящих в них записей),

либо ни одна конкретная группа не отображается до тех пор, пока не будет выбрана в списке доступных групп. Если на странице отображаются все группы, то можно отобразить все записи конкретной группы, щелкнув маркер ее развертывания. В нижней части каждой развернутой группы может находиться собственная панель перехода по записям. Редактировать данные на уровне группировки, имеющем поле группировки, невозможно.

Замечание

В страницах доступа к данным, как и в формах и отчетах, можно также фильтровать данные и создавать вычисляемые поля. Все это выполняется аналогично тому, как было описано в главах о формах и отчетах. Поэтому в этой главе мы не будем подробно рассматривать эти вопросы. При необходимости обратитесь к справке Access.

В режиме Конструктора можно задать необходимую группировку записей. Чтобы добавить уровень группировки, выделите элемент, к которому привязан источник данных (поле, связанный текст и пр., назовем его *элементом группировки*), и нажмите кнопку **Повысить уровень** (Promote) на панели инструментов **Макет страницы** (Page Design). Чтобы сгруппировать данные не по одному полю, а по всем полям из таблицы, которая его содержит, нажмите кнопку **Группировать по таблицам** (Group by Table). Чтобы убрать группировку или соединить два раздела группировки в один, выделите элемент группировки (поле или другой элемент) и нажмите кнопку **Понизить уровень** (Demote).

На панели инструментов **Панель элементов** (Toolbox) присутствует кнопка **Развернуть** (Expand), позволяющая добавить на страницу элемент управления в виде кнопки с изображением плюса или минуса, который называется *маркером развертывания/свертывания группы*. С помощью этого элемента можно развернуть или свернуть список элементов одной группы.

При добавлении на страницу нового уровня группировки появляется новый раздел, который называется *заголовком группы*. Автоматически туда помещаются элемент группировки и маркер развертывания. Выделить этот раздел можно щелчком мыши по его заголовку. Выделив раздел, можно изменить его размеры с помощью мыши или настроить свойства группы. Открыть окно свойств уровня группы (рис. 11.32) можно, щелкнув по заголовку группы правой кнопкой и выбрать в контекстном меню команду **Свойства уровня группы** (Group Level Properties). В этом же контекстном меню отображаются команды, позволяющие отобразить или скрыть тот или иной раздел группы. Если снять флажок **Верхний колонтитул** (Header), будет удален основной раздел — заголовок группы, и все связанные с ним разделы.

Рассмотрим изменение свойств группировки на примере страницы `Продажи по сотрудникам и странам`, описанной в *разд. "Создание с помощью Мастера"*. Прежде всего исправим недостатки, полученные вследствие ее автоматического создания (проведенного в целях демонстрации):

1. Удалите со страницы имеющиеся уровни группировки.
2. В целях повышения быстродействия страницы удалите все текстовые поля и добавьте вместо них элементы типа **Связанный текст** (Bound Span) с помощью соответствующей кнопки на панели инструментов **Панель элементов**.

3. Отредактируйте надписи новых элементов.
4. Привяжите к этим элементам необходимые поля из источника данных, открыв окно свойств и задав значение свойства **ControlSource** (Источник данных), щелкнув по соответствующему полю в окне свойств *(см. следующий раздел)* и выбрав в раскрывающемся списке доступных полей необходимое.
5. Сгруппируйте записи по названиям стран, а затем по фамилиям сотрудников, добавив два таких уровня группировки с помощью кнопки **Повысить уровень**.

Чтобы настроить параметры групп:

1. Откройте окно свойств уровня группы (см. рис. 11.32).
2. В появившемся окне свойств отображаются значения свойств выделенного уровня группировки. Задайте свойства каждого из уровней группировки страницы в соответствии с табл. 11.5.
3. Закройте окно свойств уровня группы.
4. Перенесите надписи с подписями к полям в раздел подписей так, чтобы они отображались над полями. Разместите поля (и надписи) в одну линию.
5. Уменьшите высоту разделов подписей и данных.



Рис. 11.32. Окно свойств для уровня группировки

Таблица 11.5. *Параметры разделов группировки*

Свойство/Группа	Страна	Фамилия	Данные
Раздел подписей (Caption Section)	Ложь (False)	Ложь (False)	Истина (True)
Заголовок группы (Group Header)	Истина	Истина	Истина
Примечание группы (Group Footer)	Ложь	Ложь	Ложь

Таблица 11.5 *(окончание)*

Свойство/Группа	Страна	Фамилия	Данные
Раздел кнопок перехода (Record Navigation Section)	Истина	Истина	Истина
Развернуто по умолчанию (Expanded By Default)	Истина	Истина	Ложь
Размер страницы доступа (Data Page Size)	1	5	5
Группировка (Group On)	По полному значению (`dscEachValue`)	По полному значению (`dscEachValue`)	
Интервал (Group Interval)	1	1	
Сортировка по умолчанию (Default Sort)	`[<имя элемента группировки>]`	`[<имя элемента группировки>]`	`[ДатаИсполнения] ASC, [ПромежуточнаяСумма] DESC`

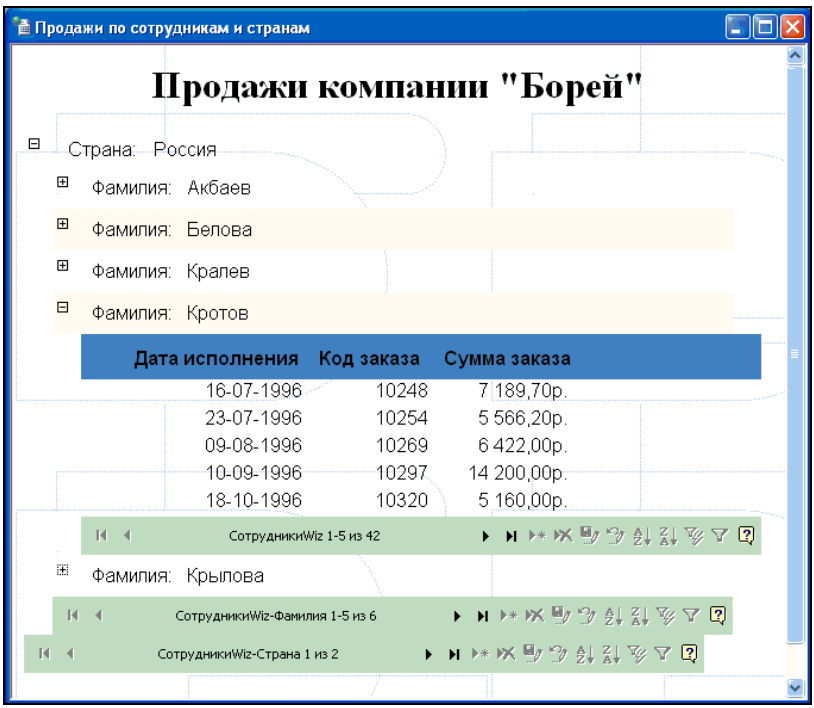


Рис. 11.33. Страница **Продажи по сотрудникам и странам** в режиме Просмотра

Выполнив эти действия, в качестве завершающего штриха мы применили к странице тему **Закругление** (Radial) для ее оформления *(см. разд. "Оформление страницы доступа к данным" этой главы)*. То, что у нас получилось, представлено на рис. 11.33.

Сравните это с тем, что было показано на рис. 11.22.

Изменение свойств страницы доступа к данным и ее элементов

Вы наверняка разрабатывали формы Access в режиме Конструктора и знаете, что у каждого элемента управления, у самой формы (как и у любого объекта) есть собственный набор свойств, которые можно изменять с помощью визуального средства — диалогового окна свойств объекта. В режиме Конструктора страницы доступа к данным тоже есть такая возможность. Чтобы отобразить диалоговое окно изменения свойств объекта в режиме Конструктора страницы доступа к данным, необходимо выделить нужный объект и выполнить одно из следующих действий:

- выберите команду **Вид | Свойства** (View | Properties);
- нажмите кнопку **Свойства** (Properties) на панели инструментов **Макет страницы** (Page Design);
- щелкните правой кнопкой по объекту и выберите в контекстном меню команду **Свойства элемента** (Element Properties);
- дважды щелкните по объекту.

Например, на рис. 11.34 изображено диалоговое окно свойств раскрывающегося списка `Обращение` (TitleOrCourtesy) страницы `Личная карточка`.

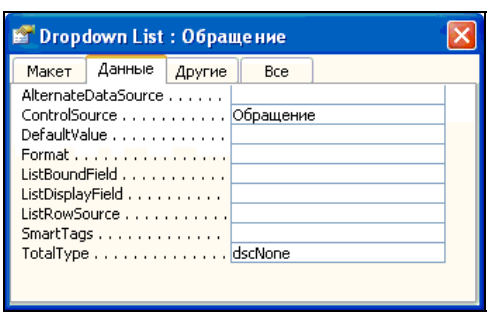


Рис. 11.34. Диалоговое окно свойств раскрывающегося списка `Обращение`

Свойства объекта в этом диалоговом окне расположены на нескольких вкладках, каждая из которых позволяет изменить определенную группу свойств:

- вкладка **Макет** (Format) — свойства, относящиеся к оформлению объекта;
- вкладка **Данные** (Data) — свойства, относящиеся к данным, отображаемым объектом;
- вкладка **Другие** (Other) — прочие свойства объекта;
- вкладка **Все** (All) объединяет все свойства, отображаемые на других вкладках.

В *разд. "Добавление, удаление и перемещение элементов управления" этой главы* мы рассмотрели создание раскрывающихся списков для полей подстановки Обращение (TitleOfCourtesy) и Подчиняется (ReportsTo). Однако в результате наших действий на странице хотя и появились соответствующие элементы управления, содержащие списки нужных значений, но поле Обращение не отображает значений, соответствующих данным из таблицы Сотрудники (Employees). Чтобы это исправить, нам нужно изменить значение свойства **ControlSource** (Источник данных) этих элементов управления.

Свойство **ControlSource** позволяет задать источник данных для получения значения, которое будет отображаться элементом управления. Некоторые свойства элементов управления, использующихся на страницах доступа к данным, приведены в табл. 11.6. Описание всех свойств можно найти в справочной системе Microsoft Access. К сожалению, интерфейс изменения свойств в режиме Конструктора страниц доступа к данным не локализован, поэтому названия свойств приведены на английском языке.

Таблица 11.6. *Некоторые свойства элементов управления*

Свойство	Описание
ControlSource (Источник данных)	Источник данных для элемента управления (это может быть имя поля в таблице, запросе или инструкции SQL или выражение)
Disabled (Блокировка)	Чтобы запретить в режиме Просмотра возможность выделения элемента щелчком мыши, установите для свойства **Disabled** значение **Истина** (True)
Id (Имя)	Уникальный идентификатор (имя) элемента управления. Используется для ссылки на элемент управления при программировании страниц доступа к данным
ReadOnly (Только чтение)	Чтобы запретить изменение данных, отображаемых в элементе управления, установите значение **Истина** (True)
TabIndex (Индекс перехода по Tab)	Номер элемента в последовательности перехода. Чтобы пропустить элемент в последовательности перехода, задайте значение –1
Title (Всплывающая подсказка)	Текст всплывающей подсказки для элемента управления
Visibility (Вывод на экран)	Определяет режим отображения элемента управления на странице. Значение **hidden** (скрытый) позволяет скрыть элемент, значение **visible** (видимый) — отобразить, значение **inherit** (унаследовано) установлено по умолчанию и означает отображение или скрытие элемента управления в зависимости от видимости родительского элемента управления

Чтобы изменить свойство **ControlSource** раскрывающегося списка Обращение:

1. В режиме Конструктора страницы Личная карточка выделите раскрывающийся список Обращение.

2. Откройте диалоговое окно его свойств одним из известных вам способов, например, нажав кнопку **Свойства** на панели инструментов **Макет страницы**.
3. Появится диалоговое окно, представленное на рис. 11.34. Раскройте вкладку **Данные**.
4. Щелкните в поле ввода **ControlSource**. В правом конце поля появится изображение кнопки со стрелкой, позволяющей раскрыть список полей источника данных этой страницы доступа к данным (для страницы `Личная карточка` источником данных является таблица `Сотрудники` (Employees)).
5. Выберите в этом раскрывающемся списке название поля `Обращение` (TitleOfCourtesy).

Теперь в режиме Просмотра страницы `Личная карточка` для текущего сотрудника в раскрывающемся списке `Обращение` будет отображаться значение, соответствующее данным, хранящимся в одноименном поле таблицы `Сотрудники`.

Заметим, что созданный с помощью мастера раскрывающийся список получил автоматически созданное название `Список0` (DropdownList0), а не `Обращение`. Чтобы изменить название элемента управления, следует указать нужное имя в поле свойства **Id** (Имя). Измените также значение свойства **Id** для привязанной к раскрывающемуся списку надписи вместо автоматически заданного имени `Список0_метка` (DropdownList0_Label) задайте имя `Обращение_метка` в окне свойств надписи. В Access 2003 изменение имени надписи не вредит ее привязке к элементу управления.

Чтобы отобразить диалоговое окно свойств целой страницы доступа к данным, а не ее отдельного элемента, нужно выделить страницу. Для этого необходимо выбрать команду **Правка | Выделить все** (Edit | Select All), тогда в диалоговом окне свойств будут отображены свойства страницы доступа к данным (рис. 11.35). Его можно отобразить также, щелкнув правой кнопкой в любом месте страницы, кроме заголовка уровня группы, и выбрав в контекстном меню команду **Свойства страницы** (Page Properties).

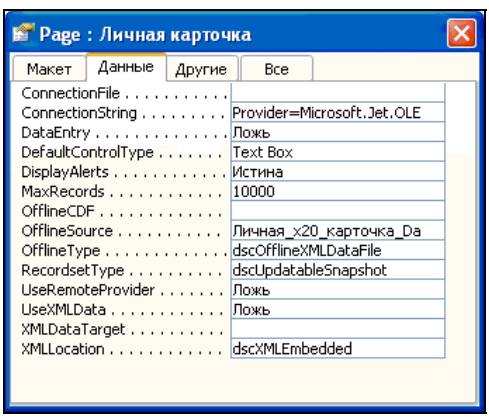


Рис. 11.35. Диалоговое окно свойств страницы доступа к данным

Команда **Файл | Свойства страницы** (File | Page Properties) позволяет настроить свойства самого файла страницы доступа к данным (рис. 11.36). Это окно содержит

свойства, похожие на свойства базы данных, свойства документа Word или любого другого документа, созданного в одном из приложений Microsoft Office: сведения об авторе, состав, статистику документа и пр. Обратите внимание на свойство **База гиперссылки** (Hyperlink base). С его помощью можно указать базовую часть адреса гиперссылки, которая будет добавляться ко всем относительным гиперссылкам, содержащимся на страницах. Очень удобно использовать именно относительные гиперссылки, т. к. при публикации страниц, когда становится известно реальное место расположения публикуемых файлов, потребуется изменить лишь базовую часть гиперссылки *(о добавлении на страницу гиперссылок см. следующий раздел)*.

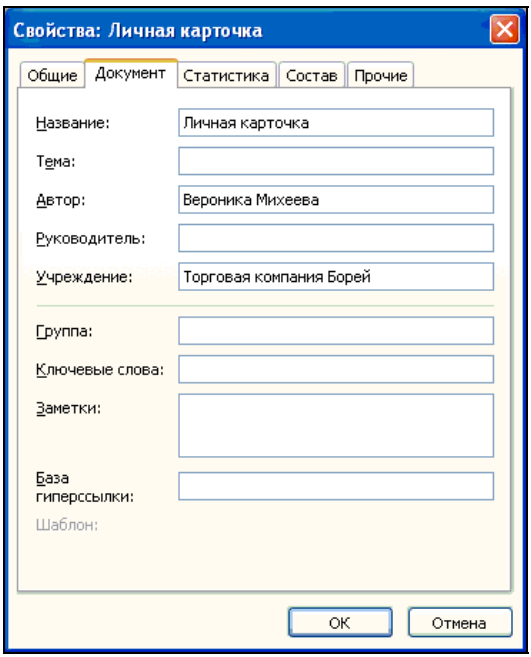


Рис. 11.36. Диалоговое окно свойств файла страницы доступа к данным

Добавление на страницу графических элементов и гиперссылок

Вернемся к нашему примеру разработки страницы `Личная карточка`, описанному в *разд. "Автоматическое создание на основе таблицы или запроса" данной главы*. Как вы помните, на этой странице не хватало фотографии сотрудника, также полезна гиперссылка на документ с резюме сотрудника. Рассмотрим, как это исправить.

Если вы откроете таблицу `Сотрудники` (Employees), которая является источником данных для нашей страницы, то заметите, что ссылка на фотографию сотрудника уже хранится в таблице в виде текстового поля `Фотография` (Photo) с именем файла изображения. Например, для новой записи, содержащей описание личной карточки сотрудницы Михеевой Вероники Дмитриевны, в этом поле мы указали относительный путь и имя файла: Photo\Veronica.bmp.

Если бы это было поле типа **Поле объекта OLE** (OLE Object) для хранения рисунка в формате Bitmap прямо в поле таблицы, пришлось бы создать дополнительное текстовое поле с именем файла, т. к. страницы доступа к данным не поддерживают работу с полями объектов. Кроме того, мы решили добавить в таблицу `Сотрудники` поле `Резюме` типа **Гиперссылка** (Hyperlink), содержащее путь и имя файла документа Word, представляющего собой резюме сотрудника. Добавьте это поле в таблицу и заполните его гиперссылками, например, для последней записи добавьте гиперссылку с надписью `Резюме В.Д.Михеевой` и адресом `CV\Veronica.doc` *(о добавлении полей в таблицу см. гл. 2, о создании гиперссылок см. разд. "Использование гиперссылок в объектах базы данных" данной главы).*

Чтобы добавить рисунок, привязанный к полю в источнике данных:

1. Откройте страницу доступа к данным в режиме Конструктора.
2. Нажмите кнопку **Рисунок** (Image) на панели управления **Панель элементов**.
3. Щелкните по свободному участку на странице и, не отпуская левую кнопку мыши, растяните контур будущего рисунка. Отпустите кнопку мыши.
4. Появится диалоговое окно **Выбор рисунка** (Insert Picture), представленное на рис. 11.37, с помощью которого можно выбрать графический файл, расположенный на локальном диске, в сети или папке Web. Выберите любой рисунок и нажмите кнопку **Добавить** (Insert).

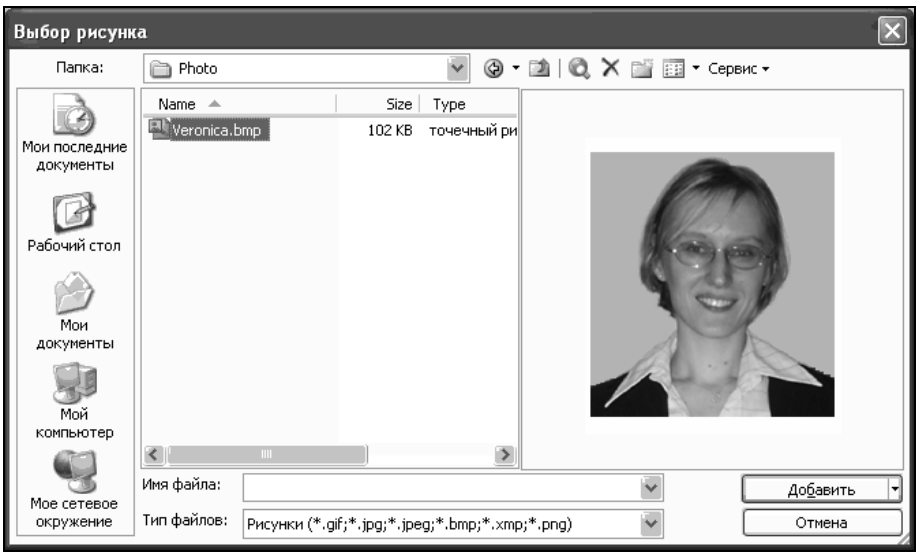


Рис. 11.37. Диалоговое окно **Выбор рисунка**

Внимание!

В нашем примере мы хотим поместить на страницу пустой объект **Рисунок** (Image), поэтому вместо кнопки **Добавить** (Insert) нажмите кнопку **Отмена** (Cancel).

5. На странице появится рисунок. Его положение на странице и размеры можно изменить с помощью мыши.

Таким образом, мы добавили на страницу один выбранный рисунок. Однако в нашем примере мы хотим использовать разные рисунки — фотографии сотрудников — отдельный рисунок для каждой записи в таблице `Сотрудники`. Это можно сделать, настроив соответствующим образом свойства созданного рисунка:

1. Выделите рисунок и отобразите для него окно свойств.
2. На вкладке **Данные** в поле со списком **ControlSource** укажите имя поля `Фотография` (Photo), которое будет источником данных для рисунка.
3. На вкладке **Макет** в поле со списком **Width** (Ширина) укажите значение **auto** (авто), чтобы ширина рисунка на странице менялась в зависимости от размеров текущего рисунка (а в поле **Height** (Высота) для размера высоты оставьте фиксированное значение).

Замечание

Хотя ширину мы сделали автоматической, размеры изображения определятся автоматически лишь по размеру первого загруженного на страницу рисунка. Поэтому, чтобы не было искажений при загрузке изображений из следующих записей, убедитесь, что все отображаемые в одной рамке объекта рисунки имеют одинаковые размеры.

4. На вкладке **Другие** в поле **Id** (имя) введите имя элемента управления `Фото`, в поле **Alt** (Подпись) введите текст `Фотография`, который будет отображаться на странице, пока не загрузился рисунок, а в поле **Title** (Название) введите текст `Фотография сотрудника`, который будет отображаться во всплывающей подсказке при наведении указателя мыши на рисунок. Из поля **Src** (источник) удалите ссылку на выбранное ранее изображение.
5. Закройте окно свойств рисунка.

Теперь добавим на страницу гиперссылку:

1. Нажмите кнопку **Гиперссылка** (Hyperlink) на панели инструментов **Панель элементов**.
2. Щелкните по свободному участку на странице. Появится диалоговое окно **Добавление гиперссылки** (Insert Hyperlink) (рис. 11.38).
3. Заполните поля: введите текст гиперссылки, отображающийся на странице, и укажите имя файла, на который она ссылается. Нажмите кнопку **OK**.

Так мы добавили на страницу гиперссылку на конкретный документ. Однако нам нужно, чтобы для каждой записи гиперссылка указывала на соответствующий документ с резюме сотрудника. Для этого можно соответствующим образом настроить свойства созданной гиперссылки, но мы воспользуемся средством автоматического создания необходимого элемента управления:

1. Удалите созданную в целях демонстрации гиперссылку.
2. Нажмите кнопку **Список полей** (Field List) на панели инструментов **Макет страницы** (Page Design), чтобы открыть окно **Список полей**.

3. Раскройте список таблиц, если он не раскрыт, и раскройте список полей таблицы Сотрудники (Employees).
4. Перетащите поле Резюме из окна **Список полей** на свободное место страницы доступа к данным. В результате Access автоматически создаст гиперссылку, привязанную к полю Резюме.

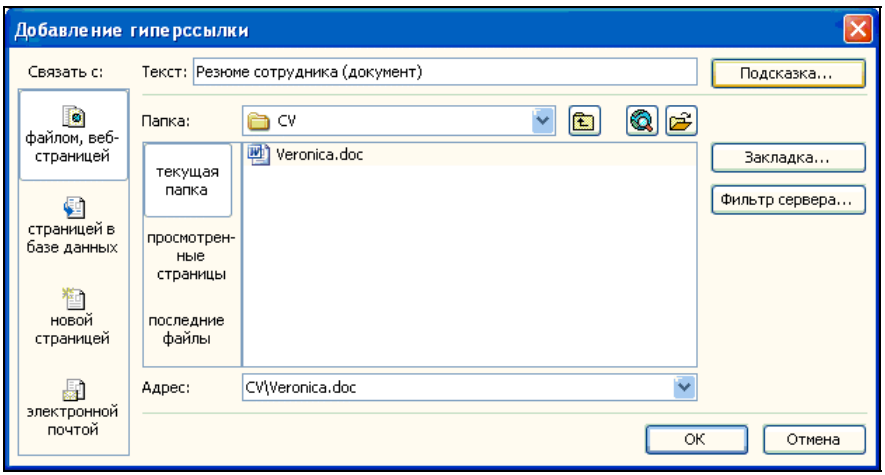


Рис. 11.38. Диалоговое окно **Добавление гиперссылки**

Теперь поинтересуемся, как были заполнены свойства данного элемента Мастером:

1. Выделите гиперссылку и откройте для нее окно свойств.
2. На вкладке **Данные** в поле со списком **ControlSource** указан источник данных для надписи гиперссылки. Это выражение, содержащее ссылку на поле Резюме: `Выражение2: MSODSC.HyperlinkPart([Резюме],0)`.
3. В поле **Title** (Название) введите текст `Резюме сотрудника (документ Word)`, который будет отображаться во всплывающей подсказке при наведении указателя мыши на гиперссылку. В поле **InnerText** (Текст) вводится текст, который будет отображаться как надпись гиперссылки. Но нам он не важен, т. к. надпись будет взята из источника данных — поля Резюме.
4. На вкладке **Привязка** (Link) в поле со списком **ControlSource (Ancor)** указан источник данных для адреса гиперссылки. Это выражение, содержащее ссылку на поле Резюме: `Выражение1: MSODSC.HyperlinkPart([Резюме1],5)`.
5. Закройте окно свойств гиперссылки. Увеличьте высоту и ширину элемента.

Теперь остается ввести текст заголовка страницы (щелкните в области заголовка над несвязанным разделом и введите название `Личная карточка сотрудника`), и наша страница готова. Сохраните страницу (кнопка **Сохранить** (Save)) и откройте ее в режиме Просмотра (кнопка **Вид** (View) на панели инструментов **Макет страницы**).

На рис. 11.39 представлен результат наших стараний — страница доступа к данным `Личная карточка`, открытая в приложении Access.

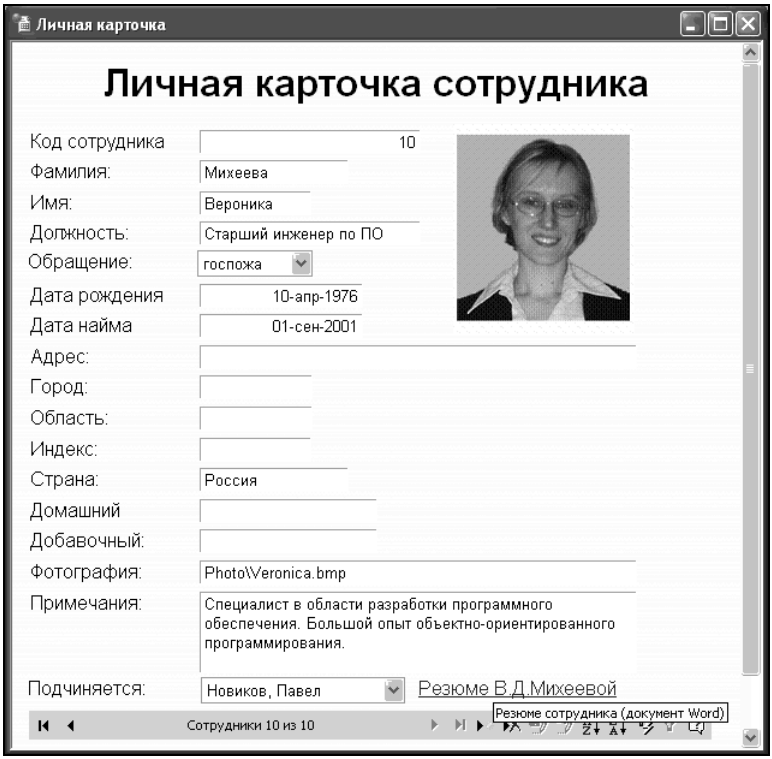


Рис. 11.39. Страница **Личная карточка**

Оформление страницы доступа к данным

Полученная страница достаточно удобна и изящна, не хватает только красочного фона, красивых шрифтов и яркого текста. Это легко исправить, ведь Microsoft Access предоставляет очень удобное средство для автоматического оформления страниц доступа к данным, подобное автоформату для форм и отчетов, так называемые *темы*. Под темой (theme) понимается совокупность элементов форматирования, применяемых для оформления страницы: цвет или рисунок фона, цвета и шрифты для основного текста, надписей и гиперссылок, графические элементы для оформления списков, параметры выравнивания и отступов для текста. Чтобы изменить сразу все эти параметры форматирования, нужно выбрать понравившуюся вам тему из предложенного списка (который достаточно обширен) и применить ее для оформления страницы доступа к данным. При необходимости можно настроить некоторые параметры выбранной темы по своему вкусу.

Все это позволяет выполнить диалоговое окно **Тема** (Theme) (рис. 11.40):

1. Откройте страницу доступа к данным в режиме Конструктора.
2. Выберите команду **Формат | Тема** (Format | Theme). Появится диалоговое окно **Тема** (Theme) (см. рис. 11.40).

3. Выделите в списке понравившуюся тему, установите или сбросьте флажки, позволяющие задать дополнительные параметры оформления, и нажмите кнопку **OK**.

Для оформления страниц нашего примера мы выбрали тему **Капсулы** (Capsules). К сожалению, для подготовки иллюстраций к книге использовались лишь 256-цветные рисунки, поэтому на них не виден фоновый узор и прочее цветовое оформление.

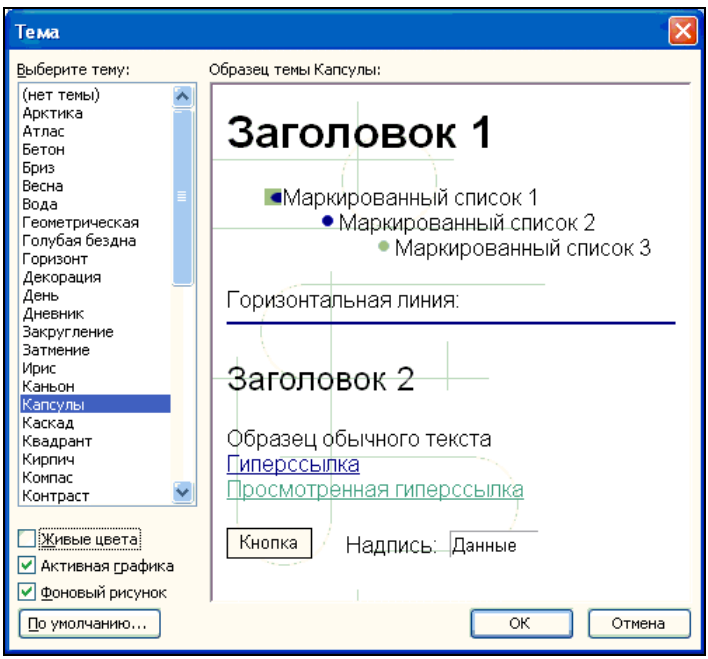


Рис. 11.40. Диалоговое окно **Тема**

После применения темы к странице для нее автоматически создаются дополнительные файлы с графическими элементами и стилями для ее оформления. Эти файлы помещаются в папке с именем, совпадающим с именем страницы, и расширением files. Эта папка находится в той же папке, где сама страница.

Внимание!

Если вы захотите в дальнейшем переместить страницу в другое место, переместите ее вместе с папкой с расширением files, содержащей дополнительные файлы для ее оформления.

Добавление на страницу элементов ActiveX

Очень часто в базах данных создаются формы и отчеты, содержащие иерархические списки (группы) данных. Для их отображения можно использовать удобный элемент

ActiveX (специальный элемент управления OLE), а именно элемент `TreeView`, с помощью которого отображается дерево подчиненных элементов данных. Можно таким элементом снабдить форму, отчет и страницу доступа к данным, т. к. на страницы можно добавлять любые зарегистрированные в Access элементы ActiveX.

Чтобы зарегистрировать новый элемент ActiveX в Access:

1. Перепишите файлы элемента управления на свой жесткий диск *(подробную информацию см. в документации элемента управления)*.
2. Выберите в меню Access команду **Сервис | Элементы ActiveX** (Tools | ActiveX Controls).
3. В диалоговом окне **Элементы ActiveX** (ActiveX Controls) (рис. 11.41) нажмите кнопку **Подключить** (Register) и укажите путь к элементу управления в диалоговом окне **Добавление элемента ActiveX** (Add AciveX Control), а затем нажмите кнопку **OK**.
4. Нажмите кнопку **Закрыть** (Close).

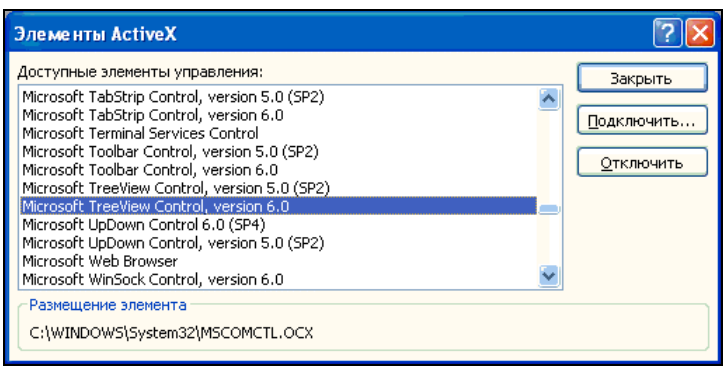


Рис. 11.41. Диалоговое окно **Элементы ActiveX**

Мы говорили об элементе `Microsoft TreeView Control, version 6`. Он определен в файле MSCOMCTL.OCX (см. рис. 11.41). Этот элемент входит в стандартный комплект элементов Microsoft Access, поэтому файлы элемента уже есть на диске в папке SYSTEM32 (для Windows NT/2000/XP) или в папке SYSTEM (для Windows 95/98) и подключены к Access. То есть этот элемент уже зарегистрирован. Остается его добавить на страницу. Создайте для этого новую страницу в режиме Конструктора и сохраните ее с именем `Иерархия`.

Чтобы добавить элемент ActiveX на страницу доступа к данным:

1. Откройте страницу доступа к данным в режиме Конструктора.
2. Нажмите кнопку **Другие элементы** (More Controls) на панели инструментов **Панель элементов** (Toolbox) и выберите нужный элемент из списка.

Замечание

Элементы ActiveX, установленные с другими приложениями, например Microsoft Excel, будут также присутствовать в списке элементов, выводимом при нажатии

кнопки **Другие элементы** панели элементов или при выборе команды **Вставка | Элемент ActiveX** (Insert | ActiveX Control). Однако некоторые элементы ActiveX могут некорректно работать в Microsoft Access. Сведения о том, был ли элемент успешно протестирован в Microsoft Access, можно найти в файле Readme или получить у поставщика этого элемента.

3. Выберите место на странице доступа к данным и поместите туда элемент управления, обрисовав его контур с помощью мыши.
4. Чтобы отобразить окно свойств элемента ActiveX, щелкните по элементу правой кнопкой и выберите команду **Свойства элемента** (Element Properties). Чтобы получить справку по какому-либо свойству, выберите это свойство и нажмите клавишу <F1>.

Чтобы в элементе `TreeView`, добавленном на страницу `Иерархия` отображалось дерево данных, нужно самостоятельно добавить узлы в дерево программным путем, используя свойства и методы этого элемента.

Использование Web-компонентов Microsoft Office

Для представления и анализа данных на Web-страницах и страницах доступа к данным фирмой Microsoft были разработаны так называемые *Web-компоненты* — это специальные элементы управления ActiveX, которые размещаются на Web-страницах или страницах доступа к данным и поддерживаются для обычных Web-страниц программой Internet Explorer 4.01 и выше, а для страниц доступа к данным — Internet Explorer 5.0 и выше. Страницы с такими элементами могут быть просмотрены при наличии установленных Web-компонентов, даже если на локальном компьютере не установлен пакет Microsoft Office. Установка Web-компонентов выполняется с дистрибутивного диска Microsoft Office.

Добавление Web-компонента на страницу доступа к данным в режиме Конструктора осуществляется с помощью соответствующей кнопки на панели инструментов **Панель элементов** (Toolbox). Перечислим основные Web-компоненты Access.

- *Сводная таблица* (PivotTable list). Аналог сводных таблиц Microsoft Excel. Отображает данные в виде строк и столбцов, которые можно реорганизовывать для анализа данных различными способами. Это делается путем перемещения строк в столбцы и столбцов в строки, отображения итоговых сумм на пересечении строки столбцов, а также с использованием других способов упорядочения и суммирования данных. Сводный список содержит собственную панель инструментов и встроенную справку.
- *Электронная таблица* (Spreadsheet). Аналог листа Microsoft Excel. Позволяет вводить и редактировать данные или выполнять вычисления с данными. Электронная таблица содержит собственную панель инструментов и встроенную справку.
- *Диаграмма* (Chart). Использует базу данных для визуального отображения тенденций, закономерностей и сравнений. При изменении базы данных происходит соответствующее изменение диаграммы. Если диаграмма связана со сводным списком или электронной таблицей на странице доступа к данным, то диаграмма изменяется при изменении связанного сводного списка или электронной таблицы.

Панель кнопок перехода (Record Navigation), упомянутая в *разд. "Элементы управления" этой главы*, также является Web-компонентом.

Более подробную информацию об использовании этих и других Web-компонентов можно найти в справочной системе Microsoft Access.

Программирование страниц доступа к данным

Данная книга не является справочником по программированию, поэтому о программировании страниц доступа к данным мы скажем здесь лишь несколько слов.

Несмотря на то, что средства Конструктора страниц доступа к данным Microsoft Access предоставляют мощные средства для представления данных на Web-страницах, без возможности программирования страницы были бы недостаточно функциональны. Поскольку страницы доступа к данным созданы по стандарту расширенного языка разметки гипертекста (Extended Markup Language, XML), они поддерживают возможность программирования и обработки событий на тех же языках сценариев, что применяются для программирования обычных страниц HTML (HyperText Markup Language) и DHTML (Dynamic HyperText Markup Language). А именно, для программирования страниц доступа к данным применяются языки сценариев VBScript и JavaScript. Язык VBScript разработан корпорацией Microsoft и является разновидностью языка Visual Basic (как и язык VBA). Язык JavaScript разработан компанией Sun Microsystems. Однако VBScript специально разработан для программирования в Microsoft Office, поэтому мы рекомендуем использовать именно его. Опытным разработчикам на языке VBA не составит труда освоить язык VBScript, поскольку они очень схожи и принадлежат одному семейству языков Visual Basic.

Для программирования и отладки страниц доступа к данным используется приложение *Редактор сценариев* (Microsoft Scripting Editor). Чтобы его открыть, откройте страницу в режиме Конструктора или выделите ее ярлык в окне базы данных и выберите команду **Сервис | Макрос | Редактор сценариев** (Tools | Macro | Microsoft Script Editor) или нажмите кнопку **Редактор сценариев** (Microsoft Script Editor) на панели инструментов.

Возможности DHTML, которыми обладают и страницы доступа к данным, позволяют изменять содержимое Web-страницы программным путем непосредственно во время ее загрузки и отображения в программе просмотра Интернета. Дело в том, что страница DHTML рассматривается не только как поток текстовой информации (особенность HTML), но и как набор объектов. Каждый объект определяется одним или двумя тегами (открывающий и закрывающий). Свойства этих объектов могут быть заданы внутри тега в виде *тегированных значений* или с помощью специальных тегов `<PARAM>` между открывающим и закрывающим тегами. Одним из свойств является уникальный идентификатор, или, другими словами, имя объекта. Оно задается как значение параметра `Id`. С помощью этого имени можно обращаться к объекту страницы в программе на языке сценариев. Изменяя свойства объектов, можно программно управлять поведением и внешним видом страницы, отображенной в программе просмотра Интернета.

Замечание

Можно задавать имена объектов (`Id`) на русском языке. Например, `Id=МояКнопка`.

Объектная модель DHTML представляет собой богатую иерархию объектов, описывающую всевозможные структурные элементы страницы, которые могут быть настроены и в процессе разработки страницы, и программным путем. Подробное описание объектной модели DHTML можно найти в справочной системе Редактора сценариев.

Создание XML-форм в InfoPath

Из описания страниц доступа к данным в этой главе вы уже могли составить представление о мощности и удобстве этих специализированных Web-страниц, предоставляющих возможности, подобные тем, что доступны в формах и отчетах Access. В новой версии Microsoft Office 2003 появился еще один способ создания подобного рода Web-страниц, но уже в формате XML — с помощью нового приложения InfoPath. Это приложение предназначено для разработки и использования динамических бизнес-форм в формате XML, которые делают процесс сбора и использования информации, а также совместную работу с ней в организации еще более эффективными.

Чтобы познакомиться с этой возможностью, рассмотрим простой пример создания интерактивной XML-формы, подключенной к базе данных `Борей`, предназначенной для ввода новых товаров и генерации списка существующих товаров.

1. Закройте базу данных `Борей`, если она была открыта.
2. Запустите приложение InfoPath 2003.
3. На панели **Заполнить форму** (Fill Out a Form) щелкните по гиперссылке **Создать форму** (Design a Form) или выберите команду **Файл | Конструктор форм** (File | Design a Form).

Замечание

Если панель **Заполнить форму** (Fill Out a Form) не отображается, выберите команду меню **Файл | Заполнить форму** (File | Fill Out a Form).

4. На панели **Конструктор форм** (Design a Form) в группе **Создать новую форму** (Design a new form) щелкните по гиперссылке **Создать из источника данных** (New from Data Source).
5. Появится первое диалоговое окно Мастера настройки источника данных новой формы (рис. 11.42). Можно выбрать один из трех типов источников данных: файл схемы или данных XML, базу данных Access или SQL Server, или Web-службу. В нашем примере требуется выбрать переключатель **База данных (только Microsoft SQL Server или Microsoft Office Access)** (Database (Microsoft SQL Server or Microsoft Office Access only)). Нажмите кнопку **Далее** (Next).
6. В следующем окне мастера нажмите кнопку **Выбрать базу данных** (Select Database).
7. Появится диалоговое окно **Выбор источника данных** (Select Data Source) (рис. 11.43). В раскрывающемся списке **Тип файлов** (Files of type) выберите элемент **Базы данных Access** (Access Databases). В раскрывающемся списке **Папка** (Look in) выберите папку, в которую установлены примеры Access (обычно она

называется C:\Program Files\Microsoft Office\Office11\Samples). В списке файлов выделите элемент **Борей.mdb** (Northwind.mdb). Нажмите кнопку **Открыть** (Open).

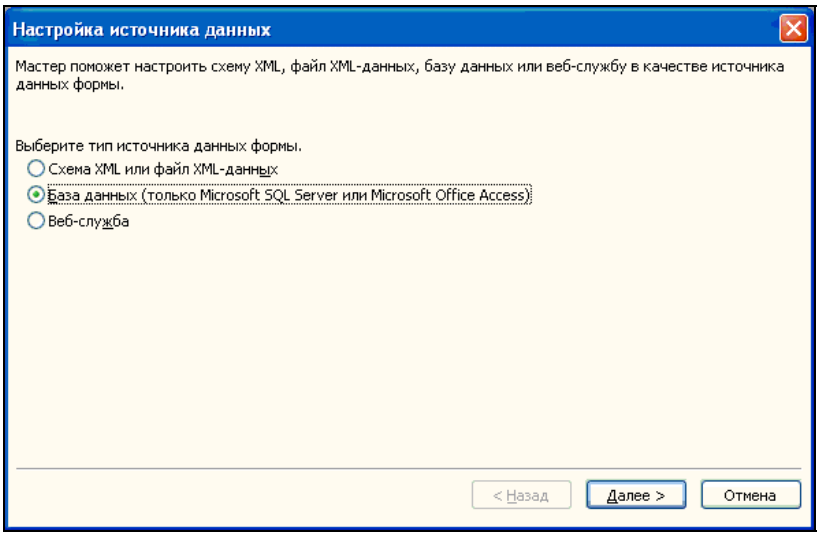


Рис. 11.42. Первое диалоговое окно Мастера настройки источника данных

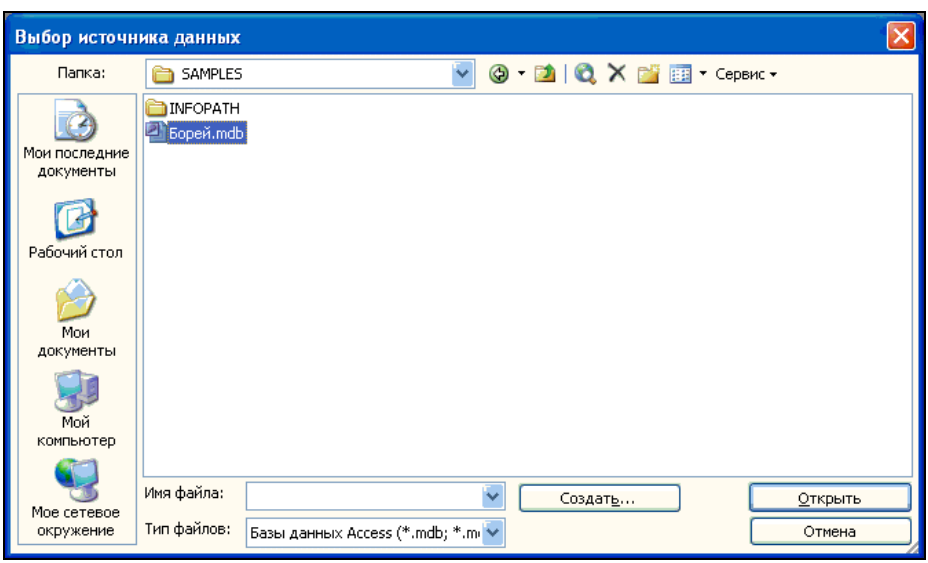


Рис. 11.43. Диалоговое окно **Выбор источника данных**

8. Появится диалоговое окно **Выделить таблицу** (Select Table), в котором отображается список всех таблиц и запросов из выбранной на предыдущем шаге базы данных (рис. 11.44). Выделите таблицу `Товары` (Products). Нажмите кнопку **OK**.

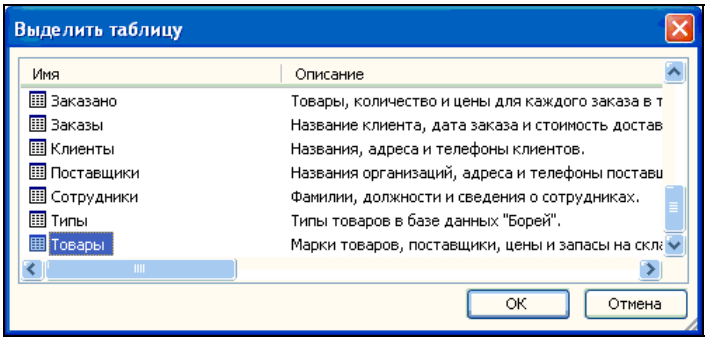


Рис. 11.44. Диалоговое окно **Выделить таблицу**

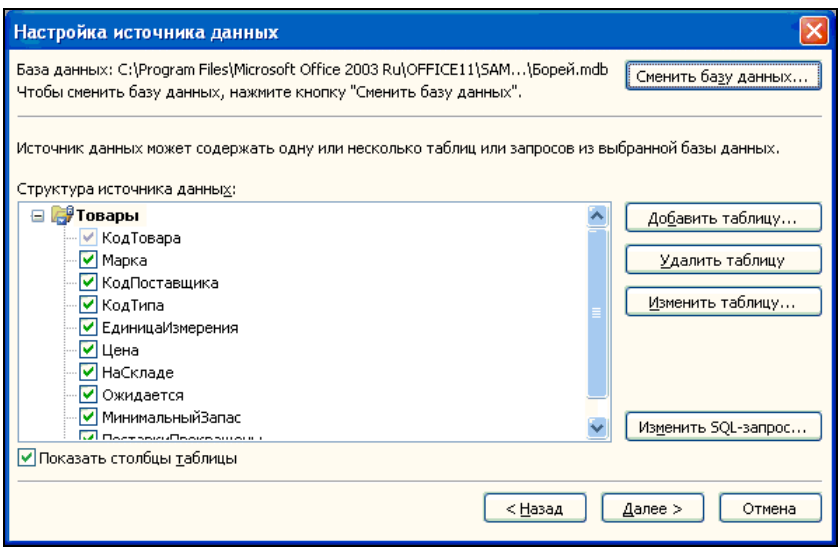


Рис. 11.45. Второе диалоговое окно Мастера настройки источника данных

9. Снова появится диалоговое окно Мастера настройки источника данных, но уже с добавленной таблицей (рис. 11.45). Нам требуется создать источник данных на основе нескольких таблиц: `Товары` (Products), `Поставщики` (Suppliers) и `Типы` (Categories). Поэтому на этом этапе нужно добавить каждую из них с помощью кнопки **Добавить таблицу** (Add Table) и задать поля связи этих таблиц. В диалоговом окне мастера у всех добавленных таблиц отображаются списки полей (если установлен флажок **Показать столбцы таблицы** (Show table columns)). Напротив названий ненужных полей можно сбросить флажок, чтобы не включать их в источник данных. Для нашего примера на этом этапе добавьте сначала таблицу `Поставщики` (Suppliers) с полями `КодПоставщика` (SupplierID) и `Название` (CompanyName) и полем связи `КодПоставщика` (SupplierID). Затем в списке таблиц источника данных снова выделите название основной таблицы `Товары` (Products) и добавьте связанную с ней таблицу `Типы` (Categories) с полями `КодТипа` (Cate

goryID) и Категория (CategoryName) и полем связи КодТипа (CategoryID). Нажмите кнопку **Далее** (Next).

10. Появится последнее диалоговое окно Мастера настройки источника данных, в котором отображаются параметры выбранного источника данных (рис. 11.46). Нажмите кнопку **Готово** (Finish).

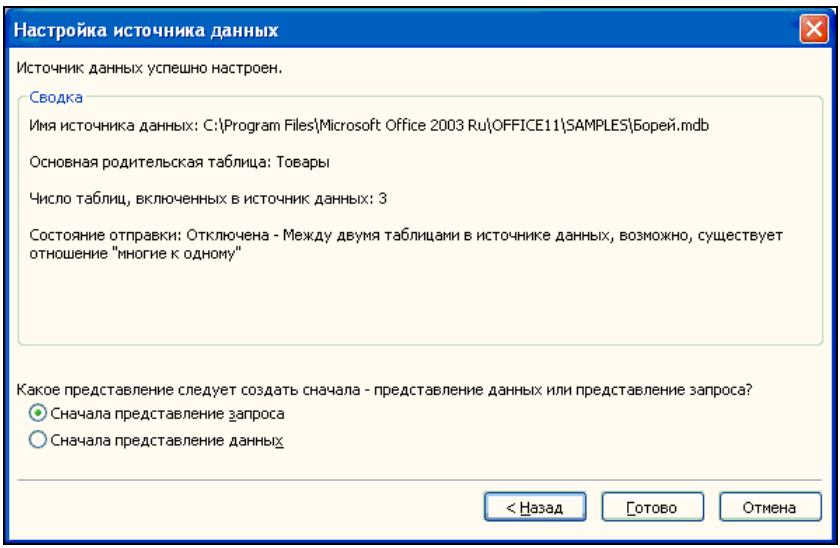


Рис. 11.46. Последнее диалоговое окно Мастера настройки источника данных

11. На основе выбранного источника данных будет сгенерирована новая XML-форма, содержащая представление для ввода данных и представление запроса. Новая форма отобразится в приложении InfoPath в режиме Конструктора. По умолчанию будет открыто представление запроса (рис. 11.47).

Представление запроса предназначено для переключения в режим ввода или просмотра данных, удовлетворяющих заданным в этом представлении условиям фильтрации. В представление запроса были автоматически добавлены все поля из таблицы источника данных Товары. Необходимо настроить внешний вид представления, чтобы сделать его более удобным и красивым.

1. Чтобы преобразовать элемент управления ПоставкиПрекращены (Discontinued) из раскрывающегося списка во флажок, щелкните по нему правой кнопкой и в контекстном меню выберите команду **Изменить на | Флажок** (Change To | Check Box). Аналогично преобразуйте поля ввода КодПоставщика (SupplierID) и КодТипа (CategoryID) в раскрывающиеся списки. Выделите по очереди появившиеся элементы управления и уменьшите их ширину с помощью маркеров изменения размеров.
2. Чтобы вместо кода поставщика и кода категории вводить их названия, нужно настроить источник данных у этих раскрывающихся списков. Щелкните правой кнопкой по элементу КодПоставщика (SupplierID) и в контекстном меню выберите команду **Свойства: Раскрывающийся список** (Drop-Down List Box Properties).

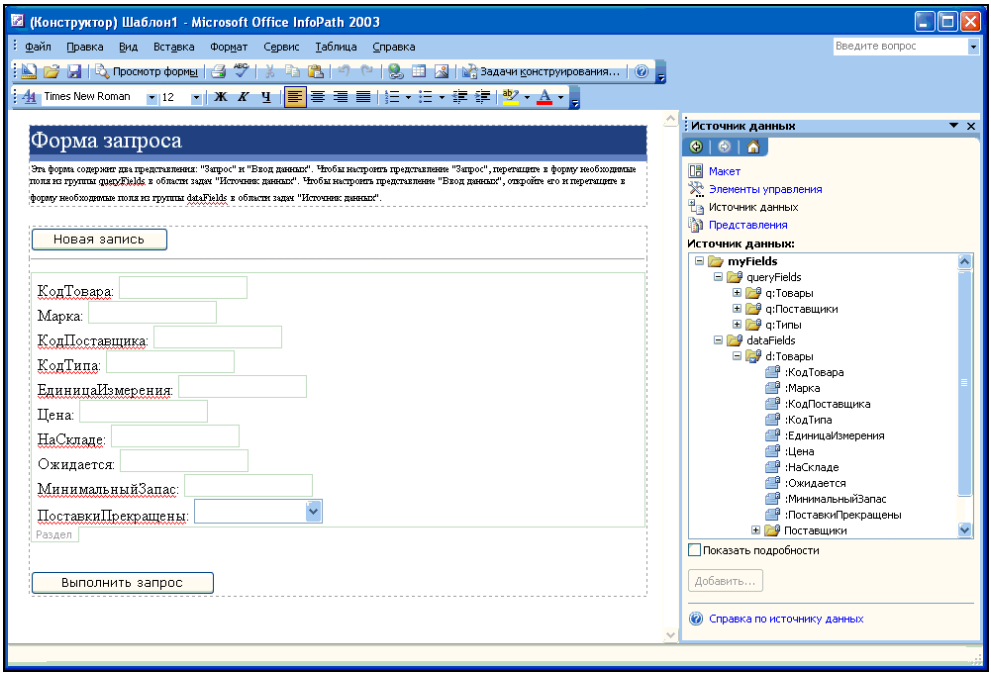


Рис. 11.47. Новая форма InfoPath в режиме Конструктора

3. Появится диалоговое окно свойств выделенного раскрывающегося списка (рис. 11.48). На вкладке **Данные** (Data) выберите переключатель **Подставить из базы данных, веб-службы или файла** (Look up in a database, Web service, or file). Нажмите кнопку **Дополнительный источник данных** (Secondary Data Source). В появившемся диалоговом окне **Дополнительные источники данных** (Secondary Data Sources) нажмите кнопку **Добавить** (Add).

4. Появится уже знакомое вам первое окно Мастера настройки источника данных. С его помощью откройте уже известным вам способом базу данных `Борей`, выберите таблицу `Поставщики` (Suppliers), выберите из нее только поля `КодПоставщика` (SupplierID) и `Название` (CompanyName). Нажмите кнопку **Далее** (Next). В последнем окне мастера нажмите кнопку **Готово** (Finish).

5. Во вновь появившемся окне **Дополнительные источники данных** нажмите кнопку **Закрыть** (Close). В окне свойств в раскрывающемся списке **Источник данных** (Data source) будет выбран элемент **Поставщики** (Suppliers).

6. Напротив поля ввода **Элементы** (Entries) нажмите кнопку **Выбрать Xpath** (Select XPath). Раскройте список полей в появившемся диалоговом окне, выделите элемент **d:Поставщики** (d:Suppliers) и нажмите кнопку **OK**.

7. Напротив поля ввода **Значение** (Value) нажмите кнопку **Выбрать Xpath** (Select XPath). В появившемся диалоговом окне в списке полей выделите элемент **:КодПоставщика** (:SupplierID) и нажмите кнопку **OK**.

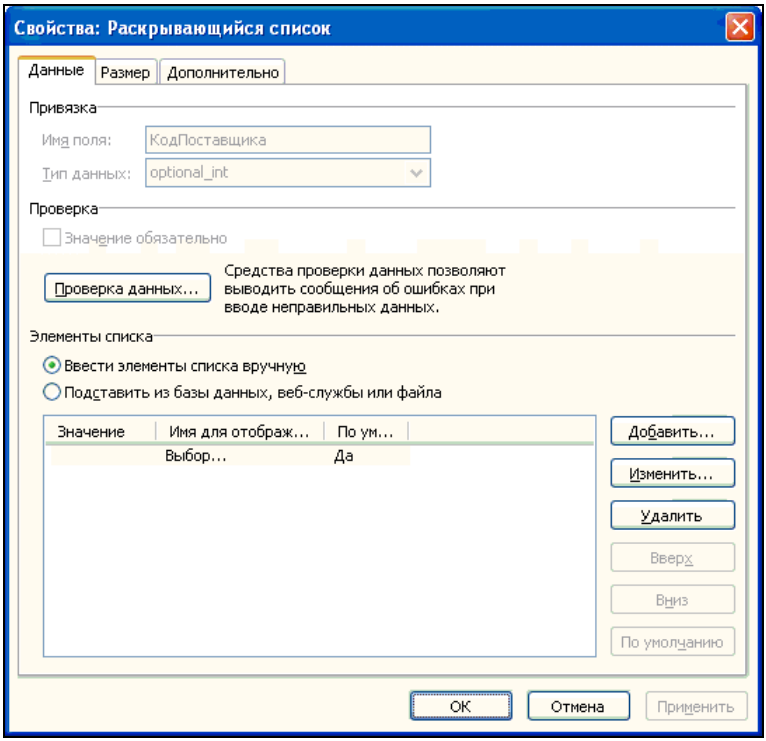


Рис. 11.48. Диалоговое окно свойств раскрывающегося списка

8. Напротив поля ввода **Имя для отображения** (Display name) нажмите кнопку **Выбрать Xpath** (Select XPath). В появившемся диалоговом окне в списке полей выделите элемент **:Название** (:CompanyName) и нажмите кнопку **OK**.
9. Нажмите кнопку **OK** в окне свойств раскрывающегося списка `КодПоставщика`.
10. Выполните аналогичные действия (с шага 2 по 9) для раскрывающегося списка `КодТипа` (CategoryID), но в качестве источника данных подключите из базы данных `Борей` таблицу `Типы` (Categories) с полями только `КодТипа` (CategoryID) и `Категория` (CategoryName).

Значения, выбираемые в раскрывающихся списках, привязаны к источнику данных. Чтобы просмотреть или изменить привязанное поле данных:

1. Щелкните правой кнопкой по раскрывающемуся списку, например по списку `КодПоставщика` (SupplierID), и в контекстном меню выберите команду **Изменить привязку** (Change Binding). Появится диалоговое окно **Раскрывающийся список: привязка** (Drop-Down List Box Binding) (рис. 11.49).
2. В списке полей выделено то поле из таблицы, к которому привязано значение, выбранное в списке. Нажмите кнопку **OK**.

Теперь измените названия раскрывающихся списков в представлении запроса. Вместо названия раскрывающегося списка `КодПоставщика` (SupplierID) на форме введите

текст Поставщик (Supplier). Вместо названия раскрывающегося списка КодТипа (CategoryID) на форме введите текст Тип (Category). Отредактируйте остальные надписи и сохраните получившуюся форму с помощью команды **Файл | Сохранить** (File | Save). В появившемся диалоговом окне нажмите кнопку **Сохранить** (Save), затем задайте путь и имя файла XML-формы: Товары.xsn (Products.xsn). Нажмите кнопку **Сохранить** (Save).

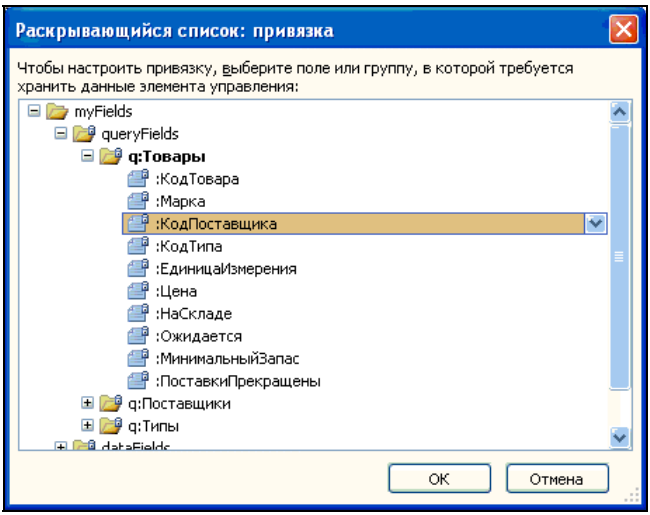


Рис. 11.49. Диалоговое окно **Раскрывающийся список: привязка**

Теперь переходим к настройке представления **Ввод данных** (Data Entry).

1. На панели **Представления** (Views) щелкните по названию представления **Ввод данных** (Data Entry). Отобразится пустое представление **Ввод данных** (Data Entry).

Замечание

Если панель **Представления** (Views) не отображается, выберите из раскрывающегося списка вверху на Панели задач элемент **Представления** (Views) или щелкните по одноименной гиперссылке.

2. На панели **Представления** (Views) щелкните по гиперссылке **Источник данных** (Data Source). Отобразится иерархия полей источника данных. Корневой узел иерархии — **myFields**. На следующем уровне иерархии отображаются две группы полей: поля представления запроса — **queryFields** и поля представления данных — **dataFields**.
3. Раскройте список полей представления данных **dataFields**. Из этого списка перетащите на форму те поля, которые требуется отобразить в представлении **Ввод данных** (Data Entry). Например, это могут быть поля: КодТовара (ProductID), Марка (ProductName), ЕдиницаИзмерения (QuantityPerUnit), Цена (UnitPrice), НаСкладе (UnitsInStock) и ПоставкиПрекращены (Discontinued) из таблицы Товары (Products), поле Название (CompanyName) из таблицы Поставщики (Suppliers) и поле Категория (CategoryName) из таблицы Типы (Categories).

4. Преобразуйте элемент `ПоставкиПрекращены` (Discontinued) из раскрывающегося списка во флажок. Отредактируйте надписи для полей на форме: например, замените надпись `Название`: (Company Name:) на надпись `Поставщик`: (Supplier:). Отформатируйте надписи и элементы управления с помощью панели инструментов **Форматирование** (Formatting).
5. Сохраните последний вариант формы с помощью кнопки **Сохранить** (Save) на панели инструментов **Стандартная** (Standard).

Чтобы протестировать форму, вновь откройте представление запроса и нажмите кнопку **Просмотр формы** (Preview Form) на панели инструментов **Стандартная** (Standard). Представление запроса откроется в режиме Просмотра (рис. 11.50).

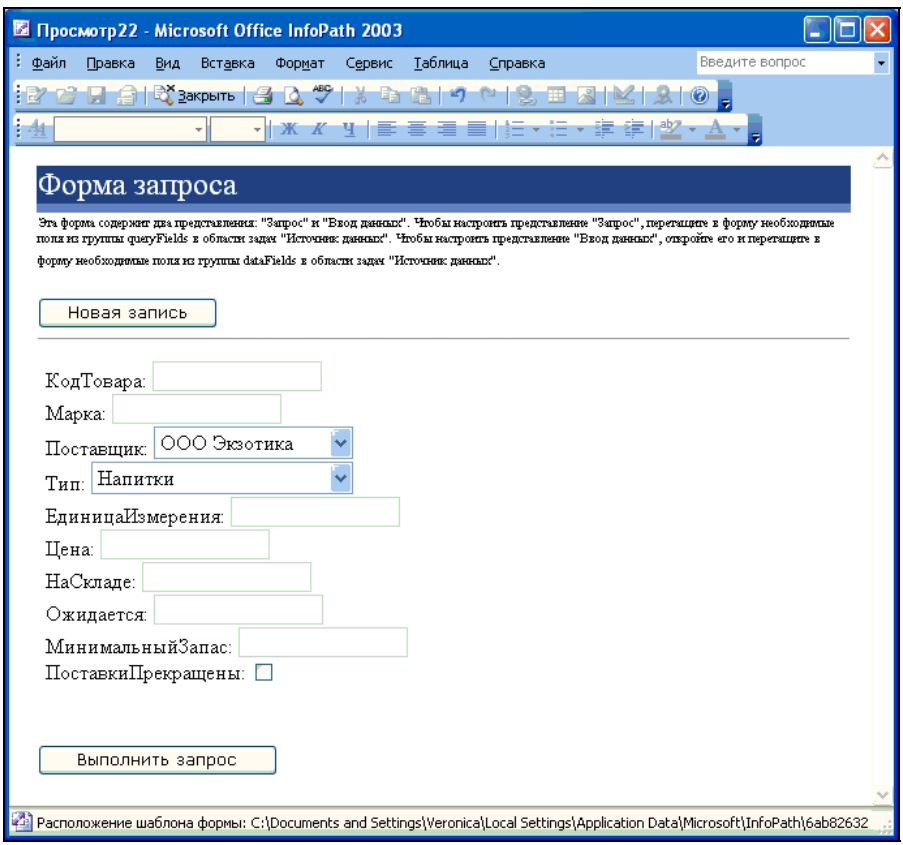


Рис. 11.50. Представление ввода данных в режиме Просмотра формы

Кнопка **Новая запись** (New Record) в представлении запроса позволяет перейти к созданию новой записи в представлении ввода данных.

В списке поставщиков выберите элемент **ООО Экзотика** (Exotic Liquids), а в списке типов товаров — элемент **Напитки** (Beverages). Нажмите кнопку **Выполнить запрос** (Run Query), находящуюся на форме. Отобразится представление ввода данных со

списком продуктов типа **Напитки** (Beverages), поставщиком которых является **ООО Экзотика** (Exotic Liquids) (рис. 11.51).

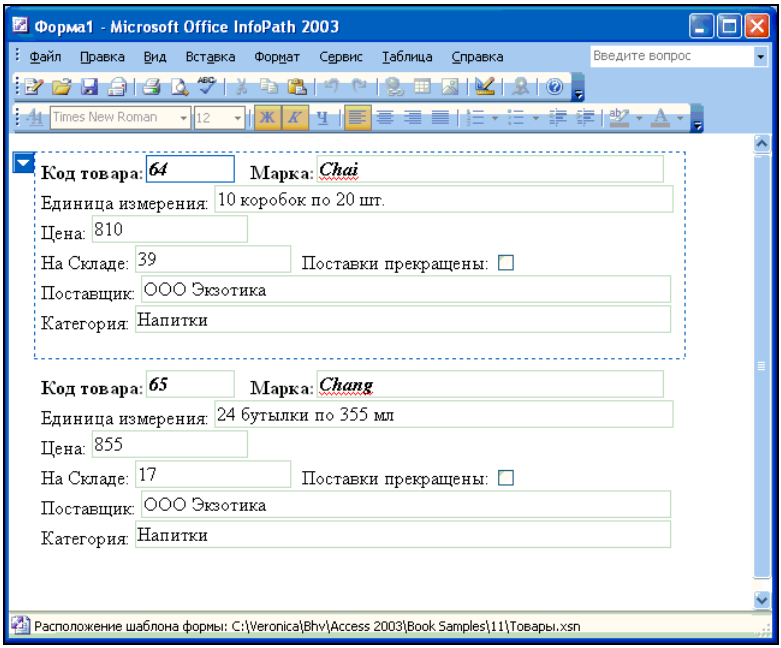


Рис. 11.51. Список продуктов, отображенный в форме

Закройте форму в режиме просмотра с помощью команды **Файл | Закрыть предварительный просмотр** (File | Close Preview). Закройте разработанную форму с помощью команды **Файл | Закрыть** (File | Close).

Рассмотренный пример позволяет освоить базовые приемы разработки форм InfoPath. Для перехода в режим использования формы InfoPath выберите команду **Файл | Заполнить форму** (File | Fill Out a Form). С помощью панели задач InfoPath или команды **Файл | Открыть** (File | Open) откройте созданную форму `Товары.xsn` (Products.xsn). Форма откроется в режиме запроса. Введите значения полей для выборки нужных записей из источника данных и нажмите кнопку **Выполнить запрос** (Run Query), либо просто нажмите кнопку **Новая запись** (New Record) для добавления записи в источник данных. Для перехода между представлениями используйте соответствующие команды меню **Вид** (View). Открытую форму можно сохранить в формате XML с помощью команды **Файл | Сохранить** (File | Save) или **Файл | Сохранить как** (File | Save As).

Средства публикации данных

В этом разделе описываются процедуры публикации данных в Интернете и интранете, т. е. перенос Web-интерфейса в место, доступное для пользователей соответствующих сетей. Средства публикации информации в интранете и Интернете превра-

щают Access в мощный инструмент разработки распределенных приложений. Для работы таких приложений требуется единственное средство — программа просмотра Интернета.

Публикация данных с помощью Web-страниц

В этой главе мы уже неоднократно касались темы о подключении к источнику данных для разных форматов Web-страниц. Этот аспект настолько важен и так часто вызывает вопросы, что мы решили обобщить сказанное в данном разделе.

Очень важный вопрос в обеспечении работоспособности Web-страниц, предназначенных для представления данных в Web, организовать для них правильное соединение с источником данных. Для разных форматов Web-страниц применяются разные источники данных. Об особенностях форматов см. табл. 11.7.

Таблица 11.7. *Характеристики форматов Web-страниц*

Формат Web-документа	**Статические страницы HTML**	**Динамические, генерируемые сервером, страницы HTML**	**Страницы доступа к данным**	**XML-документы**
Расширения файлов	html, htm	asp, htx и idc	htm	xml, xsd, xsl, htm или asp
Источник данных	Данные содержатся в самой странице	База данных Access	База данных Access	Данные содержатся в файле XML, структура данных может содержаться отдельно — в файле XSD
Подключение к источнику данных	Не требуется	Путем регистрации *системного источника данных ODBC* на компьютере, являющемся Web-сервером. Поместите базу данных в ту же папку, где расположена страница, или укажите в источнике данных ее сетевое имя UNC	При доступе через файловую систему — не требуется; при доступе через Web — путем регистрации источника данных ODBC (с адресом источника данных в форамте UNC) на компьютере, являющемся Web-сервером	Не требуется
Чтение данных	На момент создания страницы	Актуальные данные из источника	Актуальные данные из источника	Актуальные данные из файла XML
Редактирование данных	Невозможно	Невозможно	Возможно прямо на странице	Возможно с помощью приложений, воспринимающих формат XML

Таблица 11.7 *(окончание)*

Формат Web-документа	Статические страницы HTML	Динамические, генерируемые сервером, страницы HTML	Страницы доступа к данным	XML-документы
Возможность просмотра (публикации) документа	На локальном компьютере, в общей папке на файл-сервере, на FTP-сервере, на Web-сервере Интернета или интранета	Только на Web-сервере Интернета или интранета, просмотр только через протокол HTTP	На локальном компьютере, в общей папке на файл-сервере, на FTP-сервере, на Web-сервере Интернета или интранета	На локальном компьютере, в общей папке на файл-сервере, на FTP-сервере, на Web-сервере Интернета или интранета
Программа просмотра	Любая программа просмотра Интернета, поддерживающая HTML 3.2 или выше	Любая программа просмотра Интернета	Internet Explorer 5 или выше	Internet Explorer 5 или выше
Отправка по электронной почте	Возможна	Невозможна	Возможна	Возможна
Публикация на сервере FTP	Возможна	Невозможна	Возможна	Возможна
Имя файла	Поддерживаются латинские и русские буквы в имени страницы	Поддерживаются только латинские буквы в имени страницы	При публикации на Web-сервере поддерживаются только латинские буквы в имени страницы	Поддерживаются латинские и русские буквы в имени страницы, за исключением представлений в формате ASP — поддерживаются только латинские буквы

Замечание

Большинство сведений о публикации нетривиальных Web-страниц мы привели основываясь на справочных материалах Microsoft. Мы реально испытали работу страниц доступа к данным и других лишь в интрасети. Публикация страниц в Интернете может иметь свои тонкости. Надеемся, что приведенной информации и справки Access будет достаточно, чтобы можно было во всем разобраться опытным путем.

Публикация данных на сервере Web

Проще всего опубликовать Web-страницу (страницу доступа к данным или объект, преобразованный в Web-страницу) при помощи Проводника Windows (Explorer).

Специальная надстройка Web-папки позволяет работать с папками и файлами на Web-серверах точно так же, как с папками и файлами на локальном диске.

Прежде чем воспользоваться этой возможностью, необходимо создать соединение с Web-сервером:

1. Запустите Проводник Windows (Explorer).
2. В левой панели выделите элемент **Web-папки** (Web Folders). Дважды щелкните по элементу **Добавить Web-папку** (Add Web Folder) на правой панели Проводника Windows (рис. 11.52). Появится диалоговое окно **Добавить Web-папку** (Add Web Folder).

Замечание

В последних версиях операционной системы Windows 2000 и XP Web-папки и прочие ярлыки к ресурсам сети (FTP-папкам и папкам локальной сети) создаются в папке Сетевое окружение (My Network Places).

3. Введите URL папки на Web-сервере, в которой необходимо сохранить Web-страницу, например `http://muzzy/chavo`, и нажмите кнопку **Далее** (Next).
4. Введите название для создаваемого соединения и нажмите кнопку **Готово** (Finish). В списке **Web-папки** появится новый элемент (см. рис. 11.52).

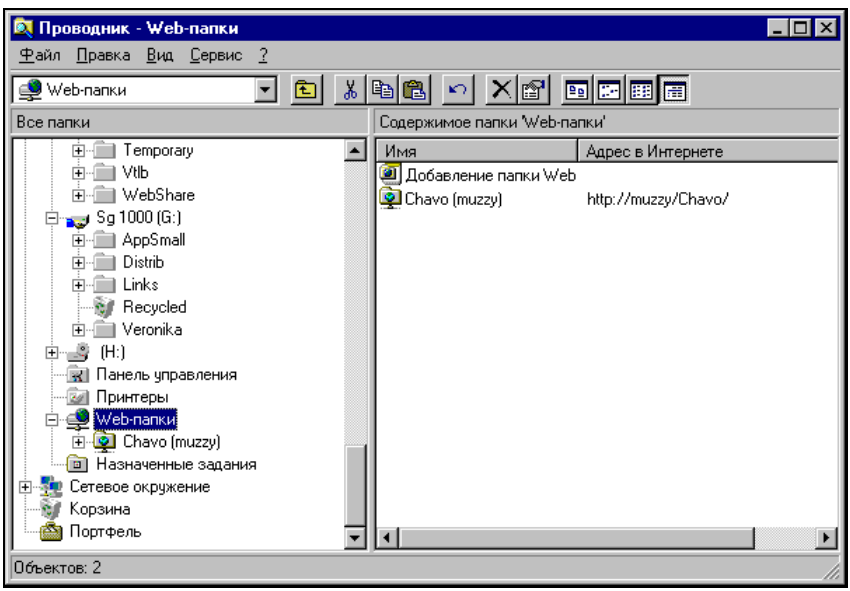


Рис. 11.52. Элемент **Web-папки** в Проводнике Windows

Чтобы скопировать данные на Web-сервер при помощи средства Web-папки:

1. В Проводнике Windows выделите файлы и папки, которые необходимо скопировать на Web-сервер.
2. Выберите команду **Правка | Копировать** (Edit | Copy).

3. Дважды щелкните по значку соединения в списке **Web-папки** (Web Folders).
4. Раскройте структуру папок для выбранного соединения и выделите папку, в которую необходимо скопировать файлы и папки, указанные на шаге 1.
5. Выберите команду **Правка | Вставить** (Edit | Paste). Выделенные на шаге 1 файлы и папки будут скопированы на Web-сервер.

Замечание

Не забудьте добавить на Web-страницы, которые уже опубликованы на Web-сервере, гиперссылки на публикуемые документы. Это упростит доступ к информации.

При публикации страниц доступа к данным необходимо помнить о том, что база данных также должна быть опубликована на Web-сервере. В противном случае страница не получит доступа к объектам базы данных.

Если Web-сервер расположен на вашем локальном компьютере, опубликовать на нем данные можно просто скопировав необходимые файлы в корневую папку Web-сервера или одну из его вложенных папок. Корневая папка Web-сервера может иметь имя \Inetpub\wwwroot, \Webshare\Wwwroot или другое, в зависимости от программного обеспечения Web-сервера и его настроек.

Отправка данных по электронной почте

Электронная почта является одним из наиболее распространенных способов обмена информацией в электронном виде. Электронную почту можно использовать для обмена информацией в процессе разработки базы данных, с помощью нее можно обсуждать информацию, хранящуюся в базе данных, проводить целевые рассылки информации, интересующей клиентов, и т. д.

Начиная с версии Microsoft Office 2000, взаимодействие приложений этого семейства с клиентами электронной почты (в особенности с Microsoft Outlook) стало простым и прозрачным. В Access любой объект базы данных (за исключением макросов) можно отправить по электронной почте в качестве вложения. Кроме того, страницы доступа к данным можно отправлять в качестве тела сообщения.

Чтобы отправить объект базы данных по электронной почте:

1. Выделите объект в окне базы данных Microsoft Access.
2. Выберите команду **Файл | Отправить | Сообщение** (File | Send To | Mail Recipient) или **Файл | Отправить | Сообщение (как вложение)** (File | Send To | Mail Recipient (as Attachment)) в зависимости от того, каким способом вы хотите отправить объект.
3. Если вы отправляете не страницу доступа к данным, появится диалоговое окно со списком возможных форматов для отправки объекта. Укажите формат, в котором необходимо переслать данные. Мы рекомендуем для всех типов объектов базы данных использовать формат HTML. Нажмите кнопку **OK**. В зависимости от выбранного формата данных может появиться дополнительное диалоговое окно, в котором необходимо указать параметры преобразования объекта.
4. Запустится клиент электронной почты, используемый по умолчанию (в нашем примере это Microsoft Outlook), и будет создано новое сообщение.

5. Укажите получателей сообщения, используя кнопки **Кому** (To) и **Копия** (Cc).
6. Введите тему сообщения и, если необходимо, пояснительный текст в поле самого сообщения.
7. В новой версии Microsoft Office 2003 появилась возможность совместной работы над документами в специальной *рабочей области (о рабочих областях документов см. справочную систему Microsoft Office 2003)*. Если потребуется использовать эту возможность вместо обычной рассылки получателям копий объекта, нажмите кнопку **Параметры вложений** (Attachment Options), чтобы выбрать нужный способ отправки и настроить необходимые параметры.
8. Нажмите кнопку **Отправить** (Send) (или **Отправить копию** (Send a Copy)).

Замечание

Еще в версии Microsoft Outlook 2000 была реализована поддержка формата HTML для основного текста сообщений, поэтому с помощью этой программы можно пересылать страницы доступа к данным в том виде, в котором они отображаются в программе просмотра.

Публикация данных на сервере FTP

Начиная с версии Microsoft Office 2000, можно сохранить открытый документ на сервере FTP. Чтобы опубликовать на сервере FTP открытую в Access страницу доступа к данным или объект базы данных, сохраняемый в формате страницы доступа к данным, или документ, открытый в приложении Word или FrontPage:

1. Откройте окно сохранения документа, выбрав команду **Файл | Сохранить как** (File | Save As). Если сохраняете объект базы данных, в появившемся диалоговом окне выберите тип **Страница доступа к данным** (Data Access Page).
2. Появится диалоговое окно, представленное на рис. 11.53, с заголовком **Новая страница доступа к данным** (New Data Access Page) или **Сохранение документа** (Save As). В списке **Папка** (Save in) выделите элемент **Адреса FTP** (FTP Locations) (рис. 11.53).
3. В поле **Имя файла** (File name) задайте имя публикуемого файла, в списке **Тип файла** (Save as type) выберите его тип.
4. Если нет ни одного подходящего элемента в списке узлов FTP, дважды щелкните по ярлыку **Добавить/изменить адреса FTP** (Add/Modify FTP Locations). В появившемся диалоговом окне введите имя FTP-сервера (рис. 11.54), а если нужный адрес FTP уже присутствует в списке, просто выделите его. Нажмите кнопку **ОК**. В списке адресов FTP появится новый элемент, выделите его.
5. Нажмите кнопку **Сохранить** (Save) (или **ОК**).

Чтобы опубликовать на сервере FTP объект базы данных Microsoft Access:

1. В окне базы данных выделите публикуемый объект, например, таблицу или страницу доступа к данным.
2. Выберите команду **Файл | Экспорт** (File | Export). Появится диалоговое окно **Экспорт объекта** (Export To) (см. рис. 11.5).

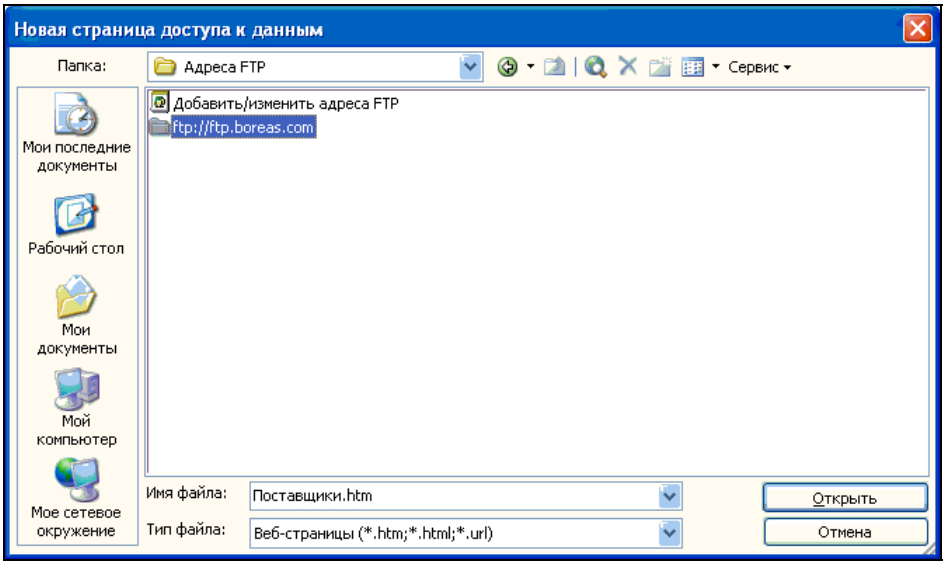


Рис. 11.53. Элемент **Адреса FTP** в окне сохранения таблицы `Поставщики` как страницы доступа к данным

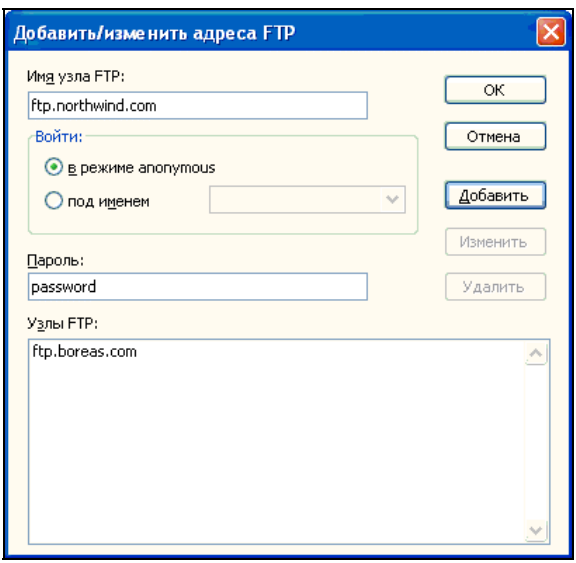


Рис. 11.54. Диалоговое окно **Добавить/изменить адреса FTP**

3. В списке **Папка** (Save in) выделите элемент **Адреса FTP** и выполните шаги с 3-го по 4-й предыдущей процедуры публикации документа на сервере FTP.
4. Нажмите кнопку **Экспорт** (Export).

ГЛАВА 12



Программирование с помощью макросов

Основной набор средств Microsoft Access, который мы рассматривали в предыдущих главах книги, ориентирован на пользователей, не владеющих языками программирования. Для программистов же к этим средствам добавлены *макросы* (небольшие программы на языке макрокоманд системы Access) и *модули* (процедуры на языке Visual Basic for Applications, VBA). С их помощью можно существенно расширить функциональные возможности создаваемого приложения и настроить его на нужды конкретных пользователей. В этой главе мы познакомимся с макросами и одним из языков программирования Access — языком макрокоманд. С модулями и языком VBA мы вас познакомим в следующей главе.

Итак, мы будем обсуждать следующие вопросы:

- Понятие макроса
- Использование макросов в приложениях
- Создание макросов и групп макросов
- Задание условий для управления процессом выполнения макроса
- Различные способы запуска макросов
- Советы по отладке макросов
- Обработка событий в формах и отчетах с помощью макросов
- Преобразование макросов в процедуры VBA

Использование макросов в базе данных Access

С помощью макросов можно выполнять над объектами Access практически все действия, описанные в предыдущих главах.

Макрос в Access представляет собой структуру, состоящую из одной или нескольких макрокоманд, которые выполняются либо последовательно, либо в порядке, заданном определенными условиями. Набор макрокоманд в Access очень широк, с по-

мощью макросов можно реализовать многое из того, что позволяют сделать процедуры на VBA. Каждая макрокоманда имеет определенное имя и, возможно, один или несколько аргументов, которые задаются пользователем. Например, при использовании макрокоманды ОткрытьФорму (OpenForm) в качестве аргументов необходимо задать, по крайней мере, имя открываемой формы и режим вывода ее на экран.

Табл. 12.1 содержит список макрокоманд Access 2003, сгруппированных по категориям: работа с данными в формах и отчетах; выполнение команд, макросов, процедур и запросов; работа с объектами; импорт/экспорт данных и объектов и прочие. В категорию "Прочие" собраны все макрокоманды, которые позволяют влиять на интерфейс приложения.

Таблица 12.1. *Макрокоманды Microsoft Access 2003*

Категория	Назначение	Макрокоманда
Работа с данными в формах и отчетах	Отбор данных	ПрименитьФильтр (ApplyFilter)
	Перемещение по данным	СледующаяЗапись (FindNext) НайтиЗапись (FindRecord) КЭлементуУправления (GoToControl) НаСтраницу (GoToPage) НаЗапись (GoToRecord)
	Обновление данных или экрана	Обновление (Requery) ПоказатьВсеЗаписи (ShowAllRecords)
Выполнение	Выполнение команды	ВыполнитьКоманду (RunCommand)
	Выполнение макроса, процедуры или запроса	ЗапускМакроса (RunMacro) ЗапускПрограммы (RunCode) ОткрытьЗапрос (OpenQuery) ЗапускЗапросаSQL (RunSQL)
	Выполнение другого приложения	ЗапускПриложения (RunApp)
	Прерывание выполнения	ОтменитьСобытие (CancelEvent) ОстановитьВсеМакросы (StopAllMacros) ОстановитьМакрос (StopMacro)
	Выход из Microsoft Access	Выход (Quit)
Импорт/экспорт	Передача объектов Microsoft Access в другие приложения	ВывестиВФормате (OutputTo) ОтправитьОбъект (SendObject)
	Преобразование формата данных	ПреобразоватьБазуДанных (TransferDatabase) ПереносБазыДанныхSQL (TransferSQLDatabase) ПреобразоватьЭлектроннуюТаблицу (TransferSpreadsheet) ПреобразоватьТекст (TransferText)

Таблица 12.1 *(продолжение)*

Категория	Назначение	Макрокоманда
Работа с объектами	Копирование, переименование и сохранение объекта	КопироватьОбъект (CopyObject) КопироватьФайлБазыДанных (CopyDatabaseFile) Переименовать (Rename) Сохранить (Save)
	Удаление объекта	УдалитьОбъект (DeleteObject)
	Изменение размеров или положения окна	Развернуть (Maximize) Свернуть (Minimize) СдвигРазмер (MoveSize) Восстановить (Restore)
	Открытие и закрытие объекта	ОткрытьФорму (OpenForm) ОткрытьФункцию (OpenFunction) ОткрытьМодуль (OpenModule) ОткрытьЗапрос (OpenQuery) ОткрытьОтчет (OpenReport) ОткрытьТаблицу (OpenTable) ОткрытьСтраницуДоступаКДанным (OpenDataAccessPage) ОткрытьСхему (OpenDiagram) ОткрытьСохраненнуюПроцедуру (OpenStoreProcedure) ОткрытьПредставление (OpenView) Закрыть (Close)
	Печать объекта	Печать (PrintOut)
	Выделение объекта	ВыделитьОбъект (SelectObject)
	Задание значения поля, элемента управления или свойства	ЗадатьЗначение (SetValue)
	Обновление объекта	ОбновитьОбъект (RepaintObject)
Прочие	Создание специальной или общей строки меню, специального или глобального контекстного меню	ДобавитьМеню (AddMenu)
	Задание состояния пунктов меню в специальной или общей строке меню	ЗадатьКомандуМеню (SetMenuItem)
	Вывод информации на экран	ВыводНаЭкран (Echo) ПесочныеЧасы (Hourglass) Сообщение (MsgBox) УстановитьСообщения (SetWarnings)

Таблица 12.1 *(окончание)*

Категория	Назначение	Макрокоманда
(прод.)	Генерация нажатий клавиш	`КомандыКлавиатуры` (SendKeys)
	Вывод на экран или скрытие встроенной или специальной панели инструментов	`ПанельИнструментов` (ShowToolbar)
	Подача звукового сигнала	`Сигнал` (Beep)

Как видно из этой таблицы, действия, которые могут быть выполнены с помощью макросов, очень разнообразны. Даже если вы не будете использовать макросы в своих приложениях, имеет смысл изучить эти макрокоманды, поскольку их можно использовать и в процедурах VBA, как будет показано далее *(см. гл. 13)*.

Использование макросов оправдано тем, что их легко создавать, и для этого не нужно изучать синтаксис языка программирования. Как будет показано в следующем разделе, для создания макроса требуется знать лишь основные приемы работы в Microsoft Access и Windows, такие как перетаскивание объектов из окна **База данных** (Database) в специальное окно — **Конструктор макросов** (Macro Design), выбор действия из списка и ввод выражений в качестве аргументов макрокоманды. Поэтому, если очень не хочется изучать синтаксис языка VBA или это представляется слишком трудным, смело применяйте макросы, и вы получите достаточно функциональное приложение.

Как уже говорилось ранее, основное назначение макросов — это создание удобного интерфейса приложения: чтобы формы и отчеты открывались при нажатии кнопок в форме или на панели инструментов или же привычным выбором команды меню; чтобы при открытии приложения пользователь видел на экране не окно **База данных** (Database), наполненное множеством таблиц, запросов, форм и отчетов, а некую понятную форму, с помощью которой можно было бы сразу производить желаемые действия, и т. д.

С помощью макросов можно создать удобные меню и панели инструментов для приложения, и чуть позже мы опишем, как это сделать.

Однако использование макросов имеет и некоторые недостатки, о которых здесь уместно сказать.

- Возможности макрокоманд ограничены по сравнению с возможностями языка VBA, поэтому в ряде случаев без программирования на VBA не обойтись, хотя сначала нужно быть уверенным, что эти дополнительные возможности действительно нужны. Язык VBA предоставляет более широкие возможности для работы с данными, позволяет использовать механизм программирования объектов для связи с другими приложениями, вызывать функции из динамически связываемых библиотек (Dynamic Link Library, DLL) Windows и создавать собственные специализированные функции.
- Макросы можно использовать практически везде, где используются процедуры VBA, однако процедуры VBA, как правило, выполняются быстрее.

- Макросы являются объектами, существующими отдельно от форм и отчетов, в которых они используются, поэтому, когда этих объектов становится очень много, их поддержка достаточно трудоемка. Процедуры обработки событий VBA, используемые в формах и отчетах, являются неотъемлемой частью этих форм и отчетов, и в этом есть свои преимущества. Например, при переносе форм и отчетов из одной базы данных в другую с ними автоматически переносятся связанные процедуры.

Тем не менее, как уже отмечалось, во многих случаях использование макросов вполне оправдано, т. к. это существенно упрощает и ускоряет разработку приложения. С помощью макросов можно создавать небольшие однопользовательские приложения, хотя при создании многопользовательских или клиент-серверных приложений, где скорость работы приложения критична, лучше применять VBA.

Создание макросов

В Access существует специальное средство для создания макросов — **Конструктор макросов** (Macro Design). Рассмотрим процесс создания макроса с его помощью.

Конструктор макросов

Открыть окно **Конструктор макросов** (Macro Design) для создания нового макроса можно разными способами:

1. Раскройте список макросов в окне **База данных** (Database), щелкнув левой кнопкой мыши по ярлыку **Макросы** (Macros), и нажмите кнопку **Создать** (New) в верхней части окна.
2. Щелкните по стрелке на кнопке **Новый объект** (New Object) панели инструментов **База данных** (Database) и выберите элемент **Макрос** (Macro).
3. Выполните команду меню **Вставка | Макрос** (Insert | Macros)

Окно Конструктора построено аналогично Конструктору таблиц, т. е. разделено по горизонтали на две части: панель описаний и панель аргументов.

Верхняя часть окна Конструктора — панель описаний — состоит из нескольких столбцов. По умолчанию на этой панели выводится два столбца: **Макрокоманда** (Action) и **Примечание** (Comments). Панель описаний позволяет определить последовательность макрокоманд, из которых состоит макрос.

Строка в столбце **Макрокоманда** (Action) представляет собой поле со списком, в котором можно выбрать нужную макрокоманду.

Строка в столбце **Примечание** (Comments) — это обычное текстовое поле, в которое можно ввести комментарий, описывающий выполняемое действие.

Когда поле **Макрокоманда** (Action) заполнено, в нижней части окна Конструктора макроса появляется панель аргументов, предназначенная для ввода значений аргументов соответствующей макрокоманды. Список полей на этой панели зависит от выбранной макрокоманды и может отсутствовать, если макрокоманда не имеет аргументов. Таким образом, при создании макросов не нужно запоминать список аргументов для каждой макрокоманды.

Панель инструментов **Конструктор макросов** представлена на рис. 12.1, а в табл. 12.2 приведены описания ее кнопок.



Рис. 12.1. Панель инструментов **Конструктор макросов**

Таблица 12.2. *Панель инструментов* ***Конструктор макросов***

Кнопка	Описание	Команда меню
Имена макросов (Macro Name)	Позволяет отобразить или скрыть столбец с именами макросов в группе	**Вид	Имена макросов** (View | Macro Name)
Условия (Conditions)	Позволяет отобразить или скрыть столбец с условиями выполнения макрокоманд	**Вид	Условия** (View | Conditions)
Добавить строки (Insert Rows)	Добавляет строку в макрос	**Вставка	Строки** (Insert | Rows)
Удалить строки (Delete Rows)	Удаляет строку в последовательности макрокоманд	**Правка	Удалить строки** (Edit | Delete Rows)
Запуск (Run)	Запускает макрос на выполнение	**Запуск	Запуск** (Run | Run)
По шагам (Step)	Включает режим пошагового выполнения макроса	**Запуск	По шагам** (Run | Step)

На рис. 12.2 проиллюстрирован выбор макрокоманды `ПрименитьФильтр` (ApplyFilter). На панели аргументов имеются два поля, которые позволяют задать имя фильтра и условие отбора записей. Для того чтобы задать условие отбора записей, можно воспользоваться построителем выражений *(см. разд. "Создание выражений с помощью Построителя выражений" гл. 4)*.

Такой макрос можно использовать в форме или отчете для того, чтобы ограничить число отображаемых записей.

Совет

Для быстрого переключения между панелями описаний и аргументов, так же как при работе с таблицами и запросами в режиме Конструктора, можно использовать клавишу <F6>.

Существует еще один очень простой способ ввода макрокоманды. Допустим, нужно создать макрос, который должен открывать форму `Клиенты` (Customers) в приложении `Борей` (Northwind). Для этого в окне **База данных** (Database) необходимо щелкнуть по ярлыку **Формы** (Forms), затем, используя панель инструментов, открыть **Конструктор макросов** (Macro Design) и перетащить с помощью мыши форму `Клиенты` (Customers) в поле **Макрокоманда** (Action) окна Конструктора макросов.

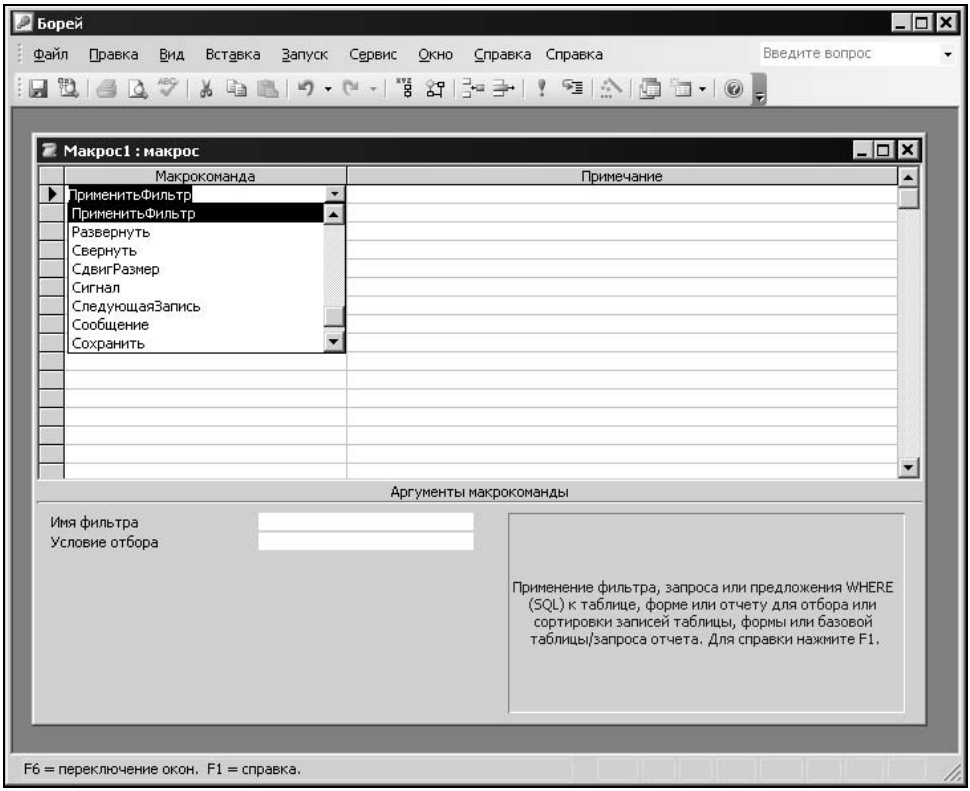


Рис. 12.2. Окно Конструктора макросов

Совет

Для удобства перетаскивания объектов из окна **База данных** (Database) в форму выберите команду **Окно | Слева направо** (Window | Tile Vertically), чтобы расположить окна на экране рядом.

Тогда в поле **Макрокоманда** (Action) автоматически появится макрокоманда `ОткрытьФорму` (OpenForm) (рис. 12.3), а на панели аргументов автоматически заполнятся поля: в поле **Имя формы** (Form Name) появится имя формы `Клиенты` (Customers), в поле **Режим** (View) — значение по умолчанию **Форма** (Form), в поле **Режим окна** (Window Mode) — значение **Обычное** (Normal). Остальные поля останутся пустыми. Мы можем в этой же макрокоманде задать фильтр для отбора записей. Для этого введем в поле **Имя фильтра** (Filter Name) имя какого-нибудь фильтра, например `Менеджеры по продажам`, который мы строили при изучении запросов *(см. разд. "Создание запроса из фильтра" гл. 4)*.

Кроме того, можно сделать записи в этой форме недоступными для изменений, для чего в поле **Режим данных** (Data Mode) необходимо ввести значение **Только чтение** (Read Only).

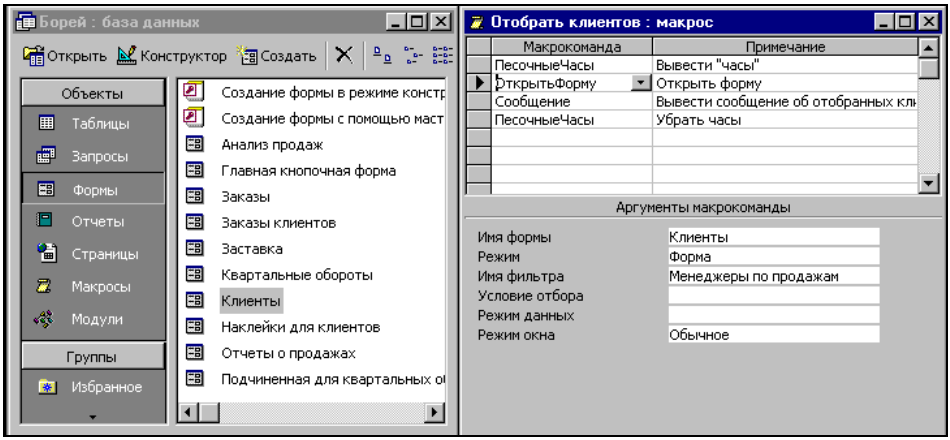


Рис. 12.3. Автоматическое создание макрокоманд путем перетаскивания объектов

Аналогично можно создавать макрокоманды, перетаскивая из окна **База данных** (Database) таблицы, запросы, отчеты и другие макросы. При этом в окне Конструктора макросов будут появляться соответствующие макрокоманды: `ОткрытьТаблицу` (OpenTable), `ОткрытьЗапрос` (OpenQuery), `ОткрытьОтчет` (OpenReport), `ЗапускМакроса` (RunMacro).

Для ввода аргументов макрокоманды чаще всего требуется выбирать значения из списков или вводить выражения. Для ввода выражений можно воспользоваться Построителем выражений, кнопка которого находится справа от поля аргумента. Еще одна кнопка Построителя выражений располагается на панели инструментов. Как и в других случаях, перед выражением нужно ставить знак равенства (=). Исключения составляют аргумент **Выражение** (Expression) макрокоманды `ЗадатьЗначение` (SetValue) и аргумент **Число повторений** (Repeat Count) макрокоманды `ЗапускМакроса` (RunMacro). Если ввести знак равенства перед выражением, задающим значение этих аргументов, то оно будет вычисляться дважды, что может привести к нежелательным результатам.

Если в качестве аргумента макрокоманды необходимо задать имя объекта базы данных, то его можно либо ввести с помощью клавиатуры, либо выбрать из раскрывающегося списка, либо указать название объекта, перетащив его из окна базы данных.

Итак, созданная нами макрокоманда будет открывать форму `Клиенты` (Customers) в режиме **Только чтение** (Read Only) и показывать в ней только менеджеров по продажам. Как уже было сказано ранее, макрос может содержать последовательность из нескольких макрокоманд. Поэтому в этом же макросе в следующей строке мы выберем макрокоманду `Сообщение` (MsgBox), которая после открытия формы будет отображать диалоговое окно с сообщением о том, какие клиенты представлены в форме. Для этого введем в поле **Сообщение** (Message) на панели аргументов текст сообщения, например `Установлен фильтр Менеджеры по продажам`, остальные поля можем оставить без изменения. Аргумент **Сигнал** (Beep) определяет, будет ли вывод сообщения сопровождаться звуковым сигналом. Аргумент **Тип** (Type) определяет вид

диалогового окна сообщения. Аргумент **Заголовок** (Title) позволяет задать текст, который будет выведен в заголовке окна сообщения.

Если база данных большая, отбор записей в форме может занять некоторое время, поэтому рекомендуется вывести на экран курсор в форме песочных часов, который будет показывать, что идет обработка данных. Для того чтобы сделать это, воспользуемся макрокомандой `ПесочныеЧасы` (Hourglass). Макрокоманды в макросе выполняются в той последовательности, в которой они записаны в столбце **Макрокоманда** (Action), т. е. сверху вниз. Макрокоманду `ПесочныеЧасы` (Hourglass) следует поставить первой, т. к. курсор должен измениться перед выводом формы. Таким образом, нам нужно вставить новую строку на панели описаний. Для этого установим курсор в первую строку панели и нажмем кнопку **Добавить строки** (Insert Rows) на панели инструментов или просто клавишу <Insert>. Появится пустая строка. В этой строке в поле **Макрокоманда** (Action) выберем макрокоманду `ПесочныеЧасы` (Hourglass). В поле **Включить** (Hourglass On) панели аргументов появится значение **Да** (Yes), установленное по умолчанию, что нам и нужно. Чтобы вернуть первоначальный вид указателю мыши, в конце созданного макроса нужно ввести такую же макрокоманду `ПесочныеЧасы` (Hourglass), но со значением **Нет** (No) аргумента **Включить** (Hourglass On).

Строго говоря, эту макрокоманду лучше поставить перед макрокомандой вывода сообщения на экран. Сделать это элементарно, поскольку макрокоманды легко меняются местами с помощью мыши. Чтобы поменять две последние макрокоманды местами, выделите последнюю строку, щелкнув левой кнопкой мыши по области выделения строк (слева в окне конструктора) и перетащите строку вверх. Горизонтальная линия показывает место нового положения строки. Отпустите кнопку мыши, когда эта линия окажется над предпоследней строкой.

Теперь остается только сохранить созданный макрос. Для этого можно выбрать в меню **Файл** (File) команду **Сохранить как** (Save As) или нажать клавишу <F12>. Появится окно сохранения макроса. Занесите в поле ввода имя макроса, соблюдая правила именования объектов Access, например `Менеджеры по продажам`.

Замечание

В Microsoft Word и Excel имеются средства автоматической записи макросов. Эти средства очень удобны для изучения VBA, поскольку позволяют автоматически создать процедуру, включив запись макроса, выполнив необходимые действия и остановив запись. Тогда все действия пользователя, выполняемые после включения записи и до ее остановки, преобразуются в текст процедуры VBA. Затем этот текст можно просмотреть и отредактировать с помощью редактора VBA. В Microsoft Access средств автоматической записи макросов нет, так же как нет и средств автоматической генерации кода VBA (за исключением того, что создают Мастера).

Создание групп макросов

При разработке приложения с использованием макросов их количество может оказаться очень большим. Поэтому важно хорошо организовать доступ к нужным макросам. Для этого в одном объекте **Макрос** (Macro) можно объединить несколько макросов. Например, рекомендуется все макросы, связанные с событиями в форме или отчете, объединить в отдельный объект, соответствующий данной форме или

отчету. Для того чтобы можно было это сделать, каждый макрос группы должен иметь свое имя, а имя объекта **Макрос** (Macro) будет являться именем группы макросов.

Чтобы создать группу макросов, выполните следующее:

1. Откройте Конструктор макросов одним из тех способов, которые были описаны в предыдущем разделе.
2. Нажмите кнопку **Имена макросов** (Macro Name) на панели инструментов. На панели описаний в окне Конструктора появится еще один столбец — **Имя макроса** (Macro Name) (рис. 12.4).
3. В этот столбец введите имя первого макроса. Остальные поля данной строки лучше оставить пустыми — это облегчит перемещение и копирование макрокоманд.
4. Начиная со следующей строки, введите все макрокоманды макроса и соответствующие аргументы для каждой макрокоманды.
5. Пропустите одну строку.
6. Повторите шаги 3—5 для каждого макроса.

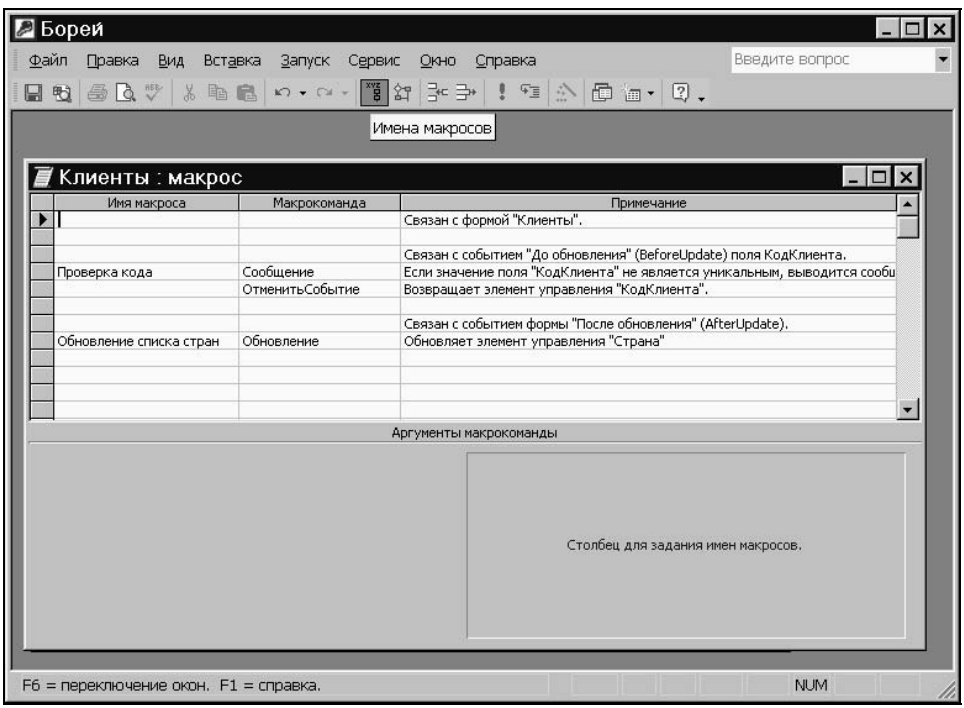


Рис. 12.4. Создание группы макросов

На рис. 12.4 в окне Конструктора представлен макрос `Клиенты` (Customers) базы данных `Борей` (Northwind). Для указания конца макроса не требуется отдельной мак-

рокоманды. Access продолжает выполнение макроса, входящего в группу, пока не обнаружит строку, содержащую имя в столбце **Имя макроса** (Macro Name), либо пока не дойдет до последней строки окна макросов.

При обработке макроса пустые строки игнорируются, поэтому их удобно использовать для разделения макросов в группе. Этот прием, а также ввод комментариев для сложных макросов, состоящих из большого числа макрокоманд, позволяет улучшить читаемость макросов.

Для запуска одного из макросов группы используется полное имя макроса, которое формируется таким образом: `имяГруппы.имяМакроса`.

Замечание

Можно настроить систему так, чтобы столбец **Имя макроса** (Macro Name) выводился всегда в Конструкторе макросов. Для этого необходимо установить соответствующие параметры на вкладке **Вид** (View) в диалоговом окне **Параметры** (Options) *(см. разд. "Настройка параметров приложения Access" гл. 14).*

Применение условий в макросах

В приведенных ранее примерах макросов использовался последовательный порядок выполнения макрокоманд. Однако, как и в любой программе, порядок выполнения макрокоманд в макросе может быть изменен. Для этого вводятся условия выполнения или пропуска макрокоманд. Эти условия задаются в виде выражений в специальном столбце, который появляется в окне Конструктора макросов, если нажата кнопка **Условия** (Conditions) на панели инструментов (см. рис. 12.1).

Замечание

Как и в случае с группами макросов, можно настроить параметры системы таким образом, чтобы этот столбец появлялся в Конструкторе макросов всегда.

На рис. 12.5 приведено окно Конструктора для группы макросов `Наклейки для клиентов` (Customer Label Dialog). Эти макросы используются для печати отчета в виде наклеек на конверты. Причем можно напечатать наклейки сразу для всех клиентов или для клиентов одной страны, выбранной из списка. В окне виден столбец **Условие** (Condition). Если условие, заданное в этом столбце, истинно, выполняется макрокоманда, находящаяся в этой строке. Если условие ложно, соответствующая макрокоманда пропускается и выполняется следующая.

Если требуется при истинности условия выполнить сразу несколько макрокоманд, то для всех макрокоманд, кроме первой, в столбце **Условие** (Condition) ставится многоточие (...). Условие, которое должно проверяться, пишется в строке первой макрокоманды. Тогда, если оно истинно, выполняется весь набор макрокоманд от этого условия и до следующей макрокоманды с заданным условием, до следующего макроса или до конца макроса. Если условие ложно, пропускаются все команды, помеченные многоточием, включая макрокоманду с заданным условием. Далее выполняется макрокоманда (следующая за пропущенными), в которой содержится новое условие или поле **Условие** (Condition) не заполнено. Таким образом, в отличие от большинства языков программирования, в макросах нет альтернативного ветвле-

ния. Для того чтобы создать макрос с двумя альтернативными ветвями, нужно сначала ввести условие и определить макрокоманды, исполняемые при выполнении этого условия. Сразу после них необходимо указать обратное условие и определить макрокоманды, исполняемые в альтернативной ветви. Затем можно ввести макрокоманды, которые исполняются безусловно. Такой способ ветвления используется, например, в макросе `Просмотр` (Preview), который связан с кнопкой **Просмотр** (Preview) в данной форме. Если пользователь выбрал вариант **для всех стран** (All Countries), что определяется значением 1 элемента управления `ПечатьНаклеек` (PrintLabelFor), то отчет открывается в режиме предварительного просмотра и форма закрывается. Если выбран переключатель **для конкретной страны** (Specific Country), а страна не выбрана (значение поля со списком `ВыборСтраны` (SelectCountry) не задано), то выдается сообщение, фокус устанавливается в поле со списком для выбора страны, и выполнение макроса прекращается. Если страна задана, то отображается отчет для выбранной страны и форма закрывается.

В этом макросе используется макрокоманда `ОстановитьМакрос` (StopMacro). Она позволяет прекратить выполнение макроса, при этом все макрокоманды, следующие по порядку за макрокомандой `ОстановитьМакрос` (StopMacro), не будут выполняться.

Замечание

Макрокоманду `ОстановитьМакрос` (StopMacro) имеет смысл использовать, только если за ней идут другие макрокоманды, которые не следует выполнять при выполнении некоторого условия. Как уже сказано ранее, для указания конца макроса специальной макрокоманды не требуется.

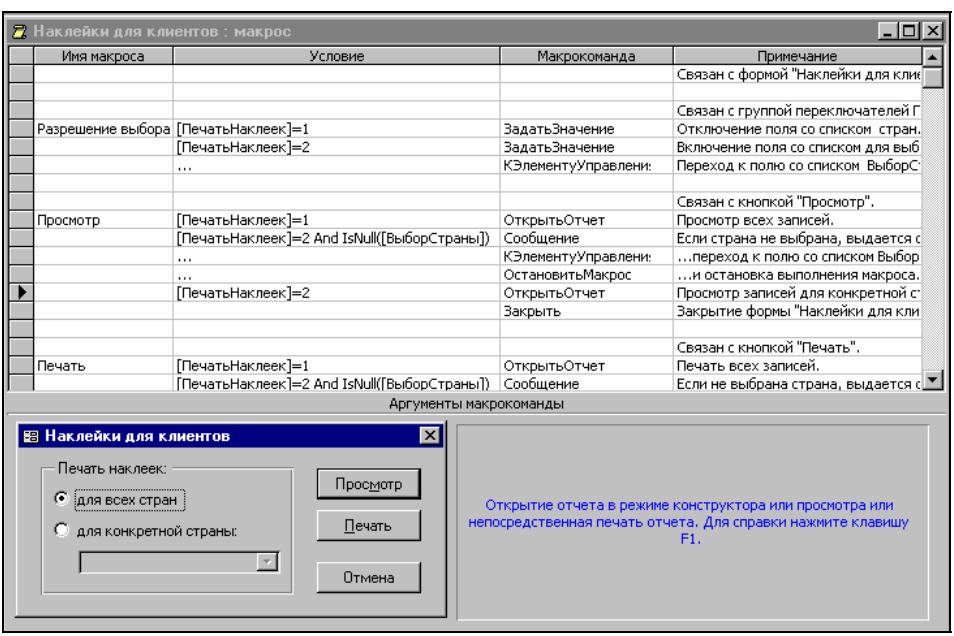


Рис. 12.5. Применение условий в макросах

Выполнение и отладка макросов

Способов запуска макросов очень много. Среди них есть прямые способы, когда макрос запускается непосредственно пользователем. Эти способы используются обычно при тестировании макросов. Когда макрос отлажен, создают специальную команду меню или кнопку на панели инструментов или связывают макрос с каким-либо событием в форме или отчете. В данном разделе мы рассмотрим все возможные способы запуска макроса.

Универсальным способом запуска макроса является использование команды меню **Сервис | Макрос | Выполнить макрос** (Tools | Macro | Run Macro) (рис. 12.6).

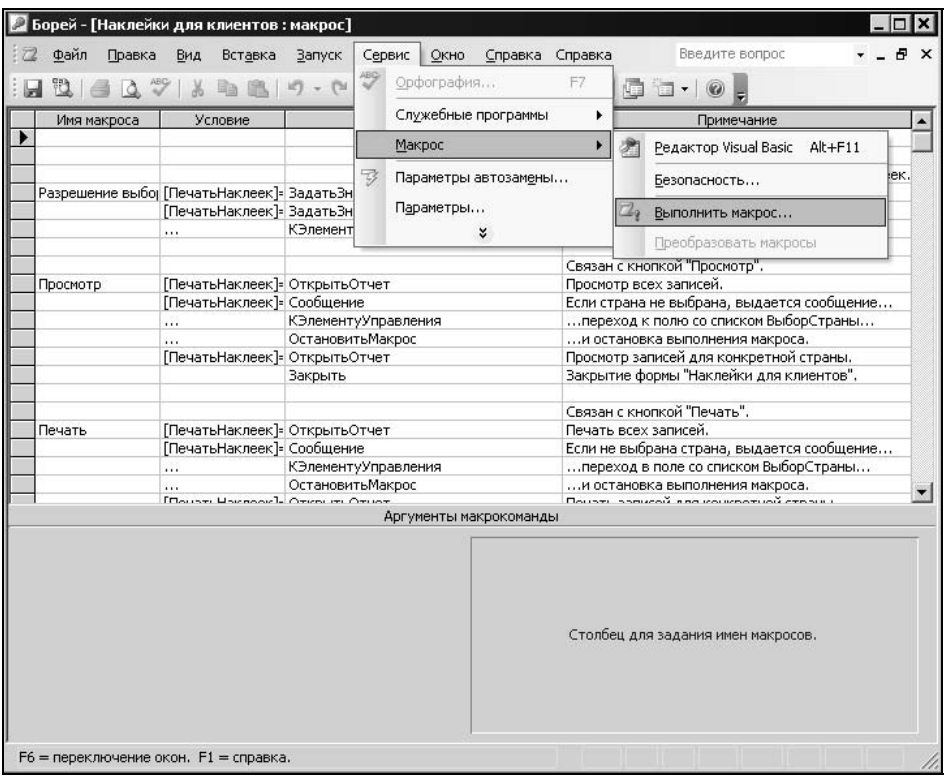


Рис. 12.6. Запуск макроса с помощью команды меню

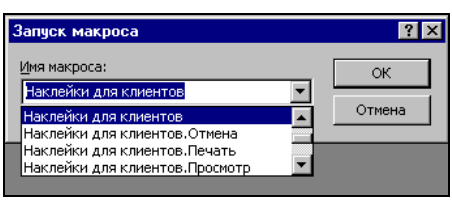


Рис. 12.7. Диалоговое окно **Запуск макроса**

В появившемся при выборе этой команды диалоговом окне нужно указать (ввести или выбрать из списка) полное имя макроса, т. е. `имяГруппы.имяМакроса` (рис. 12.7). Этот способ позволяет выполнить макрос из любого активного окна, будь то окно макросов, окно базы данных или, например, Конструктор форм.

Запуск макроса из окна Конструктора макроса

Этот способ применяется для тестирования только что созданного или исправленного макроса. Если макрос в окне Конструктора один, то для его запуска нужно просто нажать кнопку **Запуск** (Run) на панели инструментов или выбрать команду **Запуск | По шагам** (Run | Single Step) (последний способ позволяет отладить макрос). Однако если макросов несколько, то с помощью данной кнопки или команды можно запустить только первый макрос в группе. Чтобы запустить любой другой макрос из группы, нужно воспользоваться командой меню **Сервис | Макрос | Выполнить макрос** (Tools | Macro | Run Macro).

Если вы часто используете этот способ запуска макроса, имеет смысл настроить панель инструментов, добавив на нее стандартную кнопку **Выполнить макрос** (Run Macro), эквивалентную выполнению команды меню **Сервис | Макрос | Выполнить макрос** (Tools | Macro | Run Macro). *(О настройке панелей инструментов см. разд. "Создание и изменение панелей инструментов" гл. 14.)*

Запуск макроса из окна базы данных

Чтобы запустить макрос из окна **База данных** (Database), щелкните по ярлыку **Макросы** (Macros) на панели объектов, найдите в списке нужный макрос и дважды щелкните по нему кнопкой мыши или нажмите кнопку **Запуск** (Run) на панели инструментов окна **База данных** (Database). Последний способ подходит, только если в выбранном объекте содержится один макрос. В противном случае нужно воспользоваться универсальным способом выполнения макроса, описанным ранее.

Замечание

Если в условиях или аргументах макрокоманд есть ссылки на объекты Access — формы, отчеты и пр., — они должны быть открыты перед запуском макроса, в противном случае будут выдаваться сообщения об ошибках.

Запуск макроса с помощью кнопки на панели инструментов

Прямые способы запуска макросов являются простыми, но не самыми быстрыми. Существуют более удобные и быстрые способы. Наиболее удобный, с нашей точки зрения, способ запуска макроса — это создание специальной кнопки на панели инструментов. Правда, стандартных панелей инструментов в Access много, и в разных режимах работы на экране появляются соответствующие панели. Поэтому наиболее разумным является создание специальной панели инструментов, на которую вы можете поместить все нужные кнопки. Тем не менее, операция добавления специальной кнопки на стандартную панель инструментов чрезвычайно проста.

Чтобы добавить кнопку на стандартную панель инструментов Access, выполните следующее:

1. Щелкните правой кнопкой мыши по панели инструментов и выберите в контекстном меню команду **Настройка** (Customize).
2. В появившемся диалоговом окне **Настройка** (Customize) раскройте вкладку **Панели инструментов** (Toolbars).
3. Установите флажок для одной из стандартных панелей инструментов, например **Запрос в режиме таблицы** (Query Datasheet), чтобы она отобразилась на экране.
4. Закройте диалоговое окно **Настройка** (Customize), нажав кнопку **Закрыть** (Close).
5. Щелкните по ярлыку **Макросы** (Macros) на панели объектов в окне **База данных** (Database) и найдите нужный макрос.
6. Перетащите макрос с помощью мыши на панель инструментов **Запрос в режиме таблицы** (Query Datasheet).

На панели инструментов появится кнопка со значком и именем макроса (рис. 12.8). Вы можете таким образом переместить на панель инструментов созданный нами макрос `Менеджеры по продажам`. Теперь всякий раз, когда эта панель инструментов будет отображаться на экране, вам будет доступна кнопка запуска данного макроса.

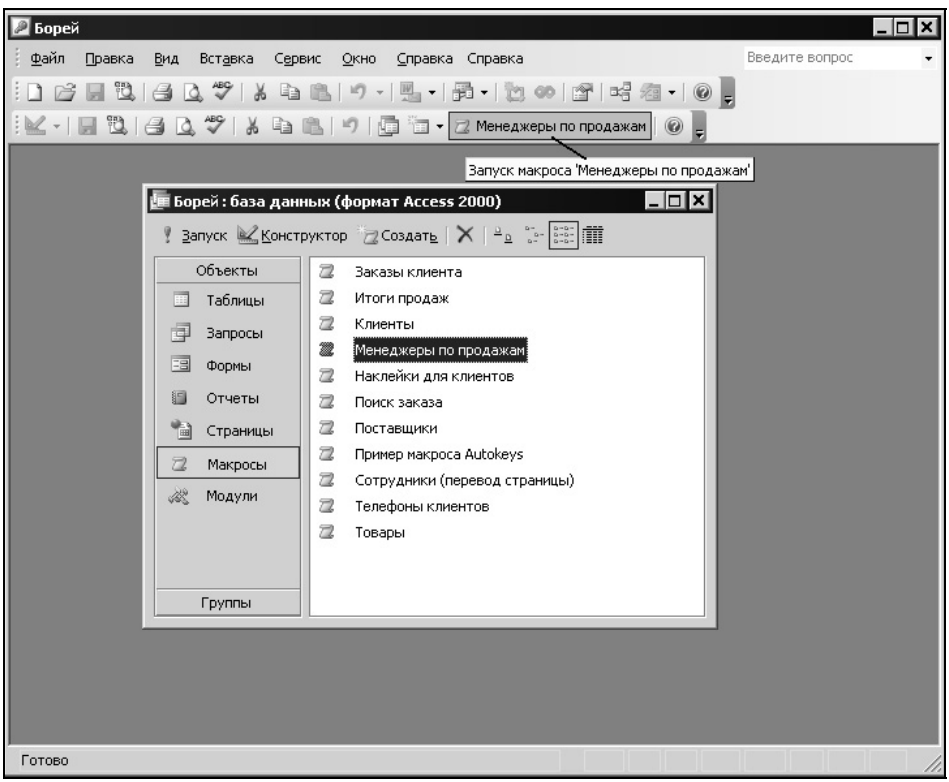


Рис. 12.8. Создание кнопки на панели инструментов для запуска макроса

Замечание

Если основной режим работы в вашем приложении — работа с формами, кнопки выполнения макросов можно поместить на панель инструментов **Режим формы** (Form View).

Чтобы удалить с экрана панель инструментов, на которую вы только что добавили кнопку, щелкните правой кнопкой мыши в области панелей инструментов и в списке стандартных панелей команд снимите флажок слева от названия панели (рис. 12.9).

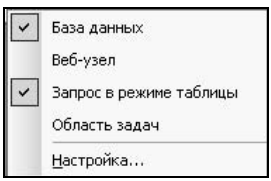


Рис. 12.9. Контекстное меню со списком панелей команд

Теперь рассмотрим создание специальной панели инструментов и размещение на ней кнопки вызова макроса.

1. Щелкните по ярлыку **Макросы** (Macros) на панели объектов в окне **База данных** (Database) и выделите нужный макрос.
2. Выберите команду **Сервис | Макрос | Создать панель инструментов из макроса** (Tools | Macro | Create Toolbar from Macro). На экране появится пустая панель инструментов.
3. Перетащите с помощью мыши нужный макрос на эту панель.

На панели появится кнопка с уже знакомым нам значком макроса. Кнопка получает такое же название, как имя макроса, для которого она создана.

Если нужно создать панель инструментов для вызова нескольких макросов, мы рекомендуем объединить эти макросы в одну группу (создание группы макросов было описано в предыдущем разделе). Тогда при выполнении команды меню **Сервис | Макрос | Создать панель инструментов из макроса** (Tools | Macro | Create Toolbar from Macro) будет создана панель инструментов, на которой каждому макросу группы будет соответствовать своя кнопка. При этом на кнопках отображаются имена соответствующих макросов. Новая панель инструментов получает такое же имя, как имя группы макросов. Попробуйте выполнить эту команду для одной из групп макросов в базе данных `Борей` (Northwind), например `Клиенты` (Customers). При желании можно заменить текст на кнопках значками. Дополнительная информация о настройке панелей инструментов приложения, и в частности об удалении кнопок с панелей и удалении самих панелей, изменении вида этих кнопок, представлена в *разд. "Создание и изменение панелей команд" гл. 14.*

Запуск макроса с помощью команды меню

Еще одним распространенным способом запуска макроса является запуск с помощью специально созданной для этого команды меню. Способы создания команд меню

для запуска макросов очень похожи на те, что применяются для создания кнопок на панели инструментов. Например, можно воспользоваться для этого командой меню **Сервис | Макрос | Создать меню из макроса** (Tools | Macro | Create Menu from Macro).

Более подробно создание и настройка пользовательских меню описаны в *разд. "Создание и изменение панелей команд" гл. 14*. Поэтому в данном разделе мы не будем рассматривать создание команды меню из макроса.

Запуск макроса с помощью комбинации клавиш

Для запуска макроса можно назначить комбинацию клавиш. Для этого необходимо создать специальную группу макросов — `AutoKeys`. Эта группа макросов должна для каждой назначенной вами комбинации клавиш содержать макрокоманду запуска соответствующего макроса. Пример группы макросов `AutoKeys` приведен в базе данных `Борей`. Откройте данную группу в режиме Конструктора. Она содержит один макрос (рис. 12.10). Имя макроса — это запись комбинации клавиш <Ctrl>+<P>. Сам макрос состоит из одной макрокоманды `ЗапускМакроса` (RunMacro), которая запускает макрос `Телефоны клиентов.Печать` (Customer Phone List.Print). Макрос `AutoKeys` просматривается каждый раз, когда пользователь использует специальные комбинации клавиш, например <Ctrl>+<P>. Если введенная комбинация клавиш найдена в `AutoKeys`, то запускается соответствующий макрос.

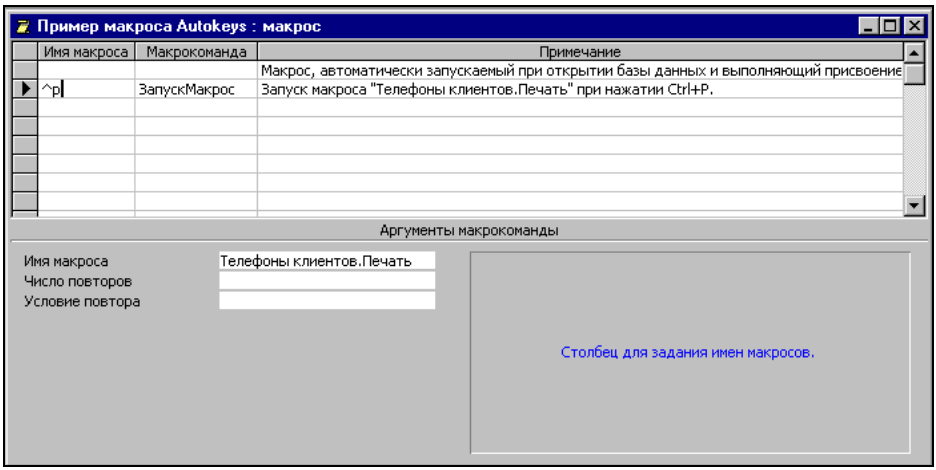


Рис. 12.10. Пример макроса `AutoKeys`

Этот способ запуска макроса, безусловно, является самым быстрым, однако он имеет существенный недостаток: свободных комбинаций клавиш очень немного. И хотя комбинации, определенные в `AutoKeys`, имеют более высокий приоритет, чем стандартные (например, <Ctrl>+<C> — копировать), заменять стандартное назначение комбинаций клавиш не рекомендуется. Разрешенные комбинации клавиш приведены в табл. 12.3. Комбинации с клавишей <Alt> в данном случае не применяются, т. к. используются для запуска команд меню и нажатия кнопок.

Таблица 12.3. *Разрешенные комбинации клавиш*

Обозначение	Комбинации клавиш
^A или ^4	<Ctrl>+буква или <Ctrl>+цифра
{F1}	Любая функциональная клавиша
^{F1}	<Ctrl>+любая функциональная клавиша
+{F1}	<Shift>+любая функциональная клавиша
{Insert}	<Insert>
^{Insert}	<Ctrl>+<Insert>
+{Insert}	<Shift>+<Insert>
{Delete} или {Del}	<Delete>
^{Delete} или ^{Del}	<Ctrl>+<Delete>
+{Delete} или +{Del}	<Shift>+<Delete>

Запуск макроса при открытии базы данных

При открытии базы данных Access могут выполняться некоторые действия. Чаще всего это открытие специальной формы, так называемой Главной кнопочной формы, вывод специальных меню или панелей инструментов, скрытие стандартных меню и т. д. Для определения этих действий используется диалоговое окно **Параметры запуска** (Startup). Однако иногда требуется при запуске приложения выполнить более сложный набор действий, чем тот, что позволяет задать это диалоговое окно. Например, может потребоваться заранее открыть несколько форм (не показывая их на экране), чтобы потом их отображение не занимало много времени, проверить некоторые условия или запросить ввод некоторых данных. Все это можно сделать с помощью специального макроса, который называется `AutoExec`. При открытии базы данных Access проверяет наличие этого макроса и, если он существует, выполняет его. При создании макроса `AutoExec` следует помнить, что Access сначала выполнит действия, определенные в окне **Параметры запуска** (Startup), а затем — макрос `AutoExec`, поэтому в них не должно быть противоречивых действий.

Запуск макроса из другого макроса

Иногда требуется вызвать макрос из другого макроса. Это можно сделать с помощью макрокоманды `ЗапускМакроса` (RunMacro). Мы уже встречались с этой макрокомандой при описании макроса `AutoKeys`. Здесь нужно отметить, что данная макрокоманда имеет три аргумента: кроме имени макроса, задается число повторов выполнения макрокоманды и условие повтора. Таким образом, эта макрокоманда позволяет организовывать циклы. Аргумент **Число повторов** (Repeat Count) задает количество вызовов макроса. Аргумент **Условие повтора** (Repeat Expression) является выражением, которое может принимать значение `Истина` или `Ложь`. Перед выполнением макрокоманды `ЗапускМакроса` (RunMacro) проверяется значение этого выражения. Если оно `Истина`, то макрос выполняется, если `Ложь`, то макрос не выполня-

ется и управление передается следующей макрокоманде. Если эти два аргумента не заданы, макрос выполняется только один раз. Если заданы оба аргумента, цикл вызова прекращается, когда макрос выполнился заданное число раз, либо когда заданное условие окажется невыполненным и получит значение Ложь.

Замечание

Имя макроса в аргументе макрокоманды ЗапускМакроса (RunMacro) должно быть полным, т. е. должно иметь вид имяГруппы.имяМакроса, даже если вызываемый макрос находится в той же группе, что и вызывающий. Если условие повтора задано таким образом, что оно всегда истинно, то цикл окажется бесконечным. Можно прервать его с помощью комбинации клавиш <Ctrl>+<Break>. Если это не поможет, придется прервать работу Access, сняв задачу в Диспетчере задач Windows.

Назначение макроса событию

Наиболее часто макросы используются в приложении Access для обработки событий. *Событие* — это любое действие, распознаваемое объектом. Можно определить реакцию объекта на событие. События происходят в результате действий пользователя, выполнения инструкций VBA или генерируются системой. Примером событий является вывод на экран формы, отчета, ввод данных в текстовое поле, нажатие кнопки мыши или клавиши. Каждому из этих событий можно назначить макрос или процедуру VBA, которые будут автоматически выполняться в ответ на произошедшее событие. Практически все программирование в Access сводится к написанию макросов или процедур, обрабатывающих события, т. е. программируется реакция объектов на события. Существует большое количество различного рода событий, на которые реагируют объекты, причем часто возникает не одно, а целая последовательность событий. Поэтому необходимо обладать определенным умением, чтобы решить, какому событию следует назначить созданный вами макрос или процедуру VBA. Все тонкости этого выбора будут обсуждаться в *разд. "Программирование в формах и отчетах" гл. 13*. Здесь мы опишем только способ назначения макроса событию и приведем примеры обработки событий с помощью макросов.

Начнем с самого простого. Откроем форму Клиенты (Customers) в базе данных Борей. В этой форме отображается информация о клиенте. Допустим, мы хотели бы видеть не только эту информацию, но и данные о купленных клиентом продуктах. Было бы хорошо создать кнопку Заказы клиента, при нажатии которой появлялась бы форма Заказы (Orders) с заказами только того клиента, который в данный момент выбран в форме Клиенты (Customers). Для того чтобы получить желаемое, создадим макрос, который будет выполняться, когда произойдет событие **Нажатие кнопки** (On Click) в форме Клиенты (Customers).

Чтобы создать макрос, определяющий описанную реакцию приложения на событие **Нажатие кнопки** (OnClick), выполните следующее:

1. Откройте форму Клиенты (Customers) в режиме Конструктора.
2. Создайте кнопку в области заголовка формы. При этом кнопка Мастера на панели элементов должна быть отжата, т. к. в противном случае вам будет предложено создать процедуру обработки событий, а не макрос.
3. Откройте окно свойств только что созданной кнопки, если оно еще не открыто, и раскройте вкладку **События** (Event).

4. Обратите внимание, сколько разных событий связано только с командной кнопкой. Помимо обычного нажатия, которое мы сейчас и будем использовать, в набор событий кнопки входят получение и потеря фокуса, двойной щелчок кнопкой мыши, простое перемещение указателя мыши над кнопкой и пр. Такое многообразие событий дает разработчику большие возможности по созданию удобного интерфейса пользователя. Теперь найдите в списке событие **Нажатие кнопки** (On Click) и установите курсор в соответствующую ячейку. Это поле со списком, и в данный момент оно пустое. Если открыть этот список, то первым его элементом будет **[Процедура обработки событий]** ([Event procedure]), а далее идет перечень всех макросов, существующих в приложении (рис. 12.11).

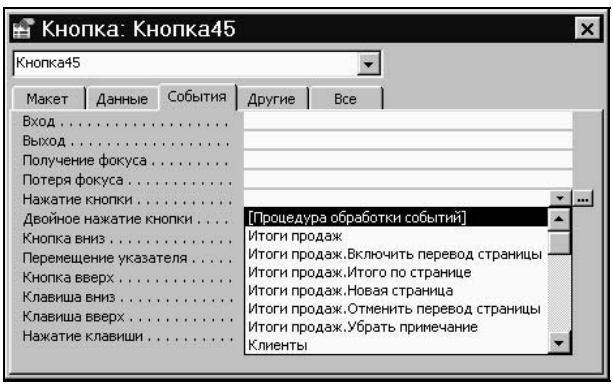


Рис. 12.11. Диалоговое окно свойств кнопки

5. Поскольку нужного нам макроса в списке нет, давайте его создадим. Для этого нажмите кнопку Построителя, находящуюся справа от поля. Появится окно **Построитель** (Choose Builder), в котором предлагается выбрать один из трех Построителей: **Выражения** (Expression Builder), **Макросы** (Macro Builder) и **Программы** (Code Builder) (рис. 12.12).

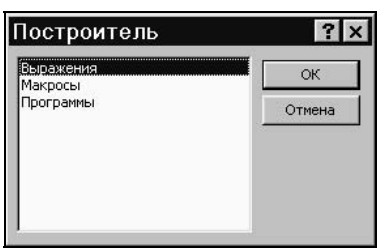


Рис. 12.12. Диалоговое окно **Построитель**

6. Выберите **Макросы** (Macro Builder) и нажмите кнопку **OK**. Откроется окно макросов и диалоговое окно, в которое нужно ввести имя создаваемого макроса. Введите имя `Заказы клиента`.
7. В макрос нужно добавить единственную макрокоманду `ОткрытьФорму` (OpenForm). Значения аргументов этой макрокоманды приведены в табл. 12.4.

Таблица 12.4. *Макрокоманда* `ОткрытьФорму`

Аргумент	Значение	Пояснение
Имя формы (Form Name)	**Заказы** (Orders)	Указывается имя открываемой формы. Это обязательный аргумент
Режим (View)	**Форма** (Form)	Определяет режим, в котором форма должна быть открыта. Он может принимать значения: **Форма** (Form), **Конструктор** (Design), **Просмотр** (Print Preview), **Таблица** (Datasheet), **Сводная таблица** (PivotTable) и **Сводная Диаграмма** (PivotChart)
Условие отбора (Condition)	`[КодКлиента]= [Forms]![Клиенты]![КодКлиента]`	Определяет условие для отбора записей, отображаемых в форме. Условие представляет собой выражение. В данном случае это выражение содержит ссылку на элемент управления `КодКлиента` (CustomerID) в форме `Клиенты` (Customers), что позволяет выбрать все записи из таблицы `Заказы` (Orders), в которых код клиента равняется значению, заданному в поле `КодКлиента` (CustomerID) формы `Клиенты` (Customers)
Режим данных (Data Mode)	**Только чтение** (Read Only)	Определяет способ работы с данными и может принимать одно из значений: **Добавление** (Add), **Изменение** (Edit) или **Только чтение** (Read Only)
Режим окна (Window Mode)	**Обычное** (Normal)	определяет тип окна: **Обычное** (Normal), **Невидимое** (Hidden), **Значок** (Icon) и **Окно диалога** (Dialog)

8. Закройте окно макроса, сохранив изменения. В окне свойств кнопки в поле **Нажатие кнопки** (On Click) появится имя макроса `Заказы клиента`.
9. Раскройте вкладку **Макет** (Format) и введите в поле **Подпись** (Caption) название кнопки: `Заказы клиента`. То же самое имя рекомендуется ввести в поле **Имя** (Name) на вкладке **Другие** (Other).

Теперь остается перейти в режим формы и проверить, как макрос отрабатывает событие **Нажатие кнопки** (On Click). Если вы нигде не ошиблись, то должны увидеть на экране картинку, похожую на ту, что изображена на рис. 12.13.

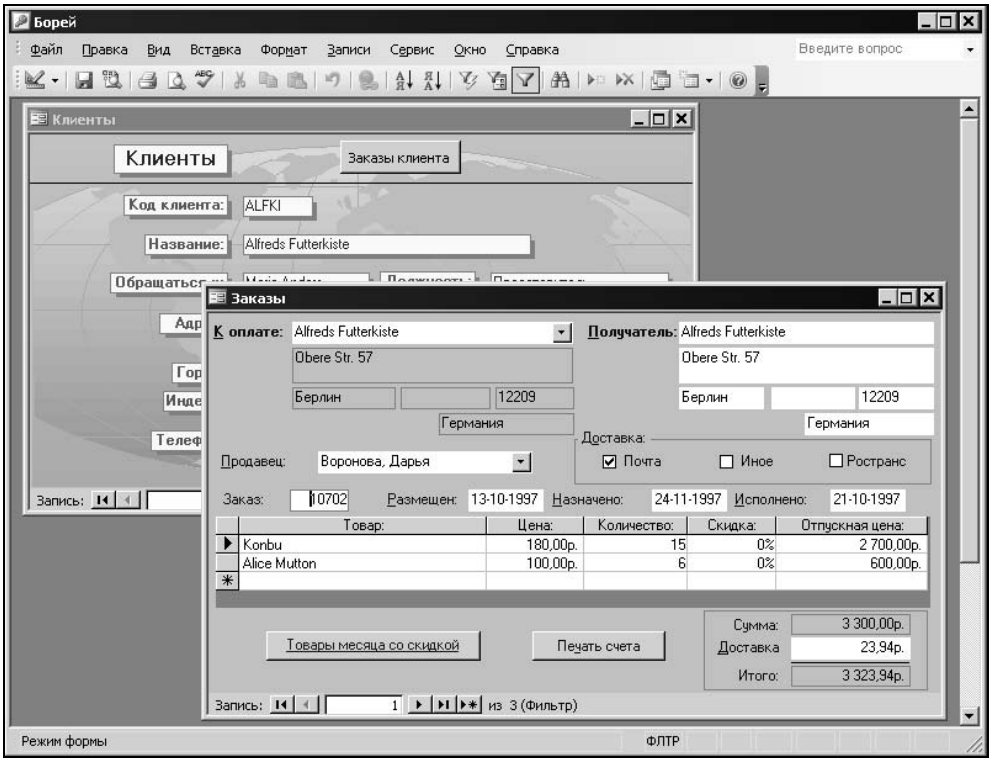


Рис. 12.13. Форма `Заказы`

Однако мы выполнили еще не все необходимые действия. Если перейти к следующей записи в форме `Клиенты` (Customers), данные в форме `Заказы` (Orders) уже не будут правильно отражать ситуацию — они не меняются. Нужно сделать так, чтобы эти данные изменялись синхронно с переходом к другим записям в форме `Клиенты` (Customers), либо чтобы это окно просто закрывалось. Рассмотрим реализацию второго варианта. Форма `Заказы` (Orders) открывается, когда мы нажимаем кнопку `Заказы клиента`, и становится активной. Требуется сделать так, чтобы она закрывалась, когда активной становится форма `Клиенты` (Customers). Для этого нужно выбрать соответствующее событие и задать процедуру или макрос его обработки. Если вы откроете окно **Свойства** (Properties) формы `Клиенты` (Customers) и раскроете вкладку **События** (Event), то среди множества событий увидите событие **Включение** (On Activate). Это как раз то событие, которое нам нужно. Выберем это событие, но теперь мы не будем создавать новую группу макросов, а добавим новый макрос в группу макросов `Клиенты` (Customers), которая уже содержит макросы для формы `Клиенты` (Customers). Выберите эту группу макросов в раскрывающемся списке и нажмите кнопку Построителя. Откроется окно, в котором отобразятся два макроса. Добавим к ним еще один макрос. Назовем его `Закрытие` и введем макрокоманду `Закрыть` (Close) с соответствующими параметрами (табл. 12.5). Но перед закрытием формы необходимо проверить, открыта ли она. Для этого воспользуемся функцией `isLoaded()`, которая возвращает значение `Истина`, если форма открыта, и `Ложь` —

в противном случае. В качестве аргумента в функцию `isLoaded()` нужно передать имя формы. Итак, в столбец **Условие** (Condition), напротив команды `Закрыть` (Close), введите `IsLoaded("Заказы")` (или `IsLoaded("Orders")`). Готовый макрос показан на рис. 12.14.

Таблица 12.5. *Макрокоманда* `Закрыть`

Аргумент	Значение	Пояснение
Тип объекта (Object Type)	**Форма** (Forms)	Определяет тип закрываемого объекта. Если этот аргумент не задан, то закрывается текущее окно
Имя объекта (Object Name)	**Заказы** (Orders)	Определяет имя закрываемого объекта
Сохранение (Save)	**Нет** (No)	Позволяет определить, будет ли Access при закрытии формы отображать диалоговое окно для подтверждения сохранения измененных данных. Поскольку мы открывали форму `Заказы` (Orders) в режиме только чтение, изменение данных запрещено, поэтому данному аргументу необходимо присвоить значение **Нет** (No)

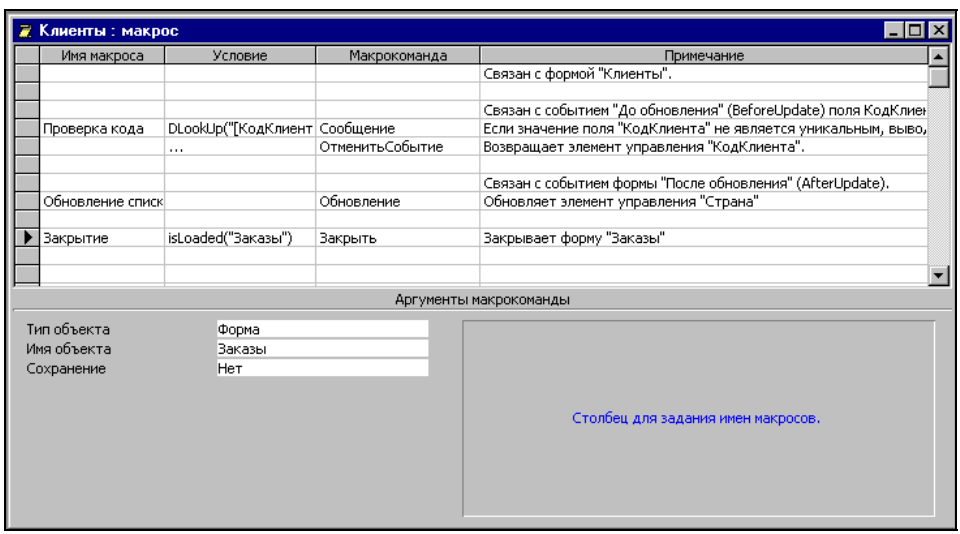


Рис. 12.14. Макрос для закрытия формы

Закройте окно макроса, сохранив изменения. Измените имя макроса, назначенного событию **Включение** (On Activate). Для этого выберите в списке или введите имя `Клиенты.Закрыть` (Customers.Close) и нажмите клавишу <Enter>. Теперь сохраните форму, перейдите в режим формы и проверьте, как работают созданные вами макросы.

Вызов макроса из процедуры VBA

Как мы уже говорили, процедуры VBA и макросы в Access являются объектами близкими и часто взаимозаменяемыми, т. е. вместо макроса можно использовать процедуру VBA и наоборот. Существуют способы запуска макроса из процедуры VBA, запуска процедуры VBA из макроса, кроме того, Access позволяет преобразовать макрос в процедуру VBA. Такое преобразование описано *в разд. "Преобразование макросов в процедуры VBA" данной главы.*

Для запуска процедуры VBA из макроса существует специальная макрокоманда `ЗапускПрограммы` (RunCode). Эта макрокоманда имеет один аргумент — имя вызываемой процедуры, хотя вызываться может только функция, а не подпрограмма.

Для запуска макроса из процедуры VBA применяется специальный метод `ЗапускМакроса` (RunMacro) объекта `DoCmd`, например:

```
DoCmd.RunMacro "Макрос1"
```

Объект `DoCmd` используется в процедуре VBA для выполнения макрокоманд Access. При этом английское имя нужной макрокоманды должно быть указано в качестве метода объекта `DoCmd`, например строка процедуры:

```
DoCmd.OpenForm "Клиенты"
```

позволяет открыть форму `Клиенты` (Customers). Здесь `"Клиенты"` — аргумент макрокоманды. Аргументы перечисляются в предложении `DoCmd` через запятую. Таким способом может быть выполнено большинство макрокоманд.

Создание процедур VBA мы рассмотрим далее в *разд. "Создание процедур обработки событий" гл. 13.*

Отладка макросов и поиск ошибок

Обычно для отладки макроса его выполняют в пошаговом режиме. Для включения этого режима необходимо нажать кнопку **По шагам** (Single Step) на панели инструментов в окне Конструктора макросов или выбрать команду меню **Запуск | По шагам** (Run | Single Step). Тогда перед выполнением каждой макрокоманды будет появляться диалоговое окно **Пошаговое исполнение макроса** (Macro Single Step) (рис. 12.15).

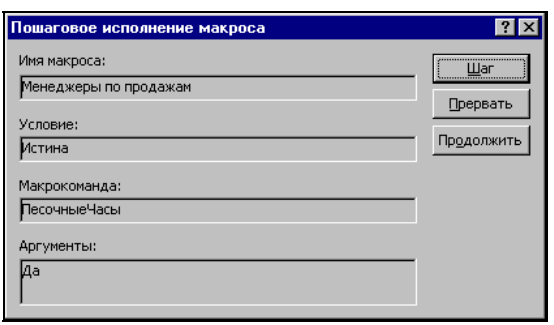


Рис. 12.15. Диалоговое окно **Пошаговое исполнение макроса**

В этом окне отображаются имя макроса, имя исполняемой макрокоманды, условие ее исполнения и аргументы. При этом в полях **Условие** (Condition) и **Аргументы** (Arguments) видны значения условного выражения и аргументов макрокоманды. Если в строке макрокоманды условие не указано, в поле **Условие** (Condition) всегда будет показываться значение **Истина** (True). Кнопки справа от полей в этом окне позволяют выполнить очередную макрокоманду, прервать выполнение макроса (например, если видно, что аргументы вычислены неправильно) и продолжить выполнение в обычном режиме, т. е. не по шагам.

Если в процессе выполнения макроса возникла ошибка, появляется диалоговое окно **Ошибка выполнения макроса** (Action Failed), аналогичное окну **Пошаговое исполнение макроса** (Macro Single Step), но без возможности продолжить процесс (кнопки **Шаг** (Step) и **Продолжить** (Continue) недоступны). Можно только прервать выполнение макроса, нажав кнопку **Прервать** (Halt). В окне отображается макрокоманда, содержащая ошибку.

Поиск ошибок в макросах, как правило, не вызывает больших затруднений. Это связано с тем, что:

- во-первых, большинство синтаксических ошибок проверяется при вводе и система просто не даст ввести неправильное название макрокоманды или условное выражение (правда, она не защитит от ввода неправильного имени объекта, например несуществующей формы или элемента управления в форме);
- во-вторых, макросы, как правило, достаточно просты и не содержат длинных последовательностей макрокоманд, в противном случае лучше написать процедуру на VBA, т. к. язык VBA предоставляет больше возможностей и имеет существенно лучшие средства отладки.

Основные ошибки, которые возникают в макросах, связаны либо с неправильным вводом имен объектов, и тогда Access не может найти соответствующий объект, либо с тем, что объекты, на которые ссылается макрокоманда, не доступны, например форма не открыта. Поэтому при возникновении ошибки в первую очередь нужно проверить, правильно ли введены имена объектов и открыты ли формы, использованные в условиях или аргументах макрокоманд.

Совет

Прежде чем использовать ссылки на форму или элемент управления, рекомендуется выполнять проверку, открыта ли форма, с помощью функции `isLoaded()`. К сожалению, эта функция не является стандартной функцией Access, а обычно включается в модуль `Служебные функции` (Utility Function) базы данных `Борей`. Вы можете скопировать ее оттуда в свою базу данных.

Стандартным приемом отладки как процедур, так и макросов, является вывод на экран специальных сообщений, говорящих о прохождении определенных этапов процесса. В макросах это осуществляется путем вставки дополнительных макрокоманд `ВыводСообщения` (MsgBox) для отображения таких сообщений. Причем в сообщении можно вывести значения свойств элементов управления, например:

```
"Цена товара = " & Forms!Товары!Цена
```

Можно легко отключить вывод на экран отладочных сообщений, для чего введите в столбец **Условия** (Condition) напротив соответствующих макрокоманд значение `Ложь`.

Еще один стандартный прием — установка точек останова. Его применяют для того, чтобы остановить выполнение макроса в определенной точке до завершения процесса. Для этого используется макрокоманда `ОстановитьМакрос` (StopMacro). Макрос будет выполняться до макрокоманды `ОстановитьМакрос` (StopMacro), после этого можно проверить результаты. Таким образом, можно локализовать ошибку с точностью до макрокоманды.

Применение макросов

Макросы могут применяться для решения самых разнообразных задач. В первом разделе данной главы мы привели таблицу макрокоманд, сгруппированных по категориям. Теперь рассмотрим применение макрокоманд некоторых категорий.

Работа с данными в формах и отчетах

В эту категорию входит набор макрокоманд, обеспечивающих отбор данных, перемещение по данным и их обновление в формах. Прежде чем перейти к примерам использования макросов, покажем, как применять ссылки на формы, отчеты и элементы управления в аргументах и условиях макрокоманд, поскольку такие ссылки используются очень часто.

Ссылки на формы, отчеты и их свойства

Чтобы задать ссылку на форму или отчет, нужно сначала определить, в какое семейство (collection) входит объект, на который создается ссылка. *(Подробнее о семействах и составляющих их объектах см. разд. "Объектные модели Microsoft Office Access 2003" гл. 13.)* Все открытые формы входят в семейство `Формы` (Forms), а открытые отчеты — в семейство `Отчеты` (Reports). Полная ссылка на форму или отчет должна состоять из двух частей: `имяСемейства!имяОбъекта`. Причем если имя объекта содержит пробелы или специальные символы, то его нужно заключить в квадратные скобки. Если пробелы в имени не используются, скобки можно не ставить. Таким образом, ссылка на форму будет выглядеть так:

`Forms![Заказы клиента]` или `Forms!Клиенты`

Для отчета ссылки выглядят аналогично:

`Reports![Отчет о продажах]` или `Reports!Прайс-лист`

Ссылка на свойство формы или отчета состоит из трех частей:

`имяСемейства!имяОбъекта.имяСвойства`

Например:

`Forms!Клиенты.Visible` или `Reports![Продажи за период].MenuBar`

Свойство **Вывод на экран** (Visible) определяет, будет ли форма видна на экране или спрятана, а свойство **Меню** (MenuBar) позволяет связать с отчетом или формой специальное меню.

Ссылки на элементы управления форм, отчетов и их свойства

Чтобы создать ссылку на элемент управления или его свойство, необходимо указать его имя. Если имя содержит пробелы, оно заключается в квадратные скобки.

Ссылка на элемент управления в форме или отчете состоит из трех частей:

```
имяСемейства!имяОбъекта!имяЭлемента
```

Например, ссылка на элемент управления `Номер заказа` в форме `Заказы клиента` выглядит так:

```
Forms![Заказы клиента]![Номер заказа]
```

Ссылка на элемент управления `Сумма` в отчете `Продажи за период` выглядит так:

```
Reports![Продажи за период]![Сумма]
```

Ссылка на свойство элемента управления состоит из четырех частей:

```
имяСемейства!имяОбъекта!имяЭлемента.имяСвойства
```

Например:

```
Forms![Заказы клиента]![Номер заказа].Enabled
```

Свойство **Доступ** (Enabled) позволяет запретить или разрешить доступ к элементу управления.

Объект может иметь свойство, используемое по умолчанию. Это свойство применяется в том случае, когда имя свойства в ссылке явно не указано. Например, у элементов управления по умолчанию используется свойство **Значение** (Value), поэтому ссылка `Forms!Товары!Цена` позволяет получить доступ к значению, отображенному
в текстовом поле `Цена`.

Замечание

Название семейства форм при написании ссылки мы используем английское — `Forms` (оно же используется в языке VBA). Построитель выражения создает ссылку с русским названием — `Формы`. И тот и другой вариант можно употреблять.

Ссылки на подчиненные формы и отчеты

На подчиненную форму или отчет можно ссылаться так же, как и на любой другой элемент управления, т. к. подчиненная форма и подчиненный отчет являются одним из типов элементов управления. Например:

```
Forms!Заказы!ПодформаТовары
```

Здесь `ПодформаТовары` — это имя элемента управления в форме `Заказы`, который представляет собой подчиненную форму.

А вот ссылка на элемент управления в подчиненной форме или отчете имеет особую структуру: после имени элемента управления, который является подчиненной формой, нужно сначала указать специальное свойство: `Form` — для форм или `Report` — для отчетов, а затем имя элемента управления, на который выполняется ссылка:

```
Forms!Заказы!ТоварыПодформа.Form![Код Товара]
```

Фильтрация записей в формах, отчетах, таблицах

Перейдем к примерам использования макрокоманд Access. Для отбора записей в формах, отчетах, таблицах применяется макрокоманда `ПрименитьФильтр` (ApplyFilter). Задать фильтр можно двумя способами: либо указав имя созданного фильтра заранее в аргументе **Имя фильтра** (Filter Name), либо непосредственно задав условие выборки в аргументе **Условие отбора** (Where Condition). Если фильтр нужно применить сразу при открытии формы, тогда с событием **Открытие** (On Open) формы необходимо связать макрос, содержащий макрокоманду `ПрименитьФильтр` (ApplyFilter). Если нужно менять набор отображаемых записей в открытой форме динамически, поступают следующим образом:

- создают в этой форме набор полей, в которых можно задать условия отбора;
- создают кнопку **Применить фильтр**, с которой связывают макрос, содержащий макрокоманду `ПрименитьФильтр` (ApplyFilter). В качестве значения аргумента **Условие отбора** (Where Condition) для этой макрокоманды указывают выражение, содержащее ссылки на эти поля.

Пример такого решения приведен на рис. 12.16. На нем представлена форма `Товары` (Products), позволяющая просматривать товары с выборкой по различным критериям.

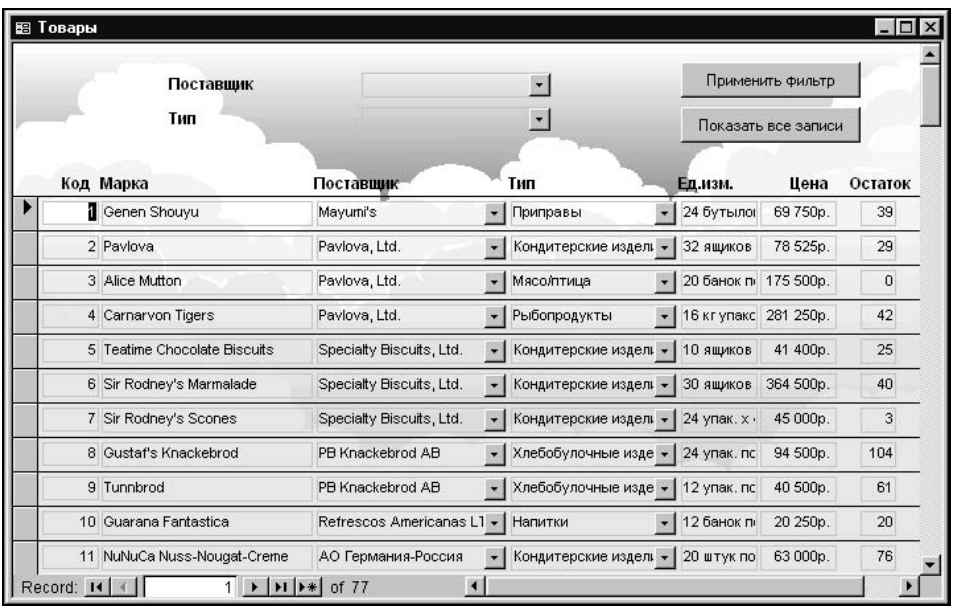


Рис. 12.16. Форма для отбора записей по заданным критериям

Эта форма построена на основе таблицы `Товары` (Products) с помощью Мастера автоматической генерации ленточных форм. Затем в режиме Конструктора форм к ней добавлены поля для определения критериев выбора и кнопки для применения и отмены фильтра. Соответствующие макросы, связанные с событием **Нажатие кнопки**

(On Click), приведены на рис. 12.17. Для отмены фильтра используется макрокоманда ПоказатьВсеЗаписи (ShowAllRecords).

В диалоговом окне **Область ввода** (Zoom) отображено значение аргумента **Условие отбора** (Where Condition) макрокоманды ПрименитьФильтр (ApplyFilter). Обратите внимание, условие получилось довольно сложным, а длина поля **Условие отбора** (Where Condition) ограничена 255 символами. Поэтому, если бы мы включили в условия отбора еще одно поле, например Поставки прекращены, нам пришлось бы отдельно создать специальный фильтр и задать его имя в аргументе **Имя фильтра** (Filter Name).

В макросе ОтменитьФильтр, кроме макрокоманды

ПоказатьВсеЗаписи(ShowAllRecords),

нужны еще две макрокоманды, чтобы очистить поля ВыборПоставщика и ВыборТипа, т. е. присвоить им значение Null. Это делается с помощью макрокоманды ЗадатьЗначение (SetValue).

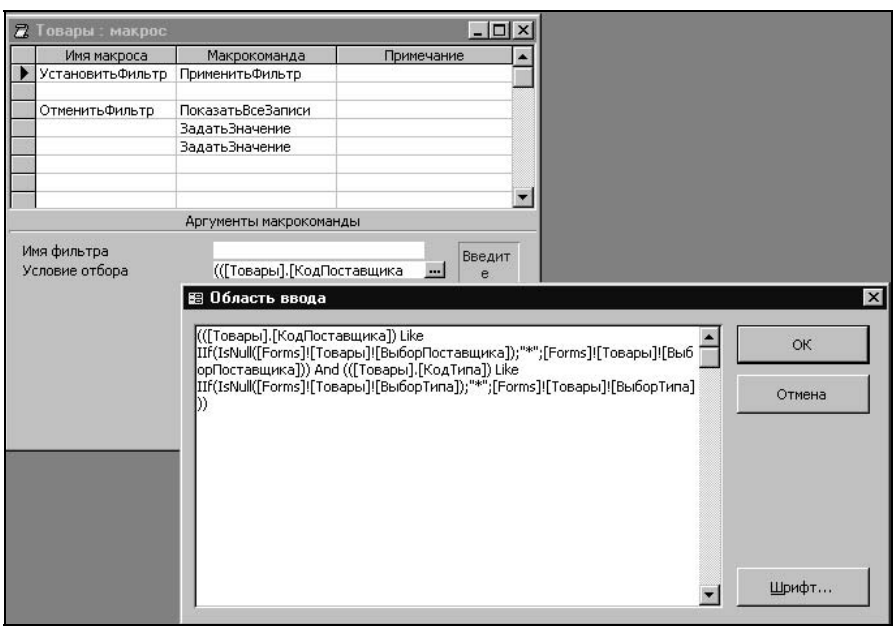


Рис. 12.17. Макросы для установки и отмены фильтра в форме Товары

Перемещение по данным

Эта группа макрокоманд связана с перемещением по записям и элементам управления. В качестве примера использования макрокоманд этой группы рассмотрим ситуацию, когда пользователь работает с формой Клиенты (Customers) и хочет найти заказ текущего клиента, номер которого ему известен. Давайте вернемся к примеру, приведенному ранее в *разд. "Назначение макроса событию" этой главы.* Тогда мы

создали в форме Клиенты (Customers) кнопку Заказы клиента для отображения заказов выбранного клиента. Попробуем усовершенствовать эти формы. Если номер заказа известен пользователю, он может ввести его в текстовое поле Поиск заказа, которое можно добавить в форму Клиенты (Customers). Тогда в открываемой форме Заказы (Orders) следует сразу показать запись, содержащую соответствующий счет. Добавим поле Поиск заказа в заголовок формы и сделаем так, чтобы при открытии формы Заказы (Orders) проверялось значение этого поля и, если оно не пустое, производился бы поиск счета с указанным номером и выполнялся переход на соответствующую запись. В противном случае текущей становилась бы первая запись из отфильтрованного набора записей. На рис. 12.18 показана форма Клиенты (Customers) с новым полем и макрос Поиск заказа, который связан с событием **Загрузка** (On Load) формы Заказы (Orders).

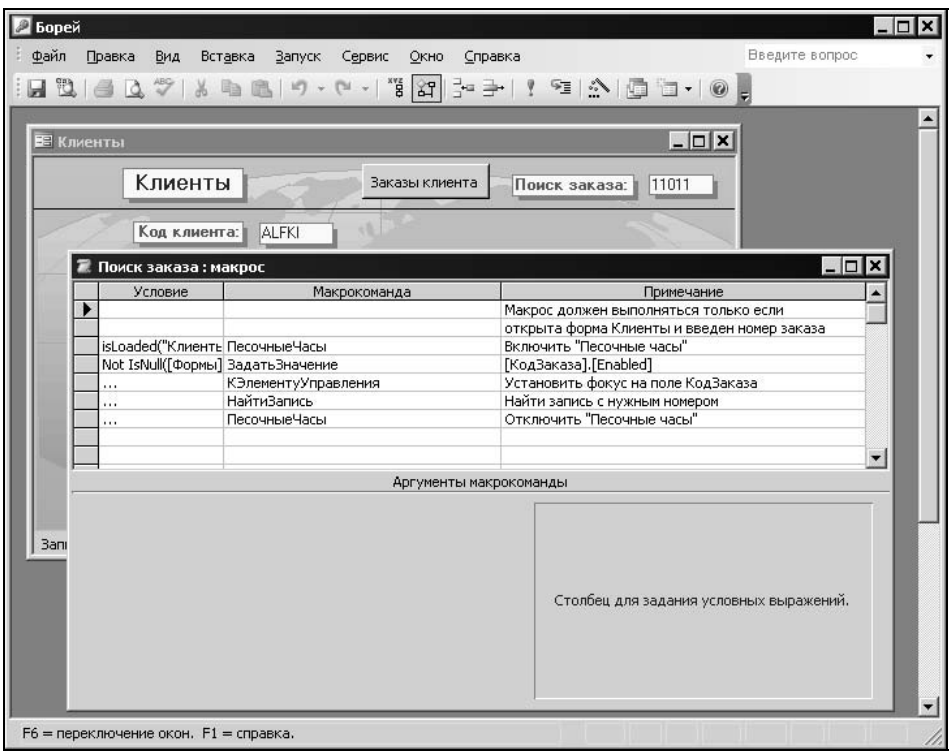


Рис. 12.18. Пример макроса для перемещения по данным

Поскольку добавленный макрос обработки события ссылается на элемент управления в форме Клиенты (Customers), он должен выполняться только в том случае, если эта форма открыта, и, кроме того, поиск должен производиться только тогда, когда поле Поиск заказа не пусто. Исходя из этого, формируются условия макроса. Прежде чем производить поиск записи по образцу в одном из полей, необходимо активизировать это поле, для чего используется макрокоманда КэлементуУправления (GoToControl). С ее помощью устанавливается фокус на поле КодЗаказа (OrderId).

Но это поле в форме Заказы (Orders) недоступно, поэтому сначала нужно изменить значение свойства **Доступ** (Enabled) данного поля. В противном случае выполнение макрокоманды КэлементуУправления (GoToControl) приведет к возникновению ошибки.

Макрокоманда КэлементуУправления (GoToControl) используется, как правило, если требуется изменить стандартный порядок перехода между полями в форме. Обычно переход между полями выполняется с помощью клавиши <Tab>, однако иногда требуется пропустить ряд полей и переместить фокус к определенному элементу управления. Это может зависеть от каких-либо условий, например от значения поля. В таком случае на событие **После обновления** (After Update) для данного поля назначается макрос, который перемещает фокус на нужный элемент управления в форме. В качестве аргумента для макрокоманды необходимо указать короткое имя элемента управления. В случае излишне длинного имени переход к элементу управления не произойдет и отобразится сообщение об ошибке.

Обновление данных в формах и элементах управления

Последняя группа макрокоманд в рассматриваемой категории связана с обновлением данных в активных формах, таблицах и запросах. Если несколько человек в сети одновременно изменяют данные, то формы и таблицы у конкретного пользователя могут не отражать актуальных данных. Для того чтобы отображаемые данные соответствовали текущему состоянию базы данных, их необходимо обновить с помощью команды **Записи | Обновить** (Record | Refresh) *(см. разд. "Работа с записями" гл. 2)*.

Похожая ситуация возникает даже в однопользовательском режиме, если в форме используется поле со списком, источником данных для которого является таблица или запрос. Если в исходную таблицу были добавлены записи, то в поле со списком они автоматически не появятся — нужно повторно выполнить запрос. Кроме полей со списком, к элементам управления, требующим обновления отображаемых данных, относятся также списки и элементы управления подчиненной формы, объекты OLE и вычисляемые элементы управления, содержащие статистические функции по подмножеству записей, такие как DLookUp() или DSum().

Для того чтобы выполнить обновление записей в формах, таблицах или элементах управления, используются макрокоманды Обновление (Requery), ПоказатьВсеЗаписи (ShowAllRecords) и ОбновитьОбъект (RepaintObject).

Макрокоманда Обновление (Requery) обновляет данные в объекте базы данных путем повторного просмотра источника данных. Макрокоманда имеет один аргумент, содержащий имя объекта, который следует обновить. Если обновляется активный объект, например форма, то поле аргумента следует оставить пустым. При этом макрокоманда будет повторно выполнять запрос, указанный в свойстве **Источник данных** (RecordSource) этой формы.

Рассмотрим пример использования макроса для обновления данных. В форме Клиенты (Customers) есть поле со списком Страна (Country). Источником данных для этого поля является запрос, который выбирает значения из поля Страна (Country) таблицы Клиенты (Customers):

```
SELECT DISTINCT Клиенты.Страна FROM Клиенты;
```

Если при вводе клиента в таблицу добавляется новое название страны, то в списке эта страна не появится, поскольку запрос будет выполнен повторно только при сле-

дующем открытии формы. Чтобы провести обновление списка стран раньше, следует назначить событию формы **После обновления** (After Update) макрос `Клиенты.Обновление списка стран`, который состоит из одной макрокоманды `Обновление` (Requery) со значением аргумента `Страна` (Country) (рис. 12.19).

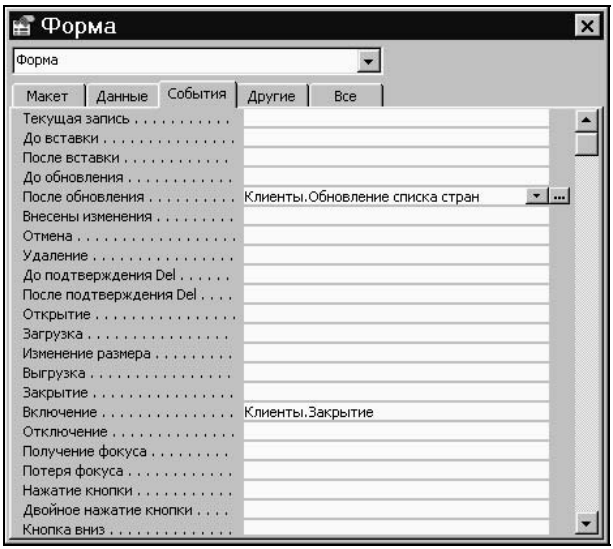


Рис. 12.19. Назначение макроса событию формы **После обновления**

Замечание

Макрокоманда `Обновление` (Requery) обновляет только один объект, поэтому если в форме существуют элементы управления, требующие обновления отображаемых данных, и в макросе, назначенном этой форме, используется макрокоманда `Обновление` (Requery) с пустым значением аргумента, то обновляться будут только записи в форме. Для каждого такого элемента управления нужно использовать отдельную макрокоманду. Если эта макрокоманда в качестве аргумента содержит имя элемента управления, то при ее выполнении обновляется только данный элемент. Записи в самой форме не обновляются.

С помощью макрокоманды `Обновление` (Requery) можно обновлять данные и в неактивной форме, точнее, не в той форме, которой назначен макрос. Однако в этом случае сначала необходимо выполнить макрокоманду `ВыделитьОбъект` (SelectObject), которая перенесет фокус на нужную форму, чтобы ее активизировать (потом фокус можно вернуть обратно).

Макрокоманда `ПоказатьВсеЗаписи` (ShowAllRecords), как уже отмечалось, отменяет действие фильтра и повторно просматривает источник записей. Ее часто используют для обновления данных в подчиненной форме.

Макрокоманда `ОбновитьОбъект` (RepaintObject) применяется только к объекту базы данных (к таблице, запросу, форме, отчету, странице, макросу и модулю) и не применяется к элементу управления. Она выполняет немедленное обновление указанно-

го открытого объекта (если имя объекта не задано, обновляется активный объект), хотя при этом не производится повторное выполнение запроса к источнику данных. Обновление объекта не влияет на отображение новых и удаленных записей, как это происходит при выполнении макрокоманды `Обновление` (Requery). Обычно макрокоманду `ОбновитьОбъект` (RepaintObject) применяют для отображения результатов изменения данных с помощью макрокоманд `ЗадатьЗначение` (SetValue), а также для повторного вычисления значений выражений в вычисляемых элементах управления.

Работа с объектами

Это наиболее обширная категория макрокоманд, часть из которых уже была описана в предыдущих разделах.

Рассмотрим сначала, как используется макрокоманда `ЗадатьЗначение` (SetValue), которая позволяет устанавливать значения свойств элементов управления в формах и отчетах. Мы уже использовали эту макрокоманду в предыдущих примерах.

Установка свойств элементов управления позволяет динамически, в зависимости от условий, делать эти элементы недоступными или невидимыми. Приведем несколько примеров.

- Можно создать одну форму, в которой в зависимости от определенных условий видимыми будут разные поля. Это позволяет использовать одну форму в нескольких случаях и не создавать для каждой ситуации дополнительную форму. Когда форм в приложении очень много, такая возможность становится очень полезной. Чтобы сделать элемент управления невидимым, следует задать значение **Ложь** (False) для свойства **Вывод на экран** (Visible) этого элемента. Чтобы показать элемент управления на экране, задайте для этого свойства значение **Истина** (True).
- В зависимости от текущего состояния работы с данными можно изменять доступность кнопок, предназначенных для выполнения определенных действий, или других элементов управления в форме. Чтобы сделать элемент управления недоступным, задайте значение **Ложь** (False) его свойству **Доступ** (Enabled). Чтобы элемент управления стал доступным, задайте значение **Истина** (True) для этого свойства.
- Можно запретить пользователю изменять данные в форме. Для этого свойству **Доступ** (Enabled) соответствующих полей следует присвоить значение **Ложь** (False), а свойству **Блокировка записей** (Locked) — значение **Истина** (True). Если необходимо запретить изменение во всех полях, установите значение **Ложь** (False) для следующих свойств формы: **Разрешить изменения** (Allow Edits), **Разрешить добавление** (Allow Additions), **Разрешить удаление** (Allow Delitions). Эти свойства можно изменять динамически, т. е. в процессе работы пользователя с формой. Например, можно разрешить или запретить редактирование данных после проверки прав пользователя.
- С помощью макрокоманды `ЗадатьЗначение` (SetValue) можно динамически изменять значение свойства **Источник записей** (RecordSource) для формы, что позволяет управлять содержанием отображаемых данных.

Существуют макрокоманды для открытия и закрытия объектов Access: `ОткрытьФорму` (OpenForm), `ОткрытьЗапрос` (OpenQuery), `ОткрытьОтчет` (OpenReport) и т. д. Для от-

крытия объекта каждого типа применяется отдельная макрокоманда, а для закрытия объекта используется общая для объектов всех типов макрокоманда `Закрыть` (Close). Тип объекта, к которому следует применить эту макрокоманду, указывается в качестве одного из ее аргументов. В предыдущих примерах уже использовались макрокоманды `ОткрытьФорму` (OpenForm) и `Закрыть` (Close) для активного объекта.

Макрокоманда `ОткрытьЗапрос` (OpenQuery) позволяет выполнить любой тип запроса, в том числе запрос на изменение данных. Если в качестве аргумента этой макрокоманды задается имя запроса на выборку или перекрестного запроса, то результатом выполнения макрокоманды будет вывод на экран выбранных записей. Если же аргумент — имя запроса на изменение данных, то макрокоманда выполнит запрос, изменяя соответствующим образом данные в таблицах.

При выполнении запроса, изменяющего данные, на экране будут отображаться предупреждающие сообщения. Чтобы отключить вывод этих сообщений, используйте макрокоманду `УстановитьСообщения` (SetWarnings) со значением аргумента **Нет** (No). Только не забудьте после выполнения запроса снова включить вывод системных сообщений, применив ту же макрокоманду, но с аргументом **Да** (Yes). Иначе в Access не будут отображаться никакие системные сообщения, что может привести к выполнению нежелательных действий в приложении.

Макрокоманда `ОткрытьПредставление` (OpenView) аналогична макрокоманде `ОткрытьЗапрос` (OpenQuery), только применяется она в проектах Access и предназначена для работы с данными, хранящимися на сервере.

Макрокоманда `ОткрытьСохраненнуюПроцедуру` (OpenStoreProcedure) позволяет выполнить или открыть в режиме редактирования хранимую процедуру сервера. *(См. гл. 17.)*

В данном разделе описаны далеко не все макрокоманды. Для получения полной информации используйте справочную систему. Информацию о макросах и макрокомандах вы найдете в разделе **Программная поддержка** (Programmability) оглавления **Справки Access** (Access Help).

Преобразование макросов в процедуры VBA

Если вы создали много макросов, а потом решили перейти к программированию на VBA, не беспокойтесь о том, что придется заново переписывать на VBA все созданные макросы. В Access существует возможность автоматического преобразования макросов в процедуры VBA. Это касается как макросов, связанных с событиями в формах и отчетах, которые преобразуются в процедуры обработки событий, так и общих макросов, не связанных с формами и отчетами. Все они преобразуются в отдельные стандартные модули.

Давайте попробуем преобразовать в процедуры обработки событий макросы, назначенные событиям формы `Наклейки для клиентов` (Customers Labels Dialog):

1. Откройте форму `Наклейки для клиентов` (Customers Labels Dialog) в режиме Конструктора.
2. Выберите команду **Сервис | Макрос | Преобразовать макросы формы** (Tools | Macro | Convert form's Macros to Visual Basic).

3. В диалоговом окне **Преобразование макросов формы** (Convert form macros) (рис. 12.20) нажмите кнопку **Преобразовать** (Convert), оставив флажки установленными по умолчанию.

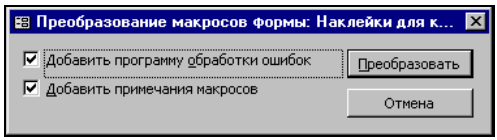


Рис. 12.20. Преобразование макроса в процедуру VBA

4. Когда преобразование закончится, в форме `Наклейки для клиентов` (Customers Labels Dialog) щелкните правой кнопкой мыши, например по кнопке **Отменить**, и выберите команду **Свойства** (Properties).
5. В диалоговом окне **Свойства** (Properties) раскройте вкладку **События** (Events).
6. В поле для события **Нажатие кнопки** (On Click) оказалось установленным значение **[Процедура обработки событий]** ([Event Procedure]). Щелкните по кнопке Построителя, находящейся справа от поля, — откроется окно редактора VBA с текстом процедуры (рис. 12.21).

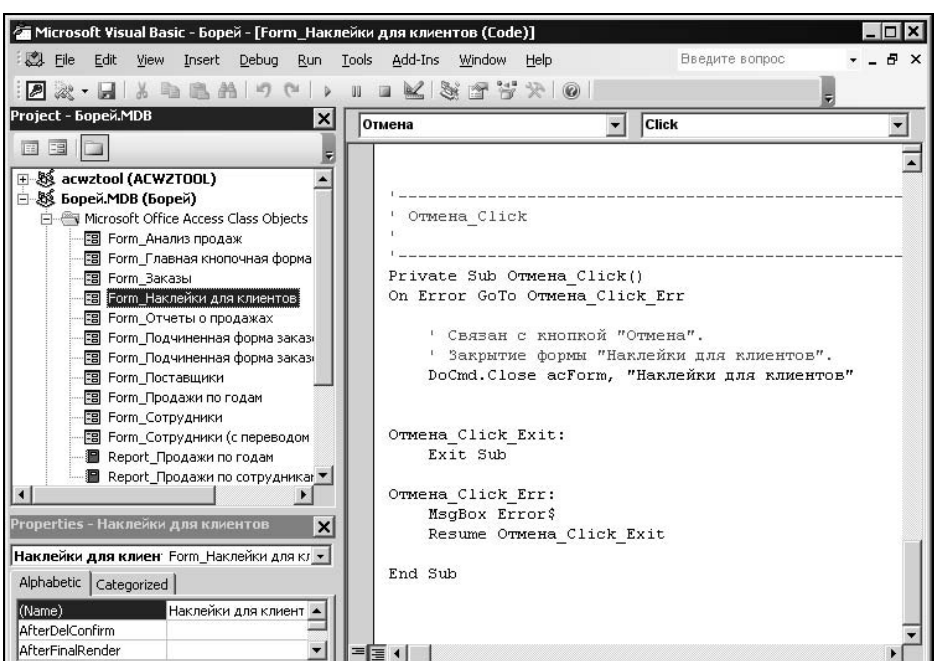


Рис. 12.21. Процедура, полученная из макроса

В этой процедуре, полученной из макроса, используется объект `DoCmd` для выполнения макрокоманды закрытия формы. В соответствии с флажками, установленными в

окне **Преобразование макроса** (Convert Macro), комментарии в макросе были преобразованы в комментарии в процедуре, и был добавлен код обработки ошибки, который выдает на экран сообщение об ошибке в случае ее возникновения *(см. разд. "Программирование в Access 2003" гл. 13)*.

Аналогично выполняется преобразование макросов, назначенных событиям отчета.

Теперь рассмотрим преобразование в процедуру VBA макроса, не связанного с событием формы или отчета, например созданного ранее макроса `Менеджеры по продажам`.

1. Щелкните по ярлыку **Макросы** (Macros) в окне базы данных `Борей`. Появится список макросов. Выделите макрос `Менеджеры по продажам`.
2. Выберите команду **Сервис | Макрос | Преобразовать макросы** (Tools | Macro | Convert Macros to Visual Basic).
3. Откроется такое же диалоговое окно, как в предыдущем примере (см. рис. 12.20), в котором необходимо нажать кнопку **Преобразовать** (Convert). По окончании процесса преобразования появится сообщение об успешном завершении процесса преобразования и откроется окно редактора VBA, в котором можно посмотреть полученный код на VBA (рис. 12.22).
4. Макрос преобразуется в функцию с аналогичным именем `Менеджеры_по_продажам` (пробелы заменились символами подчеркивания).

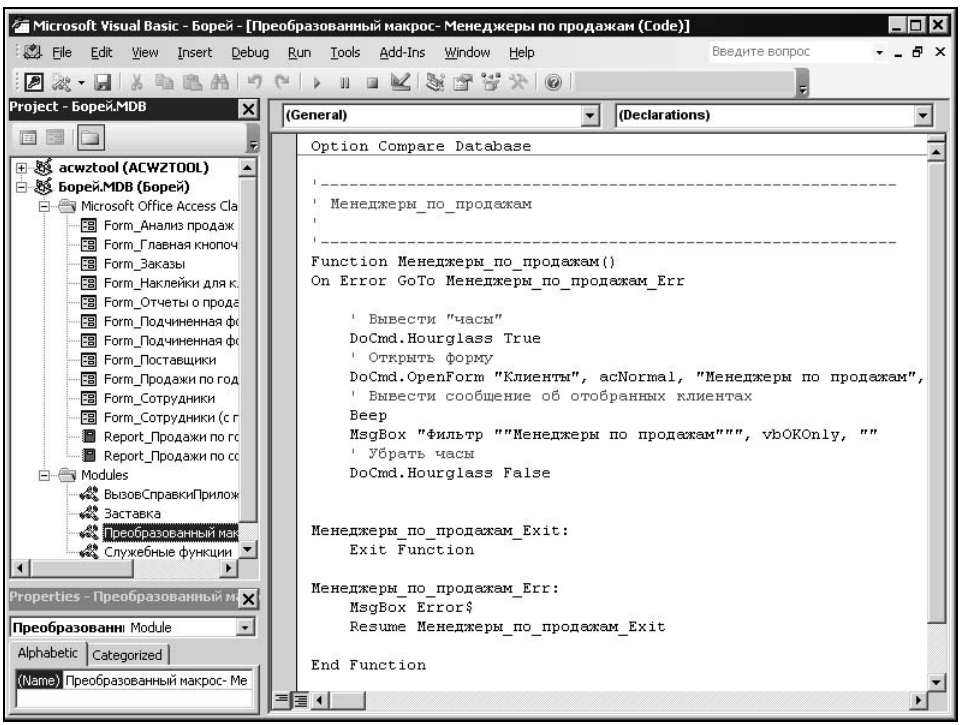


Рис. 12.22. Результат преобразования макроса `Менеджеры по продажам`

После преобразования исходный макрос Менеджеры по продажам не был удален, а в списке модулей в окне базы данных Борей появился новый модуль с именем Преобразованный макрос — Менеджеры по продажам (Converted Macro — Менеджеры по продажам).

Существует еще один способ преобразования макросов в модуль VBA:

1. Выделите макрос Менеджеры по продажам в окне базы данных Борей.
2. Выберите команду **Файл | Сохранить как** (File | Save As).
3. В появившемся диалоговом окне **Сохранение** (Save As) в раскрывающемся списке **Как** (As) выберите элемент **Модуль** (Module) (рис. 12.23) и нажмите кнопку **ОК**.

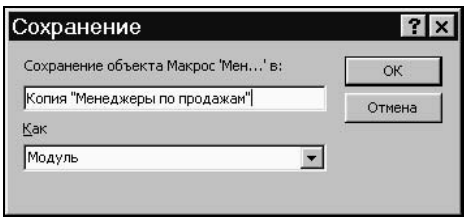


Рис. 12.23. Сохранение макроса в виде модуля

Уровни безопасности макросов

В Access 2003, так же как и во всех остальных приложениях Microsoft Office 2003, повышенное внимание уделяется защите от макровирусов, т. е. вирусов, которые могут храниться в коде процедур VBA, макросах, элементах ActiveX и т. д. Поскольку объекты Access легко импортируются из разных источников, существует вероятность заражения компьютера при таком импорте. Поэтому во всех приложениях Microsoft Office 2003 можно регулировать степень контроля потенциально опасных участков приложения с помощью *уровней безопасности*. Чтобы посмотреть или изменить установленный уровень безопасности, нужно выполнить команду меню **Сервис | Макрос | Безопасность** (Tools | Macro | Security). Появляется диалоговое окно с двумя вкладками (рис. 12.24). На вкладке **Уровень безопасности** (Security) устанавливается требуемый уровень безопасности, а на вкладке **Надежные источники** (Trusted Sources) поддерживается список надежных издателей.

По умолчанию устанавливается *средний уровень безопасности*, при котором решение о запуске потенциально опасных макросов принимает пользователь. Если база данных содержит неподписанные макросы, то при ее открытии пользователю предлагается открыть базу данных или отменить действие. Если вы принимаете решение об отмене, база данных открываться не будет. Если макросы подписаны, то проверяется цифровая подпись и источник. Если источник не входит в перечень надежных источников, или подпись не достоверна, пользователь должен будет принять решение о возможности работы с этим файлом.

Рекомендуется устанавливать либо средний, либо высокий уровень безопасности.

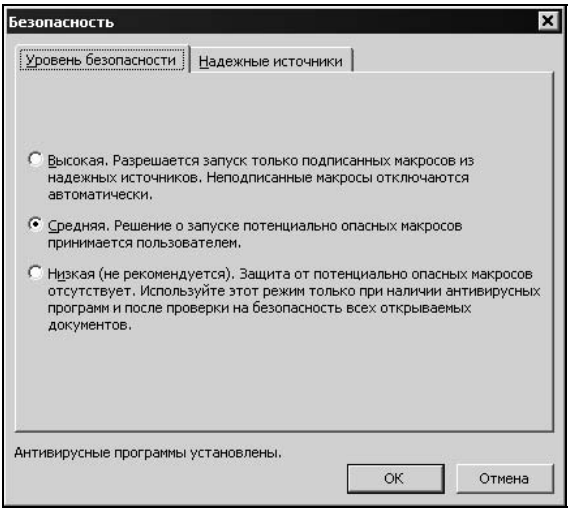


Рис. 12.24. Диалоговое окно **Уровни безопасности**

При *высоком уровне безопасности* база данных, в которой находятся неподписанные макросы, не открывается. Если макросы подписаны, то выполняется проверка источника и подписи, и файл открывается, только если источник надежен и подпись достоверна. Если подпись достоверна, но автор неизвестен, то пользователь может принять решение об открытии файла.

Низкий уровень безопасности можно использовать только в том случае, если вы не пользуетесь компонентами из других источников и грамотно используете качественные антивирусные средства, т. к. при таком уровне безопасности пользователю не предлагается оценивать надежность автора или сертификата макроса — макросы включаются автоматически.

Существует еще *очень высокий уровень безопасности*. Он недоступен в окне **Уровни безопасности** (Security), но может быть установлен с помощью системных политик. Тогда никакие файлы баз данных и проектов Access открыть невозможно.

Цифровая подпись служит гарантией того, что компонент, в том числе макрос или программа VBA, получен от определенного автора и не подвергся изменениям после того, как был подписан. Подписать макрос можно в редакторе VBA. *(См. гл. 13.)*

Список надежных источников на второй вкладке окна **Уровни безопасности** (Security) пополняется, когда пользователь принимает решение об открытии файла, содержащего подписанный компонент, но из неизвестного источника. Он поддерживается, если установлен средний или высокий уровни безопасности. В целях безопасности рекомендуется снять флажок **Доверять всем установленным настройкам и шаблонам** (Trust all add-ins and templates) на этой вкладке.

Если на компьютере установлены антивирусные программы, совместимые с Microsoft Office, то независимо от установленного уровня безопасности выполняется проверка открываемых файлов, если они содержат макросы, программный код или другие компоненты, с целью поиска известных вирусов.

ГЛАВА 13



Программирование на языке VBA

Данная глава посвящена основному средству программирования в Access 2003 — языку Visual Basic for Applications (VBA, Visual Basic для приложений). Этот язык является общим инструментом для всех приложений Microsoft Office, позволяющим решать любые задачи программирования, начиная от автоматизации действий конкретного пользователя и кончая разработкой полномасштабных приложений, использующих Microsoft Office в качестве среды разработки.

Цель данной главы — дать читателю основные сведения как о самом языке, так и о тех инструментальных средствах, которые предоставляет разработчику приложений среда Access.

Поскольку Visual Basic для приложений является объектно-ориентированным языком программирования, будут описаны объектные модели, которые могут использоваться в Access 2003. Это не только собственная объектная модель Access, но и объектные модели Visual Basic для приложений и Microsoft Office, которые являются общими для всех приложений семейства. Основная работа в приложениях Access — это работа с данными, поэтому будут рассмотрены также библиотеки управления данными: DAO (Data Access Objects), ADO (ActiveX Data Objects).

Модель программирования в Access является событийно-управляемой, т. е. в процессе работы приложения возникают события, которые запускают специальные программы — обработчики событий. Большое количество разнообразных событий определены в таких объектах Access, как формы, отчеты и элементы управления в них. Мы подробно описываем эти события и последовательности их возникновения, т. к. искусство программирования в формах и отчетах включает в себя правильное определение того события, которое требуется обработать специальной программой. *(О событиях страницы доступа к данным см. гл. 11.)* Итак, в данной главе будут рассмотрены следующие вопросы:

- синтаксис языка Visual Basic for Applications;
- интегрированная среда VBA, которая включает в себя как редактор кода программных модулей, так и большое количество средств отладки этого кода;
- объектные модели, применяемые в приложениях Access;

- события форм, отчетов и элементов управления и последовательности их возникновения;
- примеры использования процедур обработки событий в формах и отчетах.

Язык Visual Basic для приложений (VBA)

Приведем синтаксис языка программирования VBA. Это описание, хотя и не строгое и не исчерпывающее, даст вам представление о возможностях языка, а примеры, которые будут описаны далее в этой же главе, помогут вам применить полученные сведения на практике при совершенствовании ваших приложений.

Процедуры и функции, их вызов и передача параметров

Основными компонентами программы на VBA являются *процедуры* и *функции*. Они представляют собой фрагменты программного кода, заключенные между операторами `Sub` и `End Sub` или между операторами `Function` и `End Function`. Вот как может выглядеть процедура VBA:

```
Sub <имяПроцедуры> (<аргумент1>, <аргумент2>, ...)
    <операторVisualBasic1>
    <операторVisualBasic2>
    ...
End Sub
```

Функция отличается от процедуры тем, что ее имя выступает также в качестве переменной и используется для возвращения значения в точку вызова функции. Вот как может выглядеть функция:

```
Function <имяФункции> (<аргумент1>, <аргумент2>, ...)
    <операторVisualBasic1>
    <операторVisualBasic2>
    ...
    <имяФункции> = <возвращаемоеЗначение>
End Function
```

Для того чтобы использовать написанную вами процедуру или функцию, необходимо *вызвать* ее. Как же осуществляется вызов процедур и функций? Процедуру с непустым списком аргументов можно вызвать только из другой процедуры или функции, использовав ее имя со списком фактических значений аргументов в качестве одного из операторов VBA. Функцию же можно вызвать не только с помощью отдельного оператора VBA, но и поместив ее имя со списком фактических значений аргументов прямо в формулу или выражение в программе на VBA или, например, прямо в формулу в вычисляемых полях запросов, форм и отчетов Access. Процедура с пустым списком аргументов (так называемый командный макрос) может быть вызвана не только из другой процедуры или функции, но и с помощью комбинации клавиш быстрого вызова, команд раскрывающихся меню или кнопок панелей инструментов. Можно также связать такую процедуру с выполнением самых различных событий: например, с открытием формы или отчета, со щелчком мышью по кнопке

в форме, с воздействием на элементы управления форм, в частности элементы управления ActiveX. Такие процедуры называют *процедурами обработки событий*. Понятно, что функции или процедуры, нуждающиеся в передаче им аргументов, таким способом вызвать нельзя. Подробнее о том, как создаются процедуры обработки событий, будет рассказано в *разд. "Программирование в формах и отчетах" данной главы.*

Если вызываемая процедура имеет уникальное имя и находится в том же модуле, что и вызывающая процедура, то для ее вызова достаточно указать это имя и задать список фактических значений аргументов, не заключая его в скобки. Второй способ вызова процедуры состоит в использовании оператора `Call`. Сначала идет оператор `Call`, затем имя процедуры и список параметров, в этом случае обязательно заключенный в скобки. Функцию можно вызывать точно так же, как и процедуру, но гораздо чаще применяется другой, специфический способ вызова функций: использование ее имени с заключенным в скобки списком параметров в правой части оператора присваивания.

Вот примеры вызова процедуры под именем `CrossRC` с передачей ей двух аргументов (константы и выражения):

```
CrossRC 7, i + 2
```

или

```
Call CrossRC(7, i + 2)
```

А вот пример вызова двух функций — `Left` и `Mid`, и использования возвращаемого ими значения в выражении:

```
yStr = Left(y, 1) & Mid(y, 2, 1)
```

Допускается два различных способа передачи переменных процедуре или функции: *по ссылке* и *по значению*. Если переменная передается по ссылке, то это означает, что процедуре или функции будет передан адрес этой переменной в памяти. При этом происходит отождествление формального аргумента процедуры и переданного ей фактического параметра. Тем самым вызываемая процедура может изменить значение фактического параметра: если будет изменен формальный аргумент процедуры, то это скажется на значении переданного ей при вызове фактического параметра. Если же фактический параметр передается по значению, то формальный аргумент вызываемой процедуры или функции получает только значение фактического параметра, но не саму переменную, используемую в качестве этого параметра. Тем самым все изменения значения формального аргумента не сказываются на значении переменной, являющейся фактическим параметром.

Способ передачи параметров процедуре или функции указывается при описании ее аргументов: имени аргумента может предшествовать явный описатель способа передачи. Описатель `ByRef` задает передачу по ссылке, а `ByVal` — по значению. Если же явное указание способа передачи параметра отсутствует, то по умолчанию подразумевается передача по ссылке.

Поясним сказанное на примере. Пусть имеются следующие описания двух процедур:

```
Sub Main()
a = 10
b = 20
c = 30
```

```
Call Example1(a, b, c)
Call MsgBox(a)
Call MsgBox(b)
Call MsgBox(c)
End Sub

Sub Example1(x, ByVal y, ByRef z)
    x = x + 1
    y = y + 1
    z = z + 1
    Call MsgBox(x)
    Call MsgBox(y)
    Call MsgBox(z)
End Sub
```

Вспомогательная процедура `Example1` использует в качестве формальных аргументов три переменные, описанные по-разному. Далее в теле этой процедуры каждый из них увеличивается на единицу, а затем их значения выводятся на экран с помощью функции `MsgBox`. Основная процедура `Main` устанавливает значения переменных `a`, `b` и `c`, а затем передает их в качестве (фактических) аргументов процедуре `Example1`. При этом первый аргумент передается по ссылке (действует умолчание), второй — по значению, а третий — снова по ссылке. После возврата из процедуры `Example1` основная процедура также выводит на экран значения трех переменных, передававшихся в качестве аргументов. Всего на экран выводится шесть значений:

- сначала это числа 11, 21 и 31 (все полученные значения увеличены на 1 и выводятся процедурой `Example1`);
- затем это числа 11, 20 и 31 (эти значения выводятся процедурой `Main`, причем переменные, переданные по ссылке, увеличились, а переменная, переданная по значению — нет).

Программа может состоять (и обычно состоит) из многих процедур и функций, которые могут располагаться в одном или нескольких *модулях*. Модули группируются в *проекты*, при этом в одном проекте могут мирно сосуществовать несколько различных программ, использующих общие модули или процедуры.

Каждая из процедур, находящихся в одном модуле, должна иметь уникальное имя, однако в проекте может содержаться несколько различных модулей. Обычно рекомендуется использовать только уникальные имена процедур в одном проекте, но допустимы и исключения. В том случае, если в проекте содержится несколько различных процедур с одним и тем же именем, следует для уточнения имени использовать при вызове процедуры следующий синтаксис:

```
<имяМодуля>.<имяПроцедуры>
```

Если при этом имя модуля состоит из нескольких слов, следует заключить это имя в квадратные скобки. Например, если модуль называется `Графические процедуры`, а процедура — `Крестик`, вызов может выглядеть следующим образом:

```
[Графические процедуры].Крестик
```

Допускается также использование процедур, расположенных и в других проектах. При этом может потребоваться еще один уровень уточнения имени:

```
<имяПроекта>.<имяМодуля>.<имяПроцедуры>
```

Переменные, константы и типы данных

Как и в других языках программирования, в VBA для хранения временных значений, передачи параметров и проведения вычислений используются переменные. Кратко остановимся на основных особенностях описания и использования переменных в VBA.

Обычно перед тем, как использовать переменную, производится ее объявление, т. е. вы заранее сообщаете Visual Basic, какие именно имена переменных вы будете использовать в своей программе, при этом объявляется также тип данных, для хранения которых предназначена эта переменная. В VBA, как и в обычном языке Basic, для этого используется оператор `Dim`. Вот синтаксис этого оператора:

```
Dim <имяПеременной> [As <типДанных>]
```

В VBA действуют следующие правила именования переменных. Имя не может быть длиннее 255 символов, оно должно начинаться с буквы, за которой могут следовать буквы, цифры или символ подчеркивания. Оно не должно содержать пробелов, знаков препинания или специальных символов, за исключением самого последнего знака. В конце к имени переменной может быть добавлен еще один из следующих шести специальных символов — описателей типа данных:

```
! # $ % & @
```

Эти символы не являются частью имени переменной: если в программе используются одновременно имена `string1$` и `string1`, то они ссылаются на одну и ту же строковую переменную. Нельзя использовать одно и то же имя переменной с разными символами определения типа данных или одновременно явное описание типа данных и не соответствующий этому типу данных специальный символ. Мы еще остановимся на этом подробнее при обсуждении типов данных.

Кроме того, не допускается использование в качестве имен переменных ключевых слов VBA и имен стандартных объектов. Именно поэтому рекомендуется начинать имена переменных со строчной, а не с прописной буквы. Поскольку у ключевых слов VBA и имен стандартных объектов первая буква при вводе автоматически преобразуется в прописную, вы будете избавлены от риска нечаянно использовать запрещенное имя переменной.

Допускается использование в именах переменных букв не только латинского алфавита, но и кириллицы, что может оказаться удобным для русских пользователей: при желании можно давать переменным имена на русском языке.

Во многих языках программирования, например в Pascal, переменные должны быть объявлены обязательно, и эти объявления используются компилятором, чтобы зарезервировать память для переменных. В то же время в VBA объявление переменных не является обязательным. Как и в его предшественнике, обычном языке Basic, допускается использование не описанных переменных. Выделение памяти переменным может выполняться динамически, а тип данных, хранящихся в переменной, может определяться по последнему символу имени переменной.

В свое время велись длительные баталии между сторонниками сокращения записи процедур за счет отказа от объявлений переменных и сторонниками обязательного описания всех переменных. Аргументы обеих сторон были достаточно серьезны — действительно, зачастую обязательное описание всех переменных требует изрядной доли занудства при методичном выписывании объявлений типа:

```
Dim i As Integer, j As Integer
Dim x As Double
```

С другой стороны, нелегко что-нибудь возразить и против того, что одним из самых опасных источников трудно обнаруживаемых ошибок в языках программирования, допускающих применение неописанных переменных, служат опечатки в написании имен переменных. Такие опечатки истолковываются транслятором как появление еще одной, новой переменной, отличной от ранее использовавшейся, и не воспринимаются как ошибки. Порой для обнаружения такой опечатки требуется время, во много раз превосходящее то, которое потребовалось бы на явное описание всех используемых в программе переменных.

В VBA принято поистине соломоново решение — предоставить разрешение этой дилеммы самому программисту. В этом языке имеется оператор

```
Option Explicit
```

Если вы начнете свой модуль с данного оператора (он должен быть расположен в самом начале модуля, до того, как начнется первая процедура этого модуля), то VBA будет требовать обязательного объявления переменных в этом модуле и генерировать сообщения об ошибке всякий раз, как встретит необъявленную переменную. Кроме того, если вы строгий пуританин и хотели бы, чтобы это требование стало обязательным для всех ваших модулей без исключения, вы можете установить параметр **Require Variable Declaration** (Явное описание переменных) на вкладке **Editor** (Редактор) диалогового окна **Options** (Параметры) редактора VBA. *(Подробнее об установке параметров редактора VBA будет рассказано в разд. "Среда программирования VBA" данной главы.)*

Установка этого параметра приведет к тому, что редактор Visual Basic будет автоматически добавлять оператор `Option Explicit` в начало каждого вновь создаваемого модуля. Учтите, однако, что данный флажок *не влияет* на все ранее созданные модули — если вы хотите добавить подобный оператор к уже существующим модулям, вам придется проделать это вручную.

Приведем теперь краткую сводку используемых типов данных VBA (табл. 13.1).

Таблица 13.1. *Типы данных VBA*

Тип данных	Описание
`Array`	Массив переменных, для ссылки на конкретный элемент массива используется индекс Требуемая память: зависит от размеров массива
`Boolean`	Принимает одно из двух логических значений: `True` или `False` Требуемая память: 2 байта

Таблица 13.1 *(продолжение)*

Тип данных	Описание
`Byte`	Число без знака от 0 до 255 Требуемая память: 1 байт
`Currency`	Используется для произведения денежных вычислений с фиксированным количеством знаков после десятичной запятой, в тех случаях, когда важно избежать возможных ошибок округления Диапазон возможных значений: от –922 337 203 685 477,5808 до 922 337 203 685 477,5807 Требуемая память: 8 байтов Символ определения типа по умолчанию: @
`Date`	Используется для хранения дат Диапазон возможных значений: от 1 января 0100 г. до 31 декабря 9999 г. Требуемая память: 8 байтов
`Double`	Числовые значения с плавающей точкой двойной точности Диапазон возможных значений для отрицательных чисел: от –1,79769313486232E308 до –4,94065645841247E–324 Диапазон возможных значений для положительных чисел: от 4,94065645841247E–324 до 1,79769313486232E308 Требуемая память: 8 байтов Символ определения типа по умолчанию: #
`Integer`	Короткие целые числовые значения Диапазон возможных значений: от –32 768 до 32 767 Требуемая память: 2 байта Символ определения типа по умолчанию: %
`Long`	Длинные целые числовые значения Диапазон возможных значений: от –2 147 483 648 до 2 147 483 647 Требуемая память: 4 байта Символ определения типа по умолчанию: &
`Object`	Используется только для хранения ссылок на объекты Требуемая память: 4 байта
`Single`	Числовые значения с плавающей точкой обычной точности Диапазон возможных значений для отрицательных чисел: от –3,402823E38 до –1,401298E–45 Диапазон возможных значений для положительных чисел: от 1,401298E–45 до 3,402823E38 Требуемая память: 4 байта Символ определения типа по умолчанию: !

Таблица 13.1 *(окончание)*

Тип данных	Описание
`String`	Используется для хранения строковых значений Длина строки: от 0 до 64 Кбайт Требуемая память: 1 байт на символ Символ определения типа по умолчанию: $
`Variant`	Может использоваться для хранения различных типов данных: даты/времени, чисел с плавающей точкой, целых чисел, строк, объектов Требуемая память: 16 байтов плюс 1 байт на каждый символ строковых значений Символ определения типа по умолчанию: отсутствует
Определяемый пользователем тип	Назначение и размер выделяемой памяти зависят от определения. Используется для описания структур данных. Позволяет хранить в одной переменной такого типа множество различных значений разного типа

При описании переменной указание типа данных может быть опущено. Тип переменной в таком случае определяется последним символом имени переменной: @, #, %, &, ! или $ (`Currency`, `Double`, `Integer`, `Long`, `Single` или `String`, соответственно). Например, поскольку символ `$` является символом определения типа для строковых данных, то переменная под именем `text$` автоматически становится переменной типа "строка символов". В дальнейшем этот специальный символ указания типа данных может быть опущен, однако постоянное присутствие в имени переменной символа определения типа будет напоминать о том, к какому типу данных относится эта переменная, что поможет избежать ошибок использования несовместных типов данных.

Если же последний символ не является ни одним из перечисленных ранее и явное указание типа тоже не используется, в этом случае переменной будет назначен по умолчанию тип данных `Variant`, который позволяет хранить в ней данные любого типа.

Учтите также, что нельзя использовать в одной и той же процедуре имена переменных, отличающиеся друг от друга только специальным символом определения типа в конце переменной. Например, не допускается одновременное использование переменных `var$` и `var%`. Не допускается и явное объявление переменной, уже содержащей символ определения типа в конце имени, с помощью описателя `As <типПеременной>` (даже если такое определение не противоречит обычному применению символа определения типа). Так, например, вы получите сообщение об ошибке, попытавшись ввести любое из следующих определений:

```
Dim var1% As String
Dim var2% As Integer
```

Для определения типа данных аргументов процедуры или функции используется описание типа данных непосредственно в заглавной строке процедуры или функции.

Например, следующая заглавная строка процедуры описывает ее параметры как переменные строкового типа:

```
Sub SplitStr(str1 As String, str2 As String, str3 As String)
```

Определение типа данных возвращаемого функцией значения завершает заглавную строку функции, например:

```
Function FindSplitSpace(str1 As String) As Integer
```

описывает возвращаемое функцией значение как переменную короткого целого типа.

Чтобы программа работала быстрее и занимала меньше памяти, рекомендуется использовать, когда это возможно, конкретные типы переменных, а не универсальный тип `Variant`. На обработку переменных типа `Variant` требуется не только дополнительная память (сравните размеры, приведенные в табл. 13.1), но и дополнительное время: требуется выяснить, к какому конкретному типу данных принадлежит такая переменная в момент обработки, а также при необходимости выполнить преобразование данных к нужному типу. Может показаться, что в таком случае лучше вообще не использовать подобные переменные. Это не так. Часто переменные типа `Variant` просто необходимы: например, в том случае, когда вы точно не уверены, какие именно данные будут присвоены переменной.

Приведем здесь лишь один пример. Если вы хотите, чтобы пользователь программы ввел некоторое целое значение, необходимое для дальнейшей работы программы, и описали для этого переменную целого типа, то рекомендуется использовать для ввода данных промежуточную переменную, описанную как `Variant`, а затем выполнить присваивание введенного значения нужной переменной (или же описать переменную, в которую вы хотите поместить это значение, как `Variant`, несмотря на то, что ввести предполагается именно целое число). Дело в том, что пользователь может ошибиться и ввести, например, вещественное число или вообще нечаянно нажать не на ту клавишу и ввести строковое значение. В таком случае при присваивании введенного значения переменной целого типа произойдет ошибка во время выполнения программы. Если же переменная для ввода будет объявлена как `Variant`, ничего страшного не случится: введенное значение будет благополучно присвоено этой переменной, а вы получите возможность проанализировать введенное значение и попросить пользователя повторить ввод, если введенное им значение не отвечает нужным условиям.

И в завершение этого раздела обсудим использование именованных констант. Для их описания используется оператор `Const`, схожий с оператором описания переменных `Dim`. Синтаксис этого оператора следующий:

```
Const <имяКонстанты> [As <типДанных>] = <выражение>
```

где `<выражение>` — это любое значение или формула, возвращающая значение, которое должно использоваться в качестве константы. Например, следующий оператор определяет целую константу `maxLen`:

```
Const maxLen% = 30
```

Как и переменные, константы могут содержать значения различных типов данных, но при этом они не меняют своих значений во время выполнения программы.

Совет

Если вы собираетесь использовать в вашей программе какие-либо константы, то рекомендуется дать этим константам осмысленные имена и описать их в самом начале модуля, а затем использовать всюду только именованные константы. Это не только делает программу понятнее, но и проще в сопровождении и отладке. Зачастую значение той или иной константы требуется изменить (хотя бы на время отладки), и тогда достаточно поменять лишь одно значение в описании именованной константы. Если же непосредственное значение использовалось прямо в тексте кода программы, то изменить все вхождения этого значения намного сложнее.

Кроме описываемых пользователем констант, существуют еще предопределенные встроенные константы, которые включаются в тексты программ без предварительного описания. Сведения о предопределенных встроенных константах, используемых для различных объектов приложений Microsoft Office и Visual Basic, можно найти в справке — в разделах описания свойств объектов (реже — в разделах описания методов). При именовании встроенных констант используется стандартное соглашение, позволяющее определить, к объектам какого приложения относится эта константа. Например, встроенные константы, относящиеся к объектам Access, начинаются с префикса `ac`, относящиеся к объектам Excel — с префикса `xl`, относящиеся к объектам Word — с префикса `wd`, относящиеся к объектам VBA — с префикса `vb`.

Например, в команде

```
DoCmd.OpenForm "Orders", acNormal,, stLinkCriteria
```

используется встроенная константа Access `acNormal`.

Ссылки на объекты

Кроме обычных переменных, в Visual Basic часто используются переменные, представляющие собой ссылку на объект. Оказывается, использование переменных для ссылок на объекты позволяет не только сократить и упростить текст программы, но и существенно ускорить ее работу.

Использование переменной-объекта немного отличается от применения обычных переменных: нужно не только объявить такую переменную, но и перед ее использованием назначить ей соответствующий объект с помощью специального оператора `Set`. Синтаксис этого объявления и назначения следующий:

```
Dim <имяПеременной> As Object
Set <имяПеременной> = <ссылкаНаОбъект>
```

Иногда при объявлении такой переменной удобно заранее указать конкретный тип объекта — можно использовать любой конкретный объект из объектной модели Office. Приведем пример упомянутого объявления и назначения:

```
Dim MyBase As Database
Set MyBase = CurrentDb()
```

После подобного объявления и назначения вы можете использовать переменную `MyBase` для обращения к текущей открытой базе данных. Такая ссылка короче и быстрее обрабатывается. Быстрее работает программа, использующая переменные для прямых ссылок на объекты вместо сложных иерархических ссылок, применяющих большое количество операторов уточнения (точек). Время, которое уходит

у программы на разрешение ссылок, пропорционально количеству операторов уточнения, которые приходится ей обрабатывать.

Совет

Если вы всерьез озабочены быстродействием вашей программы, то рекомендуется при описании переменных типа "объект" использовать конкретные объекты модели Office, а не универсальное описание `Object`. В оправдание можно привести примерно те же соображения, что и по поводу применения универсального типа `Variant` при описании обычных переменных: обработка такого типа переменных требует дополнительного времени и памяти.

Объектная переменная будет указывать на объект до тех пор, пока мы другим оператором `Set` не присвоим ей ссылку на другой объект этого же типа или не присвоим ей значение `Nothing`, означающее, что переменная не содержит никакой ссылки. Например:

```
Set txt = Nothing
```

После такого действия переменная продолжает существовать, хотя и не ссылается ни на какой объект. Другим оператором `Set` ей можно снова присвоить ссылку на объект.

Замечание

Обратите внимание, что объектные переменные, в отличие от обычных переменных, содержащих значения, содержат только ссылки на объекты, а не сами объекты или их копии.

Массивы

Массив — это переменная, в которой хранится одновременно несколько значений одинакового типа. Формальное определение массива таково: он представляет собой совокупность однотипных индексированных переменных.

Количество используемых индексов массива также может быть различным. Чаще всего используются массивы с одним или двумя индексами, реже — с тремя, еще большее количество индексов встречается крайне редко. В VBA допускается использовать до 60 индексов. О количестве индексов массива обычно говорят как о размерности массива. Массивы с одним индексом называют одномерными, с двумя — двумерными и т. д. Массивы с большим количеством измерений могут занимать очень большие объемы памяти, так что следует быть осторожным в их применении.

Прежде чем использовать массив, нужно обязательно объявить его с помощью оператора `Dim` и указать при этом тип хранящихся в массиве значений. Все значения в массиве обязаны принадлежать к одному типу данных. Это ограничение на практике можно обойти, использовав при объявлении массива тип `Variant` — в этом случае элементы массива смогут принимать значения разных типов. Синтаксис оператора объявления массива следующий:

```
Dim <имяМассива> (<размер1>, <размер2>, ...) As <типДанных>
```

где указанные в скобках величины `<размер1>`, `<размер2>` задают размеры массива — количество индексов и максимально допустимое значение для каждого конкретного

индекса. При этом индексирование элементов массива по умолчанию начинается с нуля. Так, объявление

```
Dim Array1 (9) As Integer
```

определяет одномерный массив из 10 элементов, являющихся переменными целого типа, а объявление

```
Dim Array2 (4, 9) As Variant
```

определяет двумерный массив из пятидесяти (5×10) элементов, являющихся переменными универсального типа `Variant`.

Замечание

В качестве стандартного значения нижней границы массива (индекса) может использоваться не только ноль. Чтобы изменить это стандартное значение, нужно воспользоваться оператором `Option Base`. Например, если поместить в начало вашего модуля оператор `Option Base 1`, то индексирование элементов массивов по умолчанию будет начинаться не с нуля, а с единицы.

При объявлении массива можно указать не только верхнюю границу индекса, но и его нижнюю границу, т. е. явно задать диапазон изменения конкретного индекса массива, причем нижняя граница может быть любым целым числом, не обязательно неотрицательным. Вот синтаксис такого определения:

```
Dim <имяМассива> (<мин1> To <макс1>, ...) As <типДанных>
```

Например, если вы собираетесь работать с массивом метеорологических данных, представляющих собой средние дневные температуры за последние две недели, то может оказаться весьма удобным дать следующее определение массива:

```
Dim Temperature (-14 To 0) As Single
```

При этом, например, `Temperature(-2)` будет соответствовать позавчерашней температуре, а для определения нужного индекса для интересующего вас дня будет достаточно использовать разность дат.

В приведенных ранее примерах речь шла о массивах фиксированного размера, количество элементов в которых явно указано во время описания массива в операторе `Dim`. Такие массивы называются *статическими*. В VBA допускается использование и *динамических* массивов, размеры которых при описании не фиксируются. Определение размера динамического массива может быть сделано непосредственно во время выполнения программы.

При определении динамического массива в операторе `Dim` после имени массива стоят лишь пустые скобки и описание типа переменных. Количество индексов и диапазон их изменения не задаются. Однако перед тем как использовать массив, нужно выполнить оператор `ReDim`, который задаст размерность и диапазоны изменения индексов динамического массива.

Синтаксис объявления и определения размеров динамического массива таков:

```
Dim <имяМассива> () As <типДанных>
ReDim <имяМассива> (<размер1>, <размер2>, ...)
```

Вот как может выглядеть объявление, определение размеров и использование динамического массива, а затем последующее изменение размерности и размеров этого же массива:

```
Dim dArray() As Variant
ReDim dArray(1, 2)
dArray(0, 0) = 2
dArray(0, 1) = 3
k = dArray(0, 0) + dArray(0, 1)
ReDim dArray(k)
dArray(0) = "Строка1"
```

В этом примере массив `dArray` сначала определяется как двумерный массив из шести элементов, а затем — как одномерный массив, причем верхняя граница индекса задается значением переменной `k`.

Замечание

Чтобы определить текущую нижнюю или верхнюю границу массива, можно использовать функции `LBound` и `UBound`, соответственно.

Учтите, что по умолчанию при изменении размеров массива ему заново выделяется память и текущие значения его элементов теряются. Чтобы не потерять текущие значения массива при изменении его размеров, используется ключевое слово `Preserve`. Например, чтобы увеличить размер массива `dArray` на один элемент, не потеряв значений существующих элементов, можно поступить следующим образом:

```
ReDim Preserve dArray(UBound(dArray) + 1)
```

Область действия переменных и процедур

Все процедуры, функции, переменные и константы в VBA имеют свою *область действия*. Это означает, что они могут использоваться только в определенном месте программного кода — именно там, где они описаны. Например, если переменная `A` описана с помощью оператора `Dim` в теле процедуры с именем `Proc1`, именно эта процедура и является ее областью действия. Таким образом, если имеется другая процедура `Proc2`, вы не можете использовать в ней эту же переменную. Если вы попытаетесь сделать это, то либо получите сообщение об ошибке из-за использования неописанной переменной (в том случае, если используется упоминавшийся ранее оператор `Option Explicit`), либо просто получите другую переменную — с тем же самым именем, но никак не связанную с одноименной переменной из первой процедуры.

Начнем с определения области действия переменных. В каком месте программы и как именно описана переменная, определяет область ее действия и то, как долго она "живет" в памяти и сохраняет присвоенное ей значение. Имеется три различных уровня при определении области действия переменных:

- уровень процедуры;
- уровень модуля;
- уровень проекта.

Чтобы определить переменную на уровне процедуры, ее описание помещается в тело данной процедуры, и тогда это будет локальная переменная данной процедуры.

Чтобы определить процедуру на уровне модуля и сделать ее тем самым доступной для совместного использования во всех процедурах данного модуля, следует поместить ее описание в секции объявлений модуля — перед текстом каких-либо процедур или функций. При этом может использоваться и явное описание области действия: вместо ключевого слова `Dim` в этом случае используется ключевое слово `Private`. Нет никакой разницы в том, какой из этих описателей вы используете.

Наконец, чтобы описать переменную на уровне проекта, необходимо расположить ее описание в секции объявлений одного из модулей проекта и при этом обязательно должно использоваться ключевое слово `Public`. Описанные таким образом переменные могут использоваться в любом модуле проекта.

Замечание

Все сказанное ранее относится и к описанию, и к определению области действия констант, а также массивов.

Для переменных имеется еще один способ их описания, не изменяющий их уровня, но позволяющий сохранить значение переменной, описанной на уровне процедуры, после завершения работы данной процедуры. Для этого следует использовать описатель `Static`, тем самым определяя ее как *статическую* переменную. Такая переменная сохраняет выделенное ей место в памяти и свое значение даже после завершения процедуры, в которой она была описана и использована.

Тем не менее статическая переменная не может быть использована в других процедурах. Изменяется лишь время ее жизни, но не область действия. Если произойдет повторный вызов той же самой процедуры, в которой была описана статическая переменная, то эта переменная сохранит свое прежнее значение, которое она имела в момент завершения работы этой процедуры при предыдущем вызове. Обыкновенные, не статические, переменные всякий раз инициализируются заново и получают при входе в процедуру пустые значения.

Перейдем к обсуждению области действия процедур и функций. Процедуры и функции имеют только два уровня области действия: уровень модуля и уровень проекта. По умолчанию используется уровень проекта. Таким образом, процедура или функция может быть вызвана любой другой процедурой или функцией в этом проекте. При описании процедур и функций на уровне проекта может также использоваться необязательное ключевое слово `Public`. Никакого воздействия на процедуру наличие или отсутствие этого слова не оказывает.

Если требуется описать процедуру, используемую только на уровне модуля, то для этого применяется ключевое слово `Private`. Учтите, что такое описание не только сужает область действия для процедуры, но и запрещает ее использование как самостоятельной процедуры — ее можно вызвать только из другой процедуры.

Наконец, при описании процедур или функций может использоваться и ключевое слово `Static`. Оно никак не влияет на область действия процедуры, но воздействует на все переменные, описанные внутри этой процедуры или функции. В этом случае все локальные переменные получают статус `Static` и тем самым остаются в памяти после завершения такой процедуры и при повторном ее вызове сохраняют свои прежние значения.

Рассмотрим пример модуля, начинающегося следующим образом:

```
Public A1 As String
Private A2 As Integer
Dim A3 As Single
Sub Proc1()
    Dim A4 As Integer
    Static A5 As Integer
    A1 = "Текстовая строка 1"
    A2 = 2
    A3 = 3.14
    A4 = A4 + 4
    A5 = A5 + 5

    MsgBox A4
    MsgBox A5
End Sub

Sub Proc2()
    Proc1

    MsgBox A1
    MsgBox A2
    MsgBox A3
    MsgBox A4
    MsgBox A5

    Proc1
End Sub
```

В этом примере переменная `A1` определена на уровне всего проекта (использовано ключевое слово `Public`), переменные `A2` и `A3` определены на уровне модуля, переменная `A4` — только на уровне процедуры `Proc1`, а переменная `A5` хотя и определена в теле процедуры `Proc1`, но описана как статическая переменная.

При вызове процедуры `Proc2` произойдет следующее: из этой процедуры будет в свою очередь вызвана процедура `Proc1`, которая присвоит значения всем пяти переменным `A1`, `A2`, `A3`, `A4` и `A5`, а затем покажет текущие значения переменных `A4` и `A5` в диалоговом окне.

После завершения этой процедуры будут выведены текущие значения переменных `A1`—`A5` из процедуры `Proc2`. При этом окажется, что переменные `A1`—`A3` сохранили свои значения, поскольку они описаны на уровне модуля, а переменные `A4` и `A5` принимают пустые значения, т. к. областью действия этих переменных являются процедуры, в которых они используются. Никакие изменения этих переменных внутри одной из процедур не имеют отношения к аналогичным переменным из другой процедуры — на самом деле это разные переменные, просто для них используются совпадающие имена.

Затем происходит еще один вызов процедуры `Proc1`, и она вновь начинает изменять и выводить на экран значения переменных `A4` и `A5`. При этом переменная `A4` вновь получит значение `4`, поскольку при новом вызове процедуры для этой переменной

будет заново выделена память, и она будет инициализирована пустым значением. В отличие от A4, переменная A5, описанная как статическая, сохранит свое прежнее значение от предыдущего вызова этой процедуры, в результате ее значение при повторном вызове окажется равным 10.

Управляющие конструкции

Как и во всех других языках программирования, в VBA имеются различные управляющие конструкции, позволяющие изменять порядок выполнения программы. Если управляющие конструкции не используются, происходит последовательное выполнение операторов языка программирования, начиная с самого первого и кончая последним. Хотя в некоторых самых простых случаях этого и бывает достаточно, обычно все-таки требуется изменять порядок исполнения операторов при выполнении определенных условий либо пропуская выполнение некоторых операторов, либо, наоборот, многократно повторяя их. Оказывается, для реализации любых алгоритмов достаточно иметь только два вида конструкций управления: ветвления и циклы.

Ветвления

Управляющие конструкции ветвления позволяют проверить некоторое условие и, в зависимости от результатов этой проверки, выполнить ту или иную группу операторов. Для организации ветвлений в VBA используются различные формы оператора ветвления If и оператор выбора Select Case.

Простейшая, краткая форма оператора If используется для проверки одного условия, а затем либо выполнения, либо пропуска одного оператора или блока из нескольких операторов. Краткая форма оператора ветвления If может иметь как однострочную, так и блочную форму. В одну строку краткая форма If может быть записана так:

```
If <условие> Then <оператор>
```

В блочной форме краткое ветвление выглядит следующим образом:

```
If <условие> Then
    <оператор1>
    <оператор2>
    ...
End If
```

В качестве условия можно использовать логическое выражение, возвращающее значение True или False, или любое арифметическое выражение. Если используется арифметическое выражение, то нулевое значение этого выражения эквивалентно логическому значению False, а любое ненулевое выражение эквивалентно True. В том случае, когда условие возвращает значение False, оператор или блок операторов, заключенных между ключевыми словами Then и End If и составляющих тело краткого оператора ветвления, не будет выполняться.

Замечание

Обратите внимание, что при записи краткого оператора ветвления в одну строку ключевые слова End If не используются.

Полная форма оператора `If` используется в тех случаях, когда имеются два различных блока операторов и по результатам проверки условия нужно выполнить один из них. Такая форма `If` не может записываться в одну строку и всегда имеет блочную форму записи:

```
If <условие> Then
    <блокОператоров1>
Else
    <блокОператоров2>
End If
```

Если условие истинно, выполняется первый блок операторов, заключенный между ключевыми словами `Then` и `Else`, а в противном случае — второй блок, заключенный между ключевыми словами `Else` и `End If`.

Совет

Для того чтобы текст вашей процедуры был понятным и удобным для восприятия, рекомендуется делать отступы для групп операторов так, как это указано при описании их синтаксиса. В VBA предусмотрено удобное средство изменения отступов — нажатие на клавишу <Tab> увеличивает отступ вправо, нажатие комбинации клавиш <Shift>+<Tab> уменьшает этот отступ.

Иногда приходится делать выбор одного действия из целой группы действий на основе проверки нескольких различных условий. Для этого можно использовать цепочку операторов ветвления `If...Then...ElseIf`:

```
If <условие1> Then
    <блокОператоров1>
ElseIf <условие2> Then
    <блокОператоров2>
ElseIf <условие3> Then
    <блокОператоров3>
...
ElseIf <условиеN> Then
    <блокОператоровN>
Else
    <блокОператоров_Else>
End If
```

Такие цепочки операторов `If...Then...ElseIf` обладают большой гибкостью и позволяют решить все проблемы, однако если выбор одной из нескольких возможностей все время основан на различных значениях одного и того же выражения, гораздо удобнее использовать специально предназначенный для этого оператор выбора `Select Case`, имеющий следующий синтаксис:

```
Select Case <проверяемоеВыражение>
    Case <списокЗначений1>
        <блокОператоров1>
    Case <списокЗначений2>
        <блокОператоров2>
    Case <списокЗначений3>
        <блокОператоров3>
    ...
```

```
    Case Else
        <блокОператоровElse>
End Select
```

Проверяемое выражение вычисляется в начале работы оператора `Select Case`. Это выражение может возвращать значение любого типа, например логическое, числовое или строковое.

Список выражений представляет собой одно или несколько выражений, разделенных запятой. При выполнении оператора проверяется, соответствует ли хотя бы один из элементов этого списка проверяемому выражению. Элементы списка выражений могут иметь одну из следующих форм:

- `<выражение>` — проверяется, совпадает ли значение проверяемого выражения с этим выражением;
- `<выражение1> To <выражение2>` — проверяется, находится ли значение проверяемого выражения в указанном диапазоне значений;
- `Is <логическийОператор> <выражение>` — проверяемое выражение сравнивается с указанным значением с помощью заданного логического оператора (например, условие `Is >= 10` считается выполненным, если проверяемое значение не меньше 10).

Если хотя бы один из элементов списка соответствует проверяемому выражению, то выполняется соответствующая группа операторов и на этом выполнение оператора `Select Case` заканчивается, а остальные списки выражений не проверяются, т. е. отыскивается только первый подходящий элемент списков выражений. Если же ни один из элементов всех этих списков не соответствует значению проверяемого выражения, выполняются операторы группы `Else`, если таковая присутствует.

Циклы

В VBA имеется богатый выбор средств организации циклов, которые можно разделить на две основные группы — *циклы с условием* `Do...Loop` и *циклы с перечислением* `For...Next`.

Циклы типа `Do...Loop` используются в тех случаях, когда заранее неизвестно, сколько раз должно быть повторено выполнение блока операторов, составляющего тело цикла. Такой цикл продолжает свою работу до тех пор, пока не будет выполнено определенное условие. Существуют четыре вида циклов `Do...Loop`, которые различаются типом проверяемого условия и временем выполнения этой проверки. В табл. 13.2 приводится синтаксис этих четырех конструкций.

Таблица 13.2. *Синтаксис операторов цикла* `Do...Loop`

Конструкция	Описание

`Do While <условие>`
`    <блокОператоров>`
`Loop`
	Условие проверяется до того, как выполняется группа операторов, образующих тело цикла. Цикл продолжает свою работу, пока это условие выполняется (т. е. имеет значение `True`), иными словами, в этой конструкции указывается условие продолжения работы цикла

Таблица 13.2 *(окончание)*

Конструкция	Описание

`Do`
`  <блокОператоров>`
`Loop While <условие>`
	Условие проверяется после того, как операторы, составляющие тело цикла, будут выполнены хотя бы один раз. Цикл продолжает свою работу, пока это условие остается истинным, иными словами, в этой конструкции указывается условие продолжения работы цикла

`Do Until <условие>`
`  <блокОператоров>`
`Loop`
	Условие проверяется до того, как выполняется группа операторов, образующих тело цикла. Цикл продолжает свою работу, если это условие еще не выполнено, и прекращает работу, когда оно станет истинным, иными словами, в этой конструкции указывается условие прекращения работы цикла

`Do`
`  <блокОператоров>`
`Loop Until <условие>`
	Условие проверяется после того, как операторы, составляющие тело цикла, будут выполнены хотя бы один раз. Цикл продолжает свою работу, если это условие еще не выполнено, а когда оно станет истинным, цикл прекращает работу, иными словами, в этой конструкции указывается условие прекращения работы цикла

Имеется также две разновидности оператора цикла с перечислением `For...Next`. Очень часто при обработке массивов, а также в тех случаях, когда требуется повторить выполнение некоторой группы операторов заданное число раз, используется цикл `For...Next` со счетчиком. В отличие от циклов `Do...Loop`, данный тип цикла использует специальную переменную, называемую *счетчиком*, значение которой увеличивается или уменьшается при каждом выполнении тела цикла на определенную величину. Когда значение этой переменной достигает заданной величины, выполнение цикла заканчивается.

Синтаксис этого вида цикла выглядит следующим образом (в квадратные скобки заключены необязательные элементы синтаксической конструкции):

```
For <счетчик> = <начальноеЗначение> To <конечноеЗначение>
    [Step <приращение>]
    <блокОператоров>
Next [<счетчик>]
```

Несколько пояснений к приведенному описанию:

- `<приращение>` — может быть как положительным, так и отрицательным числом. Если использовать отрицательное приращение, то конечное значение должно быть меньше либо равно начальному значению для того, чтобы тело цикла выполнилось хотя бы один раз;
- после завершения работы цикла `For...Next` переменная, которая использовалась в качестве счетчика, получает значение, обязательно превосходящее конечное значение в том случае, если приращение положительно, и строго меньшее конечного значения, если приращение отрицательно;
- если начальное и конечное значения совпадают, тело цикла выполняется лишь один раз.

Рассмотрим еще одну разновидность цикла `For...Next`, часто использующуюся в VBA при обработке объектов, составляющих *массив* или *семейство* однородных объектов. В этой разновидности цикла счетчик отсутствует, а тело цикла выполняется для каждого элемента массива или семейства объектов. Синтаксис такого цикла следующий:

```
For Each <элемент> In <совокупность>
    <блокОператоров>
Next [<элемент>]
```

где `<элемент>` — это переменная, используемая для ссылки на элементы семейства объектов; `<совокупность>` — это имя массива или семейства.

Приведем пример использования подобного цикла. Следующая процедура предназначается для выдачи на печать списка всех полей для всех таблиц текущей открытой базы данных:

```
Public Sub EnumerateAllFields()
Dim MyBase As Database
Dim tdf As TableDef, fld As Field
    Set MyBase = CurrentDb()
    For Each tdf In MyBase.TableDefs
        Debug.Print "Таблица: " & tdf.Name
        For Each fld In tdf.Fields
            Debug.Print "   Поле: " & fld.Name
        Next fld
    Next tdf
    Set MyBase = Nothing
End Sub
```

Итак, в операторах `Dim` мы объявили переменную `MyBase` как объект "база данных DAO", переменные `tdf` и `fld` — как определение таблицы и поле таблицы, соответственно. Оператор `Set` назначает переменной `MyBase` текущую открытую базу данных. Далее для каждого определения таблицы выполняется вывод на печать названия таблицы, а затем вложенный цикл такого же типа печатает названия всех ее полей.

Приведем еще один пример использования подобного оператора цикла `For Each...Next` для обработки всех элементов многомерного массива. Пусть у нас имеется трехмерный числовой массив из 1000 элементов (размерами 10×10×10), который мы хотим заполнить случайными вещественными числами в диапазоне от 0 до 1. Если бы мы применяли обычные циклы `For...Next` со счетчиками, используя счетчики в качестве индексов элементов массива, то для решения этой задачи потребовалось бы написать три вложенных цикла `For...Next`:

```
Dim tArray(9, 9, 9) As Single
Dim i%, j%, k%
Randomize
For i=0 To 9
    For j=0 To 9
        For k=0 To 9
            tArray(i, j, k) = Rnd()
```

```
        Next k
    Next j
Next i
```

На самом же деле достаточно всего одного цикла, если вместо циклов со счетчиками воспользоваться циклом `For Each ... Next`:

```
Dim tArray(9, 9, 9) As Single
Dim elem As Variant
Randomize
For Each elem In tArray
    elem = Rnd()
Next
```

Еще раз порекомендуем использовать отступы при записи циклов так же, как и при записи операторов ветвления.

Выход из циклов и процедур

Обычно выполнение процедуры заканчивается после выполнения ее последнего оператора, а выполнение цикла — после нескольких выполнений тела цикла, когда достигнуто условие завершения его работы. Однако в некоторых случаях бывает нужно прекратить выполнение процедуры или цикла досрочно, избежав выполнения лишних операторов процедуры или лишних повторений цикла.

Например, если при выполнении процедуры произошла ошибка, которая делает продолжение ее работы бессмысленным, можно выполнить команду немедленного выхода из процедуры. Другой пример: если цикл `For...Next` используется для поиска нужного значения в массиве, то после того, как нужный элемент массива найден, нет смысла продолжать дальнейший перебор элементов массива. Досрочный выход из управляющей конструкции можно осуществить с помощью одного из операторов `Exit`. Для досрочного выхода из циклов `Do...Loop` используется оператор `Exit Do`, а для выхода из циклов `For` — оператор `Exit For`. Для досрочного выхода из процедур и функций используются операторы `Exit Sub` и `Exit Function`, соответственно.

Следует отметить, что хотя использование оператора `Exit` может быть вполне оправданным, необходимо избегать излишнего употребления этого оператора, прибегая к нему только в крайних случаях. Излишне частое применение данного оператора затрудняет понимание написанного текста программы и его отладку.

Например, в следующем цикле поиска `For...Next` используется оператор `Exit For`:

```
ub = Ubound(dArray)
fFound = False
For i=LBound(dArray) To ub
    If dArray(i) = searchValue Then
        fFound = True
        Exit For
    End If
Next
```

Однако такого применения оператора `Exit` вполне можно избежать. Приведем пример подобного цикла поиска, но без использования `Exit`:

```
i = Lbound(dArray)
ub = Ubound(dArray)
fFound = False
Do
    If dArray(i) = searchValue Then fFound = True
    i = i + 1
Loop Until (i > ub) Or fFound
```

На этом мы заканчиваем краткое знакомство с основными понятиями языка VBA. Если читателю требуется более подробное и строгое изложение синтаксиса VBA, то лучше обратиться к специальным книгам, посвященным этому языку. В других разделах настоящей главы, а также в последующих главах мы покажем, как VBA используется при разработке приложений, и познакомим с теми его особенностями, которые специфичны именно для Access. Это касается в первую очередь объектов Access и объектов данных.

Объектные модели Microsoft Access 2003

Язык программирования Visual Basic для приложений является объектно-ориентированным. Знание технологии объектно-ориентированного программирования и состава объектных моделей Visual Basic позволяет разрабатывать в среде Access профессиональные приложения, выполняющие всю необходимую обработку данных.

Основные понятия

Начнем с определения основных понятий, которые потребуются для понимания работы с объектами в языке VBA.

Объект — это абстракция, с которой мы оперируем в объектно-ориентированных языках программирования. Объект обладает собственными характерными признаками, отличающими его от других объектов, и имеет свое поведение. Примерами объектов Access являются уже известные вам таблицы, формы, отчеты, запросы.

Класс представляет собой описание совокупности однотипных объектов. Класс можно сравнить с типом данных, где переменной такого специфического типа является объект. В этом случае говорят, что объект представляет собой экземпляр определенного класса.

Каждый объект имеет свойства и методы, которые различны у разных классов объектов, но используются они одинаково, в чем вы скоро убедитесь на примерах использования объектов в процедурах VBA.

Свойством называют отдельную характеристику объекта или класса. Например, знакомые уже вам свойства формы являются свойствами объекта `Form`. Свойство объекта может принимать определенное значение. Например, свойство **Вывод на экран** (Visible) может принимать значение `True` или `False`, в зависимости от чего форма будет появляться или исчезать с экрана.

Метод представляет собой процедуру (или функцию) объекта или класса. Совокупность методов объекта определяет его "поведение". Например, объект `Form` имеет метод `Refresh`, вызов которого позволяет обновить данные в форме Access.

Объект может реагировать на определенные *события*, происходящие в процессе работы приложения и влияющие на объект. Совокупность событий, на которые объект способен реагировать, определяется создателем класса, экземпляром которого является данный объект. Например, набор событий, которые определены для формы Access, мы видим на вкладке **События** (Event) диалогового окна **Свойства** (Properties). Реакцией объекта на произошедшее событие может быть выполнение объектом некоторой специальной процедуры, которая называется *процедурой обработки события*. Любому событию объекта может быть назначена некоторая процедура его обработки.

Упорядоченный набор однотипных объектов — экземпляров одного класса называется *семейством*. Семейство тоже является объектом. Одним из методов этого объекта является процедура, возвращающая ссылку на конкретный объект в семействе. Одним из свойств семейства является число объектов, хранящихся в нем. Например, совокупность элементов управления в форме образует семейство `Controls`.

Объекты и семейства сгруппированы в виде иерархических структур, которые называются *объектными моделями*. В VBA определены специальные объектные модели для каждого компонента семейства Microsoft Office и объектные модели, общие для всех компонентов Microsoft Office. Объектные модели VBA можно изучать, используя справочную систему и окно просмотра объектов. Окно просмотра объектов представляет собой специальное средство редактора Visual Basic, позволяющее просматривать содержимое библиотек объектов и производить поиск справочной информации. *(Описание окна просмотра объектов можно найти в разд. "Среда программирования VBA" данной главы.)*

Объектные модели Microsoft Office 2003

Управление приложениями семейства Microsoft Office 2003 осуществляется интерактивно — посредством интерфейса пользователя или программно — с помощью объектных моделей. Каждый из компонентов Microsoft Office предоставляет свои объектные модели в виде одноименной библиотеки объектов (файл с расширением olb), которая может быть использована в других приложениях. Microsoft Office Access 2003, как компонент Microsoft Office, имеет такую библиотеку — Microsoft Access 11.0 Object Library. Доступ и управление данными в Microsoft Office Access могут осуществляться с помощью объектных моделей, реализованных в нескольких библиотеках:

- библиотека объектов доступа к данным (Microsoft DAO 3.6 Object Library);
- библиотека объектов ActiveX (Microsoft ActiveX Data Objects 2.6);
- расширения ADO для поддержки DDL и защиты информации (Microsoft ADO Extensions 2.6 for DDL and Security);
- библиотека объектов Microsoft Jet и репликации (Microsoft Jet and Replication Objects 2.6).

Кроме этого, в приложениях Access обычно используется библиотека объектов собственно языка Visual Basic для приложений — общая для всех приложений Microsoft Office, а также при необходимости другие библиотеки, о которых мы поговорим тогда, когда будем описывать работу с этими библиотеками.

Объектная модель Microsoft Office Access 2003

Объектная модель Microsoft Office Access 2003 реализована в виде набора объектов, собранных в библиотеке Access. Основным элементом в иерархии объектов библиотеки Access является объект `Application`. Он содержит ссылки на все объекты и семейства объектов Microsoft Office Access. Каждый объект из библиотеки Access имеет в качестве свойства объект `Application` (в том числе и сам объект `Application` имеет свойство `Application`), который ссылается на активное приложение Microsoft Office Access.

Иерархия объектов и семейств объектов Microsoft Office Access 2003 представлена на рис. 13.1—13.5. Названия объектов, являющихся элементами семейств, приведены в скобках. Новые объекты, появившиеся в Microsoft Office Access 2003, изображены на рис. 13.5. Иерархия объектов образована следующим образом:

- каждый объект может содержать набор свойств, часть из которых может являться ссылками на другие объекты;
- в каждый новый уровень иерархии входят объекты, ссылки на которые хранятся в объектах, расположенных на предыдущем уровне иерархии.

Если свойство объекта представляет собой ссылку на объект, определенный в другой библиотеке (не в библиотеке Access), для него приводится название этой библиотеки.

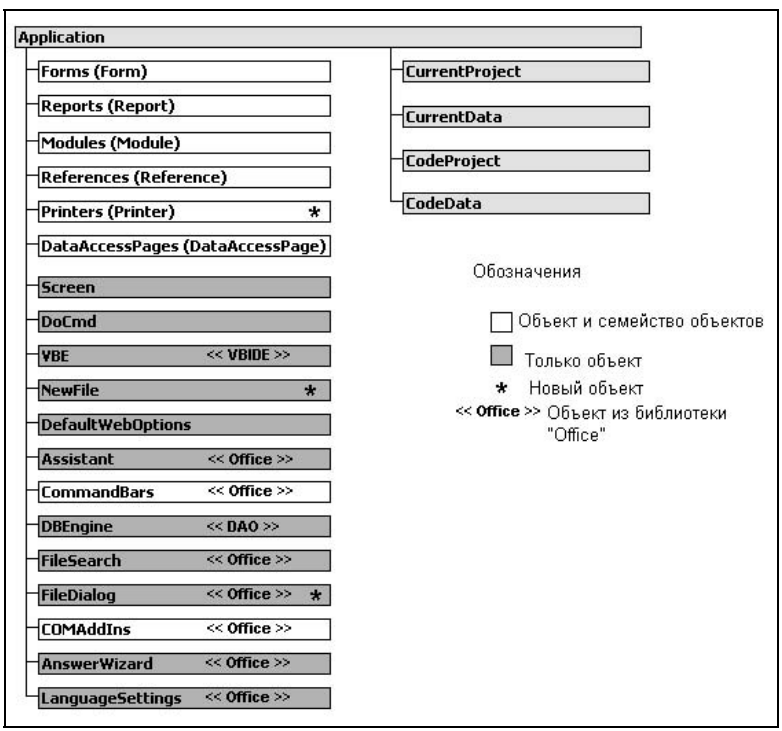


Рис. 13.1. Первый уровень иерархии объектной модели Microsoft Office Access 2003

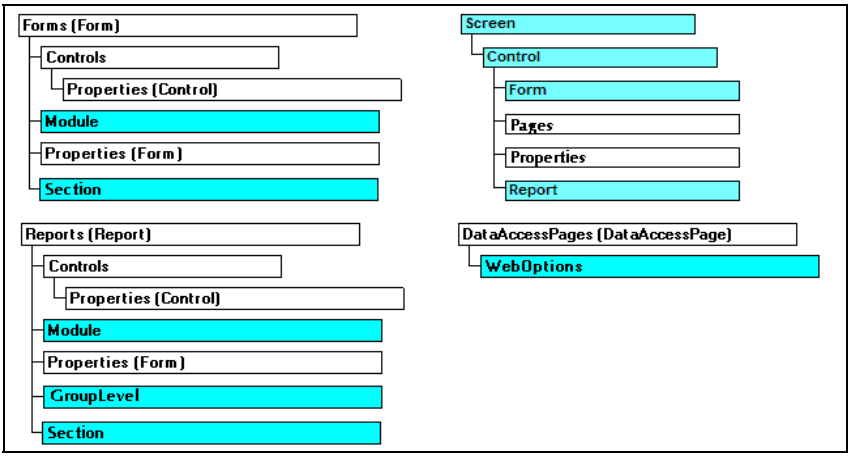


Рис. 13.2. Второй уровень иерархии объектной модели Microsoft Office Access 2003 — объекты для работы с базой данных и проектом

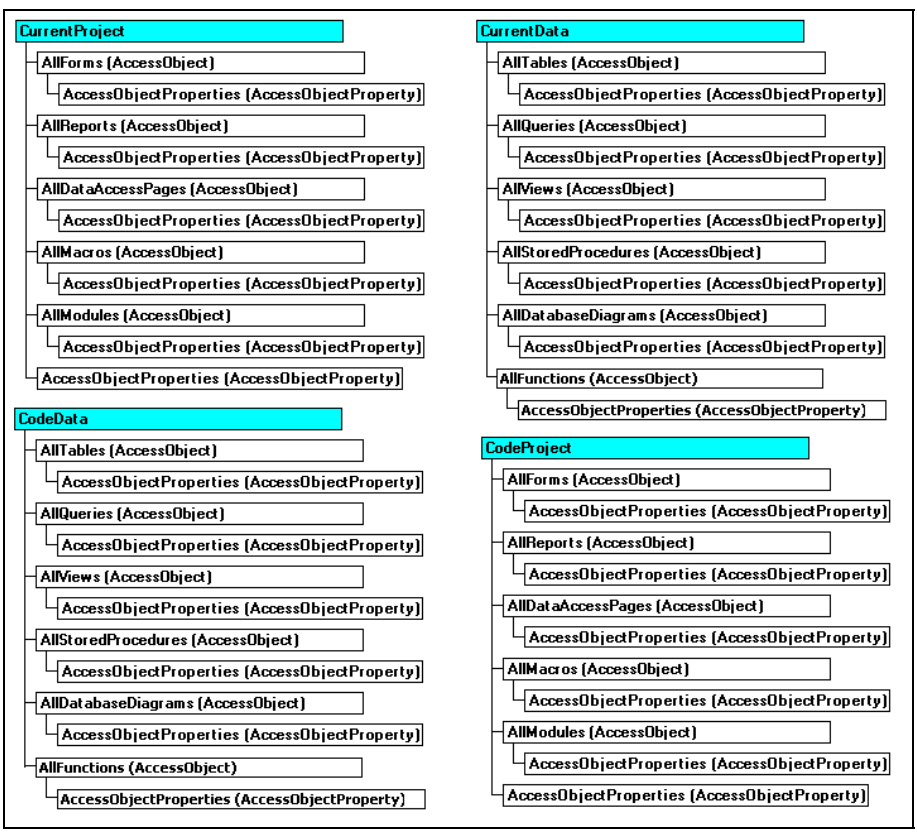


Рис. 13.3. Второй уровень иерархии объектной модели Microsoft Office Access 2003 — объекты баз данных

Глядя на схемы объектной модели, можно определить, какие объекты описывают приложение, как они связаны между собой и как составить ссылку для доступа к конкретному объекту.

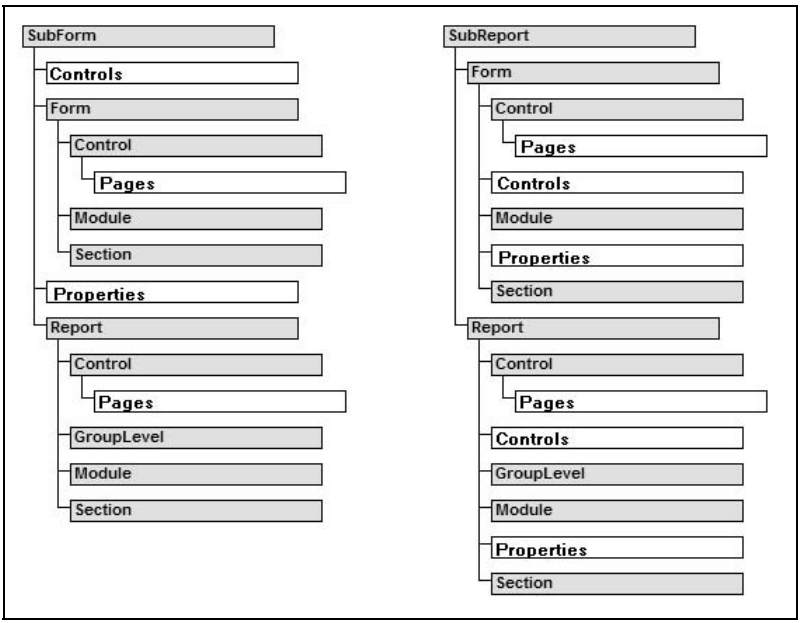


Рис. 13.4. Второй уровень иерархии объектной модели Microsoft Office Access 2003 — подчиненные формы/отчеты

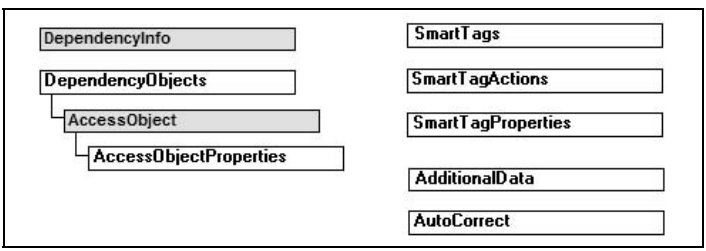


Рис. 13.5. Второй уровень иерархии объектной модели Microsoft Office Access 2003 — новые объекты

Табл. 13.3 содержит описание объектов Microsoft Office Access 2003, определенных в библиотеке Access, если не указана другая библиотека объектов. В целях сокращения объема книги объекты, составляющие семейство, не описываются отдельно, представлено только само семейство.

Таблица 13.3. *Описание основных элементов объектной модели Microsoft Office Access 2003*

Объект	Тип	Описание
`Application`	Объект	Ссылается на активное приложение Microsoft Office Access. Используется для управления приложением. Этот объект является COM-компонентом и может быть использован другим приложением, которое поддерживает Automation
`Forms`	Семейство	Содержит объекты `Form`, соответствующие всем открытым в данный момент формам в базе данных Access. Каждый объект ссылается на конкретную форму Access, описывает свойства формы, элементы управления и модуль, содержащий процедуры формы
`Reports`	Семейство	Содержит объекты `Report`, соответствующие всем открытым в данный момент отчетам в базе данных Access. Каждый объект ссылается на конкретный отчет Access, описывает свойства отчета, элементы управления и модуль, содержащий процедуры отчета
`Modules`	Семейство	Содержит объекты `Module`, соответствующие всем открытым в данный момент стандартным модулям и модулям объектов в базе данных Access
`References`	Семейство	Содержит объекты `Reference`, представляющие собой установленные в проекте Access ссылки на внешние библиотеки и проекты. С помощью данного семейства можно динамически устанавливать или удалять ссылки
`Printers`	Семейство	Содержит объекты `Printer`, представляющие собой установленные ссылки на все печатающие устройства, доступные в приложении. Обеспечивает программное управление параметрами печати
`DataAccessPages`	Семейство	Содержит объекты `DataAccessPages`, соответствующие открытым в данный момент страницам доступа к данным в базе данных или проекте Access
`Screen`	Объект	Ссылается на конкретную форму, отчет или элемент управления, который в данный момент имеет фокус
`DoCmd`	Объект	Позволяет выполнить макрос или встроенную инструкцию Access с помощью процедуры на Visual Basic

Таблица 13.3 *(продолжение)*

Объект	Тип	Описание
`VBE`	Объект из библиотеки VBIDE	Предоставляет доступ к функциональным средствам редактора Visual Basic для Microsoft Office Access
`DefaultWebOptions`	Объект	Предоставляет доступ к атрибутам приложения, используемым в Access по умолчанию при открытии и сохранении Web-страниц
`WebOptions`	Объект	Предоставляет доступ к атрибутам конкретной страницы доступа к данным, используемым в Access при открытии и сохранении Web-страниц. Эти атрибуты имеют более высокий приоритет, чем соответствующие атрибуты приложения. Если установлены атрибуты страницы доступа к данным, то соответствующие атрибуты приложения для работы с Web-страницами игнорируются. Если изменяются атрибуты страницы доступа к данным, автоматически такие же значения получают атрибуты приложения. Поэтому полезно сохранять текущие атрибуты приложения, чтобы восстанавливать их после изменения атрибутов страницы доступа к данным
`DBEngine`	Объект из библиотеки DAO	Является объектом самого верхнего уровня в объектной модели DAO. Позволяет использовать объекты доступа к данным в приложении Access
`CurrentProject`	Объект	Ссылается на программный проект (представляющий совокупность всех программных модулей, включая стандартные модули и модули классов) текущей базы данных или проекта Microsoft Office Access. Этот объект содержит семейства объектов `AccessObjects`, соответствующих реальным объектам базы данных или проекта: `AllForms`, `AllReports`, `AllMacros`, `AllModules`, `AllDataAccessPages`. Перечисленные семейства включают в себя все реальные объекты базы данных, независимо от того, открыты они или закрыты в данный момент
`CurrentData`	Объект	Ссылается на объекты, сохраненные приложением — источником данных (ядром Jet или SQL-сервером) в текущей базе данных. Содержит семейства объектов `AccessObject`: `AllTables`, `AllQueries`, `AllViews`, `AllStoredProcedures`, `All Functions`, `AllDatabaseDiagrams`. Перечисленные семейства включают в себя все реальные объекты базы данных, независимо от того, открыты они или закрыты в данный момент

Таблица 13.3 *(продолжение)*

Объект	Тип	Описание
`CodeProject`	Объект	Ссылается на программный проект кода той базы данных (или проекта Microsoft Office Access), в которой выполняется (и содержится) код VBA, включающий данную ссылку. Этот объект содержит семейства объектов `AccessObjects`, соответствующих реальным объектам базы данных или проекта: `AllForms`, `AllReports`, `AllMacros`, `AllModules`, `AllDataAccessPages`. Перечисленные семейства включают в себя все реальные объекты базы данных, независимо от того, открыты они или закрыты в данный момент
`CodeData`	Объект	Ссылается на объекты, сохраненные приложением — источником данных (ядром Jet или SQL-сервером) в той базе данных, в которой выполняется (и содержится) код VBA, включающий данную ссылку. Содержит семейства объектов `AccessObject: AllTables`, `AllQueries`, `AllViews`, `AllStoredProcedures`, `All Functions`, `AllDatabaseDiagrams`. Перечисленные семейства включают в себя все реальные объекты базы данных, независимо от того, открыты они или закрыты в данный момент
`FormatConditions`	Семейство	Содержит объекты `FormatCondition` и представляет набор условных форматов для объекта `TextBox` или `ComboBox`
`AllForms`	Семейство	Содержит объекты `AccessObject` и представляет все формы в объекте `CurrentProject` или `CodeProject`
`AllReports`	Семейство	Содержит объекты `AccessObject` и представляет все отчеты в объекте `CurrentProject` или `CodeProject`
`AllDataAccessPages`	Семейство	Содержит объекты `AccessObject` и представляет все страницы доступа к данным в объекте `CurrentProject` или `CodeProject`
`AllMacros`	Семейство	Содержит объекты `AccessObject` и представляет все макросы в объекте `CurrentProject` или `CodeProject`
`AllModules`	Семейство	Содержит объекты `AccessObject` и представляет все программные модули в объекте `CurrentProject` или `CodeProject`

Таблица 13.3 *(продолжение)*

Объект	Тип	Описание
`AllTables`	Семейство	Содержит объекты `AccessObject` и представляет все таблицы в объекте `CurrentData` или `CodeData`
`AllQueries`	Семейство	Содержит объекты `AccessObject` и представляет все запросы в объекте `CurrentData` или `CodeData`. Количество запросов в проекте Access должно равняться нулю
`AllViews`	Семейство	Содержит объекты `AccessObject` и представляет все представления в объекте `CurrentData` или `CodeData`. Количество представлений в базе данных Access должно равняться нулю
`AllStoredProcedures`	Семейство	Содержит объекты `AccessObject` и представляет все хранимые процедуры в объекте `CurrentData` или `CodeData`. Количество хранимых процедур в базе данных Access должно равняться нулю
`All Functions`	Семейство	Содержит объекты `AccessObject` и представляет все экземпляры функций, описанных в объекте `CurrentData` или `CodeData`. Используется для программного доступа к пользовательским функциям, определенным в базе данных Microsoft SQL Server
`AllDatabaseDiagrams`	Семейство	Содержит объекты `AccessObject` и представляет все схемы базы данных в объекте `CurrentData` или `CodeData`. Количество этих схем в базе данных Access должно равняться нулю
`AccessObject`	Объект	Ссылается на реальный объект Microsoft Office Access в любом из семейств: `AllForms`, `AllReports`, `AllMacros`, `AllModules`, `AllDataAccessPages`, `AllTables`, `AllQueries`, `AllViews`, `AllStoredProcedures`, `All Functions`, `AllDatabaseDiagrams`. В зависимости от того, к какому семейству он принадлежит, этот объект представляет собой любой из объектов Access: таблицу (`Table`), запрос (`Query`), отчет (`Report`), форму (`Form`), модуль (`Module`), макрос (`Macro`), страницу доступа к данным (`Data access page`), представление (`View`), хранимую процедуру (`Stored procedure`) или схему базы данных (`Database diagram`). Объект `AccessObject` ссылается на существующий объект базы данных, нельзя создать новый или удалить существующий объект `AccessObject`

Таблица 13.3 *(продолжение)*

Объект	Тип	Описание
`AccessObjectProperties`	Семейство	Содержит настраиваемые объекты `AccessObjectProperty`, описывающие свойства и однозначно характеризующие конкретный объект `AccessObject`, `CodeData`, `CodeProject`, `CurrentData` или `CurrentProject`
`Controls`	Семейство	Содержит объекты `Control`, представляющие все элементы управления в конкретной форме, отчете или секции, вложенные или присоединенные элементы управления. Это семейство является свойством объектов `Form`, `Report`, `Section` и `Control`. Объект `Control` может ссылаться на один из объектов, характеризующих элемент управления определенного типа, такой как: `CheckBox`, `TextBox`, `ComboBox`, `CommandButton`, `CustomControl`, `BoundObjectFrame`, `Image`, `ListBox`, `ObjectFrame`, `OptionButton`, `OptionGroup`, `Page`, `Section`, `SubForm`, `TogleButton`
`Section`	Объект	Представляет собой любой раздел формы или отчета: заголовок, примечание, колонтитулы, раздел данных
`GroupLevel`	Объект	Представляет собой массив, содержащий все уровни группировки, существующие в отчете. Для обращения к конкретному уровню группировки используется выражение `GroupLevel(n)`, где `n` = 0,1,2,... Через свойства этого объекта можно программно влиять на способы группировки данных в отчетах
`Pages`	Семейство	Представляет собой семейство вкладок элемента управления **Набор вкладок** (Tab Control). Это семейство является свойством объекта `Control`
`AdditionalData`	Новое семейство	Представляет собой набор таблиц и запросов, которые будут включены в XML-файл вместе с родительской таблицей. Создается с помощью метода `CreateAdditionalData` объекта `Application`. Добавить таблицу или запрос к этому объекту можно с помощью метода `Add`
`AutoCorrect`	Объект	Управляет отображением кнопки автозамены рядом с текстовыми полями и полями со списком. Создается с помощью метода `AutoCorrect` объекта `Application`. Управление отображением кнопки выполняется путем установки `True`/`False` свойства `DisplayAutoCorrectOptions`

Таблица 13.3 (окончание)

Объект	Тип	Описание
DependencyInfo	Новый объект	Содержит информацию о зависимостях объекта AccessObject от других объектов. Создается с помощью метода GetDependencyInfo объекта AccessObject
DependencyObjects	Новое семейство	Семейство объектов, ссылки на которые содержатся в объекте DependencyInfo. Применяя метод Dependants к объекту AccessObject, можно получить семейство объектов, которые зависят от этого объекта, а применяя метод Dependencies, можно получить семейство объектов, от которых зависит указанный объект
SmartTags	Новое семейство	Семейство, которое представляет собой смарт-теги (Smart Tags), связанные с элементом управления в форме, отчете или странице доступа к данным
SmartTagActions	Новое семейство	Семейство, представляющее набор действий, связанных с конкретным смарт-тегом или типом смарт-тега. Представляет собой свойство объекта SmartTag. Чтобы выполнить какое-либо действие, нужно применить метод Execute к соответствующему объекту SmartTagAction
SmartTagProperties	Новое семейство	Семейство объектов, представляющих свойства объекта SmartTag. Чтобы получить это семейство, нужно использовать свойство Properties объекта SmartTag

Объектные модели доступа к данным

Объекты доступа к данным предназначены для программного доступа и управления данными в локальной или удаленной базе данных, а также для программного управления самими базами данных, их объектами и структурой. В Microsoft Office Access 2003 сохранилась возможность двух способов работы с данными:

- старый, посредством DAO (Data Access Objects);
- новый, посредством ADO (ActiveX Data Objects).

Каждому из этих способов соответствует своя объектная модель.

Объектная модель Microsoft DAO 3.6

Объекты доступа к данным (DAO, Data Access Objects) создавались как объектно-ориентированный интерфейс для ядра баз данных Jet фирмы Microsoft. В ранних версиях Microsoft Office доступ к данным обеспечивался исключительно ядром Jet. Сначала ядро Jet могли использовать только Access и Visual Basic, но с выходом па-

кета Microsoft Office 95 ядро Jet стало сервером Automation, что позволило использовать DAO для доступа к данным из любого клиента Automation. В версии Office 97 компоненты Excel, Word и PowerPoint стали клиентами Automation и вышла новая версия DAO 3.5. В DAO 3.5 появился новый режим доступа к данным ODBCDirect, который позволяет манипулировать данными в обход ядра Jet, напрямую связываясь с источниками данных ODBC по технологии "клиент-сервер". За счет прямого доступа к данным получается выигрыш в производительности, при этом конечный пользователь имеет стандартный объектно-ориентированный интерфейс доступа к различным типам данных, начиная от баз данных Access до баз данных ISAM (Indexed Sequential Access Method, индексно-последовательный метод доступа) и SQL. Отсюда можно заключить, что DAO — это унифицированный набор объектов для доступа к данным.

В Microsoft Office 2000 вошли новая версия DAO 3.6 и новая версия Microsoft Jet 4.0, в которой реализована поддержка Unicode, т. е. в базах данных появилась возможность хранить символы любых национальных алфавитов одновременно. В версии Microsoft Office Access 2003 используются те же версии Jet и DAO.

DAO версии 3.5 и выше включает две объектные модели, соответствующие двум типам "рабочей области" (сеанс работы с базой данных), в зависимости от того, используется ли ODBCDirect. При доступе к данным с помощью ядра Microsoft Jet

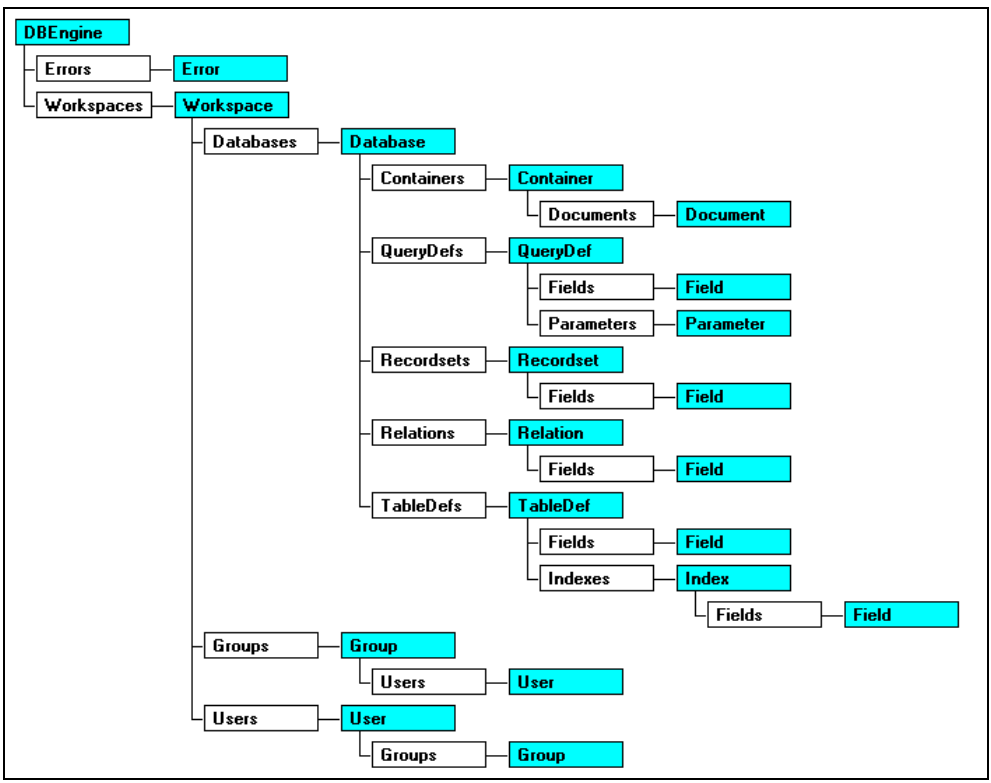


Рис. 13.6. Модель объектов доступа к данным для рабочей области Microsoft Jet

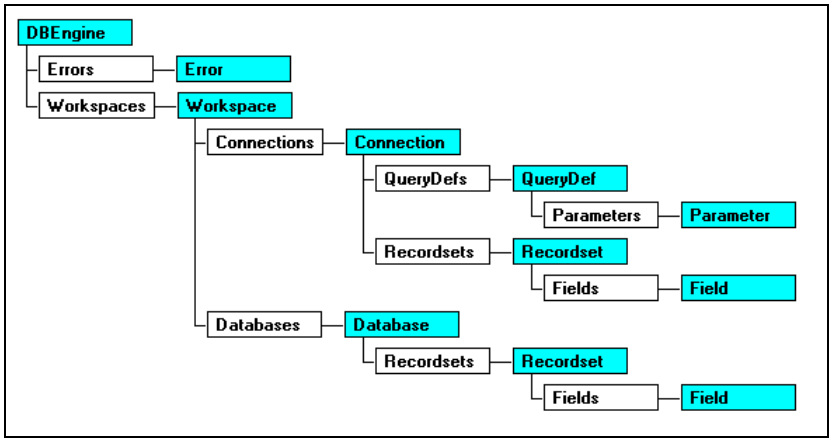


Рис. 13.7. Модель объектов доступа к данным для рабочей области ODBCDirect

применяется рабочая область Microsoft Jet. В случае прямого доступа к источнику данных ODBC используется рабочая область ODBCDirect. Тип рабочей области и связанного с ней источника данных определяет, какие объекты, свойства и методы DAO можно использовать (рис. 13.6 и 13.7).

Используйте рабочую область Microsoft Jet для доступа к базам данных Jet (таким, как MDB-файлы Access) или другим настольным базам данных ISAM или в том случае, когда требуется использовать уникальные средства ядра Jet, такие как объединение данных в разных форматах.

Применяйте в клиентском приложении рабочую область ODBCDirect вместо рабочей области Jet в следующих случаях:

- если нужно только выполнить запрос или хранимую процедуру на сервере баз данных, таком как SQL Server;
- когда требуется воспользоваться специфическими возможностями ODBCDirect, такими как пакетное обновление записей (обновленные записи кэшируются на локальном компьютере);
- когда требуется асинхронное выполнение запроса, запрос выполняется в фоновом режиме, не останавливая выполнение других.

В табл. 13.4 описаны объекты доступа к данным. В первом столбце приведены объекты-семейства, а во втором — объекты, являющиеся элементами соответствующего семейства, в третьем — описание объекта. В последующих главах книги вы встретитесь с примерами использования объектной модели DAO.

Таблица 13.4. *Описание объектов доступа к данным*

Семейство	Объект	Описание
`Connections`	`Connection`	Предоставляет информацию о соединении с источником данных ODBC (используется только в рабочей области ODBCDirect)

Таблица 13.4 *(окончание)*

Семейство	Объект	Описание
Containers	Container	Хранилище информации об объекте предопределенного типа (используется только в рабочей области Microsoft Jet)
Databases	Database	Открытая база данных
–	DBEngine	Главный объект ядра баз данных Microsoft Jet
Documents	Document	Информация о сохраненном объекте предопределенного типа (используется только в рабочей области Microsoft Jet)
Errors	Error	Информация об ошибках, ассоциированных с данным объектом
Fields	Field	Представляет поле (столбец) таблицы, запроса, индекса, поле связи между таблицами или запросами или поле набора записей
Groups	Group	Группа пользователей, определяющая права доступа к данным (используется только в рабочей области Microsoft Jet)
Indexes	Index	Определяет порядок и уникальность значений в таблице (используется только в рабочей области Microsoft Jet)
Parameters	Parameter	Параметр для параметризованного запроса
Properties	Property	Встроенная или определенная пользователем характеристика (свойство)
QueryDefs	QueryDef	Описание хранимого в базе данных запроса
Recordsets	Recordset	Набор записей в базовой таблице или запросе
Relations	Relation	Связь между полями таблиц или запросов (используется только в рабочей области Microsoft Jet)
TableDefs	TableDef	Описание хранимой в базе данных таблицы (используется только в рабочей области Microsoft Jet)
Users	User	Бюджет пользователя, определяющий права доступа к данным (используется только в рабочей области Microsoft Jet)
Workspases	Workspace	Сеанс работы с источником данных с помощью ядра баз данных Microsoft Jet

Модель объектов ActiveX для управления данными

Объекты управления данными ActiveX (ActiveX Data Objects, ADO) предназначены для обеспечения доступа к источникам данных разных видов, от текстовых файлов до распределенных баз данных. ADO представляет собой объектно-ориентированный интерфейс для приложений, использующих OLE DB. OLE DB — это про-

граммный интерфейс, удовлетворяющий спецификации COM (Component Object Model, компонентная модель объектов), который предоставляет унифицированный способ доступа к различным источникам данных. Интерфейс OLE DB разрабатывался с целью получения оптимальной функциональности для приложений разных видов и поэтому не является простым в использовании. ADO — промежуточное звено между приложением и OLE DB, предоставляющее разработчикам приложений удобный объектно-ориентированный интерфейс.

Существует три объектных модели, которые вместе обеспечивают те функции работы с данными, которые реализуются объектами DAO:

- Microsoft ActiveX Data Objects 2.6 (ADODB);
- Microsoft ActiveX Data Objects Extensions for DDL and Security 2.6 (ADOX);
- Microsoft Jet and Replication Objects 2.6 (JRO).

Замечание

Компоненты доступа к данным ADO используются не только в Microsoft Office, поэтому устанавливаться могут разными приложениями. Есть также отдельный компонент доступа к данным — MDAC, который имеет несколько версий. Устанавливая этот компонент на компьютер, вы можете получить соответствующую версию библиотеки ADO. На момент выхода книги последняя версия имеет номер 2.7.

Стандартный набор функций для работы с данными распределяется между этими тремя моделями следующим образом:

- *Управление данными.* Объектная модель ADODB обеспечивает вашему приложению доступ к источнику данных и возможность отбирать для просмотра и изменять эти данные. Однако она не предоставляет возможности изменять структуру таблиц и других объектов источника данных.
- *Определение данных и защита.* Объектная модель ADOX позволяет программно изменять структуру объектов в источнике данных, в частности добавлять и удалять таблицы, изменять структуру таблиц, создавать и изменять запросы, представления и хранимые процедуры, а также управлять схемой защиты данных: создавать группы пользователей и отдельных пользователей, наделять их требуемыми правами доступа к объектам.
- *Репликация и службы ядра базы данных Jet.* Объектная модель JRO позволяет создавать, модифицировать и синхронизировать реплики. Она была спроектирована специально для доступа к объектам Jet и, в отличие от моделей ADODB и ADOX, может работать только с базами данных Access.

Иерархическая структура моделей объектов ADO представлена на рис. 13.8 и 13.9.

Краткое описание объектов моделей ADO приведено в табл. 13.5—13.7.

Таблица 13.5. *Объектная модель ADODB*

Объект	Тип	Описание
`Connection`	Объект	Открывает сеанс обмена данными
`Command`	Объект	Представляет собой инструкцию SQL

Таблица 13.5 *(окончание)*

Объект	Тип	Описание
`Recordset`	Объект	Представляет собой набор записей и позволяет осуществлять навигацию по записям и манипулировать с данными в нем
`Record`	Объект	Представляет собой каталог или файл
`Stream`	Объект	Представляет собой содержимое файла
`Errors`	Семейство	Объекты `Error` в этом семействе создаются в ответ на одну ошибку, произошедшую во время сеанса связи. Каждый объект представляет собой информацию о возникшей ошибке
`Parameters`	Семейство	Содержит все объекты `Parameter`, ассоциированные с объектом `Command`. Каждый объект представляет собой параметр инструкции SQL
`Fields`	Семейство	Содержит все объекты `Field`, ассоциированные с набором записей `Recordset`. Каждый объект представляет собой поле (столбец) в наборе записей `Recordset`
`Properties`	Семейство	Содержит все объекты `Property`, ассоциированные с объектом `Connection`, `Command`, `Recordset` или `Field`. Каждый объект представляет характеристику (свойство) любого объекта ADO

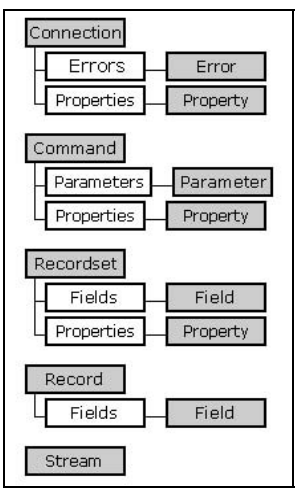


Рис. 13.8. Объектная модель ADO

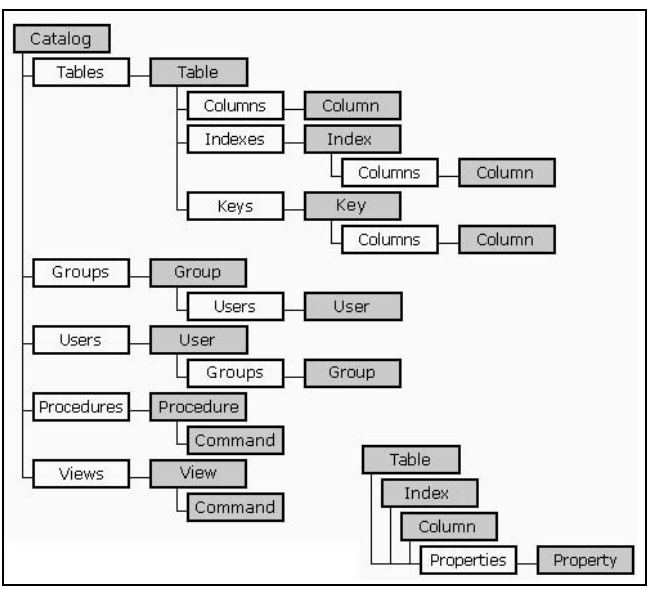


Рис. 13.9. Объектная модель ADOX

Таблица 13.6. *Объектная модель ADOX*

Объект	Тип	Описание
`Catalog`	Объект	Главный объект этой модели. Ссылается на источник данных и обеспечивает доступ ко всем объектам источника — таблицам, представлениям, процедурам, группам, пользователям
`Tables`	Семейство	Семейство таблиц, содержащихся в источнике данных. Каждый объект `Table` этого семейства ссылается на одну таблицу
`Indexes`	Семейство	Содержит все индексы таблицы. Каждый объект `Index` семейства ссылается на один из индексов
`Keys`	Семейство	Содержит все ключи таблицы. Каждый объект `Key` семейства ссылается на один из ключей
`Columns`	Семейство	Содержит объекты `Column`, которые ссылаются на столбцы в одном из объектов `Table`, `Index`, `Key`
`Groups`	Семейство	Содержит все объекты `Group` каталога или пользователя. Каждый из объектов `Group` ссылается на бюджет группы в каталоге или пользователе
`Users`	Семейство	Содержит объекты `User`, которые представляют собой бюджеты пользователей, имеющих права доступа к защищенной базе данных
`Procedures`	Семейство	Содержит все хранимые процедуры в базе данных. Каждый объект `Procedure` семейства ссылается на одну из хранимых процедур
`Views`	Семейство	Содержит все представления (`View`) в базе данных

Таблица 13.7. *Описание объектов JRO*

Объект	Тип	Описание
`JetEngine`	Объект	Обеспечивает доступ к двум сервисам ядра базы данных Jet: сжатие базы данных и выгрузка данных из кэша в файл базы данных MDB
`Replica`	Объект	Представляет собой копию реплицированной базы данных
`Filters`	Семейство	Содержит набор объектов `Filter`, каждый из которых представляет собой набор записей, которые должны реплицироваться

Более подробное описание объектов JRO и их использования приведено в *гл. 18.*

В случае разработки многоуровневого клиент-серверного приложения для доступа к данным используется еще одна объектная модель — RDS (Remote Data Service,

служба удаленного доступа к данным). Она включает три объекта, краткое описание которых приведено в табл. 13.8.

Таблица 13.8. *Объектная модель RDS*

Объект	Тип	Описание
`RDS.DataSpace`	Объект	Этот объект содержит метод, позволяющий создать объект (так называемый бизнес-объект) промежуточного сервера передачи данных. В качестве промежуточного сервера может выступать процедура, указанная пользователем, встроенная процедура, используемая по умолчанию, или библиотека динамической компоновки (DLL), исполняемая в локальной сети, в Интернете или интранете
`RDSServer.DataFactory`	Объект	Представляет собой встроенную программу — сервер данных, которая выполняет обработку данных и возвращает результат обработки
`RDS.DataControl`	Объект	Этот объект может использоваться для следующих целей: автоматическое выполнение функций объектов `RDS.DataSpace` и `RDSServer.DataFactory`, выполнение обработки данных или возвращение результата с помощью встроенных служб RDS, представление результата обработки данных в виде совокупности значений для визуальных элементов управления (в форме)

Объектная модель Visual Basic для приложений

Объектная модель Visual Basic для приложений представляет собой совокупность независимых объектов, объединенных в одну библиотеку с названием VBA, которые используются всеми приложениями семейства Microsoft Office. Семейства и объекты этой библиотеки представлены в табл. 13.9.

Таблица 13.9. *Описание объектов Visual Basic для приложений*

Объект	Тип	Описание
`Collection`	Объект из библиотеки VBA	Упорядоченная совокупность объектов, с которой можно обращаться как с единым объектом
`Debug`	Объект	Позволяет выводить текущую информацию в окно отладки непосредственно во время выполнения кода на VBA
`Dictionary`	Объект из библиотеки Scripting	Объект-пара: ключ и элемент. Представляет собой аналог элемента ассоциативной памяти

Таблица 13.9 *(окончание)*

Объект	Тип	Описание
`Drives`	Семейство из библиотеки Scripting	Содержит объекты `Drive`, предоставляющие информацию (только для чтения) обо всех доступных дисках. Является свойством объекта `FileSystemObject`. Каждый объект предоставляет доступ к свойствам конкретного локального или сетевого диска
`Err`	Объект из библиотеки VBA	Предназначен для обработки ошибок Automation и ошибок модулей VBA во время выполнения кода на VBA
`Files`	Семейство из библиотеки Scripting	Содержит объекты `File` и представляет собой совокупность всех файлов в данной папке. Является свойством объекта `FileSystemObject`. Объект `File` предоставляет доступ ко всем свойствам файла на диске
`FileSystemObject`	Объект из библиотеки Scripting	Предоставляет доступ к файловой системе компьютера
`Folders`	Семейство из библиотеки Scripting	Содержит объекты `Folder` и представляет собой совокупность всех папок внутри данной папки. Является свойством объекта `Folder` (свойство называется `SubFolders`). Каждый объект `Folder` предоставляет доступ ко всем свойствам папки на диске
`TextStream`	Объект из библиотеки Scripting	Обеспечивает последовательный доступ к текстовому файлу
`UserForms`	Семейство из библиотеки VBA	Содержит объекты `Object`, соответствующие объектам `UserForm`, и представляет собой совокупность пользовательских форм, загруженных в данный момент в приложение. Это семейство является свойством объекта `Global` из библиотеки VBA

Объектная модель Microsoft Office 2003

Иерархия объектов Microsoft Office 2003 представлена на рис. 13.10 и 13.11.

Объекты Microsoft Office используются в иерархической структуре других объектных моделей. В объектной модели Microsoft Office 2003 появилось много новых объектов. Однако эти новые объекты могут использоваться другими приложениями Microsoft Office 2003: Word, Excel, PowerPoint, поэтому в описании объектов библиотеки Microsoft Office, приведенном в табл. 13.10, мы даем только краткую справку по этим объектам.

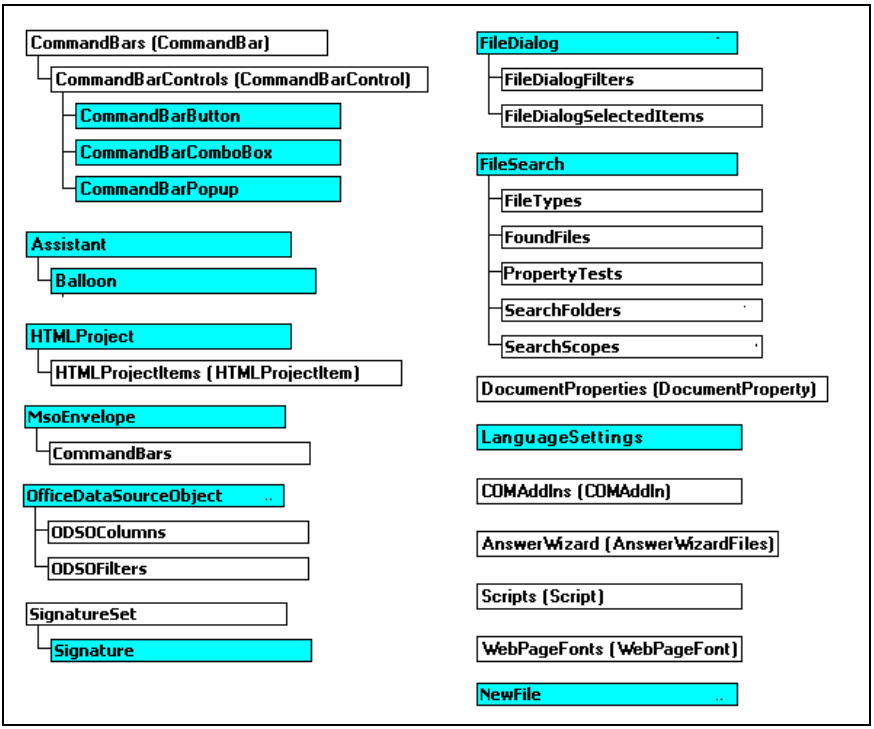


Рис. 13.10. Объектная модель Microsoft Office 2003

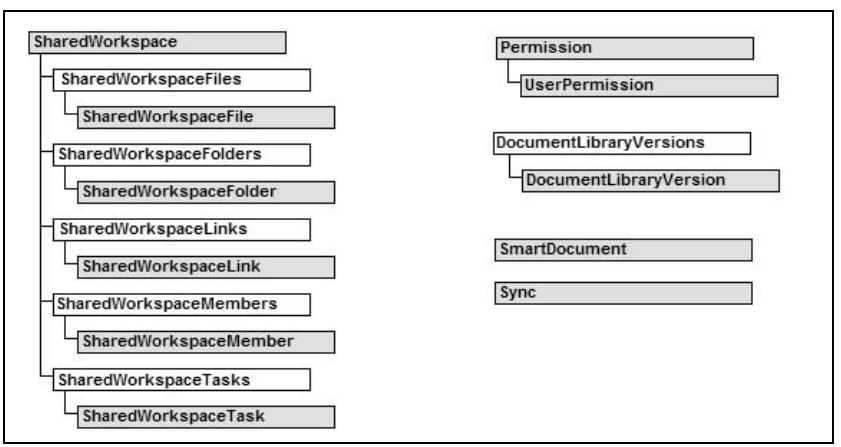


Рис. 13.11. Новые объекты модели Microsoft Office 2003

Таблица 13.10. *Описание объектов Microsoft Office 2003*

Объект	Тип	Описание
`AnswerWizard`	Объект	Соответствует Мастеру ответов приложения Microsoft Office. Все приложения семейства Microsoft Office имеют только один Мастер ответов, поэтому все изменения над этим объектом немедленно отразятся на активном приложении
`Assistant`	Объект и семейство	Ссылается на объект, представляющий активного в данный момент Помощника Microsoft Office
`COMAddIns`	Семейство	Содержит объекты `COMAddIn`, соответствующие надстройкам COM, зарегистрированным в реестре Windows
`CommandBars`	Семейство	Содержит объекты `CommandBar`, соответствующие панелям команд приложения-контейнера (это могут быть строки меню, панели инструментов, меню и контекстные меню)
`DocumentProperties`	Семейство	Содержит объекты `DocumentProperty`, каждый из которых представляет собой набор встроенных или определенных пользователем характеристик (свойств) документа-контейнера
`FileDialog`	Объект	Реализует диалоговое окно, аналогичное окнам **Открыть** (Open) и **Сохранить** (Save) Microsoft Office
`FileDialogFilters`	Семейство	Содержит объекты `FileDialogFilter`, которые представляют собой фильтры, определяющие, какие файлы будут отображаться в диалоговом окне `FileDialog`
`FileDialogSelectedItems`	Семейство	Семейство строковых значений, представляющих собой пути к файлам или папкам, выбранным в диалоговом окне `FileDialog`
`FileSearch`	Объект	Реализует функциональность стандартного диалогового окна открытия файла
`FileTypes`	Семейство	Семейство значений типов файлов, которые возвращаются методом `Execute` диалогового окна `FileDialog`
`HTMLProject`	Объект	Представляет собой корень дерева проекта Web-страницы, отображаемого в окне просмотра проекта **Project Explorer** редактора сценариев Microsoft Script Editor. Этот объект содержит семейство `HTMLProjectItems` объектов `HTMLProjectItem`, ссылающихся

Таблица 13.10 *(продолжение)*

Объект	Тип	Описание
(прод.)		на элементы иерархической структуры проекта. С помощью этого объекта можно управлять проектом Web-страницы, например открыть его в редакторе сценариев
`LanguageSettings`	Объект	Предоставляет информацию о языковых настройках приложения семейства Microsoft Office
`MsoEnvelope`	Объект	Обеспечивает возможность отправки документов по электронной почте
`NewFile`	Объект	Содержит список строковых значений, которые отображаются в группе **Создание** (New) на панели задач
`ODSOColumns`	Семейство	Объекты этого семейства представляют собой поля в файле для слияния с документом Word
`ODSOFilters`	Семейство	Объекты этого семейства представляют собой фильтры, которые применяются к файлу — источнику данных для документа слияния
`OfficeDataSourceObject`	Объект	Представляет собой источник данных для операции слияния
`SearchFolders`	Семейство	Содержит набор объектов, представляющих собой папки, в которых ведется поиск при использовании объекта `FileSearch`
`SignatureSet`	Семейство	Содержит набор объектов `Signature`, каждый из которых представляет собой цифровую подпись, связанную с документом
`WebPageFonts`	Семейство	Содержит набор объектов `WebPageFont`, каждый из которых соответствует одному из поддерживаемых национальных стандартов. Каждый объект определяет шрифты и их размеры, которые используются по умолчанию при сохранении документа в формате Web-страницы
`Scripts`	Семейство	Содержит объекты `Script`, соответствующие сценариям, определенным в документе Word, электронной таблице Excel или слайде PowerPoint. С помощью этого объекта можно управлять сценариями документа, например создавать новые сценарии
`DocumentLibraryVersions`	Новое семейство	Представляет набор версий активного документа Word или книги Excel. Используется для управления версиями документов, хранимых в библиотеке Windows SharePoint Services на сервере

Таблица 13.10 *(окончание)*

Объект	Тип	Описание
`Permission`	Новый объект	Позволяет определить ограничения на доступ к документам Word, Excel, Power Point
`UserPermission`	Новый объект	Определяет разрешения конкретного пользователя для активного документа
`SharedWorkspace`	Новый объект	Позволяет добавить активный документ в рабочее пространство документов Microsoft Windows SharePoint Services на сервере
`SharedWorkspaceFiles`	Новое семейство	Представляет набор файлов в текущем рабочем пространстве
`SharedWorkspaceFolders`	Новое семейство	Представляет набор папок в текущем рабочем пространстве
`SharedWorkspaceLinks`	Новое семейство	Представляет набор URL-ссылок в текущем рабочем пространстве
`SharedWorkspaceMembers`	Новое семейство	Представляет набор пользователей, имеющих права на совместную работу с документом (-ами) в текущем рабочем пространстве
`SharedWorkspaceTasks`	Новое семейство	Представляет набор задач, связанных с пользователем, работающим над документом в текущем рабочем пространстве
`Sync`	Новый объект	Управляет синхронизацией локальной и серверной копии документа
`SmartDocument`	Новый объект	Управляет пакетом расширений XML, присоединенным к активному документу Word или Excel

Использование объектов и семейств

Несмотря на разнообразие библиотек объектов, методы работы с объектами в процедурах VBA общие и не зависят от того, к какой библиотеке относится конкретный объект. В данном разделе мы как раз и рассмотрим эти общие методы.

Ссылки на объекты

Первый вопрос, который следует обсудить: "Как обратиться к объекту?" Для обращения к объекту существует одно общее правило: нужно проследить путь в иерархии объектов, начиная от объекта самого верхнего уровня до целевого объекта, и записать последовательность имен встреченных на пути объектов или семейств, отделяя их друг от друга точкой. Если на пути встречается семейство, то, кроме имени семейства, необходимо в скобках указать индекс или имя его элемента, т. е. задать

конкретный объект в семействе. Например, чтобы обратиться к форме, входящей в состав семейства `AllForms`, необходимо написать следующее выражение:

```
Application.CurrentProject.AllForms("Заказы клиента")
```

Таким образом, чтобы правильно создавать *ссылки* на объекты, нужно достаточно хорошо представлять себе место каждого объекта в иерархии объектов модели. Такие длинные ссылки особенно характерны для моделей, которые имеют многоуровневую, иерархическую структуру, например DAO. На самом деле, на практике чаще применяют не полные, а сокращенные ссылки. Дело в том, что наиболее часто используемые семейства, объекты, свойства и методы считаются *глобальными*. Ссылки на них хранятся в специальном объекте с именем `Global`. Для обращения к глобальному объекту можно пропустить объекты более высокого уровня.

Например, семейство `Forms` является глобальным. Для доступа к объекту этого семейства можно использовать сокращенную ссылку вида:

```
Forms("Заказы")
```

вместо полной ссылки:

```
Application.Forms("Заказы")
```

Замечание

Узнать, какие объекты, свойства и методы являются глобальными, позволяет окно просмотра объектов. Чтобы отобразить список глобальных компонентов объектной модели, выберите элемент **<globals>** в списке **Classes** в окне просмотра объектов *(см. разд. "Использование окна просмотра объектов " данной главы)*.

Кроме указанного способа ссылки на объект, входящий в состав семейства, существуют и другие способы. При любом способе обращения к объекту возвращается *ссылка на объект*, находящийся в памяти.

Обычно имя семейства и имя объекта разделяются оператором ! (восклицательный знак), например:

```
Forms!Товары
```

Если имя объекта состоит из нескольких слов, разделенных пробелом, тогда вам придется написать:

```
Forms![Заказы клиента]
```

Третий способ ссылки состоит в том, что на объект в семействе ссылаются не по названию, а по индексу, например:

```
Properties(0)
```

Такой способ применяется обычно тогда, когда имя объекта неизвестно. В то же время каждый объект в семействе имеет индекс (порядковый номер), который обычно начинается с нуля. Есть, правда, исключения из этого правила, например в семействе `CommandBars` модели Microsoft Office нумерация объектов начинается с единицы. Поэтому перед использованием индексов лучше узнать о способе нумерации в справочной системе Access.

И, наконец, последний способ ссылки на объект состоит в использовании вместо имени объекта строковой переменной, например:

```
Reports(strИмяОтчета)
```

Если какой-то объект содержит несколько семейств объектов более низкого уровня, то одно из этих семейств, обычно наиболее часто используемое, считается стандартным или семейством по умолчанию. К объекту, входящему в это семейство, можно обращаться, не указывая имени семейства. Например, для объекта `TableDef` (таблица) модели DAO стандартным семейством считается `Fields` (поля). Поэтому для обращения к столбцу таблицы можно использовать сокращенную ссылку

```
TableDefs!Должности!КодДолжности
```

вместо полной ссылки

```
TableDefs!Должности.Fields!КодДолжности
```

Еще одним способом сокращения ссылки на объект является использование *объектной переменной*. Особенно полезно применять объектную переменную, если к объекту нужно обратиться несколько раз. *(О том, как объявить объектную переменную и установить ссылку на объект см. в разд. "Язык Visual Basic для приложений (VBA)" данной главы.)*

Совет

Использование сокращенных ссылок в программах предпочтительнее еще и потому, что в этом случае программы будут выполняться быстрее.

Работа с объектами в семействах

Доступ к объектам, входящим в семейство, в большинстве случаев возможен только через упоминание имени семейства, поэтому в программах VBA приходится выполнять различные операции с семействами. Например, иногда, чтобы обратиться к объекту в семействе, нужно перебрать несколько или даже все объекты семейства. Допустим, требуется проверить, открыта ли определенная форма в нашем приложении. Для этого мы можем написать функцию, которая должна проверить, входит ли эта форма в семейство `Forms`. Данная функция может иметь следующий вид:

```
Function IsLoaded (strFormName As String) As Boolean
'Возвращает значение True, если форма открыта, и False, если нет
    Dim frm As Form
    IsLoaded = False
    For each frm in Forms
        If frm.Name = FormName Then
            IsLoaded = True
            Exit Function
        EndIf
    Next frm
End Function
```

В данной функции мы сначала объявили объектную переменную типа `Form` и установили начальное значение функции. Затем мы организуем цикл, в котором пере-

бираем элементы семейства Forms до тех пор, пока не обнаружим в нем объект Form с именем, совпадающим со строковой переменной, которая является аргументом функции. Если такой объект найден, мы присваиваем функции значение True и выходим из нее. Если нет, то когда будут перебраны все элементы семейства Forms, функция вернет значение False.

Для того чтобы организовать цикл с перебором элементов, количество которых неизвестно, в настоящем примере мы использовали оператор For Each...Next. Этот оператор обычно применяется при работе с семействами. Однако мы могли организовать такой же цикл, используя обычный оператор For...Next, т. к. любое семейство (в том числе и Forms) имеет свойство Count, которое возвращает количество элементов в семействе. Далее приводится другой вариант организации этого цикла:

```
For I = 0 To Forms.Count - 1
    If frm(I).Name = FormName Then
        IsLoaded = True
        Exit Function
    EndIf
Next I
```

Кроме указанного свойства, многие (но не все) семейства объектов доступа к данным имеют два метода, которые позволяют добавлять объекты в семейство и удалять их из семейства. Для добавления объекта применяют методы Append или Add, для удаления — метод Delete. Это обеспечивает оперативное создание объектов доступа к данным, например, временных таблиц. Чтобы создать таблицу программным путем, вы должны не только определить эту таблицу, но и добавить ее в соответствующее семейство. Далее приводится фрагмент кода процедуры, в котором создается новая таблица, определяются два ее поля и добавляются объекты в семейства Fields и TableDefs:

```
'Объявляем объектные переменные для объектов: база данных, таблица и поле
Dim db As Database, td As TableDef, fld As Field
'Устанавливаем ссылку на текущую базу данных
Set db = CurrentDb
'Создаем новую таблицу, используя метод CreateTableDef объекта Database
Set td = db.CreateTableDef("Временная")
'Создаем поле в таблице, используя метод CreateField объекта TableDef
'Поле будет иметь имя "Дата" и тип Дата/время
Set fld = td.CreateField("Дата",dbDate)
'Добавляем поле "Дата" в семейство Fielfs таблицы
td.Fields.Append fld
'Создаем второе поле с именем "Сумма" и типом Денежный
Set fld = td.CreateField("Сумма",dbCurrency)
'Добавляем поле "Сумма" в семейство Fielfs таблицы
td.Fields.Append fld
'Добавляем таблицу к семейству TableDefs базы данных
db.TableDefs.Append td
'Обновляем семейство TableDefs
db.TableDefs.Refresh
```

После выполнения этой программы на вкладке **Таблицы** (Tables) окна базы данных появится новая таблица `Временная`.

Удалить эту таблицу можно аналогичным способом, только уже воспользовавшись методом `Delete` семейства `TableDefs`.

```
db.TableDefs.Delete "Временная"
db.TableDefs.Refresh
Set db = Nothing
```

Метод `Refresh` обновляет количество объектов семейства после добавления или удаления объектов. В последнем предложении мы освобождаем объектную переменную `db`.

Замечание

К сожалению, семейства в модели Microsoft Office Access не имеют аналогичных методов. Это означает, что вы не можете, например, оперативно добавлять или удалять поля в формах или отчетах. Если в зависимости от условий в форме должны отображаться или не отображаться некоторые поля, приходится создавать элементы управления для всех возможных полей, а затем использовать свойство `Visible` соответствующих элементов управления, чтобы поля либо были видимы, либо не видимы. В семействах объектной модели Microsoft Office аналогичные свойства есть: `Add` и `Delete`. Это позволяет оперативно управлять, например, кнопками на панелях команд.

Свойства и методы объектов

Как уже говорилось ранее, каждый объект имеет свойства, которые являются его характеристиками, и методы, которые позволяют управлять поведением этого объекта. То же справедливо и для семейств. Работа с объектами и семействами заключается в установке или получении значений конкретных свойств объекта или семейства и вызове их методов. Поэтому для управления объектами приложения необходимо хорошо знать свойства и методы каждого объекта.

Замечание

Если вы хотите иметь подробную информацию обо всех свойствах и методах объекта, следует обратиться к соответствующему разделу справки. Например, чтобы получить сведения о любом объекте модели Access, нужно раскрыть раздел **Microsoft Access Object Model** в папке **Microsoft Access Visual Basic Reference** (этот раздел справки, как и другие, связанные с объектными моделями, не русифицированы) и затем щелкнуть на нужном объекте или семействе. Краткую информацию обо всех свойствах и методах объекта при написании программы в среде редактора VBA можно получить очень быстро, подключив оперативную подсказку или пользуясь окном просмотра объектов.

Установка свойства объекта — это присвоение значения данному свойству. Поэтому для установки свойства используется оператор присваивания, например, чтобы установить свойство `Visible` (Вывод на экран) элемента управления формы, можно использовать инструкцию VBA:

```
Forms!Товары!КодТовара.Visible = False
```

или

```
frm!КодТовара.Visible = False
```

Таким образом, ссылка на свойство объекта состоит из двух частей, разделенных точкой: ссылки на объект (любого вида) и имени свойства.

Получить свойство означает прочитать текущее значение этого свойства. Например, инструкция VBA

```
intCount = Forms.Count
```

присваивает переменной `intCount` значение свойства `Count` семейства `Forms`.

Метод объекта в инструкциях VBA обозначается так же, как и свойство. Это видно и из приведенных в предыдущем разделе примеров, где мы использовали методы `Append`, `Delete` и `Refresh` семейства `TableDefs`. Однако, в отличие от свойств, методы могут иметь аргументы. Например, в следующей инструкции VBA применяется метод `OpenReport` объекта `DoCmd`.

```
DoCmd.OpenReport "Вакансии", acPreview
```

Объект `DoCmd` — это специальный объект, который позволяет в программах VBA выполнять макрокоманды. Имя каждой макрокоманды является методом этого объекта. В приведенном примере создается отчет `Вакансии` в режиме предварительного просмотра. При этом метод `OpenReport` не возвращает никакого значения, и аргументы в данном случае не требуется заключать в скобки.

Если же метод используется как функция, возвращающая значения, как показано в следующем примере, аргументы необходимо заключить в скобки.

```
Dim db As Database, rs As Recordset
Set db = CurrentDB
Set rs = db.OpenRecordset ("Сотрудники")
. . . . . .
rs.Close
db.Close
```

Метод `OpenRecordset` объекта `Database` возвращает ссылку на объект `Recordset` (Набор записей), присваиваемую объектной переменной `rs`. Метод `Close`, который имеют оба объекта — `Recordset` и `Database`, — не требует аргументов.

Модули как объекты Access

Код VBA в приложении Access содержится в модулях. Модули являются объектами Access, такими же как таблицы, запросы, формы, отчеты, страницы и макросы, о чем свидетельствует ярлык на панели объектов в окне **База данных** (Database). Основное содержание модулей — это процедуры на языке VBA. Существует два типа модулей: стандартные модули и модули класса.

Стандартные модули

Стандартные модули содержат общие процедуры, которые не связаны с конкретным объектом: формой или отчетом. Эти процедуры могут вызываться из других модулей

и использоваться при обработке событий в разных объектах, для вычисления значений в разных запросах или формах и т. д. Если в процедурах модуля нет ссылок на конкретные объекты данного приложения (формы, отчеты, элементы управления), то такой модуль может быть с успехом использован другими приложениями Access. Стандартные модули применяются также для объявления глобальных (т. е. доступных из всех модулей приложения) переменных, констант, типов.

Список стандартных модулей приложения всегда можно увидеть, нажав кнопку **Модули** (Modules) в окне **База данных** (Database). В этом окне обычно и выполняется работа с модулями. Выделите в окне базы данных `Борей` (Northwind) модуль `Служебные функции` (Utility Functions) и нажмите кнопку **Конструктор** (Design). Откроется окно редактора кода VBA, в котором можно увидеть содержание модуля. Модуль состоит из трех строк описания и одной процедуры — функции `IsLoaded`, с которой мы уже встречались (рис. 13.12) *(см. гл. 12)*.

Это пример универсальной функции, которая проверяет, загружена ли форма, имя которой передается ей в качестве аргумента. Она никак не связана не только с объектом, но и с данным приложением и может использоваться как в нем самом, так и в любом другом приложении Access.

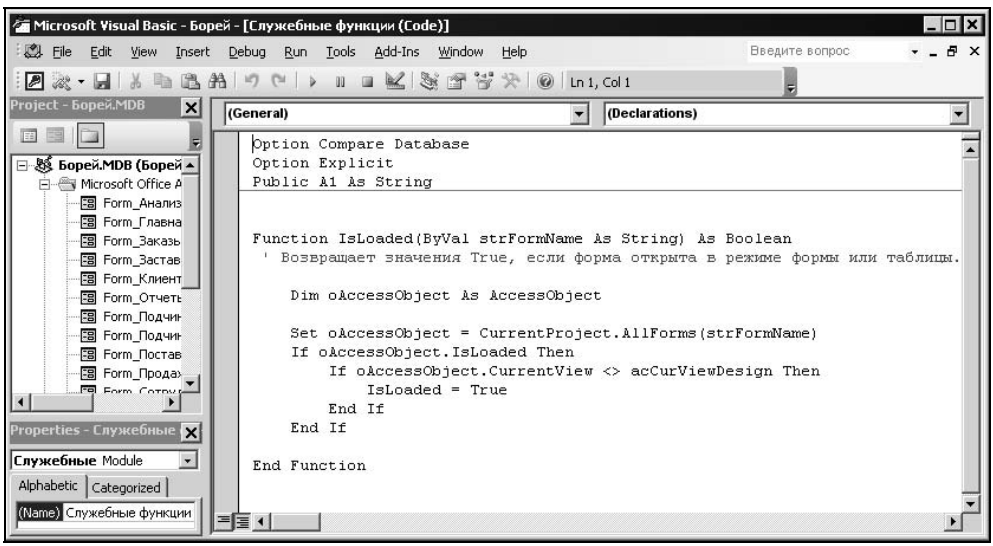


Рис. 13.12. Функция `IsLoaded`

Модули класса

Вторым типом модуля в Access является *модуль класса*. Модуль класса отличается от стандартного модуля тем, что, кроме процедур, он содержит описание объекта и используется для создания объектов. Процедуры, определенные в этом модуле, являются методами и свойствами объекта. Примерами модулей класса являются модули форм и отчетов.

Модули форм и отчетов связаны с конкретной формой или отчетом и содержат процедуры обработки событий для этой формы или отчета. Модуль формы не создается сразу при создании новой формы. Он создается и связывается с формой, как только вы попытаетесь создать первую процедуру обработки событий для этой формы или одного из элементов управления формы или же нажмете кнопку **Программа** (Code) в окне Конструктора формы.

Чтобы убедиться в этом, откройте любую форму приложения Борей (Northwind) в режиме Конструктора и посмотрите свойства этой формы. На вкладке **Другие** (Others) есть свойство **Наличие модуля** (Has Module). Оно должно иметь значение **Нет** (No). После нажатия кнопки **Программа** (Code), которая служит для открывания редактора кода VBA, значение этого свойства изменится на **Да** (Yes) (рис. 13.13).

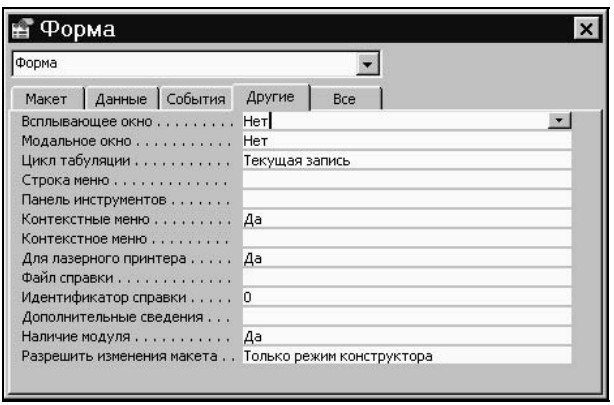


Рис. 13.13. Диалоговое окно свойств формы

В окне редактора VBA отображается объект Form (Форма), а справа — поле со списком событий, которые могут обрабатываться процедурами VBA (рис. 13.14).

Если в модуле для некоторого события существует процедура, то это событие выделяется в списке жирным шрифтом. В данном случае таких событий нет, т. к. обработка событий в форме Клиенты (Customers), которую мы открыли, выполняется с помощью макросов. При попытке открытия редактора VBA автоматически создается модуль формы, который будет иметь название Form_Клиенты (Form_Customers).

Если теперь закрыть окно редактора кода, а затем форму, то, хотя мы не вносили никаких изменений в форму, на экране появится вопрос: **Сохранить изменения макета или структуры формы Клиенты?** (Do you want to save changes to the design of form 'Customers'?) Вопрос связан с появлением модуля формы. Если вы ответите **Нет** (No), модуль не будет сохранен.

Для того чтобы из программы VBA открыть форму и изменить какие-либо свойства формы или ее элементов управления, можно использовать два метода:

- Макрокоманду OpenForm, как метод объекта DoCmd, например:

```
DoCmd.OpenForm "Товары"
Forms!Товары.RecordSource = "Товары клиента"
```

- Ссылку на соответствующий модуль класса, например:

```
Form_Товары.Visible = True
Form_Товары.RecordSource = "Товары клиента"
```

В том и в другом случае при выполнении открывается *стандартный* экземпляр формы `Товары` (Products) и подменяется источник записей для этой формы.

VBA позволяет открывать сразу несколько экземпляров класса формы. Это может потребоваться, например, когда хотят показать в разных окнах головную организацию и дочернюю фирму. Информация о той и другой может храниться в разных записях одной таблицы, и вам необходимо открыть два экземпляра одной формы с разными записями в каждом экземпляре.

Класс формы может иметь только один стандартный экземпляр, поэтому в данном случае придется создать нестандартный экземпляр формы. Для этого в программе нужно описать переменную, типом которой является имя модуля класса формы, при этом в описание переменной включить ключевое слово `New`. Например, следующая команда создает новый экземпляр формы `Клиенты` (Customers) и связывает его с переменной типа `Form`:

```
Dim frm As New Form_Клиенты
```

Теперь изменим источник данных для этой формы, расположение на экране (иначе формы просто перекроются) и присвоим свойству **Вывод на экран** (Visible) значение **Да** (Yes):

```
frm.RecordSource = "Наша организация"
DoCmd.MoveSize 1440, 2400
frm.Visible = True
```

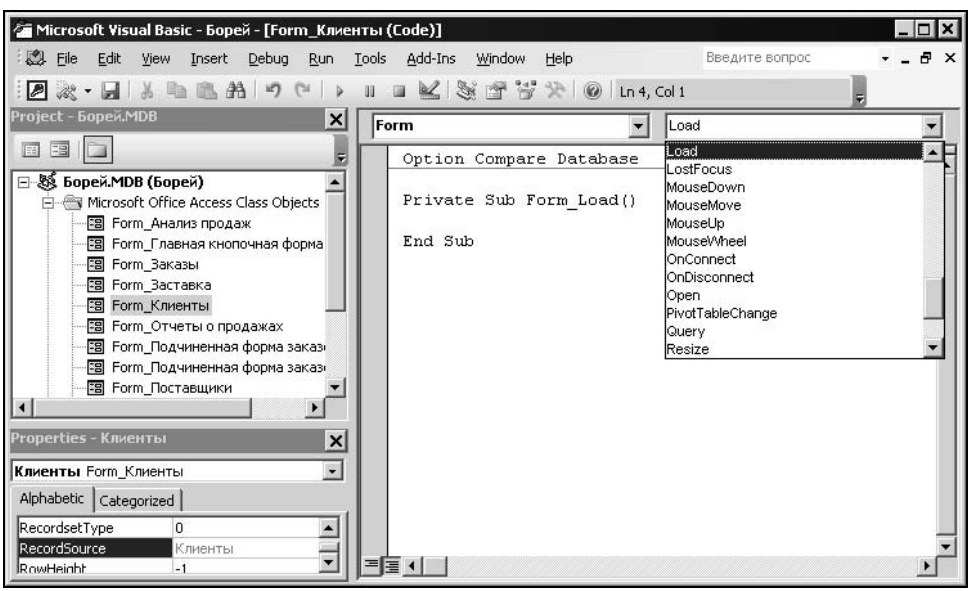


Рис. 13.14. Окно редактора кода VBA

Для изменения положения на экране используется макрокоманда `MoveSize` (СдвигРазмер), в которой в данном случае задаются координаты верхнего левого угла окна формы относительно главного окна Access.

Замечание

Для того чтобы второй экземпляр формы не исчез с экрана сразу после появления, переменная `frm` должна быть описана на уровне модуля, т. е. объявлена в разделе `(Declarations)` модуля с помощью ключевого слова `Private`.

Значения свойств формы или элементов управления формы относятся только к текущему экземпляру формы и не сохраняются после того, как он закроется. Это является отличием модуля класса от стандартного модуля: они по-разному хранят данные. В стандартном модуле можно объявить переменную `Public`, и она, будучи установлена, сохраняет свое значение до тех пор, пока явно не будет изменена другой командой в любом другом модуле. В модуле класса данные относятся к одному экземпляру класса, т. е. одному объекту. Они появляются, когда объект создается, и исчезают, когда исчезает объект. Это означает, что переменные `Public` в модуле класса могут быть доступны только до тех пор, пока существует переменная, содержащая ссылку на отдельный экземпляр класса. Это утверждение справедливо также для процедур, объявленных `Public`.

Формы и отчеты являются стандартными классами объектов в Access, однако можно использовать модули класса для создания пользовательских объектов. Имя, под которым сохраняется модуль класса, становится именем специального объекта. Процедуры типа `Sub` и `Function`, определенные в модуле класса, при этом станут методами объекта, а процедуры типа `Property Let`, `Property Get` и `Property Set` — свойствами объекта. Для описания метода, не возвращающего никакого значения, используется процедура `Sub`, а для метода, возвращающего значение, — процедура `Function`. Процедура `Property Get` возвращает значение свойства объекта. Процедура `Property Set` присваивает значение свойству объекта. Процедура `Property Let` устанавливает значение свойства, не принадлежащего объекту.

Более детальные сведения об этих процедурах можно получить из справки Visual Basic в окне редактора кода, набрав в качестве критерия поиска имя процедуры: `Property Let`, `Property Get` и `Property Set`, соответственно.

Точно так же, как и в модуле формы, новый экземпляр объекта создается путем объявления переменной, например:

```
Dim cls As New MyClass
```

После этого обращение к его методам и свойствам осуществляется с помощью данной переменной. Например, если в модуле определена процедура `MyProcedure`, для того чтобы выполнить эту процедуру, используется выражение

```
cls.MyProcedure
```

Создание модулей

Для того чтобы создать стандартный модуль или модуль класса, нужно:

1. Выбрать команду **Модуль** (Module) или **Модуль класса** (Class Module) в меню **Вставка** (Insert) или в списке кнопки **Новый объект** (New Object) выбрать соот-

ветствующий объект (рис. 13.15). При этом откроется редактор кода VBA с пустым окном модуля.

2. Создать необходимые процедуры и описания.
3. Сохранить модуль, нажав кнопку **Сохранить** (Save) на панели инструментов. При этом выдается диалоговое окно **Сохранение** (Save), в котором необходимо ввести имя нового модуля и нажать кнопку **OK**.

После этого новый модуль появляется в списке модулей окна базы данных. Чтобы его открыть, можно нажать кнопку **Конструктор** (Design) окна базы данных. Если у вас открыты форма или отчет в режиме Конструктора, то для того чтобы открыть модуль формы (отчета), следует нажать кнопку **Программа** (Code) на панели инструментов.

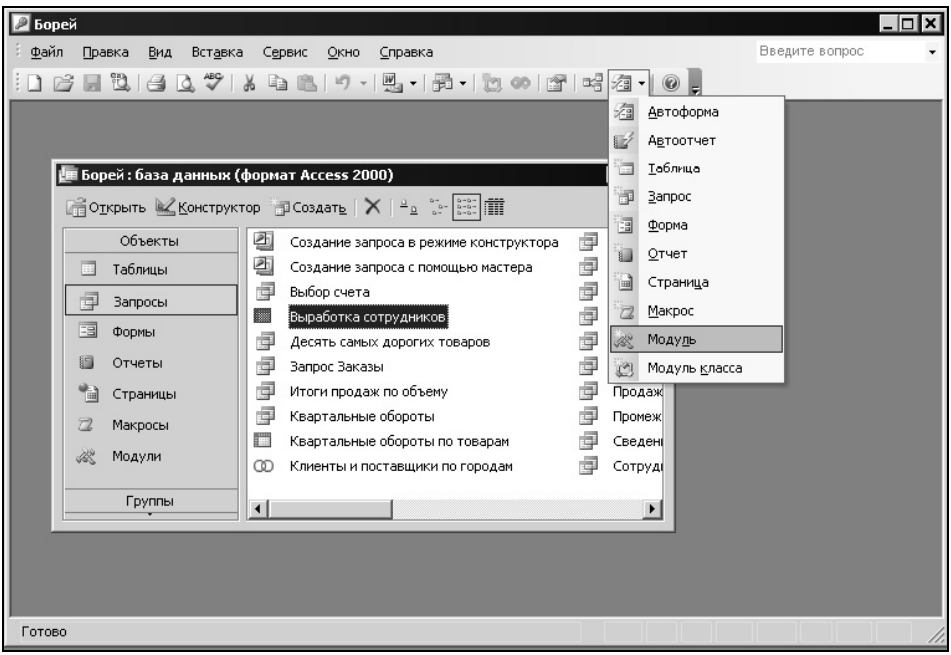


Рис. 13.15. Создание нового модуля

Ссылки на объект *Module*

Для обращения к модулям в программах VBA используется семейство `Modules`, которое содержит все открытые объекты типа `Module`. Для того чтобы открыть объект `Module`, можно использовать макрокоманду `OpenModule` (ОткрытьМодуль). Ссылка на модуль может быть создана тремя способами:

- `имяСемейства!имяОбъекта` — например `Modules!Startup`;
- `имяСемейства ("имяОбъекта")` — например `Modules ("Startup")`;
- `имяСемейства(индекс)`, где `индекс` — индекс объекта в семействе.

Для ссылки на модуль формы или отчета можно использовать или имя модуля, например `Modules!Form_Клиенты`, или свойство формы, например `Forms!Клиенты.Module`.

Объект `Module` имеет свойство `Type`. Значение этого свойства определяет тип модуля: для стандартного модуля используется значение 0, для модуля класса — 1.

Среда программирования VBA

Среда программирования VBA включает все средства, необходимые для написания и отладки процедур VBA. Самым главным компонентом этой среды является редактор кода Visual Basic Editor (VBE), обладающий богатыми возможностями для облегчения процесса ввода программного кода. Кроме того, среда программирования содержит еще массу диалоговых окон, в которых отображается дополнительная информация, требующаяся как в процессе создания, так и в процессе отладки кода.

Окно редактора кода

Для того чтобы открыть окно редактора, достаточно открыть любой модуль Access или нажать комбинацию клавиш <Alt>+<F11>. Окно редактора представлено на рис. 13.14.

Замечание

Даже в локализованных версиях Access окно редактора кода VBA, а также все другие диалоговые окна, составляющие среду программирования VBA, традиционно не локализуются. Поэтому все рисунки в этом разделе главы, а также экранные термины (названий команд меню, диалоговых окон и т. д.) приводятся без перевода.

Обычно в окне редактора используются три панели (три отдельных окна). На самом деле окон может быть и больше и меньше (скоро вы узнаете обо всех). На рис. 13.14 показано расположение только основных окон.

- **Project** (Панель проекта), располагается в верхнем левом углу редактора. В ней отображается иерархическое дерево модулей приложения. Если это окно не активно, выполните команду **View | Project Explorer** либо нажмите комбинацию клавиш <Ctrl>+<R>.
- **Properties** (Панель свойств), находится под панелью проекта. Она позволяет просматривать и изменять свойства различных входящих в проект объектов, отображаемых на панели проекта. Если это окно не отображается, выполните команду **View | Properties Window** либо нажмите клавишу <F4>. Список свойств может отображаться как в алфавитном порядке, так и по категориям.
- **Code** (Панель редактора кода). Это окно занимает большую часть экрана и является "многодокументным", т. е. можно открыть одновременно несколько окон данного типа для разных модулей. Оно представляет собой высокоинтеллектуальный текстовый процессор, существенно облегчающий написание кода VBA.

Проект приложения состоит из модулей, которые делятся на три категории:

- **Microsoft Office Access Class Objects** (Модули классов Access) — включает все модули форм и отчетов;

- **Modules** (Модули) — стандартные модули;
- **Class Modules** (Модули классов) — модули пользовательских классов, если они присутствуют в приложении.

Список объектов в окне проекта является иерархическим, кроме модулей самого проекта, он может включать ссылки на объекты из внешних библиотек и список модулей этих библиотек.

В верхней части панели проекта расположены три кнопки:

- **View Code** (Просмотр кода). При нажатии этой кнопки указатель мыши перемещается в окно кода, в котором отображается код выделенного в иерархии проекта объекта.
- **View Object** (Просмотр объекта). Эта кнопка доступна только, когда в дереве проекта выделен модуль формы или отчета. Тогда она позволяет быстро переключиться на соответствующий модулю объект, т. е. в окно Конструктора формы или отчета.
- **Toogle Folders** (Переключение видов представления дерева). При нажатой кнопке все элементы в дереве проекта группируются в папках по категориям, при отжатой — они отображаются в алфавитном порядке.

Панель свойств позволяет просматривать и изменять свойства различных объектов, входящих в проект. Для отображения свойств объекта его нужно выделить либо в дереве проекта, либо в окне Конструктора формы (отчета). Во многом эта панель дублирует диалоговое окно **Свойства** (Properties) в формах и отчетах. И в том и в другом случае изменение свойств объекта статично, поэтому они могут быть использованы для задания начальных или постоянных свойств объекта.

Чтобы изменить значение свойства объекта, необходимо:

1. Выделить имя свойства в левой колонке.
2. Изменить значение в правой колонке либо введя его вручную, либо путем выбора из списка.

Для каждого модуля, выбранного на панели проекта, открывается свое окно редактора кода. На рис. 13.16 в окне редактора представлен стандартный модуль `Заставка` (Startup) приложения `Борей` (панели проекта и свойств сейчас закрыты).

Код в каждом модуле состоит из разделов, переключение между которыми выполняется путем выбора значения из списка **Object** (Объект), который размещается в левом верхнем углу окна. Список разделов различается для разных типов модулей.

Для модуля формы этот список включает раздел **General** (Общая область), который содержит объявления переменных, констант, специальных типов, внешних процедур. Данный раздел открывается по умолчанию при открытии окна кода. Для стандартного модуля этот раздел единственный — он содержит описание всех процедур. В модуле класса имеются разделы **General** и **Class**. Модуль формы, кроме раздела **General**, содержит раздел самой формы и разделы для каждого элемента управления формы. Модуль отчета — раздел для объекта **Report** (Отчет) и разделы для каждой области отчета. Для полей отчета не определены события, поэтому отсутствуют и соответствующие разделы в модуле кода.

Каждый раздел может содержать несколько процедур, которые можно выбрать из списка **Procedure** (рис. 13.16). Список процедур в стандартном модуле содержит

Declarations (Описания) и имена всех процедур модуля в алфавитном порядке. Список процедур для модуля формы включает в себя перечень всех событий формы и ее элементов. Для модуля класса список процедур содержит только две процедуры для событий класса: `Initialize` и `Terminate`.

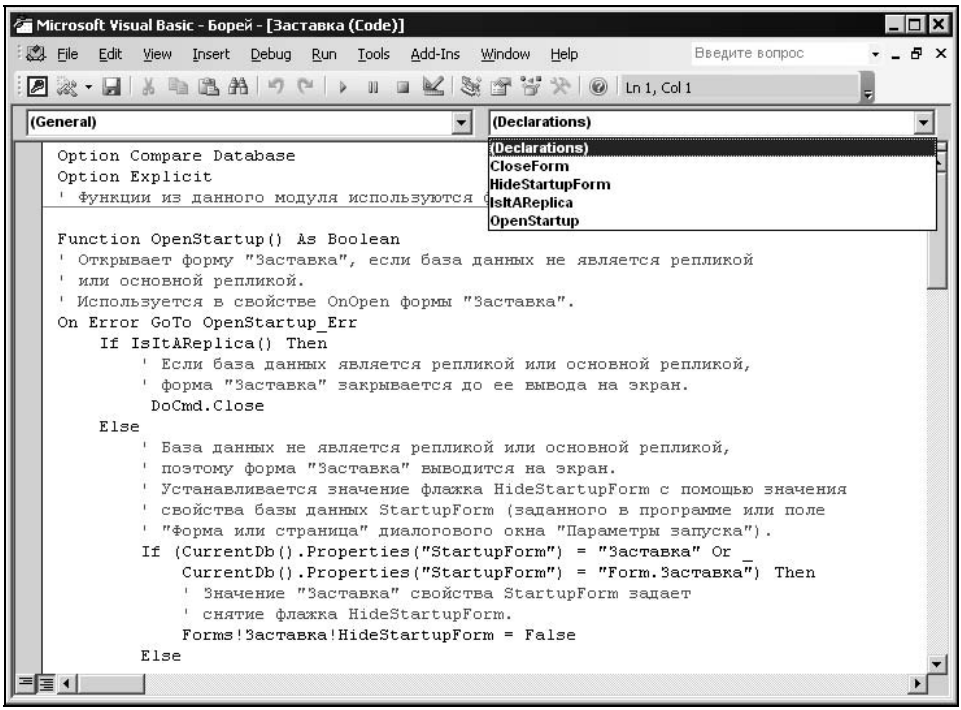


Рис. 13.16. Стандартный модуль в окне редактора

С помощью кнопок в левом нижнем углу окна можно выбрать один из видов представления кода в окне:

- **Procedure View** (Представление процедуры), когда в окне отображается только одна выбранная процедура.
- **Full Module View** (Представление полного модуля), когда в окне отображается сразу несколько процедур, отделенных друг от друга тонкой линией.

Для настройки параметров редактора используется также диалоговое окно **Options** (Параметры), которое можно открыть, выбрав команду **Options** в меню **Tools** (Сервис) (рис. 13.17). Это окно описывает только параметры редактора кода и вызывается из среды редактора. Оно содержит четыре вкладки.

Первая вкладка **Editor** (Редактор) состоит из двух групп параметров: **Code Settings** и **Window Settings**.

- Группа **Code Settings** (Программирование) содержит шесть флажков, которые позволяют включать и отключать интеллектуальные функции редактора, такие

как автоотступ, автоматическая проверка синтаксиса, подсказки при написании пользователем программ.

- Группа **Window Settings** (Параметры окна) включает в себя три флажка, влияющих на представление кода в окне редактора. Эти флажки позволяют переключаться между представлениями полного модуля и процедуры, выводить или не выводить на экран тонкие линии, разделяющие процедуры, включать или отключать функцию перетаскивания текста, т. е. перемещения выделенного фрагмента кода с помощью мыши.

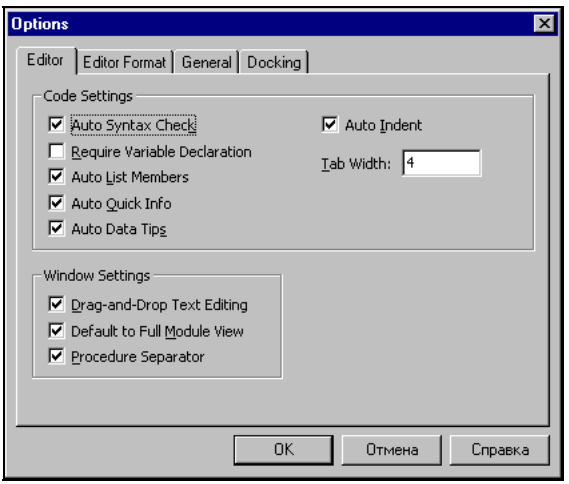


Рис. 13.17. Диалоговое окно параметров редактора кода

Вторая вкладка **Editor Format** (Формат редактора) позволяет определить шрифт, его размер и цвет для разных областей текста. Открыв любой модуль, вы можете убедиться, что текст программ представлен несколькими цветами: по умолчанию комментарии выделяются зеленым цветом, ключевые слова — синим. При отладке программ используются и другие цветовые выделения фрагментов текста: точек останова, текущей команды и т. д. Все эти способы выделения могут быть настроены в данном окне в соответствии с вашим вкусом.

Третья вкладка **General** (Общая) содержит несколько групп переключателей и флажков.

Группа **Form Grid Settings** относится к созданию форм в Microsoft Visual Basic 6.0 и в других компонентах Office и не используется в Access.

Флажок **Show ToolTips** позволяет включить или отключить вывод всплывающих подсказок для кнопок на панели инструментов.

Флажок **Collapse Proj. Hides Windows** определяет, будет ли скрываться окно редактора при закрытии проекта на панели проекта.

Флажок **Notify Before State Loss** определяет, получит ли пользователь сообщение о том, что предпринимаемое действие приведет к тому, что все переменные уровня модуля в проекте будут сброшены.

Переключатели **Error Trapping** определяют установки по умолчанию, используемые при обработке ошибок в программах:

- **Break on All Errors** — любая ошибка, независимо от того, обрабатывается ли она программой, приведет к прерыванию выполнения программы;
- **Break in Class Module** — любая необработанная ошибка в модулях класса приведет к прерыванию выполнения программы;
- **Break on Unhandled Errors** — прерывание программы вызывают все необработанные ошибки.

Последняя группа флажков определяет, когда компилируются модули проекта:

- **Compile On Demand** — определяет, будут ли все модули проекта компилироваться перед началом работы или только по требованию, в первом случае запуск приложения будет более длительным;
- **Background Compile** — определяет, будет ли компиляция модулей выполняться в фоновом режиме.

На *последней вкладке* **Docking** находятся флажки, позволяющие закрепить положение основных окон на экране.

Для того чтобы убедиться, что наши утверждения об интеллектуальности редактора не являются голословными, достаточно попробовать написать даже самую простую программу. Давайте заменим в форме `Клиенты` (Customers) макрос, который открывает форму `Заказы` (Orders) при нажатии кнопки `Заказы клиента`, процедурой VBA, которая выполняет те же действия. Мы создавали такой макрос в *разд. "Назначение макроса событию" гл. 12*. Если вы этого не делали, тогда создайте сейчас кнопку в области заголовка окна и назовите ее `Заказы клиента` (кнопка Мастера на панели элементов при этом должна быть отжата).

После этого нажмите кнопку **Программа** (Code) на панели инструментов. Откроется редактор кода VBA, который по умолчанию попытается создать процедуру обработки события `Load` формы. Выберите из списка **Object** слева объект `Заказы клиента`. Редактор автоматически вставит заголовок и концовку процедуры обработки события `Click` (рис. 13.18). Открыв список **Procedures** справа, можно увидеть, как много различных событий связано с командной кнопкой, однако наиболее часто используется именно событие `Click`. Так как с этим событием сейчас не связано никакой процедуры, редактор сразу пытается ее создать.

Начнем писать текст процедуры. Процедура будет состоять из нескольких команд. Нам необходимо открыть форму `Заказы` (Orders) и показать в этой форме только те заказы, которые относятся к определенному клиенту, поэтому придется задать условие для отбора записей в форме `Заказы` (Orders). Обычно для этого используют переменную типа `String`, которой сначала присваивают нужное значение, а потом подставляют в качестве параметра в макрокоманду `OpenForm` (ОткрытьФорму). Чтобы использовать такую переменную, ее сначала нужно описать, поэтому первое предложение в процедуре должно быть следующим:

```
Dim stLinkCriteria As String
```

Как только вы напишете первые три слова `Dim stLinkCriteria As`, на экране появится список слов (рис. 13.19), которые могут быть использованы в данном предложении.

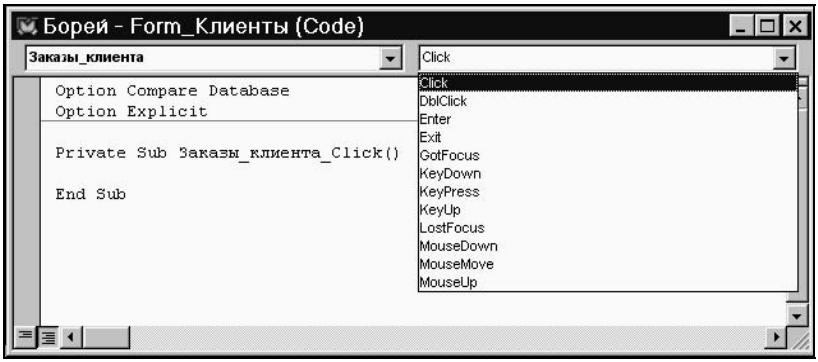


Рис. 13.18. Создание процедуры обработки события

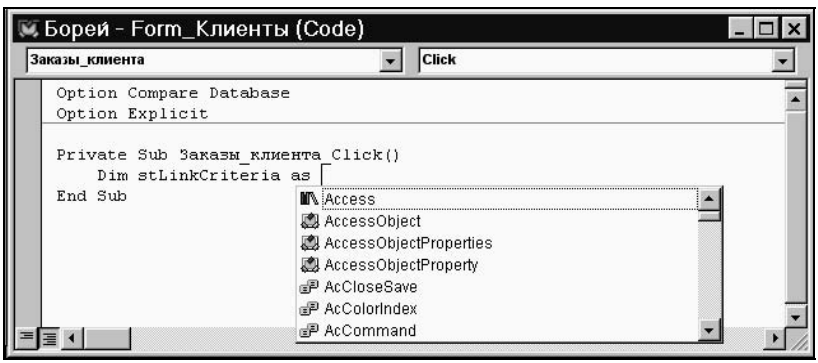


Рис. 13.19. Автоматический вывод списка компонентов

Вы набираете следующие буквы — `str`, после чего курсор в списке устанавливается на слове `String`. Дальше можно не продолжать, а просто нажать клавишу <Tab> — конец слова допишется автоматически. Нажмите клавишу <Enter>, курсор перейдет на новую строку, а текст во введенной строке изменится: все ключевые слова — `Dim`, `As`, `String` — будут выделены синим цветом. При вводе ключевых слов можно не беспокоиться о том, чтобы слово начиналось с заглавной буквы. Вы можете набирать весь текст строчными буквами — необходимое преобразование будет выполняться автоматически. Но, конечно, названия переменных, элементов управления и других объектов, на которые имеются ссылки в тексте, вы должны вводить сами с использованием нужного регистра, т. к. в них преобразования выполняться не будут.

Следующая строка будет присваивать переменной `stLinkCriteria` значение:

```
stLinkCriteria = "[КодКлиента]=" & "'" & Me![КодКлиента] & "'"
```

Две формы будут связаны по значению поля `CustomerID`. Первое выражение `[КодКлиента]` относится к форме `Заказы` (Orders), а второе значение `Me![КодКлиента]` — к текущей форме `Клиенты` (Customers), о чем говорит слово `Me`. Теперь используем макрокоманду `OpenForm` (ОткрытьФорму) для того, чтобы открыть форму `Заказы` (Orders). Как только набирается `docmd`, появляется новая подсказка — список макрокоманд (рис. 13.20).

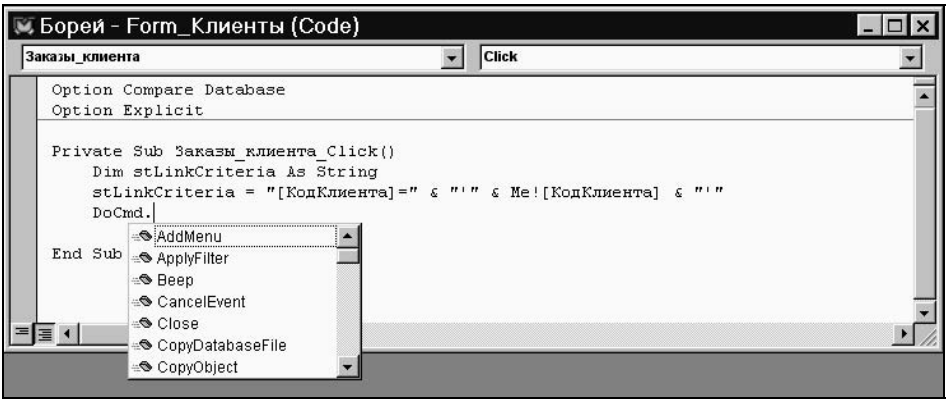


Рис. 13.20. Автоматический вывод методов объекта

Уже известным способом выбираем нужную макрокоманду, нажимаем пробел и опять подсказка — синтаксис макрокоманды `OpenForm` (ОткрытьФорму) (рис. 13.21).

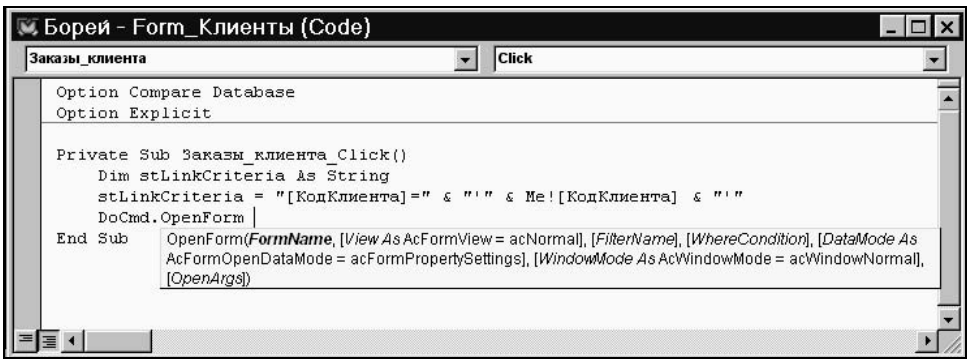


Рис. 13.21. Автоматический вывод кратких сведений

И это очень кстати, т. к. позволяет не запоминать все аргументы макрокоманды и порядок их следования. Введите имя формы: `Заказы` (Orders), запятую, и опять появится подсказка с именами возможных констант, которые могут быть указаны в следующем параметре. Выберите константу `acNormal` и продолжайте ввод. Должна получиться строка:

```
DoCmd.OpenForm "Заказы", acNormal,, stLinkCriteria
```

Две следующие подряд запятые говорят о пропущенном параметре — имя фильтра.

Теперь добавим еще одну команду — изменим заголовок формы:

```
Forms!Заказы.Caption = "Заказ"" & Название
```

И наконец, напишем комментарий, который объясняет, что делает данная программа. Для этого поместите курсор в начало второй строки программы и нажмите клавишу <Enter>. Появится пустая строка. Введите символ (') в начало новой строки и пишите дальше текст комментария. Закончите ввод нажатием клавиши <Enter>, и

вы увидите, что комментарий оказался выделен зеленым цветом. В целом текст программы должен выглядеть как на рис. 13.22.

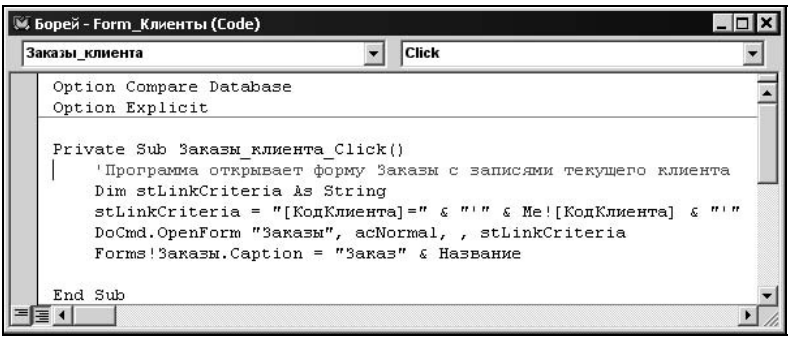


Рис. 13.22. Программа открытия формы `Заказы`

На примере этой простой программы мы продемонстрировали три очень полезных свойства редактора.

- *Автоматическая проверка синтаксиса* — ввод команды успешен и все слова выделяются нужным цветом только тогда, когда синтаксис предложения правилен, в противном случае вы услышите звуковой сигнал, текст будет выделен красным цветом и появится сообщение об ошибке.
- *Автоматический вывод списка компонентов* **Auto List Members** позволяет избежать ошибок при вводе наименований свойств объектов, встроенных констант, типов переменных. Особенно эта функция полезна при установке свойств, т. к. позволяет не только вспомнить правильное имя свойства, но и увидеть, какие свойства объекта доступны в данный момент.
- *Автоматический вывод кратких сведений* **Auto Quick Info** показывает в виде всплывающей подсказки синтаксис встроенных функций и макрокоманд. По мере ввода параметров очередной параметр выделяется в подсказке жирным шрифтом.

Когда вы уже усвоите язык VBA, такие подсказки могут показаться вам навязчивыми. Тогда вы можете отключить их, сбросив соответствующие флажки в диалоговом окне **Options** (Параметры) редактора кода (вкладка **Editor**).

Однако при необходимости вы легко получите нужную помощь, если воспользуетесь контекстным меню редактора (рис. 13.23).

Кроме описанных ранее функций, в контекстном меню присутствуют и другие очень полезные функции, способствующие быстрому написанию кода, — **List Constants**, **Parameter Info**, **Complete Word**, а также две команды, очень упрощающие навигацию среди программных модулей:

- **Definition.** Если в процедуре выполняется вызов другой процедуры, то, установив курсор на имя вызываемой процедуры и выбрав из контекстного меню команду **Definition** (а еще быстрее, нажав комбинацию клавиш <Shift>+<F2>), вы сразу попадете в тело вызываемой процедуры;
- **Last Position** — позволяет после такого перехода вернуться обратно.

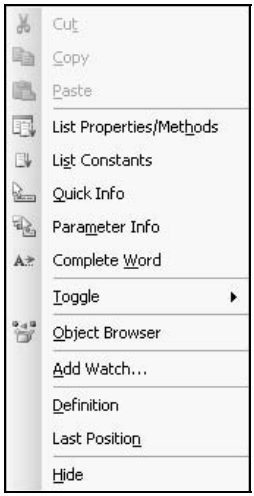


Рис. 13.23. Контекстное меню редактора кода

Команда **Definition** позволяет также быстро открыть окно **Object Browser** и посмотреть описание нужного свойства, метода или объекта приложения. Для этого нужно установить курсор на интересующий объект, свойство, метод и нажать комбинацию клавиш <Shift>+<F2>.

Функции **List Constants**, **Parameter Info** позволяют в любой момент отобразить на экране список констант, которые нужно подставить в качестве параметра процедуры или значения свойства, и информацию о параметре процедуры соответственно. Функция **Complete Word** обеспечивает завершение ввода ключевых слов, когда введены первые символы этих слов.

При работе в окне редактора по умолчанию открыта панель инструментов **Standard**. Но мы рекомендуем открыть еще одну панель инструментов — **Edit**, т. к. на ней находятся несколько очень полезных кнопок (рис. 13.24).

Например, кнопки **Indent** (Отступ) и **Outdent** (Выступ) удобно использовать, когда нужно привести в порядок структуру программы. Целые блоки кода можно легко сдвигать вправо или влево. Кнопка **Comment Block** позволяет временно закомментировать блок кода, что тоже бывает полезно при отладке программ. Несколько кнопок обеспечивают удобный способ навигации в коде, когда перемещение курсора происходит по закладкам (bookmarks), которые маркируют строки кода. Описание команд (кнопок) обеих панелей приведено в табл. 13.11 и 13.12.



Рис. 13.24. Панели инструментов **Standard** и **Edit**

Таблица 13.11. *Панель инструментов* ***Standard***

Команда	Назначение	Команда меню
View Microsoft Office Access	Переключение в окно приложения Access	**File	Close and Return to Microsoft Office Access**
Insert (Вставка) **Insert Module** **Insert Class Module** **Insert Procedure**	Кнопка со списком выбора, которая позволяет создать новые модель, модуль класса или процедуру	**Insert Procedure	Insert Module	Insert Class Module**
Save (Сохранить)	Сохранить изменения в проекте VBA	**File	Save**
Cut (Вырезать)	Вырезать блок и скопировать его в буфер	**Edit	Cut**
Copy (Копировать)	Скопировать блок в буфер	**Edit	Copy**
Paste (Вставить)	Вставить блок из буфера	**Edit	Paste**
Find (Найти)	Позволяет найти подстроку в тексте текущей процедуры, модуля, проекта и, если требуется, заменить ее другой подстрокой	**Edit	Find**
Undo (Отмена)	Отменяет последнее изменение	**Edit	Undo**
Redo (Отмена отмены)	Возвращает последнее изменение	**Edit	Redo**
Run Sub/UserForm	Продолжает выполнение процедуры после точки останова, отключая при этом пошаговый режим, если он был установлен	**Run	Run Sub/UserForm**
Break (Прервать)	Прекращает выполнение процедуры	**Run	Break**
Reset (Сброс)	Прекращает выполнение процедуры и присваивает переменным начальные значения по умолчанию	**Run	Reset**
Design Mode	Переключает между режимами Конструктора форм/отчетов и окном редактора кода	**Run	Design Mode**
Project Explorer	Открывает (или активизирует, если она открыта) панель проекта	**View	Project Explorer**
Properties Window (Окно свойств)	Открывает (или активизирует, если оно открыто) окно свойств объекта	**View	Properties Window**
Object Browser (Окно обозревателя объектов)	Открывает обозреватель объектов	**View	Object Browser**
Toolbox (Панель элементов)	Управляет отображением панели элементов (не используется в Access)	**View	Toolbox**

Таблица 13.11 *(окончание)*

Команда	Назначение	Команда меню
Справка: Microsoft Visual Basic	Отображает область задач справка Visual Basic	**Help | Справка: Microsoft Visual Basic**
Ln x, Col x	Текстовая область, в которой отображается позиция (строка, столбец) курсора в окне кода	

Таблица 13.12. *Панель инструментов* ***Edit***

Команда	Назначение	Команда меню
List Properties/Methods	Соответствуют одноименным командам контекстоного меню окна кода	**Edit | List Properties/Methods**
ListConstants		**Edit | ListConstants**
Quick Info		**Edit | Quick Info**
Parameter Info		**Edit | Parameter Info**
Complete Word		**Edit | Complete Word**
Indent (Отступ)	Обеспечивает сдвиг выделенного блока строк кода вправо на один символ табуляции	**Edit | Indent**
Outdent (Выступ)	Обеспечивает сдвиг выделенного блока строк кода влево на один символ табуляции	**Edit | Outdent**
Toggle Breakpoint (Установка/сброс точек останова)	Устанавливает/снимает точку останова на строку, в которой находится курсор	**Debug | Toggle Breakpoint**
Comment Block (Коментировать блок)	Преобразует выделенный блок строк кода в комментарий	Нет
Uncomment Block (Раскомментировать блок)	Снимает комментарий с выделенного блока строк кода	Нет
Toggle Bookmark (Шаг с выходом)	Пометить/снять отметку со строки кода	**Edit | Bookmark | Toggle Bookmark**
Next Bookmark (Окно Locals)	Перейти к следующей помеченной строке кода	**Edit | Bookmark | Next Bookmark**
Previous Bookmark (Окно Immediate)	Перейти к предыдущей помеченной строке кода	**Edit | Bookmark | Previous Bookmark**

Таблица 13.12 (окончание)

Команда	Назначение	Команда меню
Clear All Bookmarks (Окно Watches)	Снять все пометки	**Edit | Bookmark | Clear All Bookmarks**

Использование окна просмотра объектов

Окно **Object Browser** (Обозреватель объектов) позволяет просматривать все объекты, их свойства и методы, доступные для текущего проекта. Объектами могут быть встроенные объекты Access или VBA, созданные вами в приложении объекты, а также объекты, входящие во внешние библиотеки, на которые имеются ссылки в текущем проекте. Вызвать окно просмотра объектов можно тремя способами:

- нажать клавишу <F2> — самый быстрый;
- нажать кнопку **Object Browser** (Обозреватель объектов) на панели инструментов — удобный;
- выбрать команду **View | Object Browser** (Вид | Обозреватель объектов) — если вы забыли другие способы.

Окно **Object Browser** (Обозреватель объектов, рис. 13.25) состоит из нескольких списков, которые обеспечивают трехуровневое представление информации.

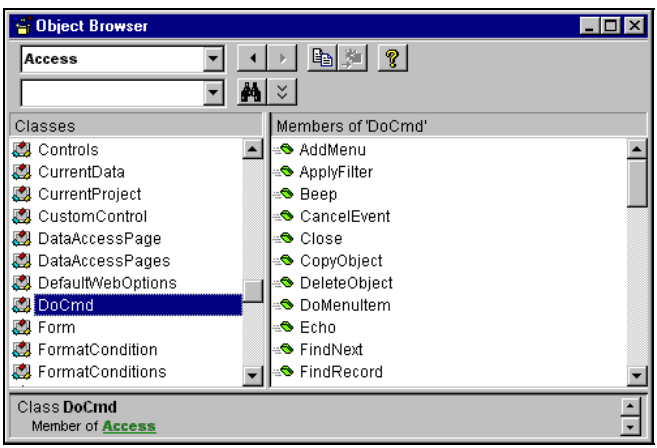


Рис. 13.25. Окно просмотра объектов

Список **Project/Library** (Проект/Библиотека) в левом верхнем углу окна содержит перечень всех библиотек и проектов, на которые имеются ссылки в данном проекте. Как минимум, он включает библиотеку Access, библиотеку VBA, одну из библиотек доступа к данным, библиотеку текущего проекта.

При выборе из списка одной из библиотек в нижнем левом поле **Classes** (Классы) отображается список следующего уровня — перечень всех объектов, входящих в эту

библиотеку. Например, если выбрать библиотеку Access, то в списке **Classes** вы увидите много знакомых объектов. Выбрав один из них, например `DoCmd`, в правом поле **Members of** можно увидеть все методы этого объекта. Если бы мы выбрали объект `Form`, то в правом списке отобразились бы все свойства и методы объекта `Form`.

При этом в нижней части окна, которая называется *областью описания*, отображается описание выбранного элемента.

Если выбрать в поле **Project/Library** (Проект/Библиотека) текущее приложение `Борей` (Northwind), то в поле **Classes** (Классы) отобразятся все модули форм, отчетов и стандартные модули приложения (рис. 13.26).

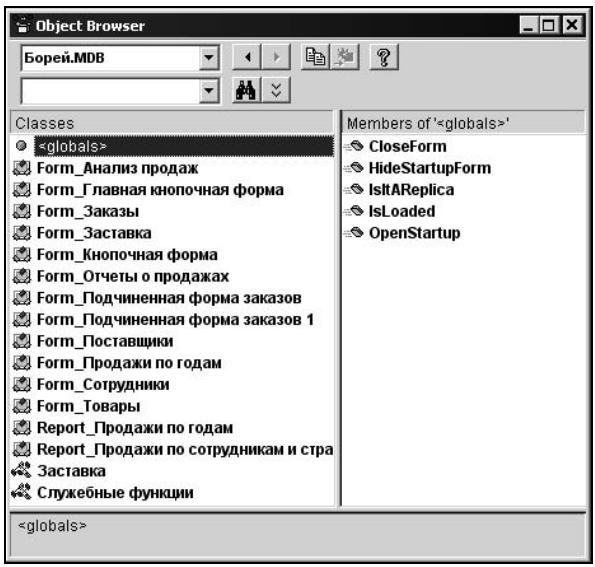


Рис. 13.26. Объекты приложения `Борей`

Если вы написали много процедур, то это удобный способ найти нужную процедуру. Кроме того, если вы, выбрав в списке компонентов модуля интересующую вас процедуру, нажмете кнопку **View Definition** (Посмотреть описание) (см. рис. 13.26), то откроется окно данного модуля, и вы увидите в нем именно эту процедуру.

Для поиска необходимого объекта, метода или свойства существует поле **Search Text** (Образец поиска). Это второе поле со списком в верхней части окна. В него можно ввести фрагмент текста для поиска и нажать кнопку с биноклем, которая находится рядом. Тогда открывается дополнительная область **Search Results** (Результаты поиска), в которой отображается список компонентов, удовлетворяющих условию поиска (рис. 13.27).

Выделив в списке нужную строку, можно снова воспользоваться кнопкой **View Definition**, чтобы открыть нужную процедуру.

Еще одна очень полезная кнопка в этом окне **Copy to Clipboard** (Копировать в буфер обмена). Она позволяет копировать выбранный объект в буфер, после чего объект можно будет вставить прямо в текст программы.

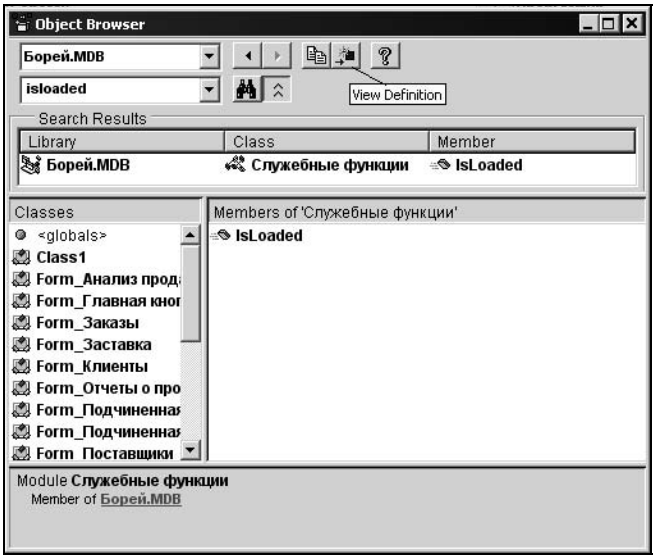


Рис. 13.27. Отображение результатов поиска

Установка ссылок на объектные библиотеки

При обращении к объекту из процедуры VBA вы можете получить сообщение об ошибке, которое может быть вызвано тем, что не подключена соответствующая библиотека. При установке Access 2003 по умолчанию подключаются следующие библиотеки:

- Visual Basic for Applications;
- Microsoft Access 11.0 Object Library;
- Microsoft DAO 3.6 Object Library.

Все остальные библиотеки при необходимости должны быть подключены вручную, для чего нужно установить ссылку на соответствующую библиотеку.

Откройте редактор VBA и выполните команду **Tools | References** (Сервис | Ссылки). В открывшемся диалоговом окне **References** (Ссылки, рис. 13.28) найдите требуемую библиотеку и установите необходимый флажок.

Все библиотеки, на которые установлены ссылки, помещаются в начале списка, причем, пользуясь кнопками **Priority** (Приоритет), вы можете поменять порядок их следования. Обращение к этим библиотекам при разрешении ссылок на объекты выполняется в том порядке, в каком они следуют в списке.

Если нужная библиотека отсутствует в списке, вы можете подключить ее вручную, если знаете, где находится файл, содержащий эту библиотеку. Для этого нажмите кнопку **Browse** (Обзор), в диалоговом окне **Add Reference** (Добавить ссылку) найдите нужный файл и нажмите кнопку **Open** (Открыть). Ссылка на соответствующую библиотеку должна появиться в окне **References** (Ссылки) данного проекта.

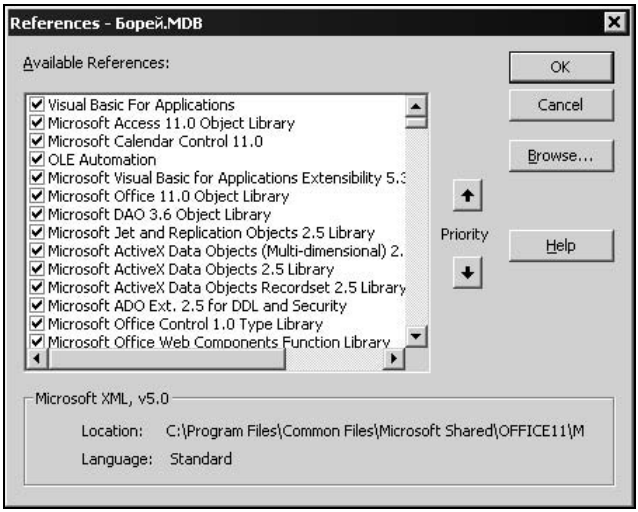


Рис. 13.28. Диалоговое окно **References**

Инструментальные средства отладки

Помимо интеллектуального редактора текста профессиональная среда программирования обычно содержит инструментальные средства отладки. Инструментальные средства отладки призваны дать разработчику максимально ясное представление о том, как работает его программа. И уже искусство разработчика состоит в том, чтобы, используя все имеющиеся в его распоряжении средства, быстро выявить ошибки. Набор средств отладки в Access широк: это и специальное меню **Debug** (Отладка), и во многом дублирующие его кнопки на панели инструментов, и специальные окна отладки. Далее кратко описывается каждое средство.

Меню *Debug*

Меню **Debug** (Отладка) и специальная панель инструментов **Debug** (Отладка) представлены на рис. 13.29.

Часть кнопок этой панели повторяют кнопки, которые присутствуют на панелях **Standard** и **Edit**, но, тем не менее, ее тоже обычно отображают при отладке программы, т. к. она обеспечивает быстрый доступ к специальным окнам отладки, а также пошаговый режим выполнения программы. Назначение команд (кнопок) описано в табл. 13.13.

Таблица 13.13. *Панель инструментов* ***Debug***

Команда	Назначение	Команда меню
Design Mode (Режим Конструктора)	Переключает между режимами Конструктора форм/отчетов и окном редактора кода	**View | Object**

Таблица 13.13 *(окончание)*

Команда	Назначение	Команда меню
Run Sub/UserForm (Продолжить)	Продолжает выполнение процедуры после точки останова, снимая при этом пошаговый режим, если он был установлен	**Run | Run Sub/UserForm**
Break (Прервать)	Прекращает выполнение процедуры	**Run | Break**
Reset (Сброс)	Прекращает выполнение процедуры и присваивает переменным начальные значения по умолчанию	**Run | Reset**
Toggle Breakpoint (Установка/сброс точек останова)	Устанавливает/снимает точку останова на строку, в которой находится курсор	**Debug | Toggle Breakpoint**
Step Into (Шаг с заходом)	Исполняет очередную строку кода с заходом в процедуры	**Debug | Step Into**
Step Over (Шаг с обходом)	Исполняет очередную строку кода без захода в процедуры, т. е. функции и процедуры выполняются за один шаг	**Debug | Step Over**
Step Out (Шаг с выходом)	Выполняет остаток текущей процедуры и останавливается в вызывающей программе на следующей строке после вызова этой процедуры	**Debug | Step Over**
Locals Window (Окно Locals)	Открывает окно отладки **Locals**	**View | Locals Window**
Immediate Window (Окно Immediate)	Открывает окно отладки **Immediate**	**View | Immediate Window**
Watch Window (Окно Watches)	Открывает окно отладки **Watches**	**View | Watch Window**
Quick Watch (Быстрый просмотр)	Выводит в специальном окне текущее значение выражения в точке останова	**Debug | Quick Watch**
Call Stack (Окно Стек вызовов)	Открывает окно **Call Stack** (Стек вызовов)	**View | Call Stack**

В меню **Debug** обращаем ваше внимание на три команды, которые не имеют аналогов на панелях инструментов, но часто используются в процессе отладки:

- **Run To Cursor** — выполняет фрагмент кода от текущей команды, на которой остановлено выполнение программы, до той команды, на которой установлен курсор, т. е. она действует как "временная" точка останова.
- **Set Next Statement** — изменяет текущую команду. Обычно ее используют, если нужно повторить выполнение какого-то блока кода, например, изменяют некоторые переменные, устанавливают курсор на команду, которая уже выполнялась

раньше, и выполняют эту команду. После этого программа начнет выполняться с той строки, куда был установлен курсор.

- **Show Next Statement** — показывает текущую команду и перемещает на нее курсор. Применяется, если при анализе программы в процессе отладки нужно быстро перейти к текущей команде (переместить на нее курсор), когда на экране отображается другой фрагмент кода.

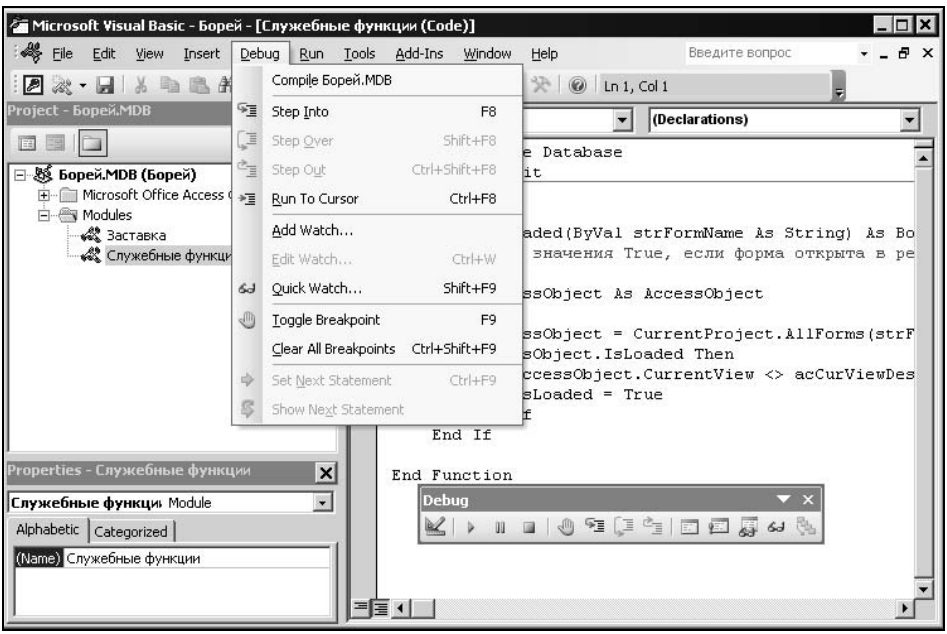


Рис. 13.29. Меню и панель инструментов **Debug**

Окно *Locals*

Специальные окна отладки используются главным образом для того, чтобы просматривать промежуточные значения данных: переменных, выражений, свойств объектов и т. д. К ним относятся два окна: **Locals** (Локальные) и **Watches** (Контрольные). Чтобы познакомиться с этими окнами, установите точку останова на одной из команд написанной ранее процедуры `Заказы_Click` в форме `Клиенты` (Customers). Для этого нужно:

1. Открыть данную процедуру в окне редактора кода, как это описывалось ранее.
2. Установить курсор на строку

```
DoCmd.OpenForm "Заказы", acNormal,, stLinkCriteria
```

3. Нажать кнопку на панели инструментов с изображением руки или выбрать команду **Toggle Breakpoint** в меню **Debug**. Строка будет выделена коричневым фоном и на панели слева, напротив этой строки, появится специальный значок — жирная коричневая точка.

Совет

Самый простой и естественный способ установки точек останова: щелкните мышью по серой полосе у левого края окна редактора напротив нужной строки. Снять точку можно щелчком мыши по коричневой точке напротив нужной строки.

Теперь окно модуля можно закрыть, открыть форму `Клиенты` (Customers) в режиме Формы и нажать кнопку `Заказы клиента`. При нажатии этой кнопки запускается наша процедура обработки событий и, как только выполнение дойдет до помеченной строки, произойдет останов, а на экране появится окно редактора кода, в котором помеченная строка выделена желтым цветом, означающим, что это текущая команда. Нажмите кнопку **Local Window** на панели инструментов **Debug** или выберите соответствующую команду из контекстного меню редактора. Тогда в нижней части экрана появится новое окно **Locals** (Локальные, рис. 13.30). В этом окне можно увидеть все локальные переменные и константы текущей процедуры, а также иерархический список свойств всех объектов формы, который скрывается за коротеньким словом `Me`. Именно эти свойства и интересуют, как правило, разработчика, т. к. в большинстве случаев значения переменных в точке останова можно увидеть, просто подведя курсор к этой переменной в окне редактора. Тогда появляется всплывающая подсказка со значением переменной, как это показано на рис. 13.32.

Замечание

Если подсказка не появляется, проверьте значение флажка **Auto Data Tips** (Выводить подсказки к данным) на вкладке **Editor** (Редактор) окна **Options** (Параметры).

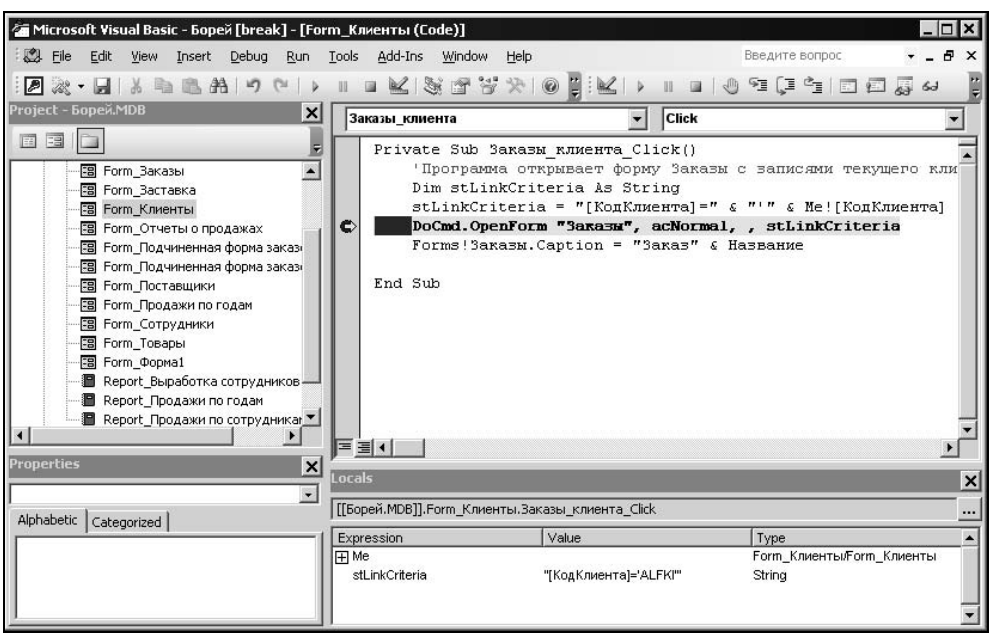


Рис. 13.30. Установка точки прерывания в программе

Если выполнять процедуру по шагам, то можно наблюдать, как меняются значения свойств.

Окно *Watches*

Окно **Watches** (Контрольные) содержит контрольные выражения. Данные выражения контролируются самой Access в процессе выполнения программы, и когда выполнение программы останавливается, в этом окне можно увидеть текущие значения этих выражений. Можно составить контрольное выражение таким образом, что приложение останавливается, когда это выражение принимает определенное значение. Такой прием, например, используется, когда отлаживаются длинные циклы, которые слишком долго выполнять по шагам.

Для того чтобы отследить контрольные значения, их сначала нужно задать. Для этого используется команда **Add Watch** (Добавить контрольное значение) в меню **Debug** (Отладка). Добавлять контрольные значения можно как в режиме редактирования кода, так и в режиме останова. При выборе команды появляется диалоговое окно **Add Watch** (Добавление контрольного значения, рис. 13.31). В поле **Expression** (Выражение) можно ввести контрольное значение — это может быть переменная, свойство или любое допустимое выражение. Введите в это поле условие останова приложения `stLinkCriteria <> ""`, а в группе переключателей **Watch Type** (Тип контрольного значения) выберите переключатель **Break When Value Is True** (останов, если значение истинно).

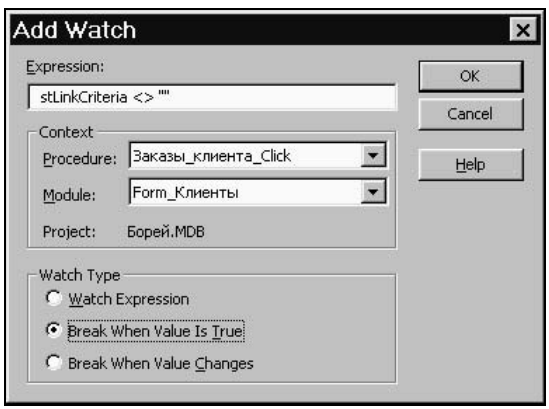


Рис. 13.31. Добавление контрольного значения

Замечание

Группа **Context** (Контекст) в окне **Add Watch** позволяет определить область контроля заданного значения или выражения: конкретная процедура текущего модуля, все процедуры текущего модуля или все модули текущей базы данных.

Теперь нужно снять точку останова, которую мы установили. Для этого щелкните правой кнопкой мыши в строке останова и выберите команду **Toggle | Breakpoint** из контекстного меню (или нажмите клавишу <F9>). Строка примет обычный вид.

Закройте окно редактора кода и снова нажмите кнопку `Заказы клиента` в форме `Клиенты` (Customers). Выполнение программы остановится на той же команде. Нажмите кнопку **Watch Window** на панели инструментов **Debug** или выберите соответствующую команду в меню **View**. В нижней части редактора появляется новое окно **Watches** (рис. 13.32), в котором будет выведена одна строка, содержащая контрольное выражение, его значение, тип и контекст, т. е. процедура, в которой вычисляется контрольное значение. Выражение истинно, и программа остановлена.

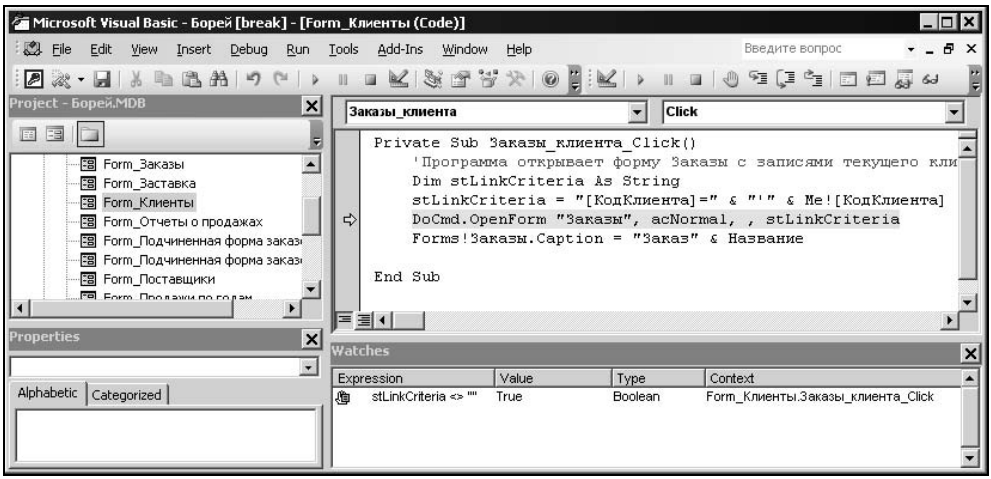


Рис. 13.32. Окно **Watches**

Существует очень простой способ добавления контрольных значений в окно **Watches**. Можно просто выделить в тексте процедуры выражение и перетащить его в это окно мышью. Попробуйте проделать эту операцию с выражением `Forms!Заказы.Caption` и выполните несколько шагов процедуры, наблюдая, как изменяется выражение в окне **Watches**.

Если нужно изменить или удалить контрольное выражение, щелкните правой кнопкой мыши на соответствующей строке окна **Watches** и выберите в контекстном меню команду **Edit** (Редактировать) или **Delete** (Удалить) соответственно.

Окно *Quick Watch*

Окно **Quick Watch** (Быстрый просмотр) используется для просмотра значения свойства, переменной или выражения, которое не определено заранее как контрольное значение. Чтобы открыть это окно, необходимо:

1. Выделить в тексте программы интересующее выражение.
2. Нажать кнопку **Quick Watch** (Быстрый просмотр) на панели инструментов, или выбрать команду **Debug | Quick Watch**, или нажать комбинацию клавиш <Shift>+<F9>.

Диалоговое окно на рис. 13.33 отображает значение переменной `stLinkCriteria`.

Если вы решите добавить выбранное выражение к контрольным значениям, просто нажмите кнопку **Add** (Добавить).

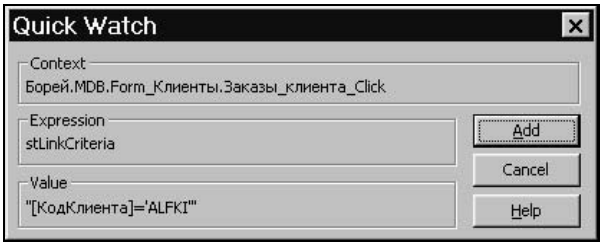


Рис. 13.33. Диалоговое окно **Quick Watch**

Окно *Immediate*

Кроме всех перечисленных окон отладки, в Access 2003 сохранилось окно **Immediate** (Отладка) для тестирования данных и процедур, которое присутствовало еще в Access 2.0. Это окно используется двояко: можно вручную проверить значение выражения с помощью команды `Print` или `?` (рис. 13.34) либо, вставив в код программы команду `Debug.Print`, посмотреть в этом окне контрольную печать.



Рис. 13.34. Окно **Immediate**

Окно **Immediate** открывается при нажатии кнопки **Immediate Window** на панели инструментов **Debug**, или при выполнении соответствующей команды в меню **View**, или при нажатии комбинации клавиш <Ctrl>+<G>. Это окно отображается на экране, даже если окно редактора закрыто и выполняется приложение. При этом в нем можно наблюдать значения указанных данных. Выводить в окно можно любые допустимые выражения, включая свойства объектов.

Для того чтобы вывести значение переменной, нужно ввести знак `?` и имя переменной, после чего нажать клавишу <Enter>. Значение переменной отобразится на следующей строке.

Метод `Print` объекта `Debug` направляет вывод в это окно в процессе выполнения программы. Например, команда

```
Debug.Print "Цена.Enabled = "; Цена.Enabled
```

печатает значение свойства `Доступ` (Enabled) поля `Цена` (UnitPrice) в окне **Immediate**.

В окне **Immediate** можно не только выводить значения переменных и свойств, но и задавать их, например можно в режиме останова ввести команду

```
Цена.Enabled = True
```

При этом на экране должна быть открыта форма, в которой присутствует данное поле. Аналогично присваиваются значения переменным, вызываются процедуры и выполняется большинство других команд. Существует ограничение на вводимые команды — они должны полностью помещаться на одной строке. После ввода команды и нажатия клавиши <Enter> Access выполняет введенную команду и снова переходит в режим останова. Таким образом, можно тестировать фрагменты программ, процедуры и функции.

Окно *Call Stack*

Окно **Call Stack** (Стек вызовов) содержит список всех незавершенных процедур приложения и используется для трассировки вложенных процедур. При этом самая первая из вызванных процедур будет размещаться в нижней части стека, а самая последняя — в верхней части. Наименование процедуры состоит из трех частей: имени проекта, имени модуля (в том числе модуля формы), имени процедуры (рис. 13.35).

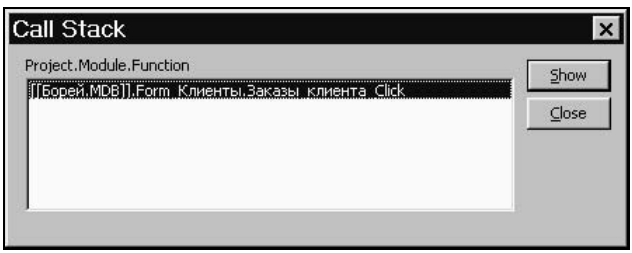


Рис. 13.35. Диалоговое окно **Call Stack**

Выбрав в стеке нужную процедуру, можно нажать кнопку **Show** (Показать), тогда окно **Call Stack** закроется и в окне кода модуля появится выбранная процедура. При этом курсор устанавливается на строку кода, вызывающую следующую процедуру из стека. Если выбирается текущая процедура, то курсор устанавливается на текущую команду.

Окно **Call Stack** вызывается несколькими способами:

- нажать кнопку **Call Stack** на панели инструментов;
- выбрать команду **View | Call Stack**;
- нажать комбинацию клавиш <Ctrl>+<L>;
- нажать кнопку с многоточием в верхнем правом углу окна **Locals**.

Окно **Call Stack** может быть открыто только в режиме останова программы.

Использование комбинаций клавиш быстрого вызова

Поскольку работа в редакторе — это главным образом работа с клавиатурой, существует большое количество специальных клавиш и комбинаций клавиш быстрого вызова для выполнения определенных функций. Вы можете увидеть их в меню **Run | Debug** и **View**. Список наиболее часто используемых клавиш приведен в табл. 13.14.

Таблица 13.14. *Комбинации клавиш быстрого вызова*

Комбинации клавиш	Назначение
<F7>	Открыть окно редактора кода
<Ctrl>+<F>	Найти
<Ctrl>+<F>	Заменить
<F3>	Найти далее
<Shift>+<F3>	Найти ранее
<Ctrl>+<↓>	К следующей процедуре
<Ctrl>+<↑>	К предыдущей процедуре
<Ctrl>+<J>	**List Properties/Members**
<Ctrl>+<Shift>+<J>	**List Constant**
<Ctrl>+<I>	**Quick Info**
<Ctrl>+<Shift>+<I>	**Parameter Info**
<Ctrl>+<Пробел>	Закончить слово
<F5>	Продолжить
<Shift>+<F5>	Сброс
<F8>	Шаг с заходом
<Shift>+<F8>	Шаг с обходом
<Ctrl>+<Shift>+<F8>	Шаг с выходом
<F9>	Установка/сброс точки останова
<Shift>+<F9>	Вызов окна **Quick Watch**
<Ctrl>+<Shift>+<F9>	Сброс всех точек останова
<F2>	Вызов окна просмотра объектов **Object Browser**
<Shift>+<F2>	**Definition**
<Ctrl>+<Shift>+<F2>	**Last Position**
<Ctrl>+<G>	Вызов окна **Immediate**
<Ctrl>+<L>	Вызов окна **Call Stack**

Программирование в формах и отчетах

Программирование в формах и отчетах, как правило, составляет большую часть кода приложения, т. к. именно формы и отчеты являются основой интерфейса пользователя и с помощью программирования этот интерфейс гибко настраивается нужным образом. Основой для программирования в формах и отчетах является множество

событий, обрабатывающихся специальными процедурами. О создании таких процедур, а также использовании макросов для обработки событий мы уже говорили ранее *(см. разд. "Применение макросов" гл. 12)*.

В данном разделе мы хотим сделать обзор этих событий и описать последовательность их возникновения с тем, чтобы читатель мог их правильно использовать.

Помимо процедур обработки событий, программирование может применяться для динамического (т. е. в процессе работы приложения) изменения свойств форм, отчетов и элементов управления.

События Access

Обычно события инициируются действиями пользователя. В зависимости от производимых им действий события можно разделить на несколько типов: события данных, события фокуса, события клавиатуры, события мыши, события печати, события фильтра, события окна, события ошибок и события таймера. Далее мы рассмотрим все эти типы событий.

События данных

К этому типу относятся события, происходящие тогда, когда пользователь вводит, удаляет или изменяет данные в форме, а также перемещается от одной записи к другой.

Текущая запись

Событие **Текущая запись** (Current) происходит, когда очередная запись получает фокус или выполняется повторное обращение к источнику данных формы — таблице или запросу. Таким образом, оно возникает как при открытии формы, так и при переходе от одной записи к другой. Чаще всего это событие используется для синхронизации записей в связанных формах.

Удаление

Событие **Удаление** (Delete) происходит, когда пользователь пытается удалить запись из формы. Оно происходит до того, как запись реально удаляется из базы данных. Процедура обработки этого события имеет параметр `Cancel`. Установка значения этого параметра в процедуре равным `True` позволит предотвратить удаление записи.

До подтверждения Del

Удаление записи из базы данных имеет такое большое значение, что с ним связано несколько событий. После события **Удаление** (Delete) Access выдает специальное окно, запрашивающее подтверждение удаления записи. Событие **До подтверждения Del** (BeforeDelConfirm) возникает до появления этого окна. Процедура обработки данного события имеет два параметра: `Cancel` и `Response`. Присваивая в процедуре значение `True` параметру `Cancel`, можно отменить удаление, и окно подтверждения выдаваться на экран не будет. Так что, это еще одна возможность отменить удаление программно (третья возможность отмены будет предоставлена пользователю в диалоговом окне подтверждения удаления). Если же параметру `Cancel` присвоить значение `False`, то параметр `Response` можно использовать, чтобы определить, нуж-

но ли выдавать окно подтверждения. Если `Response=1`, то запись будет удалена без подтверждения, если же `Response` установить равным 0, то Access выдаст окно, запрашивающее у пользователя подтверждение удаления записи.

Замечание

Если окно подтверждения удаления не выдается или событие **До подтверждения Del** (BeforeDelConfirm) не возникает, проверьте установку флажка **Изменения записей** (Record Changes) в окне **Параметры** (Options) (на вкладке **Правка и поиск**, группа **Подтверждение**).

После подтверждения Del

Событие **После подтверждения Del** (AfterDelConfirm) происходит как после подтверждения удаления записи, так и при отмене удаления. Процедура обработки данного события имеет один параметр — `Status`, который принимает значения 0, 1 или 2 и определяет, была ли удалена запись. Значение 0 указывает, что запись была успешно удалена, 1 означает, что удаление отменено программой обработки события, а значение 2 указывает, что удаление было отменено пользователем в окне подтверждения удаления. Это событие может быть использовано в программе для проверки, была ли удалена запись.

До вставки

Со вставкой новой записи связаны два события: **До вставки** (BeforeInsert) и описанное далее **После вставки** (AfterInsert). Событие **До вставки** (BeforeInsert) происходит, как только пользователь вводит первый символ в новую запись (одно из полей, не обязательно первое), но до того, как запись фактически будет создана. Процедура обработки этого события может быть использована для проверки того, разрешена ли вставка. Процедура имеет один параметр: `Cancel`. Если установить его значение равным `True`, то вставка записи будет запрещена. После этого события отменить вставку будет уже нельзя, можно только удалить вставленную запись.

После вставки

Событие **После вставки** (AfterInsert) происходит после того, как в таблицу добавлена новая запись. Обычно это бывает при переходе к следующей записи в форме. Процедура обработки этого события обычно используется для того, чтобы сделать повторный запрос к источнику данных с целью вывода новой записи.

До обновления

Событие **До обновления** (BeforeUpdate), так же как и следующее событие **После обновления** (AfterUpdate), возникает при любом изменении данных в записи или элементе управления. Это событие может относиться как к элементу управления, так и к записи в целом. Процедура обработки данного события имеет один параметр — `Cancel`, использующийся для того, чтобы отменить введенные изменения. Для этого ему необходимо присвоить значение `True`. Данное событие обычно применяется с целью проверки условий на значение в поле таблицы или записи в целом, если эти условия сложные (простые условия обычно задаются в свойстве **Условие на значение** (ValidationRule) элемента управления). Условия проверяются сразу для нескольких значений, причем в них используются ссылки на элементы управления в других

формах. При разных значениях введенных данных выдаются различные сообщения об ошибках.

При невыполнении условий можно отменить введенные изменения перед переходом на другую запись.

В следующем разделе будет описан порядок возникновения событий, относящихся как к элементу управления, так и к записи в целом.

После обновления

Событие **После обновления** (AfterUpdate) происходит после обновления данных в записи или элементе управления. И хотя обновление уже произошло, можно восстановить старые значения, воспользовавшись свойством `OldValue` элемента управления. Оно сохраняет старое значение элемента управления, которое сменится только после события **После обновления** (AfterUpdate).

Замечание

События **До обновления** (BeforeUpdate) и **После обновления** (AfterUpdate), а также **До вставки** (BeforeInsert) и **После вставки** (AfterInsert) не возникают, когда значения элементов управления формы изменяются с помощью программы VBA или макрокоманды `SetValue` (УстановитьЗначение). Кроме того, события **До обновления** и **После обновления** не возникают для вычисляемых элементов управления.

Изменение

Событие **Изменение** (Change) возникает в следующих случаях:

- при изменении содержимого текстового поля или поля со списком, при этом изменением может считаться любой непосредственно введенный или удаляемый символ;
- при изменении значения свойства **Текст** (Text) элемента управления с помощью макроса или процедуры VBA;
- в элементе управления **Набор вкладок** (Tab Control) при переходе с одной вкладки на другую.

Замечание

Событие **Изменение** (Change) не возникает при изменении значения вычисляемого элемента управления, а также, если с помощью макроса или программы VBA установлено значение текстового поля или поля со списком или если значение поля со списком выбрано из списка.

Некорректная программа обработки данного события может привести к каскадным событиям. Чтобы избежать этого, не следует использовать в настоящей процедуре команды, которые меняют содержимое элемента управления, а также не стоит создавать два и более поля, которые воздействуют друг на друга, например обновляют друг друга.

Отсутствие в списке

Событие **Отсутствие в списке** (NotInList) возникает в поле со списком, когда пользователь вводит вручную значение в текстовую часть поля, которое отсутствует в

списке, и после этого пытается перейти в другое поле или сохранить запись. Для того чтобы данное событие происходило, нужно присвоить свойству **Ограничиться списком** (LimitToList) значение `Yes`. Если это свойство имеет значение `No`, то разрешается ввод в поле данных, не совпадающих ни с одним значением из списка. Процедура обработки настоящего события имеет два параметра: `NewData` и `Response`. Параметр `NewData` содержит введенные данные, а `Response` управляет обработкой события и может иметь значения 0, 1 или 2. Значение 0 позволяет вывести на экран стандартное сообщение о том, что введенные данные отсутствуют в списке, и запретить ввод. Значение 1 позволяет вместо стандартного сообщения вывести специальное сообщение, например, запрашивающее, следует ли сохранить введенное значение. Новые данные при этом не добавляются в список. Значение 2 разрешает добавить новое значение в список. При этом в процедуре обработки данного события нужно добавить значение к источнику строк для поля со списком, после чего поле обновляется, т. к. Access повторно запрашивает источник строк

Однако если источником строк для поля со списком является таблица-справочник, простого добавления значения может оказаться недостаточно. Скорее всего, придется вывести специальную форму, в которой пользователь должен будет заполнить все необходимые поля. После сохранения записи в этой форме новые данные добавляются в список. Типичная ситуация, когда требуются такие действия — добавить нового клиента при выписке ему стандартного документа: счета, накладной и т. д.

Внесены изменения

Событие **Внесены изменения** (Dirty), так же как и **Изменение** (Change), возникает в следующих ситуациях:

- при изменении содержимого текстового поля или поля со списком, при этом изменением может быть любой непосредственно введенный или удаляемый символ;
- при изменении значения свойства **Текст** (Text) элемента управления с помощью макроса или процедуры VBA;
- в элементе управления **Набор вкладок** (Tab Control) при переходе с одной вкладки на другую.

Но в отличие от события **Изменение** (Change) оно относится к форме. Процедура имеет один параметр: `Cancel`. Если установить его значение равным `True`, то событие будет отменено. Отмена события будет вызывать откат всех изменений в записи, что эквивалентно нажатию клавиши <Esc>. Это событие удобно использовать для проверки, были ли изменения в записи.

При обновлении

Событие **При обновлении** (Updated) возникает при изменении объекта OLE и применяется только к свободным и присоединенным рамкам объекта.

Процедура обработки данного события используется для проверки, были ли данные в объекте OLE изменены после последнего сохранения. Процедура имеет один параметр — `Code`, который указывает, каким образом обновлялся объект, и может иметь значения 0, 1, 2 и 3. Значение 0 указывает, что данные объекта изменены. Значение 1 указывает, что данные объекта сохранялись приложением, создавшим объект. Значение 2 указывает, что файл объекта OLE закрывался приложением, ко-

торое его создало. Значение 3 указывает, что файл объекта OLE переименован создавшим его приложением.

Уход с записи

Событие **Уход с записи** (RecordExit) происходит всякий раз, когда пользователь пытается выйти (переместить фокус) с текущей записи: перейти к другой записи, закрыть форму, обновить данные в форме и т. д. Процедура обработки этого события может использоваться для проверки корректности данных в текущей записи. Процедура имеет один параметр: `Cancel`. Если установить его значение равным `True`, то можно запретить пользователю покидать текущую запись.

Отмена

Событие **Отмена** (Undo) происходит, когда пользователь отменяет изменения, сделанные в текущем поле или текущей записи, например, нажав клавишу <Esc> или кнопку **Undo Field/Record** на панели инструментов. Процедура обработки этого события имеет один параметр: `Cancel`. Установив его значение равным `True`, можно прервать операцию отмены, и тогда данные в текущем поле или записи останутся измененными.

События фокуса

События фокуса происходят, когда форма, отчет или элемент управления в форме получают или теряют фокус, а также когда форма или отчет становятся активными или, наоборот, неактивными.

Вход

Событие **Вход** (Enter) происходит перед тем, как элемент управления в форме получает фокус от другого элемента управления в той же форме или когда при открытии формы получает фокус первый элемент управления. Его удобно использовать для вывода на экран каких-либо сведений об этом элементе. Оно происходит до события **Получение фокуса** (GetFocus), но после события **Текущая запись** (Current).

Выход

Событие **Выход** (Exit) происходит перед тем, как данный элемент управления передаст фокус другому элементу управления той же формы, но до события **Потеря фокуса** (LostFocus).

Замечание

События **Вход** (Enter) и **Выход** (Exit) не определены для флажков и переключателей в группах, они определены только для группы как целого. События **Вход** (Enter) и **Выход** (Exit) не происходят, если фокус переходит к элементу другой формы или отчета. Это отличает их от событий **Получение фокуса** (GetFocus) и **Потеря фокуса** (LostFocus).

Получение фокуса

Событие **Получение фокуса** (GetFocus) происходит, когда форма или элемент управления формы получают фокус. Элемент управления может получить фокус только,

если оно видимо и доступно (его свойства **Вывод на экран** (Visible) и **Доступ** (Enabled) имеют значения `Да`). При этом событие **Получение фокуса** (GetFocus) происходит после события **Вход** (Enter). Форма может получить фокус, только если все поля в ней заблокированы, в противном случае событие **Получение фокуса** (GetFocus) для формы не возникает.

Потеря фокуса

Событие **Потеря фокуса** (LostFocus) происходит каждый раз, когда форма или элемент управления в форме теряют фокус. Событие происходит после события **Выход** (Exit).

Замечание

События **Получение фокуса** (GetFocus) и **Потеря фокуса** (LostFocus) происходят не только, когда фокус перемещается путем действий пользователя (с помощью клавиатуры или мыши), но и в результате вызова метода `SetFocus` объекта в форме или выполнения макрокоманд `SelectObject` (ВыделитьОбъект), `GoToRecord` (НаЗапись), `GoToControl` (КЭлементуУправления) и `GoToPage` (НаСтраницу). Кроме того, эти события определены для элементов (флажков и переключателей) внутри группы.

Включение

Событие **Включение** (Activate) возникает, когда форма или отчет получают фокус, становясь активной формой или отчетом. Это происходит, когда форма или отчет открываются, когда пользователь щелкает мышью на одном из элементов управления, перенося, таким образом, фокус, и когда в программе VBA выполняется метод `SetFocus` объекта. Форма при этом обязательно должна быть видима. Событие **Включение** (Activate) возникает до события **Получение фокуса** (GetFocus). Это событие удобно использовать для того, чтобы вывести на экран панель инструментов, связанную с формой.

Отключение

Событие **Отключение** (Deactivate) происходит, когда фокус из формы или отчета переносится на другое окно (таблицы, запроса, формы, отчета, макроса, модуля или окно базы данных). Однако это событие не возникает, когда фокус переходит в диалоговое окно или другое приложение. Событие **Отключение** (Deactivate) возникает после события **Потеря фокуса** (LostFocus).

События клавиатуры

События клавиатуры происходят в форме и элементе формы, когда пользователь нажимает клавиши на клавиатуре или же выполняется макрокоманда `SendKeys` (КомандыКлавиатуры).

Все события клавиатуры связываются с тем объектом в форме, который имеет в данный момент фокус. Обычно это один из элементов управления. Форма может получить фокус (а значит, и события клавиатуры могут относиться к форме), только если все ее элементы управления заблокированы или невидимы. Если нужно привязать эти события именно к форме, а не к элементу формы, то можно присвоить

свойству **Перехват нажатия клавиш** (KeyPreview) для формы значение **Да** (Yes). Тогда все события клавиатуры возникают сначала для формы, а потом уже для элемента управления, имеющего фокус. Это дает возможность запрограммировать реакцию формы на нажатие определенных клавиш вне зависимости от того, в каком элементе управления формы находится фокус.

Замечание

События клавиатуры не определены для элементов управления в отчетах, а также для флажков и переключателей в группах. Они определены только для группы в целом.

Клавиша вниз и *Клавиша вверх*

События **Клавиша вниз** (KeyDown) и **Клавиша вверх** (KeyUp) возникают всякий раз, когда пользователь нажимает или отпускает клавишу на клавиатуре и при этом фокус находится на элементе управления или форме. Процедуры обработки этих событий используют, когда требуется определить, какую клавишу нажал пользователь — функциональную, клавишу управления курсором, клавишу цифровой панели или комбинацию клавиш с <Shift>, <Ctrl> или <Alt>. Они имеют два параметра: `KeyCode` и `Shift`. Параметр `KeyCode` — это целое число, представляющее код нажатой клавиши. Параметр `Shift` позволяет определить, какие сочетания клавиш нажимались: 1 — соответствует <Shift>, 2 — <Ctrl>, 4 — <Alt>, 0 — не использовалось сочетание клавиш. Если же нажималась комбинация клавиш <Shift>, <Ctrl>, <Alt> в любом сочетании, то параметр `Shift` будет равен сумме значений каждой клавиши.

Нажатие клавиши

Событие **Нажатие клавиши** (KeyPress) происходит, если пользователь нажимает и отпускает любую комбинацию клавиш для элемента управления или формы, имеющей фокус. В отличие от событий **Клавиша вниз** (KeyDown) и **Клавиша вверх** (KeyUp), данное событие не происходит, когда нажимаются функциональные клавиши, клавиши управления курсором и клавиши <Shift>, <Ctrl> или <Alt>. Кроме того, эти события различны для верхнего и нижнего регистров. Процедура обработки настоящего события имеет один аргумент: `KeyAscii` — целое число, представляющее код нажатой клавиши.

Если пользователь нажимает и удерживает некоторую клавишу, то возникают повторяющиеся события **Клавиша вниз** (KeyDown) и **Нажатие клавиши** (KeyPress) до тех пор, пока он не отпустит клавишу. Тогда возникает событие **Клавиша вверх** (KeyUp).

Если результатом нажатия клавиши является перевод фокуса с одного элемента на другой, то событие **Клавиша вниз** (KeyDown) возникает для первого элемента, а события **Нажатие клавиши** (KeyPress)и **Клавиша вверх** (KeyUp) — для второго.

Если в результате нажатия клавиши появляется диалоговое окно, то возникают события **Клавиша вниз** (KeyDown) и **Нажатие клавиши**, а событие **Клавиша вверх** (KeyUp) не возникает.

События мыши

События мыши происходят, когда какое-либо действие в форме или ее элементе управления выполняется с помощью мыши. События мыши не определены для

элементов управления в отчетах, а также для флажков и переключателей в группах, они определены только для группы в целом.

Нажатие кнопки

Это наиболее широко распространенное событие. Оно возникает как в самой форме, так и в элементах управления формы. Для формы событие **Нажатие кнопки** (Click) возникает, когда пользователь щелкает мышью на пустой области формы или на области выделения записи в форме. Для элемента управления событие **Нажатие кнопки** (Click) возникает при щелчке мыши не только на самом элементе, но и на присоединенной к нему надписи. Однако для элемента управления оно возникает не только при щелчке мыши, но и в некоторых других случаях, а именно:

- при выборе элемента из списка, независимо от того, был ли он выбран с помощью мыши или клавиш управления курсором с последующим нажатием <Enter>;
- при нажатии клавиши <Пробел>, когда фокус установлен на флажке, переключателе или командной кнопке;
- при нажатии клавиши <Enter> в форме, которая содержит кнопку свойства **По умолчанию** (Default) со значением **Да** (Yes), тогда именно на эту кнопку по умолчанию устанавливается фокус;
- при нажатии клавиши <Esc> в форме, которая содержит кнопку свойства **Отмена** (Cancel) со значением **Да** (Yes);
- при нажатии клавиш доступа, если они связаны с кнопками на форме.

Таким образом, процедуры обработки событий **Нажатие кнопки** (Click) для кнопок запускаются независимо от того, каким образом эта кнопка выбрана — щелчком мыши, нажатием клавиши <Enter> или клавиши доступа. Процедура обработки события запускается только один раз. Если требуется, чтобы она запускалась несколько раз, пока кнопка остается нажатой, необходимо использовать свойство **Автоматический повтор** (AutoRepeat) для кнопки. Если нужно определить, какой кнопкой мыши выполнялся щелчок, следует использовать события **Кнопка вниз** (MouseDown) и **Кнопка вверх** (MouseUp).

Двойное нажатие кнопки

Событие **Двойное нажатие кнопки** (DblClick) происходит после быстрого двойного щелчка клавиши мыши на форме или элементе управления, при этом интервал между щелчками не должен превышать предельного времени, заданного в Панели управления Windows. Событие **Двойное нажатие кнопки** (DblClick) для формы и элемента управления формы определено так же, как и событие **Нажатие кнопки** (Click). Однако для элементов управления результат этого события зависит от типа элемента управления. По умолчанию двойной щелчок мыши в текстовом поле приводит к выделению слова, а в объекте OLE — запускает этот объект для редактирования. Вводя процедуру обработки для настоящего события, можно переопределить стандартные действия Access. Процедура имеет один параметр: `Cancel`. Если присвоить ему в процедуре значение `True`, то можно отменить это событие.

Замечание

Двойное нажатие кнопки мыши на элементе управления на самом деле вызывает сразу два события: сначала **Нажатие кнопки** (Click), а потом — **Двойное нажатие кнопки** (DblClick).

Перемещение указателя

Это событие генерируется непрерывно, когда пользователь перемещает указатель мыши по объектам формы. Пока указатель движется в границах объекта, событие **Перемещение указателя** (MouseMove) генерируется для данного объекта, когда указатель попадает на пустую область формы, область выделения записи или полосу прокрутки, генерируется событие **Перемещение указателя** (MouseMove) для формы. Событие возникает также при перемещении формы или элемента управления, например с помощью процедуры VBA, при неподвижном указателе мыши. Процедура обработки события имеет четыре параметра:

- `Button` — определяет состояние кнопок мыши в момент возникновения события (перемещение указателя может происходить при нескольких нажатых или не нажатых кнопках мыши);
- `Shift` — определяет состояние клавиш <Shift>, <Ctrl> и <Alt> в тот момент, когда нажата кнопка, определяемая параметром `Button`;
- `X` и `Y` — текущие координаты указателя мыши.

Колесико мыши

Событие **Колесико мыши** (MouseWheel) возникает, когда пользователь перемещает указатель мыши с помощью колесика скроллирования. Процедура обработки события имеет два параметра: `Page` — принимает значение `True` при перемещении указателя на другую страницу, `Count` — количество линий, на которое переместился указатель при прокрутке формы с помощью колесика мыши.

Кнопка вниз и Кнопка вверх

События **Кнопка вниз** (MouseDown) и **Кнопка вверх** (MouseUp) возникают, когда пользователь нажимает и, соответственно, отпускает кнопку мыши, и, в отличие от событий **Нажатие кнопки** (Click) и **Двойное нажатие кнопки** (DblClick), они позволяют определить, какая кнопка нажата. Процедуры обработки этих событий имеют четыре параметра: `Button`, `Shift`, `X` и `Y`. Эти параметры аналогичны параметрам процедуры для события **Перемещение указателя** (MouseMove) за исключением первого параметра — `Button`. Так как в данном случае нажимается конкретная кнопка мыши, параметр `Button` определяет, какая это кнопка. Если пользователь нажмет сразу две кнопки, то возникнут отдельно события для первой и второй кнопок.

Замечание

Если кнопка мыши была нажата, когда указатель находился на одном из элементов управления формы, то именно к этому объекту будут относиться все остальные события мыши до последнего **Кнопка вверх** (MouseUp) включительно.

События печати

События печати вызываются отчетом из любой его областей при печати или предварительном просмотре.

Форматирование

Событие **Форматирование** (Format) происходит после того, как отобраны данные для отчета, но перед тем, как фактически форматируется каждый раздел отчета. При

этом для раздела данных это событие происходит для каждой записи в отчете, что позволяет при необходимости по-разному форматировать каждую запись. Для заголовка группы в отчете событие возникает для каждой группы. Процедура обработки данного события имеет два параметра — `Cancel` и `FormatCount`. Параметр `Cancel` позволяет отменить форматирование данного раздела, для чего нужно присвоить ему значение `True`. Это дает вам возможность пропускать разделы отчета, не оставляя пустого места на странице. Параметр `FormatCount` — счетчик, который считает, сколько раз произошло событие **Форматирование** (Format).

Возврат

Событие происходит, если при форматировании раздела требуется вернуться к разделу, который уже отформатирован. Оно происходит после события **Форматирование** (Format), но до события **Печать** (Print). Процедура обработки данного события позволяет изменить любое уже выполненное форматирование и обеспечить, таким образом, нужное расположение элементов отчета на странице.

Замечание

Событие **Возврат** (Retreat) не определено для верхних и нижних колонтитулов отчета.

Печать

Событие **Печать** (Print) возникает после того, как выполнено форматирование раздела отчета, но до того, как он напечатан. Это событие возникает практически после каждого события **Форматирование** (Format), кроме тех разделов, которые не будут печататься. Так же как и при форматировании, процедура обработки событий имеет два аргумента — `Cancel` и `PrintCount`. Параметр `Cancel` позволяет отменить печать текущего раздела или текущей записи в отчете, для чего нужно присвоить ему значение `True`. Однако при этом остается пустое место на странице, поэтому данную процедуру можно использовать, когда изменения не касаются формата страницы отчета. Параметр `PrintCount` — счетчик, который считает, сколько раз произошло событие **Печать** (Print).

Страница

Событие **Страница** (Page) возникает после форматирования страницы отчета, но до вывода ее на печать и позволяет с помощью процедуры обработки этого события добавить на страницу некоторые элементы оформления, например рамку.

Отсутствие данных

Событие **Отсутствие данных** (No Data) возникает после форматирования отчета, но до его вывода на печать (до первого события **Страница** (Page)) и позволяет обнаружить отсутствие записей для отчета, в этом случае печать можно отменить. Процедура обработки данного события имеет один параметр `Cancel`, которому следует присвоить значение `True`, если нужно отменить печать отчета.

События фильтра

События фильтра происходят при использовании или удалении фильтра в форме.

Применение фильтра

Событие **Применение фильтра** (ApplyFilter) возникает во всех случаях, когда пользователь выполняет фильтрацию записей в форме с помощью соответствующих команд меню, контекстного меню или кнопки панели инструментов (применить или удалить фильтр). Программу обработки этого события обычно используют либо для проверки условия в фильтре, либо для изменения вида формы перед применением фильтра, если требуется скрыть лишние поля или, наоборот, показать скрытые. Программа обработки события имеет два параметра — `Cancel` и `ApplyType`. Параметр `Cancel` позволяет отменить операцию фильтрации, если, например, условие сформулировано неправильно, для чего нужно присвоить ему значение `True`. Параметр `ApplyType` определяет исполняемое действие и может принимать значения 0, 1 и 2. Значение 0 указывает на удаление фильтра, 1 — на применение фильтра, 2 — закрытие окна фильтра.

Замечание

Событие не возникает при выполнении операций фильтрации с помощью макрокоманд `ApplyFilter` (ПрименитьФильтр), `OpenForm` (ОткрытьФорму), `ShowAllRecords` (ПоказатьВсеЗаписи), соответствующих им методов объекта `DoCmd` или при закрытии окна фильтра (любого) с помощью макрокоманды `Close` (Закрыть).

Фильтрация

Событие **Фильтрация** (Filter) возникает перед открытием окна фильтра или расширенного фильтра, когда пользователь пытается выполнить команду **Изменить фильтр** (Filter by Form). Использовать это событие очень удобно, если требуется, например, ввести в фильтр некоторые условия по умолчанию или запретить включать в условия отбора некоторые поля. Чтобы запретить включать некое поле в условие отбора в окне фильтра, достаточно скрыть его в процедуре обработки события **Фильтрация** (Filter). Правда, это относится только к окну обычного фильтра, т. к. в окне расширенного фильтра выводятся все поля, в том числе и скрытые. Можно даже заменить стандартное окно фильтра своим собственным, в котором пользователь и будет задавать условия отбора. Процедура обработки события имеет два параметра — `Cancel` и `FilterType`. Параметр `Cancel` позволяет отменить открытие стандартного окна фильтра, если вместо него будет выводиться специальная форма, для чего нужно присвоить ему значение `True`. Параметр `FilterType` определяет, какое окно открывается, и может принимать значения 0 или 1. Значение 0 соответствует обычному фильтру, 1 — расширенному фильтру.

События окна

События окна запускаются при открытии и закрытии форм и отчетов, а также при изменении размеров формы.

Открытие

Событие **Открытие** (Open) происходит после того, как выполнен запрос, лежащий в основе формы или отчета, но до отображения первой записи или печати отчета. Процедура обработки этого события имеет один параметр — `Cancel`, при установке которого в значение `True` отменяется открытие формы или отчета. Обычно процедура обработки события **Открытие** (Open) используется для проверки условий и

предотвращения открытия формы, т. к. следующее по времени событие **Загрузка** (Load) уже не может быть отменено.

Закрытие

Событие **Закрытие** (Close) является последним перед тем, как форма будет удалена с экрана. Обычно его используют для открытия другой формы. Для отчета событие происходит, когда закрывается режим Предварительного просмотра или заканчивается печать отчета. Как и в случае с формой, его можно использовать для определения дальнейших действий пользователя.

Загрузка

Событие **Загрузка** (Load) происходит сразу после события **Открытие** (Open), но в отличие от него не может быть отменено. Обычно его используют для динамического изменения свойств формы или элементов управления перед тем, как форма будет выведена на экран.

Выгрузка

Событие **Выгрузка** (Unload) происходит при закрытии формы до события и может быть отменено. Обычно это событие используется для проверки различных условий, которые определяют, можно ли закрывать форму. Процедура обработки этого события имеет один параметр — `Cancel`, при установке которого в значение `True` отменяется закрытие формы.

Замечание

Если вы используете процедуру обработки события **Выгрузка** (Unload), в которой параметру `Cancel` присваивается значение `True`, не забудьте явно присвоить ему значение `False` в случае выполнения всех условий для закрытия формы. Иначе после того, как этот параметр будет установлен в `True`, форму нельзя будет закрыть никогда.

Изменение размера

Событие **Изменение размера** (Resize) возникает при открытии формы и при изменении ее размеров. Его обычно применяют, если требуется подстроить размер элементов управления под изменяющиеся размеры формы или рассчитать заново вычисляемые элементы. Если нужно, чтобы при каждом изменении размеров формы происходило обновление экрана, используйте в процедуре обработки этого события метод `Repaint`.

Замечание

При создании процедур обработки событий часто возникают сомнения, какое из двух событий — **Открытие** (Open) или **Загрузка** (Load) — использовать (или, соответственно, **Закрытие** (Close) или **Выгрузка** (Unload)). Рекомендуется принимать во внимание следующее соображение: если требуется возможность отменить событие, то пользуйтесь событиями **Открытие** (Open) и **Выгрузка** (Unload), в противном случае можно использовать любое.

События ошибок

Событие **Ошибка** (Error) возникает, когда в процессе обработки формы или отчета ядром Access возникает ошибка. В процедуре обработки этого события можно перехватить стандартное сообщение об ошибке, которое выдает Access, и выдать собственное сообщение. Процедура имеет два параметра — `DataErr` и `Response`. Параметр `DataErr` содержит код ошибки, а параметр `Response` может принимать два значения — 0 и 1. Значение 0 отменяет выдачу стандартного сообщения об ошибке, а 1 — позволяет его отобразить.

Замечание

Это событие не возникает, когда ошибка встречается в коде VBA.

События таймера

Событие **Таймер** (Timer) возникает регулярно через интервал времени, который задается свойством **Интервал таймера** (TimerInterval) формы. Оно позволяет определять действия, которые должны выполняться периодически по сигналу таймера. Обычно используется для регулярных обновлений экрана в многопользовательском приложении, тогда в процедуре обработки события **Таймер** (Timer) нужно использовать метод `Requery`, который будет выполнять повторный запрос источника данных формы.

События режима Сводная таблица

Довольно большой набор событий существует специально для режимов сводной таблицы и сводной диаграммы. Даже краткое описание этих событий, которое мы приводим далее, показывает, насколько гибко можно программно управлять этими режимами.

До экранной подсказки

Событие **До экранной подсказки** (BeforeScreenTip) возникает перед тем, как на каком-то элементе сводной таблицы или диаграммы появляется подсказка. Позволяет изменить текст подсказки. Процедура обработки этого события имеет два параметра: `TipText`, который определяет текст подсказки, и `ContextObject`, который является ссылкой на объект, связанный с подсказкой.

Включение команды

Событие **Включение команды** (CommandEnabled) происходит, когда Web Component проверяет, доступна ли команда, указанная в первом параметре процедуры — `Command`. Второй параметр — `Enabled` — является выходным. Это объект, значение (свойство `Value`) которого устанавливается равным `True` или `False`, в зависимости от того, включена или нет команда.

Проверка команды

Событие **Проверка команды** (CommandChecked) происходит, когда Web Component определяет, проверена ли команда, указанная в первом параметре процедуры, — `Command`. Второй параметр — `Checked` — является выходным. Это объект, значение

которого устанавливается равным `True` или `False`, в зависимости от того, проверена или нет команда.

До выполнения команды

Событие **До выполнения команды** (CommandBeforeExecute) возникает до выполнения команды, указанной в первом параметре процедуры — `Command`. Второй параметр — `Cancel` — устанавливается равным `True`, если нужно отменить выполнение команды.

Выполнение команды

Событие **Выполнение команды** (CommandExecute) возникает после выполнения команды, указанной в параметре процедуры — `Command`. Позволяет выполнить нужные действия после выполнения команды.

Изменение данных

Событие **Изменение данных** (DataChange) возникает после изменения любого свойства или выполнения метода в сводной таблице или диаграмме. Позволяет определить действия, после возникновения такого изменения или выполнения метода. Параметр процедуры обработки события — `Reason` — представляет собой одну из констант, указывающих причину, т. е. что привело к возникновению данного события.

Изменение набора данных

Событие **Изменение набора данных** (DataSetChange) возникает всякий раз, когда изменяется набор записей для сводной таблицы или диаграммы, например, накладывается фильтр. Позволяет выполнить какие-то действия после такого изменения.

Изменение сводной таблицы

Событие **Изменение сводной таблицы** (PivotTableChange) возникает после добавления или удаления полей или итоговых данных, т. е. при изменении структуры сводной таблицы. Позволяет определить действия, после возникновения такого изменения. Параметр процедуры обработки события — `Reason` — представляет собой одну из констант, указывающих, что произошло с таблицей и привело к возникновению данного события.

Изменение выбора фрагмента

Событие **Изменение выбора фрагмента** (SelectionChange) возникает всякий раз, когда пользователь меняет выборку в сводной таблице. Эта процедура не имеет параметров и не может быть отменена.

Изменение представления

Событие **Изменение представления** (ViewChange) возникает всякий раз, когда сводная таблица или сводная диаграмма перерисовываются. Параметр `Reason` представляет собой одну из констант, указывающих причину, т. е. что привело к возникновению данного события.

Соединение

Событие **Соединение** (OnConnect) возникает всякий раз, когда происходит соединение с источником данных для сводной таблицы или диаграммы. Параметров не имеет.

Отсоединение

Событие **Отсоединение** (OnDisconnect) возникает всякий раз, когда происходит отсоединение от источника данных для сводной таблицы. Параметров не имеет.

До запроса

Событие **До запроса** (BeforeQuery) возникает перед любым запросом исходных данных. Параметров не имеет.

Запрос

Событие **Запрос** (Query) происходит всякий раз при выполнении запроса исходных данных. Позволяет отслеживать запросы к данным для сводной таблицы или диаграммы.

До разметки

Событие **До разметки** (AfterLayout) возникает после того, как диаграммы на сводной диаграмме уже подготовлены, но еще не выведены. Еще можно повлиять на размещение диаграммы на листе или на другие параметры, изменяя свойства объекта, ссылка на который передается в параметре процедуры — `drawObject`.

До отображения

Событие **До отображения** (BeforeRende) возникает после события **До разметки** (AfterLayout) и позволяет добавить на поле сводной диаграммы разные объекты — линии, прямоугольники, эллипсы, текст. Процедура имеет три параметра: `drawObject` — объект `chChartDraw`, который представляет собой саму сводную диаграмму, `chartObject` — объект, который должен быть размещен на диаграмме, `Cancel` — объект, который позволяет прервать отображение объекта на диаграмме. Для этого нужно присвоить значение `False` свойству `Value` объекта `Cancel`.

После отображения

Событие **После отображения** (AfterRender) возникает после того, как объект размещен на сводной диаграмме. Процедура имеет два параметра: `drawObject` — объект `chChartDraw`, а второй — `chartObject` — объект, который был размещен на диаграмме.

После конечного отображения

Событие **После конечного отображения** (AfterFinalRender) возникает после того, как все объекты на сводной диаграмме отобразились. Имеет один аргумент — `drawObject` — объект `chChartDraw`.

Последовательности событий Access

При написании процедур обработки событий очень важно понимать, в каком порядке они происходят, т. к. приложение Access управляется событиями и результат работы зависит от того, в каком порядке эти процедуры будут выполняться. В данном разделе мы опишем порядок возникновения событий в формах и отчетах.

Последовательность событий фокуса для элементов управления в формах

При установке фокуса на элемент управления щелчком мыши, при нажатии клавиши <Tab> или при открытии формы происходят события:

Вход (Enter) ⇒ **Получение фокуса** (GotFocus).

Когда элемент теряет фокус, например при закрытии формы или переносе фокуса на другой элемент управления той же формы, происходят события:

Выход (Exit) ⇒ **Потеря фокуса** (LostFocus).

При переносе фокуса в другую форму элемент первой формы теряет фокус, но не возникает события **Выход** (Exit). Соответственно, когда фокус переносится обратно, т. е. к элементу, который уже имел фокус, когда форма последний раз была активной, то не происходит события **Вход** (Enter), а только событие **Получение фокуса** (GotFocus).

Замечание

В дальнейшем при описании последовательностей событий мы не будем приводить английские эквиваленты названий событий, дабы не загромождать представление. Кроме того, вводятся сокращения для обозначения объектов, к которым относится событие, например: э — элемент, ф — форма.

При переносе фокуса из элемента управления в главной форме на элемент управления в подчиненной форме события происходят в следующем порядке:

Выход (г.ф., т.ф.) ⇒ **Потеря фокуса** (г.ф., т.ф.) ⇒ **Вход** (г.ф., э.п.) ⇒ **Выход** (п.ф., т.ф.) ⇒ **Потеря фокуса** (п.ф., т.ф.) ⇒ **Вход** (п.ф., п.ф.) ⇒ **Получение фокуса** (п.ф., п.ф.).

Это самая сложная цепочка событий, в которой задействованы сразу четыре элемента управления: г.ф., т.ф. — элемент главной формы, теряющий фокус; г.ф., э.п. — элемент главной формы, содержащий подформу; п.ф., т.ф. — элемент подформы, теряющий фокус (если мы переносим фокус на тот элемент подформы, который до этого фокуса не имел); п.ф., п.ф. — элемент подформы, принимающий фокус.

Если перенос фокуса выполняется на элемент подчиненной формы, который имел фокус, когда она последний раз была активной, то в цепочке отсутствуют события: **Выход** (п.ф., т.ф.) ⇒ **Потеря фокуса** (п.ф., т.ф.) ⇒ **Вход** (п.ф., п.ф.).

Последовательность событий изменения данных в элементе управления

Когда вводят или изменяют данные в элементе управления, а затем переходят к следующему элементу управления, возникает следующая цепочка событий:

{**Клавиша вниз** ⇒ **Нажатие клавиши** ⇒ **Внесены изменения** ⇒ **Изменение** ⇒ **Клавиша вверх**}⇒ **До обновления** ⇒ **После обновления** ⇒ **Выход** ⇒ **Потеря фокуса**.

Фигурные скобки в приведенном выражении означают, что выделенная цепочка событий возникает при каждом нажатии клавиши на клавиатуре.

Если ввод выполняется в поле со списком, то в конец последней цепочки, после события **Клавиша вверх**, могут добавиться события **Отсутствие в списке** (NotInList) и **Ошибка** (Error), и цепочка будет выглядеть следующим образом:

{**Клавиша вниз** ⇒ **Нажатие клавиши** ⇒ **Внесены изменения** ⇒ **Изменение** ⇒ **Клавиша вверх**} ⇒ **Отсутствие в списке** ⇒ **Ошибка** ⇒ **До обновления** ⇒ **После обновления** ⇒ **Выход** ⇒ **Потеря фокуса**.

Последовательность событий в форме, связанных с записями

Если просто просматриваются записи в форме, то при переходе к каждой новой записи выполняются событие **Текущая запись** (Current) для формы и все события, связанные с установкой фокуса в элементах формы. Если данные в записи меняются, то сохранение изменений происходит только при переходе к следующей записи или при закрытии формы (если нашли нужную запись, изменили и закрыли форму). Поскольку все изменения в записи происходят в элементах управления формы, одновременно будут возникать соответствующие события в элементах формы. Рассмотрим типичную ситуацию, когда в форме выводится запись, в ней перемещаются по элементам управления до нужного элемента, изменяют в нем данные и переходят к следующей записи. В этом случае последовательность возникновения событий будет выглядеть так:

Текущая запись (ф) ⇒ {**Вход** (э) ⇒ **Получение фокуса** (э) ⇒ **Выход** (э) ⇒ **Потеря фокуса** (э)} ⇒ **Вход** (э) ⇒ **Получение фокуса** (э) ⇒ **До обновления** (э) ⇒ **После обновления** (э) ⇒ **До обновления** (ф) ⇒ **После обновления** (ф) ⇒ **Выход** (э) ⇒ **Потеря фокуса** (э) ⇒ **RecordExit** ⇒ **Текущая запись** (ф) ⇒ **Вход** (э) ⇒ **Получение фокуса** (э).

В этой цепочке для упрощения опущены все события клавиатуры, а фигурные скобки выделяют цепочки событий, которые возникают при переходе между элементами управления формы. При изменении данных события **До обновления** (BeforeUpdate) и **После обновления** (AfterUpdate) происходят сначала для элемента, а затем для формы. Затем последний элемент (в данном случае тот, в котором происходили изменения) теряет фокус, происходит событие **RecordExit** (Выход из записи), выводится следующая запись, и фокус устанавливается на первый элемент в этой записи. События **До обновления** (BeforeUpdate) и **После обновления** (AfterUpdate) всегда возникают непосредственно перед переходом к следующей записи. После этого запись (элемент управления, в котором перед этим находился фокус) теряет фокус.

Замечание

Если запись сохраняется с помощью команды меню **Записи | Сохранить запись** (Records | Save Record), то последних событий **Выход** (Exit) и **Потеря фокуса** (LostFocus) не происходит. Это полезно знать, т. к. иногда требуется обойти эти события.

При удалении записи происходят события:

Удаление ⇒ **До подтверждения Del** ⇒ **После подтверждения Del** ⇒ **Текущая запись** ⇒ **Вход** ⇒ **Получение фокуса**.

Если событие **Удаление** (Delete) отменяется, то остальные события не возникают. После удаления записи фокус переходит на следующую запись, поэтому происходят события **Текущая запись** (Current) формы и **Вход** (Enter) и **Получение фокуса** (GetFocus) первого элемента в этой записи.

Добавление новой записи осуществляется после того, как пользователь вводит первый символ новой (пустой) записи. При этом события происходят в следующем порядке:

Текущая запись (ф) ⇒ **Вход** (э) ⇒ **Получение фокуса** (э) ⇒ **До вставки** (ф) ⇒ **Изменение** (э) ⇒ **До обновления** (ф) ⇒ **После обновления** (ф) ⇒ **После вставки** (ф).

В этой цепочке пропущены все события клавиатуры и **До обновления** (BeforeUpdate) и **После обновления** (AfterUpdate) для элементов формы, т. к. они описаны ранее. Событие **Изменение** (Change) происходит, если первый символ новой записи вводится в текстовое поле.

Последовательность событий в формах

При открытии формы происходит следующая цепочка событий:

Открытие ⇒ **Загрузка** ⇒ **Изменение размера** ⇒ **Включение** ⇒ **Текущая запись** ⇒ **Вход** ⇒ **Получение фокуса**.

События **Вход** (Enter) и **Получение фокуса** (GetFocus) возникают для первого элемента в первой записи. Если форма не имеет видимых или доступных элементов управления, то последовательность событий несколько иная:

Открытие ⇒ **Загрузка** ⇒ **Изменение размера** ⇒ **Включение** ⇒ **Получение фокуса** ⇒ **Текущая запись**.

В этом случае событие **Получение фокуса** (GetFocus) относится к форме.

При закрытии формы последовательность событий следующая:

Выход ⇒ **Потеря фокуса** ⇒ **Выгрузка** ⇒ **Отключение** ⇒ **Закрытие**.

То есть сначала теряет фокус последний элемент в форме, а затем выполняются события для формы. Если в форме не было видимых или доступных элементов, то последовательность другая:

Выгрузка ⇒ **Потеря фокуса** (ф) ⇒ **Отключение** ⇒ **Закрытие**.

При переключении между двумя открытыми формами последовательность событий следующая:

Потеря фокуса (э1) ⇒ **Отключение** (ф1) ⇒ **Включение** (ф2) ⇒ **Вход** (э2) ⇒ **Получение фокуса** (э2).

Если из открытой формы открывают другую форму, то сначала в открываемой форме происходят события **Открытие** (Open), **Загрузка** (Load) и **Изменение размера** (Resize), и только после этого в первой форме произойдет событие **Отключение** (Deactivate):

Открытие (ф2) ⇒ **Загрузка** (ф2) ⇒ **Изменение размера** (ф2) ⇒ **Отключение** (ф1) ⇒ **Включение** (ф2) ⇒ **Текущая запись** (ф2).

Это дает возможность проконтролировать открытие второй формы. Кроме того, событие **Включение** (Activate) происходит каждый раз, когда форма получает фокус, а события **Открытие** (Open) и **Загрузка** (Load) не происходят, если форма уже открыта, даже если переход в эту форму выполняется с помощью макрокоманды `OpenForm` (ОткрытьФорму). Событие **Отключение** (Deactivate) для формы не происходит, если фокус переносится в диалоговое окно.

Если открываемая форма содержит подчиненные формы, то сначала загружаются подчиненные формы и осуществляются все события в них, которые обычно происходят при открытии формы, кроме события **Включение** (Activate) — оно не возникает. Затем загружается главная форма в обычном порядке и для нее выполняется событие **Включение** (Activate).

Аналогично при закрытии такой формы сначала выгружаются все подчиненные формы, но в них не возникает событие **Отключение** (Deactivate). События при этом происходят в следующем порядке:

1. События для элементов подчиненной формы, например **Выход** (Exit) и **Потеря фокуса** (LostFocus).
2. События элементов управления в главной форме, в том числе для того элемента, который содержит подчиненную форму.
3. События для главной формы.
4. События для подчиненной формы.

Последовательность событий клавиатуры и мыши

При нажатии и отпускании любой клавиши на клавиатуре в том случае, если фокус находится в одном из элементов управления формы, возникает следующая цепочка событий:

Клавиша вниз ⇒ **Нажатие клавиши** ⇒ **Клавиша вверх**.

При щелчке мышью на элементе управления формы, соответственно:

Кнопка вниз ⇒ **Кнопка вверх** ⇒ **Нажатие кнопки**.

Об особенностях этих событий подробно говорилось ранее при их описании, поэтому сейчас мы позволим себе не повторяться.

Последовательность событий в отчетах

В отчетах события возникают только для самого отчета и его разделов. Для полей отчета события отсутствуют. При выводе отчета на печать события обычно возникают в следующей последовательности:

Открытие ⇒ **Включение** ⇒ {**Формат** ⇒ **Печать**} ⇒ **Закрытие** ⇒ **Отключение**.

События **Форматирование** (Format) и **Печать** (Print) относятся к разделам отчета и возникают при обработке каждого раздела. Дополнительными событиями могут быть события:

- **Отсутствие данных** (NoData), которое возникает перед первым событием **Печать** (Print), если источник записей не содержит записей.
- **Страница** (Page), которое возникает после всех событий **Печать** (Print) для страницы отчета.
- **Возврат** (Retreat) возникает в процессе форматирования страницы отчета, когда требуется вернуться к предыдущему разделу, перед событиями **Печать** (Print) для страницы.

Создание процедур обработки событий

Теперь рассмотрим, как создавать процедуры обработки событий. Для большинства элементов управления формы, а также самой формы и отчета, стандартный набор действий следующий:

1. Откройте форму в режиме Конструктора. Если при этом окно свойств отсутствует на экране, щелкните кнопку **Свойства** (Properties) на панели инструментов.
2. Выберите нужный элемент управления (или щелкните мышью на маленьком черном квадрате в верхнем левом углу формы, тогда выберется вся форма). В окне свойств отобразятся свойства выбранного элемента.
3. Откройте вкладку **События** (Events).
4. Выберите событие, для которого будет создаваться процедура обработки, и щелкните по нему правой кнопкой мыши.
5. Выберите из контекстного меню (рис. 13.36) пункт **Построить** (Build). В открывшемся диалоговом окне **Построитель** (Choose Builder) выберите из списка элемент **Программы** (Code Builder) и нажмите кнопку **ОК**. Откроется окно редактора VBA, в котором появятся первая и последняя строки процедуры (рис. 13.37).

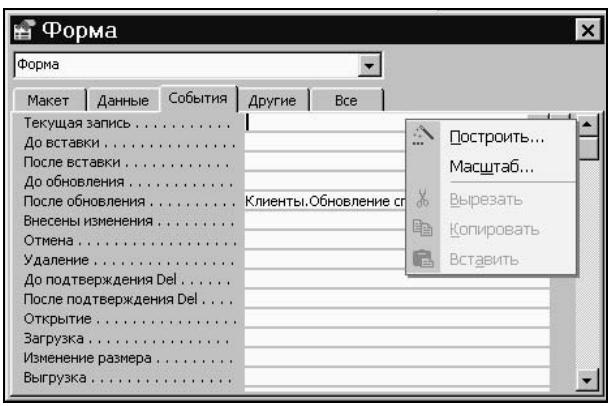


Рис. 13.36. Диалоговое окно событий формы

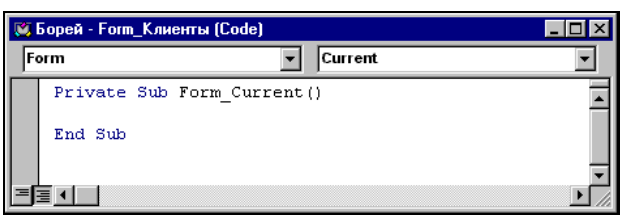


Рис. 13.37. Заготовка процедуры обработки события

Если процедура обработки выбранного события имеет аргументы, они будут также присутствовать в заголовке процедуры (рис. 13.38).

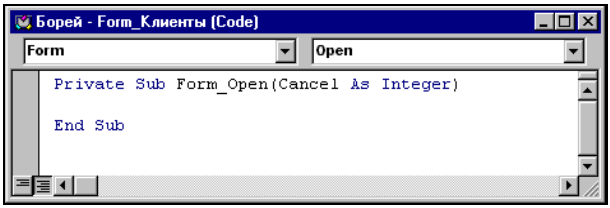


Рис. 13.38. Процедура обработки событий с аргументом `Cancel`

Теперь нужно ввести код процедуры между этими двумя строками. Далее приведены некоторые примеры процедур обработки событий.

Синхронизация данных в связанных формах

Пример синхронизации данных в двух связанных формах с помощью процедуры обработки события **Текущая запись** (On Current) можно увидеть в модуле формы `Поставщики` (Supplier). В форме есть кнопка **Просмотр товаров** (Review Products), при нажатии на которую выводится форма `Список товаров` (Product List), показывающая товары данного поставщика. Естественно, что при переходе к новой записи в форме `Поставщики` (Supplier), записи в форме `Список товаров` (Product List) тоже должны быть сменены. Процедура представлена в листинге 13.1.

Листинг 13.1. Процедура обработки события *Текущая запись*

```
Private Sub Form_Current()
On Error GoTo Err_Form_Current

' Отображение товаров текущего поставщика при открытии формы "Список
' товаров".

    Dim strDocName As String
    Dim strLinkCriteria As String

        strDocName = "Список товаров"
        strLinkCriteria = "[КодПоставщика] =
Forms![Поставщики]![КодПоставщика] "

    If IsNull(Me![Название]) Then
        Exit Sub
    ElseIf IsLoaded("Список товаров") Then
        DoCmd.OpenForm strDocName,,, strLinkCriteria
    End If

Exit_Form_Current:
    Exit Sub

Err_Form_Current:
    MsgBox Err.Description
    Resume Exit_Form_Current

End Sub
```

Если текущая запись в форме Поставщики (Supplier) пустая, т. е. добавляется новая запись, то сразу выполняется выход из процедуры. Если текущая запись отображает конкретного поставщика, то проверяется, загружена ли форма Список товаров (Product List). Если форма загружена, то меняется набор записей в ней. Делается это с помощью макрокоманды OpenForm (ОткрытьФорму) с соответствующим условием отбора записей. При этом на самом деле форма не открывается, просто повторно запрашивается источник данных.

Проверка дублирования значений первичного ключа

Обычно проверка дублирования значений первичного ключа задается на уровне таблицы, т. е. когда вы определили поле в таблице как первичный ключ, значение свойства **Индексированное поле** (Indexed) автоматически устанавливается равным **Да (Совпадения не допускаются)**. Однако эта проверка выполняется только тогда, когда запись сохраняется в базе данных. Если значение ключевого поля вводится в форме пользователем, как, например, в таблице Клиенты (Customers), то такую проверку лучше выполнить сразу после ввода данных в это поле. Наиболее подходящим событием для этого является событие **До изменения** (On BeforeUpdate). В форме Клиенты (Customers) элемент управления КодКлиента (CustomerID) содержит идентификатор клиента. Событие **До изменения** (Before Update) этого поля обрабатывается с помощью макроса Клиенты (Customers.ValidateID), который выполняет необходимую проверку. Мы покажем, как можно обработать это событие с помощью процедуры VBA. Данная процедура представлена в листинге 13.2.

Листинг 13.2. Процедура обработки события *До изменения*

```
Private Sub КодКлиента_BeforeUpdate (Cancel As Integer)
    Dim rs As Recordset
    Set rs = CurrentDB.Openrecordset("Клиенты",dbOpenTable)
    rs.Index = "PrimaryKey"
    rs.Seek "=", Me!КодКлиента
    if Not rs.NoMatch Then
        MsgBox "Клиент с таким идентификатором уже существует в базе"
        Cancel = True
    End If
    rs.Close
End Sub
```

Поиск записи со значением ключа, совпадающим с введенным значением поля КодКлиента (CustomerID), выполняется с помощью метода Seek объекта Recordset. Этот метод обеспечивает быстрый поиск необходимой записи. Применить его можно только к набору записей табличного типа, поэтому при создании этого набора записей используется внутренняя константа dbOpenTable. Если такая запись найдена, свойство NoMatch объекта Recordset принимает значение False. В этом случае процедура выведет сообщение, что такой пользователь уже существует, и присвоит значение True аргументу Cancel. Это позволяет отменить обновление значения элемента управления. Если значение свойства NoMatch объекта Recordset равно True,

процедура закрывает набор записей (рекомендуется не забывать это делать) и завершает свою работу.

Создание перекрестного отчета с изменяющимся числом столбцов

Рассмотрим возможности обработки событий в отчете на примере отчета `Выработка сотрудников`. Этот отчет строится на базе перекрестного запроса и показывает выработку сотрудников отдела продаж за год по месяцам. Максимальное число столбцов в отчете — 14. Первый столбец содержит фамилии сотрудников, следующие двенадцать столбцов — выработку для каждого месяца и последний столбец — итоговый. *(Как выглядит этот отчет, мы показывали в разд. "Перекрестные отчеты" гл. 10.)*

В качестве источника данных для такого запроса используется перекрестный запрос `Выработка сотрудников`, представленный на рис. 13.39.

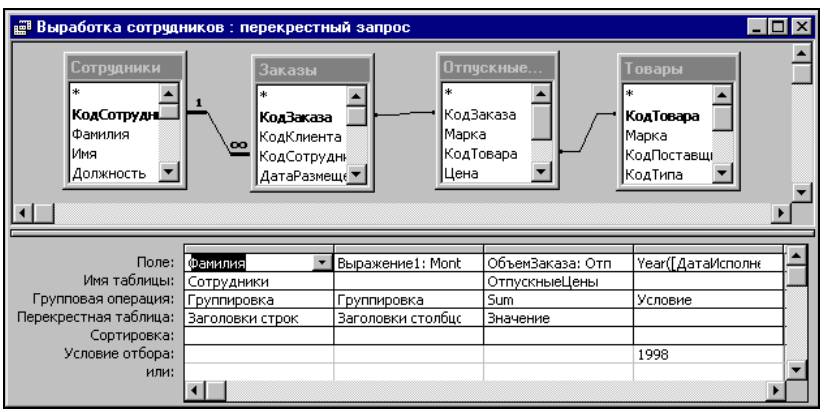


Рис. 13.39. Перекрестный запрос `Выработка сотрудников`

Поле `Отпускная цена` является вычисляемым, и для его расчета используется формула

```
CCur(CLng([Заказано].[Цена]*[Количество]*(1-[Скидка])*100)/100)
```

Создание такого отчета требует написания довольно большого числа процедур обработки событий.

В запросе в качестве критерия выборки заказов указан год — 1998. Нужно вывести приглашение пользователю, чтобы он, в свою очередь, мог ввести желаемый год. Это можно сделать при открытии отчета. Сначала нужно создать базовый набор записей для отчета и посчитать, сколько получилось столбцов в этом наборе. Процедура, обрабатывающая событие **Открытие** (On Open) отчета, представлена в листинге 13.3.

Листинг 13.3. Процедура обработки события *Открытие* отчета

```
Private Sub Report_Open(Cancel As Integer)
On Error Resume Next
```

```
    ' Создает базовый набор записей для отчета.
    Dim intX As Integer
    Dim qdf As QueryDef
    Dim frm As Form
    Dim StrSql As String

    ' Связывает переменную с текущей базой данных.
    Set dbsReport = CurrentDb
    ' Открывает запрос (объект QueryDef).
    Set qdf = dbsReport.QueryDefs("Выработка сотрудников")
    ' Запрашивает год.
    Год = InputBox("Отчет за год:", "Год", 1998)
    StrSql = Left(qdf.SQL, InStr(qdf.SQL, "where") - 1) & " WHERE_
       (((Year([ДатаИсполнения]))= " & Год & "))" & Right(qdf.SQL,
       Len(qdf.SQL) - InStr(qdf.SQL, "GROUP BY") + 1)
    qdf.SQL = StrSql
    ' Открывает набор записей.
    Set rstReport = qdf.OpenRecordset()

    ' Определяет количество столбцов в перекрестном запросе.
    intColumnCount = rstReport.Fields.Count

End Sub
```

Здесь переменная `intColumnCount` должна быть определена на уровне модуля формы, т. к. она используется и другими процедурами данной формы.

Для событий **Форматирование** (On Format) верхнего колонтитула (листинг 13.4) и области данных отчета (листинг 13.5) необходимо определить процедуры, которые бы заполняли поля заголовков и значений и скрывали неиспользуемые поля.

Листинг 13.4. Процедура обработки события *Форматирование* верхнего колонтитула

```
Private Sub PageHeader0_Format(Cancel As Integer,_
                                    FormatCount As Integer)
    Dim intX As Integer

    ' Помещает заголовки столбцов в поля в верхнем колонтитуле.
    Me("Head" + Format(0)) = rstReport(0).Name
    For intX = 1 To intColumnCount - 1
        Me("Head" + Format(intX)) =_
                        MonthRus(CInt(rstReport(intX).Name))
    Next intX

    ' Вводит в ближайшее свободное поле заголовок "Итого".
    Me("Head" + Format(intColumnCount)) = "Итого"

    ' Скрывает пустые поля в верхнем колонтитуле.
    For intX = (intColumnCount + 1) To conTotalColumns - 1
```

```
        Me("Head" + Format(intX)).Visible = False
    Next intX
End Sub
```

В этой процедуре используется функция `MonthRus`, которая по порядковому номеру месяца выдает его название. Мы здесь не приводим текста этой функции, надеясь, что вы сможете написать ее сами.

`conTotalColumns` — константа, которая описана на уровне модуля и значение которой определяет максимальное число столбцов в отчете. В данном примере `conTotalColumns` равна 14.

Листинг 13.5. Процедура обработки события *Форматирование* области данных

```
Private Sub Detail1_Format(Cancel As Integer,_
                                    FormatCount As Integer)
    ' Вводит значения в поля и скрывает пустые поля.
    Dim intX As Integer
    ' Проверяет, что не достигнут конец набора записей.
    If Not rstReport.EOF Then
        ' Помещает значения из набора записей в поля области данных.
        If Me.FormatCount = 1 Then
            For intX = 0 To intColumnCount - 1
                ' Преобразует пустые значения в 0.
                Me("Col" + Format(intX)) =_
                                    xtabCnulls(rstReport(intX))
            Next intX
            ' Скрывает неиспользуемые поля в области данных.
            For intX = intColumnCount + 1 To conTotalColumns - 1
                Me("Col" + Format(intX)).Visible = False
            Next intX
            ' Переходит к следующей записи в наборе.
            rstReport.MoveNext
        End If
    End If
End Sub
```

Свойство `FormatCount` отчета содержит значение, равное количеству столбцов для форматирования, поэтому в процедуре проверяется это свойство, и присвоение значений полям в области данных выполняется, только если это значение не 0. В процедуре используется также функция `xtabCnulls`, преобразующая пустое значение в 0.

Для события **Печать** (On Print) области данных отчета нужно создать процедуру, которая бы суммировала значения по строке отчета, выводила полученное значение в последнем столбце и, кроме того, добавляла эту сумму в массив итоговых значений по столбцам (листинг 13.6). Массив итоговых значений по столбцам `lngRgColumnTotal` и переменная `lngReportTotal`, определяющая общий итог, должны быть описаны на уровне модуля. Кроме того, они должны быть инициализиро-

ваны, т. е. им нужно присвоить начальные значения 0. Это можно сделать в процедуре обработки события **Загрузка** (On Load) отчета.

Листинг 13.6. Процедура обработки события *Печать* области данных

```
Private Sub Detail1_Print(Cancel As Integer, PrintCount As Integer)
    Dim intX As Integer
    Dim lngRowTotal As Long

    ' Вычисляет сумму по строке и добавляет ее к итоговому значению
    ' по столбцу и общему итогу.
    If Me.PrintCount = 1 Then
        lngRowTotal = 0
        For intX = 1 To intColumnCount - 1
            ' Начиная со столбца 1 (первый столбец с перекрестными
            ' значениями), вычисляет сумму по строке.
            lngRowTotal = lngRowTotal + Me("Col" + Format(intX))
            ' Добавляет итоговое значение для текущего столбца.
            lngRgColumnTotal(intX) = lngRgColumnTotal(intX) +_
                                     Me("Col" + Format(intX))
        Next intX
        ' Заносит сумму по строке в поле в области данных.
        Me("Col" + Format(intColumnCount)) = lngRowTotal
        ' Прибавляет сумму по строке к общему итогу.
        lngReportTotal = lngReportTotal + lngRowTotal
    End If
End Sub
```

Процедура обработки события **Печать** (On Print) примечания отчета должна заполнить поля примечания итоговыми значениями по столбцам из массива `lngRgColumnTotal` (листинг 13.7).

Листинг 13.7. Процедура обработки события *Печать* примечания

```
Private Sub ReportFooter4_Print(Cancel As Integer, PrintCount As Integer)
    Dim intX As Integer

    ' Помещает суммы по столбцам в поля примечания.
    For intX = 1 To intColumnCount - 1
        Me("Tot" + Format(intX)) = lngRgColumnTotal(intX)
    Next intX
    ' Помещает общий итог в поле примечания.
    Me("Tot" + Format(intColumnCount)) = lngReportTotal
    ' Скрывает неиспользуемые поля в примечании отчета.
    For intX = intColumnCount + 1 To conTotalColumns - 1
        Me("Tot" + Format(intX)).Visible = False
    Next intX
End Sub
```

Для корректной работы нужно еще добавить две небольшие процедуры в свойства отчета:

- при закрытии отчета нужно закрыть базовый набор записей — обработка события **Закрытие** (On Close) — листинг 13.8;
- при отсутствии данных в базовом наборе записей нужно закрыть этот набор и прервать формирование отчета — событие **Нет данных** (On No Data) — листинг 13.9.

Листинг 13.8. Процедура обработки события *Закрытие* отчета

```
Private Sub Report_Close()
    On Error Resume Next
    rstReport.Close
End Sub
```

Листинг 13.9. Процедура обработки события *Нет данных* отчета

```
Private Sub Report_NoData(Cancel As Integer)
    MsgBox "Не найдены записи, удовлетворяющие указанным_
                    условиям.", vbExclamation, "Записи не найдены"
    rstReport.Close
    Cancel = True
End Sub
```

Использование функций для обработки событий

Обычно для обработки событий в формах и отчетах используют процедуры типа `Sub` (подпрограммы) или макросы. Однако иногда можно и даже нужно использовать функции. Дело в том, что если в рамках одной формы делается множество однотипных задач, то лучше создать одну процедуру — функцию для выполнения этих задач, описать ее на уровне модуля формы, т. е. в разделе **General** (Общий), а потом вызывать из любого места в форме. Если это необходимо, для такой функции определяется один или несколько параметров, которые передаются при вызове данной функции. И хотя значение, возвращаемое функцией, не используется (а обычно и не определяется), применение ее оправдано не только потому, что требуется писать меньше строк кода, а главным образом потому, что минимизация кода в модуле формы ускоряет ее открытие.

Если же идентичные задачи решаются в разных формах, например, одна и та же реакция предусматривается для одинаковых кнопок в разных формах, то такую функцию нужно написать в стандартном модуле базы данных.

Совет

Если форма сложная, например, содержит большое количество элементов управления, в том числе поля со списками, то в целях повышения ее быстродействия рекомендуется удалить модуль формы, а все процедуры обработки событий заменить функциями, которые следует при этом вынести в стандартный модуль.

Допустим, вы отображаете в форме список каких-либо объектов, например счетов клиента. Обычно это делается в подчиненной форме табличного или ленточного вида, в которой отображаются наиболее важные характеристики счета: номер счета, дата выписки, наименование клиента, сумма. И вы хотите дать возможность пользователю при желании посмотреть тот счет, который он выбрал, установив курсор на соответствующую строчку. Это можно сделать, создав процедуру открытия формы счета с нужной записью при двойном щелчке мыши на строчке счета. Естественно, что форма должна открываться при щелчке в любом поле подчиненной формы. Вместо того чтобы в каждом поле подчиненной формы на событие **Двойное нажатие** (DblClick) создавать процедуру обработки событий, можно создать одну функцию и присоединить ее ко всем полям. Эта функция представлена в листинге 13.10.

Листинг 13.10. Процедура открытия формы счета

```
Private Function Order_DblClick()
Dim strCriteria As String
On Error Goto Err_Order_DblClick

strCriteria = "КодЗаказа = Forms![Заказы клиента]!_
[Заказы клиента подчиненная].form!КодЗаказа"
DoCmd.OpenForm "Заказы", acNormal,, strCriteria

Exit_Order_DblClick:
        Exit Function

Err_Order_DblClick:
        MsgBox Err.Description
        Resume Exit_Order_DblClick

End Function
```

А присоединить эту функцию к событию элемента управления формы можно так, как показано на рис. 13.40. Мы сделали это для формы `Заказы клиента` (Customer Orders) приложения `Борей` (Northwind).



Рис. 13.40. Использование функции при обработке событий в форме

Еще один пример использования функции для обработки событий в форме вы можете увидеть в базе данных `Адресная книга` (Address Book) *(см. пример к гл. 13 на компакт-диске)*. В этом приложении с помощью мастера кнопочных форм создана главная кнопочная форма, которая открывается при открытии приложения. На форме имеются несколько кнопок, каждая из которых выполняет определенную функцию, но обрабатываются они все одинаково. Для этого Мастер создает функцию `=HandleButtonClick`, которая имеет один аргумент — идентификатор нажатой кнопки. Вызов функции выполняется, как показано на рис. 13.41. Аргумент определяет нажатую клавишу. Он обрабатывается программой в блоке `Select Case`, после чего выполняется соответствующее действие.

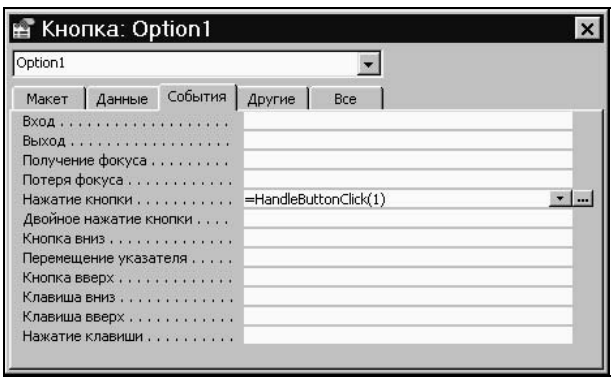


Рис. 13.41. Вызов функции для обработки события **Нажатие кнопки**

Импорт файла XML

В *гл. 3* мы показывали, как можно выполнить импорт и экспорт файлов формата XML средствами интерфейса Access. Теперь мы хотим показать, как можно управлять этими действиями программно, т. к. объектная модель Access включает объекты и методы, которые это позволяют.

Для импорта данных из файла XML можно использовать метод `ImportXML` объекта `Application`. Он позволяет импортировать данные XML как в базу данных Access, так и в базу данных Microsoft SQL Server.

Метод имеет два параметра:

- `DataSource` — определяет полный путь к импортируемому файлу XML и имя этого файла. Это обязательный параметр. Имя импортируемой таблицы содержится в корне файла XML;
- `ImportOptions` — набор флагов, которые определяют, как будет выполняться импорт. Задается константами Access: `acAppendData` — данные из файла XML будут добавляться в существующую таблицу, `acStructureAndData` — импортируется таблица вместе с данными, `acStructureOnly` — импортируется структура таблицы. Параметр не обязательный. По умолчанию импортируются и структура и данные.

В листинге 13.11 приводится пример импорта данных с добавлением в существующую таблицу.

Листинг 13.11. Процедура импорта файла XML

```
Private Sub ImportXMLFile()
   ' Импорт платежек из базы данных нижнего уровня.
   Const APPEND_DATA = 2

   Application.ImportXML DataSource:="C:\Payments.xml", _
       OtherFlags:=APPEND_DATA

End Sub
```

Экспорт файла XML

Для экспорта данных в XML-файл используется метод `ExportXML` объекта `Application`. Мы уже показывали в *гл. 3*, что экспорт в XML-файл существенно улучшен в версии Access 2003, поэтому метод `ExportXML` имеет много параметров. Все они перечислены в табл. 13.15.

Таблица 13.15. *Параметры метода* `ExportXML`

Параметр	Описание
`ObjectType`	Тип экспортируемого объекта Access. Экспортировать можно объекты базы данных Access — таблицы, запросы, формы, отчеты, — а также серверные хранимые процедуры, представления и функции
`DataSource`	Имя экспортируемого объекта. По умолчанию экспортируется активный объект указанного в первом параметре типа
`DataTarget`	Имя результирующего файла XML, содержащего данные. Если этот параметр пропущен, то данные не экспортируются, например, может создаваться только описание структуры данных
`SchemaTarget`	Имя файла XSD, содержащего описание структуры экспортируемых данных. Если этот параметр пропущен, то файл не создается, описание структуры будет в том же файле, что и данные
`PresentationTarget`	Имя файла XSL, описывающего представление данных. Этот параметр, так же как и следующий, обычно используется при импорте таких объектов, как форма или отчет
`ImageTarget`	Папка, в которую помещаются вложенные в отчет рисунки
`Encoding`	Тип кодировки файла XML. По умолчанию используется UTF8

Таблица 13.15 *(окончание)*

Параметр	Описание
OtherFlags	Битовая маска, представляющая набор флагов, определяющих способы экспорта: acEmbedSchema — описание структуры находится в том же файле, что и данные; acExcludePrimaryKeyAndIndexes — не экспортируются первичные ключи и индексы; acRunFromServer — экспортирует форму или отчет в ASP-страничку, которая выполняется на сервере Internet Information Server (по умолчанию создается HTML странички); acLiveReportSource — в файлы XML и HTML в процессе экспорта будут включены не сами данные, а запрос, который позволит получить данные из источника при открытии странички с отчетом (так называемые "живые" данные). Этот флаг используется только в проектах Access для отчетов, источником данных для которых являются таблицы Microsoft SQL Server 2000; acPersistReportML — экспорт структуры формы или отчета в формат ReportML. ReportML — это язык, предназначенный специально для описания объектов базы данных Access. Он состоит из набора тегов, описывающих свойства, события и атрибуты форм, отчетов и страниц доступа к данным
WhereCondition	Условие отбора записей из источника данных DataSource
AdditionalData	Ссылка на объект AdditionalData, который описывает связанные таблицы, экспортируемые вместе с основной

В листинге 13.12 представлена процедура экспорта таблиц Поставщики, Товары и Типы в файл XML.

Листинг 13.12. Процедура экспорта таблиц в файл XML

```
Private Sub ExportXMLFile()
   ' Экспортирует несколько связанных таблиц в файл XML.

   Dim objAD As AdditionalData

   ' Создает объект AdditionalData.
   Set objAD = Application.CreateAdditionalData

   ' Добавляет в него связанные таблицы.
   With objAD
      .Add "Товары"
      .Add "Типы"
   End With
```

```
   Application.ExportXml acExportTable, "Товары", _
       "C:\Товары.xml", "C:\Товары_Schema.xsd", _
       "C:\Товары_Style.xsl", AdditionalData:= objAD
End Sub
```

Преобразование файла XML

Когда приложения обмениваются данными с помощью файлов определенных форматов, в частности файлов XML, возникает необходимость преобразовать импортируемый файл к требуемой структуре перед импортом или, наоборот, при экспорте данных нужно создать файл с данными, структура которых отличается от данных в источнике. Для этого применяются файлы преобразований, создаваемые с использованием языка XSLT. В Access 2003 появился новый метод `TransformXML`, который позволяет применять информацию из файлов XSLT для того, чтобы преобразовать XML-файлы в процессе импорта или экспорта. Этот метод имеет пять параметров, которые описаны в табл. 13.16.

Табица 13.16. *Параметры метода* `TransformXML`

Параметр	Описание
`DataSource`	Имя исходного файла XML вместе с путем к нему
`TransformSource`	Имя файла XSL, содержащего схему преобразования форматов
`OutputTarget`	Имя результирующего файла XML. Это может быть тот же файл, что и исходный, в этом случае старый файл будет заменен новым
`WellFormedXMLOutput`	Логическое значение. Если равно `True`, то при преобразовании будет использоваться метод `TransformNodeToObject` объектной модели DOM. Если этот параметр равен `False` (значение по умолчанию), то будет использоваться метод `TransformNode`. *(Об объектной модели DOM см. гл. 15.)*
`ScriptOption`	Определяется одной из трех констант: `acPromptObjects` — в процессе преобразования при создании любого объекта выдается запрос пользователю на подтверждение; `acEnableObjects` — любой скрипт выполняется автоматически, никаких запросов не будет; `acDisableObjects` — возникает ошибка всякий раз, когда в процессе преобразования будет предприниматься попытка создать объект

В листинге 13.13 приведен пример процедуры, использующей метод `TransformXML` для файла Customer.xml.

Листинг 13.13. Процедура экспорта таблиц в файл XML

```
Sub TransformXMLFile()
    ' Берет файл Customer.xml и преобразует его с помощью файла
    ' преобразований SortNames.xsl. Старый файл будет заменен.

    Application.TransformXML _
        DataSource:="C:\Customers.xml", _
        TransformSource:="C:\SortNames.xsl", _
        OutputTarget:="C:\Customers.xml", _
        WellFormedXMLOutput:=True

End Sub
```

ГЛАВА 14



Настройка пользовательского интерфейса

Для того чтобы создать приложение Access, требуется решить две важнейшие задачи: спроектировать эффективную структуру таблиц и их взаимосвязей (модель данных) и обеспечить удобный пользовательский интерфейс. О том, как эффективно спроектировать базу данных, мы рассказали в *гл. 2*, первые главы книги были посвящены разработке основных объектов Access: форм, отчетов, страниц доступа к данным. Чтобы создать единое целое из набора отдельных объектов базы данных, используются макросы и процедуры VBA, разработка которых рассматривалась в *гл. 12 и 13*. Для того чтобы приложение было законченным, применяются дополнительные элементы, такие как специальное меню команд или заменяющая его Кнопочная форма, панели инструментов, контекстное меню, а если вы любите пользоваться клавиатурой и готовы запомнить десятка полтора комбинаций клавиш, можно создать клавиши быстрого вызова.

Данная глава будет посвящена тому, как придать законченный вид приложению. В ней мы рассмотрим следующие вопросы:

- Как настроить параметры среды Access?
- Какие существуют способы запуска приложения и как обеспечить выполнение начальных действий при запуске?
- Зачем и как создать Кнопочную форму приложения?
- Как создать и настроить меню и панели инструментов, как их "прятать" и показывать в нужный момент?
- Как назначить меню и панели инструментов формам, отчетам и элементам управления.

Настройка параметров приложения Access

При установке Microsoft Access параметры среды получают значения, определенные по умолчанию, однако в процессе разработки приложения вы можете переустановить их таким образом, чтобы они соответствовали особенностям конкретного приложения и были удобны для его пользователей.

Большинство параметров Access настраивается с помощью диалогового окна **Параметры** (Options), которое отображается при выборе команды **Сервис | Параметры** (Tools | Options). Все эти параметры сгруппированы по своему назначению и размещены на нескольких вкладках. На некоторые из этих параметров и вкладок мы ссылались в предыдущих главах книги. В этой главе приведены описания всех параметров.

На рис. 14.1 представлена вкладка **Вид** (View) окна **Параметры** (Options) со значениями параметров, установленными по умолчанию.

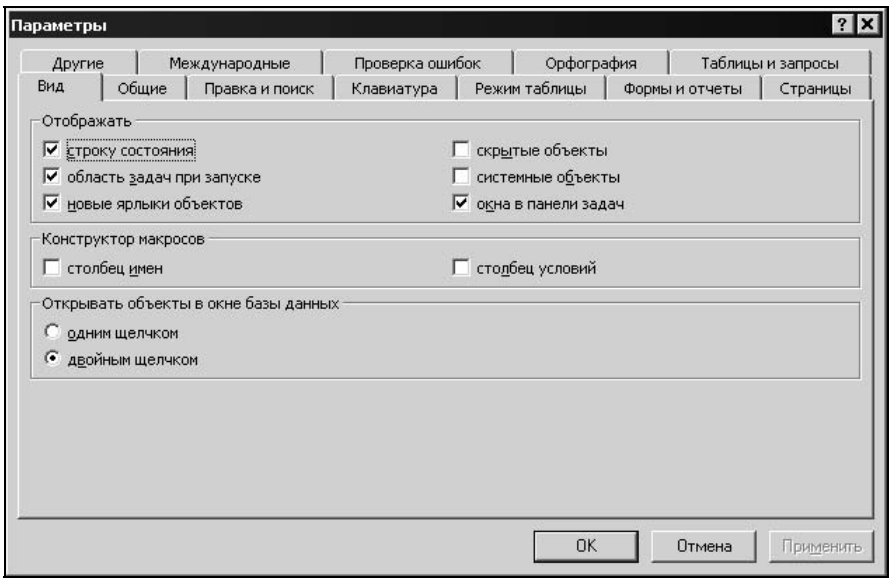


Рис. 14.1. Вкладка **Вид** диалогового окна **Параметры**

Эта вкладка содержит следующие группы параметров:

- **Отображать** (Show) — задает внешний вид среды Access. Флажки в этой группе позволяют скрыть или показать строку состояния (Status bar), область задач при запуске (Startup Task Pane), ярлыки для создания новых объектов в окне **База данных** (New object shortcart), системные объекты и скрытые объекты. При установленном флажке **окна в панели задач** (Use taskbar for each document) на панели задач в нижней части экрана отображаются значки открытых объектов базы данных: форм, отчетов, запросов. Если вы не хотите загромождать панель задач, сбросьте этот флажок. Кроме таблиц приложения, созданных разработчиком, существуют системные таблицы Access. По умолчанию они не отображаются в окне базы данных, но если установить флажок **системные объекты** (System Objects), то эти таблицы появятся. Все они начинаются с префикса `MSys`. Разработчик имеет права только на просмотр этих таблиц. Для любого объекта базы данных можно установить свойство **скрытый** (Hidden) *(см. разд. "Окно базы данных" гл. 1)*. По умолчанию скрытые объекты не отображаются в окне базы данных. Если установить флажок **скрытые объекты** (Hidden Objects), то все такие объекты будут видны.

- **Конструктор макросов** (Show in macro design) — позволяет показать дополнительные столбцы в окне Конструктора макросов: имена макросов и условия выполнения макрокоманд *(см. разд. "Создание групп макросов" гл. 12)*.
- **Открывать объекты в окне базы данных** (Click options in database window) — содержит очень полезный, на наш взгляд, переключатель, который позволяет открывать объекты из окна базы данных не только по двойному щелчку мыши (режим по умолчанию), но и по одинарному — это гораздо удобнее.

Следующая вкладка **Общие** (General) (рис. 14.2) содержит ряд общих параметров.

- Группа **Границы печатного поля** (Print Margins) позволяет установить значения полей, применяемые по умолчанию при печати любого объекта базы данных. Их можно изменить для конкретного объекта с помощью команды **Файл | Параметры страницы** (File | Page Setup) *(см. разд. "Печать таблицы" гл. 2)*.
- Два флажка в группе **Формат года из четырех цифр** (Use four-year digit formatting) позволяют задать формат отображения дат в текущей базе данных и во всех базах данных, работающий под управлением Access. По умолчанию во всех датах показываются четыре цифры года.
- Группа флажков **Автозамена имен** (Name AutoCorrect) обеспечивает возможность при изменении пользователем имени таблицы или поля корректно изменить все ссылки на эту таблицу или поле во всех связанных объектах — очень полезная функция, не забудьте о ней, т. к. изменить имя таблицы или поля иногда нужно, а самим исправить все ссылки на него — задача не из легких.
- Несколько независимых флажков определяют следующие параметры:
  - указать, какое количество последних имен файлов MDB будет сохраняться и отображаться в списке в меню **Файл** (File);
  - включить или выключить звуковое сопровождение событий в приложении;
  - сжатие базы данных при закрытии. Известно, что при интенсивной работе MDB-файлы имеют очень неприятную тенденцию быстро увеличиваться в размере. Вы можете установить флажок **Сжимать при закрытии** (Compact on close), и тогда файл будет сжиматься каждый раз при выходе из приложения;
  - флажок **Удалять личные сведения из свойств файла при сохранении** (Remove Personal Information), будучи установленным, позволяет прятать персональную информацию на вкладке **Документ** (Summery) диалогового окна свойств базы данных (меню **Файл | Свойства** (File | Properties)).
- Поле **Рабочий каталог** (Default Database Folder) определяет папку, которая отображается по умолчанию в диалоговых окнах при открытии и сохранении файлов баз данных. Обычно это папка Мои документы (My documents) в Windows 2000, и Windows XP или Личная (Personal) в Windows NT.
- Поле со списком **Порядок сортировки базы данных** (New database sort order) имеет по умолчанию значение **Универсальный** (General). Это означает, что порядок сортировки в создаваемых базах данных будет определяться региональными стандартами, установленными на компьютере с помощью Панели управления (Control Panel) Windows. Можно поменять порядок сортировки, изменив значение в этом поле. Но обращаем ваше внимание, что значение этого поля определяет порядок сортировки в новой базе, т. е. оно должно быть установлено перед

тем, как создается новая база. Изменение этого поля не меняет порядок сортировки в уже созданной базе данных. Для того чтобы изменить порядок сортировки в уже существующей базе данных, нужно выполнить ее сжатие.

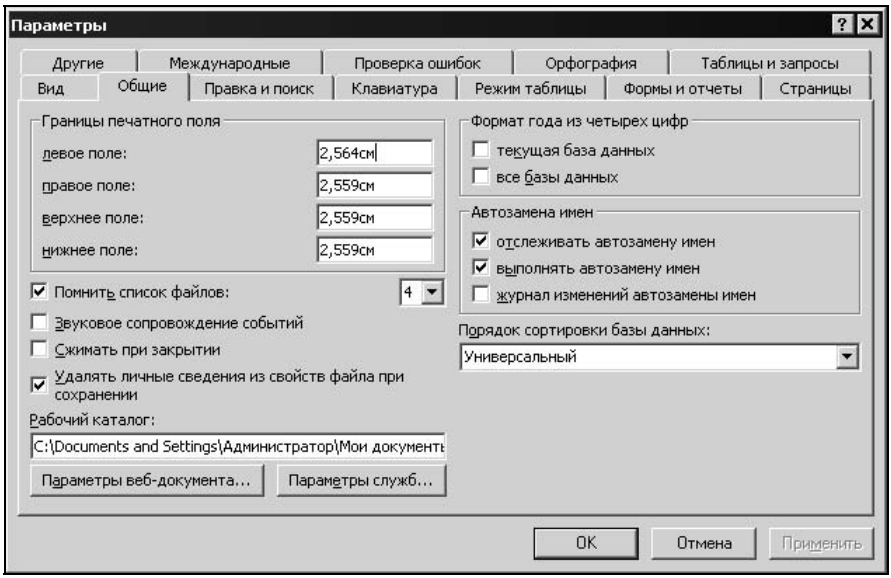


Рис. 14.2. Вкладка **Общие** диалогового окна **Параметры**

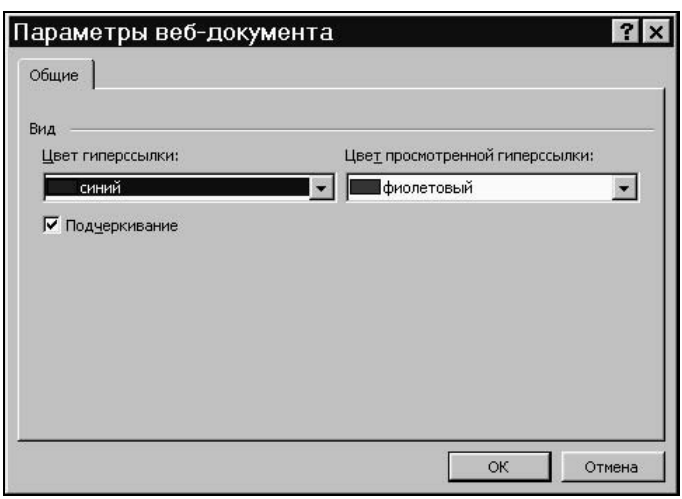


Рис. 14.3. Окно **Параметры веб-документа**

- На этой же вкладке есть еще две кнопки: **Параметры веб-документа** (Web Options), которая позволяет открыть одноименное диалоговое окно и установить цвет и вид гиперссылок (рис. 14.3), и **Параметры служб** (Service Options) (рис. 14.4).

При нажатии кнопки **Параметры служб** открывается диалоговое окно, содержащее две категории параметров: **Параметры отзывов пользователей** (Customer Feedback Options) и **Содержимое в сети** (Online Content). Параметры отзывов пользователей позволяют вам принять участие в программе улучшения качества ПО, проводимой компанией Microsoft. Параметры содержимого в сети позволяют настроить справочную систему Access так, чтобы всегда получать сведения с узла Microsoft Office Online.

Замечание

Это же окно открывается по ссылке в области задач **Справка Access**.

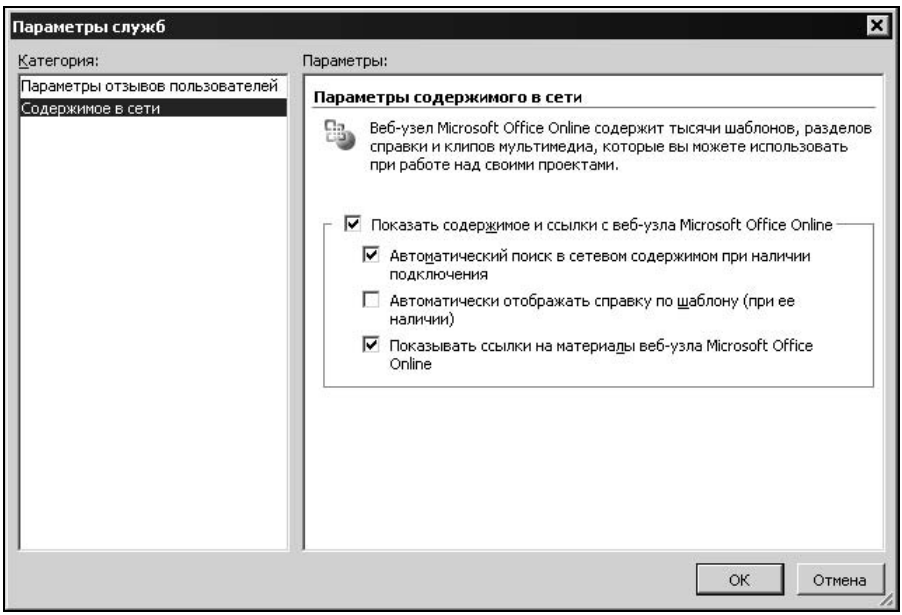


Рис. 14.4. Окно **Параметры служб**

Вкладка **Правка и поиск** (Edit/Find) содержит параметры, которые используются при поиске, фильтрации и редактировании записей (рис. 14.5).

- Группа флажков **Поиск и замена по умолчанию** (Default Find/Replace Behavior) определяет стратегию поиска данных в таблицах Access, которая принята по умолчанию:
  - флажок **поле - целиком** (Fast search) означает, что поиск будет вестись в текущем поле и на полное совпадение значения поля с образцом;
  - флажок **везде - любая часть** (General search) обеспечивает поиск по всем полям и поиск вхождения искомой строки в значение поля;
  - флажок **поле - с начала** (Start of field search) означает поиск в текущем поле на совпадение образца с первыми символами значения поля.

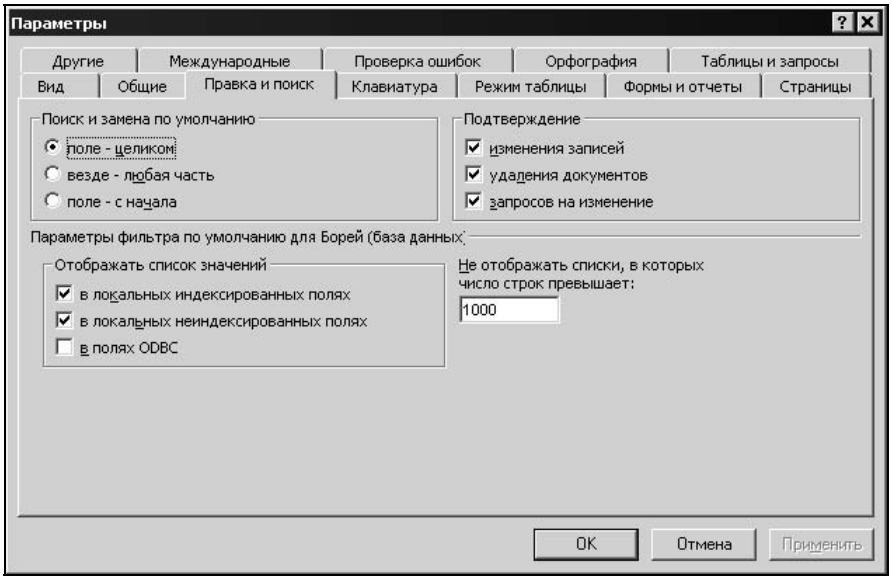


Рис. 14.5. Вкладка **Правка и поиск** диалогового окна **Параметры**

- Группа флажков **Подтверждение** (Confirm) позволяет отключить при необходимости вывод на экран предупреждающих сообщений при изменении и удалении объектов и данных. Это эквивалент макрокоманды `SetWarnings` (УстановитьСообщения) со значением аргумента `False` (Ложь). Однако будьте внимательны, т. к. забыв их потом снова включить, можно случайно испортить готовые формы, отчеты или изменить данные нежелательным образом.
- Группа параметров **Параметры фильтра по умолчанию** (Filter By Form Defaults) определяет поведение системы при установке фильтра. Флажки в группе **Отображать список значений** (Show list of values in) позволяют определить, когда в окне задания фильтра будут появляться списки для выбора критериев. Они могут появляться в индексированных полях локальной таблицы, в неиндексированных полях локальной таблицы и в полях внешней таблицы. Специальное поле ввода позволяет указать максимальное количество элементов, которые могут содержать отображаемые списки значений. Если количество элементов в списке превышает указанный предел, такой список отображаться не будет *(см. разд. "Сортировка, поиск и фильтрация данных" гл. 2)*.

На вкладке **Клавиатура** (Keyboard) расположены параметры, которые определяют возможные перемещения курсора при вводе и редактировании данных в формах (рис. 14.6). По умолчанию определены следующие перемещения:

- переход при нажатии клавиши <Enter> — следующее поле;
- переход по клавише со стрелкой вправо — следующее поле;
- поведение при входе в поле — выделить все поле;
- останов на первом/последнем поле — нет (флажок сброшен).

Вы можете настроить эти параметры удобным для вас образом.

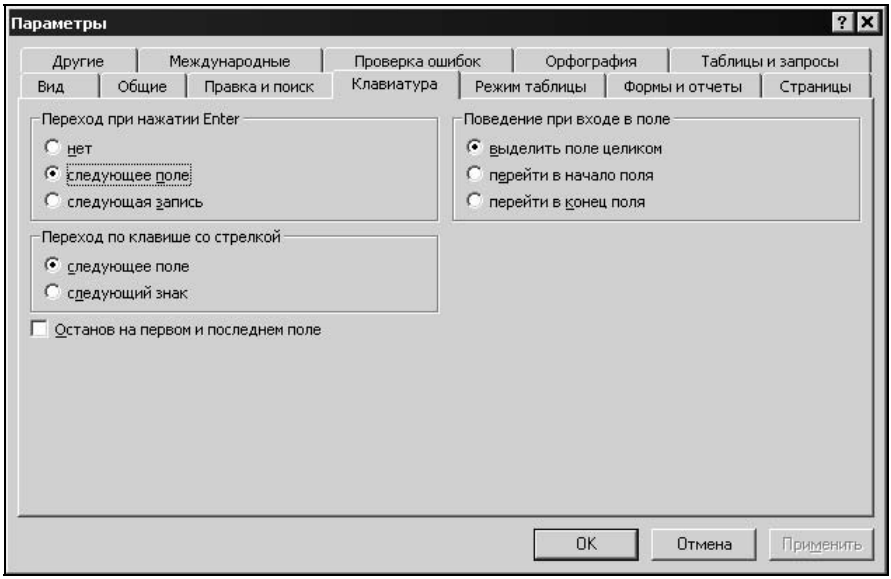


Рис. 14.6. Вкладка **Клавиатура** диалогового окна **Параметры**

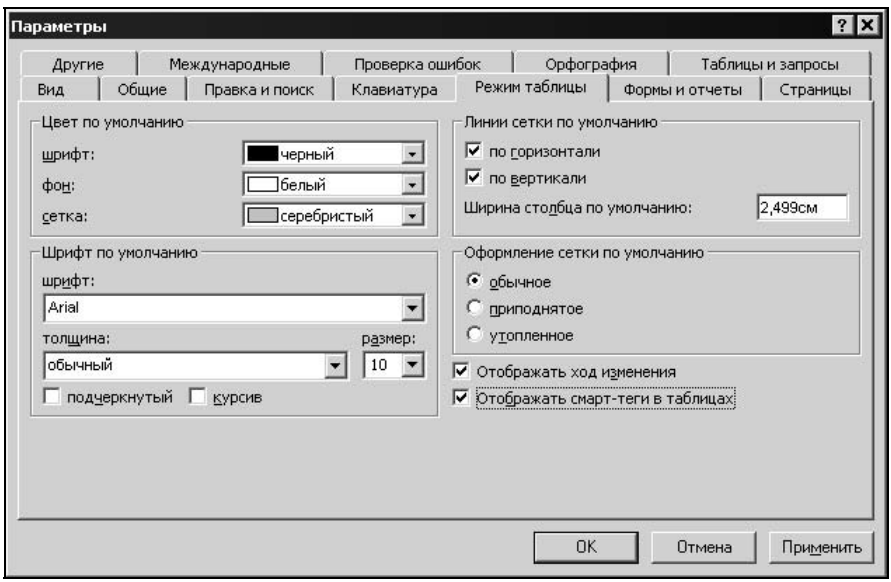


Рис. 14.7. Вкладка **Режим таблицы** диалогового окна **Параметры**

На вкладке **Режим таблицы** (Datasheet View) (рис. 14.7) находятся параметры, которые определяют вид таблиц: тип и размер шрифта, цвета символов и фона, наличие, отсутствие и цвет сетки, специальные эффекты оформления сетки, ширину столб-

цов. Это те параметры, которые заданы по умолчанию для всех таблиц. При желании их можно изменить для конкретной таблицы с помощью команд меню **Формат** (Format). Кроме этого на вкладке есть еще два флажка:

- **Отображать ход изменения** (Show animations) — устанавливается для включения анимации при визуальном отображении таких изменений, как вставка новых столбцов в таблицу;
- **Отображать смарт-теги в таблицах** (Show Smart Tags in Datasheets) — включает или отключает отображение смарт-тегов в режиме таблицы.

Параметры на вкладке **Формы и отчеты** (Forms/Reports) позволяют задать некоторые общие для форм и отчетов значения (рис. 14.8).

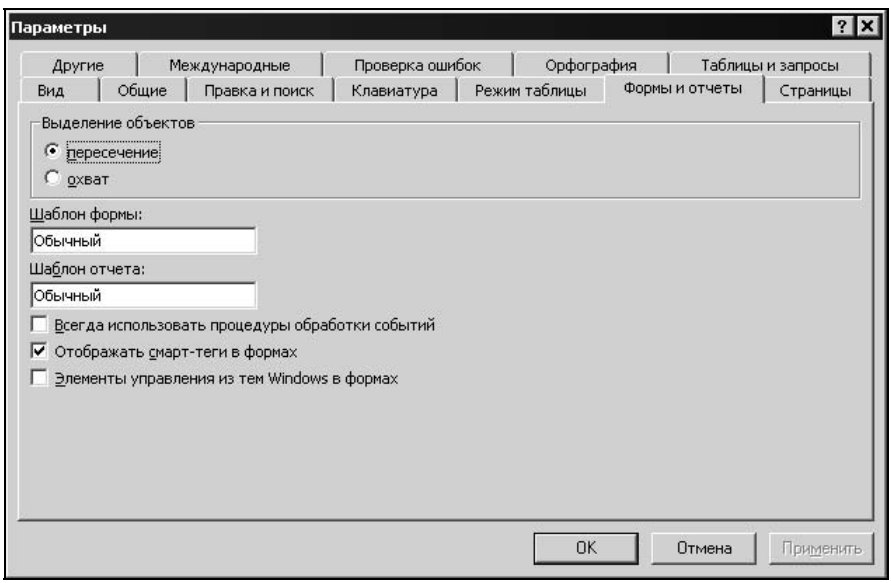


Рис. 14.8. Вкладка **Формы и отчеты** диалогового окна **Параметры**

- Можно задать шаблоны, которые применяются по умолчанию для форм и отчетов. В качестве шаблона используется имя формы или отчета, которые оформлены в нужном стиле. Тогда все вновь создаваемые формы и отчеты будут сразу создаваться в вашем стиле, что позволит существенно сократить время разработки интерфейса. Правда, эти параметры относятся только к формам и отчетам, создаваемым без помощи Мастеров.
- Группа **Выделение объектов** (Selection Behavior) определяет, как выполняется операция выделения объектов в режиме Конструктора форм и отчетов. По умолчанию выделяются все объекты, которые даже частично попадают в область выделения. Если выбрать переключатель **охват** (Fully enclosed), то будут выделены только те объекты, которые попадут в эту область целиком. Когда элементы управления в форме или отчете расположены очень плотно, лучше использовать первый переключатель — **пересечение** (Partially enclosed).

- Флажок **Всегда использовать процедуры обработки событий** (Always use event procedures) позволяет избежать появления диалогового окна **Построитель** (Choose Builder). Обычно это окно появляется, когда пользователь назначает обработчик события в окне свойств формы, отчета или элемента управления. По умолчанию этот флажок сброшен. Если вы для обработки событий не используете макросов, можно его установить.

По сравнению с предыдущей версией Access в Access 2003 появилось два новых флажка на вкладке **Формы и отчеты**:

- **Отображать смарт-теги в формах** (Show Smart Tags in Forms) — действует аналогично такому же флажку для режима таблицы.
- **Элементы управления из тем Windows в формах**. Если на компьютере установлена Windows XP с SP1 или выше, то значение этого флажка влияет на вид элементов управления в формах. Они будут наследовать тему Windows, установленную на вкладке **Темы** диалогового окна **Отображение** панели управления Windows.

Следующая вкладка **Страницы** (Pages) содержит параметры, касающиеся страниц доступа к данным (рис. 14.9).

- Первая группа параметров **Стандартные свойства конструктора** (Default Designer Properties) касается Конструктора (Designer). Вы можете задать расстояние между разделом и полями страницы, цвет текущей строки, стили для заголовков и примечаний разделов.
- Вторая группа — **Стандартные свойства БД или проекта** (Default Database/Project Properties). Вы можете установить флажок и определить папку, в которой по умолчанию будут сохраняться файлы страниц, и файл, содержащий параметры подключения страницы к базе данных *(см. разд. "Страницы доступа к данным" гл. 11)*.

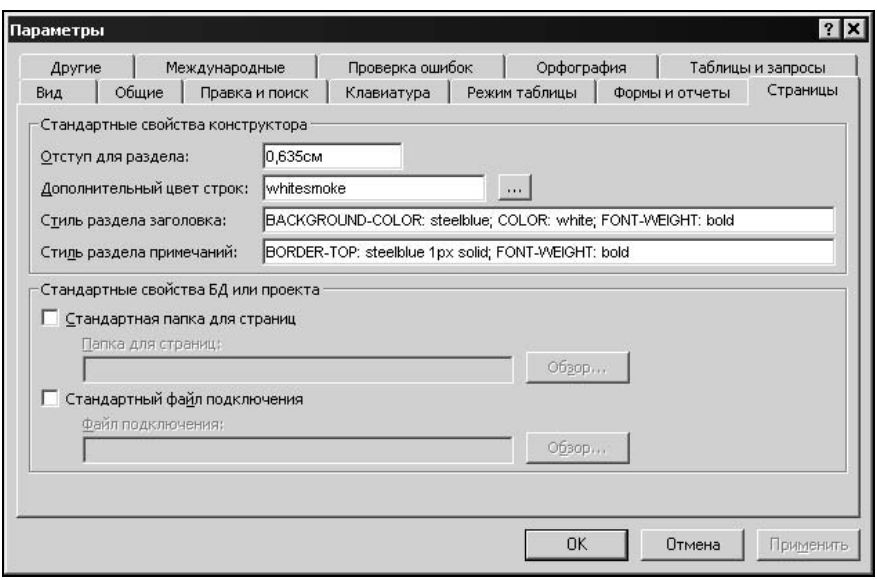


Рис. 14.9. Вкладка **Страницы** диалогового окна **Параметры**

Вкладка **Другие** (Advanced) объединяет параметры, определяющие функционирование приложения в многопользовательском режиме или в режиме "клиент-сервер" (рис. 14.10). Назначение этих параметров будет описано далее *(см. гл. 16 и 17)*.

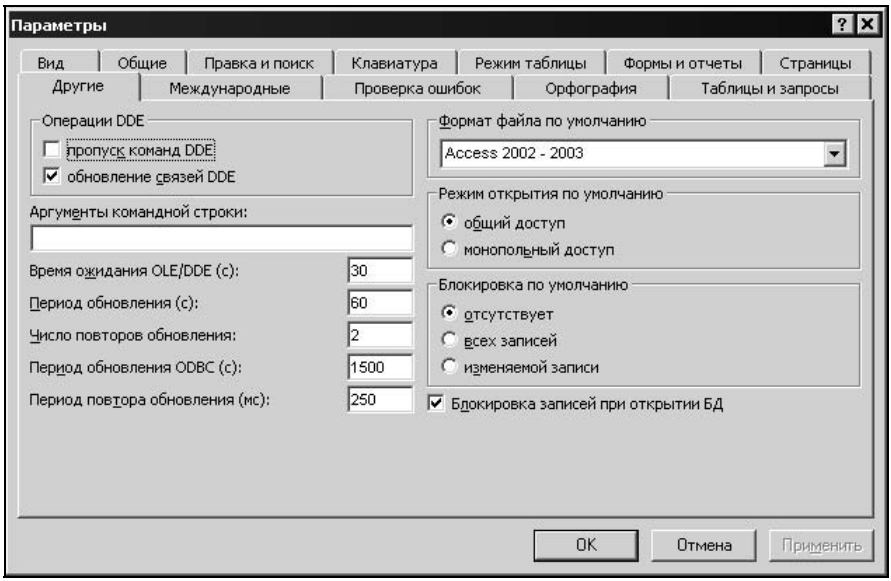


Рис. 14.10. Вкладка **Другие** диалогового окна **Параметры**

Кроме того, вы можете определить используемый по умолчанию формат файла базы данных: Access 2000 или Access 2002—2003. По умолчанию создаваемые файлы базы данных имеют формат Access 2000. Это позволяет использовать их и в Access 2000, и в Access 2002, и в Access 2003. Файлы, имеющие формат Access 2002—2003, доступны только в Access 2002 и в Access 2003.

- Поле ввода **Аргументы командной строки** (Command-line Arguments) позволяет задать параметры командной строки, которые будут переданы приложению Microsoft Access при запуске.
- Группа переключателей **Режим открытия по умолчанию** (Default open mode) позволяет задать режим доступа к открываемой базе данных — монопольный (Exclusive) или многопользовательский (Shared).

Специальная вкладка **Международные** (International) (рис. 14.11) позволяет задать параметры, учитывающиеся использование языков, в которых текст пишется не слева направо, как мы привыкли, а справа налево, и календарь не григорианский, а юлианский. Можно, например, установить эти параметры таким образом, что если в одном предложении содержатся и арабский и европейский тексты, то при перемещении по арабскому тексту курсор будет перемещаться справа налево, а при перемещении по европейскому (русскому, английскому и т. д.) тексту — слева направо. Изменение же календаря рекомендуется делать сразу, как только создан новый файл базы данных, и потом этот параметр не менять.

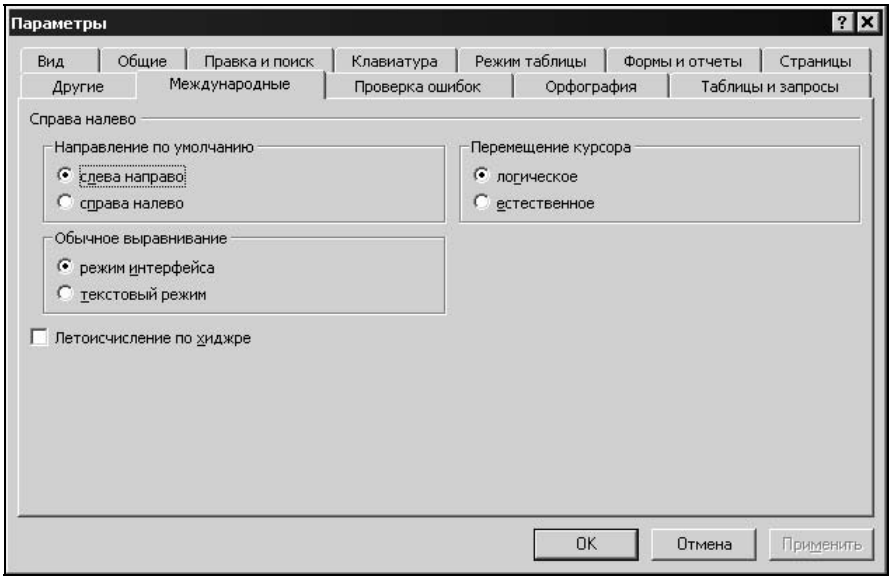


Рис. 14.11. Вкладка **Международные** диалогового окна **Параметры**

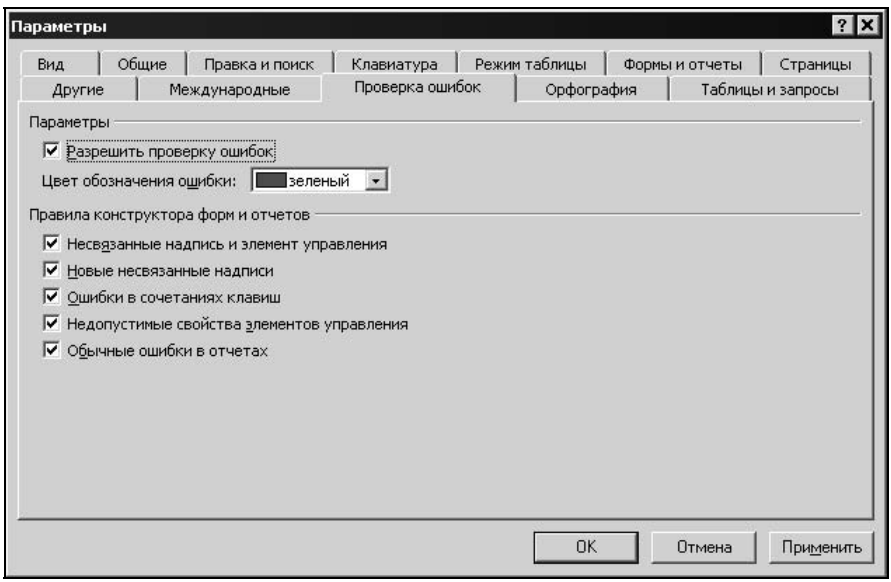


Рис. 14.12. Вкладка **Проверка ошибок** диалогового окна **Параметры**

Новая вкладка **Проверка ошибок** (Error Checking) (рис. 14.12) содержит флаги, которые включают или выключают режим проверки ошибок в различных режимах работы с Access. Во-первых, там есть флажок, который просто включает или выключает

режим проверки ошибок. Во-вторых, при включенном режиме проверки ошибок можно указать, какие ситуации следует рассматривать как ошибку и сигнализировать об их возникновении. При возникновении ситуации, рядом с которой флажок установлен, появляется смарт-тег, предлагающий набор возможных действий для разрешения данной ситуации. И, кроме того, будет помечен элемент управления или другой объект, в котором возникла ошибка (по умолчанию зеленым цветом).

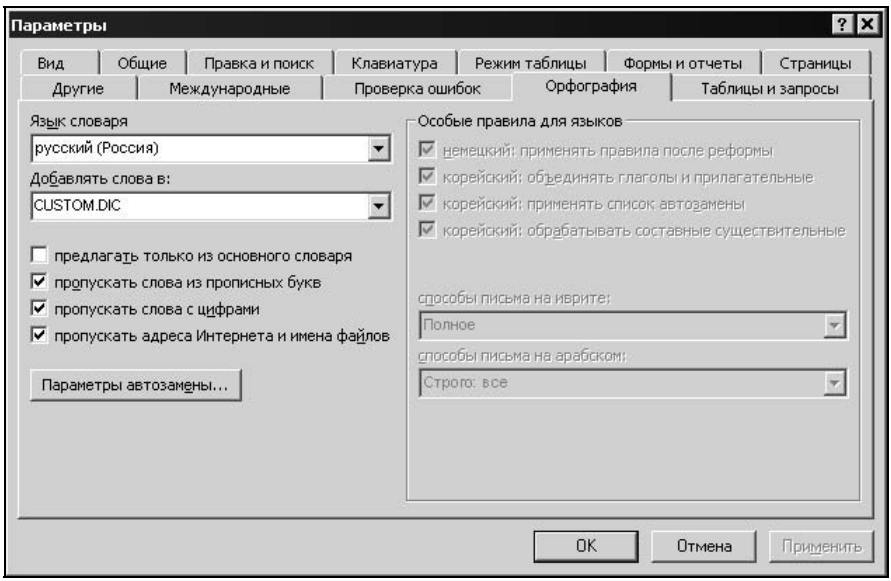


Рис. 14.13. Вкладка **Орфография** диалогового окна **Параметры**

Параметры, связанные с проверкой орфографии и автозамены, тоже вынесены на отдельную вкладку **Орфография** (Spelling) (рис. 14.13). Это те же параметры, которые используются функцией проверки орфографии в Microsoft Word. Вы можете вызвать эту функцию, нажав кнопку <F7> в текстовом поле таблицы, запроса или формы.

- По умолчанию в поле **Язык словаря** (Dictionary Language) установлен словарь, соответствующий региональным установкам на вашем компьютере. Нужный словарь можно выбрать из списка в этом поле.
- В поле **Добавлять слова в** (Add Words to) можно определить файл, который будет содержать ваш собственный словарь.
- Несколько флажков позволяют указать, какие слова нужно игнорировать в процессе проверки правописания.
- Кнопка **Параметры автозамены** (AutoCorrect Options) открывает диалоговое окно, в котором задаются параметры автозамены (рис. 14.14).

Назначение параметров Автозамены ясно из их названий. Автозамена используется в данном случае для выполнения следующих функций.

- Автоматическое исправление опечаток при вводе данных в текстовые поля. Исправление производится путем замены слова в левом столбце списка словом

в правом столбце. По умолчанию этот список достаточно длинный. Данная функция будет эффективно использоваться, если вы настроите список таким образом, чтобы он включал в себя типичные опечатки для конкретного пользователя.

- Автоматическая замена при вводе определенных сочетаний обычных символов специальными символами.
- Автоматическое исправление типичных ошибок ввода, связанных с неправильным нажатием клавиш <Shift> и <Caps Lock>. Возможные исправления определяются флажками в верхней части диалогового окна **Автозамена** (AutoCorrect).

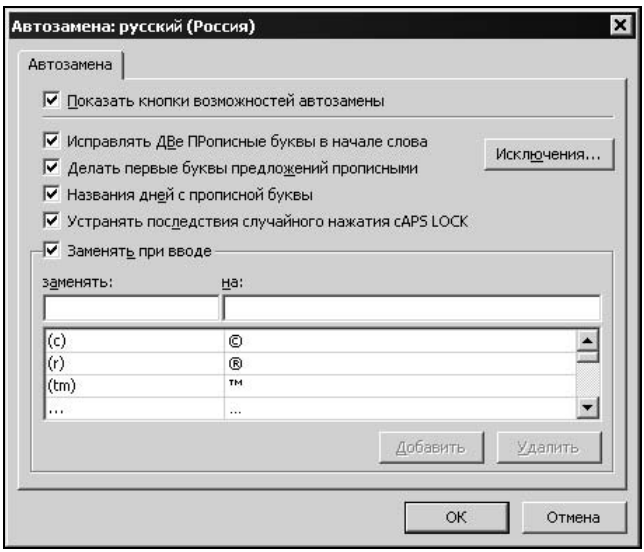


Рис. 14.14. Диалоговое окно **Автозамена**

Чтобы пополнить список Автозамены, необходимо ввести заменяемое сочетание букв в поле **заменять** (Replace), а заменяющее слово или словосочетание — в поле **на** (With) и затем нажать кнопку **Добавить** (Add).

На последней вкладке **Таблицы и запросы** (Tables/Queries) размещены две группы параметров (рис. 14.15).

- Группа параметров **Конструктор таблиц** (Table design) определяет свойства поля, устанавливаемые по умолчанию при добавлении нового поля в таблицу: тип поля, размер для текстового и числового полей. Обратите внимание на поле **Автоиндекс при импорте и создании** (AutoIndex on Import/Create), в котором перечисляются через точку с запятой некоторые слова. Дело в том, что когда одно из этих слов встречается в имени вновь создаваемого поля, Конструктор таблиц автоматически строит индекс по этому полю. Таким образом, если вы будете придерживаться определенных правил при наименовании полей таблиц, например все ключевые поля будете обозначать именами с суффиксом `Id`, то ваша работа по созданию индексов для таблиц заметно упростится. Новый флажок **кнопки параметров обновления свойств** (Show Property Update Options Buttons) включает

или отключает режим автоматического обновления наследуемых свойств полей таблицы во всех связанных объектах при изменении значений этих свойств в таблице.

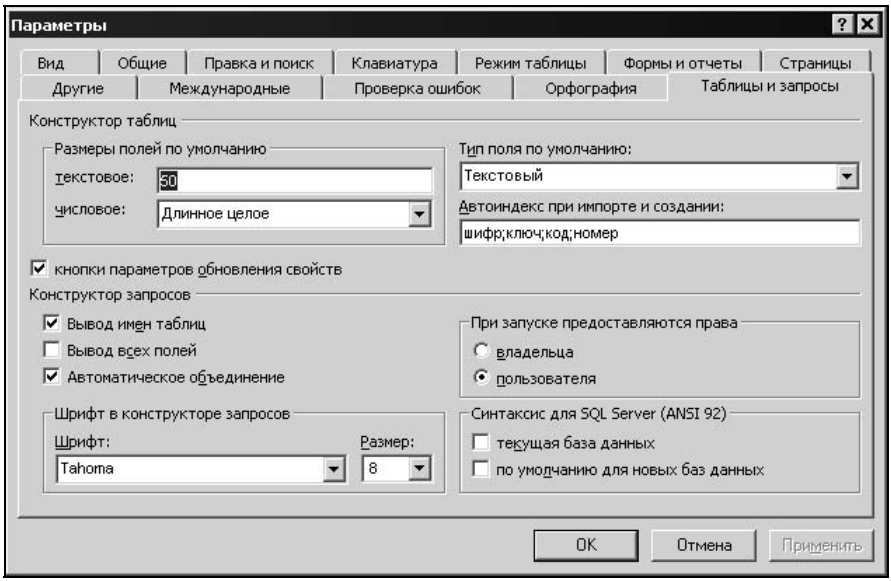


Рис. 14.15. Вкладка **Таблицы и запросы** окна **Параметры**

- Группа параметров **Конструктор запросов** (Query design) влияет на вид Конструктора запросов и на некоторые свойства запроса, задаваемые по умолчанию.
  - Флажок **Вывод имен таблиц** (Show Table Names) позволяет включать или отключать отображение строки **Имя таблицы** (Table) в бланке запроса.
  - Флажок **Вывод всех полей** (Output all fields) устанавливает значение по умолчанию **Да** (True) или **Нет** (False) для одноименного свойства запроса. Если это свойство равно **Да**, то при выполнении запроса будут выводиться все поля базовых таблиц или запросов независимо от того, какие поля указаны в бланке запроса. Обычно это свойство имеет значение **Нет**, и флажок сброшен.
  - Флажок **Автоматическое объединение** (Enable AutoJoin) обычно установлен и это означает, что при добавлении таблиц в запрос, Конструктор автоматически объединяет их, т. е. отображает связи, если в них имеются поля с одинаковыми именами и одно из них является ключевым. Если этот флажок сброшен, то автоматического объединения происходить не будет.
  - Группа параметров **Шрифт в конструкторе запросов** (Query Design Font) позволяет определить гарнитуру и размер шрифта, который будет использоваться в конструкторе запросов и в окне **SQL**.
  - Переключатели **При запуске предоставляются права** (Run permissions) определяют значение одноименного свойства запросов по умолчанию. Могут пре-

доставляться права **владельца** (Owner's) или **пользователя** (User's). Предоставляя права владельца, вы позволяете всем пользователям просматривать результаты запросов или запускать запросы на изменение. Если предоставлены права пользователя, то каждый пользователь будет иметь те права, которые выдаются ему системой безопасности *(см. гл. 20)*. По умолчанию предоставляются права **пользователя**.

- Два флажка в группе **Синтаксис для SQL Server (ANSI 92)** (SQL Server Compatible Syntax) влияют на синтаксис запросов SQL, генерируемых Конструктором *(см. разд. "Создание запросов SQL" гл. 8)*.

Замечание

Изменение всех параметров Конструктора запросов влияет только на новые запросы.

Замечание

В диалоговое окно **Параметры** (Options) не включены параметры, *определяющие* работу в редакторе VBA и Редакторе сценариев. Они задаются в среде этих средств разработки *(см. разд. "Программирование страниц доступа к данным" гл. 11 и "Среда программирования VBA" гл. 13)*.

Замечание

Все параметры, устанавливаемые в окне **Параметры** (Options), могут изменяться динамически в процессе выполнения программы VBA с помощью метода `SetOption` объекта `Application`. Кроме того, можно прочитать значение любого из этих параметров с помощью метода `GetOption` того же объекта.

Настройка параметров запуска

Существует два способа задать действия, выполняемые приложением при запуске, и ряд параметров, влияющих на работу приложения: установка параметров запуска в специальном диалоговом окне и создание макроса `AutoExec`. Создание и назначение макроса `AutoExec` уже рассматривалось *(см. разд. "Запуск макроса при открытии базы данных" гл. 12)*.

В данном разделе мы будем описывать установку параметров запуска. Для этого выберите команду **Сервис | Параметры запуска** (Tools | Startup). На экране появляется диалоговое окно **Параметры запуска** (Startup) (рис. 14.16).

В диалоговом окне **Параметры запуска** (Startup) можно задать:

- значок и заголовок приложения, которые будут отображаться в главном окне приложения и на панели задач вместо стандартного значка и заголовка "Microsoft Access". Значок должен быть задан в файле с расширением ico, а выбрать данный файл можно с помощью кнопки построителя. Этот же значок может быть использован во всех формах и отчетах, для чего необходимо установить флажок **Значок форм и отчетов** (Use as Form and Report Icon). Эти параметры вступают в силу сразу после закрытия диалогового окна **Параметры запуска** (Startup);
- меню, которое будет появляться при запуске вместо стандартного меню Access и определять основные функции приложения;

- форму или страницу, которая будет появляться на экране при открытии базы данных;
- специальное контекстное меню, которое будет заменять встроенные контекстные меню во всех окнах приложения, кроме тех, с которыми связаны другие контекстные меню.

Флажки **Окно базы данных** (Display Database Window) и **Строка состояния** (Display Status Bar) позволяют скрывать при запуске окно базы данных и строку состояния.

Следующая группа флажков позволяет запретить пользователям вносить изменения в разработанное приложение. Пока вы разрабатываете приложение, все эти флажки установлены, но когда вы будете передавать его пользователям, целесообразно сбросить флажки **Полный набор меню Access** (Allow Full Menus), **Встроенные панели инструментов** (Allow Built-in Toolbars) и **Изменение панелей инструментов/меню** (Allow Toolbar/Menu Changes). Вместо стандартных меню и панелей инструментов можно предоставить пользователям свои собственные меню и панели инструментов или использовать встроенные сокращенные меню, не допускающие пользователя к конструкторам. Чтобы пользователь не мог получить доступ к полным меню через специальную комбинацию клавиш <Ctrl>+<F11> или к окну базы данных с помощью клавиши <F11>, нужно еще сбросить флажок **Специальные клавиши Access** (Use Access Special Keys). Если этот флажок сброшен, пользователь не сможет также открыть окно с текстом программ с помощью комбинаций клавиш <Ctrl>+<G> или <Ctrl>+<Break>. Но при этом он может все-таки попасть в текст программы, если произойдет ошибка, которая не перехвачена в коде программы.

Замечание

Все эти параметры вступают в силу только после следующего запуска приложения.

Существует возможность обойти параметры запуска и макрос `AutoExec`, для чего необходимо запускать приложение с нажатой клавишей <Shift>.

Замечание

Все параметры запуска, задаваемые в диалоговом окне **Параметры запуска** (Startup), могут быть установлены также с помощью программы VBA, т. к. все они являются свойствами объекта `Database`.

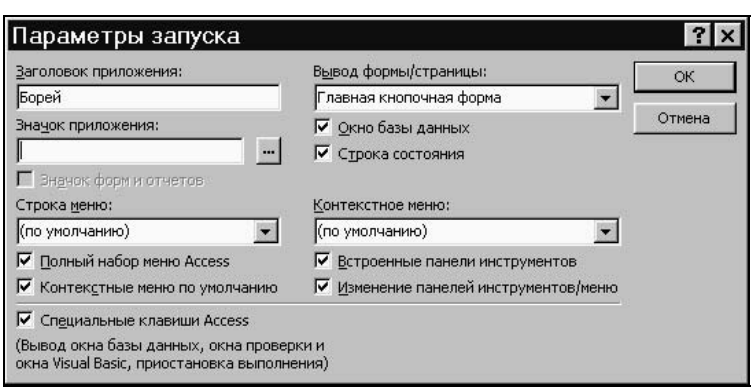


Рис. 14.16. Диалоговое окно **Параметры запуска**

Запуск приложения с параметрами командной строки

При запуске Access можно прямо в командной строке указать ряд параметров, которые будут влиять на работу приложения. Список этих параметров приведен в табл. 14.1.

Таблица 14.1. *Параметры командной строки*

Параметр	Описание
`Имя файла`	Указывается файл базы данных (с расширением mdb) или файл проекта (с расширением adp), который будет открываться
`/excl`	Заданная база данных открывается в режиме монопольного доступа. Параметр нельзя применять к файлу проекта (файлу adp)
`/ro`	Заданная база данных или проект Access открывается в режиме "только для чтения"
`/user имя пользователя`	Задается идентификатор пользователя. Параметр нельзя применять к файлу проекта
`/pwd пароль`	Задается пароль пользователя. Параметр нельзя применять к файлу проекта
`/profile конфигурация`	При запуске Access будут использоваться параметры, заданные в указанной конфигурации вместо стандартных параметров в реестре Windows
`/compact имя файла`	Указывается имя результирующего файла, который получится в результате сжатия и восстановления файла базы данных или проекта, заданного в первом параметре. Если имя файла в данном параметре пропущено, то результирующий файл создается с тем же именем. После выполнения операции Access закрывается. Если в качестве имени файла в первом параметре указан файл проекта Access, то сжимается именно этот файл, а не файл базы данных
`/repair`	Выполняется восстановление указанного в первом параметре файла. Этот параметр оставлен только для совместимости с более ранними версиями Access, т. к., начиная с версии Access 2000, операции сжатия и восстановления объединены (см. предыдущий параметр). Параметр нельзя применять к файлу проекта
`/convert имя файла`	Задает операцию преобразования файла базы данных, созданного в предыдущих версиях Access, в формат, установленный по умолчанию (Access 2000 или Access 2002—2003). Указывается имя результирующего файла. Имя исходного файла задается в первом параметре командной строки. Параметр нельзя применять к файлу проекта

Таблица 14.1 *(окончание)*

Параметр	Описание
`/x имя макроса`	Позволяет выполнить при запуске Access указанный макрос
`/cmd список значений`	Список значений представляет собой аргументы, которые передаются программе. Эти аргументы потом можно получить с помощью функции `Command`. Этот параметр должен быть последним в командной строке. Вместо него можно использовать просто символ точки с запятой (;)
`/nostartup`	При запуске Access не будет отображаться область задач в правой части окна приложения
`/wrkgrp имя файла`	Указывается имя файла рабочей группы. Применяется только к файлам базы данных Access

Как видно из приведенной таблицы, используя параметры командной строки, можно не только запустить Microsoft Access, но и сразу загрузить файл базы данных или проекта, выполнить сжатие и восстановление файла, преобразовать файл базы данных, созданный в одной из предыдущих версий Microsoft Access, в нужный формат. Параметры командной строки могут быть использованы для того, чтобы применить параметры настройки, отличные от стандартных.

Командную строку используют в двух случаях: когда запускают приложение Access из меню **Пуск | Выполнить** (Start | Run) Windows или когда создают ярлык на рабочем столе.

В первом случае нужно нажать кнопку **Пуск** (Start) Windows, а затем выбрать команду **Выполнить** (Run) в Windows NT, Windows 2000 или Windows XP. Затем в поле **Открыть** (Run) диалогового окна **Запуск программы** ввести командную строку. Для этого можно нажать кнопку **Обзор** (Browse) и найти папку, в которой установлен файл MSACCESS.EXE (по умолчанию он находится в каталоге C:\Program Files\Microsoft Office\Office11). Когда строка с именем программы появится в поле **Открыть**, введите нужные параметры справа от нее. Обратите внимание, что имя файла программы MSACCESS.EXE вместе с путем к нему и имя файла приложения, если оно присутствует в командной строке и содержит полный путь, должны обязательно быть заключены в двойные кавычки ("").

Чтобы создать ярлык для запуска приложения на рабочем столе пользователя, нужно:

1. Открыть папку, в которой находится файл MSACCESS.EXE, щелкнуть на нем правой кнопкой мыши и выбрать из контекстного меню команду **Создать ярлык** (Create Shortcut). Ярлык к этому файлу появится в этой же папке.
2. Перетащить его на рабочий стол.
3. Щелкнуть на ярлыке правой кнопкой мыши и выбрать команду **Свойства** (Properties). Откроется диалоговое окно свойств для этого ярлыка (рис. 14.17).
4. Раскрыть вкладку **Ярлык** (Shortcut). В поле **Объект** (Target) будет содержаться строка с именем запускаемого файла Access.

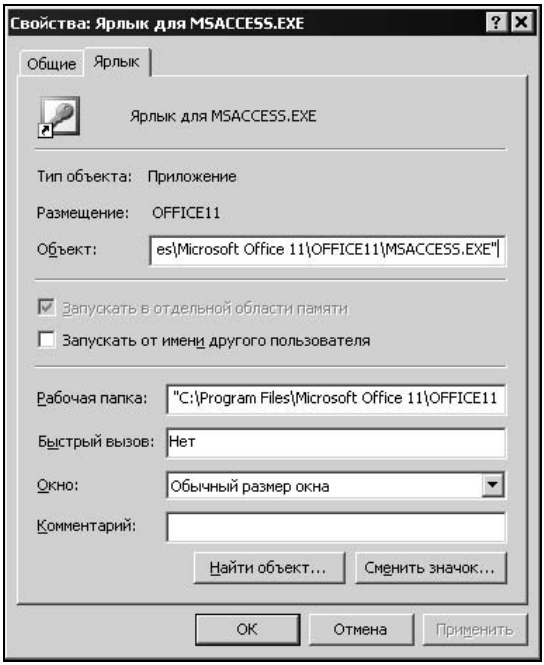


Рис. 14.17. Диалоговое окно свойств ярлыка

5. Дописать справа от этой строки нужные параметры, например, имя запускаемого приложения (не забудьте про кавычки!).
6. Можно установить для этого ярлыка другой (нестандартный) значок, воспользовавшись кнопкой **Сменить значок** (Change Icon) и затем выбрав файл, содержащий нужный значок.
7. Закрыть окно **Свойства** (Properties), нажав кнопку **ОК**.

Теперь для запуска приложения достаточно просто дважды щелкнуть на этом ярлыке.

Командный интерфейс приложения

Интерактивное управление приложением осуществляется посредством командного интерфейса. Командный интерфейс приложения Access обеспечивает возможность выбора пользователем для выполнения команд Access или действий, реализованных программно в виде макросов или процедур VBA. Есть несколько способов организации командного интерфейса приложения. Можно создать специальную строку меню или кнопочную форму, чтобы предоставить пользователю удобное средство выбора действий, которые нужно выполнить в приложении. Команды следует группировать в меню или на страницах кнопочной формы по своему назначению так, чтобы пользователю было интуитивно ясно, где должна находиться определенная команда и в каких случаях следует ее выполнять.

Использование кнопочной формы

Обычно для работы с приложением создают специальную форму, которая носит название **Главная кнопочная форма** (Main Switchboard) и содержит кнопки, позволяющие выбирать основные функции или подсистемы приложения. Такую форму можно создать, как обычно, с помощью Конструктора форм, а можно использовать специальный Мастер, который называется **Диспетчер кнопочных форм** (Swithboard Manager). В базе данных `Борей` создана **Главная кнопочная форма** (Main Switchboard) без применения Мастера. Но можно создать аналогичную форму с помощью Диспетчера кнопочных форм.

1. Выберите команду меню **Сервис | Служебные программы | Диспетчер кнопочных форм** (Tools | Database Utilities | Swithboard Manager). Поскольку база данных не содержит кнопочных форм (по крайней мере, известных Мастеру), выдается диалоговое окно, запрашивающее, нужно ли создавать кнопочную форму. Нажмите кнопку **Да** (Yes). Теперь Мастером создается таблица **Switchboard Items** (Элементы кнопочной формы) и кнопочная форма (Switchboard), которые мы рассмотрим позднее. После этого появляется диалоговое окно со списком страниц кнопочной формы (рис. 14.18). В этом окне перечисляются все страницы кнопочной формы (их может быть несколько). Пока создана только одна страница, и мы видим ее в списке.

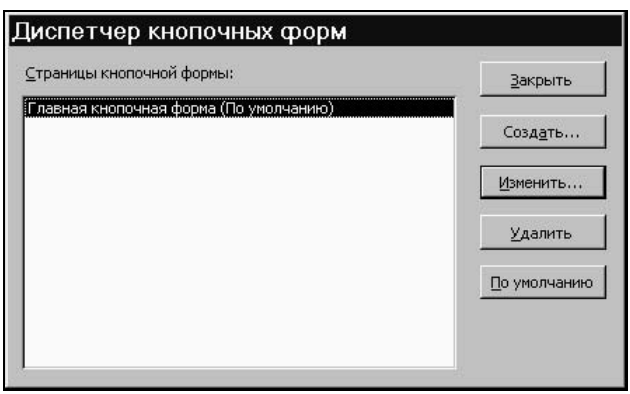


Рис. 14.18. Диалоговое окно **Диспетчер кнопочных форм**

2. Чтобы создать на странице нужные кнопки, нажмите кнопку **Изменить** (Edit). Открывается диалоговое окно **Изменение страницы кнопочной формы** (Edit Switchboard Page) (рис. 14.19), содержащее список элементов формы. В этом окне можно добавлять, изменять и удалять элементы выбранной страницы формы. Можно также менять порядок их размещения на странице (с помощью кнопок **Вниз** (Move down) и **Вверх** (Move up)). В поле **Название кнопочной формы** (Swithboard Name) мы можем изменить стандартное название, если это необходимо. Список элементов формы пока пуст и доступна только одна кнопка **Создать** (New).
3. Чтобы создать первый элемент, нажмите кнопку **Создать** (New). Появляется диалоговое окно **Изменение элемента кнопочной формы** (Edit Switchboard Item) (рис. 14.20).

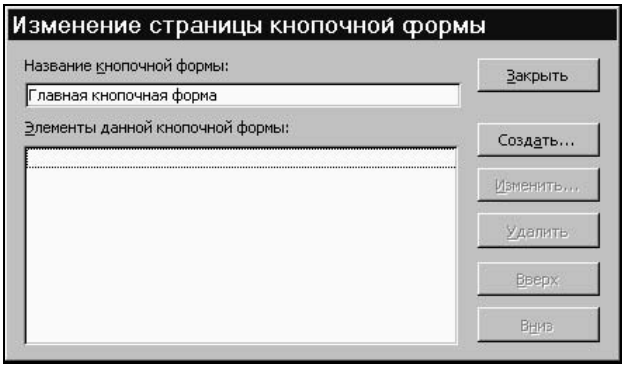


Рис. 14.19. Диалоговое окно **Изменение страницы кнопочной формы**

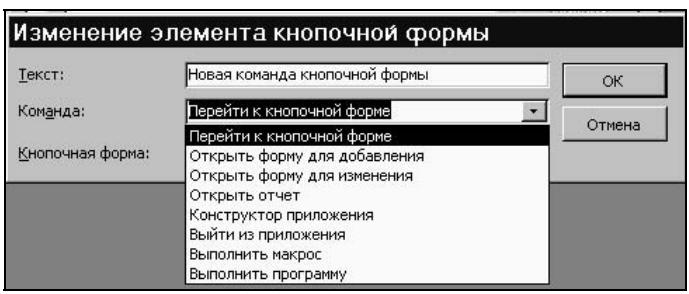


Рис. 14.20. Изменение элемента кнопочной формы

4. В поле **Текст** (Text) введите наименование создаваемой кнопки, например `Товары`. Второе поле **Команда** (Command) позволяет определить действие, которое будет выполняться после нажатия данной кнопки. Список этих действий представлен на рис. 14.20. Их всего 8. Это может быть открытие формы в режиме добавления записей или редактирования, открытие отчета, запуск макроса или процедуры VBA, запуск мастера для редактирования кнопочной формы, переход к другой кнопочной форме и выход из приложения. Выберем, например, **Открыть форму для изменения** (Open Form in Edit Mode). Тогда в следующем поле нужно ввести имя открываемой формы `Товары` (Products) (рис. 14.21) и нажать кнопку **ОК**.

Замечание

Название последнего поля в форме на рис. 14.21 изменяется в зависимости от выбранной команды во втором поле. Если выбранное в списке действие не требует задания аргумента, третье поле вообще исчезает.

5. Аналогично добавим кнопки для других объектов приложения: `Типы` (Categories), `Поставщики` (Suppliers), `Заказы` (Orders). Создадим кнопку **Отчеты**, которая будет открывать форму `Отчеты о продажах` (Sales Reports Dialog), и кнопку выхода из приложения **Выход**.
6. Закроем диалоговое окно **Изменение страницы кнопочной формы** (Edit Switchboard Page), нажав кнопку **Закрыть** (Close).

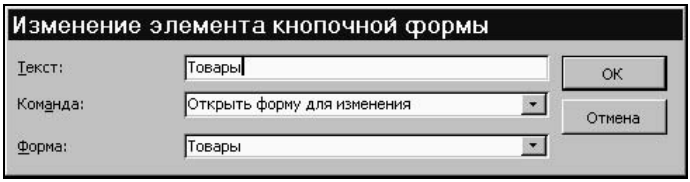


Рис. 14.21. Диалоговое окно **Изменение элемента кнопочной формы**

Количество кнопок на одной странице формы ограничено числом 8. Это не значит, что их нельзя создать больше, просто это придется делать вручную. Однако есть другой способ преодолеть это ограничение. Если число действий, которые хотелось бы разместить на кнопки, превышает 8, можно сгруппировать часть действий на отдельной странице кнопочной формы. Для примера создадим еще одну страницу нашей кнопочной формы и разместим на ней кнопки действий с клиентами. Для этого:

1. В диалоговом окне **Диспетчер кнопочных форм** (Switchboard Manager) нажмите кнопку **Создать** (New). В появившемся диалоговом окне (рис. 14.22) введите имя новой страницы `Клиенты` и нажмите кнопку **OK**.

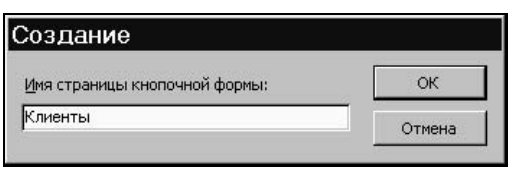


Рис. 14.22. Диалоговое окно ввода имени новой страницы

2. Далее в редакторе страниц уже известным способом создадим элементы: **Клиенты**, **Наклейки для клиентов**, **Телефоны клиентов**, **Заказы клиентов**, которые открывают соответствующие формы. Не забудьте создать кнопку, которая позволит вернуться в **Главную кнопочную форму** (Main Switchboard). Для этого нужно связать с ней команду **Перейти к кнопочной форме** (Go to Switchboard) (рис. 14.23).



Рис. 14.23. Добавление ссылки на главную страницу кнопочной формы

3. Закройте это окно, чтобы вернуться к диалоговому окну **Диспетчер кнопочных форм** (Switchboard Manager). В списке страниц у вас должны появиться две страницы.
4. Теперь нужно отредактировать первую страницу — добавить к ней элемент, с помощью которого можно будет перейти на вторую страницу. Для этого выде-

лите в списке страницу **Главная кнопочная форма** (Main Switchboard) и нажмите кнопку **Изменить** (Edit). В окне редактора страниц добавьте новый элемент Клиенты, также используя команду **Перейти к кнопочной форме** (Go to Switchboard).

Теперь обе страницы оказались связанными. Можно закрыть окно Мастера и посмотреть, что у нас получилось, открыв форму **Кнопочная форма** (Switchboard). Должно появиться окно, похожее на то, что представлено на рис. 14.24.

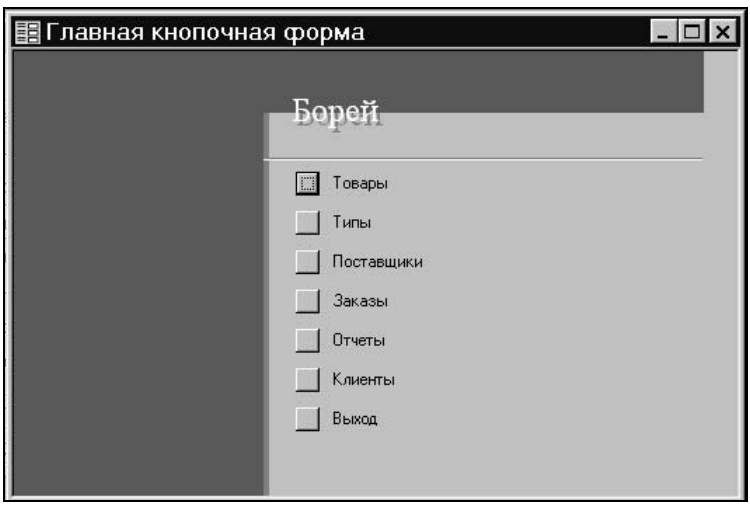


Рис. 14.24. Главная кнопочная форма

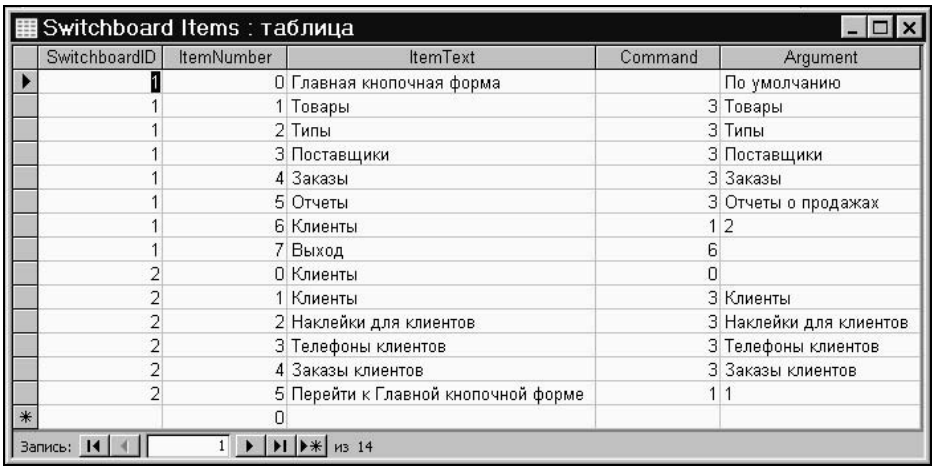
Вы можете проверить, как действуют кнопки. Например, если нажать кнопку **Клиенты**, содержание формы изменится, и вы увидите вторую страницу формы. Форма выглядит не слишком привлекательно, но теперь вы можете перейти в режим Конструктора и оформить ее внешний вид, применив все средства Конструктора.

Открыв эту форму в режиме Конструктора, вы не увидите ее элементов, форма кажется совсем пустой. Однако если вы откроете модуль, связанный с этой формой, то увидите достаточно большой по объему код, который и организует работу формы должным образом. Причем для этого используется информация, хранящаяся в специальной таблице, о которой мы упоминали в начале раздела: **Switchboard Items** (Элементы кнопочной формы). Вы увидите эту таблицу в списке **Таблицы** (Tables) окна **База данных** (Database). В этом случае она будет выглядеть примерно так, как показано на рис. 14.25.

Первый столбец содержит идентификатор страницы, второй — номер элемента на странице. Этот номер определяет порядок, в котором будут располагаться элементы на странице. Третий столбец содержит текст, появляющийся справа от кнопки. Последние два столбца определяют команду, связанную с элементом: код команды (как мы помним, их 8) и ее аргумент. Код 0 соответствует специальному элементу, который задает заголовок страницы и определяет ее начало.

Таким образом, с помощью Диспетчера кнопочных форм (Switchboard Manager) можно создать иерархическую систему страниц кнопочной формы, которая может

быть альтернативой обычному иерархическому меню. Правда, в отличие от меню эта форма не видна постоянно на экране, она может закрываться другими формами. Если вы хотите обеспечить возможность пользователю открывать сразу несколько форм и отчетов, вам придется сделать кнопку на панели инструментов, которая позволяла бы в любой момент отобразить **Главную кнопочную форму** (Main Switchboard).



Switchboard Items : таблица

SwitchboardID	ItemNumber	ItemText	Command	Argument
1	0	Главная кнопочная форма		По умолчанию
1	1	Товары	3	Товары
1	2	Типы	3	Типы
1	3	Поставщики	3	Поставщики
1	4	Заказы	3	Заказы
1	5	Отчеты	3	Отчеты о продажах
1	6	Клиенты	1	2
1	7	Выход	6	
2	0	Клиенты	0	
2	1	Клиенты	3	Клиенты
2	2	Наклейки для клиентов	3	Наклейки для клиентов
2	3	Телефоны клиентов	3	Телефоны клиентов
2	4	Заказы клиентов	3	Заказы клиентов
2	5	Перейти к Главной кнопочной форме	1	1
*	0			

Запись: 1 из 14

Рис. 14.25. Таблица **Switchboard Items**

Создание и изменение панелей команд

Если мы вас не убедили создавать кнопочные формы для управления приложением, вам придется создать для него меню. Это более привычный способ организации действий пользователя.

Альтернативой меню, а чаще дополнением к нему, являются панели инструментов. Во всяком случае, если при разработке вы чаще пользуетесь кнопками на панелях инструментов, чем командами меню, вам наверняка захочется создать такие же удобные кнопки для пользователя и дать возможность осуществлять операции с помощью полезных стандартных кнопок.

Для этого существует очень удобное средство, которое называется панели команд (Command bars). *Панели команд* — это объекты Microsoft Office, которые обеспечивают стандартный способ организации взаимодействия пользователя с приложением Microsoft Office. Они бывают трех типов:

- строка меню (Menu Bar);
- контекстное меню (Popup);
- панели инструментов (Toolbar).

В приложении можно использовать как стандартные панели команд Access, настроив их нужным образом, так и специальные пользовательские панели команд. Обычно в стандартных панелях команд удаляют лишние кнопки, чтобы пользователь не

мог что-либо испортить в приложении, либо наоборот — нужные кнопки со стандартных панелей переносят на пользовательские панели инструментов. Общие правила создания и функционирования панелей команд различных типов позволяют легко переносить кнопки с панелей инструментов в строки меню и обратно. Как и в самом Access, вы можете создать панели команд, которые относятся к приложению в целом, появляются при запуске приложения и остаются на экране в течение всего сеанса работы с приложением. С другой стороны, панели команд можно связать с конкретной формой или отчетом, и они будут появляться на экране только тогда, когда открыта данная форма или отчет, для чего необходимо только установить значения соответствующих свойств формы или отчета: **Строка меню** (Menu Bar), **Панель инструментов** (Toolbar), **Контекстное меню** (Popup).

Создание и изменение панелей команд любого типа выполняется в режиме настройки, который может быть вызван либо с помощью команды меню **Вид | Панели инструментов | Настройка** (View | Toolbars | Customize), либо из контекстного меню, которое выводится на экран после щелчка правой кнопкой мыши в области панелей команд приложения.

Создание меню

Для создания меню:

1. Откройте окно **Настройка** (Customize), например, щелкнув правой кнопкой мыши на панели инструментов и выбрав команду **Настройка** (Customize). Таким способом осуществляется переход в режим настройки панелей команд.
2. По умолчанию открывается вкладка **Панели инструментов** (Toolbars). (Если у вас открылась другая вкладка, раскройте именно эту.) Пока в списке на данной вкладке отображаются только стандартные панели команд. К этому списку нужно добавить новую панель команд типа **Строка меню** (Menu Bar). Нажмите кнопку **Создать** (New).
3. В диалоговом окне, запрашивающем имя создаваемой панели команд, введите название: `Главное меню` (рис. 14.26) и нажмите кнопку **OK**.

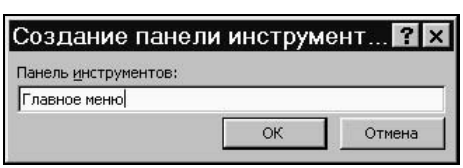


Рис. 14.26. Создание новой панели команд

4. Появится маленькая пустая панель, которая по умолчанию является панелью инструментов. Чтобы превратить ее в строку меню, необходимо изменить ее тип. Для этого выделите элемент **Главное меню** в списке на вкладке **Панели инструментов** (Toolbars) (он добавляется в конец списка) и нажмите кнопку **Свойства** (Properties). Откроется диалоговое окно **Свойства панели инструментов** (Toolbar Properties) (рис. 14.27).
5. В раскрывающемся списке **тип** (Type) выберите значение **Строка меню** (Menu Bar).

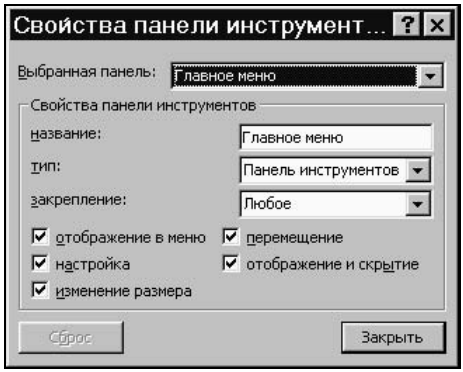


Рис. 14.27. Диалоговое окно **Свойства панели инструментов**

Кроме типа, для данной панели команд можно задать еще ряд свойств:

- **закрепление** (Docking). Можно задать расположение панели команд на экране: в любом месте, только по горизонтали, только по вертикали или вообще запретить изменять заданное расположение;
- **отображение в меню** (Show on Toolbars Menu). Установите этот флажок, если вы хотите, чтобы название данной панели команд отображалось в списке, появляющемся при выполнении команды меню **Вид | Панели инструментов** (View | Toolbars);
- **настройка** (Allow Customizing). Сброс этого флажка запретит изменение панели команд;
- **изменение размера** (Allow Resizing). Если этот флажок не установлен, изменение размеров данной панели команд будет невозможно;
- **перемещение** (Allow Moving). Сбросьте этот флажок, если нужно запретить перемещение панели команд по экрану;
- **отображение и скрытие** (Allow Showing/Hiding). Когда этот флажок сброшен, пользователь не будет иметь возможности скрывать и снова отображать панель команд на экране.

Пока строка меню пустая. Теперь в нее можно добавить меню — раскрывающиеся списки команд. Чтобы добавить меню на панель команд:

1. Раскройте вкладку **Команды** (Commands) в диалоговом окне **Настройка** (Customize).
2. Выделите в списке **Категории** (Categories) значение **Новое меню** (New Menu).
3. Перетащите элемент **Новое меню** (New Menu) из списка **Команды** (Commands) на новую панель.

Теперь вы увидите на ней элемент **Новое меню** (New Menu) (рис. 14.28).

Изменить название созданного меню можно в окне свойств. Для этого необходимо щелкнуть правой кнопкой мыши по меню и выбрать в контекстном меню команду **Свойства** (Properties). Появится диалоговое окно **Свойства элемента** (Control Properties) (рис. 14.29).

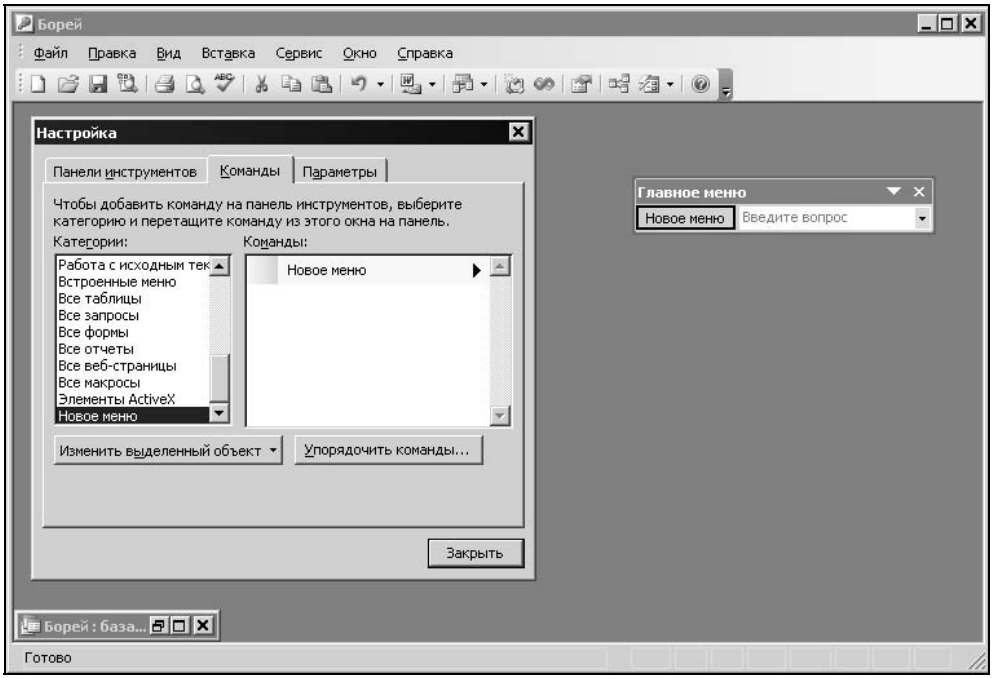


Рис. 14.28. Создание пользовательского меню

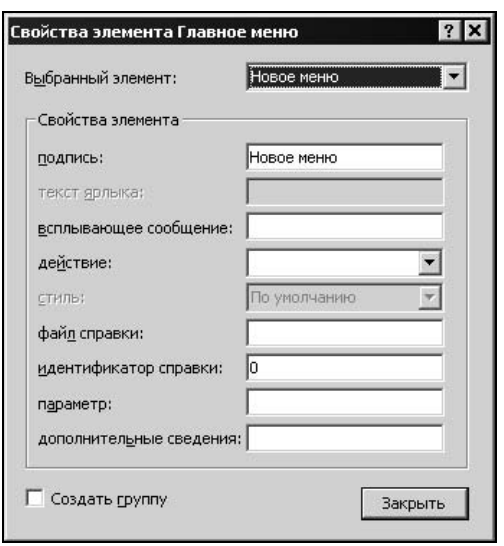


Рис. 14.29. Диалоговое окно **Свойства элемента**

В этом окне можно задать следующие значения:

1. В поле **подпись** (Caption) введите название меню: `Клиенты`.
2. В поле **всплывающее сообщение** (ScreenTip) можно ввести текст всплывающей подсказки.
3. Поле **действие** (On Action) следует оставить пустым. Обычно оно определяет действие, выполняемое при выборе данного элемента панели команд. При выборе меню **Клиенты** должен раскрываться список команд данного меню. Это будет выполняться автоматически, т. к. в качестве элемента строки меню мы добавили меню.
4. Поле **стиль** (Style) позволяет определить вид элемента панели команд, в данном случае — меню. Раскрывающийся список имеет следующие значения:
   - **По умолчанию** (Default Style) — на элементе меню выводятся одновременно значок и надпись;
   - **Только текст (всегда)** (Text only (Always)) — и на элементе меню и на кнопке выводится только надпись;
   - **Только текст (в меню)** (Text only (In Menus)) — выводит надпись на элементе меню и значок на кнопке;
   - **Рисунок и текст** (Image and Text) — и на элементе меню и на кнопке выводится как значок, так и надпись.
5. Два поля — **файл справки** (Help File) и **идентификатор справки** (HelpContextID) — позволяют связать с данным элементом панели команд раздел из справочной системы приложения, который будет отображаться при нажатии клавиши <F1>, если данный элемент выделен.
6. Поле **параметр** (Parameter) связано с полем **действие** (On Action) и определяет значение аргумента функции, если в поле **действие** (On Action) определен вызов функции. В данном случае оно тоже остается пустым.
7. В поле **дополнительные сведения** (Tag) можно ввести информацию, которая будет доступна из процедуры VBA.
8. Установленный флажок **Создать группу** (Begin a Group) означает, что данный элемент панели команд является началом группы и слева от него (в этом случае в строке меню) будет отображаться разделительная линия.

Закройте окно **Свойства элемента**, теперь будем добавлять в меню команды. Для этого можно использовать ту же вкладку **Команды** (Commands) диалогового окна **Настройка** (Customize). Выбрав категорию, а затем команду из выбранной категории, можно перетащить ее в данное меню. Например:

1. Выберите категорию **Все формы** (All Forms).
2. Перетащите в меню те формы, которые мы использовали при создании кнопочной формы в предыдущем разделе: `Клиенты` (Customers), `Наклейки для клиентов` (Customer Label Dialog), `Телефоны клиентов` (Customer Phone List), `Заказы клиентов` (Customer Orders) (рис. 14.30).

Совет

Перед тем как перетаскивать первую команду в меню, щелкните по нему правой кнопкой мыши — около него отобразится маленькая серая панель пустого меню.

Туда и нужно перетащить первую команду. При перетаскивании остальных команд место вставки помечается горизонтальной линией. Новая команда размещается под этой линией. При вставке меню в строку меню место вставки помечается вертикальной линией. Новое меню размещается справа от этой линии.

Чтобы добавить в строку меню следующее меню со списком команд, нужно снова выбрать на вкладке **Команды** (Commands) диалогового окна **Настройка** (Customize) категорию **Новое меню** (New Menu), затем элемент **Новое меню** (New Menu) и перетащить его в данное меню, поместив справа от первого элемента. Чтобы изменить вид меню или подменю, щелкните по соответствующему элементу правой кнопкой мыши. Можно использовать команды появившегося при этом контекстного меню или открыть окно **Свойства элемента** (Control Properties).

Удаление меню выполняется очень просто. Достаточно в режиме настройки панелей команд перетащить соответствующий элемент в пустую область окна Microsoft Access. Аналогично можно перетаскивать подменю и команды меню из одних меню в другие, также можно команду меню превратить в кнопку на панели инструментов и наоборот. Поскольку меню и панели инструментов являются разными типами одного объекта — панели команд, на панели инструментов можно размещать меню, а в строку меню можно добавлять кнопки.

Замечание

Выполнять все описанные действия можно только в режиме настройки панелей команд, т. е. при открытом окне **Настройка** (Customize).

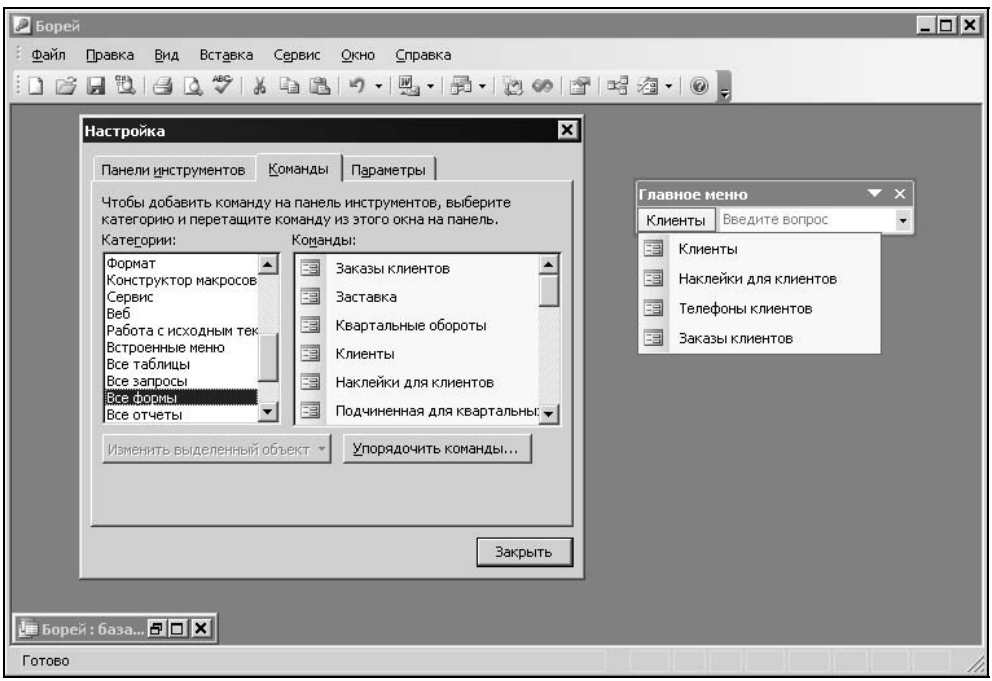


Рис. 14.30. Добавление команд в меню

Чтобы окончательно придать меню профессиональный вид (как в стандартном меню Access), необходимо каждому меню и команде назначить клавиши доступа. Буквы, соответствующие этим клавишам, выделяются в названии элемента подчеркиванием. Клавиши доступа используются следующим образом. Для того чтобы раскрыть меню, нужно при нажатой клавише <Alt> нажать клавишу, соответствующую подчеркнутой букве в названии этого меню. Например, комбинация клавиш <Alt>+<Ф> (<Alt>+<F>) раскрывает меню **Файл** (File). Далее, чтобы выбрать команду меню, достаточно просто нажать клавишу, соответствующую подчеркнутой букве в названии команды.

Чтобы указать символ, который будет определять клавишу доступа, нужно в имени команды (в поле **подпись** (Caption) окна свойств элемента меню) вставить перед этим символом знак `&`, например `&Товары`. При этом в разных командах одного меню должны быть использованы обязательно разные символы, определяющие клавиши доступа, хотя в разных меню символы могут совпадать.

Создание и изменение панелей инструментов

Для того чтобы обеспечить быстрый доступ к наиболее часто используемым формам, отчетам или макросам, можно создать специальные кнопки, которые помещаются либо на панель инструментов, либо непосредственно в меню. Создать эти кнопки очень просто:

1. Закройте окно **Настройка** (Customize).
2. Раскройте список форм в окне базы данных приложения `Борей`, выделите форму `Товары` (Products) и перетащите ее в строку меню, которое мы только что создали. В строке меню появится кнопка со значком и названием формы (рис. 14.31), при нажатии которой открывается форма `Товары` (Products).

Как видите, в одной строке меню могут размещаться как подчиненные меню, так и кнопки. Точно так же можно добавить кнопку в одну из стандартных панелей инструментов. Если кнопок, которые нужно создать для вызова различных функций приложения, много, лучше создать отдельную панель инструментов, в которую включить все эти кнопки, а также полезные кнопки со стандартных панелей инструментов.

Создается пользовательская панель инструментов аналогично строкам меню:

1. Снова откройте окно **Настройка** (Customize), щелкнув правой кнопкой мыши в области панелей команд окна приложения.
2. В появившемся диалоговом окне раскройте вкладку **Панели инструментов** (Toolbars).
3. Нажмите кнопку **Создать** (New).
4. В появившемся диалоговом окне введите имя новой панели инструментов `Мои макросы` и нажмите кнопку **ОК**. Появится пустая панель инструментов.
5. В данном случае не нужно менять ее тип (по умолчанию создается именно панель инструментов), раскройте вкладку **Команды** (Commands).
6. Выберите в списке **Категории** (Categories) значение **Все макросы** (All Macros), а в списке **Команды** (Commands) найдите макрос `Менеджеры по продажам`.
7. Перетащите этот макрос с помощью мыши на пустую панель инструментов.

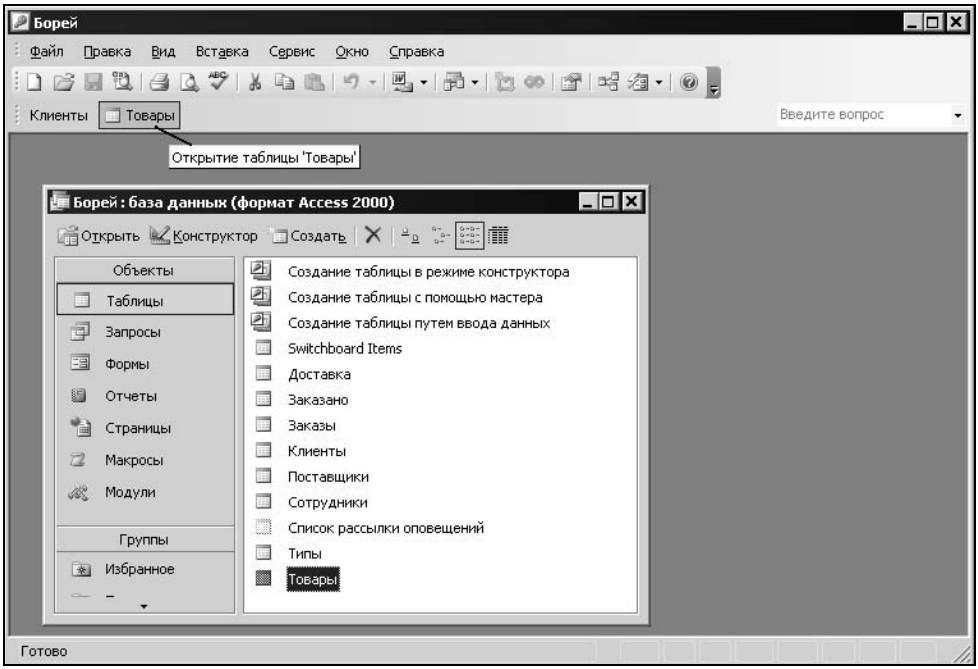


Рис. 14.31. Добавление кнопки в строку меню

Вы можете изменить внешний вид данной кнопки. Для этого в диалоговом окне **Настройка** (Customize) нажмите кнопку **Изменить выделенный объект** (Modify Selection) или щелкните правой кнопкой мыши по только что созданной кнопке. Отобразится контекстное меню, представленное на рис. 14.32. При этом режим отображения кнопки помечен как **Значок и текст** (Image and Text). Но можно поместить на кнопке или только значок, или только текст. Если вы хотите изменить текст на кнопке, введите этот текст в поле **Имя** (Name) (по умолчанию в этом поле отображается имя макроса).

Вы можете поместить на кнопку другой значок, выбрав понравившуюся картинку на специальной панели. Для отображения набора стандартных значков выделите в контекстном меню команду **Выбрать значок для кнопки** (Change Button Image) и щелкните по понравившемуся значку. Если же ни одна из картинок не подходит, можно выбрать наиболее близкий по оформлению значок, а потом исправить его в специальном редакторе (рис. 14.33). Чтобы открыть окно **Редактор кнопок** (Button Editor), выберите в контекстном меню команду **Изменить значок на кнопке** (Edit Button Image).

Наконец, можно просто скопировать понравившийся вам значок с другой кнопки.

Чтобы скопировать значок с одной кнопки на другую:

1. Откройте диалоговое окно **Настройка** (Customize) и выделите ту кнопку на панели инструментов, с которой нужно скопировать значок.
2. Нажмите кнопку **Изменить выделенный объект** (Modify Selection) в диалоговом окне **Настройка** (Customize).

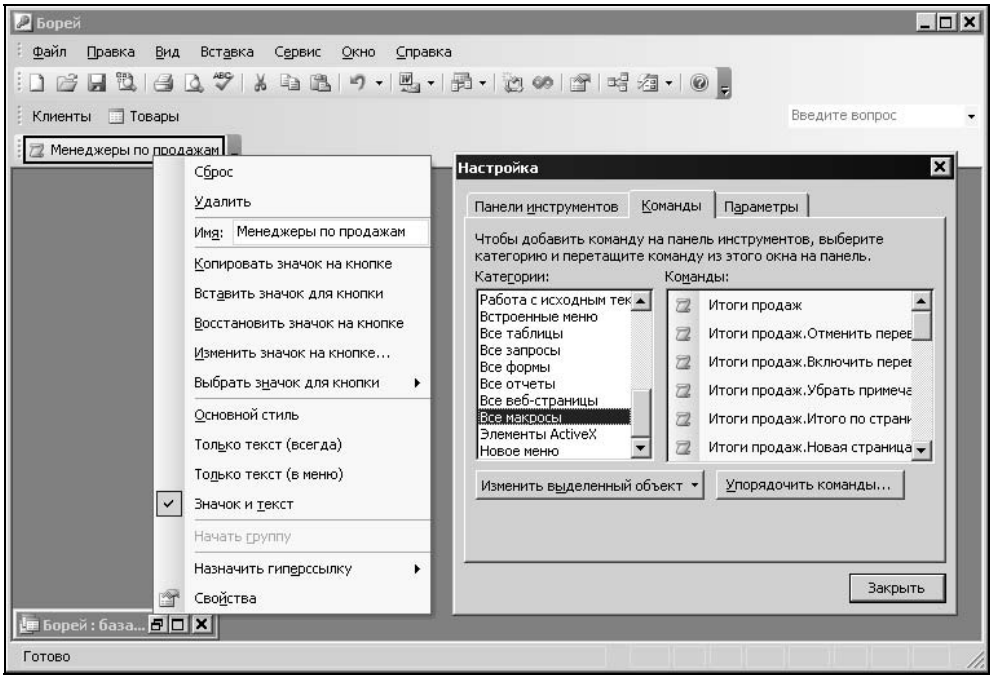


Рис. 14.32. Контекстное меню

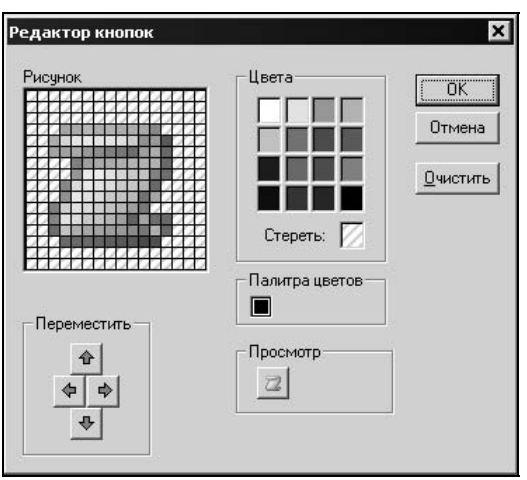


Рис. 14.33. Редактор кнопок для изменения значка на кнопке

3. Выберите в контекстном меню команду **Копировать значок на кнопке** (Copy Button Image).
4. Выделите кнопку, значок которой нужно изменить.

5. Нажмите кнопку **Изменить выделенный объект** (Modify Selection) в диалоговом окне **Настройка** (Customize).
6. Выберите в контекстном меню команду **Вставить значок для кнопки** (Past Button Image).

С помощью диалогового окна свойств кнопки можно изменить текст всплывающей подсказки и задать клавиши быстрого вызова.

Чтобы изменить текст всплывающей подсказки для кнопки на панели инструментов:

1. В режиме настройки выделите кнопку на панели инструментов. В диалоговом окне **Настройка** (Customize) нажмите кнопку **Изменить выделенный объект** (Modify Selection).
2. Выберите в контекстном меню команду **Свойства** (Properties). Появится диалоговое окно свойств кнопки.
3. Введите желаемый текст в поле **Всплывающая подсказка** (ScreenTip).

На созданной вами панели инструментов можно разместить любые другие полезные кнопки, не только кнопки для запуска макросов. Когда создание панели инструментов закончено, вы можете переместить ее на свободное место в верхней части экрана (или поместить у любой другой границы экрана).

Создание и удаление контекстных меню

Контекстное меню — это меню специального вида, которое появляется при нажатии правой кнопки мыши. Обычно контекстные меню создаются на последнем этапе разработки приложения. Создание контекстных меню очень похоже на создание обычных строк меню. Только в окне **Свойства панели инструментов** (Toolbar Properties) (см. рис. 14.27) в раскрывающемся списке **тип** (Type) должно быть выбрано значение **Контекстное меню** (Popup). При этом пользователь получает предупреждение, что данное контекстное меню исчезнет с экрана. Чтобы посмотреть или изменить контекстные меню, нужно выбрать (установить флажок) **Контекстные меню** (Shortcut Menus) в списке панелей команд на вкладке **Панели инструментов** (Toolbars) диалогового окна настройки панелей команд. Тогда на экране появится панель контекстных меню. Создаваемые пользователем контекстные меню отображаются в меню **Настраиваемый** (Custom) (рис. 14.34).

Добавлять команды в контекстное меню нужно точно так же, как в обычное меню.

Чтобы удалить контекстное меню:

1. Удалите все команды из этого меню, перетаскивая их по очереди на свободное место в окне Microsoft Access.
2. Раскройте вкладку **Панели инструментов** (Toolbars) в диалоговом окне **Настройка** (Customize) и нажмите кнопку **Свойства** (Properties).
3. В раскрывающемся списке **Выбранная панель** (Selected toolbar) выберите нужное контекстное меню.
4. В раскрывающемся списке **Тип** (Type) выберите элемент **Панель инструментов** (Toolbar), чтобы изменить тип удаляемого объекта, и нажмите кнопку **Закрыть** (Close).

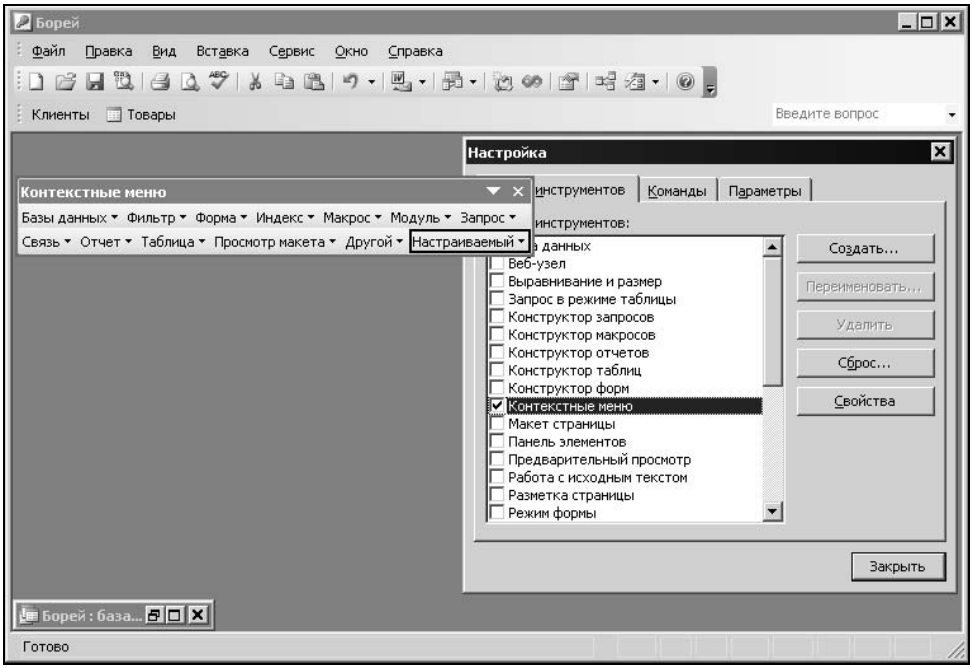


Рис. 14.34. Панель контекстных меню

5. Выделите в списке **Панели инструментов** (Toolbars) соответствующее название панели команд и нажмите кнопку **Удалить** (Delete).

Изменение порядка команд

В Access 2003 появилась новая кнопка на вкладке **Команды** (Commands) диалогового окна **Настройка** (Customize). Она называется **Упорядочить команды** (Rearrange Commands). При нажатии этой кнопки открывается диалоговое окно **Изменение порядка команд** (Rearrange Commands), в котором отображается список команд пользовательских панелей инструментов и можно, пользуясь кнопками **Вверх** (Move Up) и **Вниз** (Move Down), изменить порядок команд в выбранной строке меню или кнопок панели инструментов (рис. 14.35). Кнопки **Добавить** (Add) и **Удалить** (Delete) позволяют соответственно добавлять в меню новые команды и удалять существующие. При нажатии кнопки **Изменить выделенный объект** (Modify Selection) появляется уже знакомое контекстное меню (см. рис. 14.32) с командами, позволяющими настроить внешний вид кнопок и команд меню и панелей инструментов. Таким образом диалоговое окно **Изменение порядка команд** (Rearrange Commands) позволяет не только упорядочить команды и кнопки меню, а и полностью настраивать панели команд.

Связывание строк меню, панелей инструментов и контекстных меню с формами и отчетами

Созданные строки меню, контекстные меню и панели инструментов можно связывать с формами и отчетами. Для этого необходимо определить соответствующие

свойства формы или отчета. Список этих свойств для формы показан на рис. 14.36. Есть аналогичные свойства и у отчета (кроме параметров, определяющих контекстное меню).

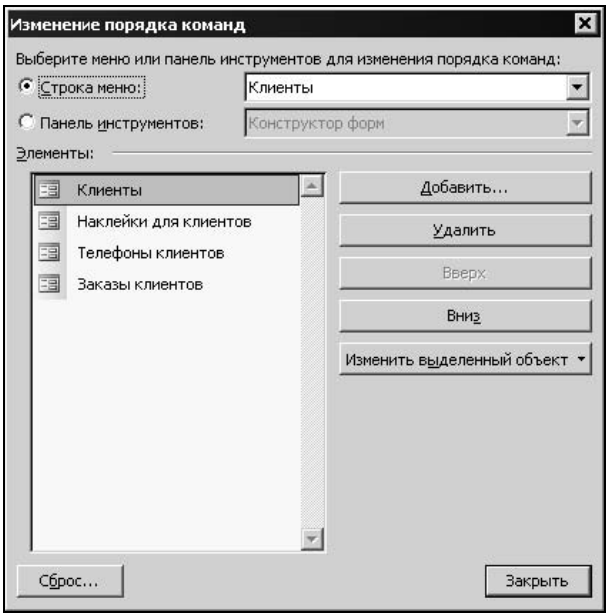


Рис. 14.35. Диалоговое окно **Изменение порядка команд**

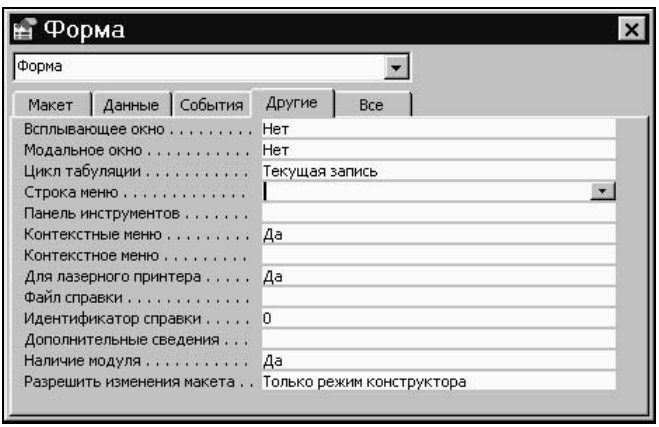


Рис. 14.36. Свойства формы

Свойство **Контекстные меню** (Shortcut Menu) позволяет разрешить или запретить вывод на экран контекстного меню для этой формы. Причем, если специальное контекстное меню в поле **Контекстное меню** (Shortcut Menu Bar) не указано, а поле

Контекстные меню (Shortcut Menu) имеет значение **Да** (Yes), то при щелчке правой кнопкой мыши по этой форме будет выводиться стандартное контекстное меню.

Следует еще отметить, что контекстное меню может быть связано не только с формой, но и с элементом формы, т. к. элемент управления имеет свойство **Контекстное меню** (Shortcut Menu Bar) на вкладке **Другие** (Others) (рис. 14.37).

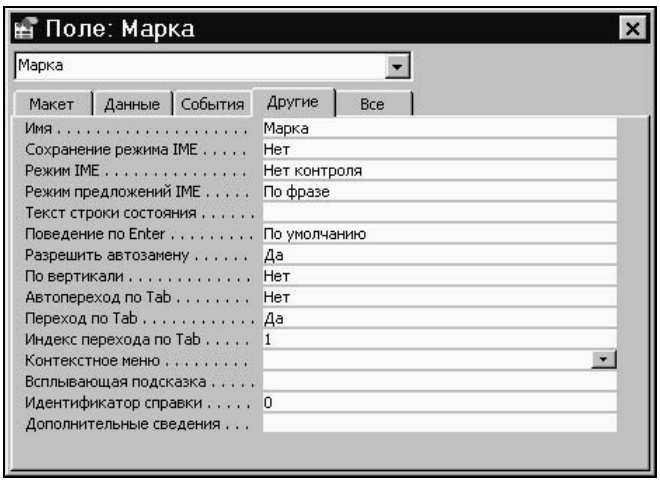


Рис. 14.37. Свойства элемента управления формы

Замечание

Все изложенные ранее способы изменения меню и панелей инструментов можно применять также к встроенным меню и панелям инструментов Access, настраивая их нужным образом. Если требуется создать новую панель инструментов, доступную во всех базах данных, с которыми вы работаете, используйте встроенные панели инструментов **Служебная программа 1** (Utility 1) и **Служебная программа 2** (Utility 2) (они отличаются от создаваемых вами панелей инструментов тем, что доступны не только в том приложении, в котором вы их создали).

Изменение меню и панелей инструментов с помощью VBA

Изложение возможностей создания и настройки меню и панелей инструментов было бы не полным, если бы мы ничего не сказали о возможностях программного управления этими объектами. Такие возможности понадобятся вам, если нужно, например:

- сделать недоступным меню, команду меню или кнопку на панели инструментов;
- скрыть или показать меню, команду меню или кнопку панели инструментов;
- скрыть или показать строку меню или панель инструментов.

Чаще всего приходится изменять панели команд, разрешая или запрещая доступ к отдельным их элементам — командам меню или кнопкам. Это может быть связано либо с разграничениями прав доступа, либо с необходимостью запретить при опре-

деленных условиях выполнение некоторых команд. Для доступа к панелям команд и их элементам используется семейство `CommandBars` объектной модели Microsoft Office *(см. гл. 13)*. Ссылка на конкретную строку меню или панель инструментов создается стандартным способом: с помощью оператора `Set`. Далее приведен пример функции, которая позволяет изменить значение свойства **Доступ** (Enabled) элемента панели команд.

```
Function SetEnabled (barName As String, index As Integer, state As Boolean)
    Dim cbar As Object
    Set cbar = CommandBars(barName)
    cbar.Controls(index).Enabled = state
    SetEnabled = True
End Function
```

В качестве параметров в функцию передаются имя панели команд `barName`, индекс элемента панели `index` и значение свойства `state`.

Переменная `cbar` будет содержать ссылку на нужную панель (обращение к объекту семейства `CommandBars` в данном случае выполняется по имени объекта).

Семейство `Controls` содержит все элементы панели. Для доступа к конкретному элементу используется индекс, соответствующий номеру элемента в семействе. В отличие от других семейств объектной модели VBA в семействе `Controls` элементы нумеруются не с нуля, а с единицы. Это означает, что для ссылки на первый элемент семейства необходимо использовать выражение `CommandBar.Controls(1)` вместо ожидаемого `CommandBar.Controls(0)`.

Аналогичная функция может быть использована для того, чтобы скрыть или показать отдельные меню или кнопки на панели команд, только изменяться должно свойство `Visible` того же самого семейства `Controls`:

```
Function SetVisible (barName As String, index As Integer, state As _
Boolean)
    Dim cbar As Object
    Set cbar = CommandBars(barName)
    cbar.Controls(index).Visible = state
    SetVisible = True
End Function
```

Созданные вами пользовательские меню и панели инструментов должны в нужное время появляться на экране. В большинстве случаев это обеспечивается тем, что создается одна общая строка меню и одна общая панель инструментов, которые выводятся на экран при запуске приложения и в нужное время заменяются другими меню и панелями инструментов, которые связаны с формами и отчетами. При этом встроенные меню и панели инструментов Access должны быть отключены *(см. разд. "Настройка параметров запуска" ранее в данной главе)*.

Есть возможность и программно управлять выводом на экран строк меню и панелей инструментов. Например, чтобы задать свою собственную строку меню в качестве главной строки меню приложения, можно либо изменить параметр **Строка меню** (Menu Bar) в окне **Параметры запуска** (Startup), либо установить свойство `MenuBar` объекта `Application` *(см. разд. "Объектные модели Microsoft Office 2003" гл. 13)*:

```
Application.MenuBar = "Главное меню"
```

Для того чтобы показывать или прятать панели инструментов в процессе работы программы, можно использовать макрокоманду `ShowToolbar` (ПанельИнструментов). Эта макрокоманда имеет два аргумента: имя панели инструментов и значение **Показать** (Show) (рис. 14.38).

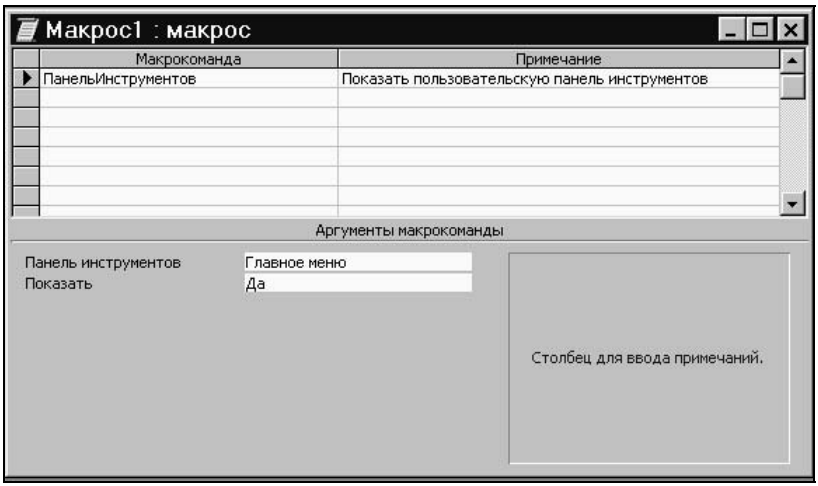


Рис. 14.38. Макрокоманда `ПанельИнструментов`

Поле **Показать** (Show) на панели аргументов макроса позволяет определить три значения:

- `Да` (Yes) — панель инструментов будет всегда отображаться на экране;
- `В обычном режиме` (Where Appropriate) — панель инструментов будет отображаться только при необходимости, данный режим используется для встроенных панелей инструментов;
- `Нет` (No) — панель инструментов будет скрыта.

Вместо макроса можно использовать метод `ShowToolbar` объекта `DoCmd`. Пример, приведенный на рис. 14.38, может быть выражен в коде VBA следующим образом:

```
DoCmd ShowToolbar "Главное меню", acToolbarYes
```

где `acToolbarYes` — встроенная константа Access.

ГЛАВА 15



Интеграция Access 2003 с компонентами Office 2003

Хотя СУБД Access является прекрасным инструментом для создания различных приложений, выполняющих обработку данных в связанных таблицах, однако некоторые операции с данными гораздо эффективнее решаются другими приложениями, входящими в семейство Microsoft Office System 2003. И при разработке приложения возникает желание создать запросы и формы Access, добавить к ним аналитические возможности Excel, обеспечить редактирование отчетов Access в редакторе Word, связать все это с текущей таблицей контактов Outlook или личным календарем и т. д. То есть, чтобы получить приложение для офиса, удовлетворяющее современным требованиям, необходимо обеспечить совместное функционирование различных приложений Microsoft Office. И нужно сказать, что продукты этого семейства прекрасно интегрируются друг с другом, что позволяет решить как перечисленные, так и многие другие задачи.

Основой для организации такого взаимодействия является технология ActiveX, которая предоставляет разработчику два мощных механизма интеграции приложений:

- *OLE* (Object Linking and Embedding) — механизм, который позволяет связывать или внедрять объекты, созданные другими приложениями Microsoft Office, в формы и отчеты Access.
- *Автоматизация* (Automation) — технология, позволяющая управлять из приложения Access объектами, которые созданы и поддерживаются другими приложениями Microsoft Office.

О том, как используются указанные механизмы и чем они отличаются, будет рассказано в данной главе.

Замечание

На самом деле понятия OLE и Automation шире, чем это определено ранее, они касаются самых разных приложений, которые поддерживают технологию ActiveX, а не только приложений семейства Microsoft Office, однако в рамках данной книги мы будем рассматривать их именно в таком контексте, как мы определили.

Еще одним механизмом, обеспечивающим совместную работу приложений Microsoft Office System 2003, является язык XML, который используется в первую очередь для

обмена данными между ними. Но вместе с тем, применяя описанные в данной главе механизмы автоматизации, можно выполнять непосредственное программное управление документами XML. Механизмы обмена данными с помощью XML-файлов описывались в *гл. 3* и здесь рассматриваться не будут. Но мы познакомимся вкратце с объектной моделью документов XML.

Итак, в настоящей главе будут затронуты следующие вопросы:

- использование протокола OLE и связанных с ним моделей построения интегрированных приложений;
- вставка рисунков в Access, демонстрирующая возможности OLE;
- совместное использование Access с Word и Excel;
- общие понятия автоматизации и иллюстрация работы автоматизации внутри комплекта Microsoft Office;
- использование элементов ActiveX;
- программный доступ к элементам документов XML.

Связывание и внедрение объектов

OLE — это средство, позволяющее создавать *составные документы*, которые включают в себя документы, созданные другими приложениями. Документ, который включает в себя другие документы, называется *документом-контейнером OLE*. В данном случае документами-контейнерами являются формы и отчеты Access. Документы, которые включаются в форму или отчет, называются *документами-источниками* или *объектами OLE*. Объектами OLE могут быть документы Word, Excel, рисунки, созданные в одном из графических редакторов, например Paint, видеоролики (файлы с расширением avi), звуковые файлы с расширением wav. Объекты OLE отличаются от объектов Automation, о которых мы будем говорить далее, тем, что они являются документами, получаемыми с помощью приложения, а не частью его модели объектов.

Объекты OLE могут быть либо *внедрены* в документ-контейнер, либо *связаны* с ним. Приложение, которое поставляет объекты для внедрения и связывания, называется *сервером OLE*. Внедренный объект представляет собой копию документа-источника, который сохраняется вместе с формой или отчетом. Связанный объект хранится в отдельном файле, и документ-контейнер содержит только указатель на исходный файл объекта. Если кто-либо обновляет исходный файл объекта, то обновляется и представление объекта в составном документе. Однако коварство связанных объектов заключается в том, что при изменении местоположения исходного файла относительно составного документа, либо при изменении местоположения составного документа таким образом, что исходный файл становится недоступным, связь разрывается. Внедренные объекты всегда доступны, однако частое их использование приводит к непомерному увеличению файла составного документа.

Выполнив внедрение или связывание объекта OLE, можно легко активизировать из документа Access приложение, которому этот объект принадлежит. Для этого достаточно дважды щелкнуть левой кнопкой мыши по внедренному объекту, после чего объект может быть изменен. Когда же активизированное приложение будет закрыто,

в документе-контейнере отразится (а в случае внедрения — сохранится) внесенное изменение.

Связывание и внедрение объектов OLE сравниваются в табл. 15.1.

Таблица 15.1. *Сравнение характеристик связывания и внедрения*

Параметр	Связывание	Внедрение
Число файлов	Необходимо поддерживать не менее двух файлов: связанного объекта (такого, как лист Excel) и составного документа, использующего этот объект (такого, как Word). Можно установить связь так, что при обновлении исходного файла объекта автоматически обновляется файл составного документа	Поддерживать необходимо лишь один файл составного документа, содержащего внедренный объект
Создание объектов	Объект создается его копированием из исходного документа и вставкой связи в документ назначения	Объект создается либо копированием и вставкой, либо перетаскиванием с помощью мыши из исходного файла в файл составного документа
Обновление объектов	Один и тот же исходный документ могут использовать несколько составных документов. При изменении исходного документа обновление объекта во всех контейнерах, которые его используют, может происходить автоматически	Необходимо отразить изменения внедренного объекта в каждом составном документе
Формат, используемый в документе-контейнере	В зависимости от природы самого объекта и возможностей контейнера, можно связывать его в различных форматах. Например, можно связать рабочий лист Excel с документом Word как графический объект, форматированный текст, неформатированный текст, рисунок, растровую картинку	Все внедренные объекты появляются в составном документе как графические
Поддержка объектов	Необходимо отслеживать соответствие путей к файлу исходного объекта в составном документе	Все, что необходимо — это поддерживать документ, содержащий внедренный объект
Размер объекта	Поскольку исходный документ существует в отдельном файле, составной документ лишь весьма незначительно увеличивает свой размер при связывании его с файлом исходного документа	Внедренный объект должен содержать все необходимые данные внутри составного документа, поэтому размер последнего значительно увеличивается при внедрении

Таблица 15.1 (окончание)

Параметр	Связывание	Внедрение
Управление объектами	Используйте команду **Правка** | **Связи** (Edit | Links). Например, можно установить связь таким образом, чтобы можно было вручную или автоматически обновлять объект, временно блокировать связь, разрывать или восстанавливать ее	Большинство объектов можно преобразовать в составном документе так, чтобы они существовали как рисунки. Вернуть их в исходный формат внедренных объектов практически никогда нельзя

Вставка рисунков в формы и отчеты

Использование рисунков может улучшить внешний вид создаваемых в Access форм и отчетов. Например, в отчеты можно включать логотип фирмы, а в форме предоставлять возможность просмотра не только данных о сотрудниках фирмы, но и их фотографий. Для вставки рисунков в формы и отчеты используются специальные элементы управления — *свободная* и *присоединенная* рамки объектов (Unbound Object Frame, Bound Object Frame) и **Рисунок** (Image).

Способ вставки рисунка зависит от того, как предполагается его использовать.

- Любой объект OLE можно хранить в одном из полей таблицы Access. Например, в таблице `Сотрудники` (Employee) базы данных `Борей` хранятся фотографии сотрудников. В таком случае для отображения этих объектов в формах или отчетах следует использовать **Присоединенную рамку объекта** (Bound Object Frame).
- Объекты OLE могут храниться не в таблицах Access, а прямо в форме или отчете. Так можно, например, сохранить логотип фирмы. В этом случае для отображения объекта OLE следует использовать **Свободную рамку объекта** (Unbound Object Frame) или **Рисунок** (Image).
- Можно также для добавления рисунка в форму или отчет использовать свойство **Рисунок** (Picture) этой формы или отчета. Обычно так поступают, когда требуется включить фоновый рисунок (подложку), занимающий часть окна или все окно формы или отчета.

Чтобы добавить фоновый рисунок в форму:

1. Откройте форму в режиме Конструктора.
2. В окне свойств формы на вкладке **Макет** (Format) найдите свойство **Рисунок** (Picture). В поле ввода этого свойства введите путь и имя файла рисунка. Для этого можно нажать кнопку Построителя и выбрать в окне **Выбор рисунка** (Insert Picture) необходимый файл рисунка (рис. 15.1).

Замечание

Несколько стандартных файлов для фоновых рисунков находятся в папке MICROSOFT OFFICE\OFFICE\BITMAPS.

3. Установите значение свойства **Тип рисунка** (Picture Type): **Внедренный** (Embedded) или **Связанный** (Linked). Если рисунок внедренный, он сохраняется в файле

базы данных (с расширением mdb). Связанный рисунок хранится в отдельном файле на диске. О преимуществах и недостатках того и другого способа мы уже говорили ранее. Если один и тот же рисунок вставляется в несколько форм или отчетов, то внедрение его приведет к "распуханию" файла mdb, т. к. храниться будет столько копий рисунка, сколько раз вы его внедряли. Если вы решите сделать его связанным, необходимо заботиться о том, чтобы его случайно не удалили или не переместили.

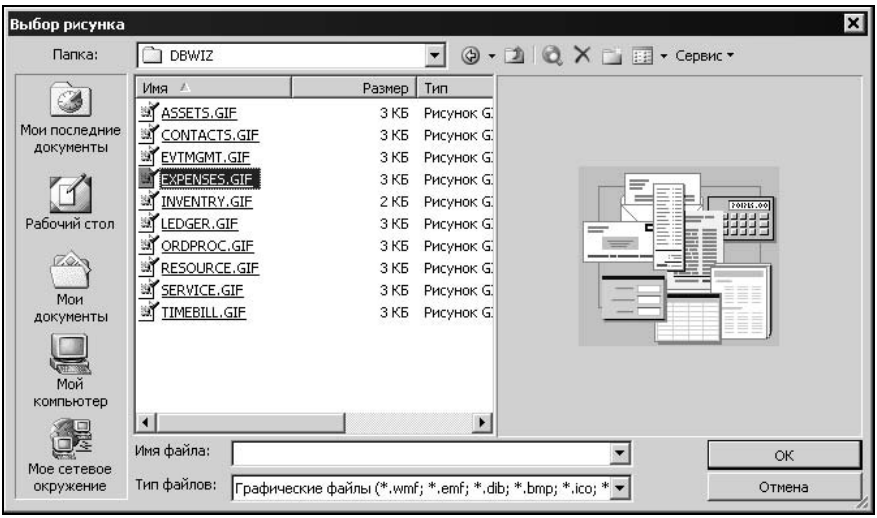


Рис. 15.1. Диалоговое окно **Выбор рисунка**

4. С помощью следующих свойств настройте отображение фонового рисунка.
   - Свойство **Масштабы рисунка** (Picture Size Mode) имеет три значения: **Фрагмент** (Clip), **Вписать в рамку** (Stretch), **По размеру рамки** (Zoom). Значение **Фрагмент** отображает рисунок в его исходном размере. Если размер рисунка превышает размер окна формы, то изображение рисунка обрезается. Значение **Вписать в рамку** изменяет размер рисунка так, чтобы он соответствовал окну формы. Использование данного значения может привести к искажению изображения рисунка. Значение **По размеру рамки** изменяет размер рисунка так, чтобы он соответствовал окну формы либо по ширине, либо по высоте. Использование данного значения не приведет ни к обрезанию рисунка, ни к искажению его пропорций.
   - Если вставляется рисунок, занимающий не всю форму, свойство **Выравнивание рисунка** (Picture Alignment) позволяет определить положение его в форме: **Сверху слева** (Top Left), **Сверху справа** (Top Right), **По центру** (Center), **Снизу слева** (Bottom Left), **Снизу справа** (Bottom Right), **По центру формы** (Form Center).
   - Свойство **Мозаичное заполнение** (Picture Tiling) позволяет выводить повторяющееся изображение рисунка.

Пример использования рисунка в качестве фона в форме приведен на рис. 15.2.

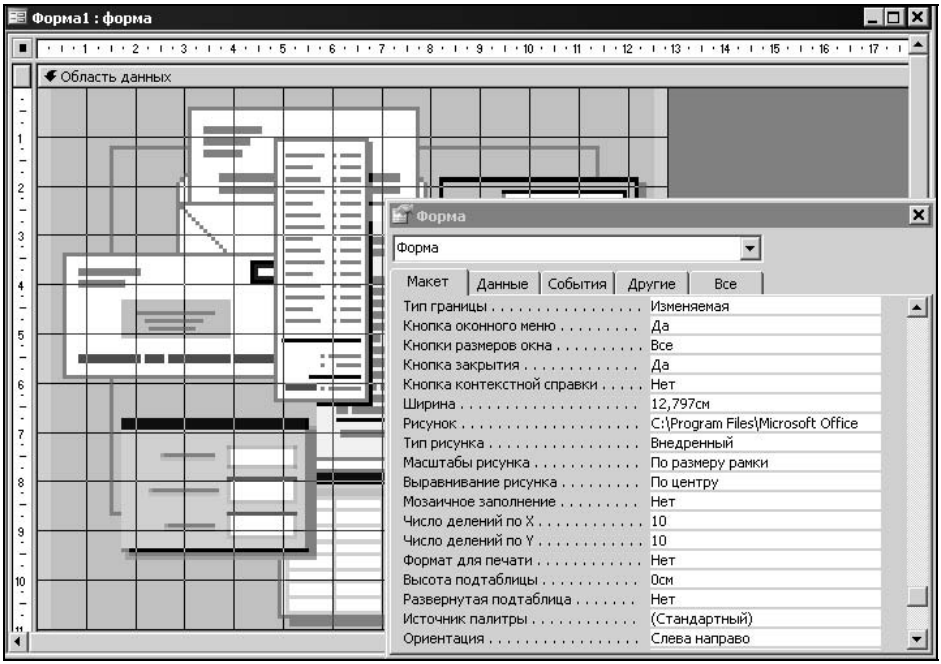


Рис. 15.2. Использование фонового рисунка в форме Microsoft Access

Аналогично вставляется фоновый рисунок в отчет. Пример использования рисунка в качестве фона в отчете приведен на рис. 15.3.

Рисунок может быть вставлен в форму и в виде элемента управления формы. Для этого можно использовать два типа элементов управления: **Рисунок** (Image) и **Свободная рамка объекта** (Unbound Object Frame). Если не требуется прямое изменение рисунка в форме или отчете, то лучше использовать **Рисунок** (Picture) (форма будет работать быстрее). Если же предполагается часто изменять рисунок, нужно вставить его с помощью свободной рамки объекта.

Чтобы вставить элемент управления **Рисунок** (Image), необходимо:

1. Открыть форму или отчет в режиме Конструктора.
2. Нажать кнопку **Рисунок** (Image) на панели элементов (кнопка Мастера элементов должна быть нажата, чтобы запустился Мастер рисунка).
3. Выбрать в форме или отчете место, в которое помещается рисунок, и щелкнуть на нем левой кнопкой мыши.
4. В раскрывшемся окне **Вставка рисунка** (Insert Picture) выбрать имя файла, содержащего добавляемый рисунок, и нажать кнопку **ОК**. Будет создан элемент управления, в котором помещается рисунок (рис. 15.4).
5. Установить значения свойств, которые влияют на то, как рисунок будет отображаться в форме или отчете. Эти свойства аналогичны соответствующим свойствам формы.

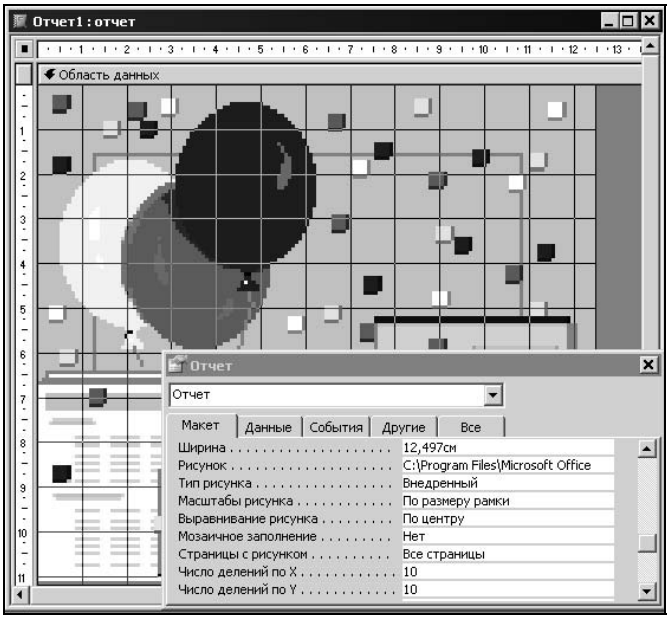


Рис. 15.3. Использование фонового рисунка в отчете Microsoft Access

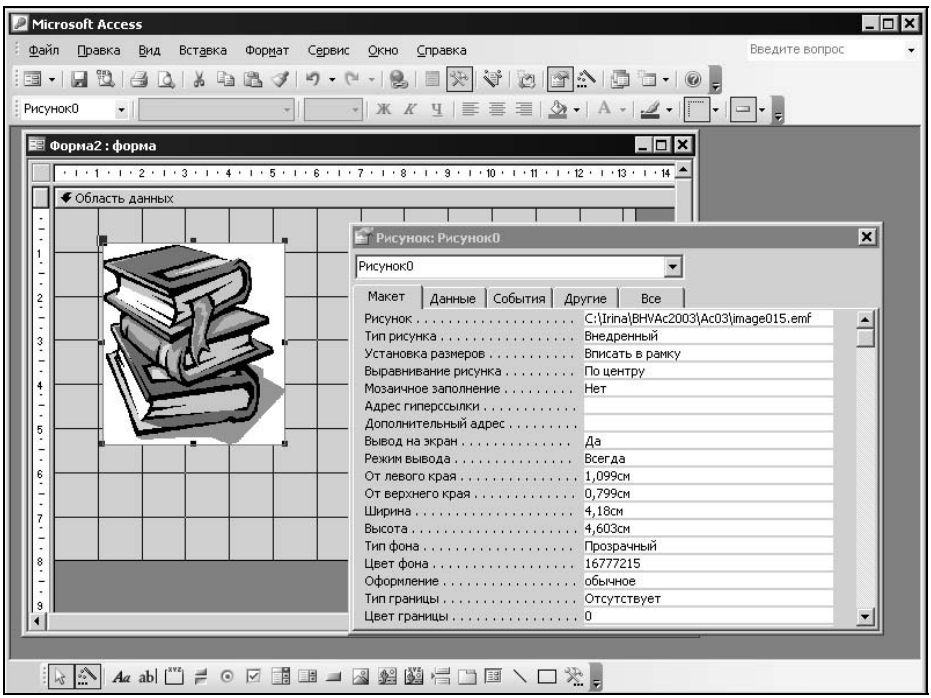


Рис. 15.4. Использование элемента **Рисунок** для вставки рисунка в форму

Чтобы вставить рисунок в форму или отчет с помощью свободной рамки объекта, необходимо:

1. Открыть форму в режиме Конструктора формы или отчет в режиме Конструктора отчета.
2. Нажать кнопку **Свободная рамка объекта** (Unbound Object Frame) на панели элементов.
3. Выбрать в форме или отчете место для вставляемого объекта и щелкнуть на нем левой кнопкой мыши. Появится диалоговое окно Мастера **Вставка объекта** (Insert Object). С помощью Мастера можно вставить уже готовый рисунок из файла или сначала создать его в соответствующем приложении, а затем вставить. Поэтому дальнейшие шаги могут быть разными.

Если вам еще предстоит создать рисунок, проделайте следующие операции.

1. В диалоговом окне вставки объекта (рис. 15.5) выберите переключатель **Создать новый** (Create New), а затем в списке **Тип объекта** (Object Type) приложение, с помощью которого будет создаваться рисунок.

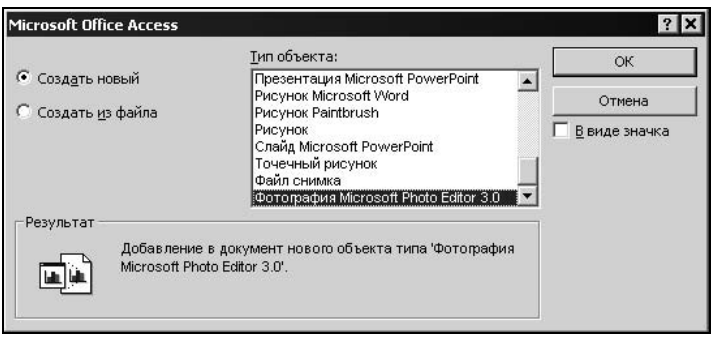


Рис. 15.5. Окно вставки объекта, используемое для вставки рисунка с помощью свободной рамки объекта

2. Объект может отображаться в виде значка — это удобно, когда он содержит сведения, которые не должны все время выводиться на экран. В таком случае нужно просто установить флажок **В виде значка** (Display as Icon).
3. Нажмите кнопку **OK**.
4. Создайте объект с помощью приложения, выбранного на шаге 1.
5. Закончив создание объекта, выберите команду **Закрыть** (Close) в меню **Файл** (File) приложения, чтобы вернуться в Microsoft Access. Microsoft Access создает элемент управления **Свободная рамка объекта** (Unbound Object Frame) и отображает в ней рисунок.

Если же файл с рисунком уже готов и находится в файле, нужно:

1. В диалоговом окне вставки объекта (рис. 15.6) выбрать переключатель **Создать из файла** (Create From File), а затем указать путь к файлу. Если точный путь вы не знаете, используйте кнопку **Обзор** (Browse).
2. Если нужно отобразить объект в виде значка, установить флажок **В виде значка** (Display As Icon).

3. Для установки связи с объектом используйте флажок **Связь** (Link). Если флажок сброшен (по умолчанию), объект будет внедрен.
4. Нажать кнопку **ОК**.

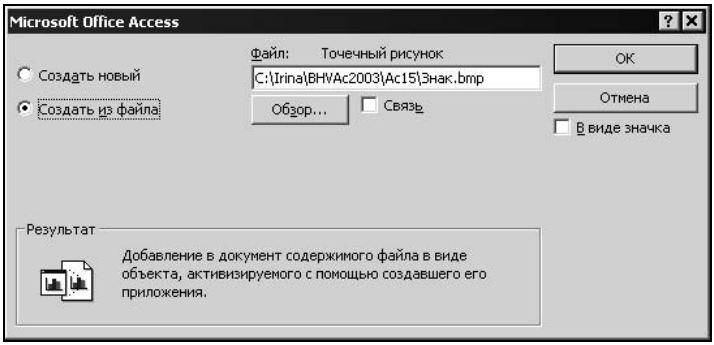


Рис. 15.6. Окно вставки объекта при вставке рисунка из файла с помощью свободной рамки объекта

Пример вставки рисунка, созданного с помощью Microsoft Paint, в форму с помощью свободной рамки объекта приведен на рис. 15.7.

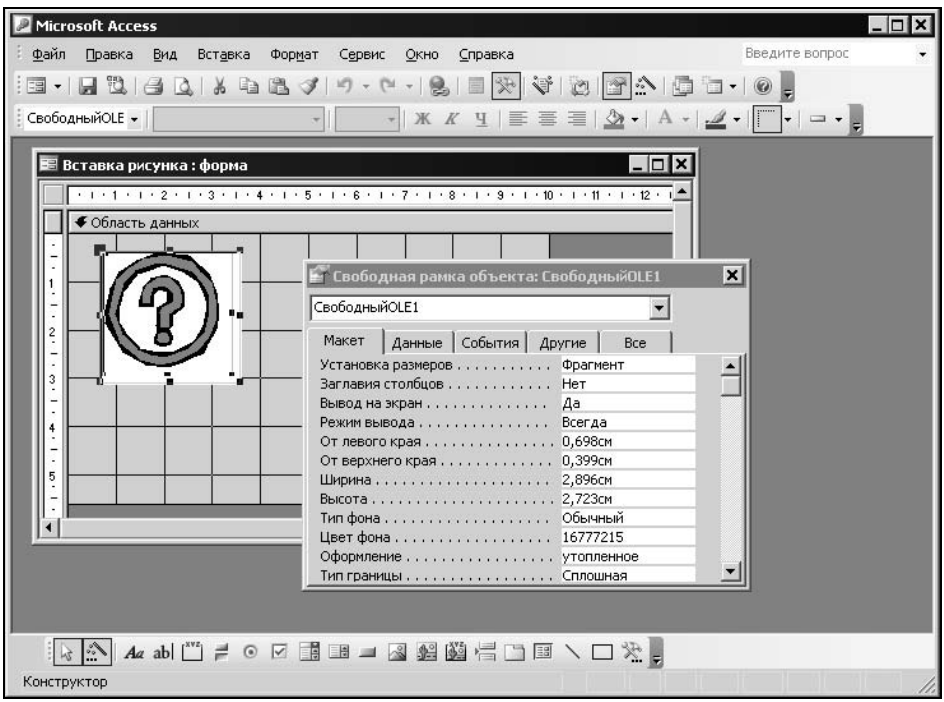


Рис. 15.7. Вставка рисунка Microsoft Paint в форму с помощью свободной рамки объекта

После вставки объекта можно изменять его размеры и пропорции с помощью свойства **Установка размеров** (Size Mode). В большинстве случаев изменения в свободный объект могут вноситься в режиме Конструктора формы или отчета. Для того чтобы иметь возможность изменять объект в режиме формы, необходимо задать для свойства **Доступ** (Enabled) свободной рамки объекта значение **Да** (Yes), а для свойства **Блокировка** (Locked) — значение **Нет** (No). Чтобы отредактировать объект, дважды щелкните на нем мышью. Для этого только необходимо, чтобы на вашем компьютере было установлено приложение, в котором может редактироваться файл данного типа. Пример редактирования внедренного точечного рисунка представлен на рис. 15.8.

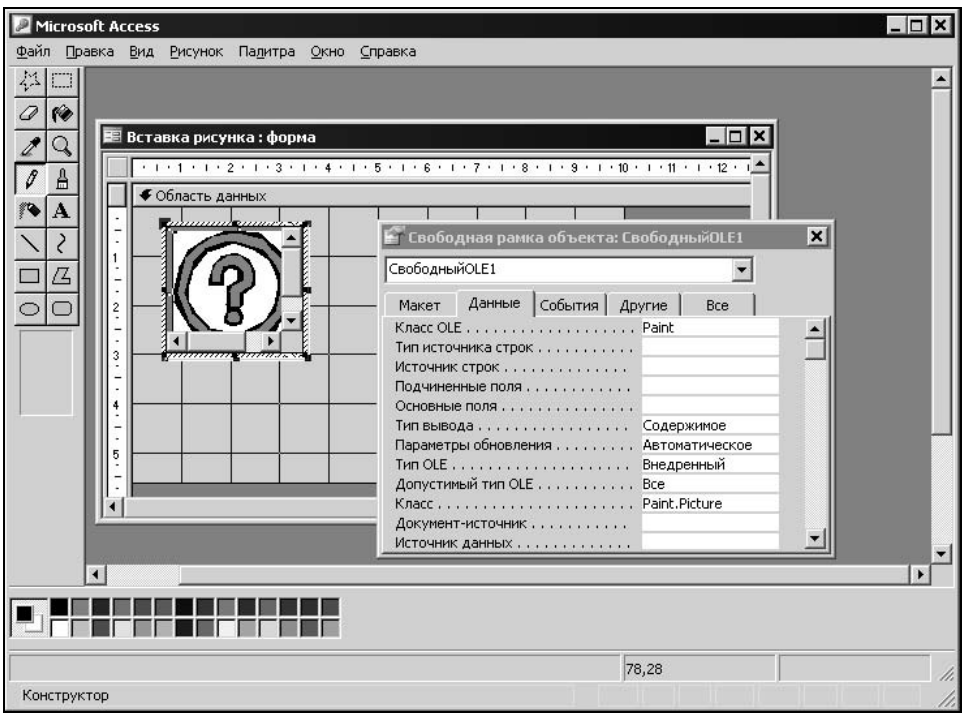


Рис. 15.8. Пример редактирования точечного рисунка с помощью Microsoft Paint

Для того чтобы выйти из режима редактирования, достаточно щелкнуть мышью за пределами рисунка.

В отличие от других элементов управления в форме, которые связаны с записями в таблице-источнике, рисунок в свободной рамке объекта не меняется при перемещении по записям в форме — он связан с самой формой, а не с данными, которые отображаются в форме. Рисунки, которые хранятся в записях таблицы, отображаются в форме с помощью присоединенной рамки объекта.

Для хранения рисунков и других объектов OLE в таблицах Access используется специальный тип поля: **Объект OLE** (OLE Object). Элемент управления **Присоединенная рамка объекта** (Bound Object Frame) создается одним из стандартных способов:

- перемещением с помощью мыши нужного поля из списка полей таблицы-источника. В этом случае будет сразу создан элемент управления, связанный с полем в таблице;
- с помощью кнопки **Присоединенная рамка объекта** (Bound Object Frame) на панели элементов. В данном случае будет создан элемент управления, для которого нужно сразу установить свойство **Данные** (RecordSource) — имя поля, содержащего объект OLE (рис. 15.9).

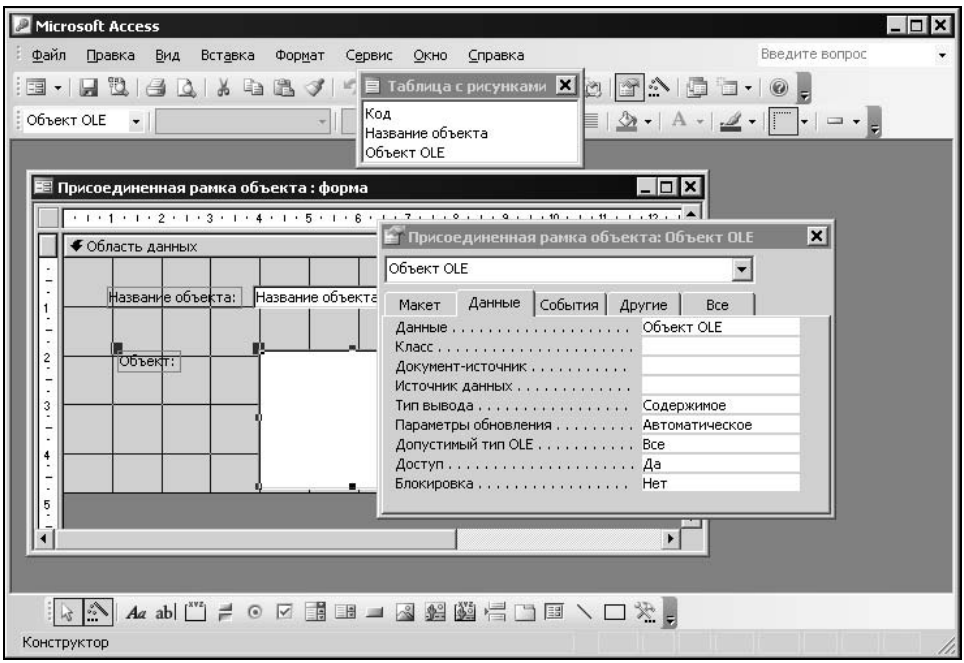


Рис. 15.9. Создание присоединенной рамки объекта в режиме Формы

Вставить рисунок в поле таблицы можно как в таблице, открытой в режиме Таблицы, так и в форме. Для этого нужно:

1. Открыть таблицу в режиме Таблицы или форму в режиме Формы.
2. Перейти на запись, в которую требуется вставить рисунок, и выбрать поле объекта OLE.
3. Выполнить команду **Объект** (Object) из меню **Вставка** (Insert).
4. В диалоговом окне вставки объекта (см. рис. 15.5, 15.6) выбрать переключатель **Создать новый** (Create New), а затем в списке **Тип объекта** (Object Type) выбрать тип приложения, в котором будет создаваться объект. Если файл с рисунком уже готов, то нужно выбрать переключатель **Создать из файла** (Create From File), а затем указать путь к файлу.
5. Для того чтобы отобразить объект в виде значка, установить флажок **В виде значка** (Display As Icon).

6. Нажать кнопку **ОК**.
7. Если вы создаете новый рисунок, то нужно создать его с помощью приложения, выбранного на шаге 4.
8. Закончив создание объекта, выбрать команду **Закрыть** (Close) в меню **Файл** (File) приложения для возвращения в Microsoft Access.

Рисунок будет вставлен в запись в таблице. При этом в режиме Формы рисунок или иной объект автоматически будет отображаться на экране. В режиме Таблицы будет отображаться только тип объекта, например **Растровый рисунок**.

Вставка графических файлов происходит либо напрямую, либо при помощи отдельных графических фильтров. Эти фильтры устанавливаются при инсталляции Microsoft Access. Для вставки файлов формата Enhanced Metafile (EMF), точечных рисунков Windows (BMP, RLE, DIB), метафайлов Windows (WMF) и значков (ICO) графический фильтр не требуется. Однако такие фильтры необходимы для вставки файлов графических форматов, перечисленных в табл. 15.2.

Таблица 15.2. *Форматы файлов, требующие отдельного графического фильтра*

AutoCAD Format 2-D (DXF)	Computer Graphics Metafile (CGM)
CorelDRAW (CDR)	Encapsulated PostScript (EPS)
Graphics Interchange Format (GIF)	JPEG File Interchange Format (JPG)
Kodak Photo CD (PCD)	Micrografx Designer/Draw (DRW)
PC Paintbrush (PCX)	Portable Network Graphics (PNG)
Tagged Image File Format (TIFF)	Targa (TGA)
WordPerfect Graphics (WPG)	HG Graphics Language (HGL, PLT)
Macintosh PICT (PCT)	

Вставка рабочего листа Excel в форму или отчет

С помощью таких элементов управления, как свободная и присоединенная рамки объектов, в формах и отчетах Access можно отображать не только различные графические объекты, но и листы Microsoft Excel.

Так же как и рисунок, лист Microsoft Excel можно внедрить в форму или отчет или связать его с ними. При этом внедренный лист автоматически сохраняется в файле базы данных и всегда является доступным. Если обновить такой объект в форме или отчете, он будет изменен только в базе данных. Если лист Microsoft Excel связан, его можно просматривать и вносить изменения в форме или отчете. При этом изменения сохраняются в исходном файле объекта, а не в файле базы данных. Файл объекта можно обновлять независимо от базы данных. Последние изменения будут выведены на экран при следующем открытии формы или отчета. Связывание объектов удобно при работе с очень большими файлами, которые не желательно включать в файл базы данных, а также с файлами, используемыми в нескольких формах и отчетах. Если связанный файл объекта перемещен, необходимо повторно установить с ним связь.

Рассмотрим возможность вставки листа Microsoft Excel в форму и отчет Microsoft Access с помощью свободной рамки объекта. Чтобы вставить лист Microsoft Excel в форму Microsoft Access, необходимо:

1. Открыть форму или отчет в режиме Конструктора.
2. Добавить в нее элемент управления **Свободная рамка объекта** (Unbound Object Frame). При этом для вставки нового листа в окне **Вставка объекта** (Insert Object) выбрать переключатель **Создать новый** (Create New), а в списке **Тип объекта** (Object Type) выбрать **Лист Microsoft Excel** (Microsoft Excel Worksheet) (рис. 15.10). Если файл XLS уже создан, выберите переключатель **Создать из файла** (Create From File) и введите путь и имя нужного файла.

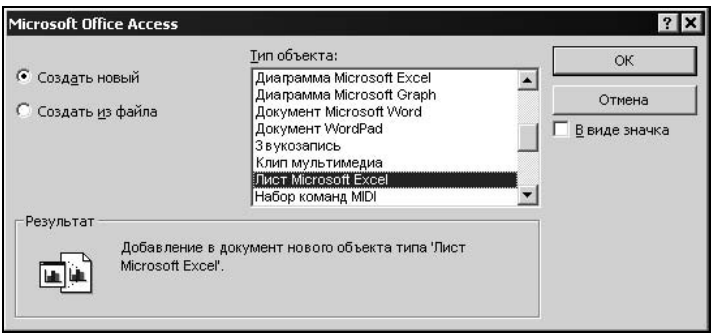


Рис. 15.10. Вставка нового листа Excel в форму Access с помощью свободной рамки объекта

3. Когда в форме появится элемент управления, в котором отображается пустая таблица Excel, введите в нее данные (это можно сделать прямо в режиме Конструктора) (рис. 15.11). Если вы вставляли готовый файл, то сразу будет отображена таблица Excel (рис. 15.12).
4. Чтобы сделать доступным изменение данных на листе в режиме Формы, установите значение свойства **Доступ** (Enabled) для вставленного элемента управления равным **Да** (Yes), а свойство **Блокировка** (Locked) — **Нет** (No).

Увидеть данные, расположенные на листе Microsoft Excel, можно, открыв форму в режиме Формы. Чтобы иметь возможность изменять или добавлять данные на лист, нужно дважды щелкнуть кнопкой мыши на листе Microsoft Excel в форме (рис. 15.13).

При работе с отчетами Microsoft Access следует выполнять аналогичную пошаговую процедуру. Увидеть данные, расположенные на листе Microsoft Excel, можно, открыв отчет в режиме Просмотра.

Рассмотрим теперь возможность вставки листов Microsoft Excel с помощью присоединенной рамки объекта. В этом случае листы Excel будут храниться в записях таблицы Access (более разумным представляется хранить не сами таблицы, а ссылки на них, т. е. не внедрять листы Excel, а связывать их с таблицей Access). Сначала в таблице должно быть создано поле типа **Объект OLE** (OLE Object), которое будет содержать листы Excel. Затем в форме, отображающей записи этой таблицы, должен быть создан элемент **Присоединенная рамка объекта** (Bound Object Frame), связанный

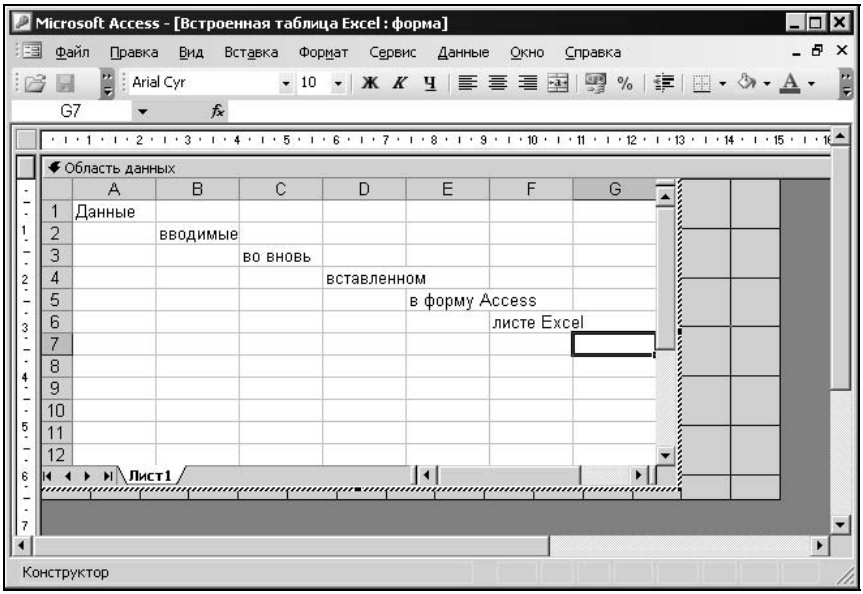
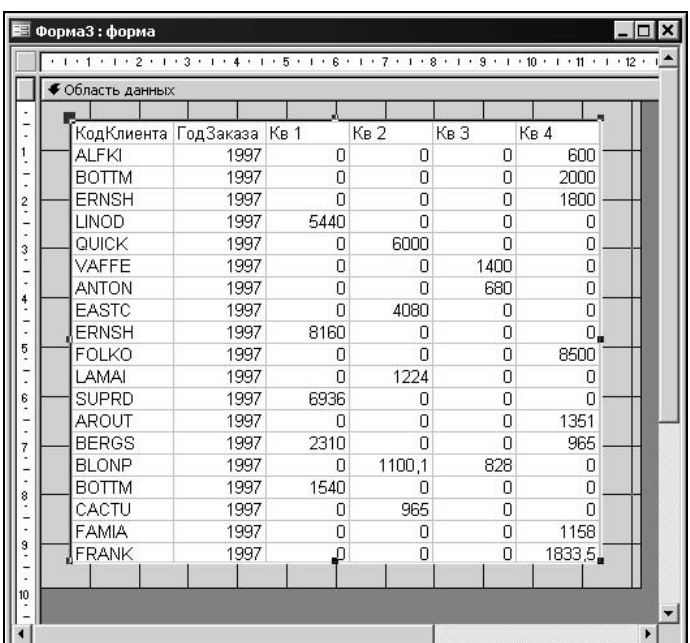


Рис. 15.11. Ввод данных на лист Excel в режиме Конструктора форм Access



КодКлиента	ГодЗаказа	Кв 1	Кв 2	Кв 3	Кв 4
ALFKI	1997	0	0	0	600
BOTTM	1997	0	0	0	2000
ERNSH	1997	0	0	0	1800
LINOD	1997	5440	0	0	0
QUICK	1997	0	6000	0	0
VAFFE	1997	0	0	1400	0
ANTON	1997	0	0	680	0
EASTC	1997	0	4080	0	0
ERNSH	1997	8160	0	0	0
FOLKO	1997	0	0	0	8500
LAMAI	1997	0	1224	0	0
SUPRD	1997	6936	0	0	0
AROUT	1997	0	0	0	1351
BERGS	1997	2310	0	0	965
BLONP	1997	0	1100,1	828	0
BOTTM	1997	1540	0	0	0
CACTU	1997	0	965	0	0
FAMIA	1997	0	0	0	1158
FRANK	1997	0	0	0	1833,5

Рис. 15.12. Использование свободной рамки объекта для вставки существующего листа Excel в форму Access

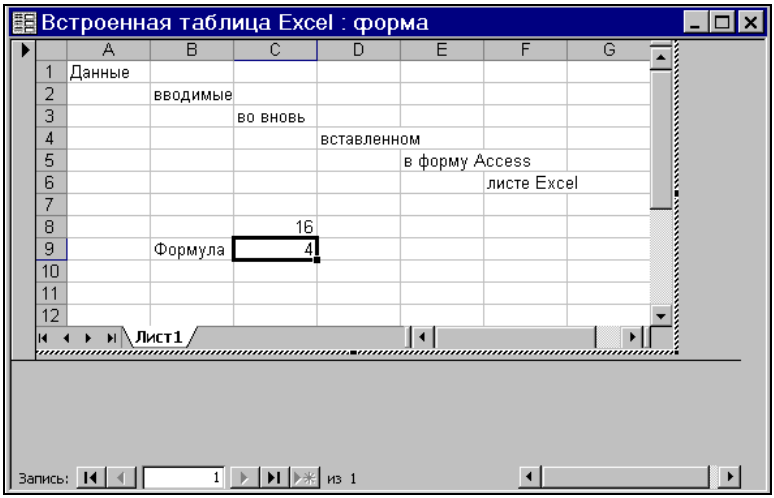


Рис. 15.13. Ввод данных на лист Excel в режиме Формы

с этим полем. Вставка листа Microsoft Excel в поле таблицы или формы может быть выполнена с помощью команды меню **Вставка | Объект** (Insert | Object), как это уже описывалось ранее. На рис. 15.14 и 15.15 представлены примеры вставки листа Microsoft Excel в форму и отчет Access.

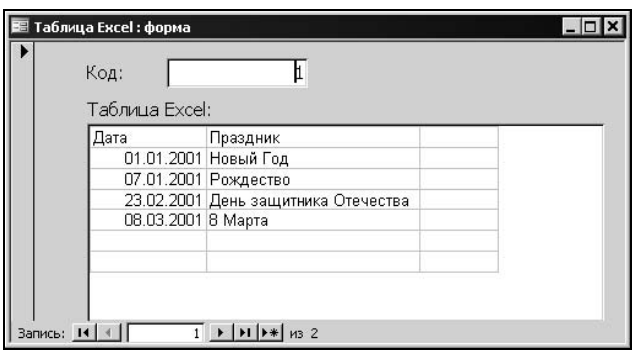


Рис. 15.14. Использование присоединенной рамки объекта для вставки листа Excel в форму Access

Вставка документов Word в таблицы, формы и отчеты Access

Использование документов Microsoft Word в таблицах, формах и отчетах Microsoft Access осуществляется аналогично использованию листов Microsoft Excel или рисунков. Документ Microsoft Word рассматривается как объект OLE, который может быть помещен в таблицу, с помощью свободной или присоединенной рамки объекта, в форму или отчет Microsoft Access.

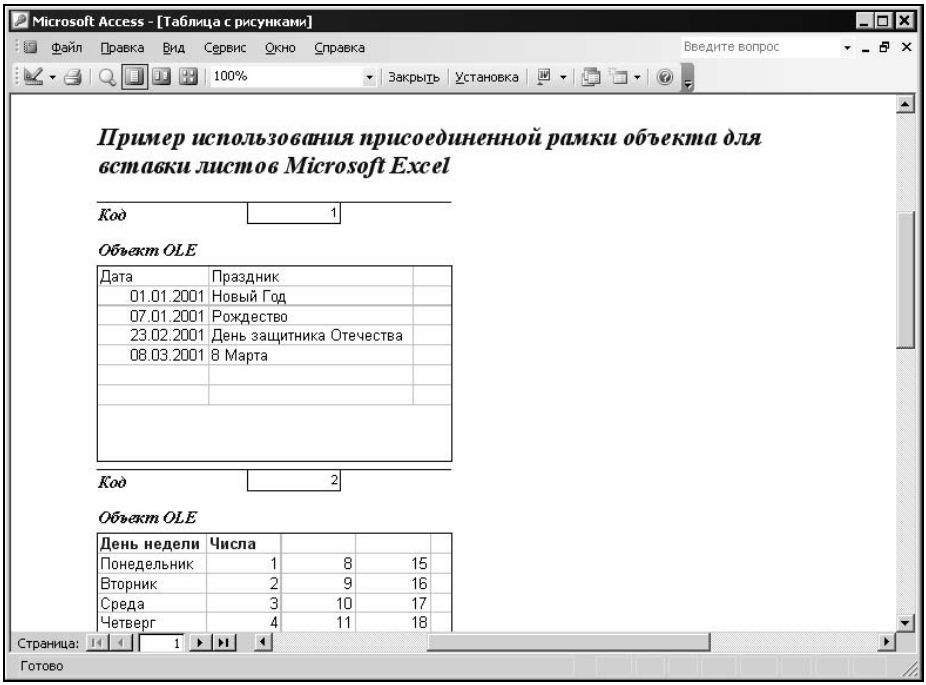


Рис. 15.15. Использование присоединенной рамки объекта для вставки листов Excel в отчет Access

Рассмотрим пример, в котором Microsoft Access используется для учета документов Microsoft Word. Для реализации учета документов Word с возможностью их просмотра и редактирования необходимо создать таблицу, в которую должны быть включены поля для порядкового номера документа, названия документа, шифра документа, самого документа и примечаний (табл. 15.3).

Таблица 15.3. *Поля таблицы и их типы*

Имя поля	Тип поля	Назначение
Номер	**Счетчик** (AutoNumber)	Порядковый номер документа
Наименование	**Текстовый** (Text)	Название документа
Код	**Текстовый** (Text)	Шифр документа
Текст	**Поле объекта OLE** (OLE Object)	Документ
Примечание	**Поле Memo** (Memo)	Примечания

Для создания таблицы необходимо:

1. Создать в режиме Конструктора таблицу, состоящую из указанных ранее полей (рис. 15.16).
2. Сохранить таблицу под именем `Учет документов`.

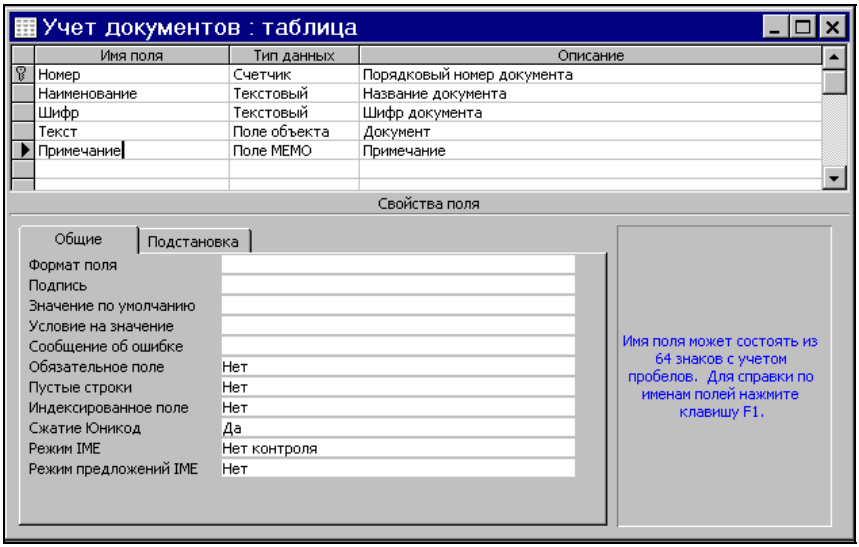


Рис. 15.16. Таблица `Учет документов` в окне Конструктора

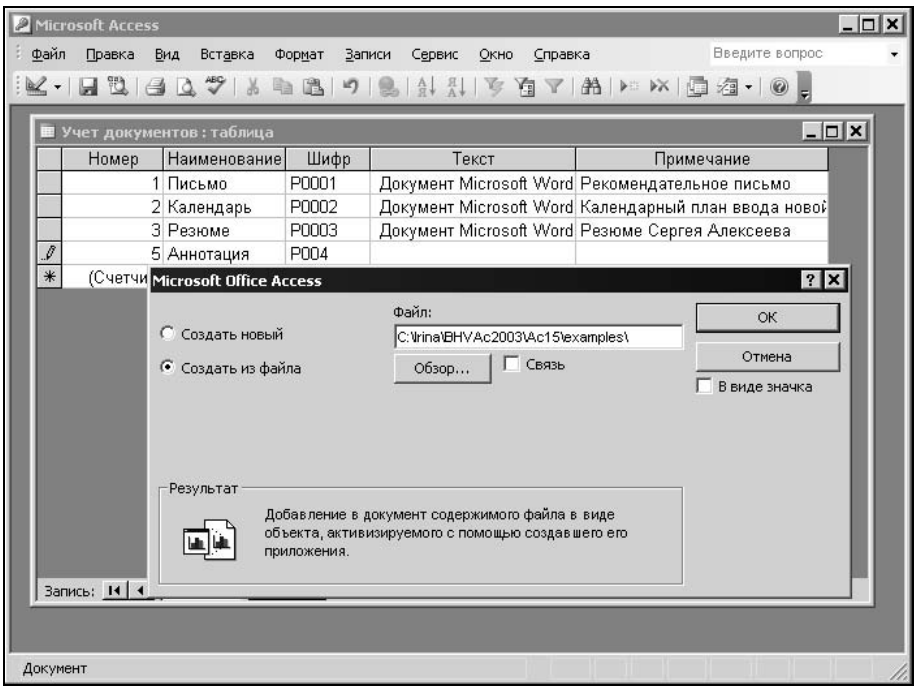


Рис. 15.17. Окно вставки объекта при вставке документа Word из файла в таблицу в виде объекта OLE

3. Открыть таблицу `Учет документов` в режиме Таблицы.
4. Заполнить поля таблицы.
5. Для вставки документа Word в качестве объекта OLE требуется установить указатель на нужное поле, выполнить команду **Объект** (Object) из меню **Вставка** (Insert), в окне вставки объекта (см. рис. 15.17) выбрать переключатель **Создать из файла** (Create From File) и в поле **Файл** (File) указать имя и путь к файлу. Так как в нашем примере учитываются уже созданные документы, которые могут изменяться не только из Microsoft Access, необходимо установить связь с этими документами, для чего установите флажок **Связь** (Link) в окне вставки объекта. Флажок **В виде значка** (Dysplay As Icon) в данном случае не устанавливается, т. к. нужно иметь возможность быстрого просмотра документов (рис. 15.17).

В результате создана таблица `Учет документов`, приведенная на рис. 15.18.

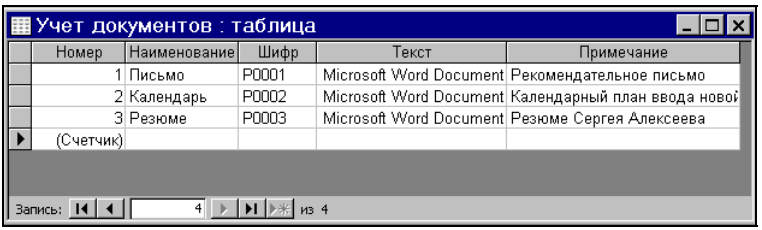


Рис. 15.18. Представление данных в таблице `Учет документов` в режиме Таблицы

После создания таблицы, в которой содержатся данные о документах, необходимо создать форму для удобного просмотра, редактирования и добавления новой информации о документах. Чтобы создать форму, необходимо:

1. В окне базы данных раскрыть список форм и нажать на кнопку **Создать** (New).
2. В окне **Новая форма** (New Form) в поле со списком **Выберите таблицу или запрос, на основе которого будет построена форма** (Choose the table or query where the object's data comes from) выбрать таблицу `Учет документов`, выбрать **Конструктор** (Design View) и нажать кнопку **OK**.
3. Нажать на кнопку **Список полей** (Field List) и перенести поля из списка в форму.
4. Отредактировать элементы в форме, изменив при необходимости их размеры и положение в форме. Пример расположения элементов приведен на рис. 15.19.
5. Закрыть форму, сохранив ее под именем `Учет документов`.

Для просмотра содержимого таблицы необходимо открыть форму в режиме Формы (рис. 15.20). Чтобы получить возможность редактирования документа, требуется дважды щелкнуть левой кнопкой мыши на документе в форме. При этом откроется приложение Microsoft Word с выбранным документом.

Для создания отчета необходимо:

1. В окне базы данных раскрыть список отчетов и нажать кнопку **Создать** (New).
2. В окне **Новый отчет** (New Report) в поле со списком **Выберите таблицу или запрос, на основе которого будет построен отчет** (Choose the table or query where the

object's data comes from) выбрать таблицу `Учет документов`, затем **Конструктор** (Design View) и нажать кнопку **OK**.

3. Нажать на кнопку **Список полей** (Field List) и перенести поля из списка в секцию **Область данных** (Detail) отчета.

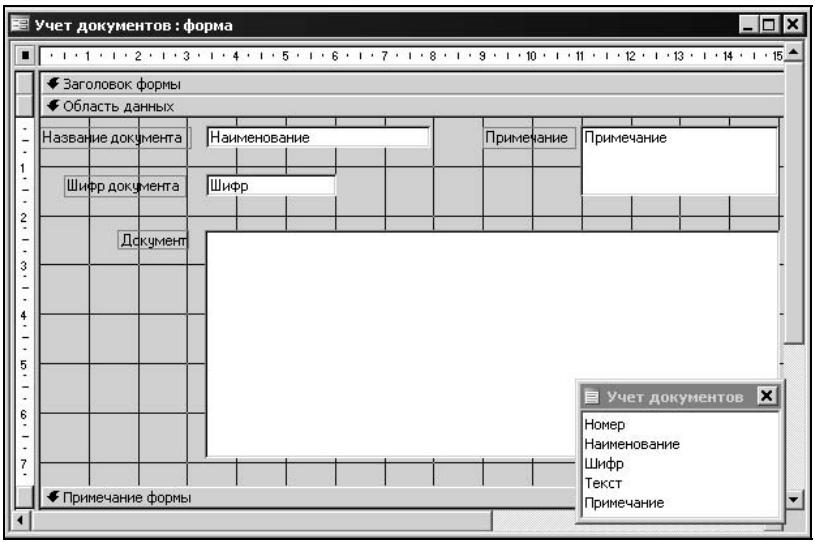


Рис. 15.19. Расположение элементов в форме `Учет документов`

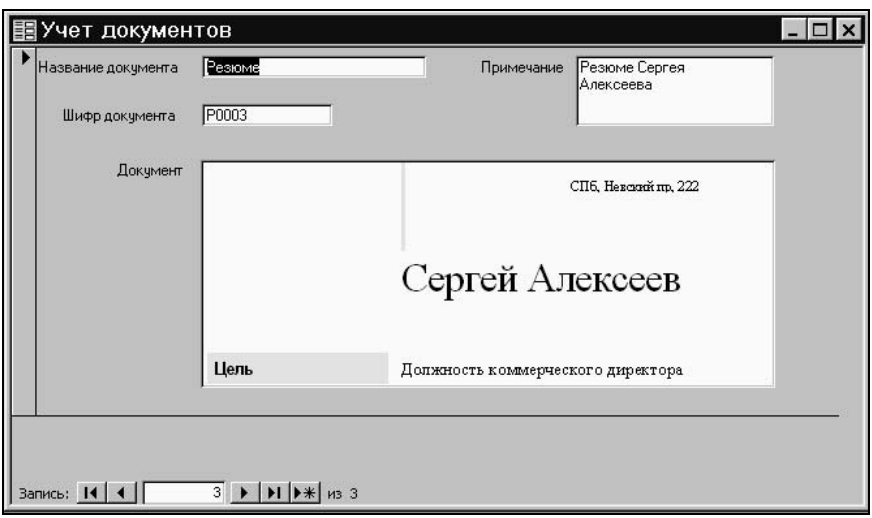


Рис. 15.20. Форма `Учет документов` в режиме Формы

4. Отредактировать элементы в отчете, изменив при необходимости их размеры и положение в форме. Пример расположения элементов приведен на рис. 15.21.
5. Закрыть отчет, сохранив его под именем `Учет документов`.

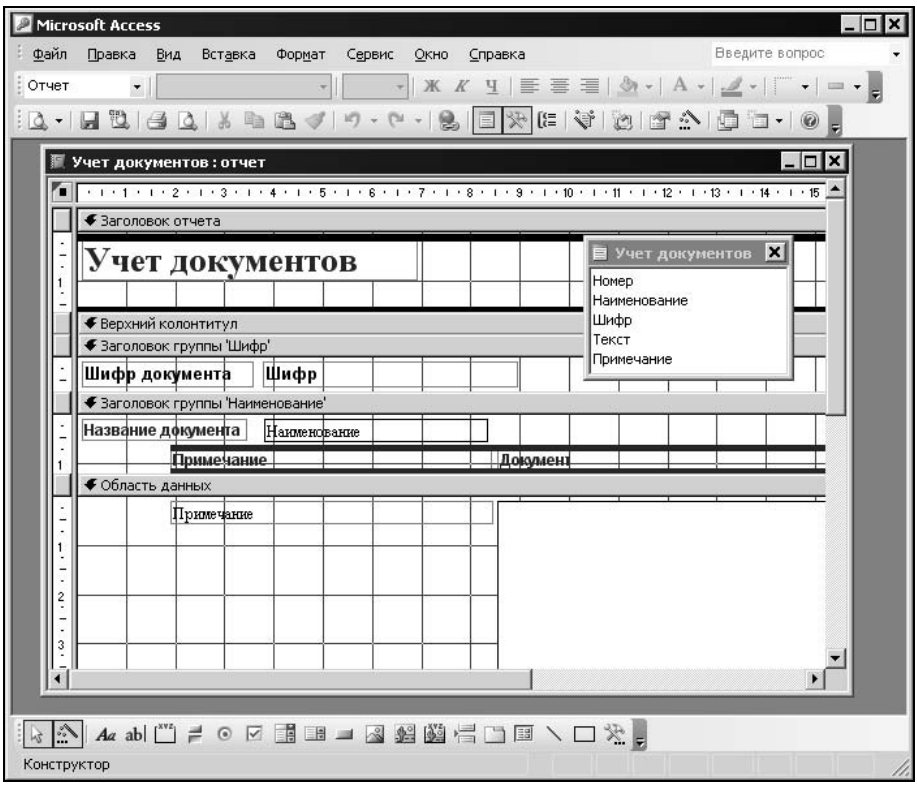


Рис. 15.21. Расположение элементов отчета `Учет документов` в режиме Конструктора

Чтобы просмотреть содержимое отчета, необходимо открыть его в режиме Просмотра (рис. 15.22).

При изменении содержимого документов Word эти изменения будут отображаться в форме и отчете при их следующем открытии.

При желании можно вставлять документы Word в виде значка. Для этого в окне вставки объекта (см. рис. 15.17) необходимо устанавливать флажок **В виде значка** (Display As Icon). В этом случае в форме и отчете не будут отображаться данные документов, а будут указываться только значки, изображающие документы. Для обращения к нужному документу из формы достаточно дважды щелкнуть левой кнопкой мыши на значке соответствующего документа. При этом откроется приложение Microsoft Word с надлежащим документом в окне. Примеры отображения документов в виде значка в форме и отчете приведены на рис. 15.23 и 15.24.

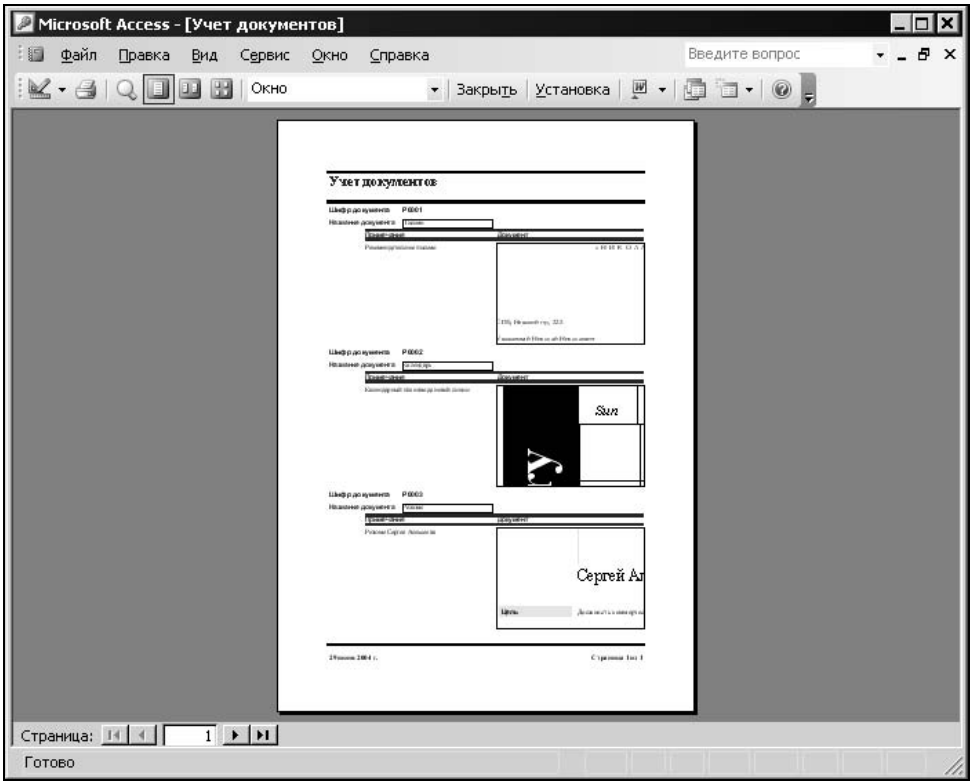


Рис. 15.22. Отчет `Учет документов` в режиме Просмотра

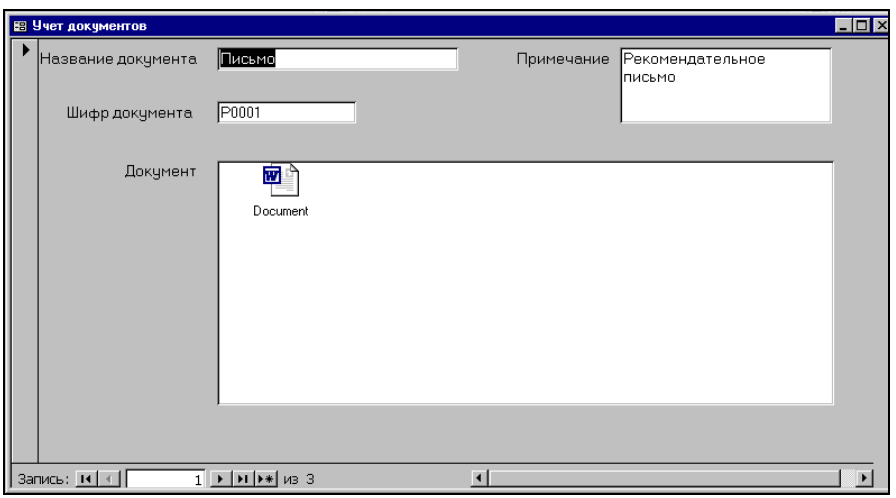


Рис. 15.23. Представление документа Word в форме в виде значка

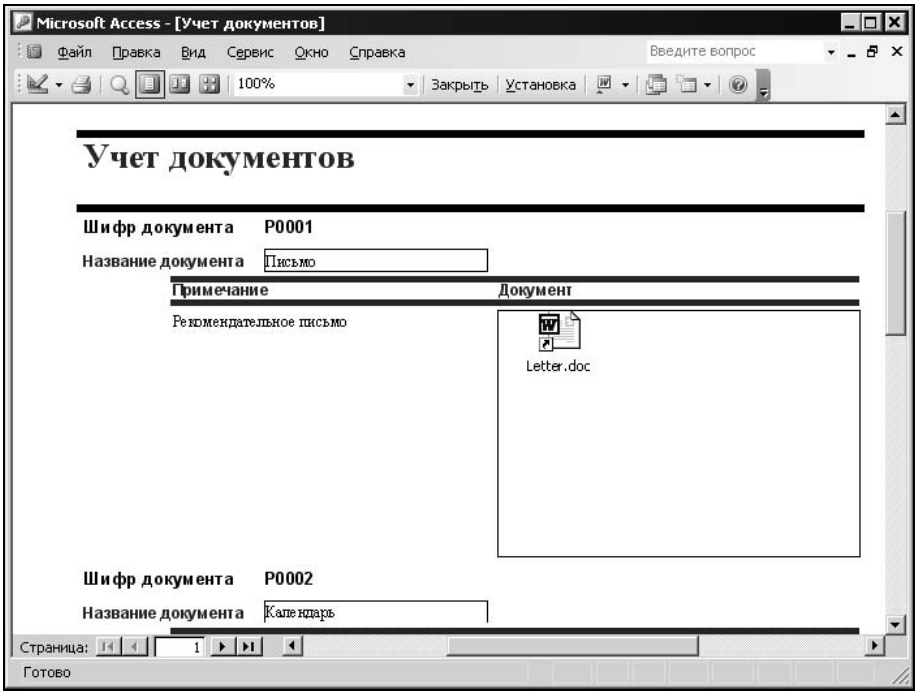


Рис. 15.24. Представление документа Word в отчете в виде значка

Автоматизация

Автоматизация (ранее известная как *OLE-автоматизация* — OLE Automation) — это одно из наиболее важных средств технологии ActiveX, позволяющее программно управлять объектами из других приложений. И это основное средство, с помощью которого можно интегрировать функциональные возможности различных приложений.

Приложения, поддерживающие автоматизацию, делятся на две категории: *клиенты автоматизации* и *серверы автоматизации*. Причем некоторые приложения могут быть только клиентами либо только серверами автоматизации, но есть и такие (и к ним относится Microsoft Access), которые могут выступать и в том и в другом качестве.

Клиенты и серверы автоматизации

При интеграции двух приложений одно предоставляет свои объекты для использования, а другое использует объекты первого приложения. Приложение, объекты которого доступны для других приложений, называется *сервером автоматизации* (иногда его еще называют компонентом). Приложение, которое использует объекты другого приложения, называется *клиентом* (или *контроллером*) *автоматизации*.

Объекты, которые доступны для других приложений, называют *объектами автоматизации.*

Через объекты автоматизации приложение-сервер открывает доступ другим программам к тем своим функциям, которые могут быть им полезны. Например, текстовый редактор может открывать другим программам доступ к проверке орфографии, служба связи — доступ к созданию и отправке сообщений. Это позволяет разработчикам ускорить процесс разработки своих приложений, благодаря использованию готовых функций сервера.

Объекты приложения-сервера образуют библиотеку объектов, которая может быть подключена к приложению-клиенту путем установки ссылки в проекте VBA *(см. разд. "Установка ссылок на объектные библиотеки" гл. 13)*. Приложение-клиент использует объекты приложения-сервера путем доступа к их свойствам и методам. Например, Microsoft Excel имеет модель объектов, которая включает такие объекты, как `Workbook`, `Worksheet`, `Sell` и др. К этим объектам можно обращаться из Microsoft Access, а также из других приложений, поддерживающих автоматизацию.

Одним из преимуществ автоматизации является возможность запуска сервера автоматизации без создания его видимого экземпляра. Если серверу не указано активизировать окно, он невидим, и в списке задач его имя не содержится.

Взаимодействие компонентов Microsoft Office 2003

Для программного управления объектом автоматизации из любого приложения Microsoft Office необходимо:

1. Установить ссылку на библиотеку объектов приложения-сервера автоматизации.
2. Создать объект нужного класса.
3. Осуществить все необходимые действия над объектом, обращаясь к его свойствам и методам.
4. Закрыть объект.

Замечание

Можно использовать объекты сервера автоматизации и не устанавливая ссылку на конкретную библиотеку объектов, однако в этом случае программа будет работать медленнее. Это связано с тем, что после установки ссылки на библиотеку, при объявлении объектной переменной, можно указать ее конкретный тип, если же ссылка отсутствует, то для этой переменной придется указать обобщенный тип — `Object`. *(О различии таких объявлений см. разд. "Ссылки на объекты" гл. 13.)*

Далее мы рассмотрим подробнее, как выполнить эти действия из Microsoft Access.

Использование Microsoft Access в качестве клиента автоматизации

Чтобы создать объект для использования в операциях автоматизации, нужно создать экземпляр его класса и присвоить ссылку на него объектной переменной. Однако создать можно не любой объект из объектной модели приложения-сервера, а только глобальные объекты *(см. разд. "Использование объектов и семейств" гл. 13)*. Таким глобальным объектом для всех объектных моделей приложений семейства Microsoft

Office является объект `Application`, который находится на вершине иерархии объектов. Однако есть и другие глобальные объекты, например в Microsoft Excel глобальным объектом является `Workbook`. Это позволяет создавать данный объект непосредственно, не используя ссылку на объект `Application`. Напомним, что узнать, какие свойства моделей объектов являются глобальными, можно, используя окно просмотра объектов в редакторе VBA. Выберите в раскрывающемся списке **Project/Library** нужную библиотеку, а затем выберите элемент <**globals**> в списке **Classes**.

Существует несколько способов создания экземпляра класса объекта.

- С помощью ключевого слова `New`, которое описывает переменную типа `Object` и одновременно устанавливает ссылку на новый экземпляр класса объекта. Синтаксис оператора VBA, создающего объект автоматизации, в данном случае аналогичен обычному оператору создания нового объекта, только при описании типа объекта необходимо указать имя приложения-сервера. Например,

  ```
  Dim objExl As New Excel.Application
  ```

  При использовании ключевого слова `New` для создания нового экземпляра класса `Application` запускается соответствующее приложение (например, Word или Excel). Если это приложение уже запущено, то, чтобы не запускать второй экземпляр, лучше использовать для создания нового экземпляра класса функцию `GetObject` (см. далее в том же списке).

Замечание

Чтобы использовать ключевое слово `New` для создания экземпляра класса объекта автоматизации, необходимо предварительно добавить ссылку на библиотеку объектов приложения-сервера в диалоговом окне редактора VBA **References** (Ссылки).

- С помощью функции `CreateObject`, которая создает ссылку на новый экземпляр класса автоматизированного объекта. Значение, возвращаемое этой функцией, должно быть присвоено объектной переменной с помощью оператора `Set`. Например, следующим оператором открывается приложение Word, и ссылка на него присваивается объектной переменной `docObj`. При этом, даже если Word уже был открыт, будет запущен еще один его экземпляр:

  ```
  Set docObj = CreateObject("Word.Application")
  ```

  Объектная переменная в данном случае может быть объявлена как `Object`, и тогда ссылку на библиотеку объектов Microsoft Word устанавливать не надо. А может быть объявлена так:

  ```
  Dim docObj As Word.Application
  ```

  В этом случае должна быть обязательно установлена ссылка на библиотеку объектов Microsoft Word 11.0 Object Library.

Замечание

Если на компьютере установлено несколько версий Microsoft Office, то можно указать номер версии приложения, которое будет использоваться при автоматизации, например `Set docObj = CreateObject("Word.Application.11")`.

- С помощью функции GetObject. Этот способ используется для того, чтобы получить ссылку на уже существующий экземпляр класса, т. е. для доступа к существующим документам, хранящимся в файлах, или для доступа к объекту Application уже запущенного приложения-сервера. Функция GetObject имеет следующий синтаксис:

  ```
  Set <объектнаяПеременная> = GetObject([<путь>][,<класс>])
  ```

  Необязательный аргумент <путь> определяет полный путь к существующему файлу и используется тогда, когда нужно обратиться к объекту, находящемуся в уже существующем файле.

  Аргумент <класс>, как и в других случаях, определяет тип открываемого объекта автоматизации и имеет структуру <имяПриложения>.<типОбъекта>.

  Обязательно должен быть указан хотя бы один из аргументов функции. Если первый аргумент опущен, то второй должен определять объект Application соответствующего приложения, например

  ```
  Set objXL = GetObject (,"Excel.Application")
  ```

  Аргумент <класс> может быть опущен, если в реестре Windows существует связка расширения файла с объектом приложения. Например, в случае Microsoft Excel файлы XLS по умолчанию имеют объектный тип Excel.Workbook. Это означает, что если в первом аргументе указать имя файла с расширением xls и не указывать аргумент <класс>, то будет создана ссылка на объект Workbook.

После того как создан объект автоматизации Application или другой глобальный объект, через него можно получить доступ ко всем остальным семействам и объектам соответствующей модели объектов стандартным способом *(см. разд. "Использование объектов и семейств" гл. 13)* и работать с ними из программы VBA Access точно так же, как и с собственными объектами — считывать и устанавливать значения свойств и вызывать методы.

Совет

Для того чтобы легче было создавать код для работы с объектами Word или Excel, вспомните о таком замечательном средстве этих приложений, как запись макросов. С помощью записи макросов можно выполнить все необходимые действия, используя интерфейс пользователя, затем просмотреть сгенерированный код VBA и встроить этот код в свою процедуру VBA, только слегка подправив его.

После того как все необходимые операции с объектами приложения-сервера выполнены, эти объекты следует закрыть, чтобы освободить занимаемую ими память. Большинство объектов поддерживают для этого методы Close или Quit. Кроме того, следует освободить объектную переменную, присвоив ей значение Nothing.

Операции с объектами Microsoft Excel

Следующий пример показывает, как из Access можно управлять рабочим листом Excel.

Создание рабочего листа Товары.xls

Для этого нам потребуется файл Товары.xls, содержащий рабочую книгу Excel с единственным рабочим листом Товары, который представляет собой список товаров

из демонстрационной базы данных Борей. Вы можете использовать готовый файл, находящийся на сопровождающем книгу компакт-диске, либо создать его сами, экспортировав в Excel таблицу Товары с помощью команды меню **Сервис | Связи с Office | Анализ в Microsoft Office Excel** (Tools | Office Links | Analyze It with Microsoft Office Excel) *(см. разд. "Экспорт данных в другие форматы" гл. 3).*

Для того чтобы подготовить файл Товары.xls к дальнейшим экспериментам, выполните следующие действия:

1. Откройте файл Товары.xls, запустив Microsoft Excel.
2. Выделите ячейки с A4 по D12 в рабочем листе. Выберите команду меню **Вставка | Имя | Присвоить** (Insert | Name | Define). В диалоговом окне **Присвоение имени** (Define Name) введите в текстовое поле **Имя** (Names in workbook) имя диапазона: WorkRange (рис. 15.25) и щелкните по кнопке **OK**. Тем самым создается именованный диапазон, который будет использоваться в последующих примерах.

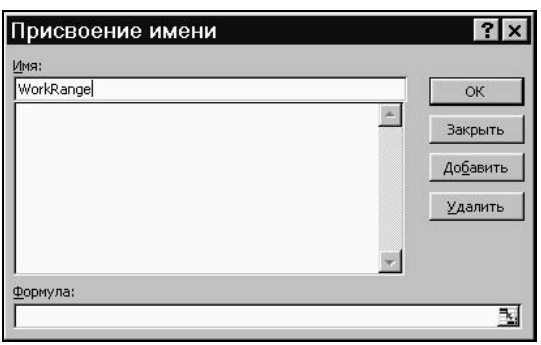


Рис. 15.25. Создание именованного диапазона в таблице Товары.xls

3. Выберите в меню **Сервис** (Tools) команду **Надстройки** (Adds-Ins) и снимите все флажки в списке **Список надстроек** (Add-Ins Available) диалогового окна **Надстройки** (Add-Ins). Удаление надстроек уменьшает время, требующееся для запуска приложения Excel. Чтобы закрыть диалоговое окно, нажмите кнопку **OK**.
4. Сохраните изменения, выполнив команду **Файл | Сохранить** (File | Save) или **Файл | Сохранить как** (File | Save As).
5. Закройте Microsoft Excel.

Иерархия объектов VBA приложения Microsoft Excel

Для того чтобы программно работать с объектами Excel, нужно иметь представление об объектной модели Microsoft Excel. Мы не будем здесь подробно описывать эту модель, поскольку она достаточно сложна, представим только ее основные объекты.

- Объект Application является экземпляром собственно Microsoft Excel. Применяя к этому объекту различные методы, можно воспользоваться практически любыми командами меню Excel. Объект Application обладает свойствами ActiveWorkbook и ActiveSheet, которые указывают на текущие объекты Workbook (рабочая книга) и Worksheet (рабочий лист). Можно указать Excel.Application в качестве зна-

чения аргумента <класс> функций CreateObject и GetObject, а также в операторе Dim objИмя As New <класс>.

- Основной объект Excel — Workbook (рабочая книга). С помощью этого объекта осуществляется доступ к свойствам и структуре рабочей книги Excel. Объекты Workbook — это файлы, которые состоят из нескольких объектов, создаваемых Microsoft Excel: Worksheet (рабочий лист) и Chart (диаграмма). Объекты Worksheet и Chart содержатся в семействах Worksheets и Charts.
- Объекты Worksheet являются подчиненными по отношению к объекту Workbook и обеспечивают доступ к свойствам и структуре рабочих листов книги Excel. Основным объектом взаимодействия приложений Access 2003 и Excel 2003 является объект Worksheet. Имеется возможность перенесения информации из строк и столбцов объекта Recordset приложения Microsoft Access в ячейки объекта Worksheet приложения Microsoft Excel и наоборот. Если в качестве значения аргумента <класс> функции GetObject либо в операторе Dim objИмя As New <класс> указать Excel.Sheet, то по умолчанию откроется первый объект семейства Worksheets — первый рабочий лист (ActiveSheet) объекта Workbook.
- Объекты Sheet аналогичны объектам Worksheet, но они входят в семейство Sheets, которое включает в себя как объекты Worksheet, так и объекты Chart, представляющие листы диаграмм в рабочей книге.
- Объект Range — диапазон ячеек рабочего листа Excel. Это могут быть отдельная ячейка, строка, столбец, прямоугольная область рабочего листа или несвязанный (произвольный набор ячеек) или трехмерный диапазон ячеек, включающий ячейки на нескольких рабочих листах. Можно получить либо установить значения диапазона ячеек, определив объект Range. Таких объектов, как ячейка, строка, столбец, просто не существует. Именно объект Range со своими свойствами и методами обеспечивает работу как с отдельной ячейкой, так и с их диапазоном. Для ссылки на необходимую группу ячеек можно использовать имя диапазона, состоящего из этих ячеек. Если именованные диапазоны отсутствуют на рабочем листе, можно использовать метод Cells объекта WorkSheet, чтобы задать координаты ячейки или группы ячеек. Далее приводятся два способа изменения значения ячейки A1:

```
ActiveSheet.Range ("A1").Value = 7
ActiveSheet.Cells (1,1).Value = 7
```

Microsoft Excel предоставляет также многие другие объекты для использования их в качестве объектов приложения сервера, но описанные ранее типы являются наиболее часто используемыми в технологии автоматизации с помощью Access VBA. С остальными объектами, их свойствами и методами вы можете ознакомиться, прочитав книгу В. Долженкова "Microsoft Excel 2003" издательства "БХВ-Петербург", выпущенную в 2004 году.

Открытие существующего рабочего листа Excel и работа с ним

Прежде чем работать с автоматизированными объектами Microsoft Excel, установим ссылку на библиотеку объектов Microsoft Excel. Для этого:

1. Запустите Access и, если необходимо, откройте базу данных Борей. Перейдите в среду редактора VBA, открыв любой модуль.

2. Выберите в окне редактора VBA команду меню **Сервис | Ссылки** (Tools | References), чтобы открыть диалоговое окно **References** (Ссылки).
3. Установите флажок у элемента **Microsoft Excel 11.0 Object Library** (рис. 15.26), затем нажмите кнопку **OK**, чтобы закрыть окно **Ссылки**.

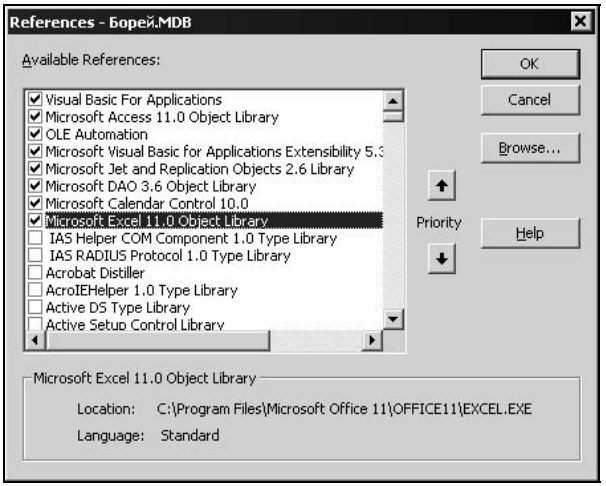


Рис. 15.26. Добавление ссылки в редакторе VBA на объектную библиотеку Microsoft Office Excel 2003

Команды автоматизации удобно изучать при помощи окна отладки **Immediate**. Поэтому выведите это окно на экран, если оно не отображается. Для чего достаточно нажать соответствующую кнопку на панели инструментов или комбинацию клавиш <Ctrl>+<G>.

Чтобы программно открыть лист рабочей книги `Товары`:

1. Закройте Excel, если он запущен.
2. Создайте новый модуль, выполнив команду меню **Insert | Module** (Вставка | Модуль).
3. Добавьте в раздел описаний следующие описания переменных:

```
Private xlaProd As Excel.Application
Private xlwProd As Excel.Workbook
Private xlsProd As Excel.Worksheet
```

4. Введите в окне отладки следующий оператор (рис. 15.27):

```
Set xlwProd = GetObject(CurDir & "\Товары.xls","Excel.Sheet")
```

Затем при нажатии клавиши <Enter> приложение Microsoft Office Excel запускается в режиме `/automation`. Функция `CurDir` возвращает полное имя текущей папки (по умолчанию это папка Мои документы). Если файл Товары.xls был сохранен где-нибудь в другом месте, измените в предыдущем операторе путь к этому файлу. В зависимости от скорости функционирования компьютера, запуск

Excel может продолжаться достаточно долго. Загрузка приложения Excel завершится, когда в строке состояния окна отладки исчезнет надпись **Выполнение** (Running) и появится надпись **Готово** (Ready). В результате будет создан экземпляр класса `Application` Microsoft Excel и переменной `xlwProd` будет присвоена ссылка на объект `Workbook`. Обратите внимание, что функция `GetObject` открывает скрытый экземпляр приложения Excel, значок Excel не появляется на панели задач и интерактивно обратиться к рабочей книге Excel нельзя.

Замечание

В данном операторе аргумент `"Excel.Sheet"` является необязательным. Если его не указать, то тип создаваемого объекта будет определен автоматически по расширению файла, указанному в первом аргументе.

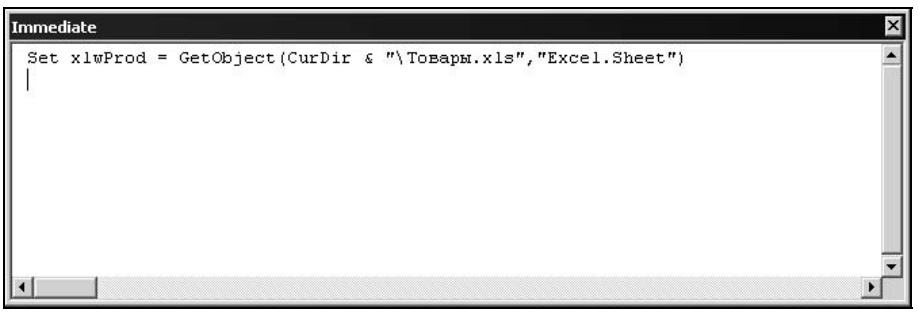


Рис. 15.27. Команды для автоматического запуска приложения Microsoft Excel

5. Чтобы убедиться, что рабочая книга открыта и мы можем к ней обратиться, введите в окне отладки следующую команду:

```
?xlwProd.Name
```

Свойство `Name` созданного объекта `Workbook` содержит имя файла Excel: Товары.xls (рис. 15.28).

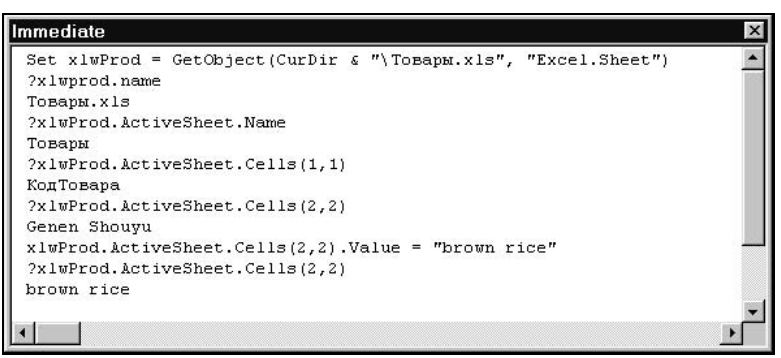


Рис. 15.28. Команды, позволяющие читать и устанавливать значения отдельных ячеек в листе `Товары`

6. Файл Товары.xls содержит только один объект `Worksheet`, поэтому рабочий лист `Товары` является активным рабочим листом — объектом `ActiveSheet`. Чтобы убедиться в этом, введите в окно отладки команду:

   ```
   ?xlwProd. ActiveSheet.Name
   ```

   Свойство `Name` этого объекта содержит имя рабочего листа: `Товары`.

7. Теперь попробуйте обратиться к первой ячейке рабочего листа. Введите `?xlwProd.ActiveSheet.Cells(1,1)`. После короткой паузы появится ожидаемый результат — строка `Код товара`. Это заголовок первого столбца таблицы.

8. Метод `Cells` позволяет обратиться к любой ячейке рабочего листа. Введите `?xlwProd.ActiveSheet.Cells(R,C)`, где `R` — номер строки, а `C` — номер столбца заданной ячейки, т. е. ее координаты (рис. 15.28).

9. Можно изменить содержимое ячейки, если ввести выражение вида: `xlwProd.ActiveSheet.Cells(2,2).Value = "brown rice"`. Подобно тому как многие объекты Access при введении имени объекта возвращают его значение, метод `Cells` не требует явного указания свойства `Value` по умолчанию. Чтобы убедиться в том, что содержимое ячейки изменилось, введите `?xlwProd.ActiveSheet.Cells(3,2)`, не дописывая выражения `.Value`.

Для установления значения ячейки можно также использовать свойство `Formula`, преимущество использования которого состоит в возможности его применения с целью введения формул с использованием "родного" синтаксиса Microsoft Excel, т. е. в виде ссылок на конкретные ячейки, например "`=A2+C6`".

Использование именованных диапазонов ячеек

Если в рабочем листе Excel создан именованный диапазон ячеек, то можно получить значения ячеек, содержащихся в этом диапазоне, если сослаться на свойство `Range` объекта `Worksheet`. Сначала посмотрим, какие именованные диапазоны присутствуют в открытом нами объекте. Введите в окно отладки команду (рис. 15.29)

```
?xlwProd.Names(1).Name
```

Семейство `Names` представляет все имена, определенные в рабочей книге. В данном случае первый элемент этого семейства содержит имя диапазона: `WorkRange`.

Можно посмотреть не только имя, но и что собой представляет этот диапазон. Введите команду

```
?xlwProd.Names(1).Value
```

Результат будет: `=Товары!$A$4:$D$12`, т. е. прямоугольная область `A4—D12` на рабочем листе `Товары`.

На рис. 15.29 приведены выражения для управления объектом `Range`.

Пусть переменная `xlsProd` ссылается на рабочий лист `Товары`. Для этого введите команду:

```
Set xlsProd = xlwProd.ActiveSheet
```

Для указания конкретной ячейки внутри именованного объекта `Range` можно использовать следующий оператор:

```
?xlsProd.Range("WorkRange").Cells(1,1)
```

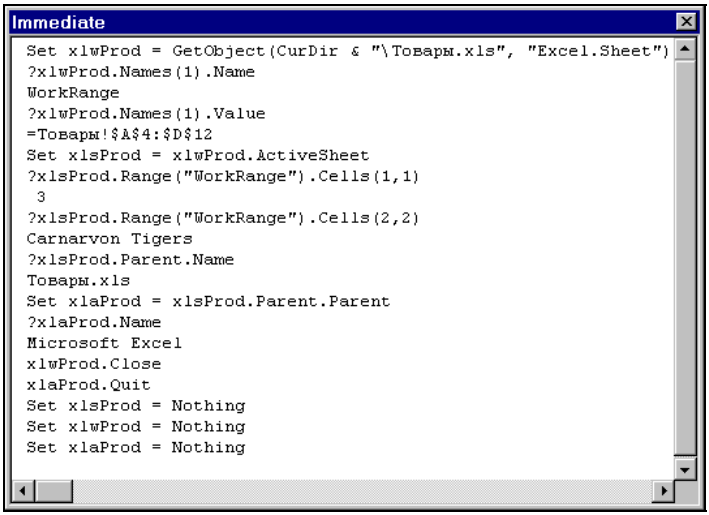


Рис. 15.29. Примеры использования именованного диапазона

Здесь используется свойство `Range` объекта `Worksheet` для доступа к именованному диапазону, а затем свойство `Cells` объекта `Range` — для указания конкретной ячейки в диапазоне. Первая цифра указывает строку, а вторая — столбец.

Для того чтобы обратиться к объекту, который находится на уровень выше в иерархии объектов модели, можно воспользоваться свойством `Parent`. На рис. 15.29 представлено, как обратиться к рабочей книге Excel, содержащей текущий рабочий лист, и как установить объектную переменную `xlaProd`, которая должна ссылаться на объект `Application` Microsoft Excel:

```
Set xlwРабочаяКнига = xlsРабочийЛист.Parent.
```

Закрытие объектов *Workbook* и *Application*

Объект Microsoft `Worksheet` закрыть нельзя. Для закрытия объекта Excel `Workbook` может быть использован метод `Close`, а для выхода из приложения — метод `Quit`. Следующие операторы закрывают объект `Workbook` и затем осуществляют выход из приложения сервера автоматизации, освобождая системные ресурсы:

```
xlwProd.Close
xlaProd.Quit
Set xlsProd = Nothing
Set xlwProd = Nothing
Set xlaProd = Nothing
```

Если программно были внесены изменения в рабочем листе, то при закрытии объекта `Workbook` будет выдан вопрос о необходимости сохранения изменений. Если вы не хотите, чтобы пользователь получил такой вопрос, введите аргумент `False` для метода `Close`. Чтобы гарантировать освобождение всех ресурсов, необходимо освободить все использованные объектные переменные.

Замечание

После присвоения переменной, указывающей на объект `Application`, значения `Nothing` соответствующее приложение не закрывается, хотя память, занятая переменной, освобождается. Поэтому необходимо закрывать приложение с помощью метода `Quit` перед освобождением соответствующей объектной переменной.

Создание рабочего листа Excel с помощью кода автоматизации

Те же действия, что происходят при нажатии кнопки **Анализ в Microsoft Office Excel**, можно осуществить при помощи кода автоматизации VBA. Преимуществом такого способа является возможность форматировать созданный объект специально под нужды конкретного приложения. Рассмотрим функцию `CreateCustomSheet`, создающую новый объект `Worksheet` и заполняющую его данными из таблицы `Товары` базы данных Access (листинг 15.1).

Листинг 15.1. Функция создания рабочего листа Excel

```
Function CreateCustomSheet() As Integer
'Создание рабочего листа MS Excel из таблицы "Товары"
'Описание локальных переменных
'(Объектные переменные описаны на уровне модуля)
    Dim dbБорей As Database    'Текущая база данных
    Dim rstProd As Recordset   'Объект Recordset
    Dim intRow As Integer      'Счетчик строк
    Dim intCol As Integer      'Счетчик столбцов

'Открытие таблицы в текущей базе данных
    Set dbБорей = CurrentDb()
    Set rstProd = dbБорей.OpenRecordset("Товары", dbOpenTable)
    DoCmd.Hourglass True
'Создание нового объекта Excel Workbook
    Set xlwProd = CreateObject("Excel.Sheet")
'Создание объекта Application для применения метода Quit
    Set xlaProd = xlwProd.Parent

    intRow = 1
    intCol = 1
    rstProd.MoveFirst     'Переход к первой записи
    Do Until rstProd.EOF
        'Цикл с шагом в одну запись
        For intCol = 1 To rstProd. Fields.Count

            'Цикл с шагом в одно поле
            If (Not IsNull(rstProd(intCol - 1))) Then
               xlwProd.ActiveSheet.Cells(intRow, intCol).Value = _
                              CStr(rstProd(intCol - 1))
            End If
        Next intCol
```

```
        rstProd.MoveNext
        intRow = intRow + 1
    Loop
    For intCol = 1 To xlwProd.ActiveSheet.Columns.Count
        'Форматирование каждого столбца рабочего листа
        xlwProd.ActiveSheet.Columns(intCol).Font.Size = 8
        xlwProd.ActiveSheet.Columns(intCol).AutoFit
        If intCol = 10 Then
        'Выравнивание по левому краю последнего столбца
            xlwProd.ActiveSheet.Columns(intCol).HorizontalAlignment = _
                xlLeft
        End If
    Next intCol
    DoCmd.Hourglass False
    xlwProd.SaveAs (CurDir & "\Товары_2.xls")
    xlaProd.Quit
End Function
```

Тип данных, возвращаемых выражением `rstProd(intCol-1)`, следует специально изменить с `Variant` на `String` при помощи функции `CStr`, иначе Microsoft Excel вместо нужной величины отобразит в соответствующем столбце `#Н/Д` (#N/A#). Если объект `Recordset` содержит поля, типы которых отличны от `Text`, то для определения типа данных в столбце используйте соответствующую функцию `CType`.

Константа `xlLeft`, присвоенная в качестве значения свойству `HorizontalAlignment` восьмого столбца, представляет собой встроенную константу Excel, определяемую в тот момент, когда устанавливается ссылка на объектную библиотеку Microsoft Excel 11.0 Object Library. Выделение элемента **Constants** (Константы) в списке **Модули/Классы** окна просмотра объектов при подключенной библиотеке Excel отображает константы `xlConst`. На рис. 15.30 приведено числовое значение константы `xlLeft`, которая является одной из констант для задания значения свойства `HorizontalAlignment`.

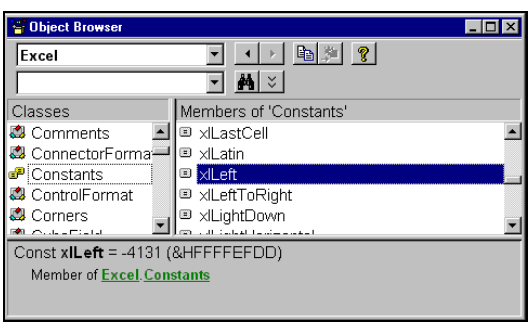
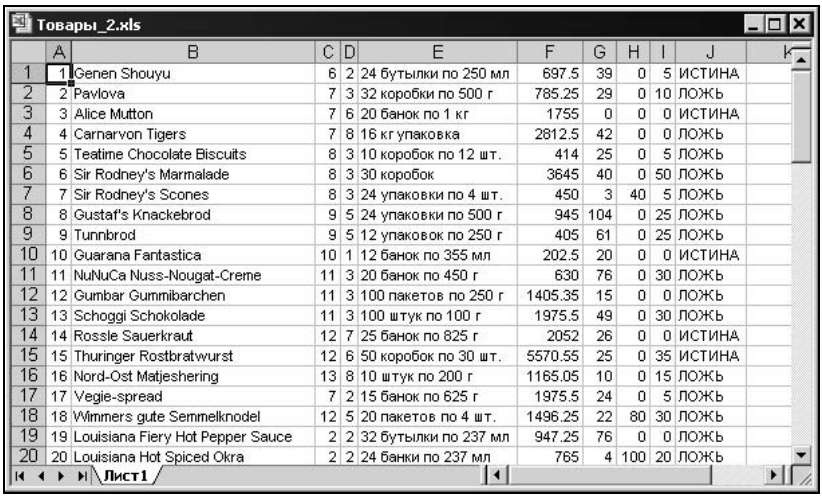


Рис. 15.30. Значения встроенных констант `xlConst` в окне **Object Browser** Access

Ввод оператора `? CreateCustomSheet` в окне отладки запускает функцию, которую мы рассматривали ранее. На рис. 15.31 приведена рабочая книга Товары_2.xls с ра-

бочим листом, созданным при помощи функции `CreateCustomSheet` и открытым в Microsoft Excel.



Товары_2.xls

	A	B	C	D	E	F	G	H	I	J
1	1	Genen Shouyu	6	2	24 бутылки по 250 мл	697.5	39	0	5	ИСТИНА
2	2	Pavlova	7	3	32 коробки по 500 г	785.25	29	0	10	ЛОЖЬ
3	3	Alice Mutton	7	6	20 банок по 1 кг	1755	0	0	0	ИСТИНА
4	4	Carnarvon Tigers	7	8	16 кг упаковка	2812.5	42	0	0	ЛОЖЬ
5	5	Teatime Chocolate Biscuits	8	3	10 коробок по 12 шт.	414	25	0	5	ЛОЖЬ
6	6	Sir Rodney's Marmalade	8	3	30 коробок	3645	40	0	50	ЛОЖЬ
7	7	Sir Rodney's Scones	8	3	24 упаковки по 4 шт.	450	3	40	5	ЛОЖЬ
8	8	Gustaf's Knackebrod	9	5	24 упаковки по 500 г	945	104	0	25	ЛОЖЬ
9	9	Tunnbrod	9	5	12 упаковок по 250 г	405	61	0	25	ЛОЖЬ
10	10	Guarana Fantastica	10	1	12 банок по 355 мл	202.5	20	0	0	ИСТИНА
11	11	NuNuCa Nuss-Nougat-Creme	11	3	20 банок по 450 г	630	76	0	30	ЛОЖЬ
12	12	Gumbar Gummibarchen	11	3	100 пакетов по 250 г	1405.35	15	0	0	ЛОЖЬ
13	13	Schoggi Schokolade	11	3	100 штук по 100 г	1975.5	49	0	30	ЛОЖЬ
14	14	Rossle Sauerkraut	12	7	25 банок по 825 г	2052	26	0	0	ИСТИНА
15	15	Thuringer Rostbratwurst	12	6	50 коробок по 30 шт.	5570.55	25	0	35	ИСТИНА
16	16	Nord-Ost Matjeshering	13	8	10 штук по 200 г	1165.05	10	0	15	ЛОЖЬ
17	17	Vegie-spread	7	2	15 банок по 625 г	1975.5	24	0	5	ЛОЖЬ
18	18	Wimmers gute Semmelknodel	12	5	20 пакетов по 4 шт.	1496.25	22	80	30	ЛОЖЬ
19	19	Louisiana Fiery Hot Pepper Sauce	2	2	32 бутылки по 237 мл	947.25	76	0	0	ЛОЖЬ
20	20	Louisiana Hot Spiced Okra	2	2	24 банки по 237 мл	765	4	100	20	ЛОЖЬ

Лист1

Рис. 15.31. Часть рабочего листа Excel, созданного из таблицы `Товары`

Обмен данными с Microsoft Word в системе автоматизации

В настоящем разделе будут рассмотрены несколько примеров управления объектами Microsoft Word из процедур VBA: открытие и печать документа, слияние документов для почтовых рассылок и заполнение полей в документе Word данными из БД Microsoft Access.

Иерархия объектов VBA приложения Microsoft Word

Так же как и при использовании объектной модели Microsoft Excel, познакомимся сначала с некоторыми объектами объектной модели Microsoft Word.

- Объект `Application` ссылается на активное приложение Microsoft Word. Через этот объект можно получить доступ ко всем остальным объектам модели. Объект `Application` имеет свойства `ActiveDocument` и `ActiveWindow`, которые представляют собой активный документ и активное окно, соответственно. Можно указать `Word.Application` в качестве значения аргумента `<класс>` функций `CreateObject` и `GetObject`, а также в операторе `Dim <объектнаяПеременная> As New <класс>`.
- Объект `Document` является глобальным и, так же как и объект `Application`, может использоваться в коде автоматизации в качестве значения аргумента `<класс>` функций `CreateObject`, `GetObject` и в операторе `Dim <объектнаяПеременная> As New <класс>`. Семейство `Documents` представляет собой набор всех документов, открытых в Word.
- Объект `Selection` представляет собой выделенную область в окне документа Word или точку вставки. Каждый документ может иметь только одну область выделения, и только одна область выделения открытых документов может быть ак-

тивна в некоторый момент времени. Это означает, что всегда существует только один объект `Selection`. Объект `Selection` подчинен объекту `Application`, а создается при применении метода `Select` к объекту `Document`.

- Семейство `Windows` объекта `Application` представляет собой набор всех окон приложения Word.
- Семейство `Bookmarks` подчинено объекту `Document`, и каждый объект этого семейства представляет собой закладку в документе.
- Объект `Range` представляет собой непрерывную область текста в документе Word. Диапазоны, как и закладки, используются для идентификации текста в документе, однако, в отличие от закладок, диапазоны существуют только во время выполнения процедур VBA. Диапазоны не зависят от области выделения в документе. Можно работать с диапазоном, не затрагивая области выделения. Диапазонов в документе может быть несколько.

Открытие документа Word

Для того чтобы открыть документ Word, можно использовать любой из перечисленных ранее способов создания экземпляров класса объекта автоматизации. Далее приводятся примеры процедур VBA для открытия документа. В первой используется функция `CreateObject`, а во второй — `GetObject` (листинги 15.2 и 15.3). Для проверки кода этих процедур вы можете, так же как и при работе с объектами приложения Excel, использовать окно отладки.

Листинг 15.2. Процедура открытия документа Word

```
Sub OpenDocument()
Dim wda As Word.Application
Set wda = CreateObject("Word.Application")
With wda
    .Visible = True
    .Documents.Open "C:\Doc\Letter.doc"
End With
'операции над документом
. . . . .
wda.Quit
Set wda = Nothing
End Sub
```

В данном случае независимо от того, запущен ли уже Word, программа запустит еще один его экземпляр и в нем откроет файл Letter.doc. После того как все операции с документом закончены, нужно закрыть Word, для чего используйте метод `Quit`. Если при этом документ был изменен, то приложение Word выдаст сообщение, спрашивающее о том, нужно ли сохранить изменения. При этом программа приостанавливает выполнение и ждет указания пользователя. Таких ситуаций следует избегать. Если все изменения делались программно, то лучше сначала закрыть документ, выполнив команду

```
wda.ActiveDocuments.Close False
```

Окно документа будет закрыто, при этом все изменения будут сохранены. Если же нужно, чтобы пользователь мог посмотреть сделанные программой изменения и при необходимости что-то откорректировать, можно не закрывать приложение в программе, а предоставить сделать это пользователю.

В следующей процедуре для открытия документа используется функция `GetObject`. То есть сначала делается попытка проверить, не запущен ли уже Word. Если он еще не запущен, будет сгенерирована ошибка (код ошибки: 429). Программа перехватит эту ошибку и использует функцию `CreateObject`, чтобы запустить Word. Если Word уже запущен, то новый документ будет открыт в этом же экземпляре приложения. Затем документ выводится на печать, после чего закрывается либо документ, либо приложение в зависимости от значения флага `modeFlag`. Значение флага показывает, запускался ли экземпляр Word, или документ был добавлен к другим открытым документам.

Листинг 15.3. Процедура открытия и печати документа Word

```
Sub OpenPrintDocument()
Dim wda As Word.Application
Dim modeFlag As Boolean

On Error GoTo ErrStartWord
modeFlag = True        'устанавливаем флаг операции
Set wda = GetObject(, "Word.Application")
With wda
    .Visible = True
    .Documents.Open "C:\Doc\Letter.doc"
    .ActiveDocument.PrintOut
    Do While .BackgroundPrintingStatus <> 0
        DoEvents       'ждем, пока документ напечатается
    Loop
    If modeFlag Then
        .ActiveDocument.Close 'закрываем только документ
    Else
        .Quit                          'закрываем все приложение
    End If
End With
Set wda = Nothing
Exit Sub

ErrStartWord:
    If Err.Number = 429 Then  ' Word не запущен
        Set wda = CreateObject("Word.Application")
        modeFlag = False      'сбрасываем флаг
        Resume Next           'возвращаемся к оператору, следующему за
                              'тем, который вызвал ошибку
    Else
       'выдаем диалоговое окно с сообщением и номером ошибки
        MsgBox Err.Description & " " & Err.Number, vbInformation
```

```
        Exit Sub        'выходим из процедуры
    End If

End Sub
```

Вывод данных из таблицы Access в документ Word

Такая операция может быть полезна, когда требуется автоматически внести данные в один из типовых документов: служебную записку, письмо, договор и т. д. Не всегда такие документы можно сформировать, используя отчеты Access. Например:

- требуется оформить договор, который, как правило, представляет собой довольно длинный текстовый документ, в который нужно вставить несколько чисел или строк;
- нужно обеспечить возможность ручного редактирования текста документа;
- документ должен быть сохранен в виде текстового файла (отчет Access может быть только напечатан).

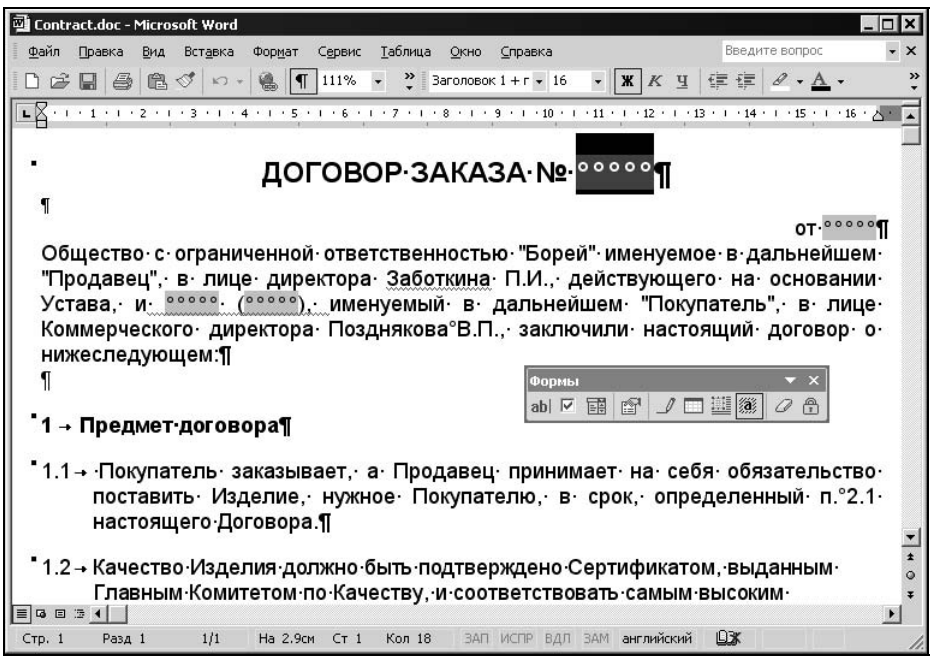


Рис. 15.32. Документ Word, содержащий поля для вставки

В качестве примера рассмотрим документ Word, который находится в файле Contract.doc на компакт-диске. Документ представляет собой текст договора и содержит несколько *полей формы*, которые должны быть заполнены данными из таблицы Access. *(Подробно о полях формы можно прочитать, например, в книге Ф. Новикова "Microsoft Word 2003" издательства "БХВ-Петербург", выпущенной в 2004 г.)* Откройте этот файл, запустив Word. На рис. 15.32 показаны начало документа и по-

ля, которые должны быть заполнены. Кроме того, представлена стандартная панель инструментов **Формы** (Forms), с помощью которой эти поля были созданы.

Замечание

Чтобы вывести на экран эту панель инструментов, щелкните правой кнопкой мыши в области панелей инструментов и выберите из списка элемент **Формы** (Forms).

Чтобы вставить такое поле в документ, нужно:

1. Установить курсор (точку ввода) в то место, где должно начинаться поле, и нажать кнопку **Текстовое поле** (Text Form Field). Поле будет вставлено в документ в точке ввода.
2. Дважды щелкнуть на этом поле. Откроется диалоговое окно **Параметры текстового поля** (Text Form Field Options, рис. 15.33), в котором требуется заполнить, по крайней мере, одно поле: **Закладка** (Bookmark).
3. Ввести имя закладки и нажать кнопку **OK**.

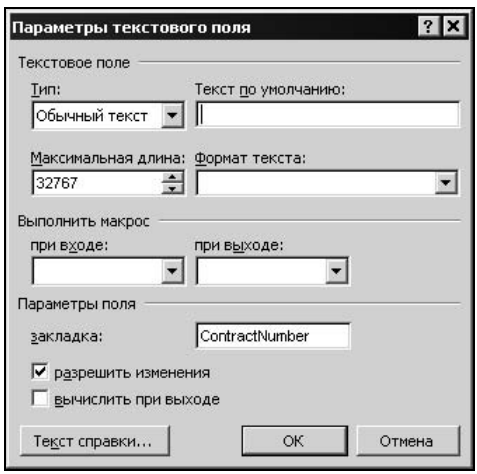


Рис. 15.33. Диалоговое окно **Параметры текстового поля**

Рассмотрим программу (листинг 15.4), которая открывает файл Contract.doc и автоматически заполняет поля договора. Такая программа может быть связана, например, с кнопкой в форме `Заказы клиентов` (Customer Orders) и обрабатывает нажатие данной кнопки. При нажатии на кнопку должен сформироваться договор с тем клиентом, который отображается в текущей записи формы. Для простоты будем считать, что номер договора будет равен номеру текущего заказа (выделенного в подчиненной форме).

Листинг 15.4. Программа автоматического заполнения полей

```
Private Sub CreateContract_Click()
Dim wda As Word.Application
```

```
Dim wdd As Word.Document
Dim intPrint As Integer
On Error GoTo ErrStartWord

Set wdd = GetObject("C:\Doc\Contract.doc")
Set wda = wdd.Parent
wda.Visible = True
wdd.Bookmarks("ContractNumber").Select
wda.Selection.TypeText Text:=[Подчиненная форма заказов 1].Form![КодЗаказа]
wdd.Bookmarks("ContractDate").Select
With wda
    .Selection.TypeText Text:=Date
    .Selection.GoTo Name:="CustemerName"
    .Selection.TypeText Text:=[Название]
    .Selection.GoTo Name:="CustomerAddress"
    .Selection.TypeText Text:=[Страна]

    ' заполнение остальных полей

    intPrint = MsgBox("Печатать договор?", vbYesNo + vbQuestion)
    If intPrint = vbYes Then
        wdd.PrintOut
        Do While .BackgroundPrintingStatus <> 0
            DoEvents
        Loop
    End If
    .ActiveDocument.Close False
    If .Windows.Count = 0 Then
        .Quit
    End If
End With

Set wdd = Nothing
Set wda = Nothing
Exit Sub

ErrStartWord:
    MsgBox Err.Description & " " & Err.Number, vbInformation
    Exit Sub
End Sub
```

В данной программе с помощью функции `GetObject` создается объект `Document`. Если при этом Word не запущен, то одновременно создается скрытый экземпляр объекта `Application` Word. Если Word уже запущен, то новый документ добавляется к уже открытым документам.

Далее показаны два способа нахождения нужной закладки в документе. В первом случае используется семейство `Bookmarks`. Вызвав метод `Select` нужного элемента

этого семейства, можно создать объект `Selection`, а в документе при этом выделяется требуемое поле.

Во втором случае к объекту `Selection` (вспомните, что он может означать не только выделенную область, но и точку ввода) применяется метод `Goto`, которому в качестве значения аргумента `Name` передается имя нужной закладки.

После того как все поля заполнены, договор выводится на печать. Но предварительно пользователь получает сообщение, требующее подтвердить необходимость печати. Документ печатается только, если пользователь подтвердит это.

Потом документ закрывается и далее проверяется, открыты ли еще какие-либо документы (используется свойство `Count` семейства `Windows`). Если других открытых документов нет, то приложение тоже закрывается. В конце процедуры освобождаются обе объектные переменные.

Если в организации отсутствует приложение Access для автоматизации различных деловых процедур, то создавать типовые бланки документов можно более простыми средствами, например, умело используя средства самого редактора Microsoft Word. Однако, когда такое приложение имеется, и в базе данных Access хранится большинство нужной информации, было бы просто нерационально вводить вручную в текст документа значения полей. Рассмотренный пример объясняет, как можно этого избежать.

Слияние документов для почтовых рассылок

Следующий пример связан с эффектным решением еще одной типовой задачи — созданием серийных документов. Это могут быть письма с одинаковым содержанием, но адресованные разным лицам, или типовые бланки, которые должны оформляться в большом количестве. Задача сводится к тому, что требуется создать шаблон такого документа и использовать какой-то источник данных для вставки их в каждый экземпляр документа. Если источником данных является таблица Access или запрос, задача решается просто — достаточно выделить эту таблицу (запрос) в списке в окне базы данных и выполнить команду меню **Сервис | Связи с Office | Слияние с Microsoft Office Word** (Tools | Office Links | Merge it With MS Word). Однако можно создать все бланки программно.

Сейчас мы создадим приглашение сотрудникам фирмы `Борей` на некоторое мероприятие. Мероприятие торжественное и важное, поэтому мы хотим напечатать приглашение на красивом бланке. Пример такого бланка представлен на рис. 15.34.

Шаблон документа должен быть приготовлен заранее. Рассмотрим, как это можно сделать.

1. Создайте красивый бланк, пользуясь средством Microsoft Clip Gallery или любой другой библиотекой картинок, рамочек и т. д. (Можно использовать шаблон, присутствующий на компакт-диске, — файл Приглашение.dot.) Это будет *основной документ* для слияния.
2. Теперь нужно установить связь *с источником данных*. В программе мы будем использовать в качестве источника данных временную таблицу Access, т. е. таблицу, которая существует только при работе процедуры VBA. Однако для установления связи нужно создать такую таблицу или запрос в базе данных. Создайте запрос на базе таблицы `Сотрудники` и определите в нем следующие поля: `Фамилия`, `Имя`, `Обращение`, `Должность`. Назовите его `СписокПриглашенных`.

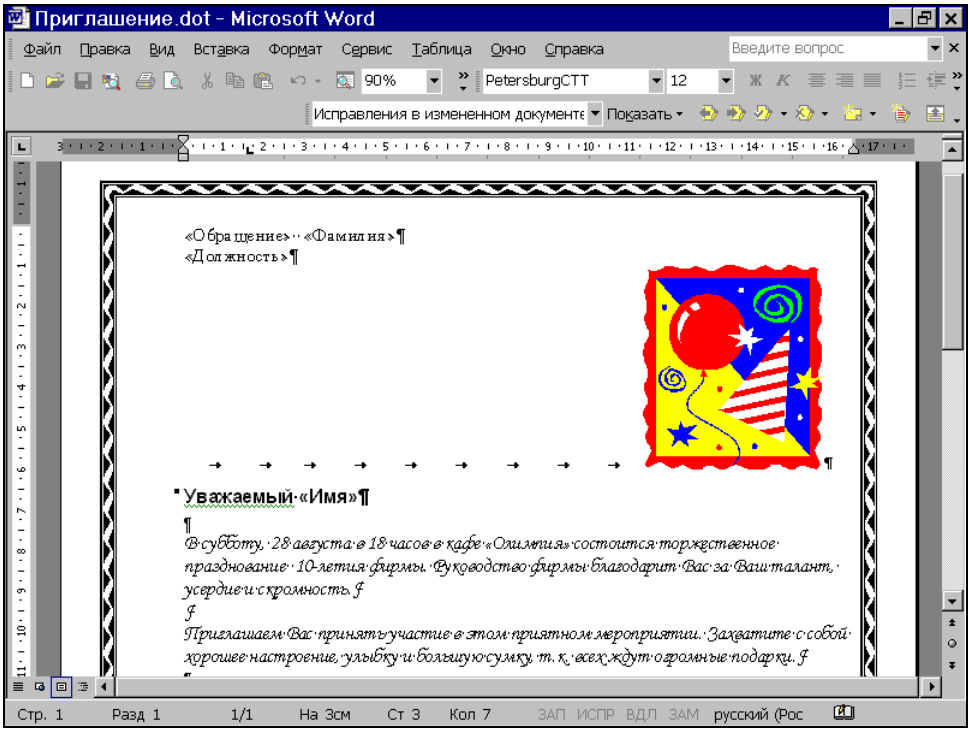


Рис. 15.34. Шаблон документа для слияния с данными из Access

3. Откройте файл Приглашение.dot и выполните команду меню **Сервис | Письма и рассылки | Мастер слияния** (Tools | Letters and Mailing | Mail Merge Wizard). Будет запущен Мастер, который реализован в виде панели задач слева на экране (рис. 15.35).
4. Далее следуйте указаниям Мастера. На первом шаге выберите тип документа — **Письма** (Letter) и нажмите кнопку **Далее** (Next), чтобы перейти к следующему шагу.
5. На втором шаге выберите переключатель **Текущий документ** (Use The current document).
6. На третьем шаге нужно указать источник данных. Оставьте значение поля **Выбор получателей** (Select Recipient) по умолчанию — **Использование списка** (Use an existing List) и нажмите на ссылку **Обзор** (Browse). Появится стандартное диалоговое окно для выбора файла.
7. Найдите в этом окне файл, в котором вы создавали запрос `СписокПриглашенных`, и нажмите кнопку **Открыть** (Open). Появится диалоговое окно со списком таблиц и запросов этой базы данных.
8. Выберите в нем запрос `СписокПриглашенных`. Появится диалоговое окно, в котором будет отображено содержимое исходной таблицы (рис. 15.36). Закройте это окно.

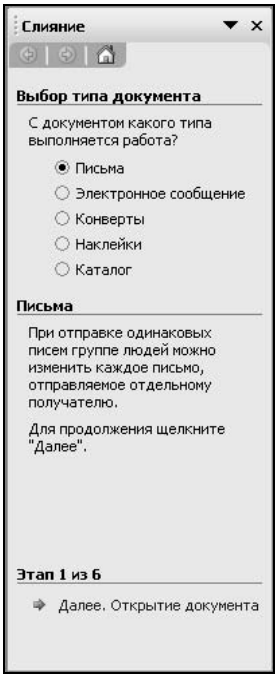


Рис. 15.35. Панель Мастера слияния

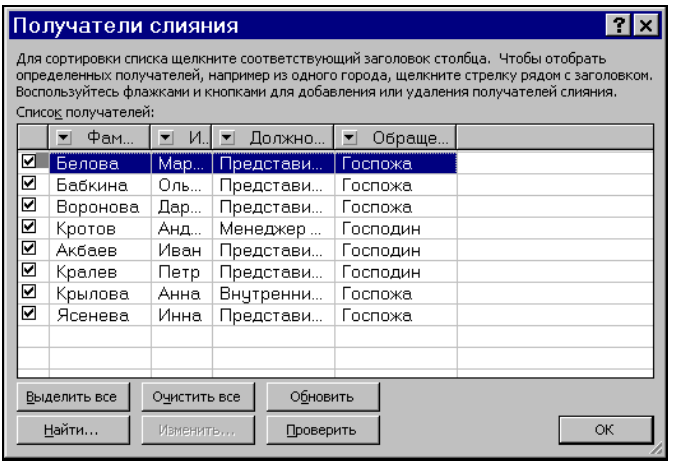


Рис. 15.36. Окно **Получатели слияния**

9. Чтобы создать в документе поля, в которые нужно вставлять данные из запроса, воспользуемся панелью инструментов **Слияние** (Mail Merge). Установите курсор в левый верхний угол документа и нажмите кнопку **Вставить поля слияния** (Insert Merge Fields). Появляется диалоговое окно со списком всех полей в источнике данных.

Замечание

Можно открыть то же окно, продолжая следовать указаниям Мастера — установить курсор в левый верхний угол документа и щелкнув ссылку **Другие элементы** на панели задач.

10. Выберите в списке поле `Обращение` и нажмите кнопку **Вставить** (Insert). Аналогично вставьте остальные поля (см. рис. 15.34). Поля слияния являются одним из видов полей Word и выглядят в основном документе как заключенные в угловые кавычки названия соответствующих полей источника данных.
11. Сохраните документ в виде шаблона (файл с расширением dot) и закройте.

Рассмотрим теперь программу, которая будет выполнять слияние документа Word с данными из таблицы Access. В этой программе мы покажем, как объект `Document` создается из шаблона документа. Предположим, что у нас есть форма, в которой отображаются отобранные записи о тех сотрудниках, которым мы хотим разослать приглашения. На форме есть кнопка, при нажатии на которую должны формиро-

ваться и печататься все экземпляры приглашений. Процедура обработки нажатия этой кнопки представлена в листинге 15.5.

Листинг 15.5. Слияние документа Word с данными из таблицы Access

```
Private Sub MergeDocument_Click()
Dim wda As Word.Application
Dim rst As Recordset, rstNew As Recordset
Dim db As Database
Dim tdf As TableDef
Dim i As Integer

Set db = CurrentDb()
' Создаем новую таблицу
Set tdf = db.CreateTableDef("СписокПриглашенных")
With tdf
    ' Создаем поля таблицы и добавляем их в семейство Fields
    .Fields.Append .CreateField("Фамилия", dbText)
    .Fields.Append .CreateField("Имя", dbText)
    .Fields.Append .CreateField("Обращение", dbText)
    .Fields.Append .CreateField("Должность", dbText)
    ' Добавляем таблицу в семейство TableDefs
    db.TableDefs.Append tdf
End With
' копируем записи из формы в созданную таблицу
Set rst = Me.RecordsetClone
rst.MoveFirst
Set rstNew = db.OpenRecordset("СписокПриглашенных")
Do While Not rst.EOF
    rstNew.AddNew
    For i = 0 To rst.Fields.Count - 1
        rstNew.Fields(i) = rst.Fields(i)
    Next i
    rstNew.Update
    rst.MoveNext
Loop

' создаем объект Application Word
On Error GoTo err_StartWord
Set wda = GetObject(, "Word.Application")
wda.Visible = True
' открываем документ на основе шаблона — он добавляется в семейство
' Documents
wda.Documents.Add "C:\Doc\Приглашение.dot"
' выполняем слияние основного документа и данных из источника
With wda.ActiveDocument.MailMerge
    .Destination = wdSendToNewDocument
    .Execute
End With
```

```
' печатаем приглашения
wda.ActiveDocument.PrintOut
Do While wda.BackgroundPrintingStatus <> 0
    DoEvents
Loop
' сохраняем получившийся документ
wda.ActiveDocument.SaveAs "C:\Doc\MailMergeDoc.doc"
' закрываем окно  с новым документом
wda.ActiveWindow.Close False
' закрываем, не сохраняя, окно с первоначальным документом
wda.Documents (1).Close False
' если нет больше открытых документов, то закрываем Word
If wda.Documents.Count = 0 Then
    wda.Quit
End If

' Удаляем временную таблицу
db.TableDefs.Delete "СписокПриглашенных"
db.Close
Set wda = Nothing
Set rst = Nothing
Set rstNew = Nothing
Exit Sub
err_StartWord:
    If Err = 429 Then ' Word не запущен
        Set wdd = CreateObject("Word.Application")
        Resume Next
    Else
        MsgBox Err.Description & " " & Err.Number, vbInformation
        Exit Sub
    End If

End Sub
```

Сначала создается таблица `СписокПриглашенных` с той же структурой, которая была создана вручную. Затем все данные, отображаемые в форме, записываются в эту таблицу. Все операции с данными выполняются с использованием объектов DAO. Объект `Application` Word создается с помощью функции `GetObject` или `CreateObject` в зависимости от того, запущен ли уже Word. Для создания нового документа на основе шаблона используется метод `Add` семейства `Documents`. При слиянии основного документа с записями в таблице создается новый документ. Для этого устанавливается значение свойства `Destination` объекта `MailMerge` равным константе `wdSendToNewDocument`. Этот документ уже не содержит поля слияния, а является просто текстовым документом, который можно сохранить в виде файла, задав его имя. Перед сохранением в файле документ выводится на печать. После этого нужно аккуратно закрыть сначала новый документ, затем основной документ, к которому присоединен источник данных, и потом, если нужно, сам Word.

Использование Access в качестве сервера автоматизации

Все примеры предыдущих разделов демонстрировали работу Access в качестве клиента автоматизации. Но это приложение является одновременно и сервером автоматизации, т. е. Access может предоставлять свои объекты для использования другим приложениям. Это значит, что другие приложения Microsoft Office могут программно управлять объектами Access. В листинге 15.6 приведен пример, когда из приложения Excel открывается база данных Access и данные из таблицы экспортируются в активный лист рабочей книги Excel.

Листинг 15.6. Access как сервер автоматизации

```
Dim objAccess As Access.Application
Dim db As Database
Dim rst As Recordset
Dim i As Integer, j As Integer

Set objAccess = GetObject("D:\Борей.mdb")
Set db = objAccess.CurrentDb
Set rst = db.OpenRecordset("Квартальные обороты по товарам")
'
' заполняем 1-ю строку заголовками столбцов таблицы
For i = 0 To rst.Fields.Count - 1
    Cells(1, I + 1) = rst.Fields(i).name
Next i
' переносим все остальные строки таблицы
j = 2
rst.MoveFirst
Do Until rst.EOF
    For i = 0 To rst.Fields.Count - 1
        Cells(j, I + 1) = IIf(IsNull(rst.Fields(i)), 0, rst.Fields(i))
    Next i
    j = j + 1
    rst.moveNext
Loop
objAccess.Close
Set obAccess = Nothing
```

Перед тем как создавать эту функцию, необходимо создать две ссылки в редакторе VBA Excel: на библиотеку Microsoft Access 11.0 Object Library и Microsoft DAO 3.6 Object Library. Создав объект `Access.Application`, можно получить доступ ко всем остальным объектам Access: таблицам, формам, отчетам. В данной программе сначала создается объект `Recordset` из нужной таблицы, а затем в цикле создается строка заголовков полей и переносятся все данные. На рис. 15.37 представлена результирующая таблица запроса `Квартальные обороты по товарам` и результат ее экспорта в Excel.

В качестве сервера автоматизации Microsoft Access применяется обычно для создания мастеров, генерирующих приложения баз данных в Access, при этом можно использовать функции `CreateForm`, `CreateControl`, `CreateReportControl`.

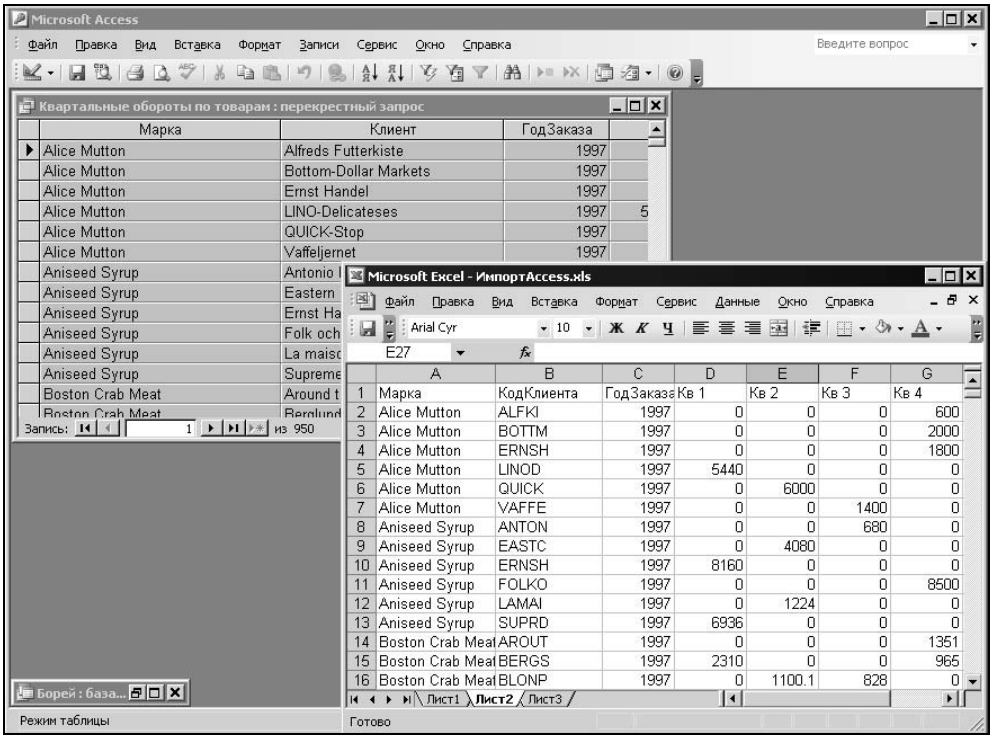


Рис. 15.37. Таблица `Квартальные обороты по товарам` и результат ее экспорта в Excel

Использование элементов ActiveX в приложении

Функциональные возможности Access могут быть расширены не только за счет управления объектами других приложений — серверов автоматизации. В форму или отчет Access могут быть просто встроены *элементы ActiveX*. Они представляют собой программный компонент, реализованный в виде отдельного файла. Но самостоятельными приложениями элементы ActiveX не являются — они выполняются только в том приложении, которое позволяет встраивать и использовать их, являясь для них контейнером. Вне приложения-контейнера ни один элемент управления ActiveX невозможно запустить на выполнение и требовать каких-либо результатов.

Существует большое количество элементов ActiveX. В этой главе мы познакомимся с одним из них, который включен в стандартную поставку Microsoft Office и используется очень часто. Это элемент **Календарь** (Calendar). Другие элементы ActiveX можно найти в Microsoft Visual Studio и прочих средствах разработки, в том числе и от других производителей.

Замечание

Если вы включили в свое приложение элементы ActiveX, необходимо следить за тем, чтобы эти элементы были установлены на всех компьютерах, на которых будет использоваться ваше приложение.

Добавление элемента управления *Календарь* в форму

Элемент управления **Календарь** может быть включен в форму для того, чтобы пользователь мог выбрать значение поля типа Дата/время, а не вводить дату вручную.

Чтобы включить элемент управления **Календарь** в форму:

1. Откройте форму в режиме Конструктора.
2. Выберите команду меню **Вставка | Элемент ActiveX** (Insert | ActiveX Controls), при этом откроется диалоговое окно **Вставка элемента ActiveX** (Insert ActiveX Controls) со списком всех зарегистрированных в системе элементов ActiveX.
3. Выберите в списке (рис. 15.38) элемент **Элемент управления Календарь 11.0** (Calendar Control 11.0) и нажмите кнопку **OK**. В форме появится календарь (рис. 15.39).

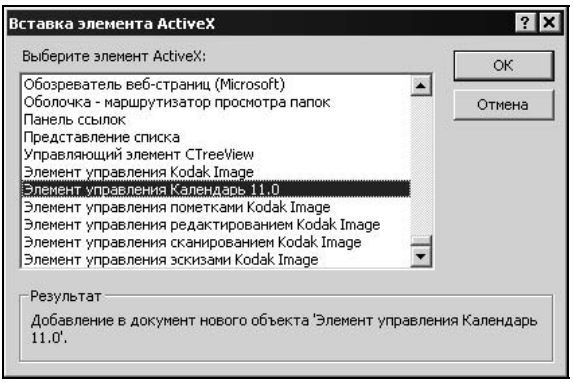


Рис. 15.38. Диалоговое окно **Вставка элемента ActiveX**

Далее можно настроить этот элемент, используя его свойства. В отличие от других элементов управления, в данном случае свойства имеют два объекта:

- элемент управления ActiveX формы (окно свойств то же самое, что у других элементов управления и у самой формы);
- объект **Календарь**, который реализуется этим элементом управления.

Откройте диалоговое окно свойств элемента ActiveX точно так же, как для любого элемента управления. Обратите внимание на свойство **Класс** (Class), определяющее класс элемента ActiveX. Значение этого свойства установилось автоматически: `MSCAL.Calendar.7`. Определите размеры элемента примерно 8×6 см.

Чтобы посмотреть свойства объекта **Календарь**:

1. Поместите указатель мыши на календарь и щелкните правой кнопкой мыши, чтобы получить контекстное меню для элемента управления формы.
2. В появившемся меню щелчком выберите элемент **Объект Календарь** (Calendar Control Object), при этом откроется объектное меню (см. рис. 15.39). Выберите в этом меню команду **Свойства** (Properties). Появится диалоговое окно свойств объекта **Календарь**.

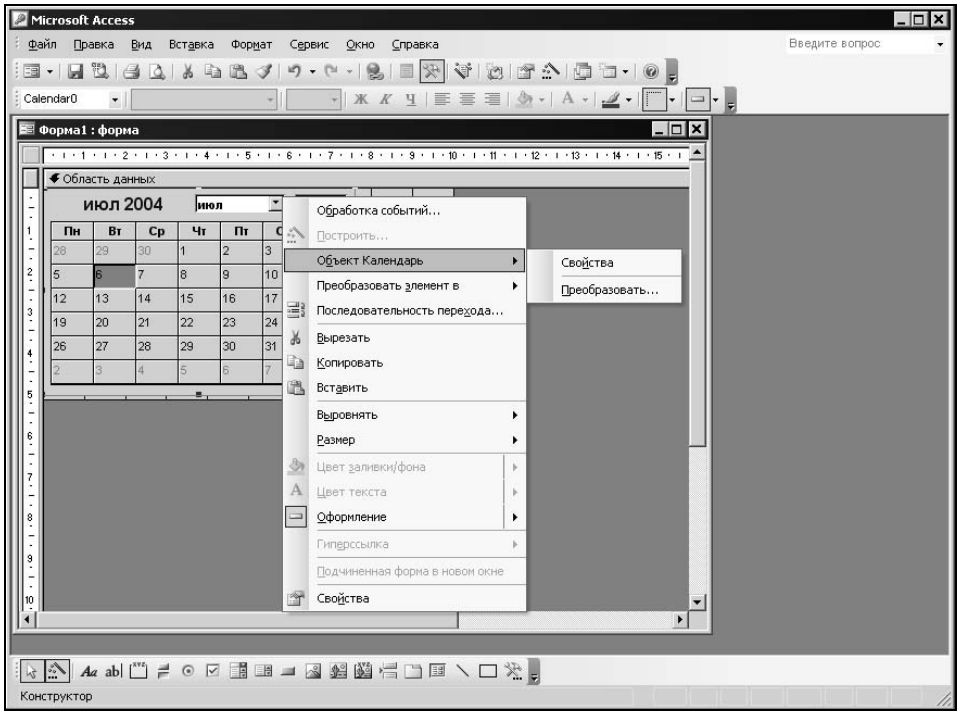


Рис. 15.39. Контекстное меню для элементов управления ActiveX

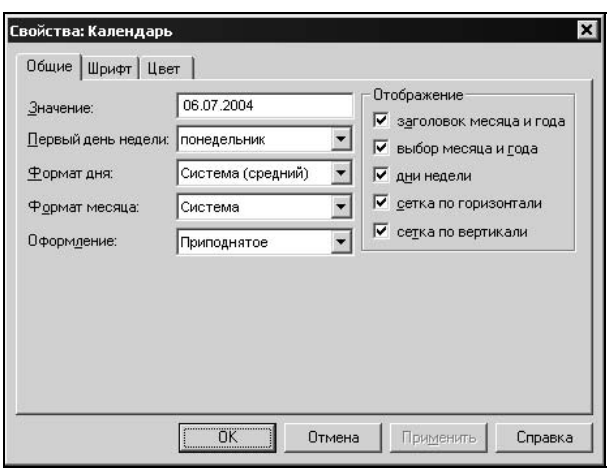


Рис. 15.40. Вкладка **Общие** диалогового окна **Свойства: Календарь**

3. По умолчанию окно свойств раскрывается на вкладке **Общие** (General), где можно установить некоторые общие свойства элемента управления **Календарь**. Установите свойства на этой вкладке согласно рис. 15.40.

4. Раскройте вкладку **Шрифт** (Fonts).
5. Для элемента **DayFont** из списка **Свойства** (Property Name) на вкладке **Шрифт** (Font) выберите шрифт **Arial Cyr**, в группе **Атрибуты** (Font Style) установите флажок **полужирный** (Bold), размер шрифта **10** (рис. 15.41).

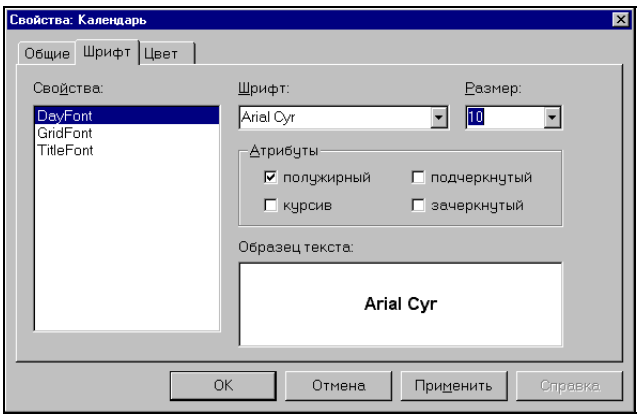


Рис. 15.41. Вкладка **Шрифт** диалогового окна **Свойства: Календарь**

6. Повторите процесс выбора шрифта, описанный на предыдущем шаге, для свойств `GridFont` (шрифт — **Arial Cyr**, начертание — **полужирный**, размер — **9**) и `TitleFont` (шрифт — **Arial Cyr**, начертание — **полужирный**, размер — **12**).
7. Раскройте вкладку **Цвет** (Colors) диалогового окна свойств. Эта вкладка предоставляет возможность изменить на любой из 16 стандартных цветов (или системных цветов Windows) цвета фона (`BackColor`), шрифта названий дней (`DayFontColor`), шрифта дат (`GridFontColor`), шрифта заголовка (`TitleFontColor`) и линий сетки (`GridLinesColor`) (рис. 15.42).

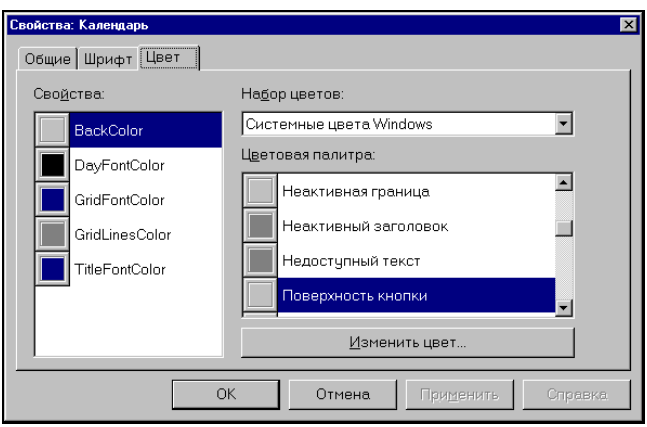


Рис. 15.42. Вкладка **Цвет** диалогового окна **Свойства: Календарь**

8. Нажмите кнопку **ОК**, чтобы закрыть окно свойств объекта. Откройте окно свойств формы. Установите значения **Отсутствуют** (Neither) для свойства **Полосы прокрутки** (ScrollBars) и **Нет** (No) для свойств **Кнопки перехода** (NavigationButtons) и **Область выделения** (RecordSelectors).
9. Переключите форму с встроенным календарем в режим Формы. Вид элемента управления **Календарь** в режиме Формы приведен на рис. 15.43. Чтобы изменить год или месяц, можно использовать раскрывающиеся списки в заголовке календаря.

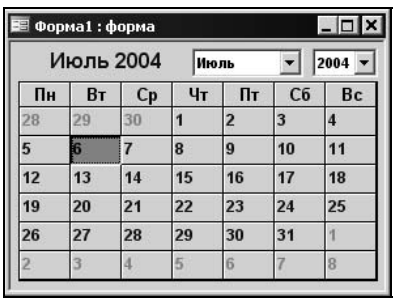


Рис. 15.43. Форма с элементом управления **Календарь** с раскрывающимися списками месяца и года

10. Вернитесь в режим Конструктора. Теперь мы рассмотрим, как можно использовать методы элемента **Календарь**.

Замечание

Чтобы познакомиться со всеми свойствами, методами и событиями любого элемента ActiveX, встроенного в форму, откройте окно свойств этого объекта и нажмите кнопку **Справка** (Help).

11. Справа от календаря расположите вертикально шесть кнопок. Назовите их так, как показано на рис. 15.44: `Следующая неделя`, `Предыдущая неделя`, `Следующий месяц`, `Предыдущий месяц`, `Следующий год`, `Предыдущий год`.
12. Ниже календаря создайте текстовое поле. В этом поле будет отображаться дата, выбранная в календаре. Присвойте свойству **Имя** (Name) этого поля значение `Дата`, а свойству **Значение по умолчанию** (Default Value) — значение `=Date()`.
13. Теперь напишем несколько процедур обработки событий. Во-первых, нужно, чтобы при открытии формы на календаре отображалась текущая дата. Для этого необходимо создать процедуру обработки события **Загрузка** (Load) формы. Далее приводится текст этой процедуры.

```
Private Sub Form_Load()
    Календарь.Object.Value = Date
End Sub
```

14. При выборе даты в календаре должно меняться значение поля `Дата`. Поэтому нужно написать процедуру обработки события **ПослеОбновления** (AfterUpdate) элемента управления **Календарь**. Как и предыдущая, эта процедура состоит из

одного оператора. Однако в ней показано, что для ссылки на значение элемента **Календарь** не обязательно использовать свойство `Value`.

```
Private Sub Календарь_AfterUpdate()
'обновление поля "Дата"
Дата = Format(Календарь.Object, "dd/mm/yy")
End Sub
```

Замечание

Вы наверно удивитесь, что в окне свойств элемента управления **Календарь** нет такого события. Есть только событие **При обновлении** (Updated). Но это событие не возникает, когда щелкаешь мышью по датам в календаре. Поэтому рекомендуем поступить так. Нажмите кнопку **Программа** (Code) на панели инструментов, чтобы открыть модуль формы. Выберите в списке **Object** (Объект) значение **Form** (Форма), а в списке **Procedure** (Процедуры) событие **AfterUpdate** (ПослеОбновления). Вставьте в шаблон процедуры указанную строчку.

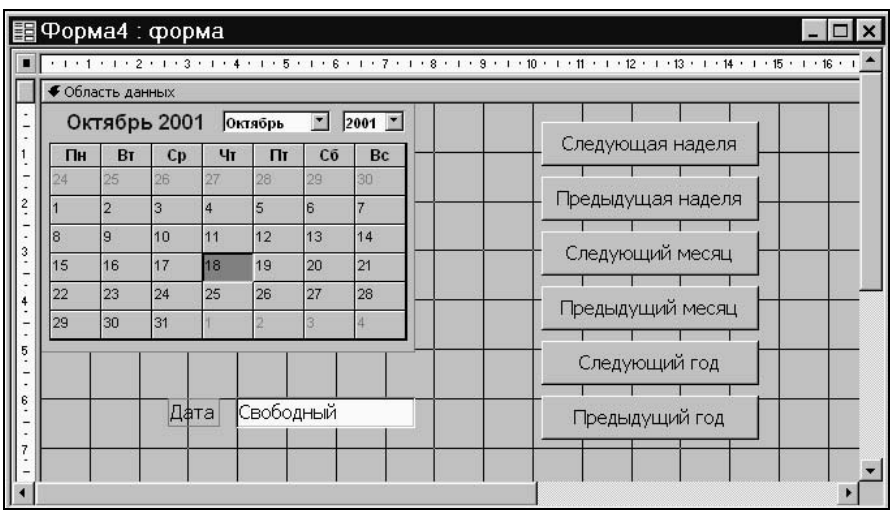


Рис. 15.44. Окончательный вид формы, содержащей элемент управления **Календарь** в режиме Конструктора

15. Еще шесть процедур должны обрабатывать событие **НажатиеКнопки** (Click) для каждой из шести кнопок. В этих процедурах используются шесть методов объекта **Календарь**:

```
Private Sub Предыдущая_неделя_Click()
Календарь.Object.PreviousWeek
End Sub

Private Sub Предыдущий_год_Click()
Календарь.Object.PreviousYear
End Sub
```

```
Private Sub Предыдущий_месяц_Click()
Календарь.Object.PreviousMonth
End Sub

Private Sub Следующая_неделя_Click()
Календарь.Object.NextWeek
End Sub

Private Sub Следующий_год_Click()
Календарь.Object.NextYear
End Sub

Private Sub Следующий_месяц_Click()
Календарь.Object.NextMonth
End Sub
```

16. Сохраните форму, дав ей имя `Календарь`.

Окончательный вид формы представлен на рис. 15.45.

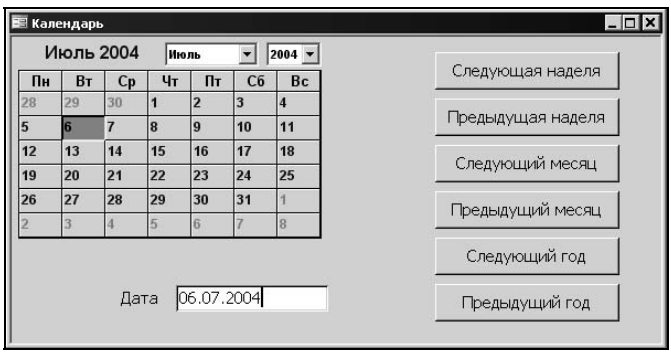


Рис. 15.45. Окончательный вид формы с элементом управления **Календарь**

Подключение специальных элементов управления в реестр Windows

Чтобы элементы ActiveX можно было использовать в приложениях Microsoft Access, сведения о них должны быть внесены в реестр Windows. В противном случае они просто отсутствуют в списке диалогового окна **Вставка элемента ActiveX** (Insert ActiveX Controls). Этот процесс называется *подключением*, или *регистрацией*. В процессе установки большая часть элементов ActiveX создает требуемые разделы в реестре Windows. На рис. 15.46 приведены разделы реестра для элемента управления **Календарь**.

Однако многие свободно распространяемые элементы ActiveX не имеют в своем составе программы установки, поэтому следует подключить их самостоятельно.

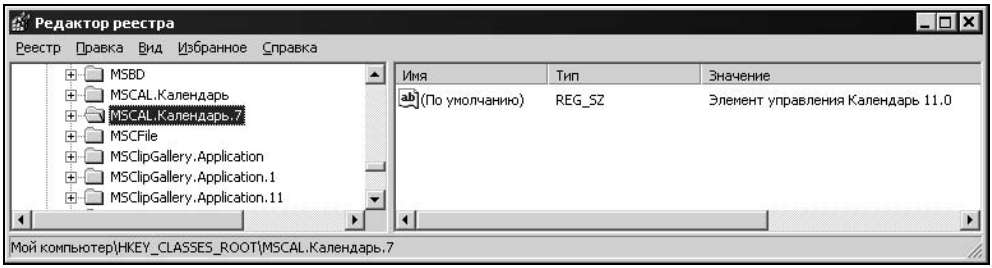


Рис. 15.46. Разделы специального элемента управления **Календарь** в реестре Windows

Чтобы внести в реестр сведения о специальном элементе управления автоматизации для Microsoft Access или других приложений:

1. Выберите команду **Сервис | Элементы ActiveX** (Tools | ActiveX Controls), чтобы открыть диалоговое окно **Элементы ActiveX** (ActiveX Controls). Затем нажмите кнопку **Подключить** (Register), что приведет к открытию диалогового окна **Добавление элемента ActiveX** (Add ActiveX Control).
2. Найдите файл, соответствующий элементу ActiveX (обычно он имеет расширение ocx или dll), который необходимо подключить (рис. 15.47), и нажмите кнопку **OK**. После этого вновь подключенный элемент должен появиться в списке **Доступные элементы управления** (Available Controls) диалогового окна **Элементы ActiveX**.

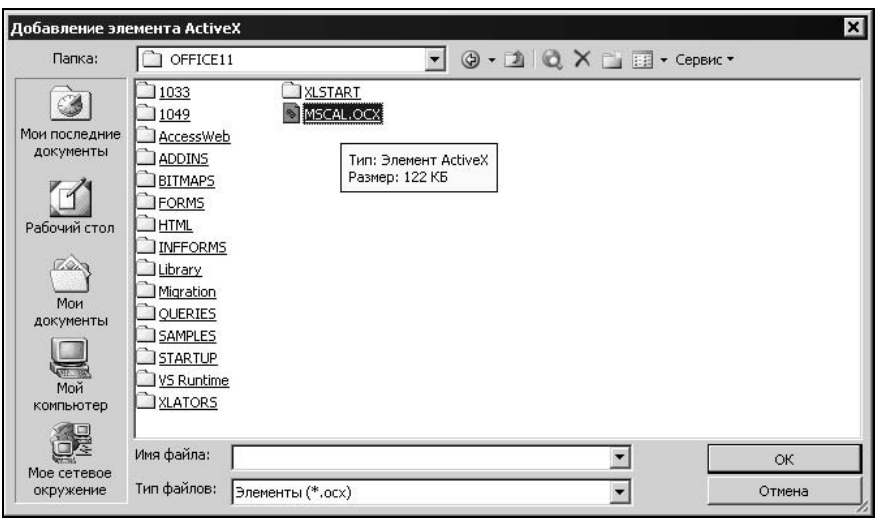


Рис. 15.47. Выделение файла с расширением ocx для подключения

При удалении элемента ActiveX с жесткого диска рекомендуется также удалить соответствующий раздел из реестра. Для удаления раздела из реестра выделите имя специального элемента управления в списке **Доступные элементы** диалогового окна **Элементы ActiveX**, затем нажмите кнопку **Отключить** (Unregister).

Программный доступ к документам XML

Документы XML легко импортируются и создаются с помощью пользовательского интерфейса Access. Кроме того, существует возможность программного импорта/экспорта данных из файлов XML, для чего используются методы объектов модели ADO. Однако существует технология, которая позволяет анализировать содержимое документов XML, и не только анализировать, но и вносить в них изменения. И все это из программы VBA. Этой технологией является не что иное, как работа с *объектной моделью XML* (XML Document Object Model, XML DOM). Эта модель создана на основе спецификаций и набора интерфейсов, разработанных Консорциумом W3C, и подробно познакомиться с ней вы можете на сайте **http://www.w3.org/DOM**.

Для работы с XML DOM необходим специальный компонент Microsoft XML Core Services (MSXML). При установке приложений Microsoft Office 2003 устанавливается и регистрируется файл MSXML5.DLL, который представляет собой версию 5.0 этого компонента. Поэтому для использования объектов DOM необходимо создать ссылку на этот файл.

1. Откройте проект VBA и выполните команду **Tools | References** (Сервис | Ссылки).
2. В диалоговом окне **References** (Ссылки) выберите компонент **Microsoft XML, v5.0** (рис. 15.48).
3. Закройте окно, нажав кнопку **OK**.

Замечание

Если вы не видите такого компонента в списке, то подключите его, как описано в предыдущем разделе. Нужный файл должен находиться в папке C:\Program Files\Common Files\Microsoft Shared\OFFICE.

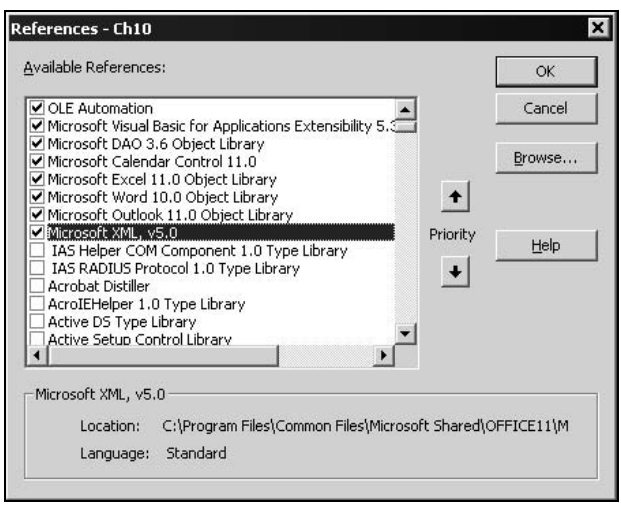


Рис. 15.48. Установка ссылки на XML DOM

После того как ссылка установлена, можно открыть **Object Browser**, выбрать библиотеку **MSXML** и увидеть объекты этой модели, их свойства и методы. Познакомимся с некоторыми из них.

Чтобы начать работать с содержимым файла XML, необходимо сначала создать объект `DOMDocument` — объект верхнего уровня, который содержит весь документ XML. Это можно сделать, например, так:

```
Dim objDoc As MSXML2.DOMDocument50
Set objDoc = New MSXML2.DOMDocument50
```

Объект имеет метод `Load`, который позволяет загрузить файл XML. В качестве аргумента при вызове этого метода передается имя загружаемого файла. Например,

```
objDoc.Load "C:\Сотрудники.xml"
```

После выполнения этой операции документ XML загружается в память в виде дерева объектов. Теперь его можно анализировать с помощью любого анализатора (parser), основанного на структуре дерева. Одним из таких анализаторов и является MSXML.

Загрузив документ, можно посмотреть его свойство `XML`. Введите в окне **Immediate** строку

```
? objDoc.XML
```

Будет выведен весь текст загруженного документа.

Дерево DOM состоит из узлов, которым соответствуют объекты `IXMLDOMNode`. Эти узлы составляют семейство `IXMLDOMNodeList`. Для нашего примера это семейство состоит из двух узлов: первый соответствует прологу документа и состоит из одной строки:

```
<?xml version="1.0" encoding="UTF-8" ?>
```

Второй соответствует корневому элементу документа, который начинается с тега `<dataroot>` и заканчивается тегом `</dataroot>` (рис. 15.49).

Корневой элемент содержит несколько вложенных элементов с именем `Сотрудники`. Этим элементам соответствуют объекты, которые входят в семейство `childNodes` объекта `IXMLDOMNode`. Метод `hasChildNodes` позволяет узнать, есть ли у данного объекта-узла потомки. Он возвращает значение `True` или `False`. Семейство `childNodes` содержит всех потомков данного узла.

Введите в окне **Immediate** строку

```
?objdoc.childNodes(1).childNodes(0).nodeName
```

и вы получите имя `Сотрудники`. И так далее. Продолжив вложения и введя

```
?objdoc.childNodes(1).childNodes(0).childNodes(0).nodeName
```

можно получить уже наименование первого поля в таблице `Код сотрудника`, а на вопрос

```
?objdoc.childNodes(1).childNodes(0).childNodes(0).text
```

вы получите сам код — 1.

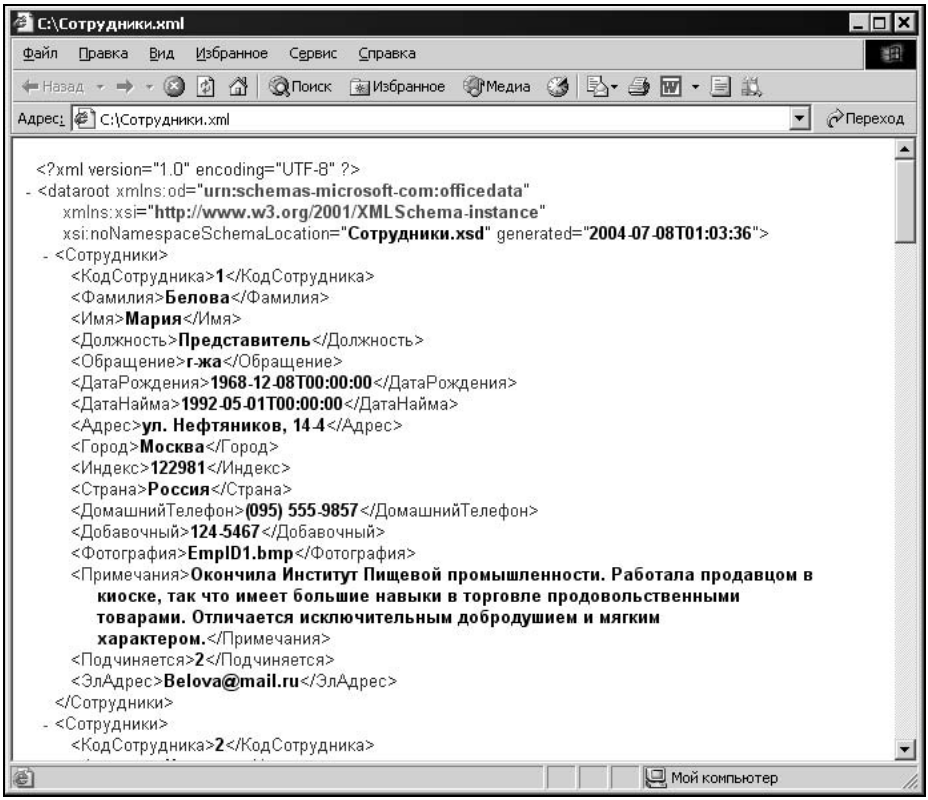


Рис. 15.49. Документ XML, полученный путем экспорта таблицы `Сотрудники`

Чтобы обратиться к первому вложенному элементу узла, можно использовать свойство `firstChild`, а к последнему — `lastChild`. Перейти к следующему элементу в семействе `childNodes` можно с помощью свойства `nextSibling`, а к предыдущему — `previousSibling`.

Родительский узел можно получить с помощью метода `parentNode`.

Введите в окне **Immediate** строку:

```
?objdoc.childNodes(1).childNodes(0).childNodes(0).parentNode.nodeName
```

и вы снова получите имя `Сотрудники`.

Значением свойства

```
?objdoc.childNodes(1).childNodes(0).firstChild.nodeName
```

будет `Код сотрудника`, а в ответ на вопрос

```
?objdoc.childNodes(1).childNodes(0).firstChild.nextSibling.nodeName
```

вы получите — `Фамилия`.

Узлы документа могут иметь разные типы. Каждый тип узла имеет свои свойства и методы. Узнать тип узла можно, прочитав значения свойства `nodeType`. Возможные значения приведены в табл. 15.4.

Таблица 15.4. *Типы узлов дерева DOM*

Константа	Значение	Комментарий
NODE_ELEMENT	1	Узел является листом дерева
NODE_ATTRIBUTE	2	Атрибут элемента
NODE_TEXT	3	Текст элемента
NODE_CDATA_SECTION	4	Данные типа CDATA
NODE_ENTITY_REFERENCE	5	Ссылка на сущность
NODE_ENTITY	6	Сущность `Entity`
NODE_PROCESSING_INSTRUCTION	7	Инструкция по обработке
NODE_COMMENT	8	Комментарий
NODE_DOCUMENT	9	Корневой узел типа `Document`
NODE_DOCUMENT_TYPE	10	Узел типа `Document`
NODE_DOCUMENT_FRAGMENT	11	Фрагмент документа
NODE_NOTATION	12	Нотация

Чтобы обращаться к конкретному типу узла, следует использовать объект `IXMLDOMElement`.

Элемент в документе XML может содержать текст, комментарии, ссылки на сущности, секцию CDATA, инструкции по обработке, атрибуты, а также вложенные элементы.

В соответствии с этим объект `IXMLDOMElement` может содержать объекты следующих типов: `IXMLDOMText`, `IXMLDOMComment`, `IXMLDOMEntityReference`, `IXMLDOMCDATASection`, `IXMLDOMProcessingInstruction`, `IXMLDOMElement` в качестве объектов-потомков.

Читать и изменять значения атрибутов можно с помощью методов `getAttribute` и `setAttribute` объекта `IXMLDOMElement`.

Таким образом можно обращаться к каждому элементу в дереве и не только читать, но и менять данные в этом элементе.

Методы `appendChild`, `removeChild`, `replaceChild`, `insertBefore` объекта `IXMLDOMElement` позволяют менять структуру документа. Аналогичные методы имеет и объект `IXMLDOMNode`.

Создать новый пустой узел типа `Element` можно с помощью метода `createElement` объекта `DOMDocument50`. Этот метод имеет один строковый аргумент — имя тега для данного элемента.

Например, можно создать узел-элемент следующим образом:

```
Set objNode = objDoc.createElement("Заметки")
```

Проверьте результат этой операции, введя

```
?objNode.nodeName
```

Вы должны получить имя тега — `Заметки`. Содержание элемента можно указать, определив свойство `text`.

```
objNode.text = "Выяснить день рождения"
```

Теперь добавим новый узел в дерево. Допустим, мы хотим добавить новое поле в таблице `Сотрудники`. Определим узел, к которому мы будем добавлять узел-потомок. Это должен быть узел с именем `Сотрудники`.

```
Dim thisNode As IXMLDOMNode
Set thisNode = objdoc.childNodes(1).childNodes(0)
```

И добавим в его семейство `childNodes` созданный ранее узел `Заметки`.

```
thisNode.appendChild(objNode)
```

Точно так же с помощью других методов `create` объекта `DOMDocument50` можно создавать другие элементы XML — атрибуты, комментарии, фрагменты документов, ссылки на сущности и т. д.



ЧАСТЬ III

Разработка и сопровождение приложений Access 2003

Глава 16. Разработка сетевых приложений Access

Глава 17. Проекты Access 2003

Глава 18. Репликация баз данных

Глава 19. Миграция приложений

Глава 20. Администрирование баз данных

ГЛАВА 16



Разработка сетевых приложений Access

База данных Access может использоваться одним пользователем на локальном компьютере или несколькими пользователями в сети. Применение базы данных в сети имеет ряд особенностей. Чтобы подготовить базу данных для использования в сети, необходимо выполнить специальную настройку среды Access, параметров базы данных и ее объектов и реализовать особенности общего доступа к данным в программном коде приложения. Об этом и пойдет речь в данной главе. Чтобы рассказ не был слишком абстрактным, мы рассмотрим основные вопросы на примере сетевого приложения Access "Игра в доминирование". Это приложение есть на компакт-диске с примерами к этой книге.

(Установка сетевого приложения "Игра в доминирование" описана в соответствующем разделе *Приложения 2.*)

В этой главе рассмотрены особенности использования и разработки сетевых приложений Access.

Способы совместного использования баз данных

Базы данных Microsoft Access могут использоваться одновременно несколькими пользователями в сети. Предоставить общий доступ к базе данных можно несколькими способами:

- предоставить совместный доступ к базе данных в сети;
- предоставить совместный доступ к таблицам в базе данных в сети;
- предоставить совместный доступ к объектам базы данных в Интернете;
- обеспечить совместную работу с базой данных с помощью репликации;
- обеспечить совместный доступ к базе данных с архитектурой "клиент-сервер" с помощью SQL Server.

Совместное использование базы данных

Чтобы предоставить совместный доступ к базе данных Access, поместите ее на файловый сервер или в общую папку на рабочей станции. Это самый простой способ обеспечить совместный доступ к базе данных и всем ее объектам: формам, отчетам, запросам, макросам и модулям. Этот способ подходит, если нужно, чтобы все пользователи могли использовать одни и те же объекты базы данных одинаковым образом.

Поместив базу данных в общую папку на файловом сервере или рабочей станции, убедитесь в том, что установлен параметр, определяющий открытие базы данных по умолчанию в режиме совместного доступа. *(Об установке этого параметра рассказано в разд. "Режимы работы с базой данных в сети" далее в настоящей главе.)*

Для работы с базой данных в сети необходимо выполнить локальную или сетевую установку Access, либо установить исполняемую версию Access.

Совместное использование только данных

Можно предоставить совместный доступ в сети лишь к таблицам в базе данных Access. Для этого необходимо разделить базу данных на файл объектов данных и файл объектов приложения. *(Разделение базы данных с помощью Мастера описано в гл. 19.)*

После разделения базы данных поместите файл объектов данных на файловый сервер или в общую папку на рабочей станции, чтобы предоставить общий доступ к таблицам, а файл объектов приложений — на компьютер пользователя в виде локальной копии. В этом случае производительность базы данных будет несколько выше, чем при совместном доступе ко всей базе данных, поскольку по сети будут передаваться только данные из таблиц. При этом отдельный пользователь может настроить формы, отчеты и прочие объекты из файла объектов приложения в соответствии со своими индивидуальными потребностями. Эти изменения не отразятся на остальных пользователях, которые используют свои локальные копии файла объектов приложения.

Совместное использование объектов базы данных через Интернет

Объекты базы данных можно предоставить в совместное использование через Интернет, создав на их основе страницы HTML следующих видов:

- статические страницы HTML;
- генерируемые сервером страницы HTML;
- страницы доступа к данным;
- документы в формате XML.

Страницы HTML можно отобразить в программе просмотра Интернет, такой как Internet Explorer. *(О работе с базами данных Access через Интернет см. разд. "Публикация данных в Интернете и корпоративной сети" гл 11.)*

Совместная работа с базой данных с помощью репликации

Портфельная репликация Windows позволяет использовать для работы с файлом (в том числе с базой данных) два компьютера: например, компьютер в офисе и переносной компьютер. Можно создавать реплики (специальные копии) базы данных Access для хранения на разных компьютерах и синхронизировать изменения в них. При этом несколько пользователей могут работать со своими копиями — репликами базы данных, которые могут быть синхронизированы через подключение по сети, удаленный доступ к сети или через Интернет. *(Репликация баз данных Access описана в гл. 18.)*

Совместный доступ к базе данных с помощью SQL Server

Если в сети организации установлена СУБД SQL Server, ее можно использовать для совместной работы с базой данных Access. Для этого нужно создать клиент-серверное приложение на основе базы данных Access. В результате станут доступными мощные средства СУБД SQL Server. Можно преобразовать существующую базу данных Access в клиент-серверное приложение с помощью Мастера *(см. разд. "Перенос сетевого приложения на платформу SQL Server" гл. 19)* или создать новое приложение в архитектуре "клиент-сервер". *(Разработка проектов Access, подключенных к SQL Server, описана в гл. 17.)*

Режимы работы с базой данных в сети

База данных может быть открыта пользователем в сети в одном из четырех режимов:

- *общего доступа* — база данных открывается для чтения и записи. Другие пользователи тоже могут открыть базу данных (в любом режиме);
- *монопольного доступа* — база данных открывается для чтения и записи. Другие пользователи не смогут открыть базу данных, пока она не будет закрыта пользователем, открывшим ее в режиме монопольного доступа;
- *общего доступа только для чтения* — база данных открывается только для чтения. Другие пользователи тоже могут открыть базу данных (в любом режиме);
- *монопольного доступа только для чтения* — база данных открывается только для чтения. Другие пользователи не смогут открыть базу данных, пока она не будет закрыта пользователем, открывшим ее в режиме монопольного доступа.

Можно указать режим открытия базы данных, используемый по умолчанию:

1. Откройте базу данных.
2. Выберите команду **Сервис | Параметры** (Tools | Options). Появится диалоговое окно **Параметры** (Options).
3. Раскройте вкладку **Другие** (Advanced).
4. В группе **Режим открытия по умолчанию** (Default open mode) выберите переключатель **общий доступ** (Shared) или **монопольный доступ** (Exclusive).

Заданный режим открытия по умолчанию действует только на локальном компьютере и не влияет на режим, используемый по умолчанию для открытия базы данных пользователями других компьютеров.

Кроме того, независимо от установленного режима по умолчанию, пользователь может открыть базу данных в любом режиме, указав его явно:

1. Выберите команду **Файл | Открыть** (File | Open). Появится диалоговое окно **Открытие файла базы данных** (Open).
2. Щелкните по стрелке, находящейся справа от кнопки **Открыть** (Open). Раскроется список всех возможных режимов открытия базы данных (рис. 16.1).
3. Выберите в списке элемент, соответствующий требуемому режиму открытия базы данных:
   - **Открыть** (Open) — режим общего доступа для чтения и записи;
   - **Открыть для чтения** (Open Read-Only) — режим общего доступа, открыть только для чтения;
   - **Монопольно** (Open Exclusive) — режим монопольного доступа;
   - **Монопольно для чтения** (Open Exclusive Read-Only) — режим монопольного доступа, открыть только для чтения.

Замечание

Можно запретить определенным пользователям сети открывать базу данных, защищенную на уровне пользователей, в монопольном режиме. Для этого нужно удалить права этих пользователей на монопольный доступ к базе данных. *(Об установке прав доступа рассказано в гл. 20.)*

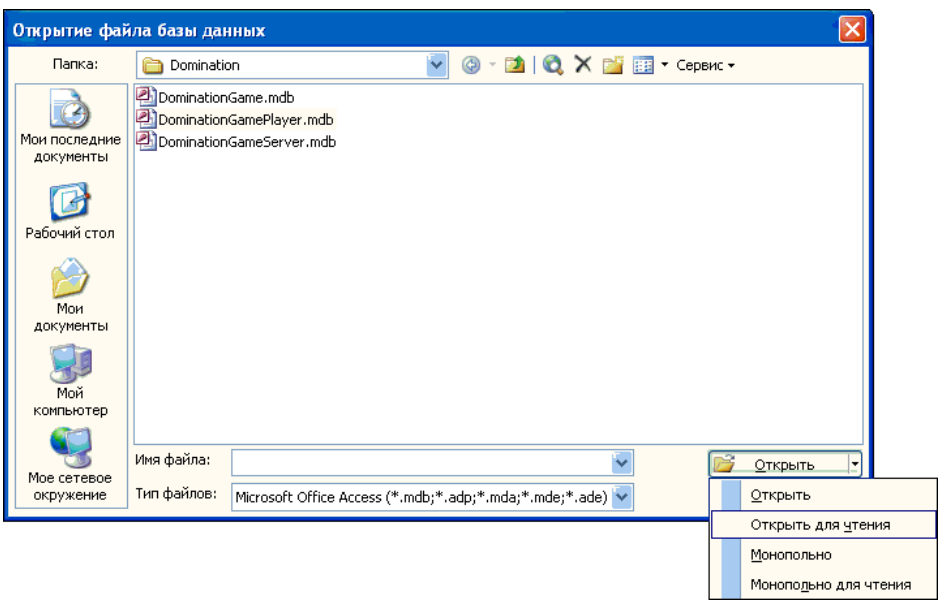


Рис. 16.1. Список режимов открытия базы данных

Если база данных (файл с расширением mdb) открывается в режиме общего доступа, Microsoft Access создает *файл блокировок* с таким же именем, как база данных, но с расширением ldb, который располагается в той же папке, где хранится файл базы данных. В файле блокировок сохраняется имя компьютера (например, `MYCOMP`) и регистрационное имя пользователя базы данных (например, `Admin`) для каждого пользователя, открывшего базу данных для совместного использования. Этот файл применяется Microsoft Access во время совместной работы пользователей с файлом базы данных. Обычно он удаляется автоматически после того, как последний пользователь закрыл базу данных.

Замечание

Иногда в результате сбоев файл блокировок не удаляется автоматически. В этом случае его следует удалить самостоятельно, иначе файл баз данных будет заблокирован и пользователи не смогут открыть базу данных.

Пример сетевого приложения

Рассмотрим в качестве примера сетевого приложения Access приложение "Игра в доминирование". Далее в этой главе на данном примере будут проиллюстрированы основные особенности разработки и использования сетевых приложений. Готовое приложение "Игра в доминирование" находится на компакт-диске с примерами к этой книге. Полное описание приложения содержится в *Приложении 2*.

Демонстрационный пример сетевого приложения "Игра в доминирование" был разработан автором (В. Михеевой) еще в версии Access 2000, и для иллюстрации этой главы в новом издании книги он был сохранен полностью по нескольким причинам. Во-первых, в новой версии Access 2003 реализована полноценная поддержка формата баз данных версии Access 2000. Во-вторых, в новой версии Access 2003 не появилось особенных новшеств, касающихся совместной работы с приложениями Access в сети.

Рассмотрим кратко, в чем заключается суть приложения "Игра в доминирование". Вообще говоря, подход к реализации этого приложения является типичным примером проектирования сетевых приложений.

Правила игры

"Игра в доминирование" — это состязание между несколькими игроками за захват максимальной части игрового поля. Поле состоит из клеток. Каждая клетка может быть занята одним из игроков на некоторых определенных игрой условиях. Чтобы занять некоторый участок поля, игрок делает ход. Ход может быть принят или отвергнут, в зависимости от того, занята ли эта или смежные клетки другим игроком в соответствии с принятыми правилами. Все игроки имеют общий доступ к полю и стремятся занять клетки одновременно. За занятые клетки игроку присуждаются очки. Кто успеет набрать наибольшее число очков, тот и победитель.

Сетевое решение в реализации архитектуры приложения

Даже из столь абстрактного описания логики игры можно сделать выводы о том, что архитектура приложения должна представлять собой несколько компонентов, взаимодействующих между собой в сети. Вот основные характеристики задачи, которые являются аргументами в пользу сетевого решения:

- у приложения несколько пользователей, работающих с общими данными (игровым полем), т. е. в задаче требуется многопользовательский доступ к общим данным;
- всю задачу можно разделить на относительно независимые подзадачи (ход игрока, решение о принятии или непринятии хода, ведение счета очков и т. п.), которые могут выполняться параллельно или последовательно (по очереди);
- каждая из выделенных подзадач — это отдельный компонент приложения;
- каждый компонент может выполняться на отдельном вычислительном узле в сети, при этом некоторые компоненты могут быть объединены на одном узле сети, т. е. приложение может быть распределено по узлам сети.

Таким образом, наше приложение имеет *многокомпонентную* или *многоуровневую* архитектуру. Для таких задач типичным решением бывает обычно *двух-* или *трехуровневая* архитектура, когда приложение состоит из двух или трех типов взаимодействующих компонентов, среди которых выделяются: "обслуживающие" и "потребляющие" компоненты (серверы и клиенты). Серверный компонент может обслуживать от одного до нескольких клиентов одновременно. При необходимости вводят третий тип компонента — "промежуточный" серверный компонент между клиентским компонентом и основным серверным компонентом, чтобы разгрузить слишком нагруженный логикой обработки данных основной серверный компонент.

Замечание

Большее число уровней встречается реже, обычно в области специализированных сетевых и коммуникационных задач.

Описанную архитектуру называют *клиент-серверной*. Наше приложение имеет такую архитектуру. Его разделение на компоненты выглядит следующим образом:

- *серверный компонент* — это игровое поле, которое является общим ресурсом приложения, вместе с управляющим этим ресурсом звеном игры — сервером игры. Сервер принимает и обрабатывает заявки от клиентов;
- *клиентский компонент* — это отдельный игрок, который посылает серверу заявки на использование общего ресурса (поля игры) и получает ответ.

Access предоставляет две возможности для реализации приложений с такой архитектурой:

- в виде сетевых баз данных (одна или несколько связанных баз данных, размещенных в сети);

Замечание

Сетевое приложение Access может состоять и из одного компонента, не разделенного на части клиент и сервер, — одной базы данных с разрешенным общим доступом. Но это не эффективное решение.

- в виде проектов баз данных, соединенных с данными на SQL Server.

(Информацию о клиент-серверной архитектуре в приложениях, связанных с SQL Server, см. в гл. 17.)

В этой главе мы говорим о первом способе — сетевых базах данных.

Сетевое приложение Access "Игра в доминирование" должно будет обслуживать несколько игроков — разных пользователей в сети. Предположим, пользователи работают в одноранговой сети. Рабочие станции пользователей, принимающих участие в игре, можно разделить на две категории: клиенты и серверы. На сервере выполняется ядро игры — управляющий компонент приложения. На клиентских рабочих станциях устанавливается компонент, предоставляющий пользователю интерфейс для участия в игре. Таким образом, приложение "Игра в доминирование" представляет собой распределенную базу данных Access с архитектурой "клиент-сервер", которая может быть использована даже в одноранговой сети. Все участники одной игры подключаются к одному серверу по схеме "звезда" (рис. 16.2).

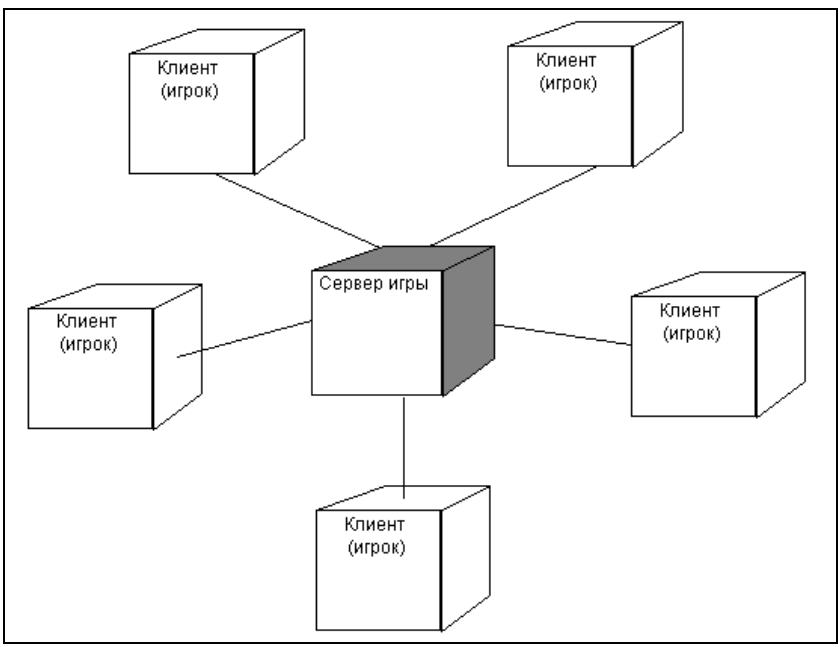


Рис. 16.2. Архитектура приложения "Игра в доминирование"

Многопользовательский доступ к данным требует организации распределения этого доступа между пользователями и обеспечение защиты данных, в соответствии с правами пользователей. Все эти особенности сетевых приложений реализованы в на-

шем примере, а методы их реализации описаны в данной главе. *(Учетные записи всех участников игры перечислены в разд. "Использование Мастера защиты" этой главы.)*

Организация совместного доступа к данным и объектам

В сети доступ к одним и тем же объектам могут одновременно получить сразу несколько пользователей. Общий доступ к данным отличается от монопольного. Например, совместная правка записей в таблице/запросе базы данных может привести к возникновению конфликтов при сохранении изменений в источнике. Изменение структуры объекта базы данных доступно только в режиме монопольного доступа. В Microsoft Access существуют стандартные механизмы для поддержки совместного доступа к данным и объектам базы данных в сети и разрешения конфликтов, возникающих при сохранении изменений.

Общий доступ к базе данных означает:

- совместный просмотр и изменение данных;
- совместный просмотр и изменение объектов.

Совместный просмотр и изменение данных

Предположим, база данных открыта в режиме общего доступа несколькими пользователями в сети. При этом несколько пользователей могут одновременно работать с одним и тем же набором записей. При попытке изменить определенную запись, в то время как ее редактирует другой пользователь или она удалена другим пользователем, возникает конфликт совместного доступа к источнику данных.

При совместном доступе к базе данных Microsoft Access помогает отследить состояние записи во время ее правки и позволяет убедиться в том, что используемые данные были обновлены. В тот момент, когда несколько пользователей пытаются одновременно изменить одну и ту же запись, Access отображает сообщение, позволяющее разрешить возникший конфликт. Например, при попытке сохранить изменения в записи, заблокированной в этот момент другим пользователем, отображается сообщение с именем этого пользователя.

Access помогает отследить состояние записи, отображая специальный символ в области выделения текущей записи. Возможные символы приведены в табл. 16.1.

Таблица 16.1. *Значок, обозначающий состояния текущей записи*

Символ	Значение
	Запись является текущей, и в данный момент еще не была изменена
	Запись изменена, но изменения еще не сохранены в источнике данных. В этом случае другие пользователи не видят текущих изменений в данной записи. Если запись заблокирована, другие пользователи не смогут ее изменить. Для того чтобы освободить запись и сделать ее доступной для других пользователей, следует сохранить или отменить внесенные изменения

Таблица 16.1 (окончание)

Символ	Значение
	Данная запись заблокирована другим пользователем. Ее нельзя изменить. При попытке внести изменения выдается звуковой сигнал

Методы управления блокировками записей

Пока один пользователь редактирует запись, Access позволяет автоматически запретить другим пользователям изменять ее до тех пор, пока первый пользователь не закончит ее правку. Предоставление одному пользователю исключительного права доступа к записи называется *блокировкой*.

Существует три параметра блокировки записей в базе данных Access:

- **отсутствует** (No Locks) — изменяемая запись не блокируется. При попытке сохранить измененную запись, которую в это же время изменял другой пользователь, на экране появляется сообщение, предлагающее следующие возможности: сохранить данный вариант записи, уничтожив изменения, внесенные другим пользователем; скопировать свою измененную запись в буфер обмена; отказаться от собственных изменений. Этот параметр позволяет изменять записи в любое время, но при внесении изменений разными пользователями могут возникать конфликты;
- **изменяемой записи** (Edited Records) — Microsoft Access блокирует изменяемую в данный момент запись, не позволяя изменять ее другим пользователям. Заблокированными могут оказаться другие записи, расположенные рядом на диске. Если другой пользователь пытается изменить заблокированную запись, в его объекте в режиме Таблицы появляется маркер заблокированной записи (перечеркнутый круг — третий значок в табл. 16.1). Данный параметр гарантирует завершение всех начатых изменений. Это хороший выбор, если конфликты совместного доступа к записи с другими пользователями возникают редко;
- **всех записей** (All Records) — Microsoft Access блокирует все записи формы или объекта в режиме Таблицы (и базовых таблиц) на весь период их открытия, поэтому другие пользователи не могут изменить или заблокировать записи. Этот параметр накладывает слишком жесткие ограничения. Его следует использовать, только в тех случаях, когда есть уверенность, что именно вы имеете приоритетное право на изменение записей.

Замечание

При редактировании данных из присоединенных таблиц SQL Server через ODBC применяются параметры блокировки записей этого сервера, а установленный параметр блокировки в Access игнорируется, будто выбран параметр "отсутствие блокировки".

Чтобы установить параметр блокировки записей в сети, используемый по умолчанию:

1. Выберите команду **Сервис | Параметры** (Tools | Options). Появится диалоговое окно **Параметры** (Options).

2. Раскройте вкладку **Другие** (Advanced) (рис. 16.3).
3. В группе **Блокировка по умолчанию** (Default record locking) выберите необходимый переключатель.

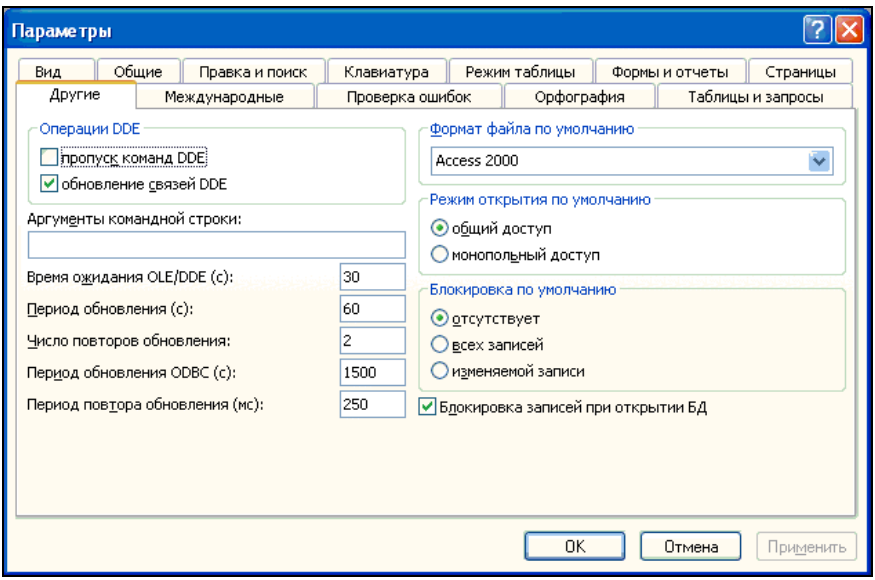


Рис. 16.3. Вкладка **Другие** диалогового окна **Параметры**

Существует возможность выбрать один из двух уровней блокировки:

- *блокировка записей* — блокируется только редактируемая запись;
- *блокировка страниц* — блокируется страница памяти на диске, которая может содержать несколько соседних записей (4 Кбайта памяти на страницу).

Чтобы задать уровень блокировки, используемый по умолчанию, установите флажок **Блокировка записей при открытии БД** (Open databases using record-level locking) на вкладке **Другие** (Advanced) диалогового окна **Параметры** (Options), если нужна блокировка на уровне записей, или сбросьте этот флажок, если нужна блокировка на уровне страниц.

Замечание

Установленный параметр начнет действовать, когда база данных будет открыта заново с помощью команды **Файл | Открыть** (File | Open), а не с помощью списка последних использованных файлов.

Блокировка на уровне записей действует только при доступе к данным через формы, страницы таблиц и запросов, или с помощью объекта `Recordset`. Этот режим не влияет на выполнение запросов и инструкций SQL.

Предотвращение конфликтов при блокировке записей в сети

Следующие четыре параметра помогают избежать конфликтов при блокировке записей в сети:

- **Период обновления** (Refresh interval) — задает число секунд, по истечении которых Microsoft Access автоматически обновляет записи в объекте в режиме Таблицы или Формы. При обновлении текущего объекта в режиме Таблицы или Формы отображаются изменения в существующих записях, причем записи не сортируются заново, не добавляются новые записи и не убираются с экрана удаленные записи. Для того чтобы увидеть эти изменения, необходимо выполнить операцию обновления данных, например, с помощью комбинации клавиш <Shift>+ +<F9>.
- **Период повтора обновления** (Update retry interval) — задает число миллисекунд, по истечении которых Microsoft Access автоматически пытается сохранить измененную запись, заблокированную другим пользователем.
- **Число повторов обновления** (Number of update retries) — задает число попыток сохранения измененной записи, заблокированной другим пользователем.
- **Период обновления ODBC** (ODBC refresh interval) — задает интервал времени в секундах, после которого Microsoft Access автоматически обновляет записи, доступ к которым осуществляется по протоколу ODBC. Для того чтобы увидеть эти изменения, необходимо выполнить операцию обновления данных, например, с помощью комбинации клавиш <Shift>+<F9>.

Значения этих параметров устанавливаются в диалоговом окне **Параметры** (Options) на вкладке **Другие** (Advanced). Их подробное описание есть в справочной системе Access.

Отображение обновленных данных

Существует несколько способов отображения на экране обновленных данных, полученных из источника данных, совместно используемого в сети несколькими пользователями. Эти способы позволяют увидеть изменения, которые были выполнены другими пользователями, после того как отображенные на экране данные были получены из источника.

Обновление данных можно выполнить несколькими способами.

- Работая с таблицей, формой или отчетом в режиме Таблицы, Сводной таблицы или Сводной диаграммы или с формой в режиме Формы, выберите команду **Записи | Обновить** (Records | Refresh). В результате отображаются изменения в существующих записях. Однако созданные другими пользователями новые записи не добавляются и удаленные ими записи с экрана не убираются.
- Для периодического обновления данных в режиме Таблицы или Формы установите необходимые значения параметра **Период обновления** (Refresh interval) и прочих параметров, описанных в *разд. "Предотвращение конфликтов при блокировке записей в сети" этой главы.*

- Чтобы выполнить повторную загрузку данных из источника (работая с объектом базы данных в режиме Таблицы, Сводной таблицы, Сводной диаграммы или Формы), нажмите комбинацию клавиш <Shift>+<F9>.
- Для отображения обновленных данных в форме можно создать кнопку, вызывающую макрос, содержащий макрокоманду `ОбновитьОбъект (Refresh)`, или процедуру VBA, вызывающую метод `Refresh` объекта `Form`.
- Для повторной загрузки данных, отображаемых в элементе управления, создайте макрос, содержащий макрокоманду `Обновление (Requery)`, в качестве значения аргумента **Имя элемента** (Control Name) укажите имя обновляемого элемента управления. Чтобы обновить данные в форме, оставьте поле, задающее значение аргумента **Имя элемента** (Control Name), пустым.
- Для повторной загрузки записей, отображаемых в форме или элементе управления, добавьте в процедуру VBA, назначенную форме или элементу управления, вызов метода `Requery` объекта `Form` или `Control`.
- Для обновления данных, отображаемых в режиме Просмотра страницы, щелкните по странице доступа к данным правой кнопкой мыши и в контекстном меню выберите команду **Обновить** (Refresh) или нажмите клавишу <F5>. Если при этом не были сохранены изменения, появится сообщение о том, что внесенные изменения будут утеряны. Чтобы предотвратить потерю изменений в текущей записи, нажмите кнопку **Отмена** (Cancel). Если загрузка обновленных данных продолжается длительное время, нажмите клавишу <Esc>. На странице может быть расположен элемент управления **Сводная таблица** (PivotTable). Этот элемент имеет собственный метод `Refresh`.
- Чтобы обновить данные в объекте базы данных, открытом в режиме Сводной таблицы или Сводной диаграммы, нажмите кнопку **Обновить** (Refresh) на панели инструментов **Сводные таблицы** (PivotTable) или **Сводная диаграмма** (PivotChart).

Совместное изменение объектов

Внесение изменений в объекты базы данных при работе в сети характеризуется некоторыми особенностями.

- Хотя допускается изменение базы данных во время работы с ней других пользователей, лучше создать макеты большинства объектов до того, как другие пользователи получат возможность открыть базу данных в сети. Для того чтобы при внесении серьезных изменений в структуру базы данных запретить другим пользователям доступ к этой базе данных, следует при открытии базы данных выбрать режим **Монопольно** (Open Exclusive) в диалоговом окне **Открытие файла базы данных** (Open). Если же вы открыли базу данных в режиме общего доступа и начинаете изменять структуру данных, то при условии, что в этот момент вы — единственный пользователь этой базы данных, Access автоматически временно переводит вашу базу данных в монопольный режим. Так что другие пользователи не смогут открыть эту базу данных, пока вы не закончите работу с изменениями структуры объектов. После этого база данных автоматически возвращается в режим общего доступа. Но если в такой момент вы — не единственный пользователь базы данных, то Access разрешит внести лишь незначительные изменения в дизайн объектов, например, настройки печати.

- Если другой пользователь открыл таблицу или просматривает данные в запросах, формах или отчетах, для которых эта таблица является источником данных, изменять структуру этой таблицы нельзя. Появляется сообщение, что таблица доступна только для чтения.
- Если объекты зависят друг от друга, обновляйте их все одновременно, чтобы другие пользователи не начали работать с несогласованными версиями объектов. Например, при добавлении полей в форму следует до обновления формы добавить поля в базовый запрос.
- Чтобы внести изменения в запрос, форму или отчет, открытые другим пользователем, этот пользователь должен закрыть и снова открыть измененный объект, чтобы получить его обновленную версию. В модулях VBA другие пользователи не могут выполнить измененные процедуры `Function` или `Sub`, пока они не закроют и снова не откроют базу данных.
- Страницы доступа к данным при изменении ведут себя по-другому. Хотя для создания, переименования, перемещения и удаления страниц доступа к данным требуется открыть базу данных в режиме монопольного доступа (поскольку это затрагивает содержимое базы данных), изменение самой страницы не потребует этого, т. к. сама страница хранится отдельно от базы данных в файле HTML.

Совместное использование страниц доступа к данным

Страницы доступа к данным могут использоваться несколькими пользователями в сети. Страницы могут быть отображены с помощью программы просмотра Интернет или в сообщении электронной почты. Для этого источник данных для страницы — базу данных Access или SQL Server — требуется поместить в общую папку на рабочей станции или на сервере сети.

В приложении "Игра в доминирование" для отображения текущего состояния игрового поля в момент инициализации новой игры создается страница доступа к данным "Игровое поле", которая сохраняется в той же папке, что и приложение Сервер игры. Эта папка должна быть доступна в сети, чтобы игроки — пользователи приложения Клиент игры — могли подключиться к игре и просмотреть страницу "Игровое поле" с помощью программы Internet Explorer.

Если база данных, являющаяся источником данных для страницы, будет перемещена в другое место, необходимо соответствующим образом изменить параметры соединения страницы с источником данных. Для этого при работе со страницей в режиме Конструктора в окне **Список полей** (Field List) нажмите кнопку **Свойства подключения для страницы** (Page connection properties). В появившемся диалоговом окне на вкладке **Соединение** (Connection) введите в поле ввода, предназначенное для указания источника данных, имя и полный путь к базе данных в соответствии с соглашением об именах UNC. Укажите имя сетевого ресурса, а не имя подключенного сетевого диска, поскольку оно может измениться.

Имя ресурса в формате UNC состоит из 4-х частей: имени сервера ресурсов, имени общей папки на сервере, пути к ресурсу и имени ресурса:

```
\\servername\share\path\filename
```

Чтобы изменить путь к странице доступа к данным, хранящийся в ярлыке страницы в базе данных Access, в окне базы данных щелкните по ярлыку страницы доступа к данным правой кнопкой мыши и в контекстном меню выберите команду **Свойства** (Properties). Откроется окно свойств ярлыка, введите в поле ввода путь к файлу страницы (файлу HTML). В таких случаях рекомендуем указывать относительные пути. Например, для страниц, хранящихся в той же папке, что и база данных, укажите только имя файла страницы без абсолютного пути к ней. Например:

```
Игровое поле.htm
```

Хотя источником данных для страницы может быть только база данных Access или SQL Server, можно организовать доступ через страницу к данным других форматов, используя связанные таблицы в базах данных Access.

Замечание

Защита совместно используемых страниц доступа к данным описана в справочной системе Access.

Для максимальной защиты страницы доступа к данным поместите базу данных, которая является источником данных для нее, на тот же Web-сервер, на котором хранится эта страница.

Организация обмена данными между компонентами сетевого приложения

Существует несколько способов обмена данными между компонентами приложения по сети:

- с помощью связанных таблиц;
- с помощью запросов;
- с помощью программирования на VBA.

Использование запросов и связанных таблиц подробно рассмотрено в других главах этой книги. Мы остановимся на программном доступе к базам данных с использованием интерфейса DAO.

(Информацию об установлении связи с таблицами см. в разд. "Использование баз данных других форматов" гл. 3, об использовании запросов см. гл. 8.)

Программирование с использованием интерфейса DAO

Объекты доступа к данным (Data Access Object, DAO) — это иерархия объектов, обеспечивающая доступ к структуре базы данных и ее содержимому. В программах Visual Basic пользователь имеет возможность использовать объектный интерфейс DAO для выполнения следующих задач:

- доступ к данным в локальных и удаленных базах данных Access и внешних источниках;

- управление базой данных и ее объектами;
- изменение структуры объектов данных;
- изменение схемы данных;
- управление защитой;
- управление репликами.

Интерфейс DAO использует рабочие области двух видов:

- *рабочая область ядра Jet* — обеспечивает доступ к базам данных процессора обработки данных Jet, источникам данных ODBC и источникам данных ISAM для доступа к базам данных в других форматах (например, к базам данных Lotus 1-2-3). Рабочая область Jet используется для доступа к источникам данных с помощью процессора Jet;
- *рабочая область ODBCDirect* — обеспечивает доступ к источникам данных ODBC напрямую, не используя процессор Jet.

Для рабочих областей Jet и ODBCDirect существуют две разные объектные модели DAO. Они приведены в *гл. 13*.

Реализация модулей VBA для обмена данными между компонентами приложения "Игра в доминирование" основана на применении интерфейса DAO с использованием рабочей области ядра Jet. Поэтому в дальнейших примерах основное внимание уделено приемам программирования на DAO, применимым в рабочей области ядра Jet. Сведения о рабочей области ODBCDirect можно найти в справочной системе Access.

В следующих разделах мы рассмотрим некоторые аспекты программирования с использованием DAO. Подробную информацию об интерфейсе DAO можно найти в справочной системе Access.

Внимание!

Чтобы использовать в процедурах VBA объекты доступа к данным, необходимо подключить библиотеку **Microsoft DAO 3.6 Object Library** и отключить библиотеку **Microsoft ActiveX Data Objects 2.1 Library** либо поместить ее ниже библиотеки объектов DAO (т. е. сделать ее менее приоритетной). *(О подключении библиотек к программному проекту рассказано в гл. 13.)*

Создание и использование рабочей области

Обе иерархии объектов DAO — для рабочей области Jet и для рабочей области ODBCDirect — начинаются с объекта `DBEngine`. Этот объект содержит свойства и методы, позволяющие управлять рабочими областями. Свойство `DefaultType` объекта `DBEngine` позволяет определить или установить тип рабочей области, создаваемой по умолчанию.

Объект `DBEngine` в качестве свойства содержит семейство `Workspaces` всех открытых рабочих областей. Можно выбрать элемент этого семейства, указав индекс или имя, чтобы получить доступ к конкретной рабочей области (листинг 16.1).

Чтобы создать новую рабочую область ядра Jet или рабочую область ODBCDirect, используется метод `CreateWorkspace` (листинг 16.2) объекта `DBEngine` с соответст-

вующим параметром. Метод `CreateWorkspace` возвращает ссылку на объект типа `Workspace` и имеет следующие параметры (они описаны в табл. 16.2):

```
Workspace CreateWorkspace(<имя>, <пользователь>, <пароль>, <тип>)
```

Таблица 16.2. *Параметры метода* `CreateWorkspace`

Параметр	Тип	Описание
`<имя>`	`String`	Уникальное имя создаваемой рабочей области. Определяет значение свойства `Name` объекта `Workspace`
`<пользователь>`	`String`	Имя пользователя, который будет владельцем создаваемой рабочей области. Определяет значение свойства `UserName` объекта `Workspace`
`<пароль>`	`String`	Пароль пользователя. Он необходим для создания рабочей области. Пароль может включать до 14 символов. Это могут быть любые символы, кроме символа ASCII с кодом 0 (Null). Определяет значение свойства `Password` объекта `Workspace`
`<тип>`	`<константа>`	Необязательный параметр. Задает тип создаваемой рабочей области. В качестве значения можно использовать константу `dbUseJet` для создания рабочей области ядра Jet или константу `dbUseODBC` для создания рабочей области ODBCDirect. По умолчанию создается рабочая область того типа, который задан значением свойства `DefaultType` объекта `DBEngine`

Несколько рабочих областей разных типов могут быть открыты одновременно. В момент создания рабочей области начинается сеанс работы с ней (session). После выполнения всех необходимых действий в рабочей области нужно завершить сеанс, закрыв рабочую область с помощью метода `Close` объекта `Workspace`.

Листинг 16.1. Использование рабочей области Jet, открытой по умолчанию

```
Dim ws As Workspace
Dim db As Database

Set ws = DBEngine.Workspaces(0)      ' Выбрали рабочую область
' Открываем базу данных:
Set db = ws.OpenDatabase("DominationGameServer.mdb")

' Код использования базы данных ...

db.Close                             ' Закрыли базу данных
Set db = Nothing                     ' Очистили объектную переменную
```

```
' (!) Плохой код:
'     рабочая область, открытая по умолчанию, будет закрыта
' ws.Close
' Set ws = Nothing
```

Листинг 16.2. Создание рабочей области Jet

```
Dim ws As Workspace
Dim db As Database

' Создаем рабочую область:
Set ws = CreateWorkspace("", "Флинт", "пиастры", dbUseJet)
' Открываем базу данных:
Set db = ws.OpenDatabase("DominationGameServer.mdb")

' Код использования базы данных ...

db.Close                              ' Закрыли базу данных
ws.Close                              ' Закрыли рабочую область
Set db = Nothing                      ' Очищаем объектные
Set ws = Nothing                      ' переменные
```

Открытие источника данных

Чтобы открыть базу данных, используйте существующий объект `Database` или создайте новый. Объект `Database` представляет собой базу данных Jet (файл MDB), базу данных ISAM или источник данных ODBC, подключенный через Jet.

Доступ к текущей базе данных осуществляется с помощью объекта типа `Database`, возвращаемого методом `CurrentDb` объекта `Application` (который представляет приложение Access). Метод `CurrentDb` входит в набор глобальных методов, поэтому для его вызова можно использовать сокращенную ссылку без префикса `Application` с точкой (листинг 16.3).

Открыть существующую базу данных можно двумя способами:

- с помощью метода `OpenDatabase` объекта `Workspace`. В этом случае база данных будет открыта в заданной рабочей области;
- с помощью метода `OpenDatabase` объекта `DBEngine`. В этом случае база данных будет открыта в рабочей области, используемой по умолчанию.

Метод `OpenDatabase` объекта `DBEngine` входит в набор глобальных методов, поэтому при использовании сокращенной ссылки на этот метод без явного указания объекта (`DBEngine` или `Workspace`) используется метод объекта `DBEngine`. Метод `OpenDatabase` возвращает ссылку на созданный объект `Database` и имеет следующие параметры (они описаны в табл. 16.3):

```
Database OpenDatabase(<имя>, <параметры>, <режим>, <соединение>)
```

Таблица 16.3. *Параметры метода* `OpenDatabase`

Параметр	Тип	Обязательный или нет	Описание
`<имя>`	`String`	Обязательный	Имя файла существующей базы данных (возможно, включая полный путь с указанием имени диска или сетевого ресурса) или имя источника данных ODBC (DSN)
`<параметры>`	`Variant`	Необязательный	Используется для задания специальных параметров базы данных. Например, в рабочей области Jet используется значение `True`, если нужно открыть базу данных в режиме монопольного доступа, или значение `False`, если нужно открыть базу данных в режиме общего доступа. Значение `False` используется по умолчанию. Параметры рабочей области ODBCDirect описаны в справочной системе Access
`<режим>`	`Variant` (подтип `Boolean`)	Необязательный	Задается значение `True`, если нужно открыть базу данных только для чтения, или значение `False`, если нужно открыть базу данных для чтения и записи. По умолчанию используется значение `False`
`<соединение>`	`Variant` (подтип `String`)	Необязательный	Строка соединения (connection string). Используется для указания параметров соединения с источником данных, включая пароль

Аналогично, создать и открыть новую базу данных можно:

- с помощью метода `CreateDatabase` объекта `Workspace`;
- с помощью метода `CreateDatabase` объекта `DBEngine`.

Глобальным является метод `CreateDatabase` объекта `DBEngine`.

Метод `CreateDatabase` создает новый объект `Database`, добавляет его в семейство `Databases` открытых баз данных в рабочей области, сохраняет базу данных на диске и возвращает открытый объект `Database`. Этот метод используется только в рабочей области ядра Microsoft Jet. Метод `CreateDatabase` имеет следующие параметры (они описаны в табл. 16.4):

```
Database CreateDatabase(<имя>, <порядок>, <параметры>)
```

Замечание

В программе на VBA для ссылки на метод `CreateDatabase` объекта `DBEngine` можно использовать сокращенную ссылку или указать префикс `DBEngine` с точкой.

А для ссылки на метод объекта Workspace нужно указать конкретную рабочую область, например:

```
' Использование метода объекта DBEngine:
Set db1 = CreateDatabase(...)
Set db2 = DBEngine.CreateDatabase(...)

' Использование метода объекта Workspace:
Set db3 = Workspaces(0).CreateDatabase(...)
```

Таблица 16.4. *Параметры метода* `CreateDatabase`

Параметр	Тип	Обязательный или нет	Описание
<имя>	String	Обязательный	Имя файла создаваемой базы данных. Следует указать полный путь и имя файла, например "C:\db1.mdb" или "\\server1\share1\dir1\db1". Если пользователь не указывает расширение имени, автоматически добавляется расширение mdb. Данный метод позволяет создавать только MDB-файлы
<порядок>	Variant (подтип String)	Обязательный	Строковое выражение, определяющее порядок символов, который будет использоваться в операциях сравнения и сортировки в создаваемой базе данных. Допускается также создание пароля для нового объекта Database путем слияния строки пароля (начинающейся с символов ;pwd=) с константой в аргументе <порядок>, например: `DbLangCyrillic & ";pwd=МойПароль"`
<параметры>	<константы>	Необязательный	Константа или комбинация констант, которая определяет один или несколько параметров: версию формата данных и режим шифрования или дешифрации базы данных во время сжатия. Константы приведены в справке Access

Чтобы получить доступ к данным в открытой одним из перечисленных способов базе данных, необходимо открыть набор записей. Он может представлять собой все записи таблицы или часть записей таблицы, удовлетворяющих указанному условию, или результат выборки из нескольких таблиц. Чтобы открыть набор записей в базе данных, используйте метод OpenRecordset объекта Database (см. листинг 16.3). Этот метод возвращает ссылку на созданный объект Recordset и имеет следующие параметры (они описаны в табл. 16.5):

```
Recordset OpenRecordset(<источник>, <тип>, <параметры>,<блокировки>)
```

Замечание

С помощью метода `OpenRecordset` в базе данных можно открыть связанную таблицу (точно так же, как и таблицу, хранящуюся в базе данных), указав ее имя.

Таблица 16.5. *Параметры метода* `OpenRecordset`

Параметр	Тип	Обязательный или нет	Описание
`<источник>`	`String`	Обязательный	Источник записей для нового объекта `Recordset`. В качестве источника записей можно указать имя таблицы или запроса, а также инструкцию SQL, которая возвращает записи
`<тип>`	`<константа>`	Необязательный	Константа, указывающая тип открываемого объекта `Recordset`. Эти константы приведены в табл. 16.6
`<параметры>`	`<константы>`	Необязательный	Произвольная комбинация определенных констант, задающих характеристики нового объекта `Recordset`. Эти константы приведены в справочной системе Access
`<блокировки>`	`<константа>`	Необязательный	Константа, определяющая тип блокировки объекта `Recordset`. Возможные константы перечислены в справке Access

Таблица 16.6. *Константы, определяющие тип набора записей*

Константа	Тип набора записей
`DbOpenTable`	Открытие табличного объекта `Recordset` (только в рабочей области ядра Microsoft Jet)
`DbOpenDynamic`	Открытие объекта `Recordset` динамического типа, аналогичного динамическому указателю ODBC (только в рабочей области ODBCDirect)
`DbOpenDynaset`	Открытие объекта `Recordset` типа динамического набора записей, аналогичного указателю ключевого набора записей ODBC
`DbOpenSnapshot`	Открытие объекта `Recordset` типа статического набора записей, аналогичного указателю статического набора записей ODBC
`DbOpenForwardOnly`	Открытие объекта `Recordset` типа статического набора записей с последовательным доступом

Далее приведен пример открытия таблицы, взятый из программного кода сервера приложения "Игра в доминирование" (листинг 16.3).

Листинг 16.3. Открытие таблицы в текущей базе данных

```
Dim db As Database
Dim rs As Recordset

' Получаем доступ к текущей базе данных:
Set db = CurrentDb

' Открываем таблицу сообщений для игрока (это связанная таблица):
Set rs = db.OpenRecordset("Сообщения", dbOpenDynaset)

' Код использования таблицы ...

rs.Close                        ' Закрыли таблицу
Set rs = Nothing                ' Очистили объектные переменные
Set db = Nothing
```

Программное изменение структуры и схемы данных

Для изменения схемы данных создан *язык определения данных* (Data-Definition Language, DDL). Инструкции на языке DDL позволяют выполнять действия по изменению схемы данных и структуры объектов данных, например:

- создавать и удалять таблицы;
- добавлять и удалять из таблиц поля и индексы;
- создавать и удалять связи между таблицами.

В интерфейсе DAO имеется набор специальных объектов, который является интерфейсом для доступа к средствам DDL. Таким образом, с помощью объектов DAO можно управлять структурой таблиц и схемой данных, не составляя самих инструкций на языке DDL.

Следующий пример программного кода (листинг 16.4) взят из процедуры `initGame` сервера приложения "Игра в доминирование". Этот пример иллюстрирует использование инструкций на языке DDL для удаления таблицы со старой структурой и создания новой таблицы с заданной структурой. Аналогичные действия можно выполнить, используя объектно-ориентированный интерфейс DAO. Целью этой процедуры является изменение структуры таблицы `ПолеИгрок`, поэтому, вообще говоря, можно было бы изменить набор полей этой таблицы с помощью инструкции DDL `ALTER TABLE`. Эта инструкция позволяет удалить или добавить в таблицу поле или составной индекс. Удаление и добавление полей в таблицу средствами DDL через интерфейс DAO представлено в листинге 16.5.

Замечание

Необходимо помнить о том, что изменение структуры связанных таблиц запрещено. Поэтому в примере (листинги 16.4 и 16.5) для доступа к таблице `ПолеИгрок` открывается база данных DominationGame.mdb, в которой сохранена эта таблица, а не используется текущая база данных DominationGameServer.mdb, в которой установлена связь с этой таблицей.

Листинг 16.4. Удаление и создание таблицы с помощью инструкций DDL

```
Dim db As Database
Dim fieldSize As Long, i As Long
Dim strDDL As String

' Узнаем линейный размер игрового поля, выраженный в клетках
  fieldSize = CLng(get_parameter("РазмерПоля"))

' Открываем базу данных, в которой хранится нужная таблица
  Set db = OpenDatabase(CurrentProject.Path & "\DominationGame.mdb")

' Удалить старую таблицу "ПолеИгрок"
  strDDL = "DROP TABLE ПолеИгрок;"

' Если таблица не существует, удаление вызовет ошибку
  On Error GoTo the_next_2
  db.Execute strDDL
the_next_2:
  On Error GoTo 0

' Создать новую таблицу "ПолеИгрок"
  strDDL = "CREATE TABLE ПолеИгрок ("
  For i = 1 To fieldSize - 1
      strDDL = strDDL & get_column_name(i) & " TEXT(20), "
  Next i
  strDDL = strDDL & get_column_name(fieldSize) & " TEXT(20));"
  db.Execute strDDL

' Прочие действия ...

  db.Close
```

Листинг 16.5. Изменение структуры таблицы с помощью интерфейса DAO

```
Dim db As Database
Dim fieldSize As Long, i As Long
Dim strSQL As String
Dim fld As Field

' Узнаем линейный размер игрового поля, выраженный в клетках
  fieldSize = CLng(get_parameter("РазмерПоля"))

' Открываем базу данных, в которой хранится нужная таблица
  Set db = OpenDatabase(CurrentProject.Path & "\DominationGame.mdb")

' Удалить старые записи в таблице "ПолеИгрок"
  strSQL = "DELETE * FROM ПолеИгрок;"
  db.Execute strSQL
```

```
' Удалить старые поля в таблице "ПолеИгрок"
  For i = 1 To db.TableDefs("ПолеИгрок").Fields.Count
      db.TableDefs("ПолеИгрок").Fields.Delete get_column_name(i)
  Next i

' Создать новые поля в таблице "ПолеИгрок"
  For i = 1 To fieldSize
      Set fld = db.TableDefs("ПолеИгрок").CreateField( _
                                    get_column_name(i), dbText, 20)
      db.TableDefs("ПолеИгрок").Fields.Append fld
  Next I

' Прочие действия ...

  db.Close
```

В листинге 16.5 проиллюстрировано удаление и добавление полей в таблицу `ПолеИгрок` средствами DDL с использованием интерфейса DAO. Удаление поля производится с помощью метода `Delete`, в качестве параметра которого указывается имя удаляемого поля. Добавление нового поля производится следующим образом: создается новый объект `Field`, обладающий необходимыми характеристиками — заданным именем поля, типом данных и размером данных; после этого с помощью метода `Append` созданный объект `Field` добавляется в семейство `Fields` объекта `TableDef`, содержащее все поля таблицы `ПолеИгрок`. Аналогично (с помощью DAO) можно выполнить создание и удаление таблиц и индексов.

Выполнение операций с данными

Для изменения данных в источнике создан *язык структурированных запросов* (Structure Query Language, SQL). Инструкции на языке SQL позволяют выполнять действия по изменению данных, например — добавить, изменить или удалить запись в таблице или запросе, выбрать набор записей из источника, удовлетворяющий заданному условию.

С помощью объектов DAO можно выполнять всевозможные манипуляции с данными:

- получить набор записей из таблицы;
- выполнить запрос SQL для получения набора записей;
- использовать запрос, сохраненный в базе данных в виде объекта;
- использовать набор методов объекта `Recordset`, предоставляющий широкие возможности по обработке данных: чтение, обработку и изменение данных, не составляя инструкций на языке SQL;
- прочие действия.

В следующих примерах приведены приемы программного изменения данных в открытом наборе записей (листинги 16.6—16.8). Переменная `rs` соответствует открытому набору записей — объекту типа `Recordset`.

Листинг 16.6. Добавление записи в таблицу

```
' Добавляем сообщение в таблицу сообщений клиента
rs.AddNew                        ' Создание новой записи
rs!ИмяИгрока = playerName        ' Запись значения в поле ИмяИгрока
rs!Сообщение = message           ' Запись значения в поле Сообщение
rs.Update                        ' Сохранение изменений в источнике
rs.Bookmark = rs.LastModified    ' Перемещение курсора на новую запись
```

Листинг 16.7. Изменение текущей записи в таблице

```
' Увеличиваем счет игрока, сделавшего ход
' Делаем текущей запись, содержащую данные для нужного игрока
rs.FindFirst "[ИмяИгрока] = '" & newPlayer & "'"
rs.Edit                          ' Переводим запись в режим правки
rs!Счет = rs!Счет + newCount     ' Изменяем значение поля Счет
rs.Update                        ' Сохраняем изменения
rs.Bookmark = rs.LastModified    ' Делаем измененную запись текущей
```

Листинг 16.8. Удаление текущей записи в таблице

```
playerQueryCode = rs!КодЗаявки ' Сохраняем параметры заявки
playerTrial = rs!Значение      ' во временных переменных
rs.Delete                      ' и удаляем заявку из таблицы
```

Для перемещения по записям используются методы `Move`, `MoveFirst`, `MoveNext`, `MovePrev`, `MoveLast` объекта `Recordset`. Метод `MoveLast` перемещает курсор на последнюю запись в наборе, это приводит к загрузке в набор всех записей. После этого можно прочитать значение свойства `Count` объекта `Recordset`. Оно будет соответствовать общему количеству записей в наборе.

Замечание

Удаление записей и объектов из базы данных приводит к тому, что файл базы данных становится фрагментированным и место, занимаемое им на диске, используется нерационально. Чтобы дефрагментировать файл базы данных, используется процедура сжатия. Ее можно выполнить с помощью команды **Сервис | Служебные программы | Сжать и восстановить базу данных** (Tools | Database Utilities | Compact and Repair Database) или программно — с помощью метода `CompactDatabase` объекта `DBEngine`. Можно также установить флажок **Сжимать при закрытии** (Compact on Close) в диалоговом окне **Параметры** (Options) (команда **Сервис | Параметры** (Tools | Options), вкладка **Общие** (General)), чтобы сжатие базы данных проводилось автоматически при ее закрытии. Подробнее процедура сжатия описана в *гл. 20*.

Программная синхронизация доступа к данным

При одновременном доступе нескольких пользователей к одной и той же записи в источнике данных могут возникнуть конфликты. Например, конфликты могут воз-

никать при выполнении метода `Update` или `Delete` объекта `Recordset` и метода `OpenDatabase` объекта `Workspace` или `DBEngine`. Предотвращение конфликтов зависит от настройки Access, касающейся методов блокировки записей, режима доступа к базе данных, периодов обновления данных. Об установке этих параметров мы уже говорили в *разд. "Организация совместного доступа к данным и объектам" этой главы.* Подробную информацию можно найти в справочной системе Access.

При программном доступе к совместно используемым базам данных необходимо организовать собственную обработку ошибок, возникающих при попытке выполнить операции, которые могут привести к возникновению конфликтов. В приложении "Игра в доминирование" используется метод блокировки на уровне записей, блокируются изменяемые записи. В этом случае при возникновении конфликта самым простым решением является ожидание момента, когда заблокированная запись будет освобождена другим пользователем. Этот подход достаточно просто реализовать в процедуре на VBA, используя оператор `Resume` для повторного выполнения действия через некоторый интервал времени (сделав паузу). Назовем этот способ синхронизацией доступа к данным.

Рассмотрим в качестве примера синхронизации функцию отправки сервером сообщения одному из игроков (листинг 16.9). Вспомогательная функция `doPause`, позволяющая сделать паузу в работе приложения на заданное количество секунд, приведена в листинге 16.10.

Листинг 16.9. Синхронизация записи изменений в источнике данных

```
' Послать сообщение подключенному игроку
Public Sub SendMessage(message As String, playerName As String)
    ' Объявления локальных объектных переменных:
    Dim db As Database
    Dim rs As Recordset
    Dim counter As Integer

    ' Определяем собственную обработку ошибок:
    On Error GoTo errHandler
    Set db = CurrentDb

    ' Открываем таблицу сообщений для игрока:
    Set rs = db.OpenRecordset("Сообщения", dbOpenDynaset)
    On Error GoTo 0

    ' Добавляем сообщение в таблицу сообщений клиента:
    rs.AddNew
    rs!ИмяИгрока = playerName
    rs!Сообщение = message

    ' Обработка ошибок, возникающих при совместном доступе
    ' к источнику данных:
    On Error GoTo tryAgain
    rs.Update
    On Error GoTo 0
```

```
    rs.Bookmark = rs.LastModified

    ' Закрываем открытые объекты:
closeAllHandler:
    rs.Close
endHandler:
    ' Очищаем объектные переменные, т. к. они больше не используются:
    Set rs = Nothing
    Set db = Nothing
    ' Завершаем работу:
    Exit Sub

' Синхронизация:
tryAgain:
    counter = counter + 1
    If counter < 400 Then
        doPause 5
        Resume
    End If

' Обработка ошибок:
errHandler:
    Dim errMsg As String
    errMsg = "Ошибка: " & Err.Number & vbCrLf & _
             "Источник: " & Err.Source & vbCrLf & _
             vbCrLf & Err.Description
    MsgBox errMsg, vbCritical, ERR_TITLE
    Resume endHandler

End Sub
```

Листинг 16.10. Сделать паузу на заданное кол-во секунд в работе приложения

```
Public Sub doPause(seconds As Integer)
    Dim var_timeStart, var_timeCurrent
    Dim ftimeOut As Boolean
    var_timeStart = Time() ' Время начала паузы
    Do
        ' Передать управление другим процессам операционной системы
        DoEvents
        var_timeCurrent = Time() ' Текущее системное время
        ftimeOut = _
            CDate(var_timeCurrent - var_timeStart) >= _
            CDate(TimeSerial(0, 0, seconds))
    Loop Until ftimeOut
End Sub
```

Выполнение транзакций

Транзакцией называется операция обмена данными между клиентом и сервером. Методы объекта `Workspace`, использующиеся для выполнения транзакций, представлены в табл. 16.7.

Таблица 16.7. *Методы объекта* `Workspace` *для выполнения транзакций*

Метод	Описание
`BeginTrans`	Обозначает начало транзакции. Транзакции могут быть вложенными
`CommitTrans`	Обозначает конец транзакции. В этот момент все изменения сохраняются в источнике без возможности их отмены
`RollBack`	Завершение транзакции и отмена результата ее выполнения. Записи в источнике возвращаются в прежнее состояние, которое было до начала транзакции, отмеченное командой `BeginTrans`

Все базы данных, открытые в рабочей области, имеют общую область действия транзакций. Это значит, что действие методов `BeginTrans`, `CommitTrans` и `RollBack` объекта `Workspace` распространяется на все базы данных в семействе `Databases` объекта `Workspace`.

Рассмотрим пример программы на VBA с использованием механизма выполнения транзакций (листинг 16.11). В приложении "Игра в доминирование" механизм транзакций не используется.

Листинг 16.11. Применение механизма транзакций

```
Sub ResetCount()
    Dim ws As Workspace
    Dim db As Database
    Dim rs As Recordset
    Dim fInTrans As Boolean

    On Error GoTo errHandler

    fInTrans = False                    ' Транзакция еще не началась
    Set ws = DBEngine.Workspaces(0)
    Set db = CurrentDb
    Set rs = db.OpenRecordset("Игроки", dbOpenTable)

    ws.BeginTrans                       ' Начало транзакции
    fInTrans = True                     ' Транзакция началась
    rs.MoveFirst
    Do Until rs.EOF
        rs.Edit
        rs!Счет = 0
```

```
        rs.Update
        rs.MoveNext
    Loop
    If MsgBox("Сохранить сделанные изменения?", _
              vbQuestion + vbYesNo, "Вопрос") = vbYes Then
        ws.CommitTrans                ' Сохранить изменения
    Else
        ws.Rollback                   ' Отменить изменения
    End If
exitHandle:
    rs.Close
    Set db = Nothing
    Set ws = Nothing
    Exit Sub
errHandler:
    MsgBox "Ошибка!"
    ' Если ошибка возникла в процессе выполнения транзакции,
    ' отменяем сделанные изменения
    If fInTrans Then
        ws.Rollback
    End If
    Resume exitHandle
End Sub
```

Защита сетевого приложения Access

Выбор политики защиты и установка защиты — предмет разработки приложения. А настройка схемы защиты приложения в соответствии с текущими потребностями в процессе использования приложения — задача *администрирования*. *Администратор* управляет составом пользователей и групп пользователей приложения и предоставляет им необходимые права доступа к объектам приложения по мере необходимости. *Разработчик* приложения предварительно определяет политику защиты приложения, выполняет установку защиты приложения и подготавливает приложение к использованию — определяет начальный состав пользователей и групп пользователей приложения и выполняет начальное распределение между ними прав доступа к объектам приложения. Таким образом, в процессе установки защиты приложения разработчик помимо прочих действий в определенный момент выполняет функции администратора.

Действия, которые выполняет администратор защищенного приложения, описаны в *гл. 20*. В этой главе мы расскажем о действиях, выполняемых разработчиком, связанных с установкой защиты и определением политики защиты приложения.

Определение политики защиты включает решение следующих вопросов:

- какова степень защиты приложения;
- по какому принципу производится выделение групп пользователей;
- по какому принципу организуется разграничение прав доступа между отдельными пользователями и группами;
- какие права доступа необходимо задать по умолчанию.

Например, разработчик может решить, предоставить ли разрешение другим пользователям просматривать или выполнять запрос без изменения данных или структуры запроса, задать ли права по умолчанию для всех запросов.

Администратор приложения обладает всеми необходимыми правами для изменения политики защиты приложения, но необходимости в этом не должно возникать. Ведь в задачу разработчика приложения входит выбор наиболее подходящей политики защиты, которая удовлетворяет требованиям, предъявляемым к приложению, и упрощает процесс администрирования.

Защита базы данных на уровне пользователей

Защита на уровне пользователей предназначена для предоставления различным пользователям разного уровня доступа к объектам приложения. Пользователи могут объединяться в группы внутри рабочей группы. Группам, как и пользователям, могут быть назначены определенные права доступа к объектам базы данных.

Существует два типа прав доступа, предоставляемых пользователю: явные и неявные. *Явные права доступа* — те, что назначены непосредственной учетной записи пользователя. *Неявные права доступа* — те, что назначены учетной записи группы, в которую входит этот пользователь. Добавление пользователя в группу приводит к предоставлению пользователю прав, назначенных группе. Удаление пользователя из группы лишает пользователя этих прав.

Права пользователя на доступ к объекту складываются из его явных и неявных прав. В системе защиты на уровне пользователей Microsoft Access, в отличие от системы защиты на уровне пользователей файловой системы NTFS, применяется политика "наименьших ограничений". Это означает следующее: если пользователю (который может входить в разные группы, обладающие различными правами доступа) назначены и разрешающие, и запрещающие права к некоторому объекту базы данных Access, то доступ к объекту пользователю предоставляется.

Наличие групп пользователей делает управление правами доступа более удобным. Вместо того чтобы всякий раз при введении нового пользователя в рабочую группу предоставлять ему все необходимые права к объектам базы данных, можно назначить эти права группе. Тогда процесс предоставления заданных прав доступа сведется к добавлению нового пользователя в эту группу.

Рабочая группа пользователей базы данных

Рабочей группой в Access называется группа пользователей сети, совместно использующих одну или несколько баз данных Access. Если база данных защищена на уровне пользователей, в файл рабочей группы (файл с расширением mdw) записываются учетные записи пользователей и групп, входящих в рабочую группу. Пароли пользователей также хранятся в файле рабочей группы. Учетным записям в рабочей группе могут быть назначены права доступа к базе данных и ее объектам — таблицам, запросам, формам, отчетам и макросам. Права доступа сохраняются в защищенной базе данных.

Замечание

В ранней версии Access 97 в файле рабочей группы сохранялась также информация о настройках Access, заданных пользователем в окне **Параметры** (Options). Теперь в Access (начиная с версии Access 2000) эти настройки сохраняются в реестре Windows. Для Access 2003 соответствующий раздел реестра называется:

`\HKEY_CURRENT_USER\Software\Microsoft\Office\11.0\Access\Settings`.

По умолчанию используется стандартный файл рабочей группы SYSTEM.MDW, автоматически созданный при установке Access. Этот файл находится в папке, где сохраняются личные настройки пользователя Access, например: C:\Documents and Settings\Veronica\Application Data\Microsoft\Access\. Создание нового файла рабочей группы и его подключение выполняется с помощью служебной программы **Администратор рабочих групп** (Workgroup Administrator) *(см. гл. 20)*.

При установке защиты на уровне пользователей, прежде чем создавать учетные записи пользователей и групп и назначать им права доступа к объектам базы данных, следует подключить соответствующий файл рабочей группы. Можно использовать стандартный файл рабочей группы или создать новый. Не рекомендуется использовать стандартный файл рабочей группы, поскольку любой другой пользователь сможет использовать собственный стандартный файл рабочей группы для доступа к базе данных. Чтобы обеспечить уникальность рабочей группы, необходимо создать с помощью программы **Администратор рабочих групп** (Workgroup Administrator) новый файл рабочей группы и указать уникальный идентификатор рабочей группы *(см. гл. 20)*.

Встроенные учетные записи Access

В любой рабочей группе пользователей базы данных Microsoft Access существуют стандартные учетные записи:

- пользователь `Admin`;
- группа `Admins`;
- группа `Users`.

Их нельзя удалить, но можно изменить права доступа, назначенные им по умолчанию.

Пользователь `Admin` по умолчанию включен и в группу `Users`, и в группу `Admins`. Любой добавляемый в рабочую группу пользователь автоматически включается в группу `Users`. Эта группа представляет собой всех пользователей Access. Никакой пользователь не может быть удален из группы `Users`.

Группа `Admins` представляет собой группу администраторов всех баз данных, используемых рабочей группой. Этой группе автоматически предоставляются полные права на доступ к этим базам данных. В каждый момент времени группа `Admins` должна содержать, по крайней мере, одну учетную запись пользователя.

Пользователь `Admin` представляет собой администратора всех баз данных Access. Однако, используя конкретный файл рабочей группы для защиты конкретной базы данных, пользователя `Admin` можно удалить из группы `Admins` и назначить ему и

группе Users ограниченные права доступа к объектам защищаемой базы данных. Таким образом, база данных становится защищенной от доступа с помощью учетной записи Admin, используемой по умолчанию.

Права доступа к объектам конкретной базы данных, назначенные пользователю Admin и группе Users, остаются прежними при подключении любого файла рабочей группы. А права доступа, назначенные группе Admins, зависят от файла рабочей группы, подключенного к базе данных в данный момент. Например, защита базы данных сервера приложения "Игра в доминирование" проведена при подключении специально созданного для этого файла рабочей группы DOMINATION.MDW. Группе Admins назначены максимальные права доступа ко всем объектам этой базы данных. Если открыть эту базу данных с использованием стандартного файла рабочей группы SYSTEM.MDW, у группы Admins не будет никаких прав на объекты этой базы данных.

Внимание!

При установке защиты необходимо помнить о сохранении прав доступа, назначенных встроенным учетным записям Admin и Users при подключении стандартного файла рабочей группы Access, чтобы сделать систему защиты базы данных неуязвимой.

Административные права доступа

Административные права предоставляют полный доступ к объектам и/или базе данных, а также разрешение на изменение прав доступа других пользователей и групп. Административными правами обладают:

- все пользователи группы Admins;
- владельцы объектов;
- пользователи, которым явно предоставлены административные права доступа.

Даже если в определенный момент пользователь не может выполнить некоторую операцию из-за отсутствия прав доступа, он может иметь разрешение назначить себе эти права. Такими пользователями являются пользователи группы Admins рабочей группы базы данных и владельцы объектов.

Только пользователи группы Admins рабочей группы базы данных могут изменять параметры защиты с помощью команд меню **Сервис | Защита** (Tools | Security) или с помощью Мастера защиты на уровне пользователей.

Право на владение объектами базы данных

Владельцем называется учетная запись пользователя, имеющего контроль над базой данных или ее объектом. По умолчанию владельцем объекта или базы данных является пользователь, создавший их (т. е. пользователь, зарегистрированный при открытии базы данных). Учетная запись группы не может быть владельцем базы данных, но может быть владельцем ее объекта. В этом случае все пользователи этой группы являются владельцами этого объекта. Владелец объекта или базы данных обладает исключительными правами, которых его нельзя лишить. Даже если ему не

предоставлены определенные права доступа, он может их вернуть, изменив права доступа к объекту или базе данных для себя и других пользователей. Владелец базы данных всегда может открыть ее.

Пользователи, которые имеют разрешение на изменение прав доступа к объекту, могут изменить его владельца с помощью диалогового окна **Разрешения** (User and Group Permissions), как описано в *гл. 20*, или воссоздав данный объект его копированием, импортом или экспортом в другую базу данных *(см. разд. "Предоставление права на владение объектами базы данных" в гл. 20)*.

Замечание

Копирование, импорт и экспорт запросов, у которых для свойства **При запуске предоставляются права** (Run Permissions) установлено значение **Владельца** (Owner's), не приводит к изменению их владельцев. Изменить владельца запроса можно только в том случае, если это свойство имеет значение **Пользователя** (User's).

Защита базы данных на уровне пользователя с помощью Мастера позволяет легко изменить владельца базы данных и ее объектов, поскольку при этом создается новая база данных, в которую импортируются все объекты из исходной базы данных.

Установка системы защиты на уровне пользователей

Рассмотрим систему защиты приложения "Игра в доминирование". Напомним, что приложение "Игра в доминирование" состоит из клиентской и серверной частей, реализованных в виде отдельных баз данных Access. В серверной базе данных хранится информация о текущем состоянии игры. Игроки, обращающиеся к серверу через клиентские приложения, имеют ограниченные права доступа к объектам серверной базы данных. Эти ограничения необходимы, чтобы игроки имели равные возможности в игре и не могли повлиять на состояние игры недопустимым образом. Поэтому нужна специальная защита серверной части приложения по схеме защиты на уровне пользователей. В то же время, нет особой необходимости в специальной защите клиентской части приложения. Можно разрешить любому пользователю сети воспользоваться приложением-клиентом, чтобы подключиться к "Игре в доминирование". Однако чтобы пользователь клиентского приложения мог зарегистрироваться на сервере, он должен иметь учетную запись в единой рабочей группе пользователей приложения. Таким образом, следует создать один файл рабочей группы, содержащий информацию обо всех участниках "Игры в доминирование", который необходимо использовать для работы как с серверной, так и с клиентской частью приложения. Так мы и поступили — создали файл DOMINATION.MDW.

Рассмотрим основные этапы установки защиты сетевого приложения Access на примере защиты приложения "Игра в доминирование".

Замечание

Прежде чем приступить к защите собственного приложения Access, сделайте копию системного файла SYSTEM.MDW и копию пока не защищенного приложения. Это необходимо, поскольку удивительно легко заблокировать приложение от себя самого. Системный файл рабочих групп, использующийся в Access по умолчанию, нахо-

дится в папке, относительный путь к которой указан в реестре Windows. Для Access 2003 соответствующий раздел реестра называется:

```
HKEY_CURRENT_USER\Software\Microsoft\Office\11.0\Access\Jet\ 4.0
\Engines\SystemDB
```

Процесс установки защиты на уровне пользователей для любой базы данных Access можно разделить на несколько этапов.

- Создание файла рабочей группы и подключение его к приложению Access.
- Назначение владельца базы данных и его объектов.
- Создание учетных записей групп и пользователей в рабочей группе.
- Назначение прав доступа к объектам базы данных пользователям и группам.

Для защиты очередного приложения не обязательно требуется создание нового файла рабочей группы. Хорошим стилем является создание и использование одного файла рабочей группы для ряда приложений Access, с которыми работает одна и та же группа пользователей, например, сотрудники одного отдела. С другой стороны в разных файлах рабочих групп можно создать группы и пользователей с одинаковыми именами и кодами. Тогда пользователь одной рабочей группы может использовать базу данных, созданную для другой рабочей группы, если он является членом групп с идентичным именем и личным кодом в обеих рабочих группах.

В следующих разделах описаны все действия, связанные с установкой защиты на уровне пользователей. Их можно выполнить самостоятельно средствами Access или с помощью Мастера защиты баз данных Access. Мастер предлагает пользователю выполнить последовательность шагов, соответствующих перечисленным этапам установки защиты. Мастер защиты очень удобен в использовании как для начинающих пользователей, так и для профессионалов, поэтому рассмотрим сначала способ установки защиты с помощью Мастера, а затем поговорим о прочих средствах Access, позволяющих настроить систему защиты базы данных на уровне пользователей.

Использование Мастера защиты

Процедура установки защиты базы данных Access на уровне пользователей автоматизирована, ее можно выполнить с помощью Мастера защиты базы данных. Воспользоваться этим Мастером могут только пользователи группы `Admins`.

Чтобы защитить базу данных на уровне пользователей с помощью Мастера:

1. Откройте базу данных, которую требуется защитить.
2. Выберите команду **Сервис | Защита | Мастер** (Tools | Security | User-Level Security Wizard).
3. Следуйте инструкциям Мастера.

Мастер защиты базы данных на уровне пользователей позволяет создать или изменить существующий файл рабочей группы (рис. 16.4). Рекомендуется создать новый файл рабочей группы и не использовать системный файл рабочей группы SYSTEM.MDW. Чтобы изменить существующий файл рабочей группы (для защиты базы данных), необходимо зарегистрироваться с именем любого пользователя (кроме пользователя `Admin`), обладающего административными правами доступа к базе

данных. Если зарегистрироваться с именем пользователя `Admin`, который по умолчанию включен в группу `Admins` и обладает административными правами, с помощью Мастера защиты можно будет только создать новый файл рабочей группы и нельзя будет изменить существующий.

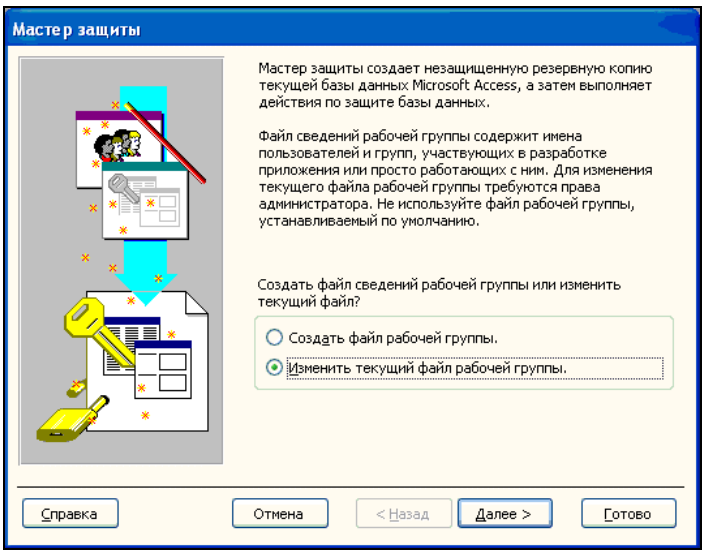


Рис. 16.4. Первое диалоговое окно Мастера защиты базы данных

Мастер защиты позволяет:

- выбрать объекты базы данных, которые требуется защитить;
- указать пароль для защиты программного кода на VBA;
- выбрать группы из списка групп с типичными правами доступа;
- предоставить ограниченные права доступа группе `Users`;
- создать учетные записи пользователей;
- определить вхождение пользователей в группы.

Мастер создает новую защищенную базу данных и импортирует в нее все объекты исходной базы данных, сохраняя резервную копию незащищенной базы данных с прежним именем (по умолчанию), но с расширением bak вместо mdb. После завершения работы Мастера на экране отображается отчет о защите базы данных, содержащий всю указанную в окнах Мастера информацию, необходимую для восстановления файла рабочей группы. Этот отчет рекомендуется распечатать или сохранить в формате снимка отчета (snapshot), чтобы по сохраненной информации можно было бы восстановить файл рабочей группы в случае его повреждения. *(См. разд. "Создание и восстановление файла рабочей группы" гл. 20 и разд. "Снимки отчетов" гл. 6.)*

С помощью Мастера в файл рабочей группы DOMINATION.MDW, созданный для "Игры в доминирование", были добавлены учетные записи пользователей, перечисленные в табл. 16.8.

Таблица 16.8. *Пользователи приложения "Игра в доминирование"*

Имя пользователя	Пароль	Код пользователя	Группы, в которые входит пользователь
Флинт	пиастры	Adkj7uLn6l84FTAwopW	Admins, Users
Джим	без пароля	jAvO6BLSS2d6KnN5X8O	Users
Ливси	без пароля	4yF3yNprLXhlzmS3iBC	Users
Треллони	без пароля	qFnFceqtWpVTbWRnSOq	Users
Admin	game	<уже существовал>	Users

Мастер защищает все выбранные на определенном этапе объекты базы данных, отбирает у группы `Users` все разрешения на доступ к этим объектам либо оставляет этой группе заданные ограниченные права доступа, а затем зашифровывает базу данных. Межтабличные связи и любые связанные таблицы воссоздаются в защищенной базе данных.

С этого момента только пользователи, входящие в группу `Admins`, учетные записи которых сохранены в файле рабочей группы, созданном или присоединенном с помощью Мастера, будут иметь полный доступ к защищенным объектам в базе данных. А текущий пользователь, обладающий административными правами, становится владельцем базы данных и всех ее объектов. Для того чтобы управлять доступом к защищенным объектам, следует предоставить соответствующие разрешения учетным записям пользователей и/или групп.

Если понадобится изменить параметры защиты, Мастер защиты можно использовать неоднократно. Еще раз отметим, что для применения Мастера защиты необходимо обладать административными правами доступа к базе данных.

Создание и подключение файла рабочей группы

Подключение файла рабочей группы DOMINATION.MDW можно провести двумя способами: с помощью программы **Администратор рабочих групп** (Workgroup Administrator) или с помощью параметра командной строки `/wrkgrp`. В первом случае файл рабочей группы присоединяется к Access и используется для всех баз данных, открываемых в нем. Во втором случае файл рабочей группы присоединяется к конкретной базе данных и не используется для других баз данных, открываемых в Access. Второй способ намного удобнее, если требуется параллельно работать и с защищенными, и с незащищенными базами данных на одном компьютере, или если используются разные файлы рабочих групп для разных защищенных баз данных.

Создание и подключение файла рабочей группы DOMINATION.MDW описано в *гл. 20*.

Снятие защиты на уровне пользователей

Бывают ситуации, когда нужно снять защиту базы данных на уровне пользователей. Это требуется, например, чтобы сделать базу данных доступной всем, или если администрирование базы данных с течением времени стало затруднительным из-за

резкого увеличения числа пользователей и возникла необходимость перераспределить пользователей по группам и назначить им права доступа по новой схеме.

Снятие защиты на уровне пользователей может выполнить только пользователь, обладающий административными правами. Процедура снятия защиты на уровне пользователей включает два этапа:

- предоставление группе `Users` полных прав на доступ к объектам базы данных;
- изменение владельца базы данных — предоставление права владения пользователю `Admin`.

Итак, чтобы снять защиту базы данных на уровне пользователей:

1. Откройте защищенную базу данных.
2. Зарегистрируйтесь с именем пользователя, являющегося администратором рабочей группы (администраторы рабочей группы входят в группу `Admins`).
3. Предоставьте группе `Users` полные права на доступ ко всем объектам базы данных.
4. Закройте базу данных и Access. На этом первый этап заканчивается.
5. Запустите Access.
6. Создайте новую базу данных и зарегистрируйтесь как пользователь `Admin`.
7. Импортируйте в новую базу данных все объекты из защищенной базы данных.
8. Если текущий файл рабочей группы будет использоваться в дальнейшем, удалите пароль пользователя `Admin`. В этом нет необходимости, если в дальнейшем будет использоваться стандартный файл рабочей группы SYSTEM.MDW, созданный при установке Access.

Замечание

Теперь защита базы данных полностью снята. Любой пользователь Access сможет использовать базу данных и получить полный доступ к ее объектам: таблицам, запросам, формам, отчетам и макросам. Эти права доступа действуют для любого файла рабочей группы, поскольку учетная запись `Admin` везде одинаковая. Текущий файл рабочей группы, подключенный к Access в момент создания новой базы данных (шаг 6), определяет группу `Admins` администраторов новой базы данных.

Создание учетных записей

Управление учетными записями, как и назначение прав доступа, является задачей администрирования защищенной базы данных. Хотя помимо управления учетными записями существует достаточно широкий спектр других задач, за выполнение которых отвечает администратор. О задачах администрирования рассказано в *гл. 20*.

Назначение прав доступа

С помощью системы защиты на уровне пользователя можно ограничить доступ к базе данных и отдельным ее объектам (таблицам, запросам, формам, отчетам и макросам).

В Access 97 можно было назначить также права для доступа к модулям VBA. Начиная с версии Access 2000, в частности в Access 2003, защита модулей не входит в систему защиты на уровне пользователей. Защита программного проекта VBA выполняется отдельно. О том, как это сделать, рассказано в *разд. "Защита программного кода на VBA" этой главы* (а о защите страниц доступа к данным — в справочной системе Access).

Способ установки прав доступа к базе данных (и ее объектам), защищенной на уровне пользователей, описан в *гл. 20*, поскольку это является не только задачей разработки, но и задачей администрирования. *(См. разд. "Назначение прав доступа к объектам базы данных" в гл. 20.)*

Права доступа в приложении "Игра в доминирование"

Обратимся к системе защиты приложения "Игра в доминирование". Серверная база данных содержит объекты, которые будут совместно использоваться в сети несколькими игроками. Необходимо обеспечить защищенность этих объектов в такой степени, которая позволит игрокам принять участие в игре и при этом не вмешаться в ход игры. Клиентская база данных содержит объекты, которые будут использоваться одним игроком, имеющим собственную копию этой базы данных или открывшим базу данных клиента в режиме общего доступа. Взаимодействие с серверной базой данных осуществляется через интерфейс DAO с помощью процедур на VBA. Поэтому в клиентской базе данных имеет смысл защитить код на VBA и предоставить всем пользователям права, достаточные для использования объектов базы данных на уровне игрока. Таким образом, необходимо предоставить необходимые права группе `Users` и защитить проект VBA, как описано в *разд. "Защита программного кода на VBA" данной главы.*

Замечание

В примере приложения "Игра в доминирование" на компакт-диске программный проект не защищен и может быть просмотрен любым пользователем.

Пользователи приложения "Игра в доминирование" делятся на две категории:

- *пользователи, ведущие игру* — группа `Admins`. Этим пользователям разрешен полный доступ к объектам баз данных клиента и сервера. Они также являются администраторами этих баз данных;
- *игроки* — группа `Users`. Игрокам разрешен ограниченный доступ к объектам баз данных клиента и сервера. Права доступа, назначенные игрокам, приведены в табл. 16.9 и 16.10.

Замечание

Встроенная учетная запись `Admin` удалена из группы `Admins`. Все права доступа учетной записи `Admin` отменены. Таким образом, пользователь, не подключивший файл рабочей группы DOMINATION.MDW, может воспользоваться приложением "Игра в доминирование", но будет обладать при этом лишь ограниченными правами игрока.

Вообще говоря, можно было бы выделить отдельную группу пользователей-администраторов баз данных клиента и сервера, а группе ведущих игру назначить более ограниченные права. Однако в этом нет необходимости, и в нашей реализации ведущие игры являются также администраторами приложения.

Права доступа к объектам приложения "Игра в доминирование" подробно описаны в *Приложении 2*.

Ограничение разрешений для встроенных учетных записей Access

Чтобы закрыть доступ к объектам базы данных посторонним пользователям, зарегистрированным с помощью стандартной учетной записи `Admin`:

1. Подключите необходимый файл рабочей группы или создайте новый, как описано в *разд. "Использование файла рабочей группы" гл. 20*.

Замечание

Для того чтобы обеспечить полную защиту вашей базы данных, не используйте стандартный файл рабочей группы SYSTEM.MDW, создаваемый при установке Microsoft Access. Следует удостовериться, что файл рабочей группы, определяющий рабочую группу, к которой вы присоединяетесь, был создан с уникальным кодом рабочей группы, или, если это не так, следует создать новую рабочую группу.

2. Активизируйте диалоговое окно **Вход** (Logon), предназначенное для регистрации пользователя. Это диалоговое окно появляется в момент открытия базы данных, если регистрация пользователя требует ввода непустого пароля. В Microsoft Access по умолчанию для регистрации пользователя используется стандартная учетная запись `Admin`. Поэтому, чтобы активизировать диалоговое окно **Вход** (Logon), необходимо задать непустой пароль для учетной записи пользователя `Admin`. Установка пароля пользователя описана в *разд. "Управление учетными записями" гл. 20*.
3. Создайте учетную запись пользователя-администратора. Необходимо добавить эту учетную запись в стандартную группу `Admins`. Создание учетной записи пользователя описано в *разд. "Управление учетными записями" гл. 20*.
4. Закройте Microsoft Access и откройте базу данных снова. Когда появится диалоговое окно **Вход** (Logon), зарегистрируйтесь с именем нового администратора.
5. Удалите пользователя `Admin` из группы `Admins`.

Замечание

Если владельцем объектов в защищаемой базе данных должен быть пользователь, не входящий в группу администраторов, создайте для него учетную запись, снова перезапустите Access и зарегистрируйтесь с именем этого пользователя при открытии базы данных. Этот пользователь должен, по крайней мере, обладать правами **Чтение данных** (Read data) и **Чтение макета** (Read design) для всех объектов базы данных, которые требуется защитить.

Защита программного кода на VBA

Защитить программный проект VBA можно с помощью пароля:

1. Откройте окно редактора Visual Basic, например, с помощью комбинации клавиш <Alt>+<F11>.

2. Выберите команду **Сервис | Свойства *<Имя программного проекта>*** (Tools | <Project name> Properties). Появится диалоговое окно свойств программного проекта.
3. Раскройте вкладку **Защита** (Protection) и введите пароль в поле **Пароль** (Password) и **Подтверждение пароля** (Confirm password). Этот пароль будет требоваться при попытке отобразить диалоговое окно свойств проекта.
4. Если необходимо защитить проект от просмотра, установите флажок **Запретить просмотр программного кода** (Lock project for viewing). Это значит, что введенный пароль будет использован также для защиты программного проекта от просмотра.
5. Нажмите кнопку **OK**.

Установленные параметры начнут действовать только при следующем открытии базы данных.

Есть другой способ защиты программного кода приложения Access — можно создать файл, содержащий откомпилированную версию программного проекта и всех объектов базы данных (файл с расширением mde) или проекта Access (файл с расширением ade). Открыв этот файл, пользователи могут использовать все созданные в нем объекты, изменять данные и выполнять макросы и процедуры VBA, но не будут иметь возможности изменить структуру объектов или программный код. Такой файл имеет намного меньший размер по сравнению с исходным файлом базы данных или проекта. Права доступа и прочие обстоятельства, необходимые пользователю для возможности создания файла MDE или ADE, приведены в справочной системе Access.

Внимание!

Создать файл MDE или ADE можно только на основе базы данных или проекта Access в формате Access 2002—2003. Если ваша база данных (или проект) сохранена в формате Access 2000, потребуется сначала преобразовать ее в формат Access 2002—2003 *(см. гл. 19)*.

Чтобы создать файл MDE на основе базы данных в формате Access 2002-2003:

1. Откройте Access 2003, но не открывайте базу данных или проект Access.
2. Выберите команду **Сервис | Служебные программы | Создать MDE-файл** (Tools | Database Utilities | Make MDE File).
3. Появится диалоговое окно **База данных для сохранения как MDE** (Database to Save as MDE). В раскрывающемся списке **Типы файлов** (Files of type) выберите элемент **Базы данных Microsoft Office Access** (Microsoft Access Databases). Затем укажите папку и имя файла исходной базы данных (файла с расширением mdb) и нажмите кнопку **Создать MDE** (Make MDE).
4. Появится диалоговое окно **Сохранение файла MDE под именем** (Save MDE As). Укажите папку и введите имя целевого файла в формате MDE.
5. Нажмите кнопку **Сохранить** (Save).

Чтобы создать файл ADE на основе проекта в формате Access 2002—2003:

1. Откройте Access 2003, но не открывайте базу данных или проект Access.
2. Выберите команду **Сервис | Служебные программы | Создать MDE-файл** (Tools | Database Utilities | Make MDE File).

3. Появится диалоговое окно **База данных для сохранения как MDE** (Database to Save as MDE). В раскрывающемся списке **Типы файлов** (Files of type) выберите элемент **Проекты Microsoft Office Access** (Microsoft Access Projects). Затем укажите папку и имя файла исходного проекта Access (файла с расширением adp) и нажмите кнопку **Создать MDE** (Make MDE).
4. Появится диалоговое окно **Сохранение файла ADE под именем** (Save ADE As). Укажите папку и введите имя целевого файла в формате ADE.
5. Нажмите кнопку **Сохранить** (Save).



ГЛАВА 17

Проекты Access 2003

В данной главе мы постараемся показать, что Microsoft Office Access 2003, будучи настольной СУБД, не ограничивает пользователя в сложности и масштабируемости разрабатываемых приложений. Кроме создания достаточно сложных многопользовательских приложений, Access может использоваться в качестве средства для разработки клиентской части приложения с архитектурой "клиент-сервер". С помощью объектов Access может быть создан интерфейс баз данных, которые размещаются на мощных серверах баз данных, таких как Microsoft SQL Server, Oracle и т. д.

Для доступа к серверным базам данных из приложений Access используется один из двух стандартных способов доступа к удаленным данным: ODBC или OLE DB. Достоинством Access как клиента серверной базы данных является наличие мощных и простых средств для разработки интерфейса — форм, отчетов и Web-страниц. Наиболее простым и перспективным способом создания приложений в архитектуре "клиент-сервер" являются проекты Microsoft Access 2003 — файлы с расширением adp. В отличие от файла базы данных Access файл проекта не содержит таблиц с данными. Все таблицы, с которыми работает клиентское приложение, размещаются на сервере базы данных, а файл проекта включает в себя только те объекты, которые создаются на базе этих таблиц: формы, отчеты, страницы, макросы и модули. Однако из проекта Access доступны не только таблицы, но и другие объекты сервера: представления (views), хранимые процедуры (stored procedures), схемы базы данных (database diagrams). Доступ к этим объектам выполняется посредством OLE DB — универсального интерфейса, разработанного фирмой Microsoft для доступа к данным произвольного типа, как реляционным, так и не реляционным.

В качестве сервера базы данных в проектах Access 2003 может быть использован либо Microsoft SQL Server, версии 6.5 и выше, либо настольная (desktop) версия Microsoft SQL Server 2000.

Замечание

В Access 2003 сохранилась возможность создавать интерфейс серверных баз данных не только в проектах, но и в базах данных через присоединенные таблицы, используя доступ к серверу с помощью драйверов ODBC. Однако в данной книге мы не будем детально описывать работу посредством ODBC, а порекомендуем читателю при необходимости обратиться к книге: *И. Харитонова, В. Михеева "Microsoft* Access *2000 в подлиннике" — СПб.: БХВ — Санкт-Петербург, 1999.*

Основные понятия

Независимо от того, как определяется понятие архитектуры "клиент-сервер" (а таких определений в литературе много), в его основе лежит распределенная модель вычислений. В самом общем случае под *клиентом* и *сервером* понимаются два взаимодействующих процесса, из которых один является поставщиком некоторого сервиса для другого.

- *Сервер* — логический процесс, который обеспечивает некоторый сервис по запросу от клиента. Обычно сервер не только выполняет запрос, но и управляет очередностью запросов, буферами обмена, извещает своих клиентов о выполнении запроса и т. д.
- *Клиент* — процесс, который запрашивает обслуживание от сервера. Процесс не является клиентом по каким-то параметрам своей структуры, он является клиентом только по отношению к серверу. При взаимодействии клиента и сервера инициатором диалога с сервером, как правило, является клиент, сервер сам не инициирует совместную работу. Это не исключает, однако, того, что сервер может извещать клиентов о каких-то зарегистрированных им событиях. Инициирование взаимодействия, запрос на обслуживание, восприятие результатов от сервера, обработка ошибок — это обязанности клиента.

Здесь и далее в книге речь будет идти о частном случае архитектуры "клиент-сервер", а именно о приложениях баз данных, в которых сервером является мощная реляционная СУБД, такая как Microsoft SQL Server (back-end), а клиентом — приложение, созданное в среде Access 2003, которое использует данные с сервера (front-end).

Отличие архитектуры "клиент-сервер" от архитектуры "файл-сервер"

Приложение "клиент-сервер" отличается от сетевого многопользовательского приложения, которое рассматривалось в предыдущей главе, тем, где конкретно ведется обработка данных.

Сетевое многопользовательское приложение строится по принципу файл-серверной архитектуры. Данные в виде одного или нескольких файлов размещаются на файловом сервере. Файловый сервер принимает запросы, поступающие по сети от компьютеров-клиентов, и передает им требуемые данные. Однако обработка этих данных выполняется на компьютерах-клиентах. На каждом из компьютеров запускается полная копия процессора обработки данных Jet Engine. Любая копия Jet независимо управляет файлами MDB, содержащими данные. Единственная связь между этими независимыми действиями — файл блокировок (файл, который имеет имя, совпадающее с именем файла приложения, но с расширением ldb), который обязательно создается для каждого файла базы данных с расширением mdb. При этом каждая копия Jet выполняет изменения индексов, работу с системными таблицами и другие функции, входящие в компетенцию СУБД.

В архитектуре "клиент-сервер" сервер базы данных не только обеспечивает доступ к общим данным, но и берет на себя всю обработку этих данных. Клиент посылает на

сервер запросы на чтение или изменение данных, которые формулируются на языке SQL. Сервер сам выполняет все необходимые изменения или выборки, контролируя при этом целостность и согласованность данных, и результаты в виде набора записей или кода возврата посылает на компьютер клиента.

Недостатки архитектуры с файловым сервером очевидны и вытекают главным образом из того, что данные хранятся в одном месте, а обрабатываются в другом. Это означает, что их нужно передавать по сети, что приводит к очень высоким нагрузкам на сеть и, вследствие этого, резкому снижению производительности приложения при увеличении числа одновременно работающих клиентов. Вторым важным недостатком такой архитектуры является децентрализованное решение проблем целостности и согласованности данных и одновременного доступа к данным. Такое решение снижает надежность приложения.

Архитектура "клиент-сервер" позволяет устранить все указанные недостатки. Кроме того, она позволяет оптимальным образом распределить вычислительную нагрузку между клиентом и сервером, что также влияет на многие характеристики системы: стоимость, производительность, поддержку.

Распределение функций в архитектуре "клиент-сервер"

Процесс разработки таких систем достаточно сложен и одной из наиболее важных задач является как раз решение о том, как функциональность приложения должна быть распределена между клиентской и серверной частью. Пытаясь решить эту задачу, разработчики получают двухзвенные, трехзвенные и многозвенные архитектуры. Все зависит от того, сколько промежуточных звеньев включается между клиентом и сервером.

Основная задача, которую решает клиентское приложение, — это обеспечение интерфейса с пользователем, т. е. ввод данных и представление результатов в удобном для пользователя виде, и управление сценариями работы приложения.

Основные функции серверной СУБД — обеспечение надежности, согласованности и защищенности данных, управление запросами клиентов, быстрая обработка SQL-запросов.

Вся логика работы приложения — прикладные задачи, бизнес-правила — в двухзвенной архитектуре распределяются разработчиком между двумя процессами: клиентом и сервером (рис. 17.1).

Сначала большая часть функций приложения решалась клиентом, сервер занимался только обработкой SQL-запросов. Такая архитектура получила название "толстый клиент — тонкий сервер". Появление возможности создавать на сервере хранимые процедуры, т. е. откомпилированные программы с внутренней логикой работы, привело к тенденции переносить все большую часть функций на сервер. Сервер становился все более "толстым", а клиент — "утоньшался". Такое решение имеет очевидные преимущества, например его легче поддерживать, т. к. все изменения нужно вносить только в одном месте — на сервере. Однако язык, на котором пишутся хранимые процедуры, не является достаточно мощным и гибким, чтобы на нем было удобно реализовывать сложную логику приложения.

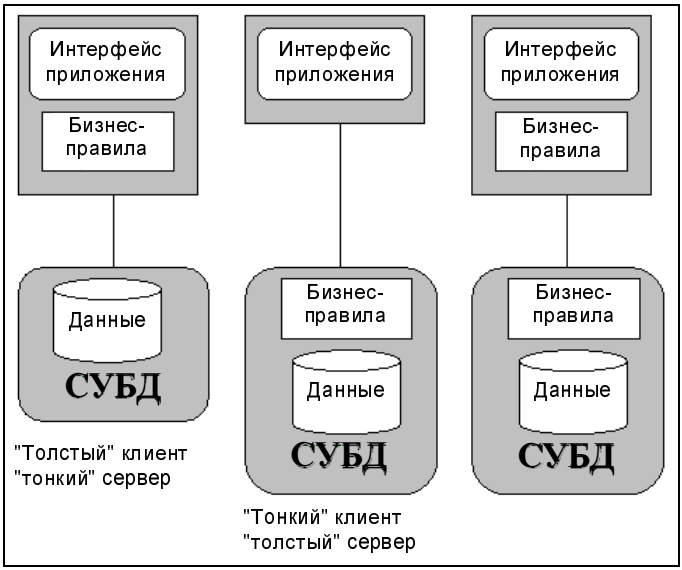


Рис. 17.1. Распределение функций между клиентом и сервером

Тогда возникла тенденция поручить выполнение прикладных задач и бизнес-правил отдельному компоненту приложения (или нескольким компонентам), которые могут работать как на специально выделенном компьютере — сервере приложений, так и на том же компьютере, где работает сервер базы данных. Так возникли трехзвенные и многозвенные архитектуры "клиент-сервер". Появилось специальное программное обеспечение (ПО) промежуточного слоя, которое должно обеспечить совместное функционирование множества компонентов приложения. Такие приложения являются гибкими, масштабируемыми, но сложными в разработке.

Далее в этой главе мы будем говорить только о двухзвенных приложениях "клиент-сервер", в которых клиент, реализованный на базе Access 2003, непосредственно обращается с запросами к серверу базы данных. Промежуточное ПО в данном случае только транслирует эти запросы в результирующие наборы записей между клиентом и сервером и не реализует никаких функций приложения.

Универсальный доступ к данным через OLE DB

Интерфейс ODBC был первым средством, которое обеспечило универсальный доступ к данным реляционного типа посредством SQL-запросов. Однако реляционные базы данных не единственный формат хранения данных, а современные приложения требуют интеграции информации из разных источников, не только SQL-ориентированных. Отсюда возникает потребность либо перевести все данные в единый формат, т. е. создать универсальную базу данных, что очень дорого и неэффективно, либо обеспечить универсальный доступ к данным разных типов без необходимости их преобразовывать и реплицировать.

OLE DB представляет собой разработанный фирмой Microsoft набор интерфейсов OLE, обеспечивающих унифицированный доступ приложений к данным из разнообразных источников, включая текстовые файлы, файлы электронной почты, электронные таблицы, данные мультимедиа и пр.

Основные отличия OLE DB от ODBC состоят в следующем:

- OLE DB обеспечивает доступ к данным произвольных типов, а не только реляционным;
- OLE DB не является набором функций, а представляет собой набор интерфейсов, построенных в соответствии с компонентной моделью объектов (COM).

Общие сведения

Архитектура приложения, использующего интерфейсы OLE DB для доступа к данным, представлена на рис. 17.2. Эта архитектура является многокомпонентной. Компоненты доступа к данным делятся на три категории: потребители, провайдеры и сервисные компоненты.

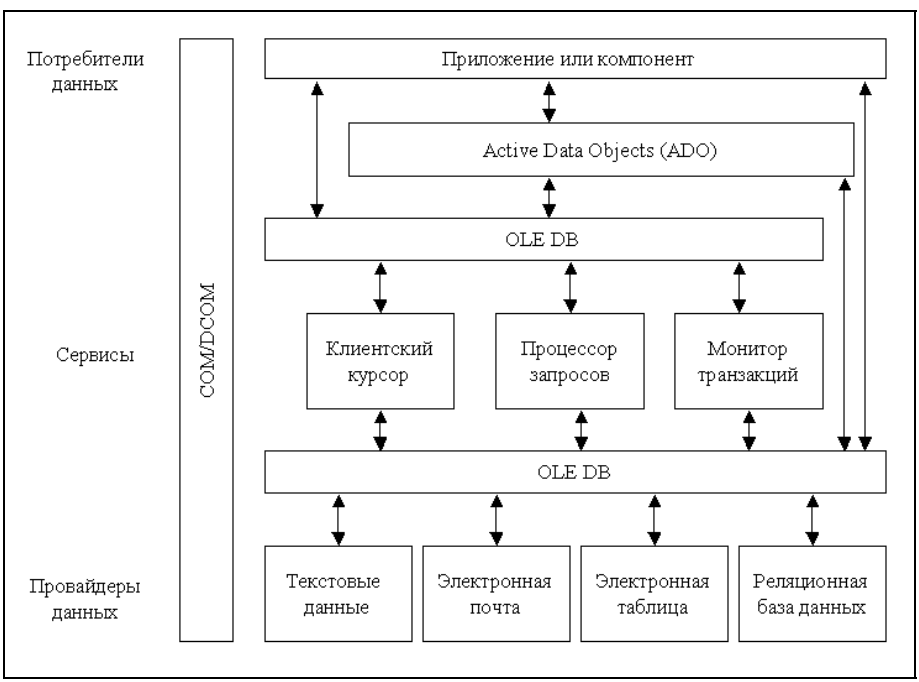


Рис. 17.2. Архитектура универсального доступа к данным

Потребители данных (Data Consumer) — это любое приложение или компонент, которые используют интерфейсы OLE DB для доступа к данным.

Провайдеры данных (Data Provider) — это компоненты, которые обеспечивают потребителям доступ к данным через строго специфицированный набор интерфейсов.

Они взаимодействуют с данными и представляют их единообразно в табличном виде, используя абстракцию, называемую *набор рядов* (rowset).

Сервисы (Services) — это дополнительные компоненты, которые обеспечивают функции, не реализованные провайдером OLE DB. Они являются как потребителями OLE DB данных, так и провайдерами. Примером сервиса может быть процессор запросов, который может объединять табличную информацию от разных провайдеров OLE DB и обеспечивать доступ к результирующим данным через интерфейс OLE DB.

Функционирование компонентов может реализовываться как на одном компьютере, так и на разных компьютерах через сетевые протоколы, такие как DCOM (Distributed Component Object Model, распределенная компонентная модель объектов) или HTTP. При этом для выполнения распределенных транзакций может использоваться координатор распределенных транзакций, например Microsoft Transaction Server (MTS).

Компоненты OLE DB

В OLE DB определена иерархия компонентов, каждый из которых является COM-объектом (рис. 17.3).

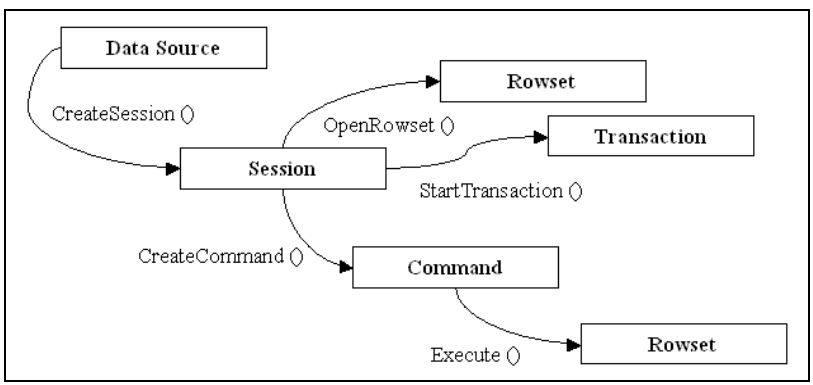


Рис. 17.3. Объекты — компоненты OLE DB

- *Источники данных* (Data Source) — объекты, которые реализуют подключение к источнику данных. Они определяют нужный провайдер OLE DB, проверяют права доступа потребителя данных и инициируют соединение с источником данных.

Замечание

Следует различать объект — источник данных и источник данных, который фактически содержит данные.

- *Сеансы* (Sessions) — объекты, которые реализуют функции поддержки соединения с источником данных. Они предоставляют контекст для выполнения транзакций и команд. Основная цель сеанса — установить рамки транзакции. Один объект — источник данных может поддерживать несколько сеансов, а значит, и несколько транзакций.

- *Транзакции* (Transactions) — объекты, которые обеспечивают реализацию механизма транзакций. Они предоставляют методы для того, чтобы начать транзакцию для сеанса или новую транзакцию внутри текущей и подтвердить или отменить транзакцию самого нижнего уровня.
- *Команды* (Commands) — объекты, которые реализуют выполнение действий с данными (например, запросов). Команды порождаются сеансом, и в одном сеансе можно создать несколько команд.
- *Наборы рядов* (Rowsets) — объекты, которые предоставляют данные в табличной форме. Они порождаются либо сеансом, либо командой в качестве результата ее выполнения. Непосредственно из сессии можно создать набор рядов, содержащий все данные таблицы. Для реализации такого простого запроса не требуется команды. В остальных же случаях для создания набора рядов используются команды.

Microsoft ActiveX Data Objects (ADO)

Хотя OLE DB является очень мощным интерфейсом для работы с данными, этот интерфейс является низкоуровневым. Для удобства работы с OLE DB, так же как и для ODBC, была разработана объектная модель, которую назвали ADO (ActiveX Data Objects). *(Эта модель была описана в разд. "Объектные модели Microsoft Access" гл. 13).* Здесь хотелось бы указать на те достоинства этой модели, которые позволяют говорить о ее ключевой роли в приложениях, связанных с обработкой данных.

- ADO является общей программной моделью для работы с данными различных типов. Она разрабатывалась специально для того, чтобы заменить все другие интерфейсы работы с данными. Впервые она была реализована в Internet Information Server (IIS), где успешно работала вместе с Active Server Pages.
- Модель включила ряд возможностей других известных объектных моделей (DAO и RDO), хотя и не полностью. Но она является расширяемой и в последней версии ADO.NET превзошла эти модели как по функциональности, так и по производительности.
- Так как ADO реализована на базе COM-объектов, то она может быть использована в любом языке, который может работать с COM-объектами, в том числе и в VBA.
- ADO обеспечивает доступ к любому источнику данных OLE DB, для которого имеется провайдер OLE DB, и, более того, она позволяет расширить функциональность провайдера.
- ADO реализована таким образом, чтобы минимизировать сетевой трафик в интернет-приложениях и сократить число промежуточных слоев между фронтальным (клиентским) приложением и источниками данных. Это требуется для того, чтобы сделать интерфейс как можно более легким и высокопроизводительным.

Установка связи с источником данных посредством интерфейса OLE DB

При установке на компьютере Microsoft Office 2003 или отдельного приложения Access 2003 автоматически устанавливаются следующие провайдеры OLE DB:

- Microsoft Jet 4.0 OLE DB Provider;
- Microsoft OLE DB Provider for SQL Server;
- OLE DB Provider for ODBC Drivers;
- OLE DB Provider for Oracle.

Для того чтобы посмотреть, какие провайдеры OLE DB установлены на вашем компьютере, необходимо открыть диалоговое окно **Data Link Properties**. Это окно открывается при создании или редактировании специальных файлов — Microsoft Data Link, или файлов UDL (Universal Data Link), в которых хранится информация о конкретном источнике данных OLE DB (тип провайдера OLE DB; сервер, на котором размещаются данные; база данных или файл, в котором они хранятся).

Информация о соединении может храниться и запрашиваться также в том приложении, которое использует интерфейс OLE DB для доступа к данным. В нашем случае таким приложением является проект Microsoft Office Access, однако о проектах Access речь еще впереди, поэтому пока мы опишем только как работать с файлами связей Microsoft Data Link.

Для того чтобы создать новый UDL-файл, проще всего воспользоваться программой Проводник (Explorer) Windows:

1. В окне Проводника откройте папку, в которую вы хотите поместить UDL-файл. Щелкните правой кнопкой мыши по правой панели и выберите из контекстного меню команду **Создать | Текстовый документ** (New | Text File).
2. Введите имя файла, например `TestSQLServer`, и дайте ему расширение udl. В окне с предупреждающим сообщением нажмите кнопку **Да** (Yes). В результате будет создан новый файл с расширением udl.
3. Откройте этот файл двойным щелчком левой кнопки мыши. Появится диалоговое окно **Data Link Properties** (Свойства связи с данными).
4. Раскройте вкладку **Provider** (Провайдер). Вы увидите список доступных провайдеров OLE DB, например такой, как показано на рис. 17.4.
5. Выберите один из провайдеров, например **Microsoft OLE DB Provider for SQL Server** (Провайдер OLE DB для Microsoft SQL Server), и нажмите кнопку **Next** (Далее). Раскроется вкладка **Connections** (Соединения).

Вкладка **Connections** (Соединения) и следующая вкладка **Advanced** (Дополнительно) содержат параметры соединения с источником данных. Вид этих вкладок зависит от используемого провайдера OLE DB. Для SQL Server OLE DB-провайдера вкладка **Connections** (Соединения) выглядит так, как представлено на рис. 17.5.

Замечание

Вкладка такого же вида отображается в диалоговом окне **Data Link Properties**, появляющемся при выборе команды **Файл | Подключение** (File | Connection) в окне приложения при работе с проектом Access.

На этой вкладке нужно определить сервер, с которым будет выполняться соединение, имя базы данных и параметры регистрации на сервере. Можно использовать интегрированную с Windows NT (**Use Windows NT Integrated security**) или стандартную (**Use a specific user name and password**) схему безопасности. При выборе пере-

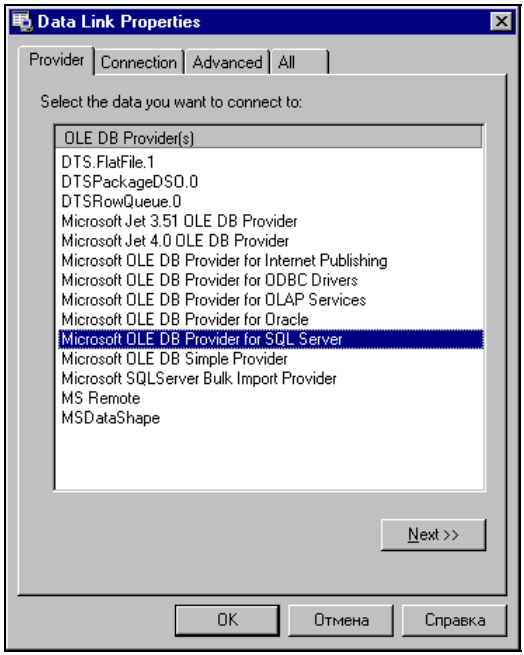


Рис. 17.4. Список доступных провайдеров OLE DB

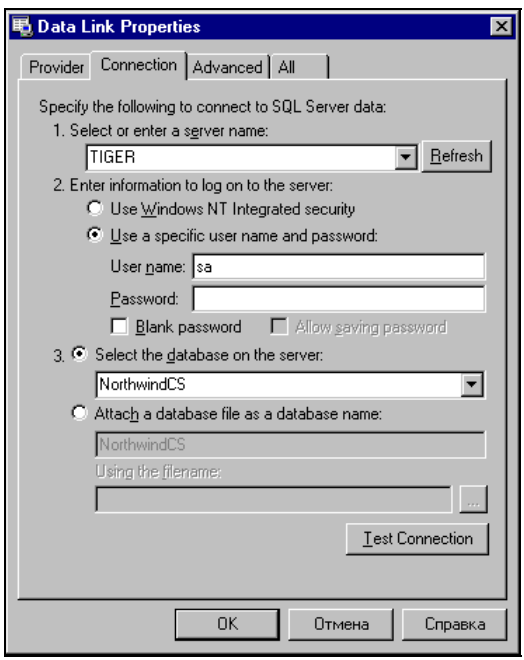


Рис. 17.5. Определение связи с SQL Server

ключателя **Use a specific user name and password** (Использовать имя и пароль пользователя) нужно ввести имя и пароль для регистрации на сервере. Флажок **Blank password** (Без пароля) предназначен для использования пустого пароля в строке соединения (connection string). Флажок **Allow saving password** (Разрешить сохранение пароля) позволяет сохранить пароль в строке соединения. Если используется интегрированная схема безопасности, то подключение к SQL Server будет выполняться с тем именем и паролем, под которым вы зарегистрировались в сети Windows NT. На этой вкладке есть также кнопка **Test Connection** (Проверить соединение), позволяющая немедленно протестировать соединение с заданными параметрами.

Следующая вкладка **Advanced** (Дополнительно) позволяет определить дополнительные свойства соединения, например **Connection timeout** (Время ожидания соединения) (рис. 17.6). Это время ожидания (в секундах) соединения с источником данных. Если через заданный промежуток времени связь с источником данных установить не удалось, выдается сообщение об ошибке.

Все свойства соединения можно увидеть и отредактировать на вкладке **All** (Все) (рис. 17.7). Перечень свойств в списке зависит от типа провайдера OLE DB. Для того чтобы отредактировать любое свойство, нужно выделить его в списке и нажать кнопку **Edit Value** (Изменить значение). Появится диалоговое окно **Edit Property Value (**Изменение значения свойства) (рис. 17.8). В поле **Property Value** (Значение свойства) можно ввести новое значение свойства, предварительно очистив его при необходимости нажатием кнопки **Reset Value** (Сброс значения).

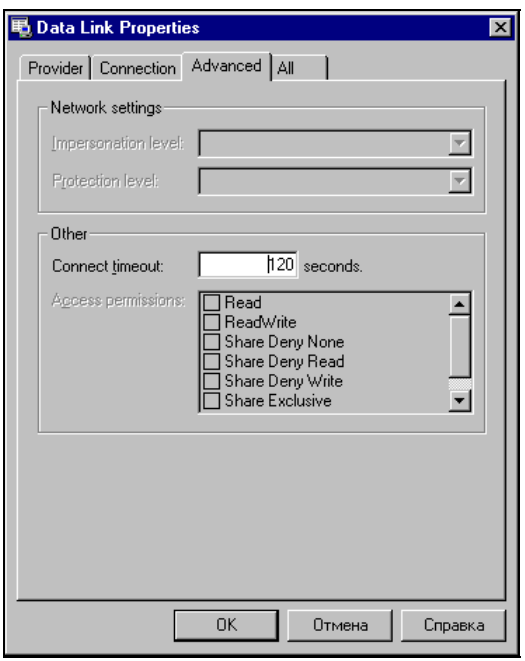


Рис. 17.6. Определение дополнительных свойств соединения

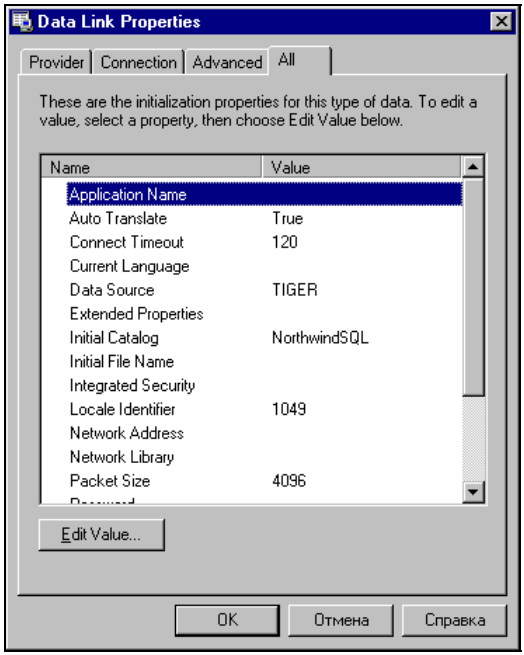


Рис. 17.7. Список свойств соединения

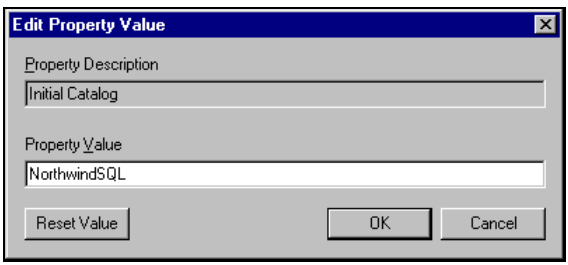


Рис. 17.8. Диалоговое окно **Edit Property Value**

Настольная версия Microsoft SQL Server 2000

Данная версия сервера представляет собой процессор обработки данных, который является альтернативой процессору Jet (первая версия этого процессора называлась MSDE — Microsoft Server Database Engine — и была совместима с Microsoft SQL Server 7.0). Он может быть использован либо как локальный сервер, и в этом случае устанавливается на тот же компьютер, на котором установлен Access 2003, либо как сервер баз данных для небольшой рабочей группы, и в этом случае будет удаленным по отношению к клиентским компьютерам. Основное его достоинство по сравнению с процессором Jet — полная совместимость с Microsoft SQL Server 2000. Это

означает, что, создав приложение целиком на персональном компьютере, можно в любой момент времени без труда перенести всю серверную его часть на Microsoft SQL Server 2000 Standard Edition или Enterprise Edition. В результате становится возможным многократное увеличение числа пользователей приложения и получение доступа к многочисленным службам SQL Server, например Data Transformation Services, OLAP Services и т. д.

В отличие от Microsoft SQL Server 2000 Standard Edition, Microsoft SQL Server 2000 Desktop Engine имеет ограничение на количество одновременно работающих с базой данных пользователей и на объем базы данных, не поддерживает симметричную мультиобработку (SMP) и в процессе репликации может функционировать только как подписчик. Максимальный объем базы данных, так же как и у Jet, равен 2 Гбайт, а количество одновременно работающих пользователей ограничивается пятью активными запросами (потоками).

Итак, с помощью проектов Access, используя Microsoft SQL Server 2000 Desktop Engine, вы можете создавать надежные многопользовательские приложения, а когда потребуется подключить к базе данных большое число пользователей или объем базы данных будет превышать 2 Гбайт, можно легко масштабировать этот проект, просто перенеся базу данных на SQL Server 2000.

Установить Microsoft SQL Server 2000 Desktop Engine можно с того же компакт-диска, что и Microsoft Office XP. Он находится в папке MSDE2000. При этом программа установки не запрашивает ни имя папки, в которую будет установлен сервер, ни имя самого сервера. Установка выполняется в папку C:\PROGRAM FILES \MICROSOFT SQL SERVER, а имя сервера по умолчанию — MSSQLSERVER. Если требуется изменить эти значения, нужно использовать при установке параметры командной строки *(см. раздел справки **Работа с проектами Microsoft Access**, **Общие сведения о проекте Microsoft Access**, **Установка и настройка SQL Server 2000 Desktop Engine** (Working with Microsoft Access Projects, Basic Tasks, Install and configure SQL Server 2000 Desktop Engine))*.

Замечание

Установка Microsoft SQL Server 2000 Desktop Engine выполняется корректно на компьютер с операционными системами Windows 2000 и выше.

Создание проекта Access 2003

Создание проекта Access аналогично созданию базы данных Access. Поэтому мы опишем эту процедуру кратко *(см. разд. "Создание новой базы данных" гл. 1)*. Существует три возможности создания проекта Access:

- создать проект и сразу же связать его с существующей базой данных на сервере;
- создать проект и одновременно создать новую базу данных на сервере;
- создать проект, не связывая его пока с базой данных сервера.

В любом случае нужно сначала выполнить команду меню **Файл | Создать** (File | New), чтобы появилась область задач **Создание файла** (New File).

Создание проекта с использованием существующей базы данных

Для создания проекта выполните следующее:

1. Выберите в группе **Создание** (New) на панели задач ярлык **Проект с имеющимися данными** (Project using existing data).
2. В диалоговом окне **Файл новой базы данных** (File new database) введите имя файла проекта. Этот файл получает расширение adp. Нажмите кнопку **Создать** (Create). Появится диалоговое окно **Data Link Properties** (Свойства связи с данными), в котором нужно задать параметры соединения с сервером.
3. Введите имя сервера, если требуется — имя и пароль, и выберите из списка нужную базу данных.

Замечание

Если вы хотите использовать SQL Server 2000 Desktop Engine, установленный локально, то поля в спецификации соединения должны быть заполнены так, как это представлено на рис. 17.9 (имя сервера — это имя вашего компьютера или специальный идентификатор `local`, и имя пользователя — `sa`).

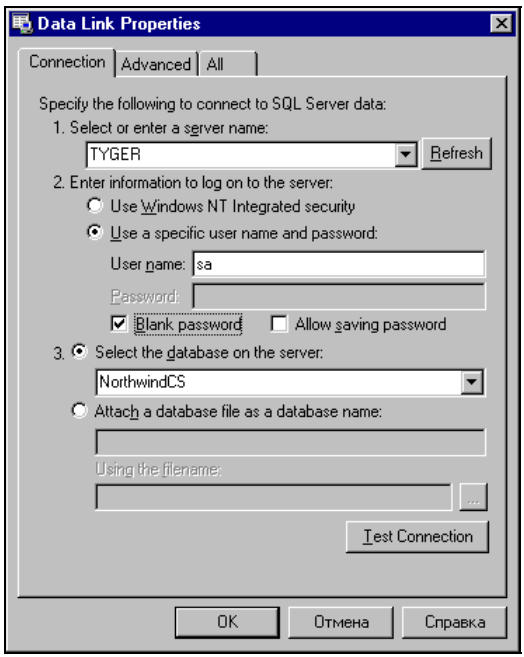


Рис. 17.9. Установление соединения с базой данных на сервере

Можно убедиться в правильности задания параметров соединения, нажав кнопку **Test Connection**. Нажмите кнопку **OK**. Появится окно проекта Access 2000, очень похожее на окно обычной базы данных Access, однако список ярлыков на панели

объектов будет несколько отличаться от традиционного списка объектов базы данных Access (рис. 17.10).

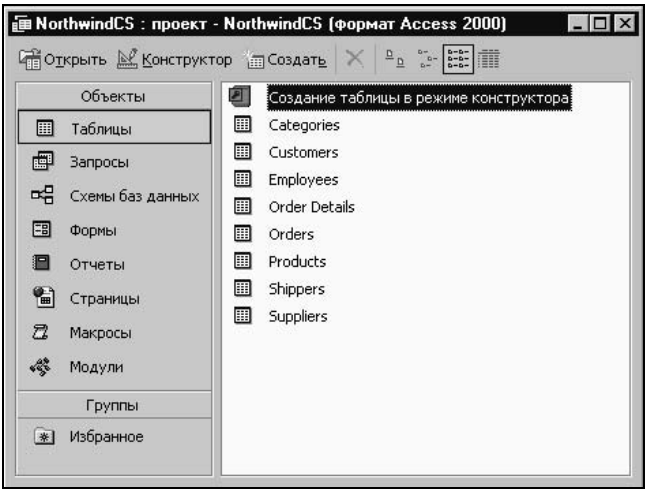


Рис. 17.10. Окно проекта Access 2003

Замечание

В качестве примера мы используем учебный проект NorthwindCS, входящий в комплект демонстрационных приложений Access 2003. При установке Access 2003 в папке Samples размещается сценарий установки этой базы на Microsoft SQL Server — файл NorthwindCS.SQL. Этот файл содержит набор предложений SQL, которые создают на сервере необходимые таблицы, представления, хранимые процедуры и загружают данные. Этот сценарий можно выполнить на сервере с помощью утилиты SQL Enterprise Manager, которая включена в дистрибутивный пакет любой версии сервера, либо он выполняется автоматически при открытии файла NordwindCS.adp.

Еще одним примером может служить демонстрационная база данных Pabs, которая создается на SQL Server при его установке и содержит данные о книгах, их авторах и издателях.

При создании нового проекта можно не устанавливать сразу соединение с базой данных на сервере, а выполнить подключение позже. Для этого в диалоговом окне **Data Link Properties** (Свойства связи с данными) нажмите кнопку **Отмена** (Cancel).

Присоединить базу данных к существующему проекту Access 2003 можно с помощью команды меню **Файл | Подключение** (File | Connection). Эту же команду можно использовать также для подключения проекта к другой базе данных.

Создание проекта с новой базой данных

Для создания проекта выполните следующее:

1. Выберите ярлык **Проект с новыми данными** (Project using new date) в группе **Создание** (New) области задач **Создание файла** (New File).

2. В диалоговом окне **Файл новой базы данных** (File New Database) в поле **Имя файла** (File name) введите имя проекта, укажите папку, в которой будет создан файл проекта, и нажмите кнопку **Создать** (Create).
3. Откроется пустое окно проекта Access и запустится **Мастер баз данных Microsoft SQL Server** (Microsoft SQL Server Database Wizard), который будет создавать базу данных на сервере. В диалоговом окне Мастера (рис. 17.11) нужно ввести имя сервера (если Access обнаруживает на локальном компьютере Microsoft Server Desktop Edition, то имя сервера автоматически подставляется), имя и пароль пользователя для регистрации на сервере и имя базы данных. После этого нажмите кнопку **Далее** (Next).

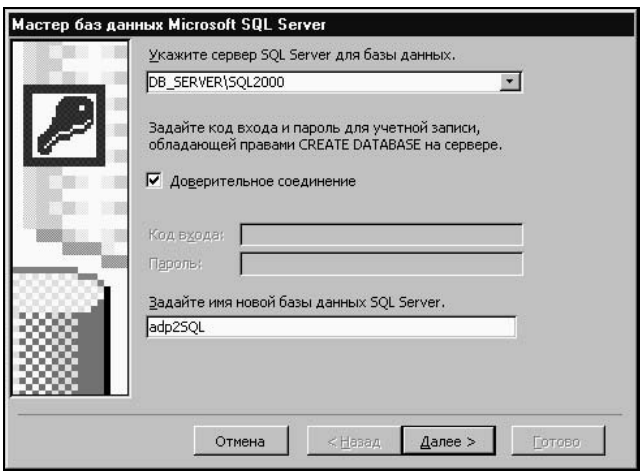


Рис. 17.11. Первое диалоговое окно **Мастер баз данных Microsoft SQL Server**

4. Если все введено правильно, во втором диалоговом окне Мастера нужно только нажать кнопку **Готово** (Finish). После этого появится окно с индикатором процесса создания новой базы данных на сервере (рис. 17.12).

Замечание

Если база данных создается на SQL Server 6.5, то появятся дополнительные окна Мастера, в которых нужно ввести названия и размеры устройств для базы данных и журнала транзакций, а также размеры базы данных и журнала транзакций.

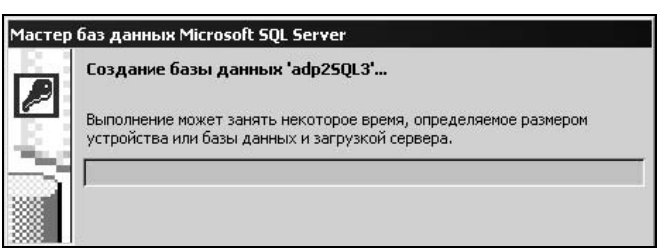


Рис. 17.12. Диалоговое окно с индикатором процесса создания базы данных на сервере

Открытие файла проекта

Файл проекта Access открывается аналогично открытию файла базы данных. Однако, в отличие от файла базы данных, файл проекта всегда открывается в монопольном режиме. Если вы пытаетесь открыть файл, который уже открыт другим пользователем, он откроется в режиме "только для чтения". Тем не менее и в этом режиме можно работать с объектами, размещенными на сервере (изменять, удалять, создавать новые объекты). Запрещается только модифицировать объекты, находящиеся в самом файле проекта. Для примера можно открыть файл NorthwindCS.adp, который размещается в папке Samples при установке Access 2003. При первом открытии данного файла Access может автоматически создать базу данных на SQL Server. Для этого используется файл сценария установки NorthwindCS.SQL, который находится в той же папке Samples. Сначала Access ищет настольную версию SQL Server на локальном компьютере, и если найдет, то спросит вас, хотите ли вы создать на нем новую базу данных `NorthwindCS`. Если на локальном компьютере сервер не найден, то Access попросит ввести имя удаленного сервера, имя пользователя и пароль. После этого исполняется сценарий установки базы данных на соответствующем сервере, и файл проекта NorthwindCS.ADP соединяется с новой базой данных `NorthwindCS`.

Окно проекта представлено ранее, на рис. 17.10. В списке таблиц вы видите те же таблицы, что и в базе данных `Борей` (Northwind). Однако эти таблицы хранятся не в файле проекта, а на сервере, хотя значок таблицы выглядит так, как если бы это были локальные таблицы. На панели объектов проекта Access по сравнению с базой данных Access появился новый ярлык — **Схемы баз данных** (Database Diagrams). Соответствующие новые объекты также размещены на сервере. Все остальные объекты (кроме страниц доступа к данным) находятся в файле проекта.

Работа с таблицами

Если вы откроете одну из таблиц, например `Products` (Товары), то увидите, что форма представления таблицы почти не изменилась (рис. 17.13). Появились только две дополнительные кнопки справа от кнопок навигации. Одна из них (крайняя справа) используется для задания максимального числа записей, которые будут передаваться с сервера. Это значение отображается в небольшом диалоговом окне. Вы можете ограничить передаваемое число записей, введя требуемое значение в текстовое поле **Максимальное число записей** (Max Record Count). Еще одна кнопка (вторая справа) подсвечивается красным цветом в процессе передачи записей. С помощью этой кнопки можно прервать процесс передачи записей с сервера. Заметим, что такой возможности явно не хватает в базе данных Access. Когда запрос выполняется достаточно долго, вы не можете прервать этот процесс.

Определение и изменение структуры таблицы

Открыв таблицу в режиме Конструктора, вы увидите, что способ отображения структуры таблицы незначительно отличается от принятого в режиме Конструктора таблиц базы данных Access (рис. 17.14).

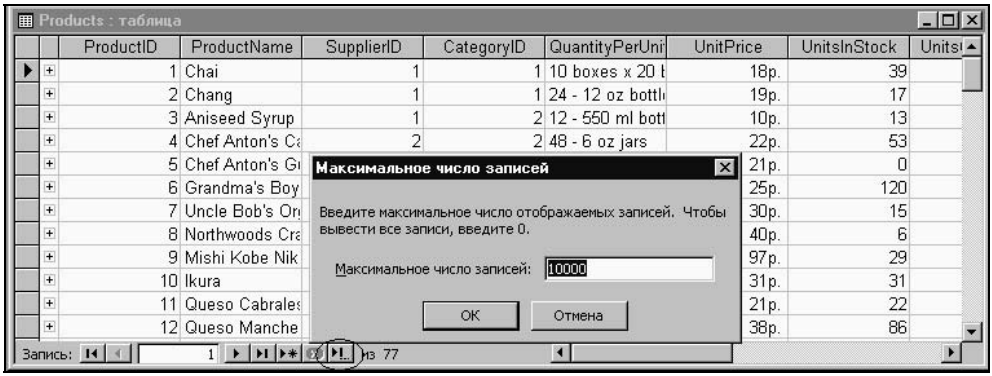


Рис. 17.13. Отображение таблицы, хранящейся на SQL Server, в проекте Access

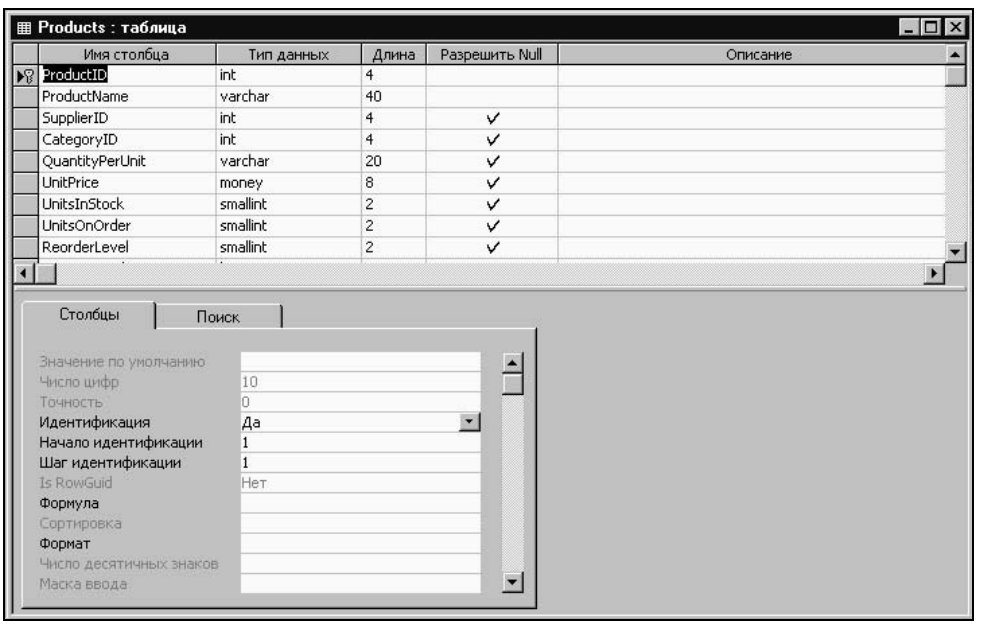


Рис. 17.14. Таблица SQL Server в режиме Конструктора

В верхней части Конструктора появились два новых столбца:

- **Длина** (Length) — длина поля в байтах;
- **Разрешить Null** (Allow Null) — флажок, определяющий, допустимо или нет значение Null в данном поле.

Значение в столбце **Длина** (Lengh) может изменяться пользователем только для текстовых полей, в остальных случаях это значение устанавливается по умолчанию при выборе типа данных и не может быть изменено. Раскрывающийся список в столбце **Тип данных** (Data Type) содержит значения, отличающиеся от типов данных, принятых в базах данных Access. Это типы данных для Microsoft SQL Server.

Столбец **Описание** (Description) — эквивалент поля **Примечание** (Comment) в Access — отображается только для SQL Server 2000.

Остальные свойства полей таблицы отображаются на вкладке **Столбцы** (Columns) в нижней части окна Конструктора. В отличие от окна Конструктора таблицы Access здесь отображаются сразу все свойства полей таблицы, но доступны только часть свойств в зависимости от типа данных в поле. Описание свойств приведено в табл. 17.1.

Таблица 17.1. *Свойства полей*

Свойство	Описание	Значения по умолчанию
Значение по умолчанию (Default Value)	Значение, которое вставляется в поле, если пользователь ничего в него не ввел. Это значение игнорируется для полей, имеющих тип данных timestamp. Если значение по умолчанию не задано, в поле вставляется значение Null	Пусто
Число цифр (Precision)	Максимальное количество цифр (для числовых полей), разрешенных для данного поля	Зависит от типа данных. Например, для int равно 4
Точность (Scale)	Максимальное количество десятичных знаков после разделителя. Это значение должно быть меньше или равно значению Precision	Как правило, 0, для типа данных money и smallmaney — 4
Идентификация (Identity)	Свойство, которое обеспечивает автоматическую генерацию уникальных значений в этом поле при вставке новых записей	Нет (флажок сброшен)
Начало идентификации (Identity Seed)	Значение, которое присваивается первой записи в таблице	1
Шаг идентификации (Identity Increment)	Шаг изменения значения поля с идентификацией. В каждой последующей записи значение поля увеличивается на это число	1
Is RowGuid	Указывает, является ли данное поле глобальным идентификатором. В отличие от поля, которое имеет значение свойства **Идентификация** (Identity) равным **Да** (Yes) и которое однозначно идентифицирует каждую запись в таблице, SQL Server версии 7.0 и выше позволяет создать поле в таблице, которое является глобальным идентификатором, т. е. каждое значение этого поля может быть уникальным в пределах всей базы данных или даже во всех базах данных на всех серверах сети. В этом случае значение данного поля должно быть **Да** (Yes), но тип данных может быть только uniqueidentifier	Нет

Таблица 17.1 *(продолжение)*

Свойство	Описание	Значения по умолчанию
Формула (Formula)	Показывает формулу для вычисляемого столбца. Только для SQL Server 2000	Пусто
Сортировка (Collation)	Для текстовых полей позволяет выбрать способ сортировки данных в поле. Только для SQL Server 2000	По умолчанию для базы данных
Формат (Format)	Позволяет задать формат отображения данных в поле. Только для SQL Server 2000	Пусто
Число десятичных знаков (Decimal Places)	Для числовых полей определяет количество десятичных знаков при отображении данных в поле. Аналог соответствующего свойства в Access. Если установлено значение [**Авто**] (Auto), то количество отображаемых десятичных знаков определяется значением свойства **Формат** (Format)	Авто
Маска ввода (Input Mask)	Аналог соответствующего свойства в Access. Только для SQL Server 2000	Пусто
Подпись (Caption)	Определяет подпись для поля, которая появляется в формах. Только для SQL Server 2000	Пусто
Индексация (Indexed)	Определяет, будет ли создаваться индекс для данного поля. Только для SQL Server 2000	Нет
Гиперссылка (Hyperlink)	Указывает, будет ли значение этого поля интерпретироваться как гиперссылка. Только для SQL Server 2000	Нет
Режим IME (IME Mode)	Позволяет задать способ ввода данных в поле при использовании китайского, японского и некоторых других языков. Только для SQL Server 2000	Нет
Режим предложений IME (IME Sentence Mode)	Определяет режим преобразования данных при использовании китайского, японского и ряда других языков. Только для SQL Server 2000	Отсутствует
Фуригана (Furigana)	Указывает столбец, в который записывается эквивалент текста в системе фуригана. Когда пользователь вводит значение в данный столбец, сохраняется это значение и дополнительно его эквивалент в системе фуригана записывается в столбец, который указан в данном свойстве. Только для SQL Server 2000	Пусто

Таблица 17.1 *(окончание)*

Свойство	Описание	Значения по умолчанию
Почтовый адрес (Postal Address)	Содержит имя элемента управления или поля, которое отображает почтовый адрес, соответствующий введенному в данное поле почтовому коду, либо штрихкод клиента, соответствующий введенному адресу. Только для SQL Server 2000	Пусто

В проекте Access таблицы тоже могут иметь поля подстановки. Раскройте вкладку **Поиск** (Lookup) и вы увидите знакомые свойства, которые позволяют определить поле подстановки. Оно задается точно так же, как и в базе данных Access.

Типы данных, которые поддерживает SQL Server, приведены в табл. 17.2.

Таблица 17.2. *Типы данных Microsoft SQL Server*

Типы данных	Типы данных SQL Server	Примечания
Двоичные	binary[(n)] varbinary[(n)]	
Символьные	char[(n)] varchar[(n)]	
	nchar nvarchar	Используются для поддержки UNICODE-символов. Только в Microsoft SQL Server 7.0
Дата и время	datetime smalldatetime	Нет отдельных типов для даты и времени
Числовые с фиксированной точностью (Exact numeric)	decimal[(p[, s])] numeric[(p[, s])]	Не теряют точность за счет округления
Приблизительные числовые (Approximate numeric)	float[(n)] real	
Целые	bigint int smallint tinyint	8 байт 4 байта 2 байта 1 байт
Денежные	money smallmoney	
Специальные	bit timestamp	
Текст	text ntext	Используются для поддержки UNICODE-символов. Только в Microsoft SQL Server 7.0

Таблица 17.2 (окончание)

Типы данных	Типы данных SQL Server	Примечания
Графика	image	
Курсор	cursor	Можно использовать для выходных параметров процедур. Только в Microsoft SQL Server 7.0
Уникальный идентификатор	uniqueidentifier	Соответствует GUID в модели COM. Только в Microsoft SQL Server 7.0
	sql_variant	Предназначен для сохранения различных типов данных, за исключением text, ntext, image, timestamp и sql_variant
Пользовательские типы данных (user-defined datatypes)		Определяются на базе системных типов данных

Диалоговое окно свойств таблицы

Если щелкнуть правой кнопкой мыши в верхней части окна Конструктора и выбрать в контекстном меню команду **Свойства** (Properties), появится диалоговое окно **Свойства** (Properties), имеющее пять вкладок. На них перечисляются свойства таблицы, которые можно просматривать и изменять. Наиболее важны три вкладки:

- **Связи** (Relationships);
- **Индексы и ключи** (Indexes/Keys);
- **Проверить ограничения** (Check Constraints).

На вкладке **Связи** (Relationships) отображаются и могут модифицироваться внешние ключи (foreign keys) таблицы, которые отражают ее связи с другими таблицами. Внешние ключи создаются автоматически в процессе задания связей между таблицами на схеме данных.

Поле со списком **Выделенная связь** (Selected relationship) содержит список связей таблицы. Выбрав из этого списка значение, в остальных полях можно увидеть описание выбранной связи.

Поле **Имя связи** (Relationship name) показывает имя данной связи.

Таблица из двух столбцов отображает две стороны связи: слева — имя таблицы и имя поля, которое является первичным ключом; справа — имя таблицы и имя поля, которое является внешним ключом, т. е. содержит значения первичного ключа другой таблицы. Например, на рис. 17.15 представлена связь между таблицами `Products` (Товары) и `Categories` (Типы). Эти таблицы связаны по полю `CategoryID` (КодТипа). Данное поле в таблице `Categories` является первичным ключом, а в таблице `Products` — внешним. Имя связи состоит из трех частей и формируется следующим

образом: первым пишется имя таблицы, включающей внешний ключ, затем имя таблицы, содержащей первичный ключ, и перед ними ставится префикс. Префикс зависит от того, на какой стороне оказывается таблица, связи которой описываются: если она содержит внешний ключ, то пишется префикс `FK_`, если она содержит первичный ключ, то префикс `PK_`. В данном примере получается имя `FK_Products_Categories`, т. к. мы смотрим связи таблицы `Products` (Товары). Перед именем связи присутствует значок "бесконечность", который указывает, что данная таблица находится в связи на стороне "многие", а значок "ключ" означает, что данная таблица находится на стороне "один". Имя связи формируется автоматически, однако оно может быть откорректировано в поле **Имя связи** (Relationship name).

Для выбранной связи можно изменить поле, которое будет содержать внешний ключ, выбрав из списка полей таблицы нужное поле. Изменить поле первичного ключа нельзя, это можно сделать только на вкладке **Индексы и ключи** (Indexes/Keys).

Флажки в нижней части вкладки имеют следующие значения.

- Флажок **Проверять имеющиеся данные при создании** (Check existing data on creation), будучи установленным, гарантирует, что для всех данных, которые были введены в таблицу до того, как ввели эту связь, будет выполнена проверка, удовлетворяют ли они этому ограничению.
- Флажок **Применить связь при репликации** (Enforce constraint for replication) позволяет включать и отключать проверку данного ограничения при копировании таблицы в другую базу данных.
- Флажок **Применить связь для INSERT и UPDATE** (Enforce constraint for INSERT and UPDATE) позволяет включать и отключать проверку данного ограничения при добавлении, изменении и удалении записей в таблице, содержащей внешний ключ. Каскадное удаление и обновление связанных полей возможно только для Microsoft SQL Server 2000.

Кнопки **Создать** (New) и **Удалить** (Delete) используются для создания и удаления связей.

На вкладке **Индексы и ключи** (Indexes/Keys) можно вводить и изменять индексы таблицы, первичный ключ и ограничения типа *уникальность* (флажок **UNIQUE**) (рис. 17.16).

Первичный ключ устанавливается в таблицах проекта так же, как в таблицах Access: нужно выделить поле или несколько полей и нажать кнопку **Ключевое поле** (Primary Key) на панели инструментов в режиме Конструктора таблицы. Первичный ключ может быть изменен с помощью диалогового окна **Свойства** (Properties). Для этого нужно выбрать в списке **Выделенный индекс** (Selected index) первичный ключ (он имеет префикс `PK_`). В поле **Тип** (Type) появится значение **Первичный ключ** (Primary key). В списке **Имя столбца** (Column name) определите поле или несколько полей, которые будут составлять ключ. При этом, если ключ составной, нужно выбирать поля в том порядке, в каком они должны быть в ключе. При выходе из поля **Имя столбца** (Column name) (при переносе фокуса в любое другое поле) проверяется правильность определения ключа и выдается сообщение об ошибке, если нарушено какое-либо правило.

Поле **Имя индекса** (Index Name) содержит имя индекса. Для первичного ключа это имя состоит из префикса `PK_` и имени таблицы.

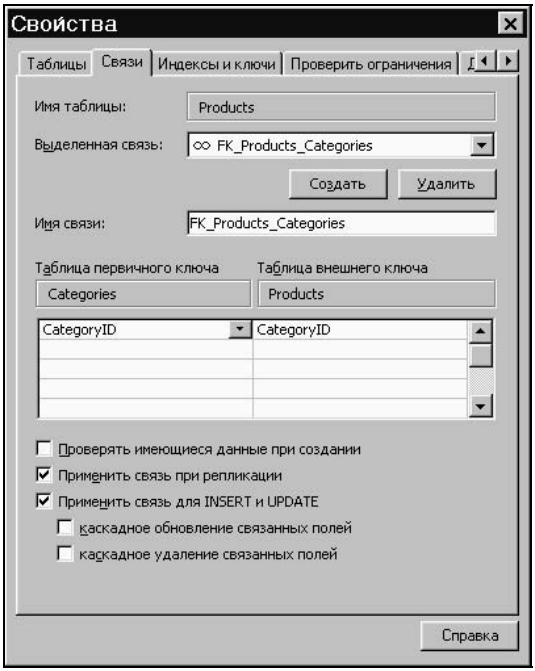


Рис. 17.15. Диалоговое окно **Свойства**, вкладка **Связи**

Флажок **CLUSTERED** для первичного ключа должен быть установлен, т. к. это кластеризованный индекс.

Замечание

Кластеризованный индекс (clustered index) определяет порядок физического следования строк в таблице, поэтому он может быть только один, и его предпочтительнее использовать, если часто приходится выбирать не по одной строке, а по целому диапазону строк — в этом случае кластеризованный индекс работает быстрее, чем некластеризованный. Если этот флажок недоступен, значит, в таблице уже есть поле, в котором определен кластеризованный индекс.

Поле **Выделенный индекс** (Selected index) содержит список всех индексов в таблице. Чтобы добавить новый индекс, нужно:

1. Нажать кнопку **Создать** (New). При этом в поле **Выделенный индекс** (Selected index) появится имя индекса, автоматически созданное системой.
2. В списке **Имя столбца** (Column name) выбрать поля, которые будут составлять индекс (максимальное число полей в составном индексе 16), и задать порядок сортировки данных в индексе (по возрастанию или по убыванию).
3. Ввести имя индекса в поле **Имя индекса** (Index name).
4. Дополнительно можно определить другие свойства. Например, если вы хотите сделать индекс уникальным, нужно установить флажок **UNIQUE**, выбрать переключатель **Индекс** (Index) и установить флажок **Пропускать повторения** (Ignore duplicate key).

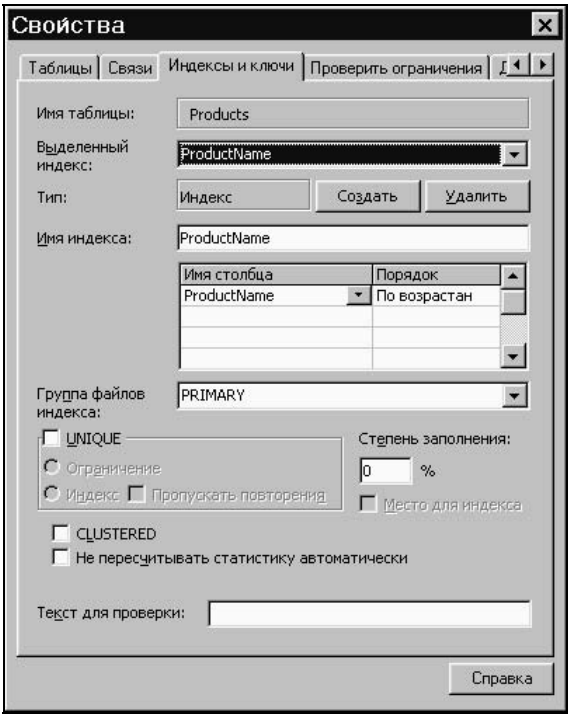


Рис. 17.16. Диалоговое окно **Свойства**, вкладка **Индексы и ключи**

Чтобы удалить индекс, нужно выбрать его из списка и нажать кнопку **Удалить** (Delete).

Ограничение типа UNIQUE на поле таблицы создается так же, как уникальный индекс, только нужно выбрать переключатель **Ограничение** (Constraint) (он выбирается по умолчанию). Такое ограничение можно ввести для поля, в котором разрешены значения Null, хотя первичным ключом такое значение быть не может.

Параметр **Степень заполнения** (Fill factor) определяет максимальный процент заполнения индексной страницы данными. Этот параметр может влиять на производительность и используется при тонкой настройке приложения, по умолчанию его значение равно 0.

На вкладке **Проверить ограничения** (Check Constraints) можно вводить и изменять ограничения типа проверки (CHECK), описывающие правила проверки значений полей. В поле со списком **Выделенное ограничение** (Selected constraint) можно выбрать одно из ограничений, которые уже введены для таблицы. На рис. 17.17 отображено ограничение CHECK на поле `Цена` (UnitPrice). Каждое ограничение имеет имя, состоящее из трех частей:

- префикс, который определяет тип ограничения, в данном случае это `CK_`;
- имя таблицы;
- имя поля.

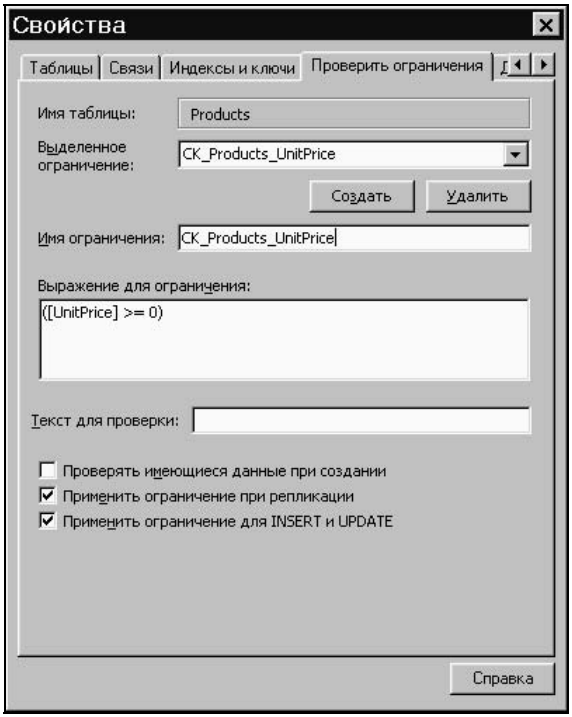


Рис. 17.17. Диалоговое окно **Свойства**, вкладка **Проверить ограничения**

Обычно префикс и имя таблицы формируются автоматически, в процессе ввода нового ограничения, последняя часть имени добавляется вручную в поле **Имя ограничения** (Constraint name).

Поле **Выражение для ограничения** (Constraint expression) служит для ввода выражения, которое и определяет ограничение. При написании выражения используется синтаксис языка Transact-SQL сервера, который отличается от синтаксиса SQL Access. *(Для справки обращайтесь к разделу **Transact-SQL Reference** справочной системы Access.)*

Флажки в нижней части вкладки имеют то же назначение, что и на вкладке **Связи** (Relationships).

Чтобы ввести новое ограничение, нужно нажать кнопку **Создать** (New), ввести выражение в поле **Выражение для ограничения** (Constraint expression) и добавить к значению в поле **Имя ограничения** (Constraint name) имя поля, на которое накладывается ограничение.

Чтобы удалить уже существующее ограничение, нужно выбрать его из списка в поле **Выделенное ограничение** (Selected constraint) и нажать кнопку **Удалить** (Delete).

Создание триггеров

Триггеры являются особой разновидностью хранимых процедур, которые активизируются не по прямому вызову пользователя, а при наступлении определенного со-

бытия в базе данных, а именно — вставки, удаления или модификации записи в таблице. Триггеры привязаны к конкретной таблице. Каждая таблица может иметь до трех триггеров — по одному на каждый вид операции с данными (INSERT, UPDATE, DELETE). Но один триггер может также обрабатывать и два и все три события.

Триггер может быть использован для соблюдения определенных правил модификации данных. При нарушении этих правил он может откатить все произведенные изменения и вернуть данные в исходное состояние. Кроме того, триггер может быть использован для автоматического осуществления каких-либо сопутствующих операций, например ведения журнала изменений данных, или автоматического обновления некоторых итоговых значений (например, значения поля, в которое вносится количество сотрудников в фирме).

Чтобы создать триггер для таблицы, нужно:

1. Щелкнуть по ярлыку **Таблицы** (Tables) на панели объектов в окне базы данных.
2. Щелкнуть правой кнопкой мыши по таблице, например, `Orders` (Заказы).
3. В контекстном меню выбрать команду **Триггеры** (Triggers). Появится диалоговое окно **Триггеры для таблицы** (Triggers for Table) со списком триггеров. Если триггеров для таблицы нет, то это окно будет выглядеть так, как показано на рис. 17.18. Нажмите кнопку **Создать** (New). Откроется окно редактора кода с шаблоном текста программы (рис. 17.19).

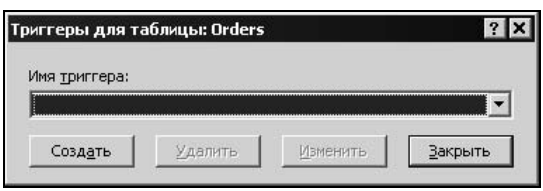


Рис. 17.18. Диалоговое окно со списком триггеров таблицы

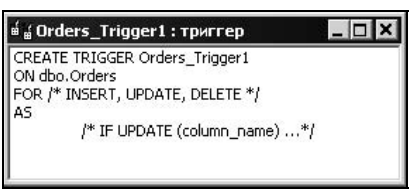


Рис. 17.19. Окно редактора текста триггера

4. Нужно определить тип операции с данными (INSERT, UPDATE, DELETE) и вставить в этот шаблон текст предложений SQL, которые будут выполнять требуемые действия. *(О языке SQL написано в разделах справки **Transact SQL Reference**.)* Чтобы сохранить код программы, выполните команду **Файл | Сохранить** (File | Save).

Для того чтобы изменить или удалить существующий триггер, необходимо в диалоговом окне **Триггеры для таблицы** выбрать из списка нужный триггер и нажать кнопку **Изменить** (Edit) или **Удалить** (Delete), соответственно.

Работа со схемами базы данных

Схемы баз данных (Database Diagrams) — это тип объектов, который присутствует только в проектах Access. Они являются аналогом схемы данных в базах данных Access, однако в проектах Access это понятие существенно расширено.

Во-первых, таких схем в одном проекте может быть несколько. Это значит, что всю совокупность таблиц можно разделить на логические области и построить для каждой из них свою схему данных. Это очень важно, т. к. количество таблиц в базе данных на сервере может быть очень велико, и, если разместить их все на одной схеме, она может быть просто необозрима. Кроме того, могут быть созданы схемы данных для определенных целей, когда нужно выделить и выразить специфический взгляд на подмножество таблиц базы (в базе данных Access сохраняется только последняя схема данных).

Во-вторых, сам инструментарий для создания схем баз данных более богат: он позволяет не только определять связи между существующими таблицами, но и изменять структуру и свойства таблиц, создавать новые таблицы и т. д. В этом смысле он является альтернативой обычному способу описания таблиц — с помощью режима Конструктора.

Вы можете выполнять операции как с отдельным объектом этой структуры, так и с группой объектов. Причем все изменения, которые вносятся в структуру базы данных, не сохраняются на сервере до тех пор, пока вы не сохраните измененную схему. Это удобно, когда нужно внести много изменений. Тогда сохранить можно только конечный результат, когда вы убедились, что все изменения согласованы.

Конструктор баз данных

Конструктор баз данных (Database Diagram Designer) обеспечивает графическое представление структуры базы данных SQL Server. На рис. 17.20 представлена схема базы данных `NorthwindCS`. Основными объектами схемы являются таблицы и связи. Действия, которые могут выполняться над объектами схемы, можно разбить на две группы:

- действия, изменяющие структуру базы данных;
- действия, изменяющие форму представления объектов на схеме.

При сохранении схемы данных и те и другие изменения сохраняются в системной таблице `dtproperties`.

К действиям, изменяющим структуру данных, относятся:

- создание новых таблиц;
- редактирование свойств существующих таблиц;
- определение индексов и ограничений для полей таблиц;
- создание связей между таблицами;
- удаление таблиц.

К действиям, изменяющим только форму представления схемы, относятся:

- добавление таблиц к схеме (уже существующих в базе данных);
- удаление таблиц из схемы (но не из базы данных);
- изменение расположения таблиц на схеме;
- изменение формы представления и размеров таблиц;
- масштабирование схемы данных;

- предварительный просмотр и печать схемы;
- копирование и перемещение таблиц.

Рассмотрим эти действия последовательно.

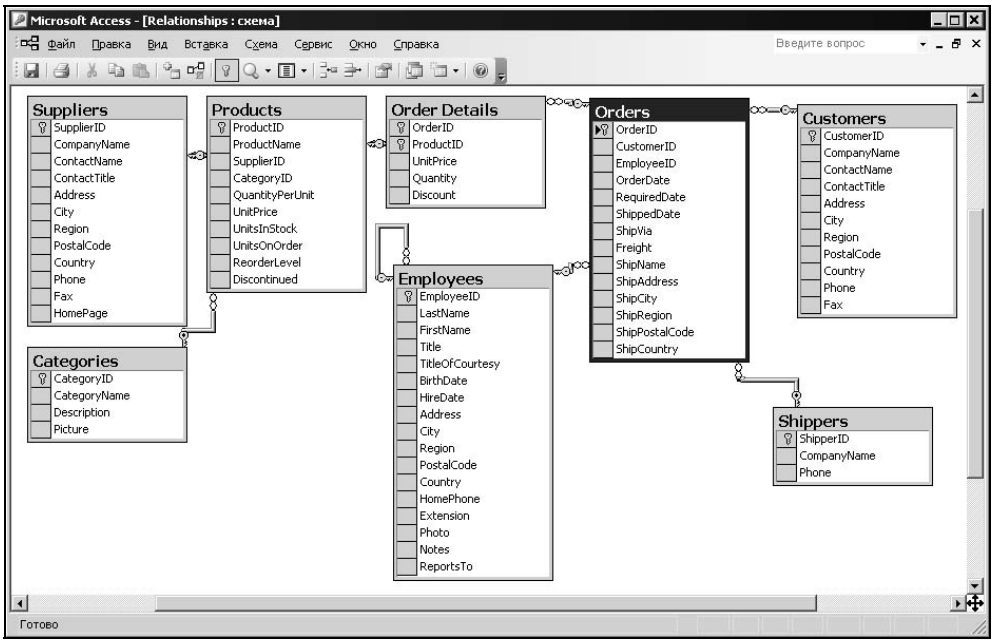


Рис. 17.20. Схема данных для базы данных `NorthwindCS`

Создание таблицы в окне Конструктора базы данных

Сначала откройте в режиме Конструктора, как любой другой объект Access, схему базы данных `Relationships` (Связи), которая присутствует в проекте NorthwindCS.adp. Для того чтобы создать с помощью Конструктора новую таблицу, нужно:

1. Щелкнуть правой кнопкой мыши по свободному месту в окне Конструктора и выбрать в контекстном меню команду **Новая таблица** (New Table). Появится диалоговое окно **Выбор имени** (Choose Name), запрашивающее имя таблицы. Мы предлагаем создать новую таблицу `Departments` (Отделы) в схеме данных, представленной на рис. 17.20, которая будет содержать подразделения фирмы `Northwind`.
2. В диалоговом окне **Выбор имени** (Choose Name) введите имя новой таблицы `Departments` и нажмите кнопку **ОК**. Появится таблица свойств с пустыми полями. Заполните эти поля так, как показано на рис. 17.21.
3. Определите первичный ключ для таблицы, выделив сначала поле `DepartmentID` (Код отдела), а затем нажав кнопку **Ключевое поле** (Primary Key) на панели инструментов.

4. Для того чтобы новая таблица выглядела на схеме так же, как и остальные таблицы, щелкните по ней правой кнопкой мыши и в контекстном меню выберите элемент **Имена столбцов** (Column Names) — это другая форма представления таблицы на схеме.



Рис. 17.21. Создание новой таблицы

Изменение свойств существующей таблицы

Рассмотрим изменение свойств таблицы на примере таблицы `Employees` (Сотрудники). Чтобы изменить свойства таблицы в схеме базы данных, нужно сначала изменить форму представления таблицы:

1. Щелкните правой кнопкой мыши по таблице `Employees` и выберите в контекстном меню форму представления **Свойства столбца** (Column Properties) или выделите таблицу `Employees`. Откройте список форм представления таблиц, нажав стрелку справа на кнопке **Перечень режимов таблицы** (Table Modes) на панели инструментов и затем выбрав в раскрывшемся списке значение **Свойства столбца** (Column Properties).

2. Появится таблица со столбцами, которые обычно отображаются в Конструкторе таблиц (рис. 17.22). Значения соответствующих свойств можно изменять так, как это описано в *разд. "Определение и изменение структуры таблицы" данной главы.* Для того чтобы увидеть остальные свойства полей таблицы, вызовите диалоговое окно свойств, нажав кнопку **Свойства** (Properties) на панели инструментов, и раскройте вкладку **Столбцы** (Columns). Чтобы получить информацию о свойствах таблицы, нажмите кнопку **Справка** (Help) в диалоговом окне свойств.

3. Вставьте в таблицу новое поле, содержащее идентификатор подразделения, в котором работает сотрудник. Для этого выделите поле `Title` (Должность) и нажмите клавишу <Ins> или кнопку **Добавить строки** (Insert rows) на панели инструментов (рис. 17.23). Появится пустая строка над строкой `Title`. Введите имя нового поля `DepartmentId` и тип данных `int`. После этого можно снова изменить представление таблицы на стандартное — **Имена столбцов** (Column Names).

Замечание

Обратите внимание, что как новые таблицы, так и измененные, помечаются на схеме звездочкой в области заголовка таблицы.

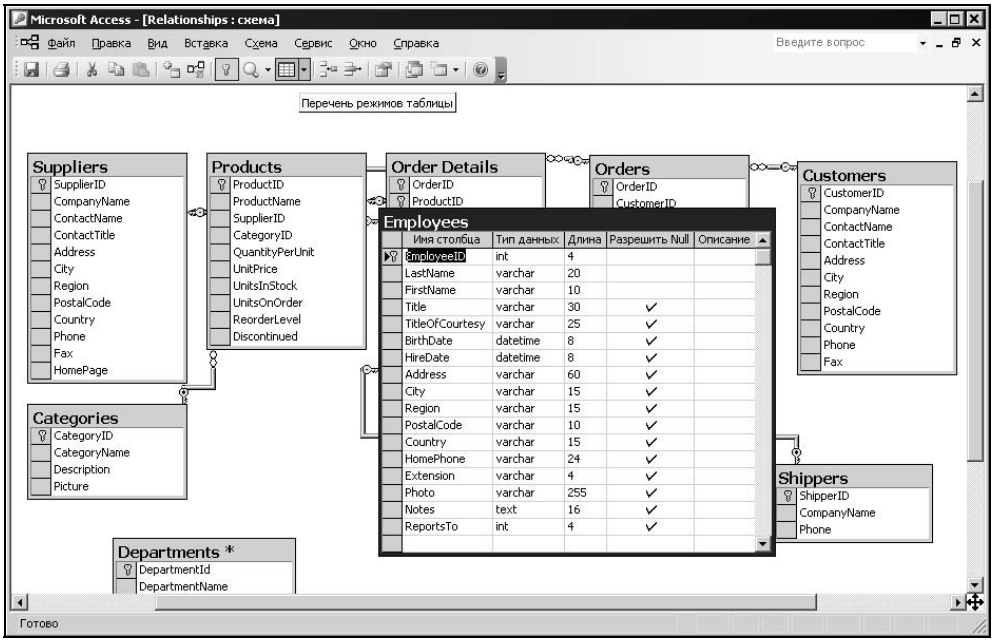


Рис. 17.22. Отображение свойств таблицы `Employees` на схеме данных

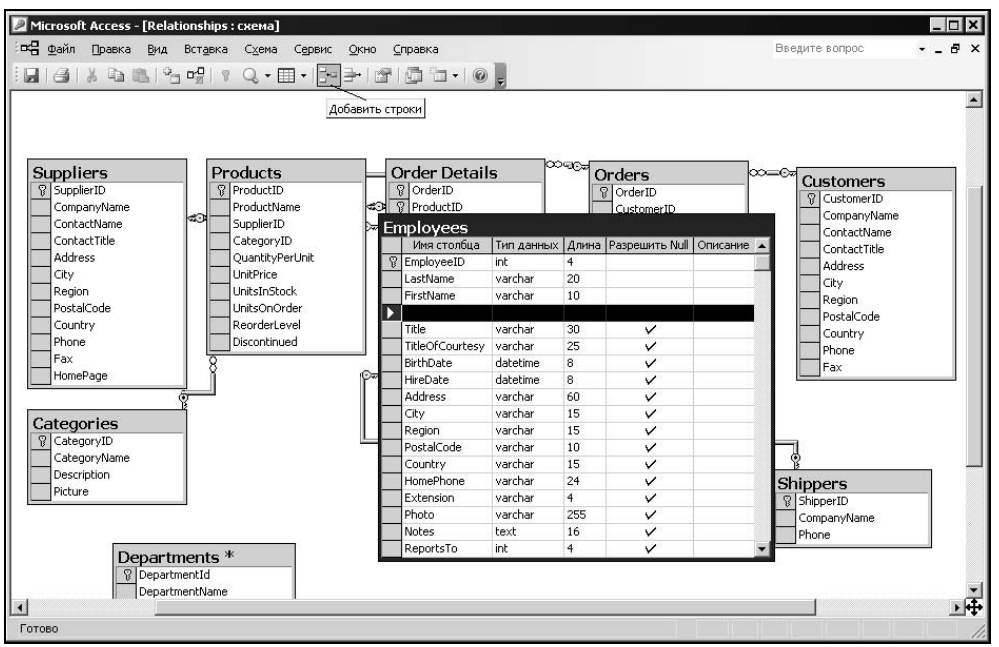


Рис. 17.23. Добавление строки в таблицу на схеме данных

Установление связей между таблицами

Теперь нужно связать новую таблицу `Departments` (Отделы) с таблицей `Employees` (Сотрудники). Для этого щелкните левой кнопкой по области выделения слева от поля `DepartmentID` (Код отдела) таблицы `Departments` и перетащите появившийся значок поля на одноименное поле таблицы `Employees`. Появится диалоговое окно **Установить связь** (Create Relationship) (рис. 17.24). В этом окне нужно сбросить флажок **Проверять имеющиеся данные при создании** (Check existing data on creation), т. к. поле `DepartmentID` в таблице `Employees` заведомо будет пустым. Нажмите кнопку **OK**.

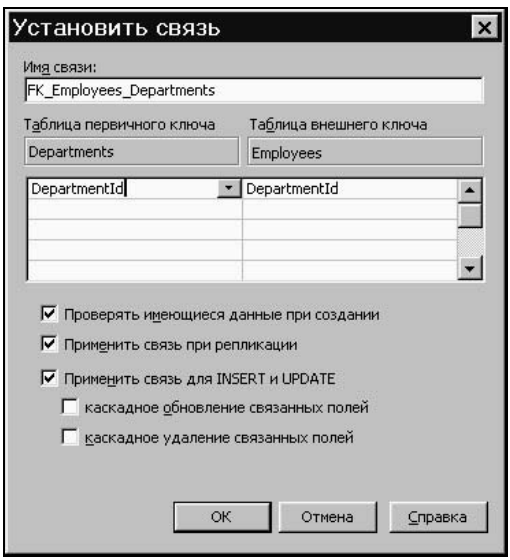


Рис. 17.24. Диалоговое окно **Установить связь**

Результат изменений отображается в схеме базы данных, как это показано на рис. 17.25. Тот конец связи, который соединен с таблицей, содержащей первичный ключ, помечается значком ключа. Конец связи, который соединен с таблицей, содержащей внешний ключ, может помечаться по-разному:

- для связей "один-к-одному" он тоже помечается ключом;
- для связей "один-ко-многим" он помечается значком "бесконечность".

Если вы подведете курсор мыши к линии связи, то появится метка связи (**Связь 'FK_Employees_Departments' между 'Departments' и 'Employees'**).

Настроить параметры связи можно и с помощью окна свойств. Чтобы открыть это окно, выделите связь на схеме и нажмите кнопку **Свойства** (Properties) на панели инструментов, или щелкните по связи правой кнопкой мыши и выберите в контекстном меню команду **Свойства** (Properties). Для вновь созданной связи автоматически устанавливается свойство проверки ссылочной целостности. Эту проверку можно отключить, если сбросить флажок **Применить связь для INSERT и UPDATE**

(Enable relationship for INSERT and UPDATE) на вкладке **Связи** (Relationship) диалогового окна **Свойства** (Properties).

Возможно создание на схеме данных рекурсивной связи, т. е. связи между полями одной и той же таблицы. Примером такой связи является связь в таблице `Employees` между полями `EmployeeID` и `ReportsTo`. Поле `ReportsTo` играет роль внешнего ключа, т. к. оно содержит идентификатор сотрудника, перед которым отчитывается данный сотрудник.

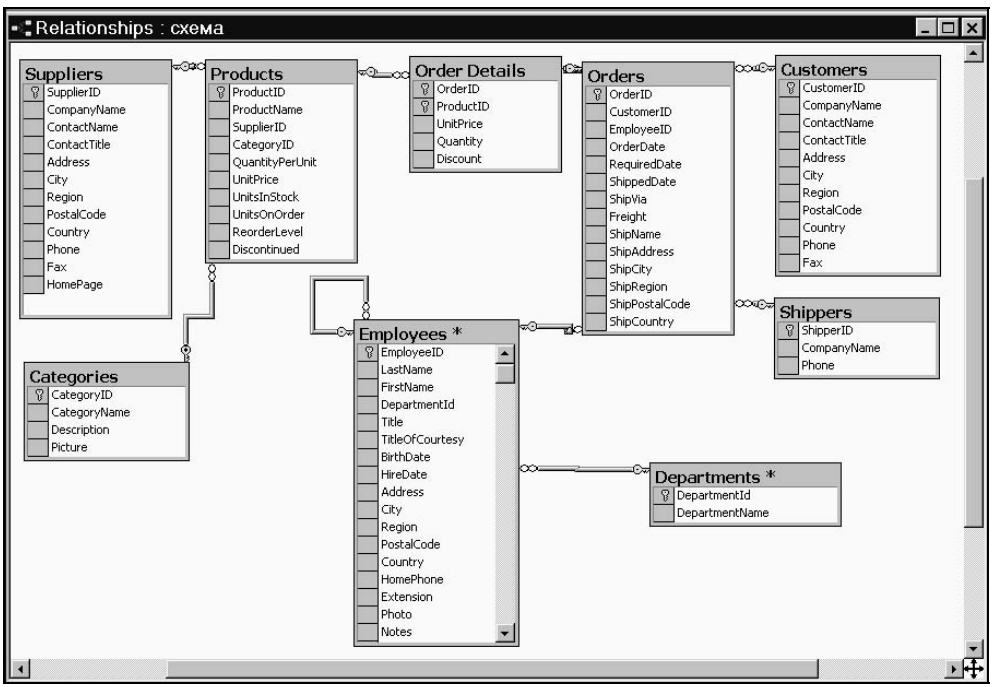


Рис. 17.25. Схема данных с новой таблицей `Departments`

Создание индексов и ограничений

Индексы и ограничения для таблиц в схеме базы данных создаются так же, как это описано ранее в *разд. "Работа с таблицами" данной главы.* Для создания в таблице индекса или ограничения необходимо открыть диалоговое окно **Свойства** (Properties). Создадим индекс для вновь созданного поля `DepartmentID` (Код отдела) в таблице `Employees` (Сотрудники). Щелкните правой кнопкой мыши по таблице `Employees` в окне схемы базы данных и выберите в контекстном меню команду **Свойства** (Properties), или выделите таблицу `Employees` и нажмите кнопку **Свойства** (Properties) на панели инструментов. Появится диалоговое окно **Свойства** (Properties). Раскройте вкладку **Индексы и ключи** (Indexes/Keys), нажмите кнопку **Создать** (New) и создайте новый индекс, как это было описано ранее.

Замечание

Индексы и ограничения не отображаются на схеме базы данных.

Сохранение и откат изменений в структуре базы данных

Сохранение изменений, которые выполнены в структуре таблиц и связей в схеме базы данных, выполняется после закрытия окна схемы. При этом выдается диалоговое окно **Сохранить** (Save), в котором перечисляются все измененные объекты. В нашем примере это окно выглядит так, как представлено на рис. 17.26. Пользователь имеет возможность проконтролировать, какие объекты подлежат изменению, и принять решение о том, сохранять или нет внесенные изменения. Можно даже сохранить информацию об измененных объектах в текстовом файле. Для этого нужно нажать кнопку **Сохранить как текст** (Save Text File). Текстовый файл, полученный при сохранении внесенных нами изменений, представлен на рис. 17.27.

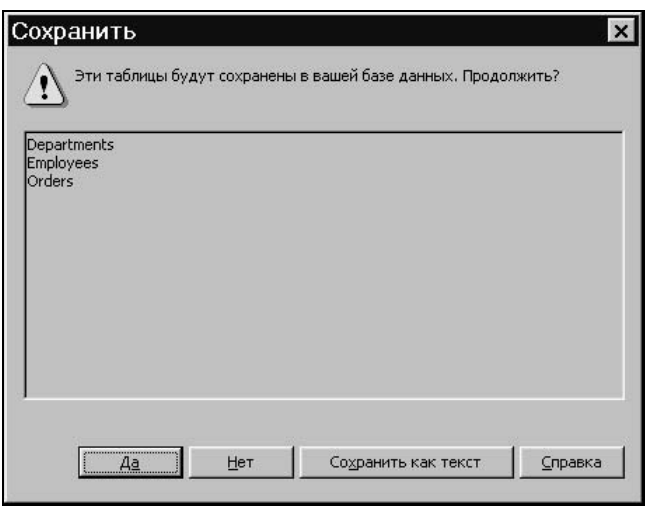


Рис. 17.26. Диалоговое окно **Сохранить**

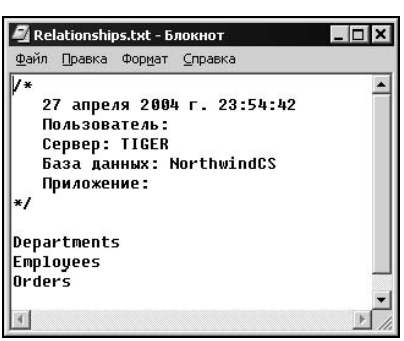


Рис. 17.27. Текстовый файл, содержащий сведения об измененных объектах

Если вы не хотите сохранять в базе данных те изменения, которые были внесены в структуру при работе со схемой данных, достаточно на вопрос **Сохранить изменения макета или структуры схемы '<имя схемы>'?**, который будет задан при закрытии окна схемы данных, нажать кнопку **Нет** (No). Произойдет откат всех изменений.

Замечание

При сохранении изменений, внесенных в схему базы данных, не всегда удается просто перенести все изменения в таблицы на сервере из-за ограничений сервера (например, в таблицах SQL Server нельзя поменять порядок полей). В этом случае таблица сначала удаляется из базы данных, а потом создается заново. Кроме того, если изменения вносятся в таблицу, которая связана с другими таблицами, то изменения могут затронуть и связанные таблицы. В этом случае в списке, который отображается в диалоговом окне **Сохранить** (Save), могут появиться таблицы, которые вы явно не изменяли.

Создание новой схемы базы данных и добавление в нее таблиц

Создание новой схемы базы данных выполняется так же, как и любого другого объекта базы данных Access — нужно открыть список схем, щелкнув мышью по ярлыку **Схемы баз данных** (Database Diagrams) на панели объектов окна базы данных. Затем нажать кнопку **Создать** (New). Откроется пустое окно Конструктора базы данных и в нем — диалоговое окно **Добавление таблицы** (Show Table) (рис. 17.28).

Выделите в списке нужную таблицу и нажмите кнопку **Добавить** (Add).

Если в схему добавляются таблицы, между которыми уже установлены связи (определены внешние ключи), то эти связи сразу же отображаются на схеме. Для любой таблицы, размещенной на схеме базы данных, очень легко отобразить все связанные с ней таблицы. Для этого достаточно щелкнуть по этой таблице правой кнопкой мыши и в контекстном меню выбрать команду **Добавить связанные таблицы** (Add Related Tables). Мы это проделали для таблицы `Orders` (Заказы). Результат представлен на рис. 17.29.

Изменение режима таблиц и расположения их на схеме базы данных

Для того чтобы удобно расположить таблицы в окне Конструктора базы данных, можно щелкнуть правой кнопкой мыши на поле схемы и выбрать в контекстном меню команду **Схема макета** (Layout Diagram). Таблицы автоматически будут размещены в окне так, как это представлено на рис. 17.30.

Можно применить аналогичную операцию для группы таблиц. Для этого сначала необходимо выделить требуемые таблицы. Чтобы выделить группу таблиц, достаточно "нарисовать" мышью прямоугольник вокруг этих таблиц, причем таблицы могут попадать в выделяемую область только частично. Выделенные таблицы можно определить по изменившемуся цвету области заголовка. Затем нужно щелкнуть правой кнопкой мыши по любой из выделенных таблиц и выбрать из контекстного меню команду **Выбор разметки** (Layout Selection).

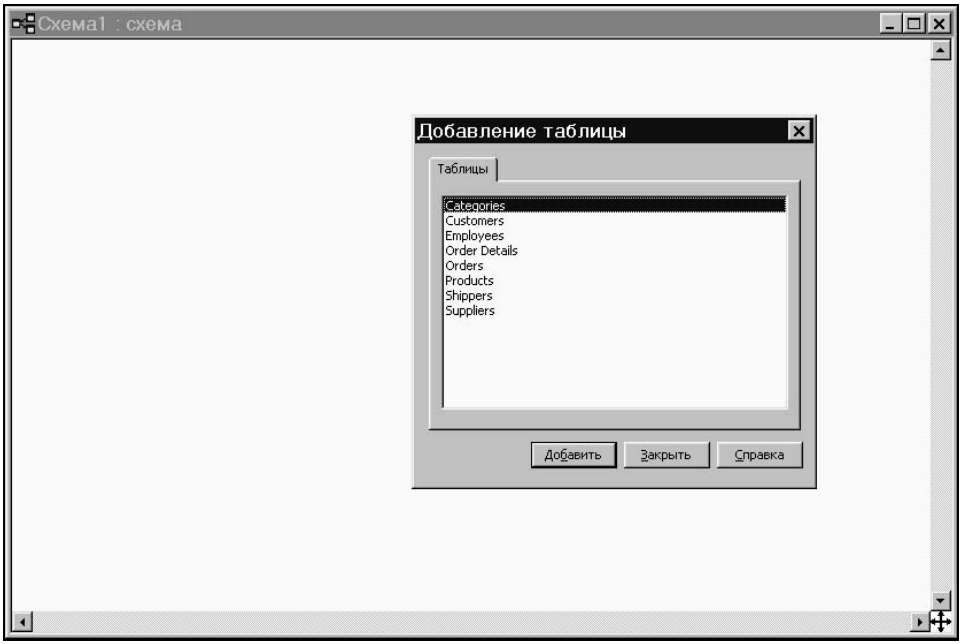


Рис. 17.28. Добавление таблиц в схему данных

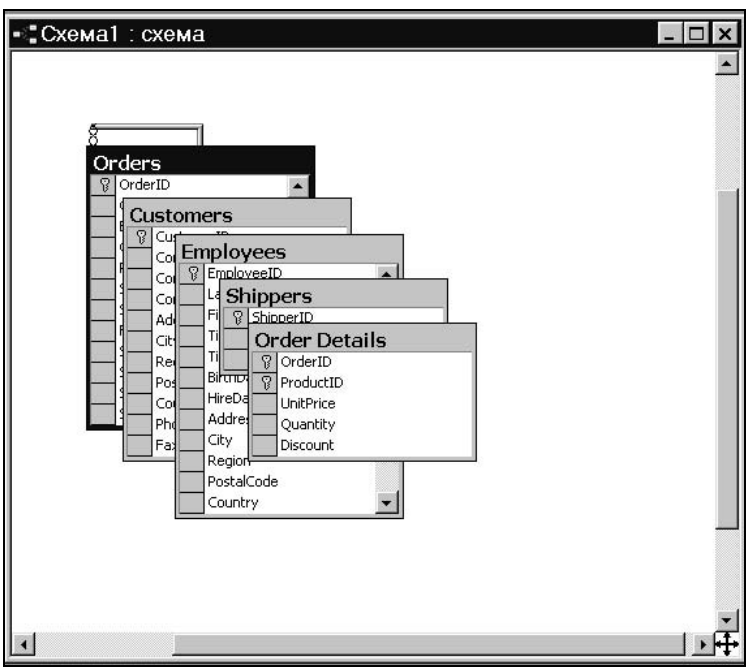


Рис. 17.29. Добавление связанных таблиц

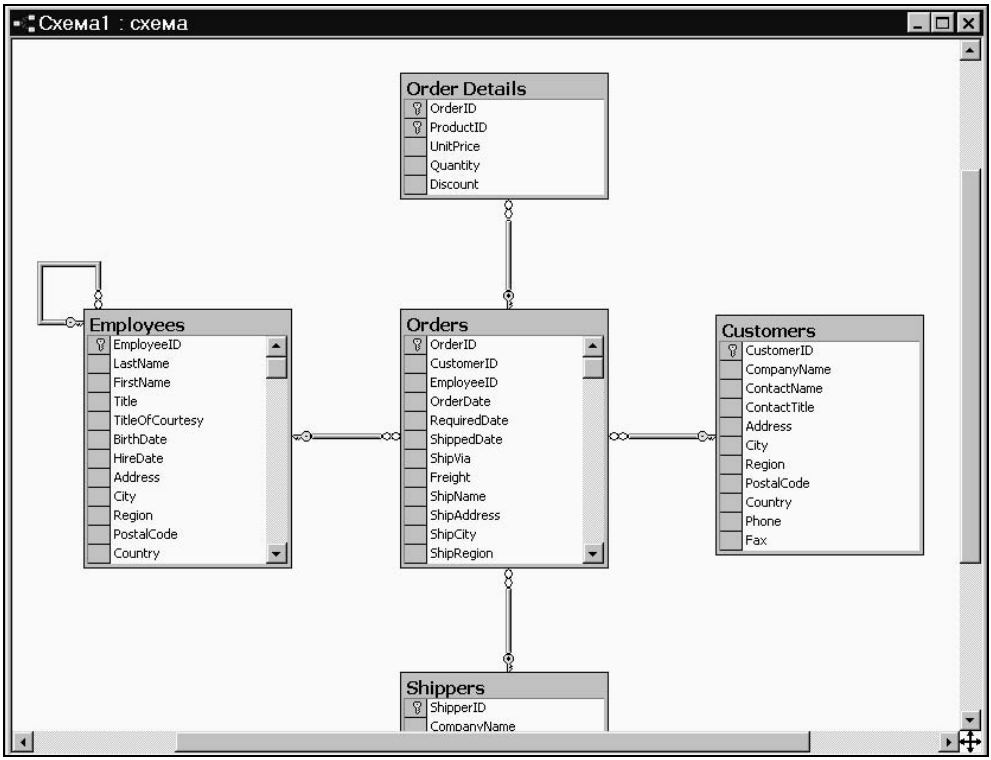


Рис. 17.30. Автоматическое размещение таблиц в окне Конструктора

Масштабировать можно или всю схему, или выделенную группу таблиц. Щелкнув правой кнопкой по окну схемы, нужно выбрать в контекстном меню команду **Масштаб** (Zoom) и указать масштаб отображения схемы в процентах (рис. 17.31). Обычно для редактирования свойств таблиц масштаб увеличивают, а для создания связей — уменьшают. Если нужно увидеть всю схему целиком, то удобно воспользоваться командой **Вписать диаграмму в окно** (Size Diagram to Window)

Существует несколько режимов представления таблиц на схеме данных, которые обеспечивают различную степень детализации свойств таблиц. Мы использовали уже два режима: **Имена столбцов** (Column names) и **Свойства столбца** (Column Properties). Но в зависимости от того, какие действия выполняются с таблицами, можно выбрать наиболее удобный режим. Они описаны в табл. 17.3.

Таблица 17.3. *Режимы таблицы на схеме данных*

Форма представления	Описание
Свойства столбца (Column Properties)	Представление в виде таблицы свойств. Эта форма удобна для редактирования свойств полей таблицы. Когда создается новая таблица, она отображается на схеме именно в такой форме

Таблица 17.3 *(окончание)*

Форма представления	Описание
Имена столбцов (Column Names)	Показываются только имена полей таблицы и обозначается ключевое поле. В таком виде показывается существующая таблица при добавлении ее в схему данных. Имена полей в этом представлении можно редактировать
Ключи (Keys)	Показываются только первичный и внешние ключи таблицы, а также столбцы, имеющие ограничение типа UNIQUE
Только имя (Name Only)	Показывается только заголовок таблицы
Новое представление (Custom)	Показываются свойства полей, но не все, а только выбранные пользователем

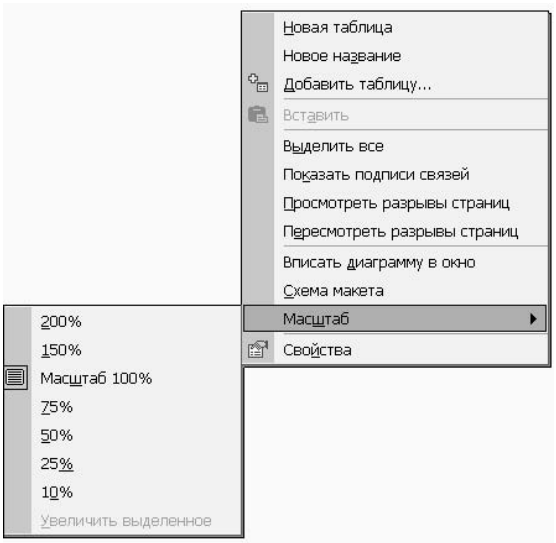


Рис. 17.31. Масштабирование схемы данных

На рис. 17.32 показаны таблицы на схеме данных в разных режимах: таблицы `Products`, `Categories`, `Suppliers`, `Orders` свернуты максимально — использован режим **Только имя**, таблица `Employees` отображена в режиме **Ключи**, таблица `Customers` — в режиме **Имена столбцов**, таблица `Shippers` — в режиме **Свойства столбца**, таблица `Order Details` — в режиме **Новое представление**.

Режим **Новое представление** является настраиваемым. По умолчанию в нем отображаются три столбца: имя поля, тип поля и возможность пустых значений. Чтобы настроить это представление:

1. Выделите таблицу в окне схемы базы данных.
2. Выберите команду меню **Схема | Изменить настраиваемое представление** (Diagram | Modify Custom View). Появится диалоговое окно **Выбор столбца** (Column Selection) (рис. 17.33).

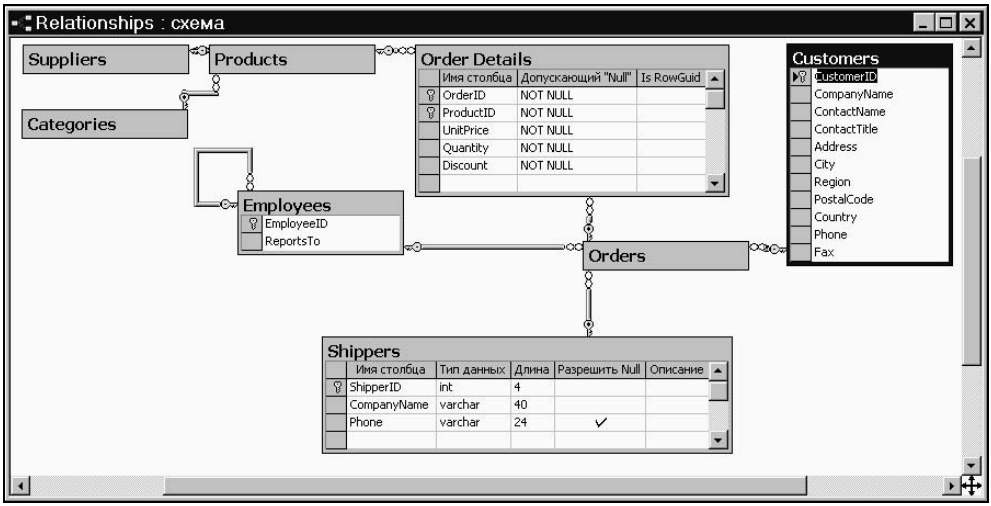


Рис. 17.32. Различные режимы представления таблиц

3. С помощью кнопок со стрелками перенесите из левой части окна в правую те свойства, которые вы хотите видеть при отображении таблицы на схеме.
4. Если необходимо изменить порядок следования столбцов свойств в таблице, воспользуйтесь кнопками со стрелками вверх и вниз и переместите имена полей в правом списке нужным образом.
5. Если вы хотите, чтобы эта форма представления использовалась во всех схемах баз данных, как **Новое представление**, установите флажок **Сохранить для использования по умолчанию** (Save as default). В противном случае такое представление будет действовать только в данной схеме.
6. Нажмите кнопку **OK**.

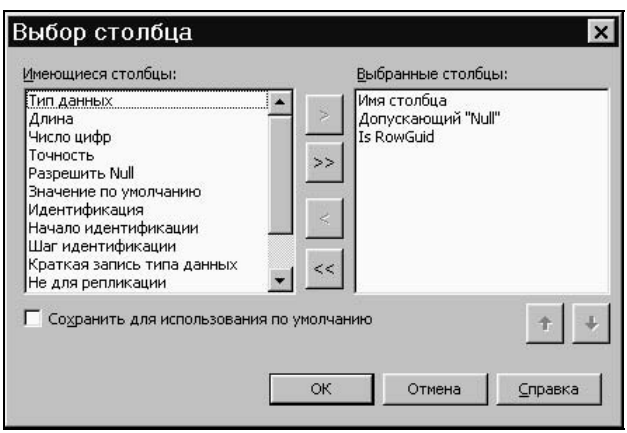


Рис. 17.33. Диалоговое окно **Выбор столбца**

Кроме таблиц и связей между ними, на схеме можно создавать надписи, которые могут содержать все необходимые комментарии к схеме. Для того чтобы создать надпись, нужно щелкнуть правой кнопкой мыши на поле схемы и выбрать в контекстном меню команду **Новое название** (New Label). На схеме появляется рамка, в которую можно ввести текст (рис. 17.34).

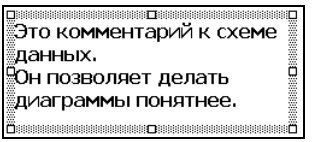


Рис. 17.34. Добавление надписей к схеме

Для того чтобы изменить шрифт надписи, нужно щелкнуть по ней правой кнопкой и выбрать в контекстном меню команду **Шрифт** (Font) или выделить надпись и воспользоваться командой меню **Схема | Шрифт для надписи** (Diagram | Set Label Font). Изменение шрифта действует на всю надпись. Нельзя одну часть надписи отформатировать одним шрифтом, а другую — другим.

Предварительный просмотр и печать схемы данных

Если схема данных содержит большое количество таблиц и не помещается на одну страницу, Access разбивает ее на страницы. Перед печатью можно просмотреть, как

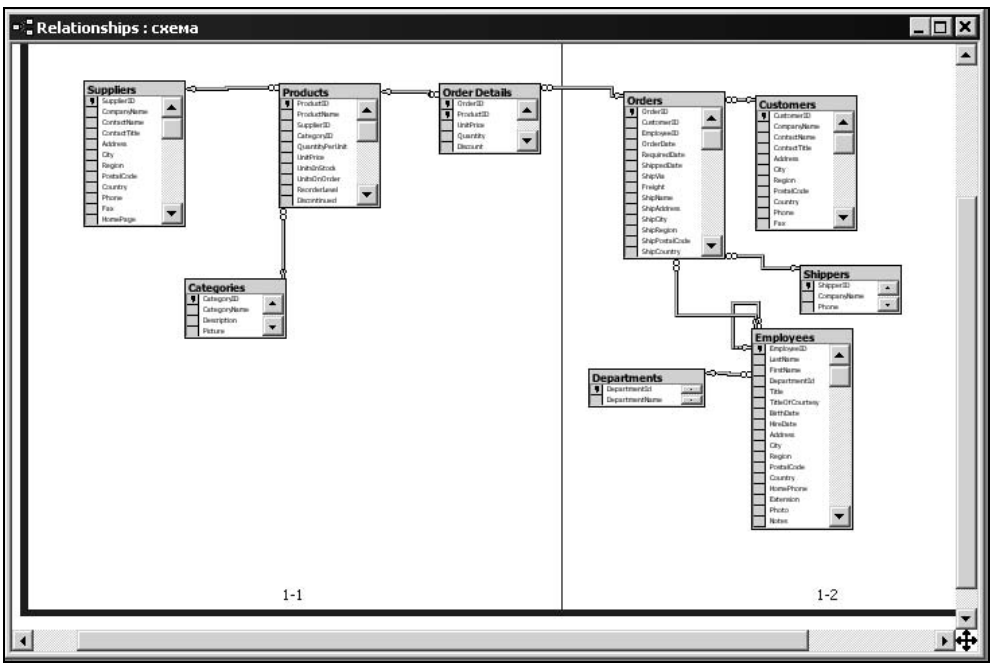


Рис. 17.35. Схема данных в режиме просмотра разрывов страниц

выполняется разбиение. Для этого нужно выполнить команду меню **Схема | Просмотреть разрывы страниц** (Diagram | View Page Breaks). Тогда на схеме появляются синие разделительные линии и номера страниц, как это представлено на рис. 17.35. Перемещая объекты, можно добиться, чтобы они целиком помещались на странице, а меняя форму представления, получить тот вид схемы, который обеспечит нужную степень детализации информации.

После того как вы убедились, что вид схемы вас удовлетворяет, выберите команду меню **Файл | Печать** (File | Print), чтобы напечатать эту схему.

Копирование и перемещение таблиц между схемами данных

Таблицы и группы таблиц легко копируются из одной схемы в другую через буфер обмена. При этом можно копировать как отдельные таблицы, так и группы таблиц.

Для перемещения таблиц из одной схемы в другую необходимо открыть на экране сразу обе схемы, выделить нужную таблицу или группу таблиц и перетащить их мышью в новую схему.

С помощью операций копирования и перемещения можно разделить одну большую схему на несколько более простых и, наоборот, из нескольких подсхем создать одну общую схему.

Также через буфер можно копировать поля из одной таблицы в другую, причем другая таблица может быть как в той же самой схеме, так и в другой.

Удаление таблиц из схемы и из базы данных

Чтобы удалить таблицу из схемы данных, нужно выделить эту таблицу и нажать кнопку **Скрыть таблицу** (Hide Table) на панели инструментов или выбрать соответствующую команду в контекстном меню. При этом таблица больше не отображается в схеме базы данных, но не удаляется из самой базы данных. Если таблица была изменена, то прежде чем удалить ее из схемы, Access запросит, хотите ли вы сохранить изменения в базе данных. Если вы ответите **Нет** (No), то таблица сохранится в памяти Access в измененном состоянии, и эти изменения перенесутся в базу данных только когда вы сохраните схему базы данных при закрытии.

Для того чтобы удалить таблицу не только из схемы, но и из базы данных, нужно выделить эту таблицу и выполнить из контекстного меню команду **Удалить таблицу из базы данных** (Delete Table from Database). При этом таблица на самом деле удаляется из базы данных только тогда, когда вы сохраните схему базы данных, из которой вы ее удаляли. Если не сохранять изменения в схеме данных, то таблица удалена из базы данных не будет.

Работа со схемами базы данных в многопользовательской среде

Схемы базы данных хранятся на SQL Server, и несколько пользователей могут работать с одной схемой, если они имеют на это права. При этом, когда вы попытаетесь

сохранить изменения в схеме базы данных, может оказаться, что другой пользователь сделал изменения в схеме или в базе данных, которые не были отражены в вашей схеме. Access сообщит вам о такой ситуации. В этом случае вы можете выполнить одно из трех действий:

- сохранить свою схему и все изменения в структуре базы данных, которые перечислены в списке в диалоговом окне **Сохранить** (Save). Это действие может повлиять на другие схемы, содержащие те же таблицы;
- закрыть диаграмму, не сохраняя никаких изменений. Тогда, когда вы ее снова откроете, она окажется согласованной с базой данных;
- сохранить список изменений в текстовом файле.

Если вы решили сохранить сделанные вами изменения в базе данных, то происходит следующее:

1. Если другой пользователь удалил объекты, которые присутствуют в вашей схеме, они будут созданы вновь.
2. Сохраняются триггеры, которые были добавлены к таблицам.
3. Удаляются объекты, которые были удалены из вашей схемы, даже если другой пользователь изменил эти объекты.

Работа с запросами

Запросы в проектах Access являются более сложным понятием, чем в базах данных Access. Это понятие включает несколько разных объектов SQL Server: **Представления** (Views), **Сохраненные процедуры** (Stored Procedures) и **Функции, определенные пользователем** (User-defined Function). Все эти объекты могут быть созданы с помощью Конструктора запросов.

Конструктор запросов

Окно Конструктора запросов показано на рис. 17.36. Как видите, оно похоже на окно Конструктора запросов в базе данных Access. На верхней панели размещаются таблицы или запросы (в данном случае таблицы), являющиеся источником данных. Средняя панель представляет собой бланк запроса, который является трансформацией бланка запроса в базе данных Access (столбцы стали строками, а строки — столбцами). Различие между ними небольшое. Например, появился отдельный столбец **Псевдоним** (Alias), в котором можно задать псевдоним названия поля. Нижняя панель содержит текст инструкции SQL, соответствующий формируемому запросу.

Не всегда все панели видны на экране. Для управления отображением панелей служат три кнопки на панели инструментов (рис. 17.37).

Окно Конструктора запросов появляется, когда вы создаете новое представление, редактируете старое или создаете инструкцию SQL в свойстве **Источник записей** (Record Source) формы или отчета с помощью кнопки Построителя. Для добавления таблицы на панель Конструктора используется диалоговое окно **Добавление таблицы** (Show Table), которое аналогично представленному на рис. 17.28, но содержит три вкладки — **Таблицы** (Tables), **Представления** (Views) и **Функции** (Function). Окно **До-**

бавление таблицы (Show Table) появляется при нажатии на кнопку **Добавить таблицу** (Show Table) на панели инструментов. Создание представления аналогично созданию запроса в базе данных Access, поэтому мы не будем здесь описывать этот процесс полностью, а покажем только отличия от аналогичного процесса в базе данных. *(О создании запросов в базе данных см. гл. 4 и 8.)*

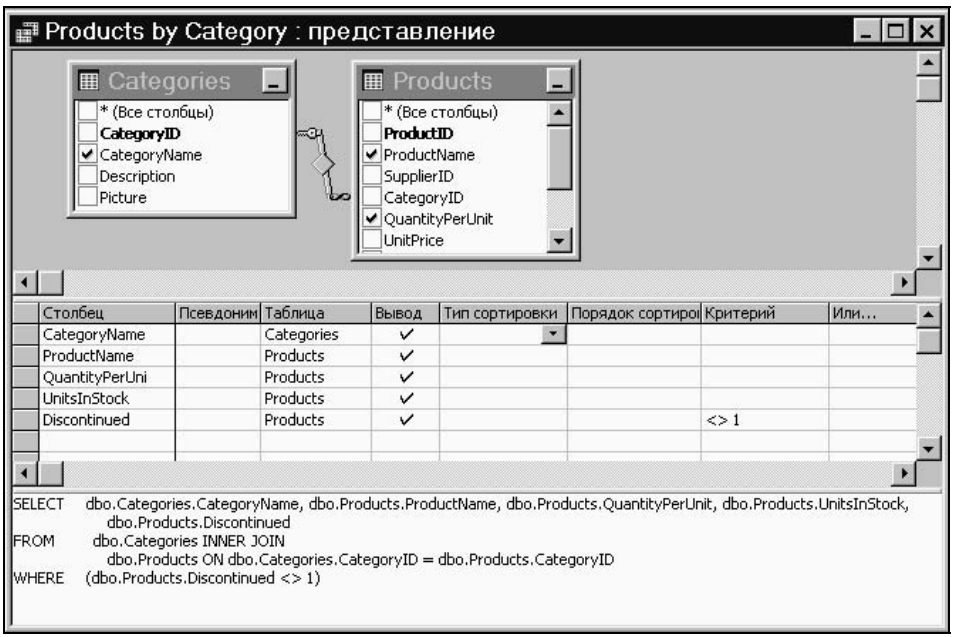


Рис. 17.36. Окно Конструктора запроса

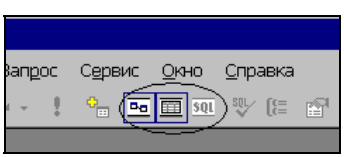


Рис. 17.37. Кнопки управления панелями Конструктора запросов

Для того чтобы включить те или иные поля в представление, их помечают в таблицах путем установки флажка у каждого поля. Одновременно эти поля появляются в бланке запроса и формируется инструкция SQL на панели SQL. Чтобы выбрать сразу все поля таблицы, нужно установить флажок *** (все столбцы)** (All Columns). Кроме того, поля могут перетаскиваться в бланк запроса мышью, как это делается в окне Конструктора запросов в базе данных Access.

Формирование критериев отбора записей аналогично формированию критериев в базе данных Access, однако выражения подчиняются синтаксическим правилам языка SQL Server.

Способы объединения таблиц в представлении

Так же как и в запросах, в базе данных Access можно создавать разные типы объединений таблиц:

- внутренние, когда из обеих таблиц отбираются только те записи, у которых совпадают значения связанных полей;
- внешние, когда в одной из таблиц выбираются все записи, а в другой — только связанные.

Существует три типа внешних объединений:

- *левое внешнее объединение*, когда выбираются все строки из левой таблицы;
- *правое внешнее объединение*, когда выбираются все строки из правой таблицы;
- *полное внешнее объединение*, когда выбираются строки из обеих таблиц.

По умолчанию используется внутреннее объединение таблиц. Если нужно изменить тип объединения, щелкните правой кнопкой мыши по линии связи (удобнее всего по ромбику в центре связи) и выберите соответствующую команду в контекстном меню. В представлении `Products by Category` (Товары по типам) (см. рис. 17.36) это будут команды:

- **Выделить все строки с Categories** (Select All Rows from Categories);
- **Выделить все строки с Products** (Select All Rows from Products).

Или можно выбрать из контекстного меню команду **Свойства** (Properties) и открыть диалоговое окно **Свойства** (Properties) (рис. 17.38).

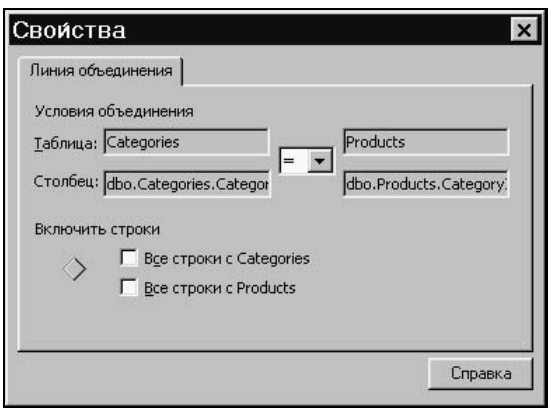


Рис. 17.38. Диалоговое окно свойств связи полей таблиц в представлении

Флажки в группе **Включить строки** (Include rows) позволяют установить любой тип внешнего соединения между таблицами. При изменении типа соединения меняется значок на связи — дополняется ромб справа или слева до квадрата.

Как видно из рис. 17.38, записи в таблицах могут связываться не только по условию равенства связанных полей. Для связи можно использовать любой из операторов сравнения (>, <, >=, <= или <>). При изменении оператора сравнения он отобра-

жается на значке связи, таким образом, тип связи очень хорошо идентифицируется визуально. Кроме того, когда вы подводите курсор мыши к связи, появляется всплывающая подсказка, указывающая тип объединения, условие объединения и связанные поля (рис. 17.39).

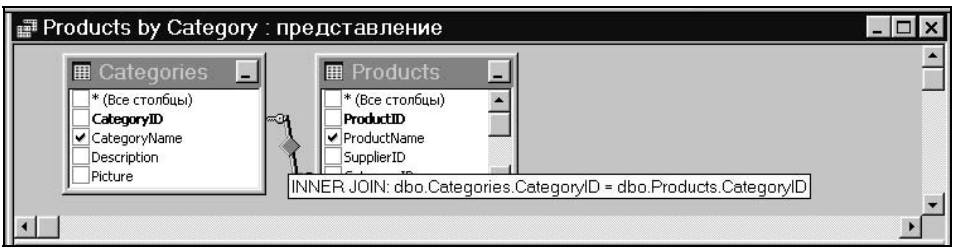


Рис. 17.39. Описание связи в окне Конструктора запросов

Группировка записей и суммирование значений в представлении

Группировка записей в представлениях и вычисления с использованием статистических функций, в том числе суммирование, выполняются так же, как в запросах базы данных Access *(см. разд. "Создание и изменение запроса с помощью Конструктора" гл. 4)*.

Вместо строки **Групповая операция** (Totals) в бланке запроса Конструктора запросов в проекте присутствует столбец **Группировка** (Group by). По умолчанию этот столбец скрыт. Для того чтобы вывести его на экран, нужно нажать кнопку **Группировка** (Group By) на панели инструментов. Столбец **Группировка** (Group by) добавляется в бланк представления, и для всех полей, включенных в представление, этому полю

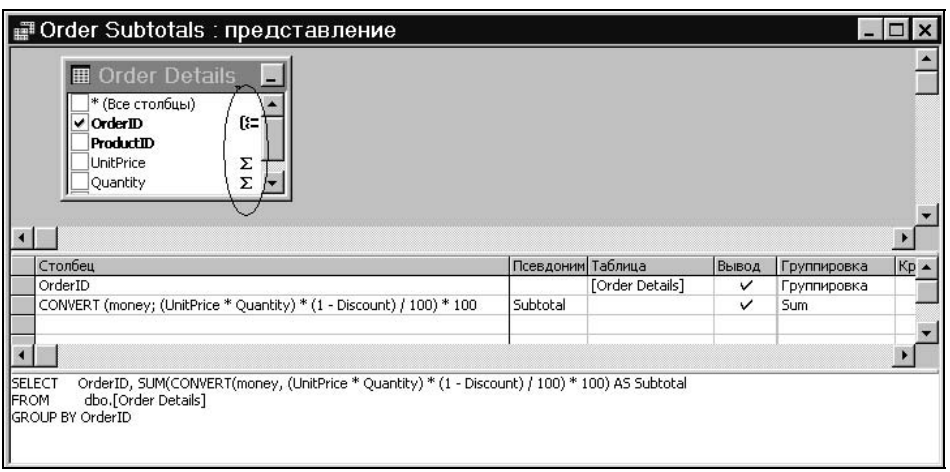


Рис. 17.40. Группировка и суммирование строк в представлении

присваивается значение **Группировка** (Group by). Можно затем в отдельных строках заменить значение (Group by) необходимыми статистическими функциями (сумма — `Sum`, среднее — `Avg`, число значений — `Count` и т. д.). В результате на нижней панели соответствующим образом изменится инструкция SQL. Чтобы ввести условие на выборку записей, можно использовать две разновидности инструкции `SELECT`:

- если в столбце **Группировка** (Group by) установлено значение параметра `WHERE`, тогда в инструкцию `SELECT` добавляется условие `WHERE` с критериями выборки;
- если столбец **Группировка** (Group by) имеет другие значения, тогда в инструкцию `SELECT` добавится условие `HAVING`.

В таблице появляются значки, указывающие, по какому полю будет выполнена группировка и какое поле будет суммироваться. Пример такого представления приведен на рис. 17.40.

Использование подчиненных представлений

Подчиненные представления обычно используются в предложении `WHERE` для определения условий выборки и в предложении `FROM`, если нужно определить некоторый результирующий набор в качестве источника данных для представления. Создать представление, содержащее подчиненное представление, можно либо путем ввода инструкции SQL на панели запроса SQL, либо с помощью бланка представления. Если вы недостаточно свободно владеете языком SQL, используйте второй способ.

Сначала создайте подчиненное представление. Для этого, получив на панели запроса SQL необходимую инструкцию и протестировав соответствующий запрос (используя кнопку **Запуск** (Run) на панели инструментов), скопируйте его в буфер.

Создайте второе представление и вставьте выражение из буфера в предложение `FROM` или `WHERE` на панели SQL. Вы можете получить инструкцию SQL, например, такого вида:

```
SELECT SupplierName, ProductName
FROM Products
WHERE SupplierID IN
     (SELECT SupplierID
      FROM Suppliers
      WHERE (Country = 'UK'))
```

Использование параметров в инструкциях SQL

В представлении (в отличие от запросов Access) параметры применять нельзя, однако их можно использовать в инструкциях SQL, которые являются источниками записей в формах и отчетах. То есть можно в свойстве **Источник записей** (RecordSource) формы или отчета ввести предложение SQL, которое содержит один или несколько параметров. Эти параметры будут либо запрашиваться у пользователя при открытии формы, либо устанавливаться равными текущему значению заданного элемента управления формы. Покажем на примере, как создаются такие формы.

Возьмем форму `Products` (Товары) и будем выбирать в ней только те продукты, цена которых выше заданного значения. Это значение и будет параметром.

1. Откройте форму `Products` в режиме Конструктора и диалоговое окно свойств формы. В качестве источника записей для этой формы используется таблица `Products`. Нажмите кнопку Построителя справа от ячейки **Источник записей** (RecordSource). Появится сообщение, спрашивающее, действительно ли вы хотите построить запрос на базе таблицы. Нажмите кнопку **Да** (Yes). Откроется окно Конструктора запросов. На панели схемы запроса уже размещена таблица `Products`.
2. С помощью кнопок на панели инструментов отобразите все три панели запроса: схему, бланк и запрос SQL. Переместите границы панелей таким образом, чтобы их все хорошо видеть.
3. В бланке запроса уже стоит символ звездочки (*), который означает, что в результирующий набор включаются все поля таблицы. Установите флажок с названием поля `UnitPrice` (Цена) на панели схемы запроса. Это поле появится в бланке запроса. Сбросьте флажок в столбце **Вывод** (Output), чтобы не включать это поле в результирующий набор (оно уже включено символом звездочки). В столбце **Критерий** (Criteria) введите выражение `>?` (рис. 17.41). Символ `?` в предложении `WHERE` и является обозначением параметра.
4. Установите флажок у поля `ProductName` (Марка), чтобы добавить его в запрос, и сбросьте флажок в столбце **Вывод** (Output) бланка запроса. В столбце **Тип сортировки** (Sort Type) выберите из списка **По возрастанию** (Ascending), чтобы отсортировать записи по наименованию продукта в прямом алфавитном порядке (см. рис. 17.41).

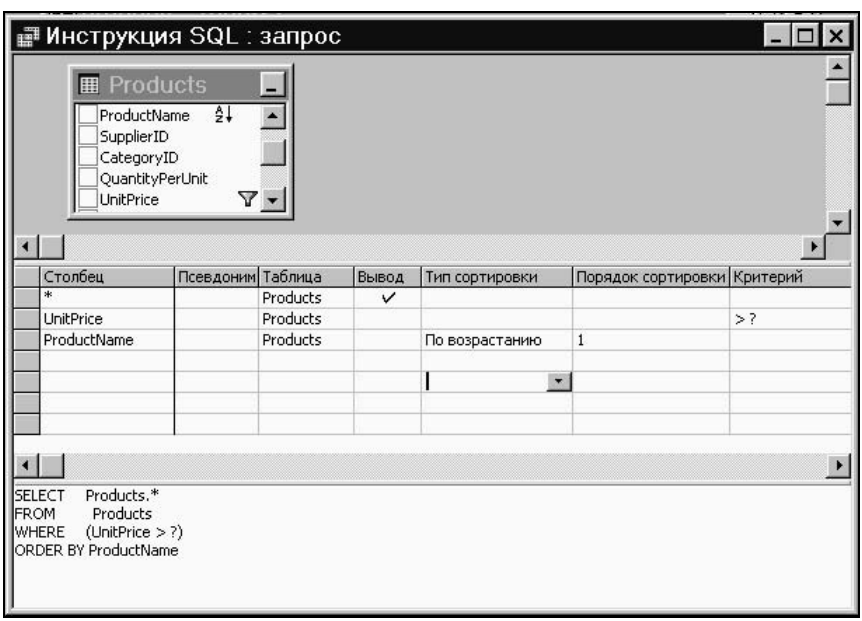


Рис. 17.41. Создание запроса с параметрами

5. Закройте окно Конструктора запроса. Подтвердите сохранение внесенных изменений.
6. Переместите курсор в ячейку **Входные параметры** (Input Parameters) диалогового окна свойств формы и введите выражение `Price money=[Введите пороговую цену]`, где `Price` — имя параметра, `money` — тип данных. Выражение справа от знака равенства определяет строку, которая будет выводиться для пользователя при запросе параметра (рис. 17. 42). Вместо этого выражения можно использовать ссылку на элемент управления формы, например `Form![Критерии выборки]![Цена]`.

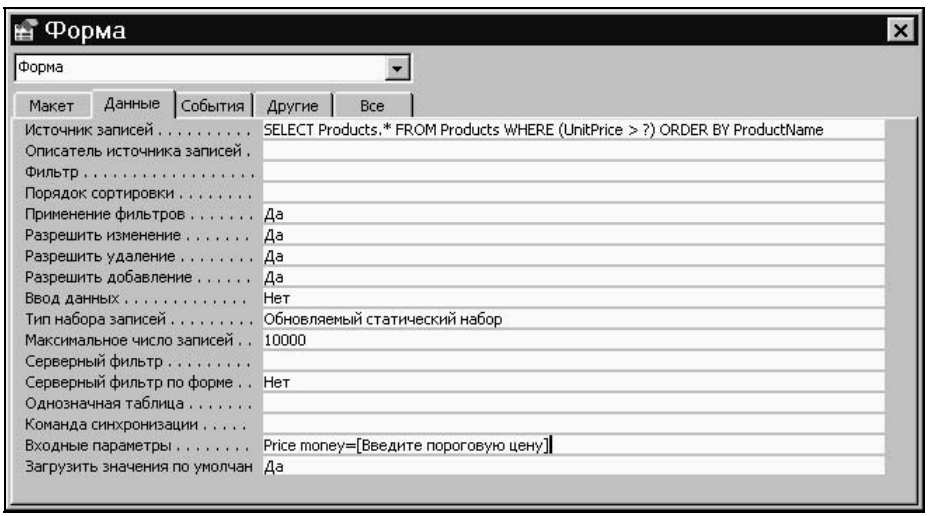


Рис. 17.42. Задание параметра в форме

Диалоговое окно свойств представления

Если вы щелкните правой кнопкой мыши на любой панели в окне Конструктора представления, появится диалоговое окно свойств представления (рис. 17.43). Оно содержит несколько параметров, которые влияют на способ выполнения запроса для формирования представления. На вкладке **Представление** (View):

- В поле **Представление** (View name) содержится имя представления.
- Поле **TOP** позволяет задать количество возвращаемых записей в результирующем наборе. По умолчанию при установке флажка в этом поле возвращаются первые 10 записей из исходной таблицы. Количество возвращаемых записей можно задать и в процентах, в этом случае нужно установить флажок в поле **PERSENT**.
- Флажок **Вывод всех столбцов** (Output all columns) позволяет включить в результирующий запрос все поля из всех таблиц запроса.
- Флажок **Привязать к схеме** (Bind to Schema) запрещает пользователям изменять базовые таблицы представления, которые могут привести к некорректности са-

мого представления. *(Подробнее о схемной привязке см. разд. "Использование индексированных представлений" данной главы.)*

- Флажок **Обновить как представления** (Update using view rules) указывает, что все операции обновления и вставки в представление транслируются в инструкции SQL, ссылающиеся на представление, а не на базовые таблицы представления непосредственно.

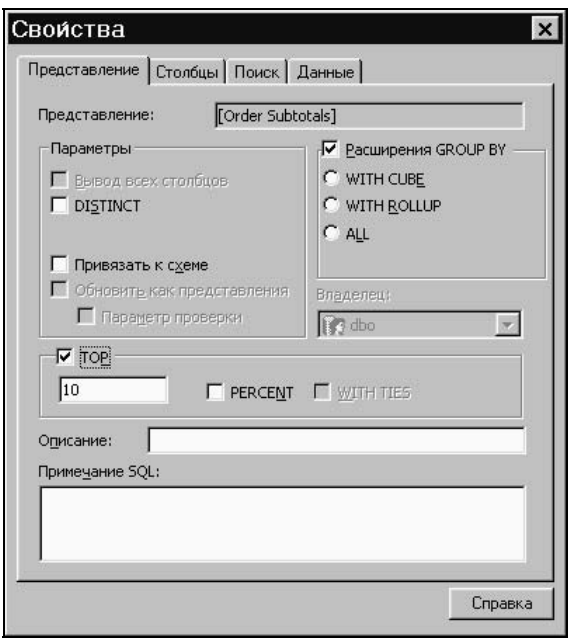


Рис. 17.43. Диалоговое окно свойств запроса

- Флажок **DISTINCT** позволяет исключить из результирующего набора повторяющиеся записи. Когда этот флажок установлен, в предложение SQL вставляется слово `DISTINCT`.
- Группа параметров **Расширения GROUP BY** доступна только, если в запросе SQL используются статистические функции.
  - Переключатель **WITH CUBE** позволяет добавить в результирующий набор записи, в которых суммируются значения в группах, задаваемых предложением `Group by` запроса. Заметим, что этой возможности очень не хватало в запросах базы данных Access. Чтобы проверить, что позволяет этот параметр, мы рекомендуем открыть в режиме Конструктора представление `Order Subtotals`, в котором суммируются записи в заказах с целью определения сумм заказов, и выбрать переключатель **WITH CUBE** в группе **Расширения GROUP BY**. Последние несколько строк результата такого запроса представлены на рис. 17.44. Этот параметр является многомерным, т. к. он суммирует все комбинации агрегирующих функций и столбцов в запросе.

- Параметр **WITH ROLLUP** аналогичен параметру **WITH CUBE**, но разрешает только один тип суммирования для каждой комбинации столбца и агрегирующей функции.
- Параметр **ALL** позволяет включить в суммирование все повторяющиеся записи.

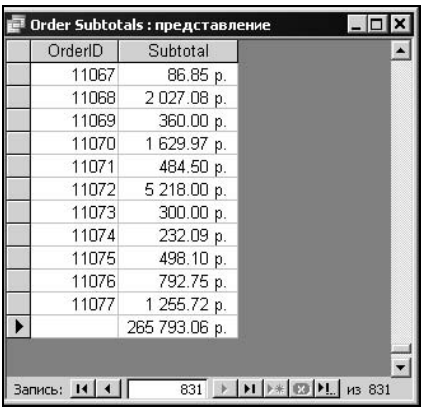


Рис. 17.44. Результат выполнения многомерного суммирования

Замечание

Английские слова заглавными буквами в окне параметров представляют собой ключевые слова инструкций SQL.

Вкладка **Столбцы** (Columns) диалогового окна **Свойства** (Properties) позволяет просмотреть свойства всех столбцов, которые включены в представление. Выберите в свойстве **Имя столбца** (Column Name) имя нужного столбца, и тогда будут отображены его свойства. *(О свойствах столбцов см. табл. 17.1.)*

Вкладка **Поиск** (LookUp) содержит набор свойств, относящихся к полям подстановок. Если выбранное в свойстве **Имя столбца** (Column Name) поле является полем подстановки, то отображаются свойства, которые обычно отображаются на вкладке **Подстановка** (Lookup) в Конструкторе таблиц базы данных Access. Они определяют, откуда брать данные для этого поля и как их отображать. *(Подробную информацию о столбцах постановок см. в разд. "Использование Мастера подстановок" гл. 2.)*

На вкладке **Данные** (Data) отображаются свойства, определяющие фильтр, сортировку и подчиненную таблицу для данного представления. Подробнее об этих свойствах можно узнать, нажав кнопку **Справка** (Help) на этой вкладке.

Создание и применение хранимых процедур

Хранимые процедуры[1] — основное средство программирования серверной логики. Они представляют собой откомпилированный модуль, написанный на языке

[1] В локализованном варианте приложения Access 2003 применяется термин "Сохраненные процедуры". — *Ред.*

Transact-SQL. В коде хранимой процедуры можно использовать не только операции выборки и модификации данных, но и логику ветвления, переменные, вызовы других процедур и некоторые другие средства, характерные для языков программирования высокого уровня. Код процедуры синтаксически анализируется при компиляции, а оптимизированный план выполнения создается при первом вызове процедуры. В отличие от запросов, хранимые процедуры имеют возможность возвращать несколько наборов записей, а также значения.

Для вызова процедуры клиентская программа или другая процедура должны указать имя выполняемой процедуры и передать ей набор входных параметров.

Хранимые процедуры используются в проектах Access везде, где могут использоваться предложения SQL. Они могут служить:

- в качестве источников записей в формах, отчетах, активных страницах;
- в качестве источников строк для полей со списком.

Однако нужно помнить, что хранимые процедуры возвращают необновляемый набор записей.

Хранимая процедура может содержать одну инструкцию SQL, и в этом случае для ее создания можно использовать графический режим Конструктора запросов. Если же она состоит из нескольких инструкций SQL, тогда она создается и изменяется в редакторе SQL.

В первом случае, чтобы создать процедуру, проще всего щелкнуть по ярлыку **Создание сохраненной процедуры в режиме конструктора** (Create stored procedure in designer). При этом открывается стандартное окно Конструктора запросов, в котором можно создать инструкцию SQL, составляющую тело процедуры. Так же как и представление, хранимая процедура в данном случае может иметь ряд дополнительных свойств. Чтобы отобразить или изменить эти свойства, необходимо открыть окно свойств. На вкладке **Сохраненная процедура** (Stored Procedure) содержится перечень свойств, которые аналогичны свойствам представления. А на вкладке **Параметры сохраненной процедуры** (Stored Procedure Properties) можно посмотреть и изменить параметры процедуры.

Для примера воспользуемся хранимой процедурой `Exployee Sales by Country`.

1. Откройте эту процедуру в режиме Конструктора.
2. Нажмите кнопку **Свойства** (Properties) на панели инструментов, чтобы отобразить диалоговое окно **Свойства** (Properties).
3. Раскройте вкладку **Параметры сохраненной процедуры** (Stored Procedure Parameters).

На вкладке отображается таблица со списком параметров процедуры. Каждый параметр обязательно должен иметь имя и тип данных. Кроме того, для него может быть указано значение по умолчанию. Обратите внимание, что имя каждого параметра начинается со значка @. Но имя параметра не может быть изменено на этой вкладке, так же как нельзя добавить новый параметр. Это делается только в бланке запроса в окне Конструктора или прямо в инструкции SQL (рис. 17.45).

Для того чтобы создать хранимую процедуру, состоящую из нескольких инструкций SQL, нужно открыть список запросов в окне проекта и нажать кнопку **Создать** (New). В диалоговом окне **Новый запрос** (New Query) следует выбрать элемент **Ввод**

сохраненной процедуры (Create Text Stored Procedure) (рис. 17.46). Тогда откроется окно редактора SQL с шаблоном новой процедуры, которое представлено на рис. 17.47.

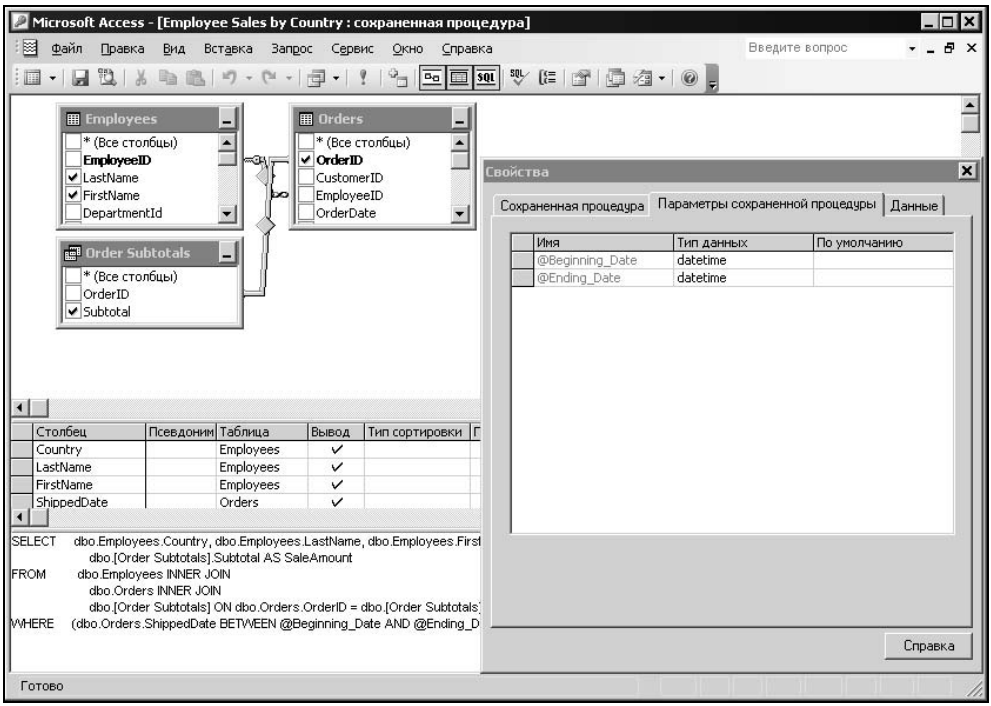


Рис. 17.45. Создание хранимой процедуры в Конструкторе запросов

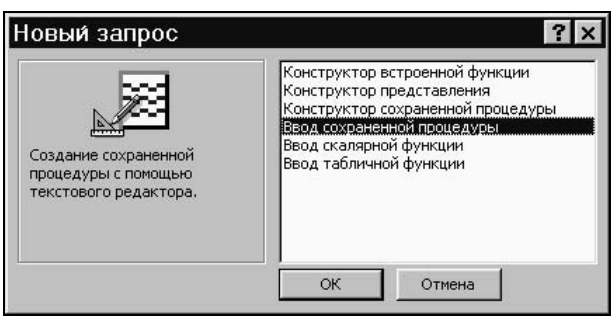


Рис. 17.46. Диалоговое окно **Новый запрос**

Нужно изменить имя процедуры в первой строке шаблона и ввести текст процедуры в соответствии с правилами языка Transact-SQL.

Если в теле процедуры нужно ввести инструкцию SQL `SELECT`, то можно воспользоваться для этого Конструктором запросов. Установите курсор на строке редактора,

в которой должна быть введена инструкция SELECT, и нажмите кнопку **Вставить SQL** (Insert SQL) на панели инструментов (рис. 17.48). Появится окно Конструктора запросов. После того как запрос будет создан, нужно закрыть это окно, подтвердив сохранение изменений инструкции SQL.

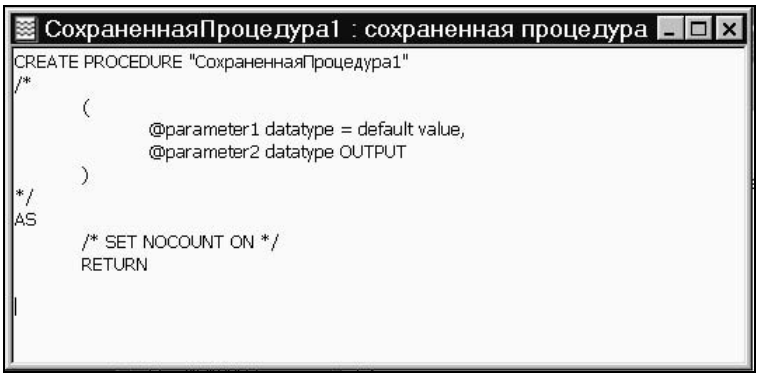


Рис. 17.47. Окно редактора хранимой процедуры

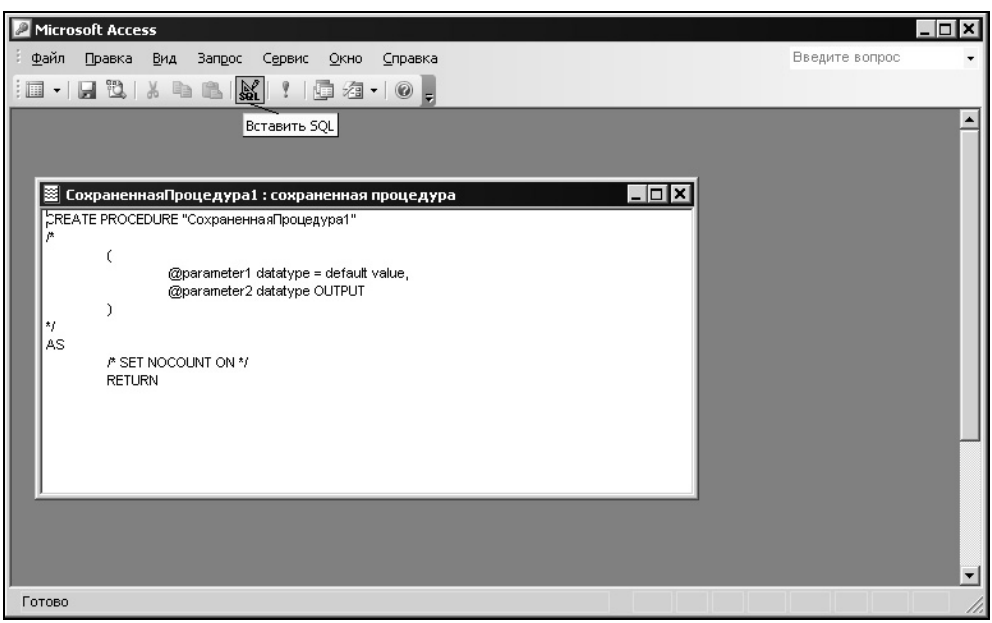


Рис. 17.48. Вызов Конструктора запросов из редактора SQL

Пример использования хранимой процедуры в качестве источника записей вы увидите, если откроете отчет Employee Sales by Country. В качестве источника записей в этом отчете используется одноименная хранимая процедура. В свойстве **Входные**

параметры (Input Parameters) этого отчета описаны два параметра: `@Beginning_date` и `@Ending_date` (рис. 17.49).

При запуске отчета эти параметры будут запрашиваться так же, как и при использовании запроса с параметрами в базе данных Access.

(О языке Transact-SQL, используемом в хранимых процедурах, см. документацию по Microsoft SQL Server.)

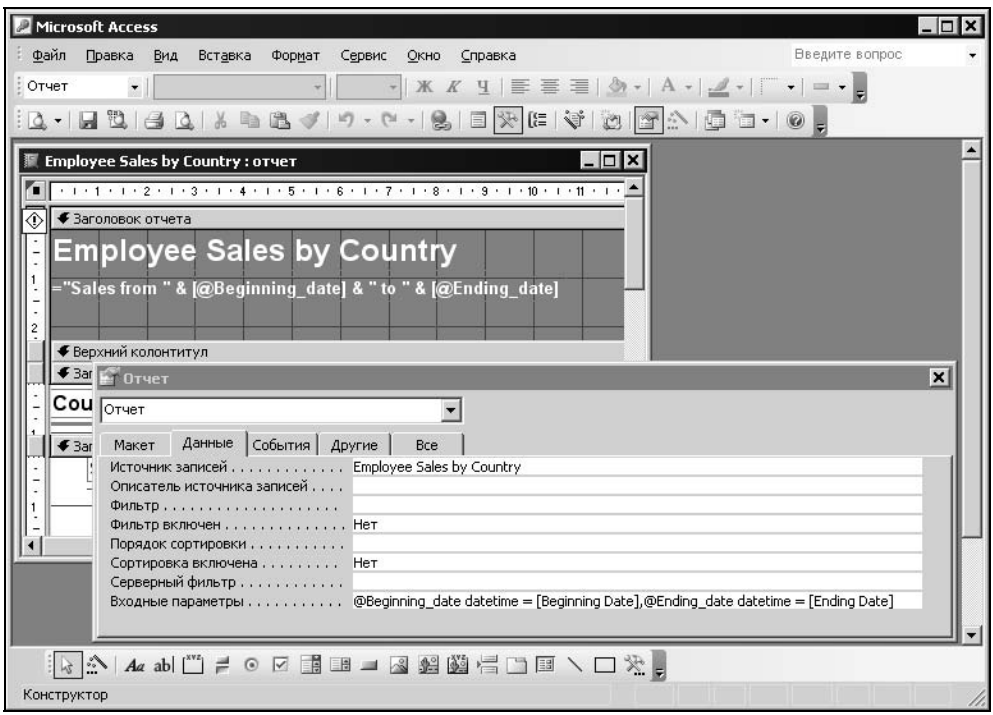


Рис. 17.49. Отчет, основанный на хранимой процедуре с параметрами

Создание и использование определяемой пользователем функции

Определяемая пользователем функция — это новый объект Microsoft SQL Server 2000, аналогичный представлению и хранимой процедуре. Отличие такой функции от упомянутых объектов состоит в том, что она всегда, как и любая другая функция, возвращает некоторое значение. Причем в зависимости от того, какое значение возвращает функция, она относится к одному из трех типов:

- встроенная;
- табличная;
- скалярная.

Встроенная определяемая пользователем функция представляет собой инструкцию `SELECT`, которая возвращает обновляемый набор записей.

Табличная определяемая пользователем функция может содержать несколько инструкций SQL и возвращает необновляемый набор записей.

Скалярная определяемая пользователем функция возвращает скалярное значение одного из следующих типов данных: `int`, `decimal`, `varchar`, `sql_variant` или `table`.

Встроенные и табличные функции обычно употребляются в предложении `FROM` инструкции SQL, скалярная же функция может быть использована в любом месте запроса, где разрешено применять имя поля таблицы. Кроме этого, определенная пользователем функция может быть использована в качестве источника данных для формы, отчета и поля со списком в форме, однако она не может быть использована в качестве источника записей для страницы доступа к данным.

Правила создания определенной пользователем функции те же самые, что и для хранимой процедуры. То есть можно использовать для ее создания Конструктор запросов, если она состоит из одной инструкции `SELECT`, или редактор SQL, если тело функции содержит несколько инструкций SQL. В первом случае вы можете использовать ярлык **Создание функции в режиме конструктора** (Create Function in designer) в списке запросов окна проекта. Во втором случае необходимо нажать кнопку **Создать** (New) и выбрать соответствующую строку в списке операций диалогового окна **Новый запрос** (New Query) (см. рис. 17.46) — **Ввод скалярной функции** (Create Text Scalar Function) или **Ввод табличной функции** (Create Table-Valuable Function). После этого открывается окно редактора SQL с соответствующим шаблоном функции.

(Подробную информацию об определенных пользователем функциях см. в документации по Microsoft SQL Server.)

Использование индексированных представлений

Представления SQL Server могут быть проиндексированы, т. е. существует возможность сохранить результаты представления и создать для них индекс. Правда это можно сделать, только если проект Access подключен к Microsoft SQL Server 2000 Enterprise Edition или SQL Server 2000 Developer Edition. Если вы используете Microsoft SQL Server 2000 Desktop Edition, то это сделать нельзя.

Представления индексируются в том случае, если требуется ускорить доступ к данным, возвращаемым этим представлением. Чтобы создать индекс для представления, необходимо:

1. Открыть представление в режиме Конструктора.
2. Щелкнуть правой кнопкой мыши по свободному полю в области таблиц и выбрать из контекстного меню команду **Управление индексами** (Manage Indexes). Появится диалоговое окно **Индексы** (Indexes) (рис. 17.50).
3. Нажать кнопку **Создать** (New) и заполнить поля для описания индекса. Это делается так же, как при создании индекса для таблицы, т. е. необходимо ввести имя индекса, указать столбцы запроса, по которым должен быть построен индекс, порядок сортировки и т. д.
4. Нажать кнопку **OK**.

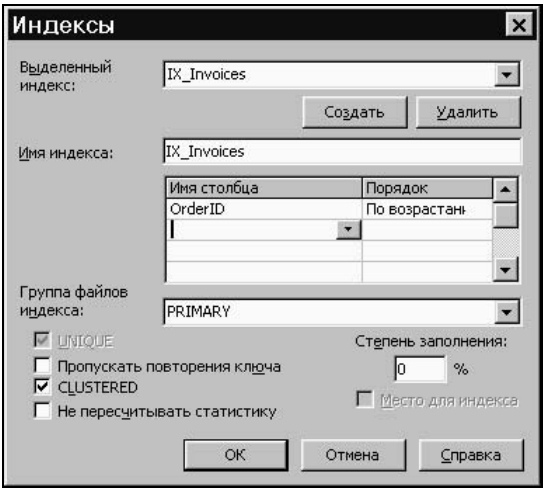


Рис. 17.50. Диалоговое окно **Индексы**

Замечание

Индекс можно добавить к представлению только в том случае, если вы являетесь владельцем всех его базовых таблиц.

Если теперь открыть окно свойств представления, то вы увидите, что установлен флажок **Привязать к схеме** (Bind to Schema). Это означает, что индексированное представление автоматически получает так называемую *схемную привязку*. После того

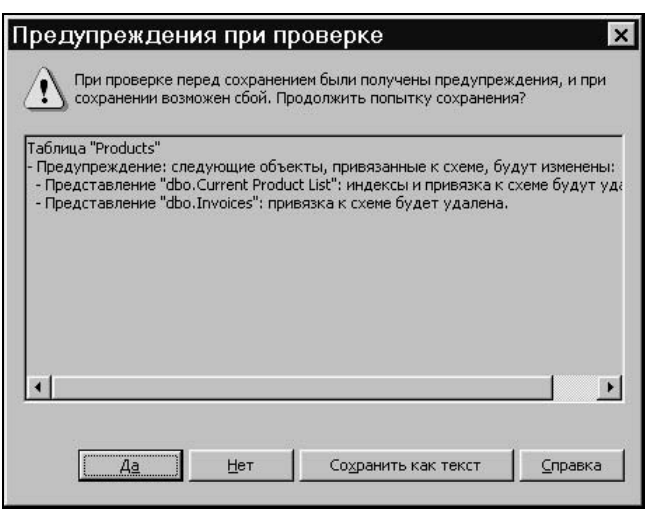


Рис. 17.51. Предупреждающее сообщение о попытке изменения базовой таблицы представлений со схемной привязкой

как такое представление сохранено, SQL Server будет контролировать, чтобы базовые таблицы не были изменены таким образом, что индексированное представление станет некорректным. Например, нельзя изменить наименование базовой таблицы или наименование столбца базовой таблицы, который включен в результат представления. При попытке внести такие изменения в таблицу или при попытке удалить базовую таблицу будет выдано предупреждающее сообщение, например такое, как на рис. 17.51.

В случае подтверждения схемная привязка и индексы в соответствующих представлениях будут удалены.

Замечание

Выполнить схемную привязку можно для любого представления, установив флажок **Привязать к схеме** (Bind to Schema).

Сортировка и фильтрация данных в формах и отчетах

В нашем достаточно беглом обзоре проектов Access нельзя не остановиться еще на двух вопросах:

- Как отфильтровать данные на сервере, для того чтобы передавать минимум данных клиенту?
- Как отсортировать записи в формах и отчетах?

Фильтровать данные лучше с помощью предложений SQL и хранимых процедур, на которых базируются формы и отчеты. Если нужно, используйте в них параметры. Еще один способ — определить так называемый *серверный фильтр*. Для этого необходимо задать значение свойства **Серверный фильтр** (Server Filter) формы или отчета. Свойство **Серверный фильтр** представляет собой строковое выражение, определяющее условие выборки. Фильтр сохраняется в форме или отчете и при загрузке формы в ней будут отображаться только отфильтрованные записи.

Если вы хотите изменять фильтр каждый раз при загрузке формы, необходимо присвоить значение **Да** (Yes) свойству **Серверный фильтр по форме** (ServerFilterByForm). В этом случае при загрузке появится диалоговое окно, в котором можно задать условия отбора записей. После этого необходимо нажать кнопку **Применить серверный фильтр** (Apply Server Filter) на панели инструментов. При этом введенные условия сохраняются в свойстве **Серверный фильтр** (Server Filter) формы и показывается форма с отфильтрованными записями.

Замечание

Нельзя применять серверный фильтр в формах или отчетах, которые созданы на основе хранимой процедуры.

Уже после того как записи отфильтрованы на сервере, можно их еще раз отфильтровать локально, используя методы **Фильтр по выделенному** (Filter by Selection), **Изменить фильтр** (Filter by Form) или **Исключить выделенное** (Filter For Input).

Для того чтобы сортировка записей выполнялась на сервере, нужно использовать в качестве источника записей формы (отчета) представление, хранимую процедуру,

определенную пользователем функцию или предложение SQL. В противном случае сортировка будет выполняться локально.

Повышение производительности приложения "клиент-сервер"

Существует набор общих правил, относящихся к приложениям "клиент-сервер", которые нужно соблюдать, чтобы уменьшить нагрузку на сеть и увеличить производительность приложения. Эти правила таковы: необходимо перенести по возможности обработку данных на сервер, минимизировать количество обращений к серверу (roundtrip) для доступа к данным и ограничить объем данных, загружаемых с сервера.

Минимизация количества обращений к серверу для доступа к данным

Связь проекта Access 2003 с базой данных на SQL Server осуществляется посредством OLE DB. При доступе к данным из формы, таблицы или страницы доступа к данным OLE DB предоставляет так называемый *обновляемый статический набор записей* (updateable snapshot recordset), полученный с помощью одного обращения к базе данных на SQL Server. Этот набор записей кэшируется на клиентском приложении. В Access данные загружаются в асинхронном режиме, что позволяет пользователю выполнять другие действия с базой данных в процессе загрузки данных с сервера. Просмотр, фильтрация, сортировка, поиск и обновление данных в форме, таблице или странице доступа к данным производятся с кэшированными на клиентской стороне данными. Таким образом, использование обновляемого статического набора записей минимизирует количество обращений к серверу для доступа к данным.

Осуществление обработки данных на сервере

Чтобы осуществить обработку данных на сервере, используйте хранимые процедуры, определяемые пользователем функции и инструкции SQL, а также выполняйте сортировку данных на сервере перед загрузкой их с сервера. Сортировка данных на сервере выполняется с помощью хранимой процедуры, определяемой пользователем функции или предложения SQL, заданного в свойстве **Источник записей** (RecordSource) формы или отчета (это предложение выполняется на сервере).

Ограничение объема загружаемых с сервера данных

Необходимо ограничить объем загружаемых данных из базы данных на SQL Server, насколько это возможно. Используйте для этого представления, хранимые процедуры, определяемые пользователем функции, фильтры, выполняющиеся на сервере, и предложение SQL `WHERE`, позволяющее выделить только нужные записи для отображения в клиентском приложении. Следует избегать таких сценариев работы, где пользователю разрешается просматривать весь объем данных.

Рекомендуется ограничить максимальное число возвращаемых запросом или отображаемых в форме записей с помощью специальной кнопки **Максимальное число записей** (Maximum record limit) или свойства **Максимальное число записей** (MaxRecords), которое можно установить в программе VBA или в окне свойств запроса или формы.

Используйте фильтры, выполняющиеся на сервере, прежде чем загружать данные с сервера, и фильтры, выполняющиеся на клиенте, чтобы ограничить объем отображаемых данных. Для формы или отчета можно установить фильтр с помощью свойства **Серверный фильтр** (ServerFilter). Увеличить быстроту загрузки списков значений можно, установив подходящее значение свойства **Применение автофильтра** (FilterLookup) (элемента управления для отображения ограниченного набора значений, загружаемых с сервера или с другой рабочей станции).

Повышение производительности при работе с отчетами

Существует ряд правил, соблюдение которых позволяет повысить скорость печати отчетов. Наряду с перечисленными далее советами полезно проверить конкретные отчеты с помощью анализатора быстродействия.

- Для печати большого набора данных создавайте составные отчеты, содержащие подчиненные данные, используя в качестве источника данных представления, хранимые процедуры, предложения SQL или фильтры.
- Используйте режим **Образец** (Layout Preview) вместо режима **Предварительный просмотр** (Print Preview) при проектировании отчета. Режим **Образец** (Layout Preview) позволяет быстро просмотреть структуру отчета, т. к. в отчет загружается лишь образец данных, а это требует гораздо меньшего времени, чем загрузка всего объема данных. Используйте режим **Предварительный просмотр** (Print Preview) для проверки данных после того, как проектирование структуры отчета завершено.
- Чтобы распространить среди сотрудников такую периодическую информацию, как расписания, недельные отчеты, планы продаж и пр., создавайте отчеты в формате снимка отчета (snapshot) или XML-файла и распространяйте их пользователям, которые могут посмотреть их с помощью программы просмотра снимков Snapshot Viewer или других приложений Microsoft Office 2003. Если необходимо предоставить пользователям оперативную информацию, используйте для этого страницы доступа к данным.

Повышение производительности при работе с формами

Представим некоторые рекомендации по повышению производительности форм в клиентском приложении.

- При проектировании формы определите, какие данные она должна предоставлять и какой функциональностью должна обладать. Отложите процесс загрузки данных с сервера до того момента, когда это потребуется пользователю. Создайте

форму, в которую данные не будут загружаться в процессе открытия, поместите в форму специальную кнопку, позволяющую загрузить данные.

- Используйте в формах минимальное количество полей, требующих длительного времени для загрузки данных, таких как текст и графические элементы. Поместите поле с текстом или рисунок в подчиненную форму, использующую тот же источник данных, что и главная форма.
- Помещайте в главную форму только основные поля, все остальные данные поместите в подчиненную форму, открываемую с помощью специальной кнопки с названием **Дополнительная информация** в главной форме. В качестве источника данных для подчиненной формы используйте запрос, параметризованный ключевыми полями источника данных главной формы.
- Для ввода данных создайте отдельные формы специального типа, установив для них значение **Да** (Yes) свойства **Ввод данных** (Data Entry). В формы такого типа данные из источника данных (находящегося на сервере) не загружаются, в них отображаются только добавляемые записи, что значительно ускоряет работу.
- Минимизируйте число строк, показываемых в полях со списком, если данные для этих списков берутся с сервера. Если данные редко модифицируются, то лучше храните их в локальных таблицах и создайте процедуру для синхронизации серверных и локальных таблиц. В локальных таблицах, хранящих такие данные, обязательно постройте индексы, чтобы ускорить создание списков.
- Если форма или отчет основаны на параметризованной хранимой процедуре, устанавливайте значение параметров с помощью свойства **Входные параметры** (Input Parameters).
- Чтобы удалить или обновить данные, используйте инструкции SQL `UPDATE` и `DELETE` с параметром `WHERE`. Этот способ более эффективен по сравнению с открытием набора записей и выполнением над ним необходимых операций.

Повышение производительности внешней базы данных на SQL Server

Полные сведения о повышении производительности баз данных на SQL Server можно найти в документации к SQL Server. Здесь приведены лишь некоторые рекомендации по повышению производительности.

- Проектируйте таблицы так, чтобы в них не содержалось избыточной информации. Используйте средство нормализации схемы данных (команда **Сервис | Анализ | Таблицы** (Tools | Analyze | Tables)) для разделения таблиц на несколько связанных. Хорошо спроектированная схема данных способствует более быстрому доступу к данным.
- Установите контроль ссылочной целостности данных. Это позволит сохранить корректность данных в связанных таблицах при добавлении, удалении и обновлении записей.
- Выбирайте подходящие типы данных для полей таблиц. Это позволит сэкономить место на диске и оптимизировать операции объединения данных из разных таблиц.

- Индексируйте поля, по которым производятся сортировка, объединение и выборка. Это приведет к существенному повышению производительности при выполнении запросов, созданных на основе таблиц, содержащих индексированные поля по обе стороны связи, и запросов, производящих выборку по индексированным полям. Однако индексирование полей замедляет процесс добавления, обновления и удаления записей.
- Добавьте в таблицы поля типа "штамп времени" (timestamp) для повышения скорости удаления и обновления записей, особенно если в таблице содержится большое количество полей. Поле "штамп времени" (timestamp) используется SQL Server перед обновлением записи для определения, была ли запись изменена. Это позволяет получить выигрыш производительности, особенно в многопользовательской среде.
- Периодически проводите сжатие базы данных на SQL Server, чтобы освободить место на диске, а также реорганизовать индексы и данные для более быстрого доступа к данным.

ГЛАВА 18



Репликация баз данных

Что такое репликация? Это средство организации работы одного или нескольких пользователей с одними данными на разных компьютерах. А более конкретно, *репликация* — это процесс создания копий файлов, между которыми может осуществляться обмен обновляемыми данными или объектами. Такие копии называются *репликами*, а такой обмен — *синхронизацией*.

Когда нужна репликация? Microsoft приводит два примера потребности в такой организации работы.

- Вы — агент по продажам и совершаете поездки для посещения клиентов. Перед поездкой вам нужно просмотреть последние заказы и маркетинговые тенденции клиентов, которых вы собираетесь посетить. Эта информация хранится в базах данных о клиентах и заказах вашей компании. Вы загружаете данные о клиентах на свой переносной компьютер. Во время посещения клиента вы обновляете загруженную информацию о клиенте, например, номера телефонов, адреса e-mail и прочую персональную информацию о клиенте на вашем переносном компьютере, а также вносите туда информацию о новых заказах. Однако вам необходимо закончить работу с заказами как можно быстрее. В конце дня вы подключаетесь через Интернет к сети вашей компании через защищенное удаленное соединение и обновляете базы данных о клиентах и заказах. Конфликты данных автоматически разрешаются благодаря имеющимся бизнес-процедурам. После этого вы печатаете и отправляете по факсу отчет о принятии заказа клиенту или отправляете снимок отчета по e-mail, который клиент может просмотреть с помощью программы Просмотр снимков (Snapshot Viewer).
- Вы — разработчик программного обеспечения и хотите закончить работу с приложением вашей компании, отслеживающим статистику об обнаружении и исправлении ошибок в проекте, дома, где нет необходимого или защищенного доступа к корпоративной сети. Вы забираете данные только о ваших ошибках, помещая их на свой переносной компьютер и отключаете его от корпоративной сети. Затем приходите домой и работаете с имеющимися данными, изменяя поля статуса ошибки и прочие поля с информацией об ошибке. На следующий день вы подключаете свой переносной компьютер к корпоративной сети и легко синхронизируете свои изменения с приложением вашей компании, отслеживающим статистику ошибок в проекте.

Существует несколько средств репликации баз данных, проектов Access и других файлов.

- *Портфельная репликация* — средство операционной системы Microsoft Windows. Оно позволяет осуществлять репликацию файлов многих типов, в том числе баз данных Access (файлов MDB), исключая проекты Access (файлы ADP). Средства портфельной репликации Windows позволяют работать с базой данных на переносном компьютере во время поездки и синхронизировать ее позже с базой данных на компьютере, находящемся в офисе.
- *Репликация баз данных и проектов средствами Access* — встроенные средства Microsoft Access. Они предназначены для репликации баз данных и проектов Access. Эти средства позволяют организовать совместную работу нескольких пользователей с базой данных или проектом путем создания реплик — копий базы данных или проекта, — изменения в которых могут быть синхронизированы.
- *Репликация файлов на сервере Web* — средство сервера Web фирмы Microsoft. Оно позволяет работать с файлами, сохраненными на узле Web, в автономном режиме — без подключения к серверу. Средство репликации сервера Web предоставляет возможность удаленным пользователям работать с базой данных, хранящейся на сервере Web даже в автономном режиме после отключения от сети Интернет.
- *Программная репликация с помощью интерфейсов DAO и JRO.* Создание и управление репликами баз данных Access может осуществляться программно — в процедурах на VBA. Разработчики приложений Access могут обеспечить автоматическую синхронизацию реплик и прочие действия, связанные с репликацией, используя специальные свойства и методы объектов из библиотек VBA: *Объекты репликации и Jet* (JRO) для репликации баз данных Access 2000 и выше и *Объекты доступа к данным* (DAO) для репликации баз данных более ранних версий Access 95 и 97.

Эта глава посвящена репликации баз данных и проектов Access с помощью различных инструментов. Репликация включает следующие действия:

- выбор средства репликации;
- создание реплик;
- синхронизация реплик;
- управление репликами.

Способ выполнения перечисленных действий зависит от выбранного средства репликации. Мы рассмотрим основные вопросы, касающиеся использования перечисленных средств репликации, но не будем касаться темы программирования. Дополнительную информацию о репликации можно найти в справочной системе Access.

Портфельная репликация

Портфельная репликация является удобным способом поддержания файла базы данных в обновленном состоянии при работе с ним на разных компьютерах, которые лишь иногда соединяются посредством локальной сети или прямого кабельного соединения, или обмен данными между ними выполняется с помощью переносного

диска. В этом случае файл базы данных можно поместить в Портфель, чтобы перенести его на переносной компьютер. При этом создается новая реплика, которая может быть синхронизирована с базой данных на стационарном компьютере.

Предположим, вы ведете на своем персональном компьютере список контактов с помощью базы данных `Контакты`. Вас отправляют в командировку на несколько дней, но вам необходимо постоянно использовать контактную информацию, сохраненную на вашем компьютере. В этом случае следует использовать средства портфельной репликации.

Сначала создайте базу данных `Контакты`. Для этого можно использовать Мастер создания базы данных Access:

1. Выберите команду **Файл | Создать** (File | New). Появится панель **Создание файла** (New File). В разделе **Шаблоны** (Templates) щелкните по гиперссылке **На моем компьютере** (On my computer).
2. В появившемся диалоговом окне **Шаблоны** (Templates) раскройте вкладку **Базы данных** (Databases) (рис. 18.1).
3. Дважды щелкните по ярлыку **Контакты** (Contact Management).
4. Откроется окно, в котором следует выбрать папку для сохранения новой базы данных и имя файла. Выберите папку Мои документы (My Documents) и введите имя файла `Контакты.mdb`. Нажмите кнопку **Создать** (New).
5. Запустится Мастер создания базы данных по шаблону **Контакты** (Contact Management). Следуйте предлагаемым инструкциям.

После того как база данных будет создана и заполнена необходимой информацией, закройте ее. Чтобы иметь возможность работать с базой данных на портативном компьютере, создайте реплику с помощью средства портфельной репликации и поместите ее на портативный компьютер.

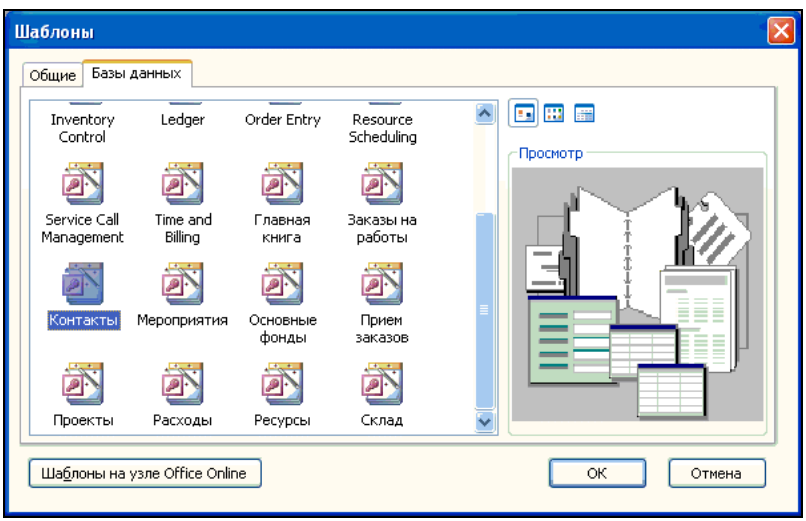


Рис. 18.1. Вкладка **Базы данных** диалогового окна **Шаблоны**

Чтобы создать реплику базы данных `Контакты`:

1. Закройте Access и запустите Проводник. Раскройте папку Мои документы, в которой хранится база данных Контакты.mdb. Нормализуйте окно Проводника, нажав кнопку **Восстановить** (Restore) в правом верхнем углу окна.
2. Перетащите с помощью мыши базу данных `Контакты` на значок **Портфель** (Briefcase) на рабочем столе Windows.

Замечание

В Windows 2000 и XP значок **Портфель** (Briefcase) изначально не отображается на рабочем столе. Откройте Проводник (Explorer), раскройте любую папку (или рабочий стол Windows — папку **Рабочий стол** (Desktop)), в которой хотите создать новый Портфель, и выберите команду **Файл | Создать | Портфель** (File | New | Briefcase) или щелкните правой кнопкой мыши по свободному месту в этой папке, выберите команду **Создать | Портфель** (New | Briefcase) в контекстном меню. Выполните шаг 2.

3. Начнется процесс обновления Портфеля.
4. Через некоторое время будет отображено сообщение, требующее подтвердить выполнение репликации (рис. 18.2). Нажмите кнопку **Да** (Yes).

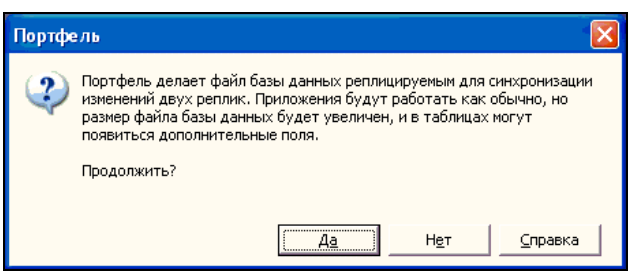


Рис. 18.2. Требование подтвердить репликацию базы данных с помощью Портфеля

5. Будет отображено сообщение с предложением создать резервную копию (рис. 18.3). Нажмите кнопку **Да** (Yes).

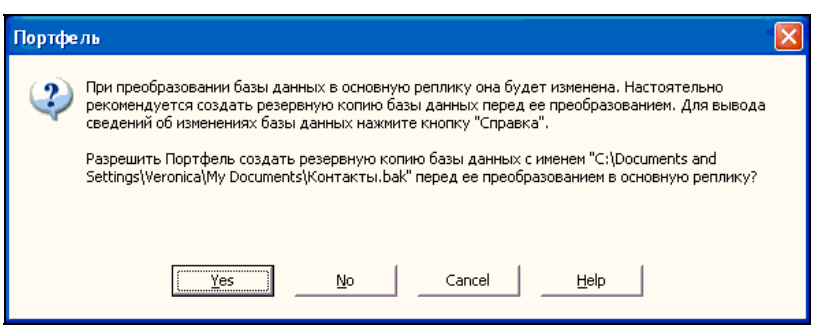


Рис. 18.3. Создание резервной копии базы данных

6. Будет отображено сообщение с вопросом о том, какую из двух реплик (реплику, полученную из исходной базы данных, или реплику, созданную в Портфеле) следует назначить основной репликой (рис. 18.4). Не забудьте, что основная реплика допускает правку структуры объектов и схемы данных, а обычная — нет. Чтобы сделать основной реплику, полученную из исходной базы данных, оставьте выбранным переключатель **Исходная копия** (Source Copy). Иначе выберите переключатель **Копия в портфель** (Copy to Briefcase). Нажмите кнопку **OK**.

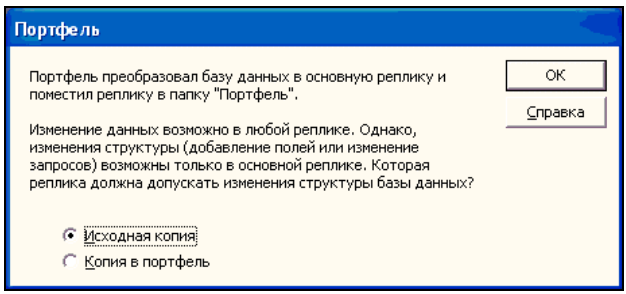


Рис. 18.4. Выбор основной реплики

Откройте Портфель. Для этого дважды щелкните по значку **Портфель** (Briefcase) на рабочем столе Windows. Появится окно со списком реплик в Портфеле (рис. 18.5).

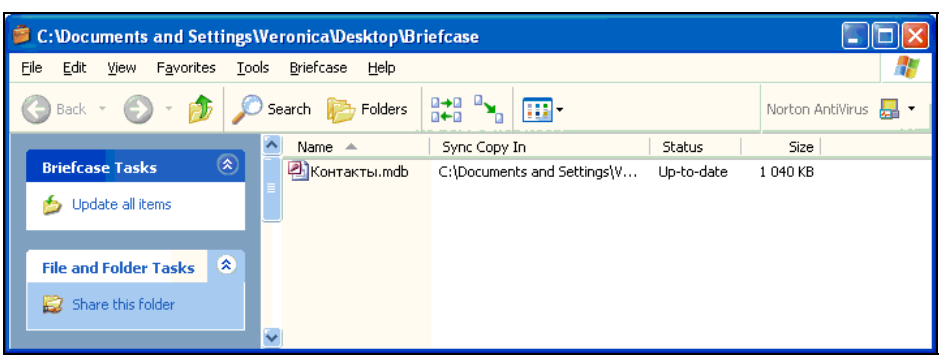


Рис. 18.5. Окно со списком реплик в Портфеле

Замечание

Если открытие или создание Портфеля выполняется впервые, на одном из перечисленных ранее шагов отобразится диалоговое окно **Welcome to the Windows Briefcase** (Создание портфеля Windows) (рис. 18.6). Ознакомившись с пояснительным текстом в этом окне, закройте его, нажав кнопку **Finish** (Готово).

Щелкните правой кнопкой мыши по элементу списка **Контакты.mdb** и выберите в контекстном меню команду **Свойства** (Properties), чтобы открыть окно свойств. Раскройте вкладку **Update Status** (Обновление) для вывода информации о состоянии синхронизации. Оригинал и реплика базы данных `Контакты` являются последними версиями, т. к. реплики не изменялись (рис. 18.7). Нажмите кнопку **OK**.

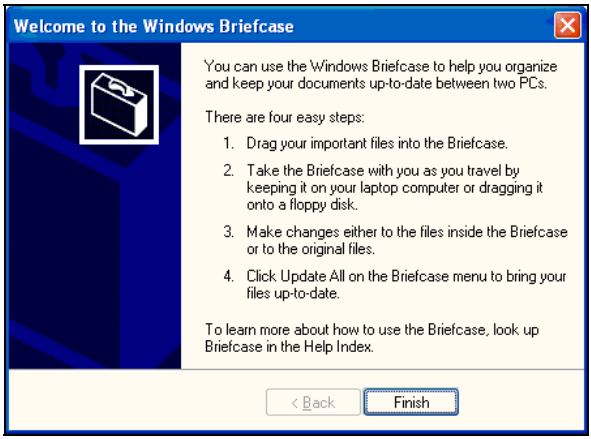


Рис. 18.6. Диалоговое окно **Welcome to the Windows Briefcase**

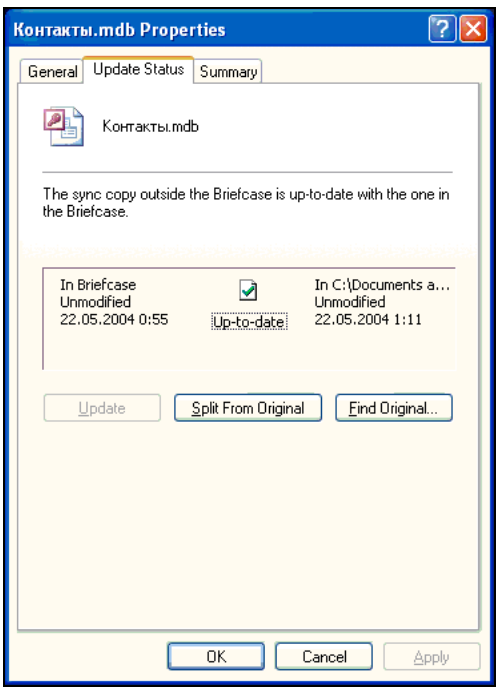


Рис. 18.7. Состояние синхронизации реплики в Портфеле

Для просмотра содержимого реплики дважды щелкните по элементу **Контакты.mdb** списка реплик в Портфеле. Запустится Access и откроется реплика базы `Контакты`. В заголовке окна базы данных появится надпись **реплика** (Replica), указывающая на то, что производится работа с репликой исходной базы данных.

Выберите команду **Сервис | Параметры** (Tools | Options) и раскройте вкладку **Вид** (View). Установите флажки **скрытые объекты** (Hidden objects) и **системные объекты** (System objects) и нажмите кнопку **OK**. Обратите внимание на то, что для обеспечения репликации в базу данных `Контакты` было добавлено множество скрытых таблиц, например, `MSysReplicas` и `MSysTombstone` (рис. 18.8).

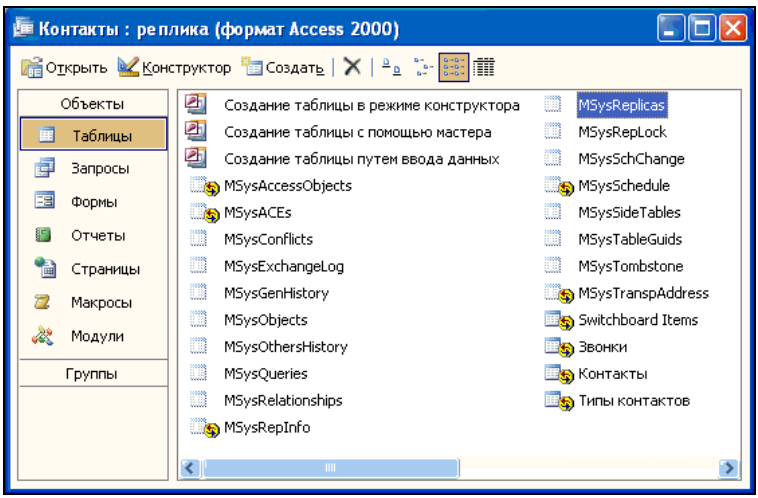


Рис. 18.8. Реплика базы данных `Контакты` с системными таблицами

Снова выберите команду **Сервис | Параметры** (Tools | Options) и сбросьте установленные на прошлом шаге флажки. Нажмите кнопку **OK** и закройте реплику базы данных `Контакты`.

Чтобы поместить реплику на портативный компьютер или отдать ее другому пользователю:

1. Создайте копию Портфеля, содержащего реплику, путем копирования и вставки Портфеля на рабочий стол, а затем переименования копии. Копии Портфеля используются для обновления реплик другими пользователями.
2. Откройте Портфель и выберите команду **Портфель | Обновить все** (Briefcase | Update All). Основная реплика обновляется данными, содержащимися в репликах других Портфелей.
3. Поместите копию портфеля на рабочий стол портативного компьютера (или передайте ее на дискете другому пользователю).

Теперь можно работать с базой данных на портативном компьютере.

Синхронизировать реплику в Портфеле на портативном компьютере с репликой в Портфеле на основном компьютере и основной репликой можно несколькими способами.

- Используя *сетевое подключение*. Связав портативный компьютер с Портфелем основного компьютера, можно обновлять реплику Портфеля непосредственно с помощью команды контекстного меню реплики **Обновить** (Update). Если реплика открыта в режиме разделения (по умолчанию), то допускается одновременное

обновление реплики Портфеля несколькими пользователями. Однако пользователям в сети проще соединиться напрямую и обновить основной (исходный) файл MDB.

- Используя *прямое кабельное соединение*. Для обеспечения доступа к Портфелю основного компьютера пользователями портативных компьютеров можно соединить последовательные (COM) порты основного и переносного компьютера с помощью кабеля последовательной связи (нуль-модемный кабель). Если компьютеры оборудованы параллельными портами, использующими протокол ECP (Расширенный протокола связи), то можно соединить параллельные порты основного и переносного компьютера с помощью кабеля ECP. При параллельном соединении обновление проходит намного быстрее, чем при последовательном соединении. Для соединения различных типов параллельных портов имеется возможность использовать кабель UCM (Универсальный модуль кабеля).
- Используя *соединение с помощью модемов*. Если установить на основном компьютере приложение Телефон (Phone Dialer), то можно связать портативный компьютер с Портфелем основного компьютера и обновить реплику в Портфеле.
- С помощью *дискет*, *переносного диска* или *флэш-памяти*. Реплику в Портфеле основного компьютера можно перенести на портативный компьютер или компьютер другого пользователя с помощью дискеты или переносного диска. Для этого переместите Портфель на дискету (роль "дискеты" может выполнять и переносной диск и флэш-память), а затем перепишите его с дискеты на рабочий стол переносного компьютера или компьютера другого пользователя. После внесения изменений на портативном компьютере или на компьютере другого пользователя проведите обратный процесс: переместите Портфель переносного компьютера на дискету, а с дискеты на рабочий стол основного компьютера. Затем выполните синхронизацию реплики в Портфеле с основной репликой, как рассказано далее. Многие пользователи применяет именно этот способ.

Замечание

Перемещение Портфеля на дискету (а также переносной диск или флэш-память) является важным этапом в осознании концепции Портфеля Windows. Реплики всех файлов хранятся в Портфеле на стационарном компьютере. Когда вы уходите, вы берете с собой Портфель. Один Портфель не может быть в двух местах одновременно. Более того, Портфель не является объектом рабочего стола Windows. По возвращении необходимо переместить Портфель с дискеты (или диска) на Рабочий стол. Если Портфель разделяется в сети, то аналогией из реальной жизни является папка для входящих бумаг.

Если изменить данные или параметры либо в реплике Портфеля, либо в основной реплике, то статус файла изменится на **Modified** (Изменен). Если со времени последней синхронизации одна из реплик изменялась, то можно обновить либо реплику в Портфеле, либо основную реплику. Чтобы синхронизировать реплики:

1. Откройте Портфель. Щелкните правой кнопкой по базе данных `Контакты` и выберите в контекстном меню команду **Свойства** (Properties). В окне свойств реплики раскройте вкладку **Update Status** (Обновление). Статусом реплики в Портфеле является значение **Modified** (Изменен) (рис. 18.9), хотя данные не изменялись. Дело в том, что на состояние реплики повлияло изменение параметров базы данных в Access.
2. Нажмите кнопку **Update** (Обновить) для обновления основной реплики.

Аналогичным образом производится обновление реплики в Портфеле, когда изменения внесены в основную реплику.

Замечание

Щелкнув правой кнопкой мыши по полю окна свойств реплики, в котором отображается статус обновления реплики, можно отобразить список для выбора направления записи изменений в двух синхронизируемых репликах.

Синхронизацию реплик в Портфеле можно проводить также с помощью команд **Портфель | Обновить выделенные объекты** (Briefcase | Update Selection) и **Портфель | Обновить все** (Briefcase | Update All). Последняя команда очень удобна, когда требуется обновить реплики в нескольких копиях Портфеля на рабочем столе, которые принадлежат разным пользователям, обновляющим реплики в Портфеле с помощью дискет. В этом случае, выполнив команду обновления всего один раз, можно внести все изменения, произведенные каждым пользователем.

Замечание

Можно создать копию Портфеля так же, как обычной папки с файлами Windows. Чтобы создать совершенно новый Портфель, не копируя имеющийся, в Проводнике Windows выберите команду **Файл | Создать | Портфель** (File | New | Briefcase).

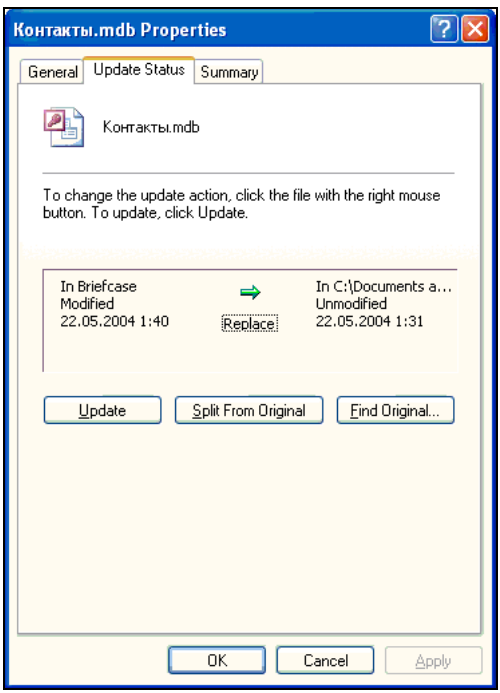


Рис. 18.9. Статус **Modified** для реплики в Портфеле

Репликация баз данных Access

В начале главы были приведены основные понятия репликации файлов. Представим эти понятия в контексте репликации баз данных средствами Access.

- *Репликой* называется каждая копия реплицированной базы данных. Каждая реплика базы данных содержит общий (для всех реплик базы данных) набор таблиц, запросов, форм, отчетов, страниц доступа к данным, макросов и модулей. Изменения данных таблицы, сделанные в одной из реплик, передаются в другие реплики. Каждая реплика может также содержать *локальные объекты*, которые существуют только в этой реплике.

- Отдельная реплика является компонентом *набора реплик* и допускает синхронизацию с другими репликами в наборе. В наборе реплик выделяется главная реплика, она называется *основной*. Основная реплика является главным компонентом набора реплик и используется для создания первой реплики в наборе. В основную реплику добавляются системные таблицы, системные поля и свойства репликации. Изменение структуры базы данных допускается только в основной реплике. Другие реплики в том же наборе также могут быть назначены основными репликами, но в каждый момент времени в наборе реплик должна быть только одна основная реплика. Основная реплика и все остальные реплики имеют общую структуру базы данных и уникальный идентификатор набора реплик.

- *Синхронизацией* называется процесс обновления двух компонентов набора реплик, при котором происходит взаимный обмен обновленными записями и объектами. После синхронизации двух компонентов набора реплик изменения из каждой реплики оказываются примененными в другой реплике.

- *Частичной репликой* называется база данных, содержащая подмножество записей полной реплики. С помощью частичной реплики пользователь имеет возможность устанавливать фильтры и задавать отношения, определяющие, какое подмножество записей полной реплики должно входить в базу данных.

- *Область видимости.* Реплики разделяются на три типа по области видимости: *глобальные*, *локальные* и *анонимные*. Тип реплики по области видимости определяет ряд важных характеристик реплики, например, типы реплик, которые могут создаваться по данной реплике, может ли данная реплика стать основной репликой в наборе реплик, а также как в ней обрабатываются конфликты при синхронизации. Возможные типы реплик описаны в табл. 18.1.

- *Приоритет реплики* — это специальная характеристика реплики базы данных, определяющая относительный приоритет реплики в наборе реплик. Приоритет выражается целым положительным числом. Приоритет реплик учитывается при разрешении конфликтов, возникших в процессе синхронизации реплик. Подробнее о приоритетах рассказано в справочной системе Access.

Выполнение действий, связанных с созданием, синхронизацией и управлением репликами в Access, осуществляется с помощью команд меню **Сервис | Репликация** (Tools | Replication).

Таблица 18.1. *Типы реплик по области видимости*

Тип реплики	Описание
Глобальная (global) реплика	Реплика, по которой можно создавать реплики всех остальных типов. При репликации базы данных первая создаваемая реплика (*основная* реплика) является глобальной репликой. Изменения, внесенные в глобальную реплику, отслеживаются полностью; возможен обмен этими изменениями с любой другой глобальной репликой в наборе. Глобальная реплика может также обмениваться изменениями с любой локальной или анонимной репликой, для которой она становится разветвителем. *Разветвителем* называется глобальная реплика, с которой все реплики в наборе синхронизируют свои изменения
Локальная (local) реплика	Реплика, в которой выводятся только данные из этой реплики и не отображаются данные из других реплик в наборе, в том числе из основной реплики. Локальная реплика синхронизирует свои данные с разветвителем, который является глобальной репликой. Синхронизация локальных реплик с другими репликами в наборе не разрешена. Если изменения в локальной реплике конфликтуют с глобальной репликой-разветвителем, эти изменения автоматически теряются в любом процессе устранения конфликтов
Анонимная (anonymous) реплика	Особый тип реплик в базах данных Microsoft Access, для которых не ведется отслеживание отдельных пользователей. Анонимные реплики особенно удобны при работе в Интернете, когда ожидается загрузка реплик многими пользователями

Создание реплик базы данных

Основная и обычная реплики базы данных создаются в Access одинаковым способом: с помощью команды меню **Сервис | Репликация | Создать реплику** (Tools | Replication | Create Replica). Частичная реплика создается с помощью специального Мастера *(см. разд. "Создание частичной реплики")*. Новую реплику можно создать на основе любой существующей реплики. В любое время основной репликой можно назначить другую реплику в наборе. Основная реплика отличается от обычной. В основной реплике разрешено изменение структуры объектов базы данных и схемы данных, а в обычной и частичной репликах — только просмотр структуры объектов и схемы данных. Но вводить, удалять и изменять данные разрешено во всех репликах набора.

Чтобы создать реплику базы данных:

1. Откройте базу данных, предназначенную для репликации, например, `Борей` (Northwind). Если работа идет в многопользовательской среде, то следует убедиться в том, что все остальные пользователи закрыли эту базу данных. Если база данных защищена паролем, то его необходимо удалить. Чтобы создать новую реплику в наборе реплик, откройте созданную ранее основную или обычную реплику.
2. Выберите команду **Сервис | Репликация | Создать реплику** (Tools | Replication | Create Replica).

3. Появится диалоговое окно с требованием подтверждения закрытия базы данных. Нажмите кнопку **Да** (Yes).

Замечание

Если команда **Сервис | Репликация | Создать реплику** используется для создания реплики еще нереплицированной базы данных, то в результате ее выполнения база данных будет преобразована в основную реплику, а затем для нее будет создана первая обычная реплика. В этом случае после требования о закрытии базы данных будет отображено сообщение с вопросом о создании резервной копии базы данных. Рекомендуем создать резервную копию базы данных перед ее репликацией. Для этого нажмите кнопку **Да** (Yes).

4. Появится диалоговое окно **Размещение новой реплики** (Location of New Replica) (рис. 18.10). Укажите папку для размещения новой реплики. Если нужно, измените имя файла реплики, заданное по умолчанию.

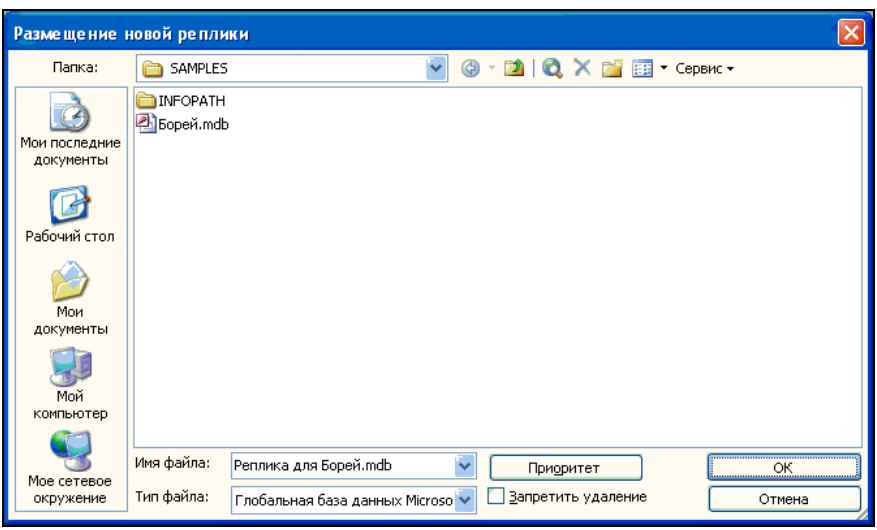


Рис. 18.10. Диалоговое окно **Размещение новой реплики**

5. Чтобы задать приоритет реплики, нажмите кнопку **Приоритет** (Priority). Появится диалоговое окно, представленное на рис. 18.11. Введите приоритет реплики и нажмите кнопку **OK**.

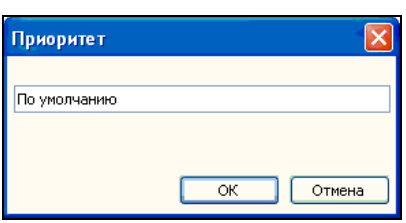


Рис. 18.11. Диалоговое окно **Приоритет**

6. Если требуется запретить пользователям удалять записи из таблиц в реплике, установите флажок **Запретить удаление** (Prevent delets).
7. Чтобы задать область видимости реплики, выберите необходимый элемент в раскрывающемся списке **Тип файла** (Save as type).
8. Нажмите кнопку **OK**. Начнется процесс создания реплики. После его завершения, возможно, появится пояснительное сообщение с текстом "Изменения структуры базы данных допускаются только в основной реплике; изменения данных могут выполняться как в основной, так и в любой другой реплике набора". Нажмите кнопку **OK**.
9. Если база данных не была реплицированной, в результате работы Access она преобразуется в основную реплику и отобразится на экране (рис. 18.12). Кроме этого будет создана первая реплика с указанным именем (шаг 4). Если текущая база данных уже была репликой, в результате работы Access на ее основе будет создана новая реплика с указанным именем. В реплику добавляются системные таблицы и поля, специально предназначенные для поддержки реплики.

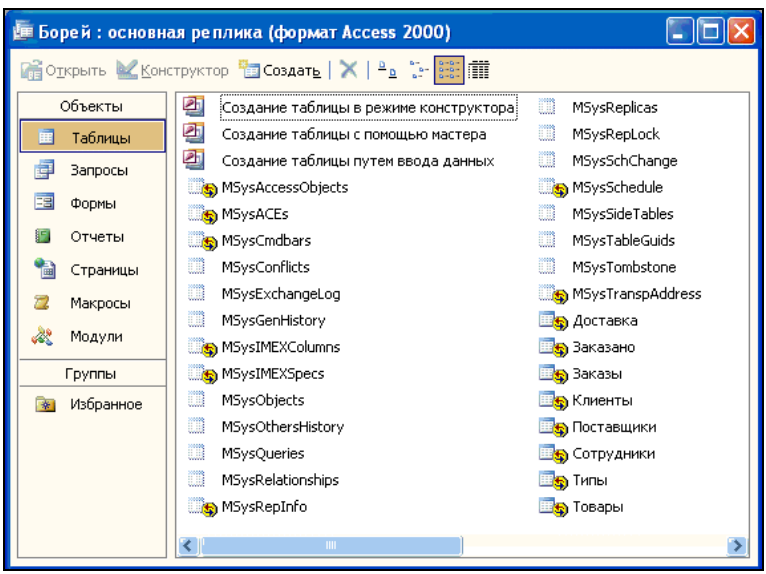


Рис. 18.12. Основная реплика базы данных Борей с системными таблицами

Создание частичной реплики

Частичная реплика (partial replica) — это реплика, содержащая лишь подмножество записей полной реплики (обычной реплики). Для создания частичной реплики требуется задать фильтр для выбора требуемых записей, которые будут синхронизироваться с соответствующими записями в полной реплике.

Частичные реплики позволяют ограничиться лишь необходимым для работы объемом данных. Синхронизация частичной реплики требует передачи меньшего коли-

чества данных, чем синхронизация полной реплики, что, конечно, уменьшает нагрузку на сеть.

Чтобы создать частичную реплику базы данных:

1. Откройте реплику базы данных, на основе которой нужно создать частичную реплику. Например, откройте реплику базы данных `Борей`, создание которой было описано в предыдущем разделе.
2. Выберите команду **Сервис | Репликация | Мастер частичной репликации** (Tools | Replication | Partial Replica Wizard). Появится первое диалоговое окно Мастера (рис. 18.13).

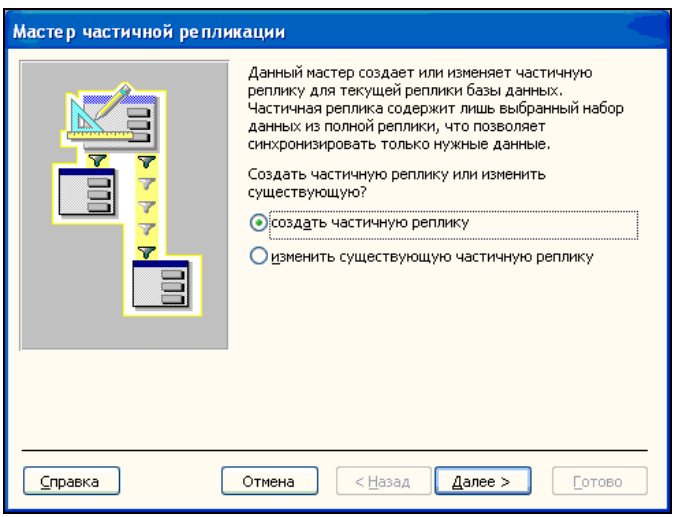


Рис. 18.13. Первое диалоговое окно Мастера частичной репликации

3. Выберите переключатель **создать частичную реплику** (Create a new partial replica) и нажмите кнопку **Далее** (Next).
4. Появится диалоговое окно, представленное на рис. 18.14. Введите путь и имя файла MDB для сохранения частичной реплики. Здесь же можно задать дополнительные свойства создаваемой реплики. Нажмите кнопку **Далее**.
5. Появится диалоговое окно, позволяющее задать фильтр для отбора данных, которые следует сохранить в частичной реплике (рис. 18.15). Частичную реплику можно создать на основе только одной таблицы из базы данных, но из этой таблицы можно выбрать несколько полей.
6. Выделите в списке **Таблица для отбора** (Table to filter) имя таблицы, из которой будут выбираться данные. Например, выделите таблицу `Заказы` (Orders). В списке **Поля таблицы** (Fileds in the table) отобразятся все поля выбранной таблицы.
7. Теперь нужно составить выражение фильтра. Оно подобно выражению SQL `WHERE`, только не может содержать вызовов статистических и пользовательских функций и подчиненных запросов. Например, составим фильтр для выбора всех

заказов, размещенных с 1-го апреля 1998 года, стоимость доставки которых превышает 100 рублей.

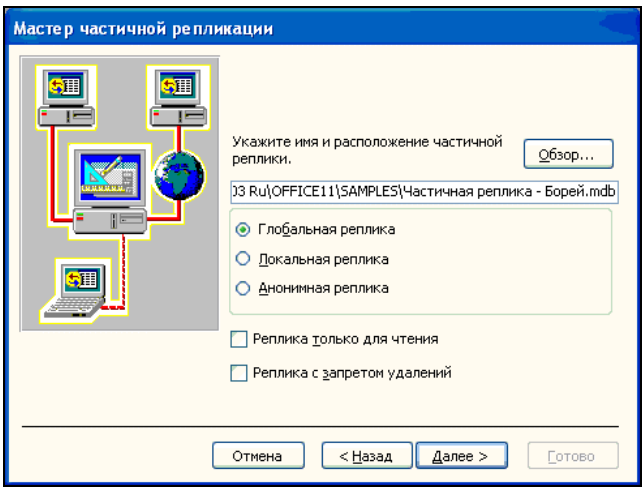


Рис. 18.14. Второе диалоговое окно Мастера частичной репликации

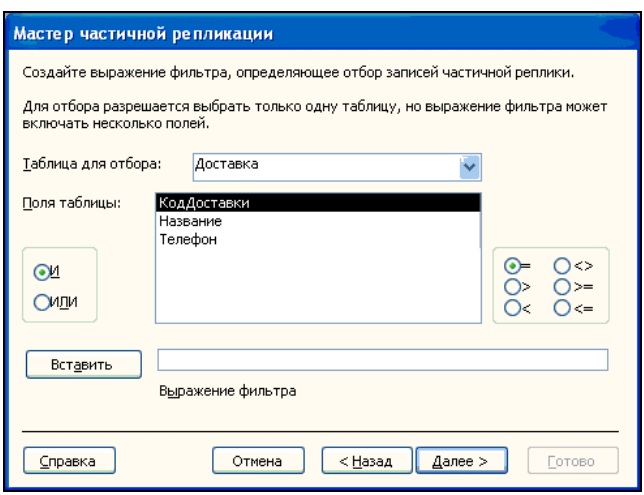


Рис. 18.15. Третье диалоговое окно Мастера частичной репликации

8. Выделите поле `ДатаРазмещения` (OrderDate) в списке **Поля таблицы**, выберите переключатель >= и нажмите кнопку **Вставить** (Paste). В поле **Выражение фильтра** (Filter expression) появится строка:

```
[ДатаРазмещения] >= [Выражение]
```

9. Вместо слова `[Выражение]` введите дату `#4/1/1998#`.

10. Выделите в списке **Поля таблицы** поле `СтоимостьДоставки` (ShippedPrice), выберите переключатель **>** в группе переключателей отношений и переключатель **И** (AND) в группе переключателей логических связок. Нажмите кнопку **Вставить**. В поле **Выражение фильтра** появится строка:

```
[ДатаРазмещения] >= #4/1/1998# AND [СтоимостьДоставки] > [Выражение].
```

11. Вместо слова `[Выражение]` введите число `100`. Получится требуемое выражение фильтра: `[ДатаРазмещения] >= #4/1/1998# AND [СтоимостьДоставки] > 100` (рис. 18.16). Нажмите кнопку **Далее**.

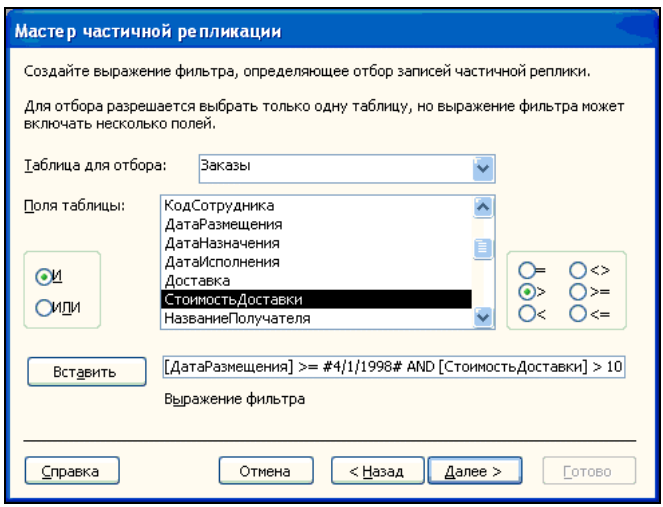


Рис. 18.16. Фильтр для отбора записей в таблице `Заказы`

12. Появится следующее диалоговое окно Мастера (рис. 18.17). В списке **Список таблиц** (List of Tables) отображаются все таблицы текущей реплики за исключением той, на основе которой создан фильтр для отбора записей. Таблицы, связанные с таблицей, к которой применяется фильтр, выделены полужирным шрифтом. Установите флажок около имени таблицы, чтобы включить в частичную реплику все записи в этой таблице. Сбросьте флажок около имени таблицы, если нужно включить в частичную реплику только записи, связанные с отобранными фильтром записями. Нажмите кнопку **Далее**.

13. Появится последнее диалоговое окно Мастера (рис. 18.18), предлагающее создать отчет с описанием содержания частичной реплики. Оставьте выбранным переключатель **Да** (Yes) и нажмите кнопку **Готово** (Finish).

Начнется процесс создания частичной реплики. Это может занять несколько минут. По окончании работы Мастера будет отображен отчет о создании частичной реплики. Сохраните его в формате снимка отчета, чтобы иметь информацию о данных, хранящихся в частичной реплике. Тогда этот отчет можно будет просмотреть в любое время с помощью программы Просмотр снимков (Snapshot Viewer) (рис. 18.19). *(См. разд. "Снимки отчетов" гл. 6.)*

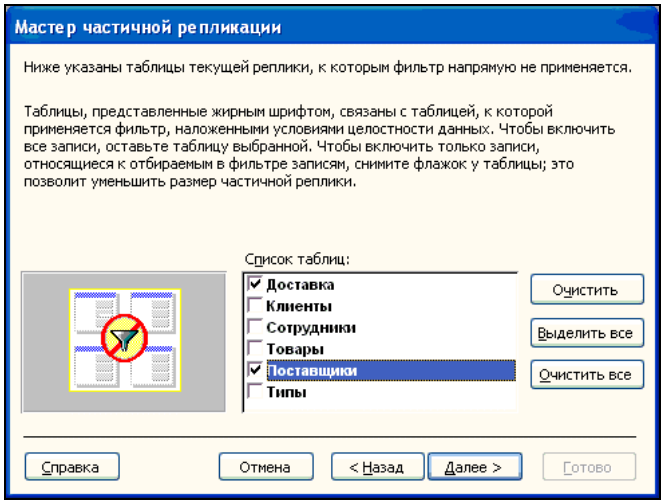


Рис. 18.17. Четвертое диалоговое окно Мастера частичной репликации

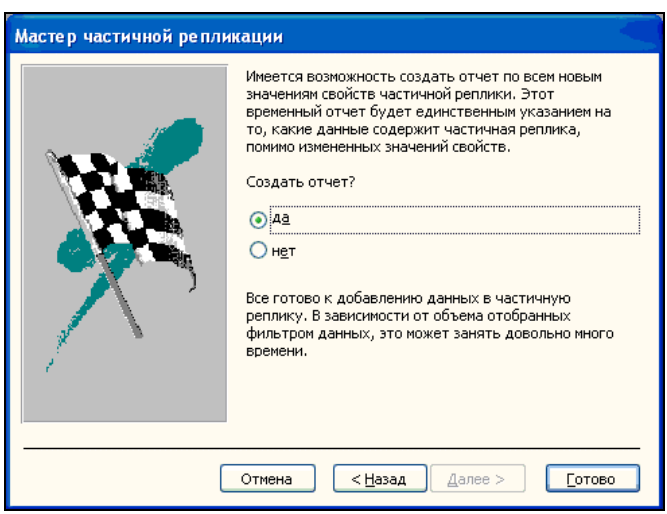


Рис. 18.18. Последнее диалоговое окно Мастера частичной репликации

Синхронизация реплик

В Microsoft Access, Диспетчере репликации Microsoft, Объектах репликации и Jet (JRO) можно выбрать один из трех различных методов синхронизации данных.

- *Прямая синхронизация* является прекрасным решением для случаев, когда реплики непосредственно подключены к локальной сети и находятся в общих сетевых папках. Прямая синхронизация не очень хорошо подходит для удаленной

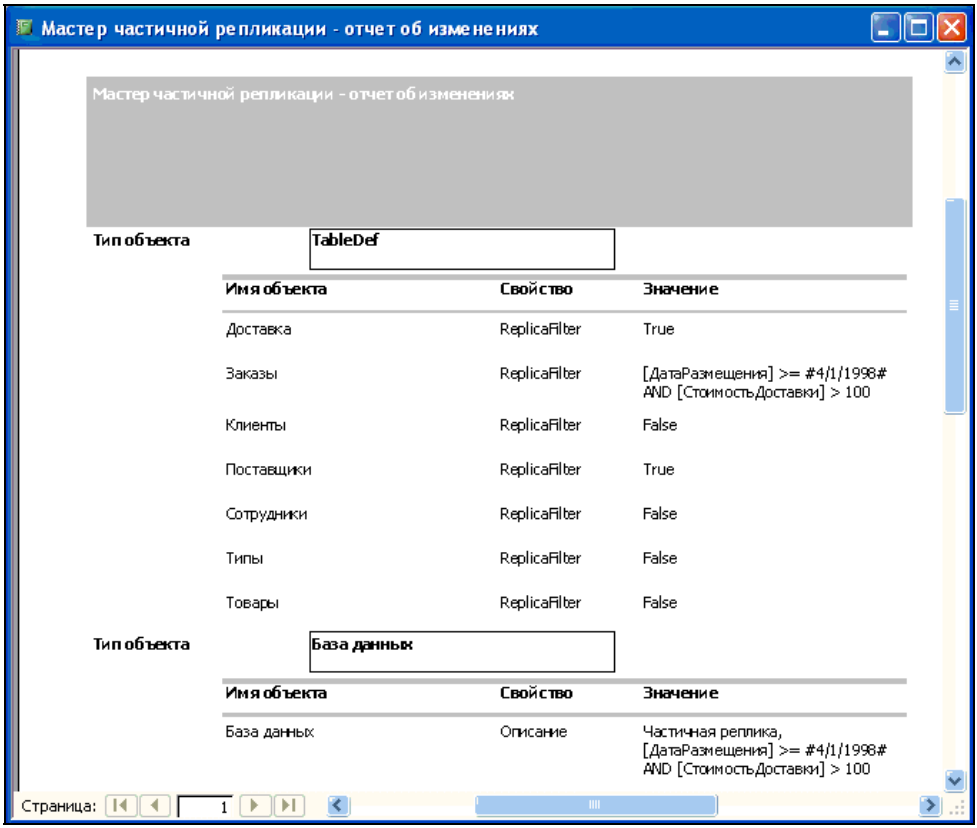


Рис. 18.19. Отчет о создании частичной реплики

синхронизации с помощью сервера удаленного доступа (RAS) или соединения удаленного доступа. В этом случае следует использовать косвенную синхронизацию или синхронизацию по Интернету.

Если при попытке выполнить прямую синхронизацию с репликой, присутствующей в списке известных реплик, эта реплика не будет найдена, то она будет исключена из набора реплик.

- *Косвенная синхронизация* полезна при работе в автономной среде, например, на переносном компьютере.

  Косвенную синхронизацию можно настроить только с помощью Диспетчера репликации (Replication Manager), входящего в комплект средств разработчика Microsoft Office 2002 Developer. После того как косвенная синхронизация будет настроена в диспетчере репликации, ее можно выполнять с помощью Microsoft Access, диспетчера репликации или JRO.

- *Синхронизация по Интернету* представляет собой удобный способ синхронизации реплик в тех автономных средах, которые имеют доступ к сети Интернет.

Для настройки синхронизации по Интернету нужно использовать Диспетчер репликации (Replication Manager), который входит в комплект средств разработчика Microsoft Office 2002 Developer.

Чтобы синхронизировать реплику с другой репликой в наборе реплик базы данных:

1. Откройте реплику, которую нужно синхронизировать. Это может быть основная, обычная или частичная реплика.
2. Выберите команду **Сервис | Репликация | Синхронизация** (Tools | Replication | Synchronize Now). Появится диалоговое окно, представленное на рис. 18.20.

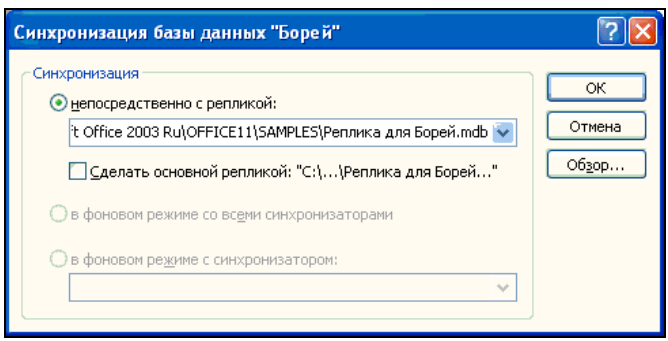


Рис. 18.20. Диалоговое окно **Синхронизация базы данных "Борей"**

3. Введите путь и имя файла элемента набора реплик, который требуется синхронизировать с текущей репликой. Это может быть основная, обычная или частичная реплика.
4. Если текущую реплику нужно сделать основной, установите флажок **Сделать основной репликой *"<имя файла>"*** (Make '<имя файла>' the Design Master). (Этот режим недоступен в том случае, когда одна или обе синхронизируемые реплики являются частичными репликами и когда обе синхронизируемые реплики являются обычными репликами — не основными.)
5. Нажмите кнопку **OK**.
6. Отобразится сообщение с требованием подтвердить закрытие базы данных для выполнения синхронизации. Чтобы выполнить синхронизацию, нажмите кнопку **Да** (Yes).

В результате синхронизации могут возникать конфликты. Устранение конфликтов выполняется с помощью специальной программы Просмотр конфликтов (Conflict Viewer). Чтобы ее запустить, откройте базу данных, которую требуется проверить на наличие конфликтов, и нажмите кнопку **Да** (Yes) в окне с сообщением о наличии конфликтов и с предложением их устранить или выберите команду **Сервис | Репликация | Устранить конфликты** (Tools | Replication | Resolve Conflicts). Если конфликтов в реплике нет, программа Просмотр конфликтов не запускается, сообщение о наличии конфликтов при открытии реплики тоже не отображается. Сведения о причинах возникновения конфликтов и об использовании программы Просмотр конфликтов находятся в справочной системе Access.

Управление репликами

Управление реплицированными базами данных включает несколько задач.

- *Восстановление основной реплики.* Если основная реплика в наборе реплик была повреждена, потеряна, переименована, удалена или перемещена, можно восстановить основную реплику. Для этого:
  1. Убедитесь, что в наборе отсутствует основная реплика.
  2. Выберите реплику в наборе, которую нужно сделать основной (для этого нельзя использовать частичную реплику).
  3. Синхронизируйте ее со всеми остальными репликами в наборе.
  4. Чтобы сделать текущую реплику основной, выберите команду **Сервис | Репликация | Восстановить основную реплику** (Tools | Replication | Recover Design Master).

Внимание!

В наборе реплик не должно быть больше одной основной реплики. В противном случае могут происходить сбои синхронизации в наборе реплик.

- *Назначение реплики основной репликой.* Чтобы одну из реплик в наборе назначить новой основной репликой:
  1. Убедитесь, что данная реплика и текущая основная реплика никем не используются.
  2. Откройте реплику, которую нужно сделать основной.
  3. Выберите команду **Сервис | Репликация | Синхронизация** (Tools | Replication | Synchronize Now).
  4. Выберите в списке текущую основную реплику для синхронизации с открытой репликой.
  5. Установите флажок **Сделать основной репликой *"<имя файла>"*** (Make '<имя файла>' the Design Master) и нажмите кнопку **OK**.
- *Преобразование реплицированной базы данных в нереплицированную.* После преобразования базы данных в основную реплику ее обратное преобразование в нереплицированную базу данных будет невозможно. Однако если дальнейшее использование репликации не планируется и требуется уменьшить объем реплицированной базы данных, следует создать новую нереплицированную базу данных, содержащую все объекты и данные реплицированной базы данных, но не содержащую дополнительные системные поля, таблицы и свойства, добавляемые при репликации. Для этого требуется выполнить импорт из реплики в новую базу данных необходимых объектов всех типов, кроме таблиц. Для переноса таблиц в новую базу данных создайте в ней запросы, выбирающие все записи из исходных таблиц и создающие на их основе новые таблицы. Воссоздайте индексы и связи между таблицами. Подробные инструкции о выполнении этой процедуры можно найти в справочной системе Access.
- *Защита реплицированной базы данных.* Для защиты реплики можно использовать систему защиты на уровне пользователей. Назначенные права доступа к объектам

реплики для пользователей рабочей группы не влияют на синхронизацию этой реплики с другими репликами в наборе. Реплику нельзя защитить паролем. Нельзя также защищенную паролем базу данных преобразовать в реплицированную базу данных. Для защиты файла реплики можно установить разрешения на общую папку, в которой хранится реплика (не назначайте доступ только для чтения, иначе не будет выполняться синхронизация с остальными элементами набора реплик), или поместить ее в папку, недоступную в сети, и использовать косвенную синхронизацию через Диспетчер репликации Microsoft.

- *Удаление реплики из набора реплик.* Удалите файл реплики, выделив его в Проводнике Windows и нажав клавишу <Del>. Выполните синхронизацию любой реплики из набора с удаленной репликой. Access автоматически определит, что заданная реплика отсутствует и удалит ее из набора реплик. Не удаляйте основную реплику из набора реплик, если только она не повреждена. Если в наборе реплик отсутствует основная реплика, невозможно изменить структуру базы данных.
- *Создание резервной копии реплицированной базы данных Microsoft Access.* При создании реплики предоставляется возможность создать резервную копию базы данных. Резервные копии полезны, однако пользоваться ими следует лишь в исключительных ситуациях. Любая копия с резервной копии базы данных или любая реплика, созданная из преобразованной резервной копии, не может быть синхронизирована с существующими элементами набора реплик.

Репликация проектов Access

Репликация проектов Access отличается от репликации баз данных. В проектах реплицируются только данные. Формы, отчеты и прочие объекты, содержащиеся в проекте, не реплицируются.

В отличие от баз данных для репликации проектов Microsoft Access предоставляет средства создания не реплик, а *публикаций* (publication) и *подписок* (subscription).

- В проекте Microsoft Access *публикация* может содержать одну или несколько статей с таблицами или хранимыми процедурами из одной базы данных пользователя. Создание публикации выполняется с помощью Мастера, который запускается с помощью команды **Сервис | Репликация | Создать публикацию** (Tools | Replication | Create Publication). Каждая пользовательская база данных может иметь несколько публикаций.
- *Статьей* в публикации называют сгруппированные данные, которые реплицируются как целое. Статьей может быть целая таблица, либо некоторые строки и столбцы, определяемые фильтром. Публикация может содержать несколько статей. Подписчики подписываются на публикацию, а не на отдельные статьи в ней.
- *Подпиской* называется база данных, получающая реплицированные таблицы и данные из базы данных издателя в проекте Microsoft Access. Создать подписку можно с помощью Мастера, который запускается с помощью команды **Сервис | Репликация | Запросить подписку** (Tools | Replication | Pull Subscription).

Синхронизация реплицированных проектов Access может быть трех типов: прямая, косвенная и через Интернет. Синхронизация реплицированных проектов осуществ-

ляется аналогично синхронизации реплицированных баз данных с помощью команды **Сервис | Репликация | Сихронизация** (Tools | Replication | Synchronize Now).

Подробная информация о репликации проектов Access содержится в документации Microsoft SQL Server.

Внимание!

Создание публикаций проектов возможно только, если проект подключен к SQL Server. Проект, подключенный к Microsoft SQL Server Desktop Edition (MSDE), можно использовать только в качестве подписчика реплицированной базы данных на Microsoft SQL Server.

Репликация файлов на сервере Web

Есть еще одно средство работы с базой данных в той ситуации, когда требуется синхронизация данных — это работа с данными в Web в автономном режиме (off-line), когда нет постоянного подключения к источнику данных.

Репликация файлов на сервере Web позволяет работать с файлами, используя программу просмотра Microsoft Internet Explorer 5 (и выше) или приложение из пакета Microsoft Office (начиная с версии Office 2000), даже после отключения от сервера. На локальном компьютере сохраняются копии файлов, хранящихся на сервере, отображаемых в Проводнике в папках Web. Локальные копии могут быть просмотрены и изменены даже при работе в автономном режиме. При подключении к серверу Web, копии файлов на локальном компьютере могут быть синхронизированы с исходными файлами, хранящимися на сервере Web.

Внимание!

На сервере Web, который будет использоваться для сохранения синхронизируемых файлов, должны быть установлены серверные расширения FrontPage.

Репликация файлов в Internet Explorer и Windows

Описание процедур репликации и синхронизации файлов на сервере Web, использования Менеджера синхронизации Windows (Synchronization Manager), так называемых *подписок* (subscription), и папки **Избранное** (Favorites) можно найти в справочной системе Windows и Internet Explorer.

Репликация страниц доступа к данным

Когда нет постоянного подключения к базе данных, можно воспользоваться средством работы со страницами доступа к данным проектов Access в автономной режиме (без подключения), чтобы было удобно использовать и изменять данные.

Для такой работы потребуется следующее программное обеспечение:

- Microsoft Access 2002 или выше, установленное на клиентском компьютере, например на переносном компьютере или домашнем компьютере;

- Microsoft SQL Server 2000 Desktop Edition (MSDE2000), установленное и работающее на клиентском компьютере;
- Microsoft SQL Server 2000, установленное и работающее на удаленном сервере;
- Microsoft Internet Explorer 5.01 с пакетом обновления SP2 или более поздней версии, установленное на клиентском компьютере и на удаленном сервере.

Инструкции по использованию синхронизируемых автономных страниц доступа к данным для проектов Access можно найти в справочной системе Access.

ГЛАВА 19



Миграция приложений

В данной главе будут рассматриваться вопросы, касающиеся преобразования приложений Access с целью переноса их в другую среду эксплуатации. Для обозначения таких преобразований мы будем применять термин *миграция*. Под миграцией приложений понимается целевое преобразование приложения с изменением его внутренней структуры и сохранением внешнего интерфейса. Таким образом, к миграции относятся как преобразование приложений, созданных в более ранних версиях Access, в среду Access 2003, так и преобразование приложения, созданного в среде Access 2003, в двухуровневое клиент-серверное приложение, в котором обработка данных выполняется сервером базы данных.

Потребность в таких преобразованиях очевидна. При переходе на новую версию инструментального программного обеспечения возникает вопрос о том, как сохранить то, что уже наработано в предыдущих версиях, и с минимальными затратами обновить свои приложения, получив возможность использовать все преимущества новых технологий. Второй вопрос — каким образом обеспечить дальнейшее развитие приложений как в сторону добавления новых функций, так и их масштабирования, т. е. возможность подключения существенно большего количества пользователей к базе данных. В *гл. 17* мы обсуждали, как создать с использованием Microsoft Access приложение, которое имело бы архитектуру "клиент-сервер". Однако не обязательно разрабатывать такое приложение "с нуля". Access имеет встроенные средства, которые позволяют выполнить преобразование уже существующих приложений, перенеся на Microsoft SQL Server все таблицы, правила проверки вводимых данных и многое другое.

В данной главе мы постараемся ответить на поставленные вопросы и рассказать о следующем:

- когда и как выполнить преобразование настольного приложения Access в приложение с архитектурой "клиент-сервер";
- каким образом можно использовать приложения, разработанные в предыдущих версиях Microsoft Access, в Access 2003;
- как выполнить преобразование приложений, разработанных в предыдущих версиях Microsoft Access, в Access 2003;
- как выполнить обратное преобразование, т. е. приложение, созданное в среде Access 2003, преобразовать в форматы предыдущих версий Access.

Преобразование настольного приложения Access в приложение с архитектурой "клиент-сервер"

Приложение, разработанное в среде Access, является настольным приложением. Оно может быть предназначено для одного пользователя, а может быть и многопользовательским, может быть достаточно сложным, как, например, приложение, которое мы рассматривали в *гл. 16*, в котором взаимодействуют несколько процессов. Однако все эти процессы работают под управлением настольной СУБД Access, которая имеет ограничения как по количеству одновременно работающих пользователей, так и по объему базы данных. С увеличением сложности приложения и накоплением данных в таблицах Access может возникнуть необходимость перенесения этих данных на сервер баз данных, который работает на значительно более мощной программно-аппаратной платформе. В этом случае приложения Access устанавливаются на клиентских машинах и играют роль клиентов, обращающихся к данным, хранящимся в базах данных SQL-сервера.

В клиент-серверных информационных системах на компонент "сервер" возлагается задача надежного хранения данных и обработки запросов клиента, в то время как от "клиентской" части требуется лишь обеспечение удобного интерфейса пользователя. Поэтому компонент "сервер" исполняется на специальной серверной платформе, которая обеспечивает серверное приложение необходимыми ресурсами и мощностью, а компонент "клиент" исполняется на менее мощной аппаратно-программной платформе. В нашем случае мы рассматриваем систему с серверной СУБД Microsoft SQL Server, работающей под управлением Windows NT Server, и клиентскими частями, управляемыми менее мощными СУБД Microsoft Access. *(О технологии "клиент-сервер" и ее применении в Access см. гл. 17.)*

Целесообразность перехода к архитектуре "клиент-сервер"

Выбор архитектуры приложения главным образом зависит от поставленной задачи. Многие задачи успешно реализуются в небольших сетях с файловым сервером. В других случаях нужно применять более мощную клиент-серверную архитектуру. Преобразование настольного приложения в клиент-серверное (upsizing) целесообразно с точки зрения обеспечения четырех важнейших требований:

- надежность;
- производительность;
- масштабируемость;
- безопасность.

Надежность

Наиболее важным требованием, которое обычно предъявляется к масштабным приложениям, является повышение надежности. Отдельные копии приложений, работающие на клиентских компьютерах в файл-серверных сетях, не имеют синхронизи-

рованного журнала транзакций (напомним, что *транзакция* представляет собой одну операцию обмена данными между клиентом и сервером), поэтому сбой на любой клиентской машине или в сети может привести к порче данных. Обычно такая база данных может быть восстановлена, но это может занять достаточно продолжительное время, в течение которого никто из пользователей работать с системой не может. В клиент-серверных системах при централизованном управлении данными сбои в сети или клиентских компьютерах редко влияют на базу данных. Кроме того, большинство серверов имеют возможность быстро и качественно восстановить данные. Для этого сервер использует специальный механизм управления транзакциями. Этот механизм гораздо более эффективен, чем соответствующий механизм в настольных СУБД, т. к. сервер контролирует работу транзакций централизованно. Например, процессор обработки данных Jet поддерживает обработку транзакций только во время текущей сессии (сеанса работы). Он не может разрешить проблемы, возникшие из-за сбоя в предыдущей сессии. Он может оставить "зависшие" блокировки в файле блокировок (файле с расширением ldb) и никто не сможет получить доступ к данным, пока вы не удалите этот файл. Он также не гарантирует защиту данных, если сбой произошел в процессе завершения транзакции. Разумеется, это очень редкое событие, но это может быть основанием для серьезного рассмотрения вопроса о переходе к использованию клиент-серверных приложений.

Производительность

Повышение производительности является другой, иногда не менее важной причиной перехода к клиент-серверной архитектуре. Хотя существует много способов повысить производительность многопользовательских приложений, подключение сервера часто является единственным способом достижения необходимого результата.

Требования к производительности приложения зависит от типа задач, которые оно должно выполнять. В табл. 19.1 указаны некоторые задачи, для которых, вероятно, лучше перейти к клиент-серверной архитектуре.

Таблица 19.1. *Преимущества использования сервера для некоторых задач*

Тип обработки	Преимущества использования сервера
Большой объем транзакций, выполняемых над многими таблицами	Сетевой трафик уменьшается за счет передачи по сети только запросов клиентов и результатов обработки. Вследствие этого уменьшается время ожидания обработки запроса при той же стоимости аппаратуры. Дополнительное преимущество заключается в улучшенном управлении блокировками
Выполнение нетривиальных запросов над совместно используемыми данными	Улучшенное управление приоритетами доступа. Сетевой трафик уменьшается, поскольку по сети передаются только результаты обработки данных
Изменение небольшого числа полей (даже одного) в большом количестве записей	Сетевой трафик значительно уменьшается, поскольку по сети передаются лишь измененные поля
Массивная обработка данных над большими таблицами	Мощность серверной СУБД в сочетании с аппаратными возможностями самого сервера позволяет значительно повысить производительность

Иногда полезно некоторые таблицы перенести на сервер, но такие процедуры, как анализ данных, составление сложных отчетов путем слияния многих таблиц, лучше оставить на клиентских станциях. Можно даже все данные держать на сервере и дать возможность загружать нужные таблицы целиком на рабочие станции для анализа.

Масштабируемость

Масштабируемыми (scalability) называют приложения, допускающие их расширение (распределение) на большее число компьютеров, использующих, возможно, разные аппаратно-программные платформы (для обслуживания большего числа пользователей). И при этом установка приложений на подключаемые к сети компьютеры должна проводиться без какой-либо их переработки. Многопользовательские системы на базе настольных СУБД сильно ограничены как по числу клиентов, так и по размерам баз данных, причем снять эти ограничения невозможно даже за счет наращивания ресурсов компьютеров, т. к. они присущи самим СУБД. Системы же, использующие серверы баз данных, могут поддерживать тысячи пользователей и огромные базы данных (в сотни гигабайт) за счет увеличения производительности как аппаратной, так и программной платформы сервера.

Безопасность

Сервер базы данных предоставляет мощные средства защиты данных от несанкционированного доступа, не возможные в настольных СУБД. Права доступа могут устанавливаться очень гибко — до уровня полей таблиц. Можно, например, вообще запретить пользователю прямое обращение к таблицам, регламентируя взаимодействие пользователя только через промежуточные объекты — представления и хранимые процедуры.

Способы преобразования приложений Access для использования их с SQL Server

Предположим, у вас имеется приложение Microsoft Access, которое необходимо приспособить для работы с сервером базы данных (например, Microsoft SQL Server). Существует несколько способов выполнения такого преобразования.

1. Можно перенести все таблицы или часть таблиц, хранящиеся в приложении Access, на сервер данных, предоставляющий доступ к данным через стандартный интерфейс ODBC. Затем присоединить эти таблицы в приложении Access с помощью механизма связывания таблиц с источником данных ODBC. Никакие другие объекты приложения при этом не изменятся.
2. Средствами Microsoft Access подключить клиентское приложение к базе данных на SQL Server. Предварительным шагом к созданию такого приложения является преобразование его таблиц в формат Microsoft SQL Server и перенос их с клиента на сервер.

   Далее клиентское приложение можно постепенно оптимизировать, ликвидируя в первую очередь узкие места, оставляя по возможности интерфейс неизменным. В процессе развития приложения в качестве инструментальной среды для реализации дополнительных функциональных возможностей может быть выбрана сре-

да программирования Visual Basic или средства создания динамических Web-страниц для работы с данными с помощью программы просмотра Интернета.

3. Можно перенести все объекты данных на SQL Server, а объекты приложений — в новый файл проекта Access (файл с расширением adp), создавая таким способом полноценное клиент-серверное приложение.

Чтобы получить клиент-серверное приложение, следует воспользоваться специальной надстройкой Access, называемой **Мастер преобразования в формат SQL Server** (Upsizing Wizard) для интеграции приложения Access с SQL Server. Она позволяет преобразовать и перенести базу данных Access в существующую или новую базу данных на SQL Server 2000, 7.0 или 6.5 (или на ограниченной версии сервера MSDE или MSDE2000), либо создать новый проект Access.

А перед этим можно разделить файл приложения Access на два файла: файл объектов данных и файл объектов приложения.

Разделение баз данных

Небольшие приложения Microsoft Access обычно состоят из одного файла базы данных. Сетевое приложение удобнее создавать так, чтобы оно включало два файла MDB: первый — *файл объектов данных* (в нем хранятся таблицы), второй — *файл объектов приложения* (в нем хранятся все остальные объекты: формы, запросы, отчеты, страницы доступа к данным, макросы и модули VBA). При этом в файле объектов приложения устанавливаются связи с таблицами, хранящимися в файле объектов данных.

Файл объектов данных называют частью *заднего плана* приложения (back-end), а файл объектов приложения — частью *переднего плана* (front-end). Файл объектов данных обычно размещается на файловом сервере, а копии приложений переднего плана — на клиентских рабочих станциях.

Разделение базы данных дает следующие преимущества:

- *В однопользовательской среде* можно обновлять объекты приложения, не оказывая влияния на существующие данные. При этом приложение обновляется простой заменой файла объектов приложения. Альтернативой этому способу является трудоемкая и часто вызывающая ошибки операция обновления объектов приложения в базе данных, состоящей из одного файла.
- *В многопользовательской среде* одни и те же данные могут совместно использоваться всеми пользователями приложения, поскольку файл объектов данных размещается на файловом сервере. Файл объектов приложения тоже можно разместить на файловом сервере для общего доступа, а можно для каждого пользователя создать отдельную копию файла объектов приложения, поместив ее на локальный диск клиентской рабочей станции. Тогда каждый пользователь сможет быстро запускать локальную копию приложения и совместно пользоваться файлом объектов данных, расположенным на файловом сервере. Использование локальных копий приложения повышает его производительность, тогда как запуск несколькими пользователями приложения, расположенного на сервере, увеличивает нагрузку на сеть. Поэтому в сетевой среде рекомендуется использовать разделение базы данных. В такой среде легче обновлять разделенную базу данных, поскольку обновление файла объектов приложений не затрагивает файла

объектов данных на сервере. Если в многопользовательской среде используется несколько копий файла объектов приложения, то нужно не забыть обновлять все эти копии.

Чтобы изучить механизм разделения базы данных, используйте копию нужной базы данных, помещенную, например, в папку Shared (Общие). Для разделения файла базы данных в Microsoft Access используется специальная надстройка — Мастер разделения баз данных (Database Splitter). Далее описана процедура разделения файла базы данных.

1. Откройте в Access 2003 базу данных, которую вы будете разделять. Например, можно разделить базу данных `Борей`, входящую в комплект примеров Microsoft Access 2003.
2. Активизируйте окно базы данных и выберите команду **Сервис | Служебные программы | Разделение баз данных** (Tools | Database Utilities | Database Splitter). Появится диалоговое окно Мастера разделения базы данных, приведенное на рис. 19.1. В окне сообщается, что таблицы текущей базы данных будут перемещены в новую базу данных и что эта операция может занять некоторое время. Рекомендуется создать резервную копию базы данных перед ее разделением на случай неожиданного сбоя работы Мастера или операционной системы. Для разделения базы данных нажмите кнопку **Разделить** (Split Database).

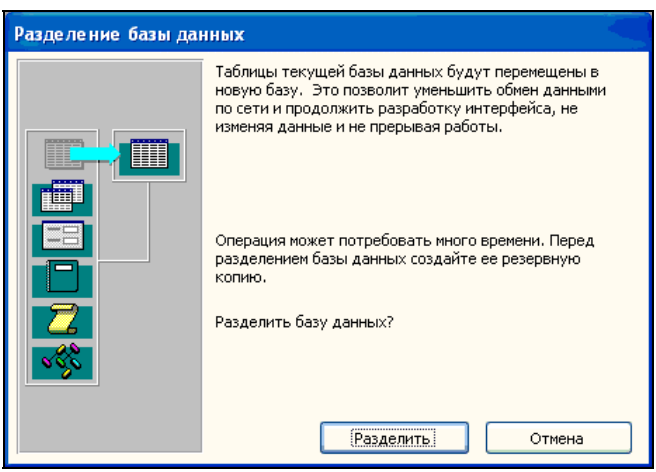


Рис. 19.1. Диалоговое окно **Разделение базы данных**

3. После нажатия кнопки **Разделить** (Split Database) появится диалоговое окно **Создание базы данных с таблицами** (Create Back-end Database), представленное на рис. 19.2. В поле **Имя файла** (File name) отображается имя файла объектов данных, по умолчанию состоящее из имени базы данных и специального суффикса, например: Борей _be.mdb (be — от back-end).
4. Нажмите кнопку **Разделение** (Split). После процесса разделения базы данных (он может длиться несколько минут) появится сообщение, что база данных успешно разделена (рис. 19.3).

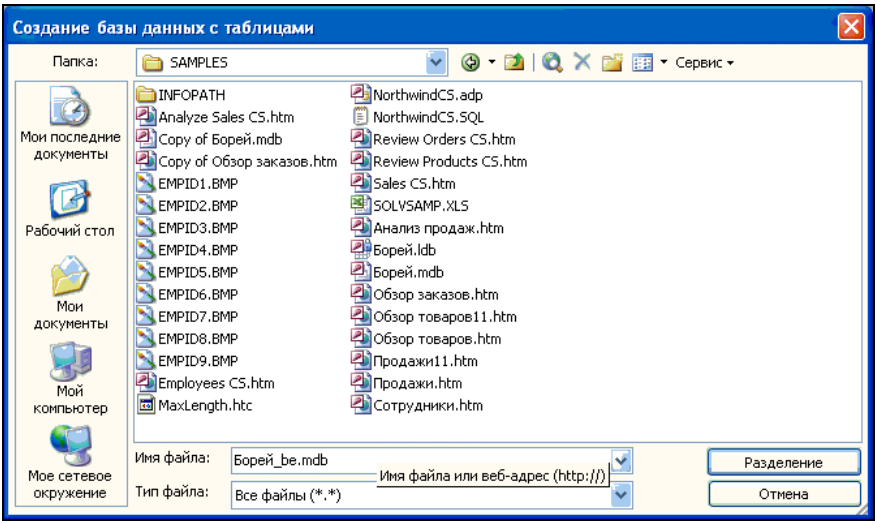


Рис. 19.2. Диалоговое окно **Создание базы данных с таблицами**

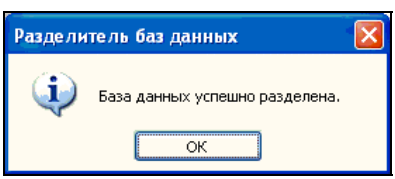


Рис. 19.3. Сообщение об успешном завершении операции разделения

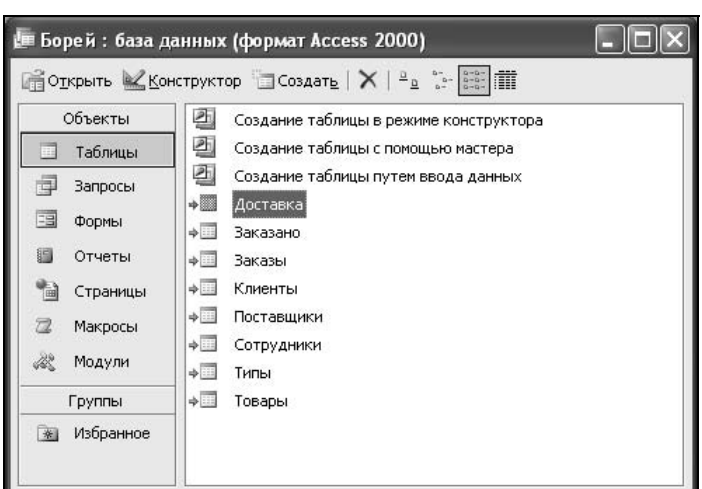


Рис. 19.4. Список таблиц в окне базы данных

5. Нажмите кнопку **OK**. Щелкните по ярлыку **Таблицы** (Tables) в окне базы данных. Слева от всех таблиц появятся стрелки, означающие, что таблицы хранятся во внешнем файле (рис. 19.4).
6. Для того чтобы убедиться, что все связи с таблицами во внешнем файле установлены правильно, выберите команду **Сервис | Служебные программы | Диспетчер связанных таблиц** (Tools | Database Utilities | Linked Table Manager). Появится диалоговое окно **Диспетчер связанных таблиц** (Linked Table Manager), в котором указаны пути расположения всех связанных таблиц (рис. 19.5). С помощью Диспетчера связанных таблиц можно выбрать таблицы для обновления путей к соответствующим файлам на диске.

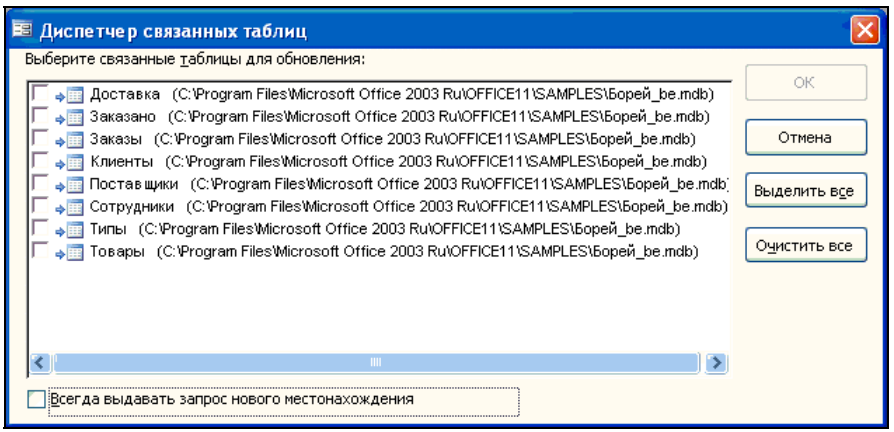


Рис. 19.5. Диалоговое окно **Диспетчер связанных таблиц**

Перенос сетевого приложения на платформу SQL Server

Прежде чем использовать Мастер преобразования в формат SQL Server (Upsizing Wizard), необходимо выполнить некоторые подготовительные действия, а после завершения процесса переноса нужно просмотреть автоматически создаваемый отчет о результатах переноса и выполнить ряд действий, позволяющих подготовить полученное клиент-серверное приложение к дальнейшему использованию. Таким образом, процесс переноса разделяется на три этапа: подготовительный этап, выполнение переноса и подготовка приложения к эксплуатации.

Прежде чем приступить к переносу реального приложения, мы советуем провести подобный эксперимент, используя приложение `Борей` или любую другую демонстрационную базу данных.

Подготовительный этап

Чтобы подготовить приложение к переносу на платформу SQL Server, рассмотрим процесс переноса базы данных Access в Microsoft SQL Server на примере приложения `Борей`, входящего в комплект демонстрационных приложений Microsoft Access:

1. Создайте резервную копию базы данных. Хотя Мастер преобразования в формат SQL Server (Upsizing Wizard) не удаляет из исходного файла переносимые объекты, лучше создать резервную копию базы данных.
2. Если нужно связать переносимое приложение с уже имеющейся на SQL Server базой данных, предварительно создайте ссылку на этот источник данных с помощью элемента **Администратор ODBC** (ODBC Data Sources) на Панели управления (Control Panel) Windows. Если планируется создать новую базу данных, необходимо иметь ссылку на **Основную базу данных** (Master database) соответствующего сервера.
3. Проверьте, что на сервере есть логическое устройство (device) достаточного размера. Лучше иметь избыток свободного дискового пространства. Если после переноса базы данных на SQL Server 6.5 планируется ее расширить или добавить большой объем данных, зарезервируйте в два или более раз больше дискового пространства, чем текущий размер базы данных. SQL Server 7.0 или выше автоматически увеличивает исходный размер базы данных при его нехватке.
4. Присоединенные к базе данных таблицы должны иметь уникальные индексы, если требуется перенести их на SQL Server. Мастер преобразования в формат SQL Server (Upsizing Wizard) переносит имеющиеся индексы на сервер, но не может создать новые. Поэтому, если хотите, чтобы перенесенные таблицы можно было обновлять, удостоверьтесь, что все переносимые таблицы снабжены уникальными индексами.
5. Если перенос осуществляется на SQL Server 6.5, при необходимости создайте новые логические устройства. Мастер преобразования в формат SQL Server (Upsizing Wizard) создает новые логические устройства на том же физическом диске, где находится основное логическое устройство. Если на сервере установлено несколько жестких дисков, можно поместить базу данных и журнал транзакций на разные диски. Тогда, если с одним диском что-нибудь случится, базу данных можно будет восстановить с другого. На обоих дисках необходимо иметь (или создать) логические устройства достаточного размера.

   Структура хранения данных Microsoft SQL Server 7.0 и выше является более гибкой: вместо логических устройств используются файлы, поэтому при переносе базы данных на SQL Server не нужно заботиться о создании логических устройств.
6. Хотя SQL Server 6.5 позволяет использовать для хранения базы данных и журнала транзакций несколько логических устройств, Мастер преобразования в формат SQL Server (Upsizing Wizard) потребует указать одно устройство для базы данных и одно устройство для журнала транзакций. Чтобы указать несколько логических устройств для базы данных или журнала транзакций, сделайте эти логические устройства (и только их) используемыми по умолчанию. После этого в процессе работы с мастером выберите для базы данных или журнала транзакций устройство по умолчанию. Если суммарный размер логических устройств, используемых по умолчанию, превышает размер базы данных или журнала транзакций, SQL Server будет применять только необходимые для хранения логические устройства.
7. Установите принтер, используемый по умолчанию. Это необходимо сделать, поскольку Мастер преобразования в формат SQL Server (Upsizing Wizard) автоматически создает отчет о результатах своей работы в виде снимка отчета (snapshot).

8. Проверьте наличие соответствующих прав доступа к базе данных Access. Необходимо обладать правами на чтение и изменение структуры (Read и Design) всех объектов базы данных, чтобы провести процесс переноса базы данных на клиент-серверную архитектуру.
9. Проверьте наличие необходимых прав доступа к базе данных SQL Server. Разные способы переноса приложения в SQL Server требуют наличия прав различного уровня:
   - чтобы провести интеграцию приложения Access с существующей базой данных на SQL Server, необходимо обладать правами на создание таблиц (Create table) и создание по умолчанию (Create default);
   - чтобы создать новую базу данных в SQL Server, необходимо обладать правами на создание базы данных (Create database) и операцию выбора (Select) в системных таблицах Основной базы данных (Master database);
   - чтобы создать новые логические устройства, необходимо обладать правами системного администратора SQL Server.
10. Проверьте имена полей и таблиц в переносимой базе данных. При переносе на платформу SQL Server 6.5 будут использованы ограничения на имена, все неправильные символы в именах в результате переноса в SQL Server будут заменены символом подчеркивания "_". Эти правила заключаются в следующем: имена могут содержать до 30 символов, первым символом должна быть буква или знак "@" (коммерческое at), для остальных символов допускается использовать букву, цифру, знак доллара "$", знак номера "#" и знак подчеркивания "_". Пробелы использовать запрещено.

    Хотя SQL Server 7.0 и выше поддерживает имена, допустимые в Access, рекомендуется придерживаться правил именования объектов, принятых в версии SQL Server 6.5.

Предположим, с помощью программы Enterprise Manager на SQL Server создана новая база данных `Sample`, которая будет использоваться для переноса приложения `Борей`. Создайте ссылку на этот источник данных в SQL Server с помощью Администратора ODBC на Панели управления Windows.

Выполнение переноса

При переносе базы данных в SQL Server, кроме таблиц, переносятся индексы и связи между таблицами. При этом индексы отображаются в соответствующие объекты SQL Server. Индексы, использующиеся в Access и SQL Server, очень похожи, отличие только в том, что SQL Server не поддерживает индексы по убыванию. Первичные ключи из Access отображаются в некластеризованные уникальные индексы в SQL Server. В SQL Server 6.*x* и выше они, кроме того, маркируются как первичные ключи.

Если перенос выполняется в SQL Server 6.*x* или выше, то при переносе связей вы можете выбрать один из методов задания связей в SQL Server: *декларативная ссылочная целостность* (Declarative Referential Integrity, DRI) или *триггер* (см. рис. 19.10). DRI — это свойство SQL Server, которое позволяет задавать связи между таблицами в описаниях таблиц в виде внешних ключей (foreign keys). Однако DRI не поддерживает каскадных обновлений и удалений. Поэтому, чтобы сохранить те описания

каскадных операций, которые были в базе данных Access, следует использовать триггеры *(примеры триггеров и индексов приведены в гл. 17)*.

При переносе таблиц переносятся все поля, которые были в таблицах Access, и такие их свойства, как правила проверки введенных значений (validation rules) и значения по умолчанию (default values).

Замечание

Особенным образом осуществляется перенос полей таблиц Access типа **Гиперссылка** (Hyperlink). SQL Server не поддерживает такие поля. При переносе на SQL Server поля типа **Гиперссылка** (Hyperlink) преобразуются в текстовые поля, содержащие все свойства гиперссылки в одной строке: ее название, всплывающую подсказку, адрес и закладку ресурса. Но при открытии таблиц, хранящихся на SQL Server, в клиентском приложении Access гиперссылки полностью работают, ведь вся информация сохранена.

Для таблиц Access, имеющих поля типа **Одинарное с плавающей точкой** (Single) (это 4 байт), **Двойное с плавающей точкой** (Double) (это 8 байт), **Поле MEMO** (Memo), **Поле объекта OLE** (OLE Object), в результате переноса в SQL Server создается еще поле типа **Штамп времени** (timestamp). Это поле Microsoft Access использует при работе с присоединенными таблицами сервера для проверки, не было ли изменений в записи. При отсутствии этого поля работа с такими таблицами может быть не всегда корректной.

Правила проверки введенных значений (validation rule) для полей таблиц Access обычно представляются в SQL Server в виде триггеров, т. к. правила (rules) в SQL Server не позволяют выдавать сообщения об ошибках.

Чтобы начать процесс переноса базы данных на платформу SQL Server:

1. Откройте в Microsoft Access базу данных, которую требуется перенести.
2. Выберите команду **Сервис | Служебные программы | Мастер преобразования в формат SQL Server** (Tools | Database Utilities | Upsizing Wizard). Появится диалоговое окно, представленное на рис. 19.6.
3. Если нужно создать новую базу данных на SQL Server, в которую будут перенесены таблицы из приложения Access, переходите к шагу 5. Чтобы связать приложение Access с существующей базой данных в SQL Server (в нашем примере это база данных `Sample` на SQL Server), выберите переключатель **использовать существующую базу данных** (Use existing database) и нажмите кнопку **Далее** (Next). Появится диалоговое окно, представленное на рис. 19.7.
4. Чтобы указать файловый источник данных, выделите один из существующих источников данных в списке на вкладке **File Data Source** или нажмите копку **New**, чтобы создать новый. Если нужно использовать некоторый специфический для данного компьютера источник данных (к ним относятся пользовательские или системные источники), раскройте вкладку **Machine Data Source** (см. рис. 19.7) и выделите соответствующий элемент в списке существующих источников данных (например, созданный с помощью Администратора ODBC системный источник данных SampleSDSN, подключенный к базе данных `Sample` на SQL Server) или нажмите кнопку **New** (Создать), чтобы создать новый. После указания источника данных нажмите кнопку **OK** и переходите к шагу 7.

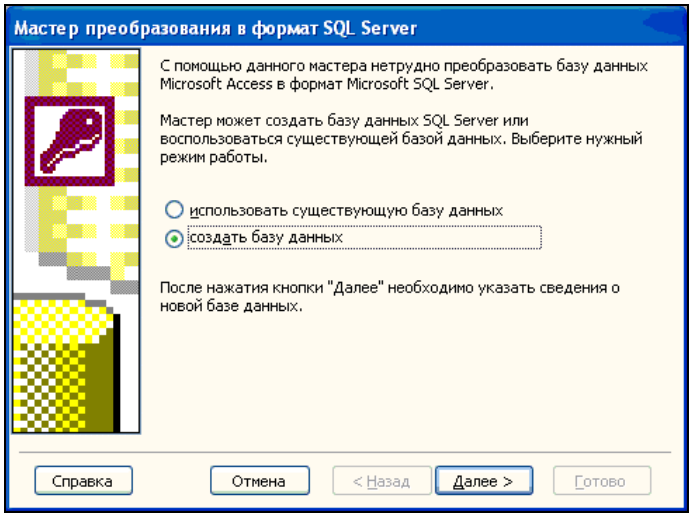


Рис. 19.6. Диалоговое окно выбора способа переноса базы данных

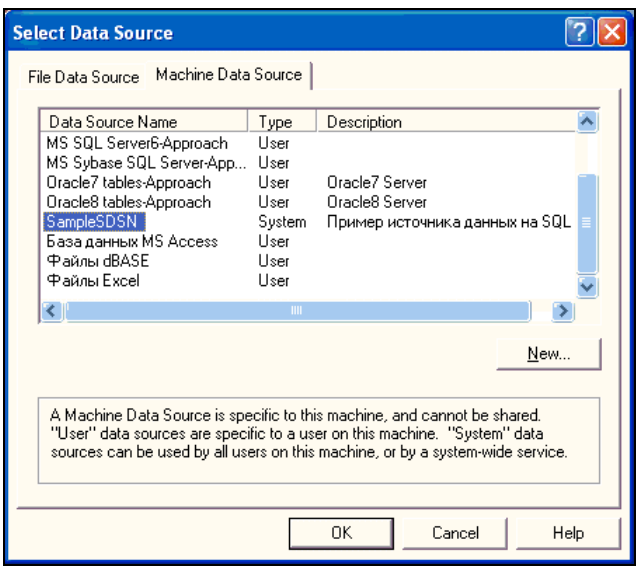


Рис. 19.7. Вкладка **Machine Data Source**

5. Если нужно создать новую базу данных на SQL Server, в которую будут перенесены таблицы из приложения Access, выберите в первом диалоговом окне Мастера преобразования в формат SQL Server (Upsizing Wizard) переключатель **Создать базу данных** (Create new database), нажмите кнопку **Далее** (Next). Появится диалоговое окно, представленное на рис. 19.8.

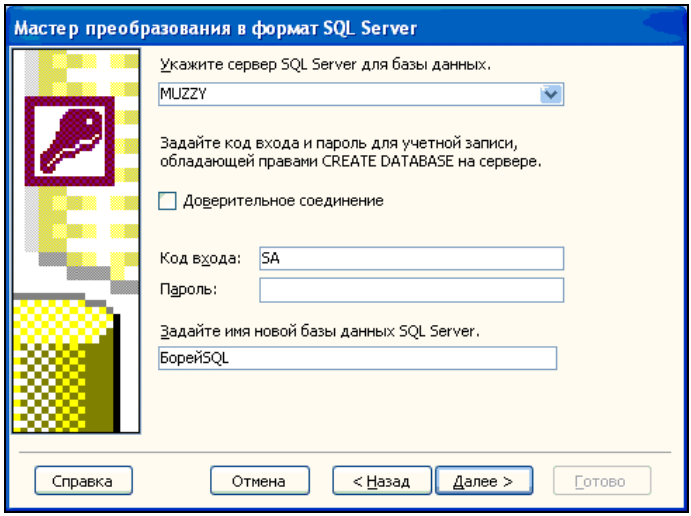


Рис. 19.8. Диалоговое окно для регистрации новой базы данных в SQL Server

6. Выберите в списке имя сервера, на котором нужно создать новую базу данных, выберите способ регистрации на SQL Server. Чтобы использовать регистрацию на SQL Server через систему защиты Windows NT, 2000 или XP, установите флажок **Доверительное соединение** (Trusted Connection). Иначе для регистрации непосредственно на SQL Server введите в текстовые поля имя пользователя и пароль. Введите также имя новой базы данных. Нажмите кнопку **Далее** (Next).

7. На этом этапе будет установлено соединение с SQL Server. Для чего, возможно, потребуется ввести имя пользователя и пароль, либо будут использованы параметры регистрации, заданные на предыдущих этапах.

8. В следующем диалоговом окне Мастера можно выбрать те таблицы, которые требуется переместить на SQL Server (рис. 19.9). Чтобы перенести все таблицы, нажмите кнопку >> и кнопку **Далее**, чтобы перейти к следующему окну.

9. Как уже говорилось ранее, при переносе таблиц на сервер, имеется возможность перенести не только сами таблицы, но и некоторые атрибуты таблиц — индексы, правила проверки вводимых значений, значения по умолчанию, а также связи между таблицами. Сбросьте лишние флажки или оставьте все, как предлагает Мастер по умолчанию (рис. 19.10). Кроме этого, Мастер предлагает добавить в некоторые таблицы специальное поле — *штамп времени* (timestamp). При этом можно добавлять эти поля в каждую таблицу, можно не добавлять совсем, а можно предоставить Мастеру самому решить, в каких случаях добавить штамп времени. Можно также установить флажок, который указывает, что на сервере требуется создать только структуру таблиц и не переносить данные. Оставьте все значения такими, как предлагает Мастер по умолчанию, и нажмите кнопку **Далее**.

10. На следующем шаге Мастер предлагает выбрать способ преобразования приложения (рис. 19.11):

- просто переместить таблицы на SQL Server, не изменяя самого приложения;
- переместить таблицы и связать их с приложением. Тогда Мастер сохраняет исходные таблицы Access, переименовывая их при этом (добавляется суффикс `_локальная` (_local)), присоединяет преобразованные таблицы SQL Server к приложению в виде связанных таблиц ODBC с теми же именами, которые они имели до переноса таблицы Access. Таким образом, для того чтобы преобразованное приложение оставалось работоспособным, больше ничего не требуется изменять;

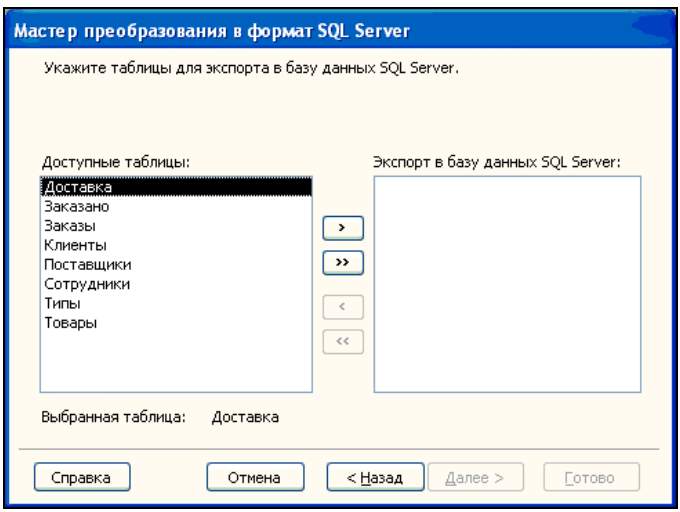


Рис. 19.9. Диалоговое окно для выбора таблиц, которые нужно перенести на SQL Server

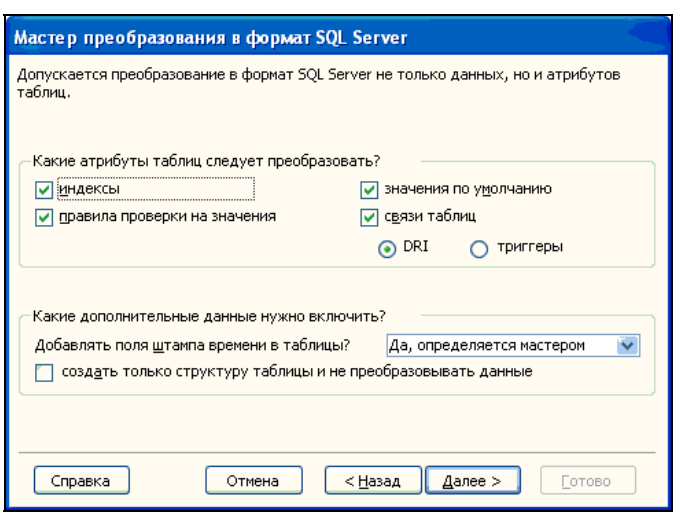


Рис. 19.10. Диалоговое окно для выбора атрибутов таблиц, которые нужно перенести на SQL Server

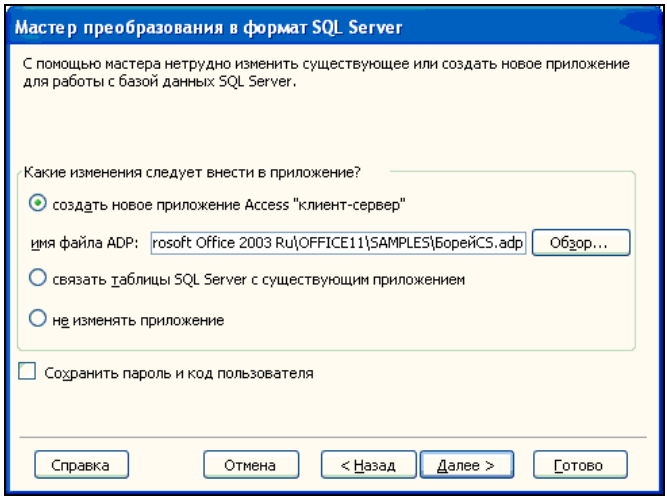


Рис. 19.11. Выбор способа преобразования приложения

- создать новое приложение — проект Access, в которое переносятся все объекты приложения и которое взаимодействует с таблицами на сервере через новый интерфейс — OLE DB/ADO.

Выберите желаемый вариант преобразования. При этом в случае создания проекта Access нужно задать имя файла проекта (с расширением adp). Можно также установить флажок, указывающий на необходимость сохранения имени пользователя и пароля для регистрации.

11. В последнем окне выдается сообщение о том, что вся необходимая информация получена и Мастер готов начать преобразование. Возможно, Мастер предложит выбрать, какой из файлов требуется открыть после преобразования: новый проект Access или оставить открытой исходный файл MDB. Выберите необходимый переключатель и нажмите кнопку **Готово** (Finish).
12. После этого начнется процесс преобразования и переноса объектов базы данных без отображения дополнительных диалоговых окон, требующих подтверждения каких-либо действий. Это удобно, когда осуществляется перенос объемной базы данных и этот процесс может продлиться достаточно долго. При этом отображается окно с индикатором выполнения переноса и кнопка **Отмена** (Cancel), позволяющая в любой момент прервать этот процесс (рис. 19.12).

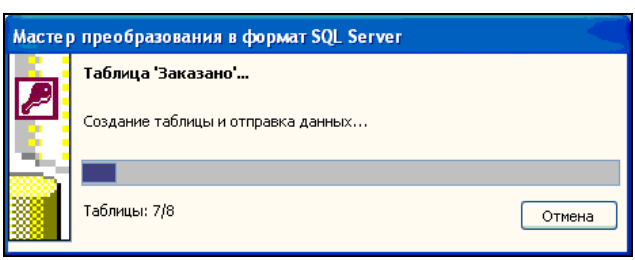


Рис. 19.12. Диалоговое окно с индикатором процесса преобразования

13. По окончании процесса переноса отображается отчет о полученных результатах, который можно сохранить с помощью команды **Файл | Экспорт** (File | Export), например, в формате снимка отчета (snapshot) или HTML. Когда отчет будет просмотрен и закрыт, на экране отобразится окно исходной базы данных или преобразованного приложения Access.

Подготовка клиент-серверного приложения к эксплуатации

После переноса базы данных в SQL Server Мастер преобразования в формат SQL Server (Upsizing Wizard) присоединяет к вашему приложению все созданные на сервере таблицы, старые таблицы делает локальными и создает документ — Отчет о переносе базы данных (Upsizing Report), в котором описаны результаты работы Мастера и все возникшие проблемы, если они были. Полезно его изучить, прежде чем работать с обновленным приложением.

Чтобы подготовить базу данных к работе после переноса на платформу SQL Server:

1. Проверьте, что те таблицы, которые нужно редактировать из Access, являются изменяемыми. Для этого нужно, чтобы такая таблица обязательно имела хотя бы один уникальный индекс.
2. Установите права доступа к таблицам на SQL Server, т. к. все группы, пользователи и права, которые были установлены в Access, не распространяются на серверные таблицы.
3. Запретите, если нужно, сохранение паролей локально в клиентском приложении.
4. Спланируйте создание резервных копий базы данных и, если нужно, "зеркальных копий" логических устройств.

Скорее всего, ваше клиентское приложение Access придется модифицировать, чтобы оно эффективно работало с SQL Server.

Использование приложений предыдущих версий Microsoft Access

Чтобы использовать в Access 2003 существующие приложения, разработанные в предыдущих версиях Access, необходимо выяснить, возможно ли это и что для этого требуется сделать.

Поддержка 2000 года в Microsoft Access

Правила обработки вычислений, использующих даты, в Access достаточно сложны, поэтому обработка дат обладает высокой точностью. При интерпретации двусмысленных дат в Access, начиная с версии Access 2000, делает некоторые предположения. Например, если дата включает только название месяца и одну или две цифры, Access предполагает, что число от 1 до 31 определяет день указанного месяца текущего года. Допустим, пользователь ввел дату `декабря 2`, тогда Access воспримет ее как 2 декабря текущего года, а не как 1 декабря 2002 г. Если пользователь ввел дату `декабря 32`, Access воспримет ее как 1 декабря 1932 г.

В Access обработка дат, имеющих сокращенное указание года одной или двумя цифрами, происходит с учетом дат XXI века, как показано в табл. 19.2.

Таблица 19.2. *Поддержка 2000 года*

Даты в формате Access	Интерпретация дат
1.1.00 — 31.12.29	1.1.2000 — 31.12.2029
1.1.30 — 31.12.99	1.1.1930 — 31.12.1999

Чтобы быть уверенным, что Access правильно проинтерпретирует введенную дату, указывайте номер года четырьмя цифрами (например, 2001 вместо 01).

Администратор приложения или системный администратор может настроить Microsoft Office, исходя из пожеланий пользователей, чтобы определить правила интерпретации дат. Например, есть возможность изменить нижнюю границу интерпретации двузначных номеров года дат 20-го столетия на меньшее число, чем 30 (см. табл. 19.2). Администратор может также изменить формат дат, используемый по умолчанию, для отображения четырехзначных номеров года вместо двузначных.

Информацию о настройке интерпретации дат можно найти на Web-узле Microsoft.

Совместимость форматов баз данных разных версий Microsoft Access

При разработке новых версий Access фирма Microsoft стремится обеспечить совместимость снизу вверх всех созданных ранее приложений.

В новой версии Access 2003 доступен усовершенствованный формат баз данных Access 2002—2003, в котором доступны все новые возможности Access, начиная с версии Access 2002. Формат более ранней версии Access 2000 поддерживается полностью в Access 2002 и более поздних версиях, т. е. базы данных и проекты Access 2000 можно беспрепятственно открывать и изменять в них объекты, данные и их структуру, пользуясь новыми инструментальными средствами версии Access 2002 и выше. Хотя, если возникнет потребность в использовании всего спектра новых возможностей Access, базу данных или проект Access 2000 можно будет *преобразовать* в новый формат Access 2002—2003.

Файлы баз данных ранних версий Access (1.0, 1.1, 2.0, 95, 97) существенно отличаются по структуре от формата баз данных Access 2000. Поэтому для полноценного использования прежних версий этих файлов необходимо их *преобразование* в форматы более поздних версий Access. Новые версии Access снабжаются специальными утилитами для преобразования файлов баз данных предшествующих версий. Многокомпонентная структура MDB-файлов усложняет процедуру их преобразования и не всегда допускает полную совместимость версий. Неполная совместимость с ранними версиями приводит к ограничению набора допустимых операций для работы с преобразованными таблицами. В формат Access 2000 или 2002—2003 можно преобразовать только базы данных, начиная с версии 2.0 и выше. Чтобы использовать в Access 2002 и выше базы данных более ранних версий, требуется сначала преобразовать их в формат Access 2.0, 95, 97 или 2000 средствами этих версий Access.

Базу данных Access 2.0, 95 или 97 не обязательно преобразовывать, чтобы использовать ее в версии Access 2002 или выше, если не требуется изменять в ней структуру данных и объектов или создавать новые объекты. Эту базу данных можно просто открыть без преобразования в версии Access 2002 или выше.

Общая схема поддержки форматов предыдущих версий Microsoft Access (без преобразования) для Access 2000, 2002 и выше представлена в табл. 19.3. Каждая строка таблицы содержит сведения о типе совместимости соответствующей версии Access с другими версиями, которые перечислены в столбцах таблицы. В этой таблице использованы следующие условные обозначения:

- *Чтение* — Access позволяет читать формат файла указанной версии без использования дополнительной программы просмотра или утилиты преобразования форматов;
- *Запись* — Access позволяет записывать изменения в исходном формате указанной версии практически без потерь эффективности полученного файла из-за различий между версиями;
- *Изменение структуры и создание* — Access позволяет создавать базы данных и изменять структуру данных и объектов баз данных в формате указанной версии.

Таблица 19.3. *Совместимость форматов баз данных разных версий Microsoft Access*

Версия Access	Формат Access 2002–2003	Формат Access 2000	Формат Access 97	Формат Access 95	Формат Access 2.0
Access 2002 и выше (например, Access 2003)	Чтение, Запись. Изменение структуры и создание	Чтение, Запись. Изменение структуры и создание	Чтение, Запись	Чтение, Запись	Чтение, Запись
Access 2000		Чтение, Запись. Изменение структуры и создание	Чтение, Запись	Чтение, Запись	Чтение, Запись

Совместное использование базы данных в нескольких версиях Access

В некоторых случаях не все пользователи многопользовательского приложения Access могут перейти к использованию новой версии Access одновременно. Хотя бы потому, что не все компьютеры организации могут иметь достаточные для этого ресурсы. Access позволяет нескольким пользователям работать с одними и теми же данными в файле базы данных, используя при этом разные версии Access. Например, база данных может быть создана в среде Access 95 или даже Access 2.0 и иметь

соответствующий формат. Часть пользователей может продолжать работать с этой базой данных в той же среде Access 2.0 или Access 95, а другие пользователи могут работать с этим же файлом в среде Access 2002 или выше, имея возможность использовать большинство новых средств Access. Единственное ограничение для пользователей Access 2002 или выше — они не смогут менять структуру базы данных и создавать в ней новые объекты. Сами данные доступны им как на чтение, так и на запись. Такую стратегию можно использовать для приложений, состоящих из одного файла MDB, и для разделенных на части front-end/back-end-приложений.

Чтобы использовать разделенную базу данных из предыдущей версии в разных версиях Access, достаточно преобразовать файл объектов приложения в формат Access 2000 или Access 2002—2003 и присоединить к нему таблицы из исходного файла объектов данных.

Чтобы использовать в разных версиях Access базу данных, состоящую из одного файла в формате одной из предыдущих версий Access:

1. Преобразуйте базу данных в формат Access 2000 или 2002—2003, используя новое имя файла для преобразованной базы данных. *(См. разд. "Использование файлов объектов приложения предыдущих версий Access" этой главы.)*
2. Разделите преобразованную базу данных на файл объектов данных и файл объектов приложения с помощью Мастера разделения баз данных (Database Splitter), как описано в *разд. "Разделение баз данных" этой главы.*
3. Удалите полученный в результате разделения файл объектов данных. В качестве части заднего плана (back-end) для данного приложения необходимо использовать данные, хранящиеся в исходной базе данных. Часть back-end должна представлять собой файл в самой ранней версии Access, из тех, в которых работают пользователи приложения.
4. Откройте файл объектов приложения в Access 2000, 2002 или выше. Щелкните по ярлыку **Таблицы** (Tables). Щелкните правой кнопкой мыши по пустому месту в окне базы данных и выберите в контекстном меню команду **Связь с таблицами** (Link Tables). Установите ссылки на таблицы, хранящиеся в предыдущей версии базы данных.
5. После этого можно расширять возможности части переднего плана (front-end) базы данных, используя все новые средства Access 2000 и выше, которые можно предоставить в распоряжение пользователей, перешедших на применение новой версии Access. Пользователи предыдущих версий Access могут продолжать пользоваться исходной версией базы данных. Например, если файл объектов данных сохранен в формате Access 2.0, можно использовать множество версий Access для части приложения переднего плана: Access 2.0 (исходное приложение), Access 95 и 97 (исходная или преобразованная версия файла объектов приложения) и Access 2000 и выше (преобразованная версия файла объектов приложения).

Если база данных предыдущей версии Access уже разделена на файл объектов данных и файл объектов приложения, остается только преобразовать файл объектов приложения в формат Access 2000 или 2002—2003:

1. Оставьте файл объектов данных без изменений.
2. Преобразуйте файл объектов приложений в формат Access 2000 или 2002—2003.

3. Откройте преобразованный файл объектов приложения в Access 2000 или более поздней версии и установите связи к таблицам, хранящимся в файле объектов данных с помощью Диспетчера связанных таблиц (Linked Table Manager).
4. После этого можно наращивать функциональность файла объектов приложения, используя все новые возможности соответствующей версии Access.

(Об использовании Диспетчера связанных таблиц см. разд. "Обновление связей с присоединенными таблицами" гл. 3.)

Использование в Access баз данных предыдущих версий без преобразования формата

В Access 2000 и выше можно открывать базы данных предыдущих версий, не преобразуя их в новый формат. Поскольку исходный формат базы данных не меняется, остальные пользователи могут продолжать использовать эту базу данных в ранней версии Access. При этом работа с базами данных Access 2000 в Access 2002 и выше ничем не будет ограничена, но пользователи Access 2000 или выше будут иметь ограничения при использовании базы данных более ранних версий: Access 2.0, 95 или 97. Можно будет просматривать и редактировать данные в таблицах, добавлять и удалять записи, но нельзя будет изменять структуру данных и объектов базы данных или добавлять новые объекты. Для этой цели базу данных необходимо открыть в той версии Access, в которой она была сохранена.

Существуют определенные ограничения при работе с базами данных предыдущих версий Access из-за неполной совместимости разных версий.

- Базу данных в формате Access 2000 или 2002—2003 нельзя открыть в предыдущих версиях Access. В частности, с базой данных в формате Access 2002—2003 нельзя работать в режиме общего доступа вместе с пользователями Access 2000 и 97.
- Нельзя установить связи с таблицами базы данных Access 2000 или 2002—2003 в базе данных предыдущей версии. Однако можно экспортировать таблицы Access 2000 или 2002—2003 в формат предыдущей версии Access.
- Microsoft Access 97 и выше используют новый стиль панелей команд — строк меню, контекстных меню и панелей инструментов. Если база данных Access 95 открыта без преобразования в Access 2000 или выше, пользовательские панели инструментов преобразуются согласно стилю Access 2000 или выше, однако это преобразование не сохраняется. Пользовательские строки меню интерпретируются как строки меню нового стиля, однако макросы строки меню не преобразуются и продолжают действовать, благодаря их поддержке в новых версиях Access.
- Защищенное приложение Access 2000 или более ранней версии остается таковым при открытии его в Access 2002 и выше. Можно с помощью Администратора рабочих групп (Workgroup Administrator) установить связь с файлом рабочей группы, созданным в предыдущей версии Access. Однако рекомендуется выполнить воссоздание файла рабочей группы в новой версии Access 2000 или выше перед установлением связи с ним. Если используется база данных Access 2.0, 95 или 97 с исходной версией файла рабочей группы, сохраняется действие защиты для всех объектов, кроме защиты модулей. В Access 2000 или выше не поддерживается защита на уровне пользователей для кода VBA. Вместо этого применяется за-

щита кода VBA с помощью пароля. Файл рабочей группы Access 2002 и выше нельзя использовать в версии 97 или еще более ранней.

- В результате открытия базы данных предыдущей версии Access без преобразования может увеличиться ее размер. Если база данных открывалась без преобразования в другой версии Access, а потом открывалась в Access 2002 или выше, ее размер может увеличиться еще больше. В редких случаях база данных Access 2.0 открывается без преобразования в Access 95, затем в Access 97, в Access 2000, в Access 2002 и т. д., тогда в проекте VBA всякий раз сохраняется информация, относящаяся к формату очередной версии.
- Относительно совместимости программного кода на VBA: Microsoft Access 2000 и выше не поддерживает библиотеку совместимости DAO 2.5/3.*x* (DAO 2.5/3.*x* Compatibility Library). Если открыть в Access 2000 или выше приложение ранней версии Access, которое содержит программный код, использующий устаревшие объекты, методы и свойства объектов DAO, связанные с этой библиотекой, отобразится сообщение об ошибках компиляции проекта VBA. Поэтому перед открытием базы данных в Access 2000 следует обновить программный код, чтобы он не зависел от библиотеки совместимости DAO 2.5/3.*x* (DAO 2.5/3.*x* Compatibility Library).

Использование файлов объектов приложения предыдущих версий Access

Если база данных, созданная в ранней версии Access (2.0, 95 или 97), впервые открывается в Access 2000 или выше, появляется диалоговое окно, представленное на рис. 19.13, предупреждающее о том, что база данных имеет более раннюю версию, поэтому она недоступна для изменений.

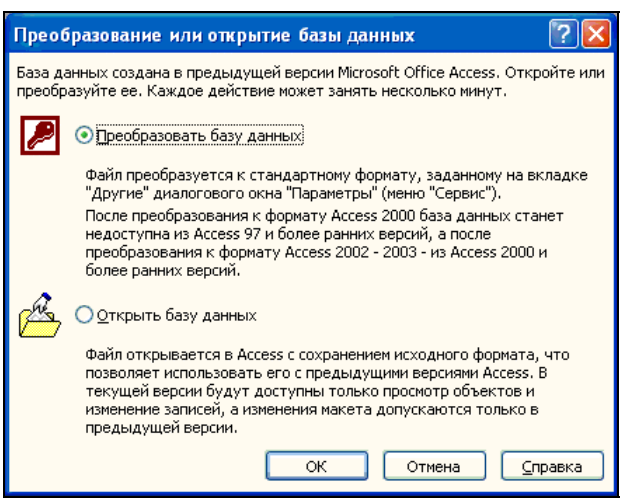


Рис. 19.13. Диалоговое окно **Преобразование или открытие базы данных**

Предлагается преобразовать выбранную базу данных в формат Access 2000 или 2002—2003, чтобы получить возможность работать с этой базой, изменяя структуру ее объектов, и использовать все возможности Access 2000 или выше. Выбор результирующего формата в данном случае зависит от того, какой формат считается выбранным по умолчанию: Access 2000 или 2002—2003. Чтобы выбрать тот или иной формат, перед преобразованием откройте какую-нибудь базу данных в Access 2003, выберите команду **Сервис | Параметры** (Tools | Options) и в появившемся диалоговом окне **Параметры** (Options) на вкладке **Другие** (Advanced) выберите желаемый формат в раскрывающемся списке **Формат файла по умолчанию** (Default File Format).

Чтобы преобразовать базу данных ранней версии в формат Access 2000 или 2002—2003, необходимо выбрать переключатель **Преобразовать базу данных** (Convert Database) и нажать кнопку **ОК**. Появится диалоговое окно, в котором нужно указать имя и путь для сохранения базы данных в формате новой версии. После нажатия кнопки **Сохранить** (Save) в строке состояния отобразятся сообщения **Преобразование** (Converting), затем **Компиляция** (Compiling), **Сжатие** (Compacting) и **Проверка системных объектов** (Verifying System Objects). После удачного завершения преобразования отобразится предупреждающее диалоговое окно о том, что преобразованную версию больше нельзя будет использовать в предыдущих версиях Access, и откроется обновленная база данных.

При преобразовании базы данных Access ранней версии, в которой содержатся ссылки на таблицы в другой базе данных, удостоверьтесь, что внешние таблицы хранятся в файле с заданным именем в той же папке, на которую указывают ссылки. Если Access не найдет связанные таблицы в процессе преобразования базы данных, вы не сможете использовать преобразованную базу данных. Как только база данных преобразована в формат Access 2000 или 2002—2003, переместите файл со связанными таблицами в нужную папку и переустановите связи на таблицы с помощью Диспетчера связанных таблиц (Linked Table Manager). Преобразование баз данных не затрагивает связанные таблицы — файл, в котором хранятся эти таблицы, остается в формате предыдущей версии Access.

Если необходимо сохранение формата базы данных, например, когда разработка базы осуществляется в команде с сотрудниками, которые по каким-либо причинам пользуются ранней версией Access, возможно открытие текущей базы данных без преобразования. В этом случае объекты будут доступны только для просмотра, но не для редактирования. Если все же принято решение открыть базу данных ранней версии в Access 2000 или выше без преобразования, следует выбрать переключатель **Открыть базу данных** (Open Database) в диалоговом окне, представленном на рис. 19.13, и нажать кнопку **ОК**. Access добавит в базу данных информацию, необходимую для ее работы в новой версии Access, и откроет ее. Если в дальнейшем потребуется преобразовать эту базу данных в новый формат Access 2000 или 2002—2003, откройте ее в Access 2003 и в меню **Сервис | Служебные программы | Преобразовать базу данных** (Tools | Database Utilities | Convert Database) выберите команду **в формат Access 2000** (To Access 2000 File Format) или **в формат Access 2002—2003** (To Access 2002—2003 File Format).

Использование файлов объектов данных предыдущих версий Access

Для разделенных баз данных возможно преобразование в формат Microsoft Access 2000 или выше только файла объектов приложения с сохранением прежней

версии файла объектов данных. Связанный файл объектов данных может быть совместно использован пользователями, работающими с 16-разрядной версией Access, с Access 95, 97, 2000, 2002 и с Visual Basic версии 3.0 и выше. Присоединение связанной таблицы из файла объектов данных происходит без вывода предупреждающих диалоговых окон. Однако при этом для файлов предыдущих версий нельзя будет воспользоваться некоторыми новыми возможностями Access 2000 или выше, например, такими как репликация.

Файлы баз данных ранних версий Access не следует преобразовывать в формат Access 2000 или 2002—2003 в следующих случаях:

- некоторые пользователи приложения не перешли к операционной системе Windows 95 или Windows NT, а работают в Windows 3.*x*, поэтому не могут использовать ни полную, ни исполняемую версию Access 2000 или выше;
- идет процесс преобразования нескольких файлов приложения из предыдущей версии в формат Access 2000 или 2002—2003. При этом необходимо сохранять совместимость файлов объектов данных со всеми используемыми файлами объектов приложения до полного окончания процесса преобразования;
- база данных разделяется с пользователями приложений Visual Basic 3.0.

Если с помощью предыдущих версий Access разработано несколько приложений для рабочей группы, необходимо оставить разделяемый файл объектов данных в исходном формате до тех пор, пока все файлы объектов приложения не будут преобразованы в формат Access 2000 или 2002—2003. Установление связи с таблицами и сохранение исходного формата файла объектов данных сопряжено с минимальными усилиями или совсем их не требует.

Варианты преобразования баз данных в Access 2003

Приложение Microsoft Access 2003 поддерживает несколько вариантов преобразования форматов баз данных. В том числе и преобразование к форматам некоторых предыдущих версий Access. Все эти варианты собраны в табл. 19.4.

Таблица 19.4. *Преобразование форматов в Microsoft Access 2003*

Исходный формат	Целевой формат	Способ преобразования
База данных Access 2.0, 95 или 97	База данных Access 2000 или 2002—2003	Если база данных впервые открывается в Access 2003, в появившемся при ее открытии диалоговом окне (см. рис. 19.13) выберите переключатель **Преобразовать базу данных** (Convert Database). Иначе откройте базу данных в Access 2003 и в меню **Сервис | Служебные программы | Преобразовать базу данных** (Tools | Database Utilities | Convert Database) выберите команду **в формат Access 2000** (To Access 2000 File Format) или **в формат Access 2002—2003** (To Access 2002—2003 File Format)

Таблица 19.4 *(окончание)*

Исходный формат	Целевой формат	Способ преобразования
База данных Access 2000	База данных Access 2002–2003 или 97	Откройте базу данных в Access 2003 и в меню **Сервис	Служебные программы	Преобразовать базу данных** (Tools	Database Utilities	Convert Database) выберите команду **в формат Access 2002–2003** (To Access 2002–2003 File Format) или **в формат Access 97** (To Access 97 File Format)
База данных Access 2002–2003	База данных Access 2000 или 97	Откройте базу данных в Access 2003 и в меню **Сервис	Служебные программы	Преобразовать базу данных** (Tools	Database Utilities	Convert Database) выберите команду **в формат Access 2000** (To Access 2000 File Format) или **в формат Access 97** (To Access 97 File Format)
Проект Access 2000	Проект Access 2002–2003	Откройте базу данных в Access 2003 и выберите команду **Сервис	Служебные программы	Преобразовать базу данных	в формат Access 2002–2003** (Tools | Database Utilities | Convert Database | To Access 2002–2003 File Format).
Проект Access 2002–2003	Проект Access 2000	Откройте базу данных в Access 2003 и выберите команду **Сервис	Служебные программы	Преобразовать базу данных	в формат Access 2000** (Tools | Database Utilities | Convert Database | To Access 2000 File Format)

Дополнительные действия, которые потребуется предпринять для преобразования форматов баз данных и проектов, защищенных и реплицированных баз данных, а также ограничения, связанные с преобразованием, подробно описаны в справочной системе Microsoft Access.

Сравнение преобразования и импорта файлов предыдущих версий Access

Есть еще одна возможность преобразования баз данных из предыдущих версий, кроме преобразования формата файла: импорт объектов баз данных. Эти два варианта имеют одно важное различие. Тот, кто импортирует объекты, становится их *владельцем* (owner) и, соответственно, имеет особые права доступа к ним. Преобразование же файла базы данных не изменяет владельцев хранящихся в нем объектов.

(О способах импорта см. гл. 3.)

Разрешение некорректных ссылок на библиотечные базы данных и надстройки Access

Если база данных Access 2000 или 2002—2003 содержит программный код, использующий объекты доступа к данным (DAO), то после ее преобразования в Access 97 и попытки ее использовать возникнет проблема с разрешением ссылок на библиотеки Access. Будет отображаться сообщение об отсутствии, по крайней мере, одной библиотеки объектов Access. Чтобы разрешить некорректные ссылки на библиотеки объектов Access 2000 или 2002—2003:

1. Запустите Access 97 и откройте в ней преобразованную базу данных. Когда появится сообщение об отсутствии библиотеки, на которую есть ссылка, нажмите кнопку **OK**.
2. Откройте любой модуль базы данных в режиме Конструктора.
3. Выберите команду **Сервис | Ссылки** (Tools | References). Появится диалоговое окно **Ссылки** (References). В списке **Доступные ссылки** (Available References) снимите флажки, соответствующие ссылкам на отсутствующие библиотеки.
4. Установите флажок с названием библиотеки DAO для Access 97 — **Microsoft DAO 3.51 Object Library** и нажмите кнопку **OK**.

Замечание

В отличие от Access 2000 и выше в приложении Access 97 невозможно использовать объекты управления данными ActiveX (ADO).

Если база данных, которую требуется преобразовать в формат Access 97, использует надстройки или библиотеки объектов, созданные в Access 2000 или выше, то их тоже нужно преобразовать в формат Access 97.

Ограничения, связанные с преобразованием баз данных Access 2000 и 2002—2003 в формат Access 97

База данных Microsoft Access 2000 или 2002—2003 может быть преобразована в формат Access 97 только в том случае, если она не является репликой (преобразование реплик выполняется путем синхронизации с преобразованной основной репликой). К тому же после преобразования в формат Access 97 приложение потеряет все функциональные возможности, относящиеся к специфичным возможностям Microsoft Access 2000 или выше.

Преобразование базы данных Access 2000 или выше в формат Access 97 приводит к следующим изменениям:

- теряются ссылки на страницы доступа к данным;
- данные, хранящиеся в формате Unicode, могут быть преобразованы неверно. Дело в том, что приложение Access 97 разработано так, что оно обеспечивает поддержку символов только одной кодовой страницы. (Кодовая страница представляет собой пронумерованное множество, состоящее из 256 символов.) Например, база данных Access 97 может использовать только ANSI-символы. В процессе преобразования базы данных результирующий набор символов для приложения Access 97 определяется параметрами сортировки, установленными в исходном

приложении Access 2000 или выше. Символы, принадлежащие другому набору символов, преобразуются некорректно;

- таблицы Access 2000 или выше могут содержать поля числового типа с атрибутом **Размер поля** (Field Size), имеющим значение **Десятичное/Действительное** (Decimal). В Access 97 такое значение атрибута **Размер поля** (Field Size) не определено. Чтобы успешно преобразовать базу данных, необходимо предварительно установить другое значение этого атрибута, например **С плавающей точкой (4 байт)/Одинарное с плавающей точкой** (Single) или **С плавающей точкой (8 байт)/Двойное с плавающей точкой** (Double), или изменить тип поля на **Денежный** (Currency).

ГЛАВА 20



Администрирование баз данных

Коллективное использование базы данных требует административного контроля. Необходимо поручить эти обязанности одному или нескольким сотрудникам, которые будут исполнять роль администраторов базы данных. Некоторые ошибочно полагают, что использование базы данных небольшим коллективом пользователей не требует специального обслуживания, и считают нецелесообразным выделение специальных средств на административную поддержку приложения, аргументируя это отсутствием свободных штатных сотрудников и нехваткой служебного времени. Однако если этот вопрос будет проигнорирован, рано или поздно такое положение дел приведет к нежелательным последствиям. Однажды база данных может быть испорчена и не окажется резервных копий. Невозможно будет решить некоторые часто возникающие задачи администрирования, требующие определенной квалификации, которой пользователи обычно не обладают. И при этом нет возможности немедленно связаться с разработчиками, или база данных была приобретена без заключения договора о сопровождении.

Материал этой главы адресован в первую очередь администраторам приложений Access. Целью администрирования является поддержка нормальной работы приложения в течение периода времени его использования. В данной главе перечислены основные задачи администрирования баз данных и описано их решение средствами Access. В частности, представлены процедуры управления системой защиты базы данных на уровне пользователей. Основные понятия и установка системы защиты базы данных на уровне пользователей рассмотрены ранее, в *гл. 16*. И конечно, все детали можно выяснить с помощью справочной системы Access.

Задачи администрирования баз данных

Среди задач администрирования приложений баз данных можно выделить следующие:

- создание резервных копий файлов (для баз данных и проектов Access);
- периодическое сжатие файлов (для баз данных);
- защита файлов средствами шифрования (для баз данных);

- изменение пароля для открытия файла (для баз данных);
- управление учетными записями и правами доступа для приложений, защищенных на уровне пользователей (для баз данных и проектов);

Замечание

Установка и управление параметрами системы защиты на уровне пользователей для баз данных выполняется средствами Access, а для проектов — средствами SQL Server.

- установка приложения, разделенного на файл объектов данных и файл объектов приложения, на новую рабочую станцию и обновление ссылок на связанные таблицы (для баз данных);
- установка клиентского приложения на новую рабочую станцию и корректное подключение ее к базе данных, установленной на сервере (для проектов);
- прочие задачи.

При использовании базы данных коллективом в десятки человек подобные задачи возникают довольно часто и их спектр достаточно широк, в этом случае рекомендуется разделить административные обязанности между несколькими специалистами. Для поддержки комплексных сетевых приложений для бизнеса вводят специальные должности.

Архивирование, сжатие и восстановление баз данных

Чтобы застраховаться от потери данных, создайте резервную копию базы данных или проекта Access. Чтобы увеличить производительность базы данных и уменьшить ее размер, используйте операцию сжатия базы данных или проекта Access. Процедуры сжатия и восстановления базы данных объединены в Microsoft Access в единый процесс (начиная с версии Access 2000).

Создание резервной копии базы данных или проекта Access

Есть несколько путей создания резервной копии базы данных или проекта. При наличии достаточного объема свободного места на диске можно создать резервную копию обычным копированием файла. Создать копию файла можно с помощью стандартного приложения Проводник (Explorer) операционной системы Windows, с помощью команды операционной системы `copy` или инструкции `FileCopy` в процедуре на VBA (только, если база данных или проект в это время не открыт в другом окне приложения Access). Чтобы сэкономить место на диске, можно создать сжатую копию файла с помощью программы архивирования, например, посредством стандартных средств архивирования, входящих в состав операционной системы Windows, или с помощью утилит сторонних производителей, таких как WinZip или WinRar. В случае повреждения файла исходной базы данных или проекта Access его можно будет заменить резервной копией.

Замечание

Если в базе данных установлена защита на уровне пользователей, необходимо создать также резервную копию соответствующего файла рабочей группы операцией копирования или архивирования соответствующего файла с расширением mdw.

В новой версии Access 2003 появилась возможность создавать резервную копию базы данных, не выходя из среды Access, посредством команды меню **Сервис | Служебные программы | Резервная копия базы данных** (Tools | Database Utilities | Back Up Database). Для резервного копирования проекта предусмотрена аналогичная команда **Сервис | Служебные программы | Резервная копия проекта** (Tools | Database Utilities | Back Up Project). *(См. разд. "Резервное копирование баз данных" гл. 7.)*

Для проектов Access (начиная с версии Access 2002) существует возможность создания резервной копии всех объектов проекта в специальном формате. С помощью команды **Сервис | Служебные программы | Создать резервную копию SQL-базы данных** (Tools | Database Utilities | Backup SQL Database) можно создать такую резервную копию всего проекта целиком, а затем, при необходимости, восстановить проект из этой копии с помощью команды **Сервис | Служебные программы | Восстановить SQL-базу данных** (Tools | Database Utilities | Restore SQL Database). Для сохранения данных на SQL Server или MSDE2000 из проекта Access в виде файла MDF, начиная с версии Access 2002, предусмотрена команда **Сервис | Служебные программы | Копировать файл базы данных** (Tools | Database Utilities | Copy Database File). Тогда же появилась команда меню Access **Сервис | Служебные программы | Удалить базу данных SQL** (Tools | Database Utilities | Drop SQL Database), позволяющая удалить источник данных проекта с SQL Server или MSDE2000. Это очень удобно, т. к. раньше для этих целей приходилось использовать утилиту Enterprise Manager для SQL Server или специальные инструкции на Transact-SQL. Все перечисленные команды работают только в том случае, если на компьютере установлен MSDE2000 или SQL Server.

Сжатие базы данных или проекта Access

При удалении данных или объектов файл базы данных или проекта Access становится фрагментированным, это приводит к тому, что дисковое пространство используется неэффективно. Сжатие базы данных или проекта позволяет получить копию, в которой данные и объекты сохраняются более рационально. Это экономит место на диске.

Сжатие повышает производительность баз данных и проектов Access. Однако сжатие проекта не влияет на объекты, такие как представления и таблицы, хранящиеся в базе данных на SQL Server. Сжатие проекта не влияет также на таблицы, содержащие поле счетчика в проектах Access, как это происходит с базами данных. Для баз данных, если перед сжатием из таблицы, содержащей поле счетчика, были удалены последние записи, то после сжатия номер первой пустой записи сбрасывается. Добавленная после этого запись получит в поле счетчика номер, на единицу превышающий значение счетчика в последней оставшейся записи.

Замечание

Сжатие баз данных, созданных в предыдущих версиях Microsoft Access, не приводит к их преобразованию в формат Microsoft Access 2002—2003. Способы преобразования баз данных предыдущих версий Access описаны в *гл. 19*.

Перед сжатием совместно используемой базы данных убедитесь, что она не открыта ни одним из пользователей в сети. Для сжатия базы данных необходимо обладать правами на ее открытие, запуск и открытие в монопольном режиме. *(См. разд. "Назначение прав доступа к объектам базы данных" в этой главе.)*

Чтобы сжать открытую базу данных или проект Access:

1. Если нужно сжать сетевую базу данных, расположенную в общей папке или на сервере, убедитесь, что она не открыта другими пользователями.
2. Выберите команду **Сервис | Служебные программы | Сжать и восстановить базу данных (проект)** (Tools | Database Utilities | Compact and Repair Database (Project)).

Чтобы сжать закрытую в данный момент базу данных или проект Access:

1. Если в данный момент открыта другая база данных или проект Access, закройте ее (его). Если база данных (или проект) расположена в общей папке или на сервере, убедитесь, что она не открыта другими пользователями.
2. Выберите команду **Сервис | Служебные программы | Сжать и восстановить базу данных** (Tools | Database Utilities | Compact and Repair Database).
3. Появится диалоговое окно **База данных для сжатия** (Database To Compact From) (рис. 20.1), похожее на окно открытия базы данных. Укажите базу данных или проект, который необходимо сжать, и нажмите кнопку **Сжать** (Compact).
4. Появится диалоговое окно **Сжатие базы данных под именем** (Compact Database Into). Выберите диск и папку и введите имя для сохранения сжатой базы данных. Нажмите кнопку **Сохранить** (Save).

Замечание

В обоих случаях процесс сжатия можно прервать с помощью комбинации клавиш <Ctrl>+<Break> или клавиши <Esc>.

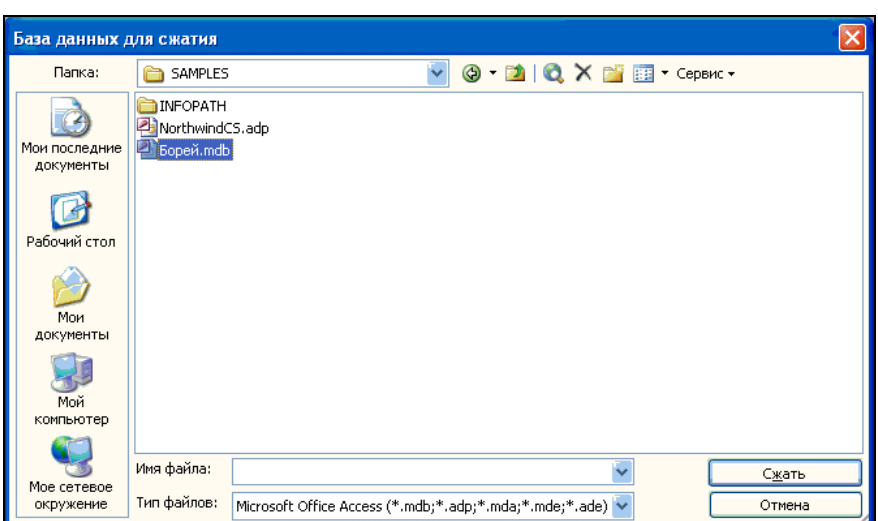


Рис. 20.1. Диалоговое окно **База данных для сжатия**

Допускается сжатие файла базы данных или проекта Access в файл с тем же именем, что и имя исходного файла, или создание файла с новым именем. При указании того же имени, диска и папки и при успешном сжатии базы данных исходный файл автоматически заменяется на сжатый.

Можно настроить Access так, чтобы конкретная база данных или проект автоматически сжимались при закрытии.

Чтобы установить автоматическое сжатие базы данных Access:

1. Откройте базу данных, которую нужно сжать.
2. Выберите команду **Сервис | Параметры** (Tools | Options). В появившемся диалоговом окне **Параметры** (Options) раскройте вкладку **Общие** (General).
3. Установите флажок **Сжимать при закрытии** (Compact on Close) и нажмите кнопку **OK**.

Установка автоматического сжатия проекта Access выполняется аналогично.

Замечание

Автоматическое сжатие не происходит, если при этом размер базы данных (или проекта) не будет уменьшен, по крайней мере, на 256 Кбайт, а также, если эта база данных (или проект) в текущий момент открыта другим пользователем в сети.

Восстановление поврежденной базы данных

В большинстве случаев Microsoft Access определяет, что база данных повреждена, при попытке открыть, зашифровать или дешифровать ее. Тогда пользователю предоставляется возможность восстановить базу данных, выполнив ее сжатие. Но в некоторых ситуациях не удается определить, что база данных повреждена. Если база данных ведет себя непредсказуемым образом, выполните ее сжатие.

При серьезных проблемах, которые приводят к вынужденному завершению работы Access, это приложение перезапускается и автоматически создается резервная копия открытой базы данных или проекта Access с тем же именем и расширением, что и исходный файл, только с суффиксом `_Backup` (резервный).

Защита информации с помощью шифрования

Защита информации в базе данных Access производится с помощью операции шифрования, которая также приводит к сжатию файла базы данных. После выполнения шифрования просмотр данных с помощью специальных программ и текстовых редакторов становится невозможен. Операция дешифрации отменяет результаты операции шифрования. Для проектов Access операция шифрования не реализована.

Если база данных используется в сети, то, прежде чем приступить к выполнению операции шифрования или дешифрации, убедитесь, что она не открыта ни одним из пользователей.

Чтобы выполнить операцию шифрования или дешифрации:

1. Запустите Access, не открывая базу данных.
2. Выберите команду **Сервис | Защита | Закодировать или раскодировать базу данных** (Tools | Security | Encode/Decode Database).
3. Появится диалоговое окно **База данных для кодирования или декодирования** (Encode/Decode Database). Укажите имя базы данных, которую требуется зашифровать или дешифровать, и нажмите кнопку **OK**.
4. Если выбранная на предыдущем шаге база данных не является зашифрованной, появится диалоговое окно **Кодирование базы данных под именем** (Encode Database As), иначе появится диалоговое окно **Декодирование базы данных под именем** (Decode Database As). Укажите имя, диск и папку для конечной базы данных и нажмите кнопку **Сохранить** (Save).

Замечание

Можно указать новое имя или имя, совпадающее с именем исходного файла. Если указаны те же имя, диск и папка, а шифрование или дешифрация выполнены успешно, то исходный файл будет заменен на зашифрованный или дешифрованный. Если же операция приводит к ошибке, то исходный файл удален не будет. Кроме того, исходный файл не удаляется, пока операция шифрования не будет завершена. Поэтому необходимо иметь достаточно места на диске для двух версий базы данных — зашифрованной и незашифрованной. Рекомендуется перед операцией шифрования делать резервную копию базы данных.

Если для базы данных определена защита на уровне пользователей, то для ее шифрования или дешифрации необходимо быть *владельцем* базы данных или входить в группу `Admins` и иметь разрешение на монопольное открытие базы данных. В противном случае попытка шифрования или дешифрации окажется неудачной. Шифрование базы данных не препятствует доступу к ней легальных пользователей или групп в Access. Разрешения на доступ к объектам являются частью системы защиты базы данных на уровне пользователей.

Администрирование защищенных баз данных и проектов Access

Существует несколько способов защиты базы данных Access от несанкционированного доступа.

- База данных может быть защищена паролем.
- Для базы данных Access может быть установлена система защиты на уровне пользователей.
- Программный код на VBA может быть защищен паролем.

В отличие от баз данных, проект (файл ADP) не может быть защищен с помощью пароля или посредством установки защиты на уровне пользователей средствами Access. Однако, как и в базах данных, в проектах можно защищать паролем программный код на VBA. На практике защита проектов сводится к помещению файла проекта в общую папку на файловом сервере, к которой пользователям сети предос-

тавляется доступ только для чтения. Файл проекта может быть отправлен пользователям по почте. Пользователи должны защитить свою локальную копию файла проекта, используя средства защиты файловой системы на своем компьютере, и периодически создавать резервные копии файла проекта при добавлении в него новых форм или отчетов, чтобы в случае повреждения файла проекта иметь возможность его восстановить.

Кроме того, в проектах используется система защиты баз данных, хранящихся на SQL Server. Об этом рассказано в следующем разделе. Защита программного кода на VBA базы данных или проекта Access описана в *гл. 16*.

Есть еще способы защиты объектов в базе данных или проекте Access — преобразовать файл в формате MDE или ADE или использовать параметры запуска для ограничения доступа к программам VBA и некоторым параметрам среды Access, можно также скрыть некоторые объекты от пользователей с помощью диалогового окна **Параметры** (Options). *(См. разд. "Защита программного кода на VBA" гл. 16 и разд. "Настройка параметров запуска" гл. 14.)*

Администрирование баз данных, защищенных с помощью пароля, сводится к изменению пароля защиты (когда это необходимо). Задача администрирования приложений, для которых установлена защита на уровне пользователей, существенно сложнее. *(См. разд. "Защита базы данных на уровне пользователей" гл. 16.)*

Защита баз данных на SQL Server

SQL Server имеет собственную встроенную систему защиты баз данных на уровне пользователей. Подключиться к SQL Server и базе данных на нем может только зарегистрированный на сервере пользователь, обладающий соответствующими правами. Подробную информацию об использовании системы защиты для проектов Access можно найти в документации SQL Server.

Защита объектов баз данных на SQL Server

В проектах Access существует две возможности защиты данных и других объектов (таблиц, представлений, хранимых процедур и схем данных), хранящихся на SQL Server.

- Можно выполнить шифрование представлений, хранимых процедур и триггеров. После шифрования структуру представления невозможно изменить. Хотя представление по-прежнему можно будет использовать для правки данных в источнике. Информацию о том, как зашифровать представление, можно найти в документации к SQL Server. Шифрование хранимой процедуры или триггера можно выполнить, отредактировав их и записав инструкцию Transact-SQL специального вида, например: `CREATE PROCEDURE ... WITH ENCRYPTION`.
- Можно использовать полный набор средств защиты, предоставляемых SQL Server, с помощью программы SQL Server Enterprise Manager. Описание этих средств защиты можно найти в документации SQL Server.

Если есть вероятность того, что структуру зашифрованного представления понадобится изменить в будущем, воспользуйтесь следующим советом. Сохраните инструкцию SQL, определяющую представление, в текстовом файле. Сохраните этот

файл в надежном месте. Зашифруйте представление. При необходимости изменить структуру зашифрованного представления:

1. Удалите прежнее зашифрованное представление.
2. Создайте новое представление с таким же именем, как у прежнего представления.
3. Скопируйте в буфер обмена инструкцию SQL из текстового файла, в котором она была сохранена. Вставьте ее в область ввода инструкции SQL в окне Конструктора нового представления.
4. Измените структуру представления.
5. Сохраните измененную инструкцию SQL в текстовом файле. Поместите этот файл в надежном месте.
6. Зашифруйте новое представление.

Управление учетными записями на SQL Server

Работая с проектом Access, можно выполнять основные административные задачи обеспечения безопасности базы данных на SQL Server с помощью команды Access **Безопасность базы данных** (Database Security) в меню **Сервис** (Tools). Эта команда доступна только в том случае, если SQL Server установлен на том же компьютере, где хранится проект Access. С помощью этого средства можно добавлять, изменять и редактировать учетные записи для регистрации на SQL Server, учетные записи пользователей базы данных и роли.

Замечание

Если установлен не SQL Server, а MSDE2000, команды **Безопасность базы данных** (Database Security) и **Репликация** (Replication) недоступны. Используйте инструкции Transact-SQL или Enterprise Manager для защиты и репликации данных на SQL Server. Подробную информацию можно найти в справочной системе SQL Server.

Существует две системы безопасности, применяемые для регистрации на SQL Server.

- Встроенная система защиты SQL Server. Для регистрации на сервере требуется указать имя пользователя сервера и пароль.
- Интегрированная система защиты с Windows NT, использующая учетные записи пользователей NT. В этом случае аутентификация пользователя производится в момент регистрации в сети на базе Windows NT. SQL Server принимает учетную запись Windows NT и не требует дополнительной регистрации пользователя.

Чтобы использовать встроенную систему защиты SQL Server:

1. По умолчанию эта возможность выключена, поэтому сначала выберите команду меню **Вид | Свойства сервера** (View | Server Properties) и в появившемся диалоговом окне (рис. 20.2) установите флажок **Учетная запись системного администратора (SA)** (Enable system administrator (SA) user name). Нажмите кнопку **OK**.
2. Откройте окно свойств соединения с SQL Server **Свойства подключения** (Data Link Properties) с помощью команды меню **Файл | Подключение** (File | Connection).

3. На вкладке **Соединение** (Connection) установите флажок **Использовать имя пользователя и пароль** (Use a specific user name and password).
4. Если при открытии проекта Access нужно каждый раз требовать ввод пароля регистрации пользователя, сбросьте флажок **Сохранять пароль** (Allow saving of password). Если ввод пароля при повторной регистрации на сервере не требуется, установите этот флажок.

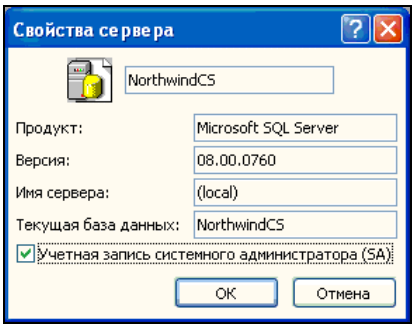


Рис. 20.2. Диалоговое окно **Свойства сервера**

С такой системой защиты вы сможете поменять пароль учетной записи SQL Server, использующейся для регистрации в проекте Access, с помощью команды меню Access **Сервис | Защита | Задать пароль для входа** (Tools | Security | Set Login Password). Не забудьте установленный пароль, чтобы не потерять доступ к проекту!

Чтобы использовать интегрированную с Windows NT систему защиты SQL Server:

1. Откройте окно свойств соединения с SQL Server **Свойства подключения** (Data Link Properties).
2. На вкладке **Соединение** (Connection) выберите переключатель **Использовать интегрированную систему защиты с Windows NT** (Use Windows NT Integrated Security).

При этом ввод пароля при регистрации на SQL Server требоваться не будет, поскольку будет действовать идентификация пользователя в сети Windows NT, Windows 2000 или Windows XP.

Защита базы данных Access с помощью пароля

Самый простой способ защиты базы данных — с помощью пароля. Можно назначить пароль базе данных Access, который будет требоваться всякий раз при ее открытии.

Установка и снятие пароля защиты базы данных

Чтобы установить пароль для защиты базы данных:

1. Закройте базу данных. Если база данных совместно используется в сети, убедитесь, что остальные пользователи ее закрыли.

2. Сделайте резервную копию базы данных и сохраните ее в надежном месте.
3. В меню Access выберите команду **Файл | Открыть** (File | Open).
4. Выделите файл базы данных.
5. Щелкните по стрелке, расположенной справа от кнопки **Открыть** (Open). В раскрывающемся списке режимов открытия базы данных выделите элемент **Монопольно** (Open Exclusive). База данных откроется в *режиме монопольного доступа.*
6. Выберите команду **Сервис | Защита | Задать пароль базы данных** (Tools | Security | Set Database Password).
7. В появившемся диалоговом окне в поле **Пароль** (Password) введите пароль для защиты базы данных с учетом регистра символов.
8. Введите пароль еще раз в поле **Подтверждение** (Verify).
9. Нажмите кнопку **OK**.

Теперь база данных защищена паролем и всякий раз, когда пользователь будет открывать базу данных, будет отображаться диалоговое окно с требованием ввести пароль. Запомните или сохраните пароль в надежном месте. Если пароль будет забыт, невозможно будет открыть базу данных.

Замечание

Не защищайте базу данных паролем перед ее репликацией, иначе ее нельзя будет синхронизировать с другими репликами.

Если база данных защищена на уровне пользователей, установить пароль для ее открытия может только пользователь, обладающий административными правами. Установка пароля не влияет на систему защиты на уровне пользователя. Эти два способа защиты могут использоваться одновременно. Пароль базы данных сохраняется в базе данных, а не в файле рабочей группы.

Чтобы удалить пароль защиты базы данных:

1. Откройте базу данных в режиме монопольного доступа.
2. В диалоговое окно **Необходимо ввести пароль** (Password Required) введите пароль.
3. Выберите команду **Сервис | Защита | Удалить пароль базы данных** (Tools | Security | Unset Database Password). Появится диалоговое окно **Удаление пароля базы данных** (Unset Database Password).
4. Введите текущий пароль базы данных.
5. Нажмите кнопку **OK**.

Установка связи с таблицами базы данных, защищенной паролем

Чтобы установить связь с таблицами базы данных, защищенной паролем, требуется ввести пароль. Если пароль был указан верно, он сохраняется вместе с другой информацией о ссылках на таблицы. После этого любой пользователь, работающий с базой данных со связанными таблицами, может открыть эти таблицы без указания пароля. Если пароль защищенной базы данных будет изменен, в следующий раз при открытии базы данных, содержащей связанные таблицы, потребуется ввести пароль.

Замечание

Microsoft Access сохраняет пароль в базе данных, содержащей связанные таблицы защищенной базы данных, в незашифрованном виде. Если это уязвляет систему защиты базы данных, не используйте средство защиты с помощью пароля. Установите систему защиты на уровне пользователей, чтобы ограничить доступ к объектам базы данных.

Администрирование баз данных Access, защищенных на уровне пользователей

В задачу администрирования базы данных, защищенной на уровне пользователей, входит выполнение следующих действий:

- подключение файла рабочей группы для работы с защищенной базой данных;
- добавление нового пользователя в рабочую группу и задание для него прав доступа к объектам и прав на владение объектами базы данных (это осуществляется обычно включением его в состав определенных групп внутри рабочей группы);
- добавление новой группы пользователей в рабочую группу и задание для нее прав доступа к объектам базы данных;
- удаление учетной записи пользователя из файла рабочей группы;
- удаление учетной записи группы из файла рабочей группы;
- изменение пароля пользователя;
- восстановление испорченного файла рабочей группы из резервной копии или путем его воссоздания, используя специально сохраненную информацию об учетных записях пользователей и рабочих групп (включающую имена пользователей и групп и их персональные идентификаторы (PID)).

Внимание!

Задачи администрирования, касающиеся защиты приложения на уровне пользователей, может выполнять только пользователь, обладающий административными правами. В Access административными правами автоматически наделяются пользователи встроенной группы `Admins`. Остальным пользователям разрешено только изменять пароль своей учетной записи и получать отчет со списком пользователей и групп рабочей группы.

О том, что такое файл рабочей группы и о задачах администрирования, связанных с его использованием, рассказано в следующем разделе. О том, как изменить состав пользователей и групп пользователей приложения Access, *см. разд. "Управление учетными записями"*. О назначении прав доступа к объектам базы данных или проекта Access *см. разд. "Назначение прав доступа к объектам базы данных"*.

Использование файла рабочей группы

Рабочей группой в Access называется группа пользователей сети, совместно использующих одну или несколько баз данных Access. Если база данных защищена на уровне пользователей, в файл рабочей группы записываются учетные записи пользо-

вателей и групп, входящих в рабочую группу. Пароли пользователей также хранятся в файле рабочей группы. Учетным записям в рабочей группе могут быть назначены права доступа к базе данных и ее объектам (таблицам, запросам, формам, отчетам и макросам). Права доступа сохраняются в защищенной базе данных.

Создание и подключение файла рабочей группы производится с помощью служебной программы, которая входит в состав Microsoft Access и называется Администратор рабочих групп (Workgroup Administrator). Восстановление файла рабочей группы производится путем его воссоздания по сохраненной информации.

Администратор рабочих групп

Чтобы запустить служебную программу Администратор рабочих групп (Workgroup Administrator):

1. Выберите команду **Сервис | Защита | Администратор рабочих групп** (Tools | Security | Workgroup Administrator).
2. Появится диалоговое окно **Администратор рабочих групп** (Workgroup Administrator), представленное на рис. 20.3.

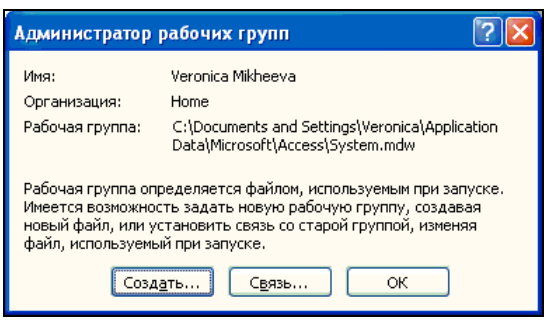


Рис. 20.3. Диалоговое окно **Администратор рабочих групп**

Подключение файла рабочей группы к приложению Access

Чтобы использовать базу данных, защищенную на уровне пользователей, присоедините к Access ее файл рабочей группы с помощью служебной программы **Администратор рабочих групп** (Workgroup Administrator):

1. Запустите Администратор рабочих групп (Workgroup Administrator).
2. Появится диалоговое окно **Администратор рабочих групп** (Workgroup Administrator), представленное на рис. 20.3.
3. В поле **Рабочая группа** (Workgroup) отображается имя текущего присоединенного файла рабочей группы Access. Чтобы присоединить другой файл рабочей группы, нажмите кнопку **Связь** (Join).
4. Появится диалоговое окно **Файл рабочей группы** (Workgroup Information File). Введите в поле **Рабочая группа** (Workgroup) полное имя файла рабочих групп или нажмите кнопку **Обзор** (Browse), чтобы выбрать нужный файл. Нажмите кнопку **Открыть** (Open), а затем кнопку **OK** в окне **Файл рабочей группы**.

5. Нажмите кнопку **OK** в окне с сообщением об успешном подключении файла.
6. Нажмите кнопку **OK**, чтобы закрыть диалоговое окно **Администратор рабочих групп** (Workgroup Administrator).

В результате выполнения этих действий путь к файлу рабочей группы записывается в реестр как параметр Access. Это означает, что для всех баз данных, открываемых в Access, будет использована защита на уровне пользователей на основе присоединенного файла рабочей группы. По умолчанию к Access присоединен файл рабочей группы с именем System.mdw, хранящийся в папке, в которую установлен пакет Microsoft Office, или в папке с личными настройками пользователя Windows. Прежде чем проводить эксперименты с файлом рабочих групп Access, рекомендуется сделать его резервную копию.

Если нужно использовать файл рабочей группы для конкретной базы данных, создайте ярлык для открытия этой базы данных, в командной строке которого укажите параметр `/wrkgrp` и полное имя файла рабочей группы:

```
"<Путь>\MSACCESS.EXE" "<Путь>\<Имя базы данных или проекта Access>" /wrkgrp
"<Путь>\<Имя файла рабочей группы>"
```

Например, чтобы использовать базу данных сервера "Игра в доминирование" (это приложение описано в *гл. 16*), которая находится в общей папке Domination на компьютере GameServer в локальной сети, создайте ярлык с командной строкой:

```
"C:\Program Files\Microsoft Office\Office11\MSACCESS.EXE"
"\\GameServer\Domination\DominationGameServer.mdb" /wrkgrp
"\\GameServer\Domination\Domination.mdw"
```

А чтобы использовать базу данных клиента приложения "Игра в доминирование", создайте ярлык, содержащий команду:

```
"C:\Program Files\Microsoft Office\Office11\MSACCESS.EXE"
"\\GameServer\Domination\DominationGamePlayer.mdb" /wrkgrp
"\\GameServer\Domination\Domination.mdw"
```

Создание и восстановление файла рабочей группы

Чтобы создать собственный файл рабочей группы (такой как Domination.mdw):

1. Запустите программу Администратор рабочих групп (Workgroup Administrator).
2. Появится диалоговое окно **Администратор рабочих групп** (Workgroup Administrator), представленное на рис. 20.3. Нажмите кнопку **Создать** (Create).
3. Появится диалоговое окно **Сведения о владельце рабочей группы** (Workgroup Owner Information) (рис. 20.4). В поля **Имя** (Name) и **Организация** (Organization) введите название рабочей группы и организации. Например, для файла рабочей группы Domination.mdw приложения "Игра в доминирование" именем рабочей группы является `Морские волки`, а названием организации — `ООО "Остров сокровищ"`.
4. В поле **Код группы** (Workgroup ID) введите уникальный идентификатор группы, состоящий из букв и цифр. Этот идентификатор может включать до 20 символов. Например, идентификатором рабочей группы для приложения "Игра в домини-

рование" является фраза `Игра в доминирование`. Однако в качестве идентификатора защиты рекомендуется использовать менее осмысленный набор символов, который сложнее угадать, чтобы обеспечить большую степень защищенности рабочей группы.

Замечание

Внимательно заполняйте поля с информацией о файле рабочей группы. Все три значения — имя, организация и код группы — интерпретируются с учетом регистра символов. Сохраните эту информацию в надежном месте, чтобы в случае повреждения файла рабочей группы его можно было бы восстановить. Если эта информация будет утрачена, при повреждении файла рабочей группы его невозможно будет восстановить, а это может привести к безвозвратной потере доступа к защищенным объектам базы данных.

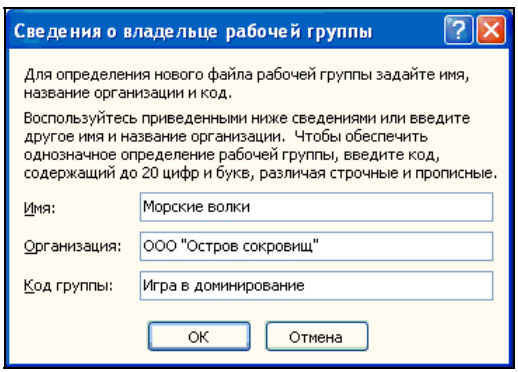


Рис. 20.4. Диалоговое окно **Сведения о владельце рабочей группы**

5. Нажмите кнопку **OK**. Появится диалоговое окно, позволяющее задать имя нового файла рабочей группы. Введите полное имя файла с расширением mdw, например C:\Games\Domination\Domination.mdw, или нажмите кнопку **Обзор** (Browse), чтобы выбрать папку, в которой будет сохранен новый файл, и укажите имя файла. Нажмите кнопку **OK**.
6. Появится диалоговое окно, позволяющее проверить введенную информацию. Нажмите кнопку **OK**.
7. Нажмите кнопку **OK** во всех остальных диалоговых окнах Администратора рабочих групп (Worgroup Administrator).

Созданный файл рабочей группы автоматически присоединяется к Access. Присоединенный файл рабочей группы используется при открытии любых баз данных в Access. С этого момента все создаваемые учетные записи будут сохраняться в этом файле рабочей группы. Чтобы организовать совместное использование файла рабочей группы в сети несколькими пользователями, поместите этот файл в общую папку на файл-сервере и попросите пользователей присоединить его к их копии Access с помощью программы Администратор рабочих групп (Worgroup Administrator), как описано в *разд. "Подключение файла рабочей группы к приложению Access"*.

Управление учетными записями

Управление учетными записями включает следующие действия:

- добавление или удаление учетной записи пользователя;
- добавление или удаление учетной записи пользователя-администратора;
- добавление или удаление учетной записи группы пользователей;
- добавление пользователя в группу или удаление пользователя из группы;
- изменение пароля учетной записи пользователя.

Все перечисленные действия можно выполнить в Access с помощью диалогового окна **Пользователи и группы** (User and Group Accounts). Чтобы открыть диалоговое окно для управления учетными записями:

1. Удостоверьтесь, что нужный файл рабочей группы присоединен к Access или базе данных. *(См. разд. "Подключение файла рабочей группы к приложению Access" в этой главе.)*
2. Откройте базу данных и зарегистрируйтесь с помощью учетной записи, обладающей административными правами (это может быть, например, встроенная учетная запись `Admin`).
3. Выберите команду **Сервис | Защита | Пользователи и группы** (Tools | Security | User and Group Accounts). Появится диалоговое окно **Пользователи и группы** (User and Group Accounts) (рис. 20.5).

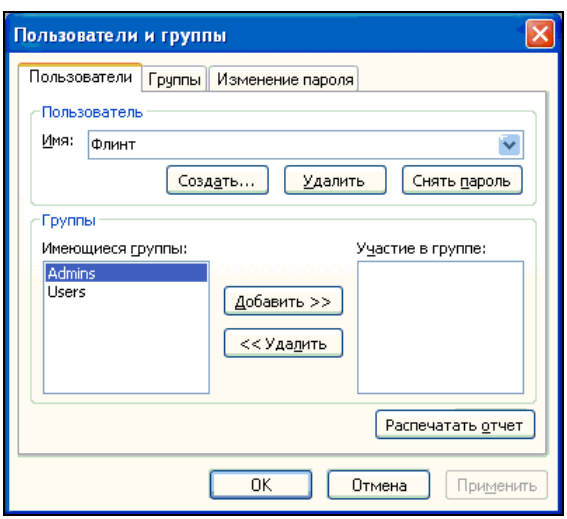


Рис. 20.5. Диалоговое окно **Пользователи и группы**

Рассмотрим процедуры выполнения действий с учетными записями с помощью диалогового окна, представленного на рис. 20.5. Чтобы эти действия повлияли на систему защиты базы данных, после их выполнения необходимо нажать кнопку **Применить** (Apply) или **OK** в диалоговом окне **Пользователи и группы** (User and Group Accounts).

Чтобы добавить учетную запись пользователя:

1. На вкладке **Пользователи** (Users) нажмите кнопку **Создать** (New).
2. Появится диалоговое окно **Новый пользователь или группа** (New User/Group) (рис. 20.6).
3. Введите в поле **Имя** (Name) имя пользователя, а в поле **Код** (Personal ID) — идентификатор пользователя. Нажмите кнопку **OK**.

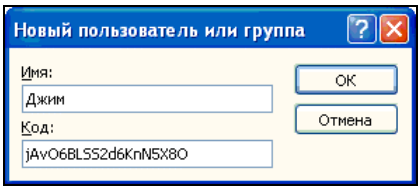


Рис. 20.6. Диалоговое окно **Новый пользователь или группа**

Имя пользователя может включать от одного до двадцати символов. В имени можно использовать буквы, цифры, символы акцента, пробелы и любые другие символы, кроме следующих " \ [ ] : | < > + =; , . ? * . Нельзя также использовать ведущие пробелы и управляющие символы (т. е. символы, ASCII-код которых находится в пределах от 10 до 31).

Необходимо ввести имя пользователя с учетом регистра и запомнить или записать его, поскольку это информация нужна для успешной регистрации в базе данных и для восстановления файла рабочей группы в случае его повреждения.

Идентификатор пользователя представляет собой последовательность от 4 до 20 символов. Идентификатор пользователя не является паролем пользователя. Вместе с именем пользователя он используется в алгоритме шифрования для вычисления идентификатора защиты учетной записи. Идентификатор пользователя, так же как идентификатор группы, есть способ идентификации учетной записи в системе защиты приложения. Если файл рабочей группы будет поврежден, учетные записи в нем можно будет воссоздать, если известны их идентификаторы.

Чтобы удалить учетную запись пользователя:

1. На вкладке **Пользователи** (Users) в раскрывающемся списке **Имя** (Name) выберите имя пользователя, соответствующее удаляемой учетной записи.
2. Нажмите кнопку **Удалить** (Delete). Появится диалоговое окно, требующее подтверждения удаления учетной записи. Нажмите кнопку **Да** (Yes).

Чтобы добавить пользователя в группу:

1. На вкладке **Пользователи** (Users) в раскрывающемся списке **Имя** (Name) выберите имя пользователя, которого нужно добавить в некоторую группу. В списке **Участие в группе** (Member Of) отобразится список групп, в которые включена учетная запись пользователя.
2. В списке **Имеющиеся группы** (Available Groups) отображаются все группы, имеющиеся в файле рабочей группы. Выделите в этом списке группу, в которую нужно добавить пользователя.

3. Нажмите кнопку **Добавить** (Add) (эта кнопка отмечена стрелками вправо). Выделенное имя группы появится в списке **Участие в группе** (Member Of).

Чтобы удалить пользователя из группы:

1. На вкладке **Пользователи** (Users) в раскрывающемся списке **Имя** (Name) выберите имя пользователя, которого нужно удалить из некоторой группы.
2. В списке **Участие в группе** (Member Of) отобразится список групп, в которые включена учетная запись пользователя. Выделите в этом списке группу, из которой нужно удалить пользователя.
3. Нажмите кнопку **Удалить** (Remove) (эта кнопка отмечена стрелками влево). Выделенное имя группы будет удалено из списка **Участие в группе** (Member Of).

Чтобы добавить учетную запись пользователя-администратора:

1. Создайте новую учетную запись пользователя.
2. Добавьте пользователя в группу `Admins`.

Чтобы удалить учетную запись пользователя-администратора:

1. Выделите учетную запись пользователя-администратора.
2. Удалите пользователя из группы `Admins` или удалите учетную запись пользователя.

Чтобы добавить учетную запись группы:

1. Раскройте вкладку **Группы** (Groups) и нажмите кнопку **Создать** (New).
2. Появится диалоговое окно **Новый пользователь или группа** (New User/Group) (см. рис. 20.6).
3. Введите в поле **Имя** (Name) имя группы, а в поле **Код** (Personal ID) — идентификатор группы. Нажмите кнопку **ОК**.

Чтобы удалить учетную запись группы:

1. На вкладке **Группы** (Groups) в раскрывающемся списке **Имя** (Name) выберите имя группы, которую нужно удалить.
2. Нажмите кнопку **Удалить** (Delete). Появится диалоговое окно, требующее подтверждения удаления учетной записи. Нажмите кнопку **Да** (Yes).

Замечание

Невозможно удалить системные группы `Admins` и `Users`, а также системную учетную запись пользователя `Admin`, но можно удалить системную учетную запись пользователя `Admin` из группы `Admins`. Однако в группе `Admins` обязательно должен быть хотя бы один пользователь и никакую учетную запись пользователя нельзя удалить из группы `Users`.

Чтобы задать или изменить пароль пользователя:

1. Откройте базу данных и зарегистрируйтесь с именем пользователя, пароль которого нужно изменить.
2. Выберите команду **Сервис | Защита | Пользователи и группы** (Tools | Security | User and Group Accounts). Появится диалоговое окно **Пользователи и группы** (User and Group Accounts) (см. рис. 20.5).

3. Раскройте вкладку **Изменение пароля** (Change Logon Password).
4. В поле **Пользователь** (User) отображается имя пользователя, которое было использовано при регистрации. В поле **Текущий пароль** (Old Password) введите текущий пароль пользователя. Если пароль пользователя не был задан, оставьте это поле пустым.
5. В поля **Новый пароль** (New Password) и **Подтверждение** (Verify) введите новый пароль пользователя. Нажмите кнопку **OK**.

Чтобы заменить пароль пользователя пустым паролем:

1. Откройте диалоговое окно **Пользователи и группы** (User and Group Accounts).
2. На вкладке **Пользователи** (Users) в раскрывающемся списке **Имя** (Name) выберите имя пользователя, пароль которого нужно удалить.
3. Нажмите кнопку **Снять пароль** (Clear password).

Чтобы получить отчет со списком пользователей и групп рабочей группы:

1. Откройте диалоговое окно **Пользователи и группы** (User and Group Accounts).
2. На вкладке **Пользователи** (Users) нажмите кнопку **Распечатать отчет** (Print Users and Groups).

Назначение прав доступа к объектам базы данных

Пользователям и группам пользователей рабочей группы можно назначить различные права для доступа к разным объектам базы данных, защищенной на уровне пользователей. В табл. 20.1 представлены возможные права доступа к объектам базы данных. В столбце "Дополнительные права" приведены права доступа, которые необходимы для получения права, указанного в столбце "Права доступа". В результате предоставления пользователю или группе определенных разрешений, дополнительные права предоставляются автоматически.

Таблица 20.1. *Права доступа, применимые к объектам в Access*

Права доступа	Разрешенные действия	Объекты	Дополнительные права
открытие/запуск (Open/Run)	Открытие базы данных, формы или отчета, запуск макроса	Базы данных, формы, отчеты и макросы	
монопольный доступ (Open Exclusive)	Открытие базы данных для монопольного доступа	Базы данных	
чтение макета (Read Design)	Просмотр объектов в режиме Конструктора	Таблицы, запросы, формы, отчеты и макросы	
изменение макета (Modify Design)	Просмотр и изменение макета объектов или удаление объектов	Таблицы, запросы, формы, отчеты и макросы	**чтение макета**, **чтение данных**, **обновление данных**, **удаление данных**

Таблица 20.1 *(окончание)*

Права доступа	Разрешенные действия	Объекты	Дополнительные права
администратора (Administer)	Для баз данных — установка пароля базы данных, репликация базы данных и изменение параметров запуска. Для объектов базы данных — все разрешения на объекты и данные, в том числе предоставление разрешений на доступ	Базы данных, таблицы, запросы, формы, отчеты и макросы	Все остальные права
чтение данных (Read Data)	Просмотр данных	Таблицы и запросы	**чтение макета**
обновление данных (Update Data)	Просмотр и изменение данных, но без их вставки или удаления	Таблицы и запросы	**чтение макета**, **чтение данных**
вставка данных (Insert Data)	Просмотр и вставка данных, но без их изменения и удаления	Таблицы и запросы	**чтение макета**, **чтение данных**
удаление данных (Delete Data)	Просмотр и удаление данных, но без их изменения и вставки	Таблицы и запросы	**чтение макета**, **чтение данных**

Защита модулей в Access 2003 (начиная с версии Access 2000) устанавливается отдельно. Установка защиты программного проекта VBA описана в *гл. 16.*

Изменение прав доступа к объектам базы данных

Назначение прав доступа к объектам базы данных для учетных записей, хранящихся в файле рабочей группы, выполняется в Access с помощью диалогового окна **Разрешения** (User and Group Permissions).

Чтобы открыть диалоговое окно для назначения прав доступа к объектам базы данных:

1. Откройте защищенную базу данных, подключив необходимый файл рабочей группы.
2. Зарегистрируйтесь с именем пользователя, обладающего административными правами.
3. Выберите команду **Сервис | Защита | Разрешения** (Tools | Security | User and Group Permissions). Появится диалоговое окно **Разрешения** (User and Group Permissions) (рис. 20.7).

В диалоговом окне **Разрешения** есть две вкладки: **Разрешения** (Permissions) и **Смена владельца** (Change Owner).

(О функциях второй вкладки рассказано в разд. "Предоставление права на владение объектами базы данных" этой главы.)

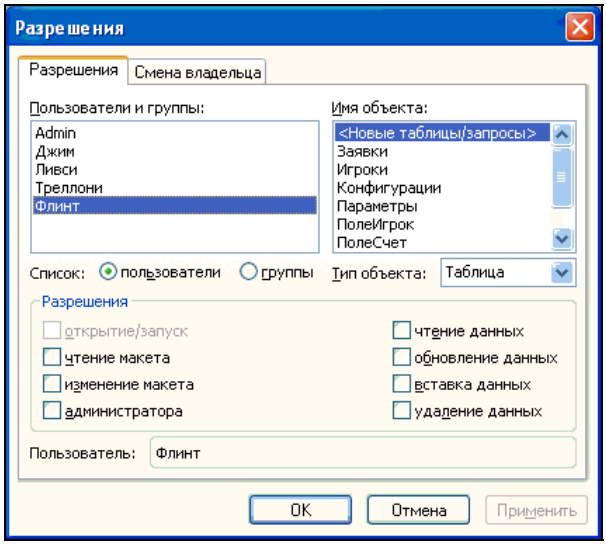


Рис. 20.7. Диалоговое окно **Разрешения**

Рассмотрим вкладку **Разрешения**. С помощью этой вкладки можно определить права доступа к конкретным объектам базы данных для конкретных пользователей и групп. В поле **Пользователь** (Current User) отображается имя пользователя, которое было использовано для регистрации в момент открытия базы данных. В зависимости от того, обладает ли текущий пользователь административными правами, ему будет позволен или запрещен просмотр и изменение прав доступа к объектам базы данных.

Чтобы назначить права доступа к объектам базы данных конкретной группе:

1. На вкладке **Разрешения** выберите переключатель **группы** (Groups).
2. В списке **Пользователи и группы** (User/Group Name) отобразится список всех групп в рабочей группе. Выделите в этом списке группу, права доступа которой нужно изменить.
3. Измените права доступа к объектам базы данных и нажмите кнопку **OK**.

Чтобы назначить права доступа к объектам базы данных конкретному пользователю:

1. На вкладке **Разрешения** выберите переключатель **пользователи** (Users).
2. В списке **Пользователи и группы** (User/Group Name) отобразится список всех пользователей в рабочей группе. Выделите в этом списке пользователя, права доступа которого нужно изменить.
3. Измените права доступа к объектам базы данных и нажмите кнопку **OK**.

Чтобы назначить выбранному пользователю или группе права доступа к объекту базы данных:

1. На вкладке **Разрешения** в раскрывающемся списке **Тип объекта** (Object Type) выберите тип объекта (**Таблица** (Table), **Запрос** (Query), **Форма** (Form), **Отчет** (Report) или **Макрос** (Macro)).

Замечание

В списке имен объектов не будут отображаться скрытые объекты, если в диалоговом окне **Параметры** (Options) на вкладке **Вид** (View) не установлен флажок **скрытые объекты** (Hidden Objects), позволяющий отображать скрытые объекты.

2. В списке **Имя объекта** (Object Name) выделите имя объекта, права доступа к которому требуется изменить.
3. Чтобы предоставить определенный вид доступа, установите соответствующий флажок в группе **Разрешения** (Permissions). Чтобы запретить определенный вид доступа, сбросьте соответствующий флажок в этой группе.
4. Нажмите кнопку **Применить** (Apply), иначе при выборе другого пользователя группы или другого объекта появится диалоговое окно, требующее подтверждения сделанных изменений. Чтобы подтвердить изменения, нажмите кнопку **Да** (Yes).

Чтобы назначить пользователю или группе права доступа к базе данных:

1. На вкладке **Разрешения** в раскрывающемся списке **Тип объекта** (Object Type) выберите элемент **База данных** (Database).
2. В списке **Имя объекта** (Object Name) отобразится элемент **<Текущая база данных>** (<Current Database>).
3. Установите необходимые разрешения и нажмите кнопку **Применить** (Apply).

Чтобы назначить права доступа к создаваемым объектам базы данных, предоставляемые пользователю или группе:

1. На вкладке **Разрешения** в раскрывающемся списке **Тип объекта** (Object Type) выберите тип объекта (например, **Форма** (Form)).
2. В списке **Имя объекта** (Object Name) выделите элемент, обозначающий новые объекты заданного типа, права доступа к которым требуется изменить (например, **<Новые формы>** (<New Forms>)).
3. Установите необходимые разрешения и нажмите кнопку **Применить** (Apply).

Замечание

Установка или сброс флажков некоторых разрешений влечет установку или сброс других флажков разрешений, поскольку предоставление или отмена определенного вида доступа может привести к предоставлению или отмене другого вида доступа, связанного с измененным. Например, предоставление доступа к таблице вида **чтение данных** (Read Data) влечет предоставление доступа **чтение макета** (Read Design), а отмена доступа **обновление данных** (Update Data) влечет отмену доступа **изменение макета** (Modify Design).

Предоставление права на владение объектами базы данных

Предоставление права на владение объектами базы данных производится с помощью вкладки **Смена владельца** (Change Owner) диалогового окна **Разрешения** (User and Group Permissions). Однако изменить владельца всей базы данных с помощью этого диалогового окна нельзя. Для этого существует отдельная процедура, описан-

ная далее. Предоставление права на владение объектом базы данных пользователю или группе может выполнить лишь пользователь, обладающий административными правами для этого объекта или правом на владение этим объектом.

Чтобы изменить владельца объекта базы данных (т. е. предоставить право на владение этим объектом):

1. Откройте диалоговое окно **Разрешения** (User and Group Permissions) (см. рис. 20.7), как было описано в *разд. "Назначение прав доступа к объектам базы данных"*.
2. Раскройте вкладку **Смена владельца** (Change Owner) (рис. 20.8).

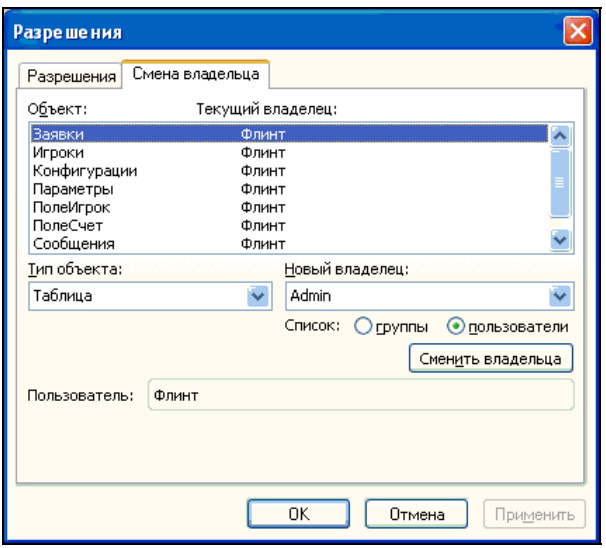


Рис. 20.8. Вкладка **Смена владельца** диалогового окна **Разрешения**

3. В списке **Тип объекта** (Object Type) выберите тип объекта, право на владение которого нужно предоставить другому пользователю или группе.
4. В списке **Объект** (Object) отобразится список всех объектов базы данных заданного типа. Напротив каждого из объектов — в списке **Текущий владелец** (Current Owner) — отобразится имя пользователя или группы, которой в настоящее время назначено право на владение объектом.
5. Выделите в списке **Объект** (Object) имя объекта, владельца которого нужно изменить.
6. Чтобы назначить в качестве нового владельца объекта конкретного пользователя, выберите переключатель **пользователи** (Users), а затем в раскрывающемся списке **Новый владелец** (New Owner) выберите имя этого пользователя. Чтобы назначить в качестве нового владельца объекта группу, выберите переключатель **группы** (Groups), а затем в раскрывающемся списке **Новый владелец** (New Owner) выберите имя этой группы.

7. Нажмите кнопку **Сменить владельца** (Change Owner). Напротив имени объекта отобразится имя его нового владельца.
8. Нажмите кнопку **OK**, чтобы закрыть диалоговое окно **Разрешения** (User and Group Permissions).

Чтобы назначить нового владельца базы данных:

1. Запустите Microsoft Access, подключив файл рабочих групп, содержащий учетную запись нового владельца.
2. Создайте новую базу данных, зарегистрировавшись с именем нового владельца.
3. Импортируйте в созданную базу данных все объекты базы данных, владельца которой нужно изменить.

Внимание!

Для импорта объектов базы данных необходимо обладать правами **открытие/запуск** (Open/Run) для базы данных и **чтение макета** (Read Design) для всех объектов. В противном случае будут импортированы только те объекты, на которые есть необходимые права доступа. Рекомендуется перед импортированием объектов открыть исходную базу данных, зарегистрировавшись с именем пользователя, обладающего административными правами, и предоставить права на **чтение макета** (Read Design) для всех объектов в группе `Users`.



ПРИЛОЖЕНИЯ

Приложение 1. Глоссарий

Приложение 2. Сетевое приложение "Игра в доминирование"

Приложение 3. Описание компакт-диска с примерами

ПРИЛОЖЕНИЕ 1

Глоссарий

В данном приложении приводятся определения терминов, употребляемых в книге. Если не найдете интересующий вас термин здесь, обратитесь к справочной системе Access или электронным словарям в Интернете, например: **http://www.whatis.com**.

ActiveX
Совокупность технологий, созданных компанией Microsoft, позволяющих разрабатывать компоненты программных систем, которые могут взаимодействовать друг с другом в сети, независимо от того, на каком языке они были разработаны.

ASP (Active Server Pages)
Технология фирмы Microsoft, позволяющая использовать языки сценариев для создания динамичного, интерактивного содержимого World Wide Web. Web-страницы в формате ASP содержат сценарий формирования содержимого страницы на языке сценариев (таком, как VBScript, JavaScript). Данные в сценарии могут быть получены с помощью запроса SQL к источнику данных ODBC. Сценарий исполняется Web-сервером (т. е. на серверной стороне) при загрузке страницы, отсюда и название. Сгенерированная страница в формате HTML пересылается клиенту и отображается с помощью программы просмотра Интернета на стороне клиента.

CSS (Cascading Style Sheets)
Язык описания стилей форматирования элементов гипертекста, расширение html.

DHTML (Dynamic HTML)
Расширение языка HTML, включающее новые теги и параметры для описания динамических и интерактивных элементов страниц HTML, более широкий программный доступ к элементам страницы, поддержку CSS и прочие возможности динамического изменения страниц HTML.

FTP (File Transfer Protocol)
Протокол передачи файлов. Предназначен для передачи файлов между удаленными компьютерами в Интернете, а также предоставляет пользователю набор команд для работы с файлами и каталогами на удаленном компьютере.

HTML (Hypertext Markup Language)
Язык разметки гипертекста для публикации документов в глобальной или корпоративной сети, построенной по технологии Интернета.

HTTP (Hypertext Transfer Protocol)
Протокол передачи гипертекста. Поддерживает передачу мультимедийной информации в World Wide Web на основе HTML и неформатированного текста, а также обеспечивает принятие и обработку запросов HTTP.

IDC/HTX (Internet Database Connector file/ HTML Extension file)
Команды для извлечения данных из базы (файл с расширением idc) и шаблон отображения данных (файл с расширением htx). Совокупность файлов IDC и HTX генерирует документ в формате HTML на основе текущих данных, полученных с помощью запроса в формате SQL к источнику данных ODBC, исполняемого сервером IIS.

IIS (Internet Information Server)
Web-сервер, разработанный компанией Microsoft. Включает серверы HTTP и FTP, поддержку технологий ASP, ActiveX и пр. Облегченный вариант IIS для рабочих станций — Personal Web Server.

ODBC (Open Database Connectivity)
Открытый интерфейс доступа к базам данных, разработанный компанией Microsoft для обеспечения стандартизированного программного доступа из прикладных программ с использованием SQL к базам данных различных форматов.

OLE (Object Linking and Embedding)
Протокол связывания и встраивания объектов. Позволяет вставлять в документ или объект одного приложения (клиента OLE) объекты (OLE-объекты) других приложений, поддерживающих протокол OLE (серверов OLE). В Access объекты OLE могут быть вставлены, например, в поле, форму или отчет.

SQL (Structured Query Language)
Язык структурированных запросов для выборки, изменения и удаления данных из таблиц базы данных.

UNC (Universal Naming Convention)
Соглашение об универсальном присвоении имен, позволяющее однозначно именовать ресурс сети, независимо от компьютера, с которого производится доступ, и от физического местонахождения ресурса.

Синтаксис: `\\файл_сервер\общая-папка\путь\имя`.

URL (Uniform Resource Locator)
Унифицированный указатель ресурса — это адрес ресурса Web, включающий в себя идентификатор протокола (например: http, ftp, file и др.) и путь к ресурсу.

Например: `http://www.microsoft.com/rus/`

VBA (Visual Basic for Applications)
Объектно-ориентированный язык программирования, разработанный компанией Microsoft для программирования в приложениях Microsoft Office. VBA является разновидностью языка Visual Basic.

VBScript (Visual Basic Scripting Language)

Язык сценариев компании Microsoft, синтаксически похожий на VBA, используемый, в частности, для интеграции приложений Microsoft Office и Интернета. VBScript, как и VBA, является разновидностью языка Visual Basic.

VML (Vector Markup Language)

Язык описания векторной информации (подмножество XML), применяемый для описания графических элементов и фигур на Web-странице. Рисунки в формате VML, в отличие от графических файлов, имеют меньшие размеры, быстрее загружаются и даже могут динамически трансформироваться с помощью сценариев.

Web-компоненты (Web Components)

Совокупность элементов ActiveX, входящих в состав пакета Microsoft Office, предоставляющих необходимую функциональность документов Microsoft Office в виде элемента управления, который может быть вставлен в Web-страницу (просматриваемую в Microsoft Internet Explorer 4.01 и выше) или страницу доступа к данным (просматриваемую в Microsoft Internet Explorer 5.0 и выше). Страницы с такими элементами могут быть просмотрены при наличии установленных компонентов Web, даже если на локальном компьютере не установлен пакет Microsoft Office.

Web-сервер

Значение 1. Компьютер с определенным программным обеспечением, на котором размещен набор документов, доступный через локальную или глобальную сеть.

Значение 2. Программное обеспечение, предназначенное для доступа к набору документов через локальную или глобальную сеть.

Web-страница

Текстовый документ, содержащий специальные команды для формирования содержимого и его оформления на языке HTML или его разновидностей (DHTML, XML) или других языков подобного назначения (ASP, IDC, IDX).

Web-узел

Набор Web-страниц и других файлов, связанных гиперссылками, объединенных общей темой, расположенных обычно на одном Web-сервере.

World Wide Web

Совокупность мультимедийной информации (текст, графика, аудио- и видеоинформации) глобальной сети Интернет, где переход между ресурсами осуществляется с помощью щелчка по гиперссылке.

XML (Extensible Markup Language)

Расширяемый язык разметки гипертекста, разработанный на базе HTML как формат универсального представления данных в глобальных сетях. XML расширяет возможности HTML, т. к. включает дополнительные элементы, позволяющие описывать и определять новые данные, объекты и свойства, отделяя данные от их представления в виде HTML. Для представления документов Office используется комбинация форматов XML, VML и CSS в дополнение к HTML.

База данных (Database)

Множество данных определенной структуры, с которыми могут работать одна или несколько прикладных программ.

Базовая таблица
Таблица, которая является источником данных запроса, формы, отчета или страницы доступа к данным.

Базовый запрос
Запрос, который является источником данных для другого запроса, формы, отчета или страницы доступа к данным.

Внешнее объединение (Left join, Right join)
Объединение таблиц в запросе, при котором в результирующий набор попадают все записи из одной таблицы и только те записи из другой, у которых связанные поля совпадают. Внешние объединения бывают левыми и правыми.

Внешний ключ (Foreign key)
Одно или несколько полей в таблице, которые содержат ссылку на поле (поля) первичного ключа в другой таблице.

Внутреннее объединение (Inner join)
Объединение таблиц в запросе, при котором в результирующий набор попадают те записи из двух связанных таблиц, у которых связанные поля совпадают.

Выражение (1)
Любое значение или формула, возвращающая значение.

Выражение (2) (Expression)
Вычисляемое значение в объекте базы данных, часто синтаксически начинающееся со знака "равно" (=). Кроме операндов может содержать математические и логические операции, вызовы функций: встроенных и определенных пользователем. Операндами могут быть константы, ссылки на элементы управления, поля таблиц и прочие объекты в базе данных и их свойства. Выражения применяются, например, в качестве значений элементов управления, других свойств объектов. Чаще всего выражение можно построить с помощью Построителя.

Вычисляемое поле
Значение 1. Поле в запросе, значение которого является результатом вычисления над другими полями этого же запроса.

Значение 2. Элемент управления в форме, отчете или на странице доступа к данным, в котором выводится результат вычисления выражения, а не данные, хранимые в таблице.

Гиперссылка (Hyperlink)
Текст, выделенный цветом или подчеркиванием, или графическое изображение, при щелчке на котором осуществляется переход к файлу, определенному месту в файле, странице HTML в World Wide Web или странице HTML во внутренней сети (интранете).

Главная форма
Форма, которая включает другую форму.

Группировка (Grouping)
Разделение данных на группы по определенному критерию.

Данные
Произвольная информация, представленная в символьной (цифровой) форме.

Диаграмма
Графический образ, отражающий количественную зависимость данных.

Диалоговое окно
Один из основных элементов организации приложения.

Заголовок группы (Header of group)
Раздел отчета или страницы доступа к данным, содержащий элементы группировки.

Запись (Record)
Структурированный тип данных, компоненты которого называются полями, имеют имя и тип.

Запрос (Query)
Обращение к СУБД, содержащее задание на выборку, добавление, изменение или удаление записей.

Индекс (Index)
Средство, обеспечивающее быстрый доступ к данным в таблице на основе значений одного или нескольких столбцов. Индекс представляет собой упорядоченный список значений и ссылок на те записи, в которых хранятся эти значения.

Инструкция SQL
Предложение на языке структурированных запросов (SQL), представляющее собой запрос для выборки или обработки данных.

Интернет (Internet)
Значение 1 (имя нарицательное). Название технологии объединения компьютерных ресурсов разных платформ и операционных систем в едином формате для глобальной сети, имеющей динамическую, непостоянную структуру.

Значение 2 (имя собственное). Название единой глобальной сети мирового масштаба, построенной по одноименной технологии.

Интранет (intranet)
Корпоративная сеть, построенная по технологии Интернета, предназначенная для объединения и организации общего доступа к ресурсам одной компании для внутреннего использования и не доступная из глобальной сети Интернет.

Источник данных (DataSource)
Совокупность данных, являющихся источником для представления и обработки данных в приложениях, связанных с базами данных. В Access источником данных называют таблицу или запрос, на основе данных из которых заполняется форма, отчет, страница доступа к данным, элемент управления или другой объект.

Источник данных OBDC
Совокупность данных и информации, необходимой для доступа к ним из приложения с помощью интерфейса (протокола) ODBC.

Источник записей (Record Source)
Таблица, запрос или инструкция SQL, обеспечивающая данные для формы, отчета или страницы доступа к данным.

Каскадное обновление (Cascade update)
Средство поддержания целостности данных в связанных таблицах, которое при изменении значения ключевого поля в главной таблице обеспечивает обновление всех связанных записей в подчиненной таблице.

Каскадное удаление (Cascade delete)
Средство поддержания целостности данных в связанных таблицах, которое при удалении записи в главной таблице обеспечивает удаление всех связанных записей в подчиненной таблице.

Клавиши быстрого доступа
Комбинация клавиш, при нажатии которых элемент управления получает фокус.

Ключевое поле
Поле, которое однозначно идентифицирует каждый объект в таблице, то есть позволяет четко отличить один объект от другого. Может быть частью составного ключа.

Кнопки перехода
Кнопки, которые находятся на нижней границе окон режима таблицы и режима формы, которые используются для перемещения по записям.

Кнопочная форма
Средство, обеспечивающее пользователям доступ к функциям приложения.

Контекстное меню (Popup)
Меню специального вида, которое появляется при нажатии правой кнопки мыши.

Ленточная форма
Форма, в которой выводится одновременно несколько записей из базовой таблицы или запроса.

Макрокоманда (Macrocommand)
Инструкция, обеспечивающая выполнение некоторого действия — элемент языка макрокоманд Access.

Макрос (Macro)
Структура, состоящая из одной или нескольких макрокоманд, которые выполняются либо последовательно, либо в порядке, заданном определенными условиями.

Маркер перемещения
Квадрат в левом верхнем углу выделенного элемента управления формы, отчета или страницы доступа к данным.

Маркер развертывания
Кнопка на странице доступа к данным, предназначенная для отображения и скрытия содержимого одного уровня группировки, или в таблице, содержащей подтаблицу.

Маркеры размера
Небольшие квадраты по краям выделенного элемента управления, раздела формы или отчета, с помощью которых можно менять размеры формы или элемента управления.

Мастера объектов (Object Wizards)
Мастера для создания объектов базы данных Access. Они присутствуют в диалоговых окнах **Новая таблица** (New Table), **Новый запрос** (New Query), **Новая форма** (New Form), **Новая страница доступа к данным** (New Data Access Page) и **Новый отчет** (New Report).

Мастера элементов управления (Control Wizards)
Мастера создания встроенных элементов управления и элементов ActiveX для добавления их в формы, отчеты и страницы доступа к данным.

Модель данных
Совокупность принципов организации базы данных. В СУБД Access используется реляционная модель данных.

Модуль
Объект базы данных Access, содержащий программный код на языке VBA: декларации переменных и функций.

Модуль класса
Модуль, описывающий объект в базе данных. Таковыми являются, например, модули форм и отчетов Access. Модули классов также могут быть созданы для описания пользовательских объектов.

Навигационные кнопки (Navigation Buttons)
Кнопки, расположенные на нижней границе формы или таблицы и использующиеся для перемещения по записям.

Объектная модель (Object model)
Совокупность взаимосвязанных объектов, описывающих программную систему.

Окно базы данных (Database Window)
Окно, которое содержит списки всех объектов базы данных и обеспечивает доступ к этим объектам.

Окно свойств (Properties Sheet)
Окно, предназначенное для просмотра и изменения свойств таблиц, запросов, форм, отчетов, страниц доступа к данным и элементов управления.

Операнд
Элемент данных, участвующий в операции.

Оператор
Допустимое в языке программирования высокого уровня предложение, задающее целостное законченное действие или описание.

Операция
Действие, выполняемое над данными.

Отладка (Debug)
Поиск и исправление ошибок в макросах или процедурах VBA.

Отчет (Report)
Объект базы данных, который используется для вывода на печать данных в отформатированном виде.

Панель инструментов (Toolbar)
Один из видов панелей команд.

Панель команд (Command Bar)
Панель, на которой могут быть размещены элементы управления.

Панель элементов
Панель инструментов, содержащая кнопки, с помощью которых создаются элементы управления в формах, отчетах и страницах доступа к данным.

Первичный ключ (Primary key)
Один или несколько столбцов (атрибутов), которые однозначно идентифицируют каждую запись в таблице, т. е. позволяют четко отличить одну запись от другой.

Подчиненная форма, подчиненный отчет (Subform, subreport)
Форма (отчет), которая встраивается в главную форму (отчет).

Поле (1) (Field)
Элемент данных в записи.

Поле (2) (Text Box)
Элемент управления, предназначенный для отображения текстовых или числовых данных.

Поле подстановок
Список возможных значений поля в таблице, используемый для упрощения ввода данных.

Поле со списком (Combo Box)
Комбинированный элемент управления, объединяющий в себе текстовое поле для ввода и отображения данных и список для выбора значения.

Построители (Builder)
Визуальный интерфейс для формирования значений свойств объектов. Когда Построитель подключается к какому-либо свойству, справа от ячейки с этим свойством появляется кнопка **Построить** () или в контекстное меню включается одноименная команда.

Примечание группы (Footer of group)
Раздел отчета, в котором выводятся итоговые значения группы.

Присоединенный элемент управления
Элемент управления в форме, отчете или на странице доступа к данным, в котором выводятся данные из поля базовой таблицы, запроса или инструкции SQL.

Программа просмотра Интернета (Web Browser)
Приложение, предназначенное для навигации и просмотра документов и другой информации Web. Например, таковым является Microsoft Internet Explorer.

Проект Access
Тип файла Access (имеет расширение adp), который позволяет создавать приложения в архитектуре "клиент-сервер". Появился впервые в версии Access 2000.

Процедура обработки события (Event Procedure)
Реакцией объекта на произошедшее событие может быть выполнение объектом некоторых заданных действий — выполнение специальной процедуры, называе-

мой процедурой обработки события. Разработчик может определять собственные процедуры обработки события и назначать их объекту.

Публикация (Publishing)
Процесс вывода таблиц, форм и отчетов в статическом или динамическом формате HTML с последующей установкой всех связанных файлов в виде приложений World Wide Web на Web-сервер, например, Microsoft Internet Information Server или Microsoft Personal Web Server.

Раскрывающийся список (Dropdown List)
Элемент управления на странице доступа к данным, используемый для выбора значения из списка.

Редактор VBA (Visual Basic Editor, VBE)
Средство разработки и отладки программного кода на VBA в приложениях Microsoft Office.

Редактор сценариев (Script Editor)
Средство разработки и отладки документов в формате HTML.

Реляционная СУБД
СУБД, основанная на реляционной модели данных, в которой связи между наборами данных реализованы на основании совпадения значений полей.

Свободный элемент управления
Элемент управления в форме, отчете или на странице доступа к данным, не связанный с данными.

Свойство (Property)
Характеристика объекта, которая может принимать определенное значение.

Связанная таблица (Linked Table)
Внешняя таблица, которая доступна из текущего файла базы данных.

Связь (Relationship)
Логическое отношение между объектами, представленными таблицами.

Система управления базами данных (СУБД)
Комплекс программ и языковых средств, предназначенных для создания, ведения и использования баз данных.

Снимок отчета (Report Snapshot)
Файл с расширением snp, являющийся копией страницы отчета, который может просматриваться с помощью специальной утилиты Snapshot Viewer.

Событие (Event)
Одномоментное идентифицируемое изменение состояния некоторой системы.

Соединение (Join)
Одна из операций над таблицами баз данных.

Сортировка (Sort)
Изменение порядка, в котором представлены данные.

Составной ключ
Первичный ключ таблицы, который состоит из нескольких полей.

Список объектов
Область в правой части окна базы данных, в которой отображаются объекты Access.

Стандартный модуль
Модуль, не ассоциированный ни с каким объектом, содержащий описания переменных и функций общего назначения, доступных в базе данных.

Статистическая функция
Функция, используемая для расчета итоговых значений.

Столбец подстановок (Lookup)
Список значений, которые могут быть введены в текстовое или числовое поле. Список значений может быть как фиксированным, так и содержаться в таблице или запросе.

Страница доступа к данным (Data Access Page)
Web-страница, которую можно использовать для добавления, редактирования, просмотра и обработки текущих данных в базе данных Microsoft Access или SQL Server.

Строка (String)
Последовательность символов, заключенная в двойные кавычки.

Строка меню (Menu Bar)
Панель команд, на которой расположены элементы управления, представляющие собой раскрывающие списки команд меню.

Схема данных (Relationship)
Графическое представление структуры базы данных Access.

Таблица базы данных
Набор данных в реляционной СУБД. Состоит из переменного количества записей постоянной структуры.

Тип данных (Data Type)
Атрибут поля в таблице, который определяет, какие данные могут содержаться в этом поле.

Уровень группировки
Уровень вложенности группы в отчете или на странице доступа к данным.

Фильтр (Filter)
Выборка из общего массива данных, удовлетворяющих определенному критерию.

Форма (Form)
Объект базы данных, который используется для ввода, просмотра или редактирования записей в таблицах Access.

Целостность данных
Система правил, используемых для поддержания связей между записями в связанных таблицах.

Элемент ActiveX (ActiveX Control)
Компонент, разработанный по технологии ActiveX, который может быть вставлен в документ или диалоговое окно в виде элемента управления.

Элемент группировки

Элемент(ы) управления на форме, отчете или странице доступа к данным, к которому привязано поле источника данных, по значению которого производится группировка значений других элементов управления, к которым привязаны другие поля источника данных.

Элемент управления (Control)

Средство графического интерфейса пользователя, предназначенное для ввода/вывода информации и управления работой программы. Воздействие пользователя на элемент управления вызывает событие в системе.

ПРИЛОЖЕНИЕ 2

Сетевое приложение "Игра в доминирование"

В целях демонстрации применения средств Access для разработки и использования сетевых баз данных и иллюстрации примеров к *гл. 16* этой книги одним из соавторов (В. Михеевой) был разработан пример сетевого приложения "Игра в доминирование". Идея и правила игры были предложены Ф. А. Новиковым, автором книг о Microsoft Office издательства "БХВ-Петербург", за что ему большое спасибо. Готовое приложение "Игра в доминирование" находится на прилагающемся к книге компакт-диске с примерами. Установка и интерфейс этого приложения описаны в отдельных разделах.

Правила игры в доминирование

Познакомимся с правилами игры. Есть игровое поле, которое состоит из *клеток*. Некоторые клетки *смежны*. Смежность клеток определяется выбранной *конфигурацией* (рис. П2.1). Считается, что каждая клетка *доминирует* над собой и над всеми смежными клетками. Доминирование может характеризоваться *весом*, например единицей или другим числом, в соответствии с заданной конфигурацией (рис. П2.2). Темным фоном выделен центр смежности для данной конфигурации.

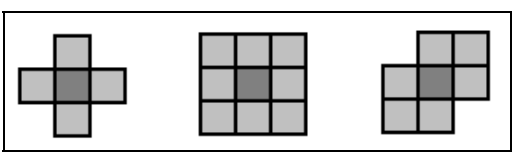


Рис. П2.1. Смежность клеток на игровом поле для разных конфигураций

1	2	1
2	3	2
1	2	1

Рис. П2.2. Конфигурация доминирования над смежными клетками

Каждая клетка игрового поля может быть *раскрашена*. Смежные клетки раскрашиваются одним цветом. Таким образом, набор клеток одного цвета определяет *доминирование* (или *напряжение*) этого цвета на игровом поле (рис. П2.3). На этом рисунке темным фоном выделены центры раскрашенных одним цветом областей поля в соответствии с конфигурацией, показанной на рис. П2.2. Числами показано результирующее напряжение этого цвета в клетках поля.

Конфигурация может быть поставлена на игровое поле в заданную позицию только в том случае, если доминирование данного цвета оказывается строго больше доми-

нирования других цветов во всех смежных клетках (т. е. во всех тех клетках, на которые распространяется доминирование конфигурации). Если конфигурация ставится в заданную позицию, то перекрываемые клетки других цветов перекрашиваются ("съедаются").

1	2	1	0	0
2	4	4	1	0
1	4	4	2	0
0	1	2	1	0
0	0	0	0	0

Рис. П2.3. Доминирование одного цвета на игровом поле

В игре могут принимать участие несколько игроков, каждому из которых выдается одинаковое количество фишек одного цвета (у каждого игрока свой цвет). Задачей игрока является достичь наибольшего доминирования на игровом поле.

Архитектура приложения

По описанию игры в доминирование можно разработать приложение Access, которое будет обслуживать несколько игроков — разных пользователей в сети. Предположим, пользователи работают в одноранговой сети. Рабочие станции пользователей, принимающих участие в игре, можно разделить на две категории: клиенты и

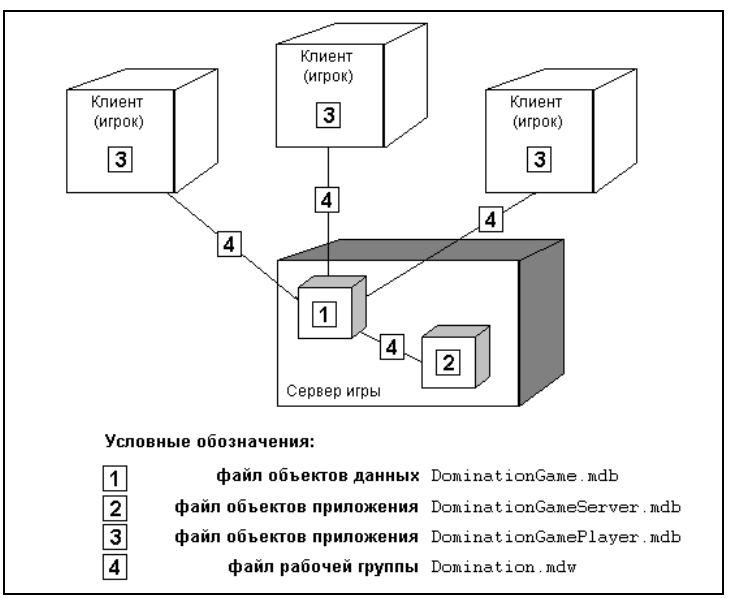


Рис. П2.4. Архитектура приложения "Игра в доминирование"

серверы. На сервере выполняется ядро игры — управляющий компонент приложения. На клиентских рабочих станциях устанавливается компонент, предоставляющий пользователю интерфейс для участия в игре. Таким образом, приложение "Игра в доминирование" представляет собой распределенную базу данных Access с архитектурой "клиент-сервер", которая может быть использована даже в одноранговой сети. Все участники одной игры подключаются к одному серверу по схеме "звезда" (рис. П2.4). (На этом рисунке изображены не только компоненты архитектуры приложения, но и файлы, из которых они состоят. Файлы описаны в *разд. "Установка сетевого приложения "Игра в доминирование" данного приложения*.)

Функциональное описание приложения

Прежде чем приступить к реализации приложения, необходимо четко определить назначение каждого компонента и выбрать метод реализации каждой его функции. Функциональные аспекты компонентов проектируемого приложения удобно представлять в виде *диаграмм вариантов использования* UML (Unified Modeling Language, унифицированный язык моделирования). Составим сначала текстовые описания компонентов приложения "Игра в доминирование", а затем на их основе создадим диаграммы вариантов использования UML.

Краткое описание работы сервера

В начале игры сервер выдает каждому игроку набор фишек определенного цвета и количества. Далее сервер работает по тактам. За один такт сервер проверяет наличие заявки на обслуживание от очередного игрока и, если она есть, выполняет ее и возвращает ответ клиенту.

Краткое описание работы клиента

Игрок подключается к серверу игры. Если подключение выполнено успешно, игрок получает свои фишки и сообщение о том, что все игроки подключены и игра началась. После этого в любой момент времени игрок может сделать ход, т. е. послать серверу заявку на захват определенной клетки игрового поля (в соответствии с заданной конфигурацией доминирования). В ответ игрок может получить от сервера подтверждение хода или отказ. Кроме этого, в любой момент времени игрок может пожелать просмотреть текущее состояние игрового поля. И наконец, в любой момент времени игрок может выйти из игры.

Завершение игры

Игра заканчивается, когда выполняются условия окончания игры. Это может быть истечение заданного промежутка времени, отсутствие незанятых клеток на игровом поле, невозможность сделать ни одного допустимого хода или что-нибудь другое. В нашей реализации игра будет заканчиваться при выполнении хотя бы одного из четырех условий:

- каждый игрок использовал все свои фишки;
- администратор завершил работу сервера игры;

- все игроки вышли из игры;
- истекло время, отведенное на игру.

После завершения игры подводятся ее итоги, и по количеству набранных очков определяется победитель. Результаты игры рассылаются всем подключенным игрокам.

Сервер "Игры в доминирование"

По описанию работы сервера "Игры в доминирование" можно составить следующий список его интерфейсных форм и функций, сгруппированных в модули:

- `Ядро игры` — модуль VBA, отвечающий за выполнение основных функций сервера на всех этапах игры (назовем их состояниями игры). К этим этапам относятся:
  - инициализация игры (создание игрового поля);
  - ожидание подключения игроков, начало игры (назначение цвета игроку, выдача фишек и посылка стартового сообщения);
  - игра начата (обработка заявки игрока — попытка принять ход игрока, изменение текущего состояния игрового поля, отключение игрока);
  - окончание игры (проверка условий окончания игры, подведение итогов с рассылкой результатов игрокам).

  Работу сервера удобно реализовать в виде конечного автомата, который может находиться в одном из перечисленных состояний и переходить из одного состояния в другое после выполнения необходимых действий или в результате возникновения определенного события.
- `Сервер` — форма, обеспечивающая интерфейс ведущего игры. Ведущему предоставляется возможность начать или прервать игру, установить конфигурацию игры, просмотреть текущее состояние игрового поля и закрыть приложение.
- `Конфигурация` — форма, обеспечивающая создание, сохранение, выбор и установку конфигурации игры в доминирование (смежность клеток с весом), а также установку числа подключаемых игроков, количества фишек и размера игрового поля.
- `Обмен данными` — модуль VBA, отвечающий за посылку сообщений игрокам.
- `Игровое поле` — модуль VBA, отвечающий за изображение текущего состояния игрового поля в виде страницы доступа к данным.
- `Адресация клеток` — модуль VBA, отвечающий за выполнение преобразования номера столбца и строки в адрес клетки игрового поля. Адресация клеток на игровом поле реализована подобно адресации клеток на рабочем листе Excel. Для идентификации столбца используются одна или две латинские заглавные буквы от A до Z в лексикографическом порядке, а для идентификации строки — номер: целое число, начиная с единицы.
- `Параметры` — модуль VBA, содержащий функции чтения/записи параметров игры.

Основная процедура серверной части игры в доминирование — это процедура обработки события **Таймер** (Timer) формы `Сервер`, представляющая собой конечный ав-

томат, состояниями которого являются этапы игры. Вызов этой процедуры соответствует такту работы сервера, он происходит периодически, через определенный промежуток времени, заданный в окне свойств формы `Сервер` в качестве значения параметра **Интервал таймера** (Timer Interval) (число миллисекунд). В каждом из состояний сервер вызывает соответствующую состоянию функцию из модуля `Ядро игры` для выполнения определенных действий.

Сервер может находиться в одном из состояний, определяющих функцию, которая будет выполнена сервером за очередной такт. Например, в состоянии "игра начата" сервер проверяет наличие заявки от одного из игроков и, если она есть, выполняет ее. Заявки игроков можно разделить на две категории: сделать ход и выйти из игры. Результатом обработки заявки игрока является выполнение сервером необходимых действий и посылка игроку сообщения о результате этих действий.

По данному описанию сервера создана диаграмма вариантов использования, представленная на рис. П2.5.

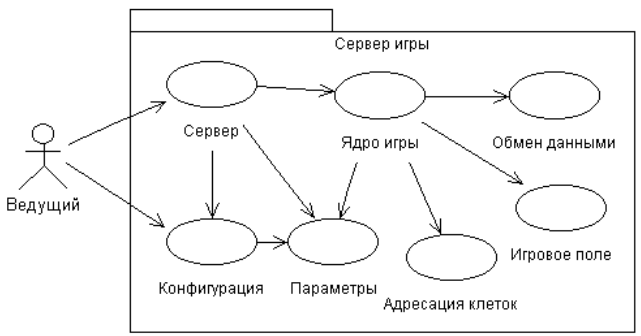


Рис. П2.5. Диаграмма вариантов использования для сервера "Игры в доминирование"

Клиент "Игры в доминирование"

Клиентская часть приложения "Игры в доминирование" значительно проще серверной части. По описанию работы клиента игры в доминирование можно составить следующий список его интерфейсных форм и функций, сгруппированных в модули:

- `Ядро клиента` — модуль VBA, отвечающий за выполнение основных функций клиента на всех этапах игры. Из всех функций в данный модуль вошла лишь функция подключения к серверу. Остальные функции распределены по другим модулям в соответствии с их назначением.
- `Клиент` — форма, обеспечивающая интерфейс игрока. Игроку предоставляется возможность подключиться к серверу игры, сделать ход, просмотреть текущее состояние игрового поля и выйти из игры.
- `Подключиться к игре` — форма, обеспечивающая интерфейс подключения к серверу.
- `Сделать ход` — форма, обеспечивающая интерфейс, позволяющий игроку сделать очередной ход.

- `Обмен данными` — модуль VBA, отвечающий за обмен данными с сервером (за отображение сообщений сервера, адресованных игроку, и посылку заявки на обработку серверу).
- `Игровое поле` — форма, содержащая гиперссылку на страницу с изображением игрового поля.
- `Параметры` — модуль VBA, содержащий функции чтения/записи параметров клиента.

Основная процедура клиентской части игры в доминирование — это процедура обработки события **Таймер** (Timer) формы `Клиент`, которая необходима для реализации обратной связи сервера с клиентом. Поскольку данный пример не предназначен для иллюстрации взаимодействия приложений с помощью OLE Automation, обратная связь сервера с клиентом осуществляется со стороны клиента и реализована в виде вызываемой периодически процедуры проверки наличия адресованных игроку сообщений. Клиент может находиться в одном из состояний, определяющих действия, которые будут выполнены клиентом в ответ на очередное событие таймера. Например, в состоянии "игра начата" клиентом производится прием сообщений от сервера.

Сообщения от сервера делятся на две категории: информационные и управляющие. Управляющие сообщения, в отличие от информационных, не отображаются на экране, а приводят к изменению состояния клиента или выполнению клиентом определенных действий.

Взаимодействие клиента с сервером осуществляется методом посылки серверу заявки. Заявкой игрока может быть ход (координаты клетки) или заявка на выход из игры.

Замечание

Обмен данными между сервером и клиентом игры осуществляется через таблицы сервера приложения. Эти таблицы совместно используются клиентскими приложениями и сервером игры (программным способом — с помощью интерфейса DAO). *(Об этом рассказано в разд. "Программирование с использованием интерфейса DAO" гл. 16.)*

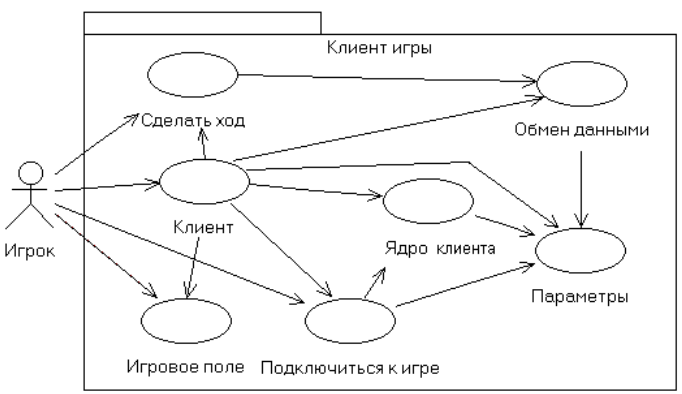


Рис. П2.6. Диаграмма вариантов использования для клиента "Игры в доминирование"

По данному описанию клиента создана диаграмма вариантов использования, представленная на рис. П2.6.

Интерфейс пользователя

Интерфейс пользователя серверной части игры реализован в виде формы `Сервер`, а интерфейс пользователя клиентской части — в виде формы `Клиент`. Параметры запуска баз данных клиента и сервера настроены таким образом, что в момент открытия базы данных отображается интерфейсная форма, а окно базы данных является скрытым. Чтобы отобразить окно базы данных, нажмите комбинацию клавиш <Alt>+<F1> или клавишу <F11>. В окне базы данных отображаются все объекты приложения, их можно изучить. Чтобы открыть окно редактора VBA для просмотра программного кода приложения, выберите команду **Сервис | Макрос | Редактор Visual Basic** (Tools | Macro | Visual Basic Editor) или нажмите комбинацию клавиш <Alt>+<F11>.

Интерфейсная форма `Клиент` (рис. П2.7) содержит кнопки, позволяющие выполнить все действия игрока. Интерфейсная форма `Сервер` (рис. П2.8) имеет кнопки, позволяющие выполнить все действия ведущего игры.

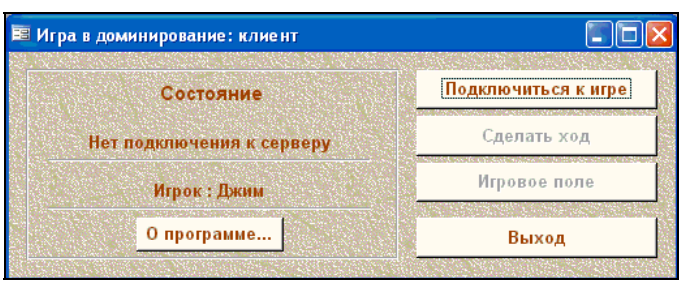


Рис. П2.7. Форма `Клиент`

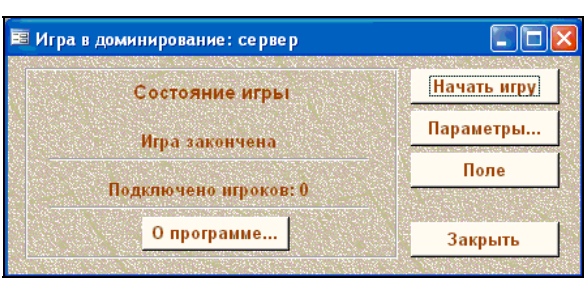


Рис. П2.8. Форма `Сервер`

Кнопка **Параметры** на форме `Сервер` отображает интерфейсную форму `Конфигурация`, позволяющую настроить параметры игры (рис. П2.9).

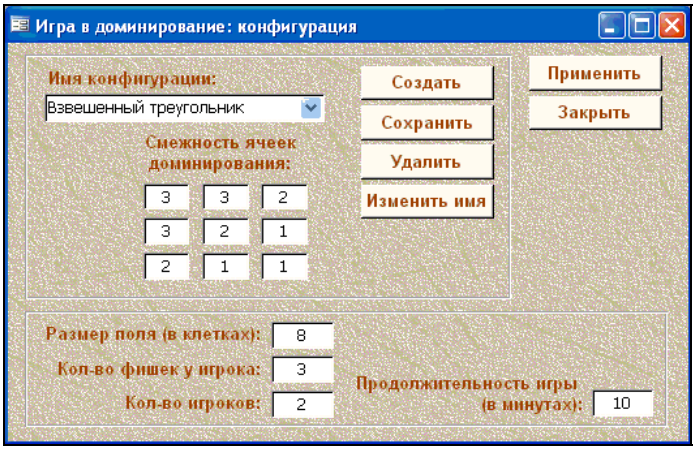


Рис. П2.9. Форма для установки параметров игры

Установка сетевого приложения "Игра в доминирование"

Компоненты сетевого приложения обычно устанавливаются в сети на нескольких компьютерах в соответствии с архитектурой приложения. Приложения обычно поставляются в виде дистрибутива с программой установки, которая позволяет выбрать необходимые компоненты приложения и выполняет установку программного продукта на локальную или сетевую рабочую станцию. Если сетевой программный продукт имеет серверную часть, она обычно поставляется в виде отдельного дистрибутива с собственной программой установки.

В состав приложения "Игра в доминирование" входят лишь четыре файла (см. рис. П2.4), поэтому нет необходимости создавать для них специальную программу установки.

- Серверная часть игры в доминирование реализована в виде базы данных объектов приложения с именем DominationGameServer.mdb и базы данных объектов данных DominationGame.mdb.
- Клиентская часть игры в доминирование реализована в виде базы данных с именем DominationGamePlayer.mdb.
- Кроме этих файлов, для работы с приложением необходим файл рабочей группы Domination.mdw.

Все эти файлы находятся на компакт-диске с примерами к данной книге.

Серверную часть игры в доминирование необходимо поместить в общую папку на файловом сервере или рабочей станции, которая служит сервером игры.

Клиентскую часть игры можно установить двумя способами:

- поместить клиентское приложение в виде копий на рабочие станции всех пользователей, которые будут участвовать в игре. В этом случае каждый игрок ис-

пользует собственную копию клиентского приложения, с которой он работает в режиме монопольного доступа;

- поместить клиентское приложение в общую папку на рабочей станции или файловом сервере. В этом случае игроки могут открыть одно и то же клиентское приложение в режиме общего доступа.

Файл рабочей группы участников игры Domination.mdw необходимо присоединить к копиям Access, установленным на компьютере каждого из участников игры, с помощью программы Администратор рабочих групп или с помощью параметра командной строки `/wrkgrp`. Для ведущего игры его можно также присоединить к базе данных сервера, а для игрока — к базе данных клиента игры, для чего следует поместить файл Domination.mdw в общую папку на файловом сервере или рабочей станции. Можно также скопировать его на рабочие станции игроков, чтобы подключить к их копиям Access локальные копии файла рабочей группы.

Чтобы присоединить существующий файл рабочей группы к конкретной базе данных, в Проводнике Windows создайте ярлык для открытия этой базы данных, в командной строке которого укажите параметр `/wrkgrp` и полное имя файла рабочей группы. Например, чтобы использовать базу данных сервера приложения "Игра в доминирование", создайте ярлык, содержащий команду:

```
"C:\Program Files\Microsoft Office\Office11\MSACCESS.EXE"
"\\GameServer\Domination\DominationGameServer.mdb"
/wrkgrp "\\GameServer\Domination\Domination.mdw"
```

А чтобы использовать базу данных клиента приложения "Игра в доминирование", создайте ярлык, содержащий команду:

```
"C:\Program Files\Microsoft Office\Office11\MSACCESS.EXE"
"\\GameServer\Domination\DominationGamePlayer.mdb"
/wrkgrp "\\GameServer\Domination\Domination.mdw"
```

Только укажите в этих командах реальные полные имена файлов.

(О подключении файла рабочей группы рассказано в разд. "Создание и подключение файла рабочей группы" в гл. 16.)

Для игры необходимо выбрать пользователя, который будет исполнять роль ведущего игры. Этот пользователь должен открыть базу данных DominationGameServer.mdb и зарегистрироваться с именем `Флинт` и паролем `пиастры`. Для использования сервера игры необходимо с помощью команды **Сервис | Служебные программы | Диспетчер связанных таблиц** (Tools | Database Utilities | Linked Table Manager) установить связь с таблицами, хранящимися в файле объектов данных DominationGame.mdb. Чтобы начать игру, ведущий должен нажать кнопку **Начать игру** на форме `Сервер`.

Чтобы принять участие в игре в качестве игрока, нужно открыть базу данных DominationGamePlayer.mdb и зарегистрироваться с использованием учетной записи игрока, например с именем `Джим` без пароля.

Параметры защиты приложения "Игра в доминирование"

Приложение "Игра в доминирование" имеет систему защиты на уровне пользователя объектов базы данных, но программный проект (модули VBA) не защищен и может быть просмотрен любым пользователем.

В файл рабочей группы Domination.mdw, созданный для "Игры в доминирование", были добавлены учетные записи пользователей, перечисленные в табл. П2.1.

Таблица П2.1. *Пользователи приложения "Игра в доминирование"*

Имя пользователя	Пароль	Код пользователя	Группы, в которые входит пользователь
Флинт	пиастры	Adkj7uLn6l84FTAwopW	Admins, Users
Джим	без пароля	jAvO6BLSS2d6KnN5X8O	Users
Ливси	без пароля	4yF3yNprLXhlzmS3iBC	Users
Треллони	без пароля	QFnFceqtWpVTbWRnSOq	Users
Admin	game	<уже существовал>	Users

Пользователи приложения "Игра в доминирование" делятся на две категории:

- *пользователи, ведущие игру* — группа Admins. Этим пользователям разрешен полный доступ к объектам баз данных клиента и сервера. Они также являются администраторами этих баз данных;
- *игроки* — группа Users. Игрокам разрешен ограниченный доступ к объектам баз данных клиента и сервера. Права доступа, назначенные игрокам, приведены в табл. П2.2 и П2.3.

Замечание

Встроенная учетная запись Admin удалена из группы Admins. Все права доступа учетной записи Admin отменены. Таким образом, пользователь, не подключивший файл рабочей группы Domination.mdw, может воспользоваться приложением "Игра в доминирование", но будет обладать при этом лишь ограниченными правами игрока.

Вообще говоря, можно было бы выделить отдельную группу пользователей-администраторов баз данных клиента и сервера, а группе ведущих игру назначить более ограниченные права. Однако в этом нет необходимости, и в нашей реализации ведущие игры являются также администраторами приложения.

Таблица П2.2. *Права доступа к основным объектам сервера "Игры в доминирование" для группы* Users

Тип объекта	Имя объекта	Владелец объекта	Права доступа	Назначение объекта
База данных (Database)	**<Текущая база данных>**	Флинт	**Открытие/запуск**	Сервер приложения "Игра в доминирование"
Таблица (Table)	**Заявки**	Флинт	**Вставка данных**	Прием заявок от игроков

Таблица П2.2 *(окончание)*

Тип объекта	Имя объекта	Владелец объекта	Права доступа	Назначение объекта
Таблица	**Игроки**	Флинт	**Вставка данных**	Подключение игроков к игре
Таблица	**Конфигурации**	Флинт	Нет разрешений	Сохранение конфигураций хода и прочих параметров игрового поля
Таблица	**Параметры**	Флинт	**Чтение данных**	Параметры игры
Таблица	**ПолеИгрок**	Флинт	Нет разрешений	Игроки, занявшие клетки игрового поля
Таблица	**ПолеСчет**	Флинт	Нет разрешений	Число очков у игроков в занятых клетках игрового поля
Таблица	**Сообщения**	Флинт	**Удаление данных**	Рассылка сообщений сервера игрокам
Таблица	**Цвета**	Флинт	Нет разрешений	Назначение цвета каждому игроку
Форма (Form)	**Конфигурация**	Флинт	Нет разрешений	Просмотр, создание, выбор и установка конфигурации хода и прочих параметров игрового поля
Форма	**Сервер**	Флинт	**Открытие/запуск**	Управление игрой
	Новый объект любого типа		Нет разрешений	Создание объекта любого типа

Таблица П2.3. *Права доступа к основным объектам клиента "Игры в доминирование" для группы* `Users`

Тип объекта	Имя объекта	Владелец объекта	Права доступа	Назначение объекта
База данных	**<Текущая база данных>**	Флинт	**Открытие/запуск**, **монопольный доступ**	Клиент приложения "Игра в доминирование"
Форма	**Подключиться к игре**	Флинт	**Открытие/запуск**	Подключение к серверу игры
Форма	**Сделать ход**	Флинт	**Открытие/запуск**	Выполнение попытки занять клетку на игровом поле

Таблица П2.3 *(окончание)*

Тип объекта	Имя объекта	Владелец объекта	Права доступа	Назначение объекта
Форма	**Игровое поле**	Флинт	**Открытие/запуск**	Просмотр игрового поля
Форма	**Клиент**	Флинт	**Открытие/запуск**	Интерфейс игрока
	Новый объект любого типа		Нет разрешений	Создание объекта любого типа

ПРИЛОЖЕНИЕ 3

Описание компакт-диска с примерами

Примеры, описанные в этой книге, содержатся на компакт-диске, который является дополнительным приложением к книге.

Большинство примеров, предназначенных для иллюстрации процедур разработки объектов баз данных и проектов, созданы авторами на основе демонстрационного приложения Борей, которое входит в состав комплекта Microsoft Office 2003. Файлы этих примеров имеют такие же названия, как и соответствующие им исходные файлы базы данных Борей.mdb из русской версии Microsoft Office 2003. Кроме того, были созданы дополнительные примеры, которые хранятся в отдельных файлах. Описание файлов примеров с указанием номеров глав книги, в которых они использованы, представлено в табл. П3.1. В описании примеров, построенных на основе базы данных Борей, мы привели не все объекты, а только те, которые были созданы нами.

Таблица П3.1. *Примеры к книге о Microsoft Office Access 2003 (по главам)*

Номер главы	Имя файла	Описание примера
3	Клиенты.xls Заказы.dbf Клиенты.dbf Поставщики.DB Сотрудники.txt Поставщики.xls Клиенты.csv Supp.sdf Клиенты.htm Сотрудники.xml	Файлы различных форматов, используемые для импорта и присоединения к базе данных Access

5	Борей.mdb	Демонстрационные формы:
		`Форма_Сотрудники`
		`Структура формы`
		`Элементы для выбора`

Таблица П3.1 *(продолжение)*

Номер главы	Имя файла	Описание примера
6	Борей.mdb	Демонстрационный отчет `Товары на складе`, созданный Мастером
	Каталог.snp	Снимок отчета `Каталог`
8	Борей.mdb	Демонстрационные запросы: `Ежемесячная выручка от продаж в 1997 году`; `Ежемесячная выручка от продаж в 1997 году по типам`; `Итоги по клиентам`; `Итоги по клиентам в Германии`; `Копирование заказов`; `Обновление цен`; `Продажи по сотрудникам и странам` (режим Сводной диаграммы); `Счета` (режим Сводной таблицы); `Удаление заказов`; `Холодные клиенты`
9	Борей.mdb	Демонстрационные формы: `Варианты оформления (пример)`; `Клиенты с заказами (пример)`; `Линии и прямоугольники (пример)`; `Набор вкладок (пример)`; `Поиск заказа (пример)`; `Поиск заказа подчиненная форма`; `Поставщики (пример)`; `Поставщики подчиненная (пример)`; `Сведения о клиентах (пример)`; `Сотрудники (пример с переводом страниц)`; `Сотрудники (пример)`; `Сотрудники с заказами (пример)`; `Сотрудники с заказами подчиненная форма`; `Типы (пример)`; `Элементы управления выбора (пример)`
	Заказы.mdb	Демонстрационная форма `Заказы` с примером автоматического ввода значения в поле формы с помощью процедуры обработки события
	Оборудование.mdb	Демонстрационная форма `Оборудование` с примером раскрывающегося списка в форме, заполняемого с помощью определяемой пользователем функции

***Таблица П3.1** (продолжение)*

Номер главы	Имя файла	Описание примера

10	Борей.mdb	Демонстрационные отчеты:
		`Квартальные обороты клиентов (пример);`
		`Подчиненный отчет Квартальные обороты (пример);`
		`Квартальные обороты клиентов 2 (пример);`
		`Квартальные обороты клиентов по товарам (пример);`
		`Заказы клиентов (пример);`
		`Список заказов (пример);`
		`Список заказов с накоплением (пример).`
		Демонстрационные запросы:
		`Квартальные обороты клиентов (пример);`
		`Квартальные обороты клиентов по товарам (пример);`
		`Заказы клиентов (пример).`
		Демонстрационная форма:
		`Подчиненная для квартальных оборотов (пример)`

11	Борей.mdb
	В папке примеров к *гл. 11* кроме файла Борей.mdb есть дополнительные файлы и папки для хранения страниц доступа к данным, их стилей, а также связанных с ними рисунков и документов
		Демонстрационные страницы доступа к данным:
		`Личная карточка;`
		`Продажи по сотрудникам и странам`

	Поставщики.html Счет.htm Счет.xml Счет.xsd Счет.xsl Страница Счет.htm	Примеры экспортированных объектов базы данных `Борей`: таблица `Поставщики` (Suppliers) в формате HTML; отчет `Счет` (Invoices) в формате XML с представлением в формате HTML; отчет `Счет` (Invoices), преобразованный в страницу доступа к данным
	Товары.xsn Товары.xml	Демонстрационная форма `InfoPath` и отобранные с ее помощью данные в формате XML

Таблица П3.1 *(окончание)*

Номер главы	Имя файла	Описание примера
12	Борей.mdb	Демонстрационные макросы: Заказы клиента; Менеджеры по продажам; Поиск заказа; Товары. Демонстрационные формы: Товары; Клиенты
13	Борей.mdb	Демонстрационный запрос Выработка сотрудников. Демонстрационный отчет Выработка сотрудников
15	Борей.mdb Calendar.doc Contract.doc Letter.doc mscal.ocx Resume.doc ИмпортAccess.xls Календарь.xls Праздники.xls Приглашение.dot Товары.xls	Таблицы, формы и отчеты, демонстрирующие возможности OLE. Файлы других приложений, используемые для внедрения, связывания и автоматизации. Элемент ActiveX Календарь
16	*Приложение "Игра в доминирование":* Domination.mdw DominationGame.mdb DominationGamePlayer.mdb DominationGameServer.mdb Игровое поле.htm *Документация:* Readme.txt	Реализация приложения: файл рабочей группы; файл объектов данных; файл объектов приложения клиентской части; файл объектов приложения серверной части; страница доступа к данным Игровое поле. Документация приложения: инструкции по установке и запуску приложения
18	*Портфельная репликация:* Briefcase Контакты.mdb *Репликация Access:* prw_ReportLog.snp	Портфельная репликация базы данных Контакты: портфель с репликой базы данных Контакты; портфельная реплика базы данных Контакты. Репликация базы данных Борей: снимок отчета о создании частичной реплики
19	UT_rptMain.snp	Снимок отчета о преобразовании базы данных Борей в формат SQL Server

Предметный указатель

A

ActiveX 540
◊ добавление на страницу доступа к данным 592
◊ регистрация в Access 592
Admin-пользователь 886
Admins-группа 886
ADO 682
ASP 180, 547, 549

C

COM 683
CSS 181

D

DAO 679, 870
◊ Application:
  ▫ CurrentDb 873
◊ Database 873
  ▫ OpenRecordset 875
◊ DBEngine 871
  ▫ CreateDatabase 874
  ▫ CreateWorkspace 871
  ▫ DefaultType 871
  ▫ OpenDatabase 873
  ▫ Workspaces 871
◊ Field 879
  ▫ Append 879
  ▫ Delete 879
◊ Recordset 875, 879
  ▫ Delete 881
  ▫ Update 881
◊ TableDef 879
◊ Workspace 883
  ▫ Close 872
  ▫ CreateDatabase 874
  ▫ OpenDatabase 873
◊ рабочая область:
  ▫ Jet 871
  ▫ ODBCDirect 871
◊ транзакции 883
DDL 877
◊ ALTER TABLE 877
◊ CREATE TABLE 878
◊ DROP TABLE 878
DHTML 594
DRI 989

H

HTA 559
HTML 539
HTML-страница:
◊ генерируемая сервером 547, 549
◊ статическая 546, 549
HTX 549
HTX/IDC 547

I

IDC 549
Index-Sequential 149
Internet Explorer 540
ISAM 680

J

JavaScript 594

O

ODBC 680
OLE 796
OLE DB 682
OLE-автоматизация 817
Open 149

S

SQL 430, 879

U

UNC 563, 869
URL 538
Users-группа 886

V

VBA, инструкция Resume 881
VBScript 594

W

W3C 180
Web-компоненты 593
Web-папки 606
Web-сервер 539
Web-страница 539
◊ подключение к источнику данных 604
◊ публикация 604
Web-узел 539
World Wide Web 538

X

XML 180, 594
◊ импорт объектов 181
XML-документ 180
◊ внешнее представление 180
◊ данные 180
◊ схема данных 180
XSD 180
XSL 180
XSLT 181

А

Автозамена 760
Автоматизация 796, 817
Автоотчет 312
Автоформат 291, 330, 334, 338, 503
◊ обновление 292
◊ создание 292
◊ удаление 292
Администратор:
◊ ODBC 988, 990
◊ рабочих групп 1017
Администрирование 884
Архитектура:
◊ клиент-сервер 862, 898, 981
◊ многоуровневая 862
◊ файл-сервер 898

Б

База данных 28
◊ архивирование 1007
◊ восстановление 1010
◊ дешифрование 1010
◊ предыдущих версий
  ▫ ограничения 999
  ▫ открытие в Access 2003 1000
  ▫ преобразование 1001
  ▫ совместное использование 998
◊ реляционная 28
◊ сжатие 1009
◊ сжатие и восстановление 1007
◊ шифрование 1010
Библиотека объектов 715
Бланк запроса 216
Блокировка 865
◊ всех записей 865
◊ изменяемой записи 865
◊ отсутствует 865
◊ параметры 867
◊ по умолчанию 865
◊ уровни 866
◊ файл LDB 861
Буфер обмена 123

В

Выражение 244
◊ контрольное 720

Г

Гиперссылка 539
◊ абсолютный адрес 544
◊ базовый адрес 545, 586
◊ в объекте базы данных 541
◊ добавление 541
◊ изменение 544
◊ использование 544
◊ копирование 544
◊ на элемент документа Office 450
◊ название 544
◊ открытие 544
◊ относительный адрес 544
◊ помещение в Избранное 544
◊ создание 450, 541
◊ удаление 544
Границы печатного поля:
◊ изменить 760
Группировка:
◊ интервал группировки 523
◊ по выражению 525
Группировка данных:
◊ в отчете 352
◊ уровни 353

Д

Диспетчер связанных таблиц 987
Документ слияния 835
Доступ к данным:
◊ выбор режима 860
◊ монопольный 859
◊ монопольный для чтения 859
◊ обновление общих данных 867
◊ общий 859
◊ общий для чтения 859
◊ режимы 859

З

Запись 28, 67
Запрос 29, 30, 211
◊ вычисляемое поле 495
◊ на изменение 377
◊ на объединение записей 440
◊ на создание таблицы 378
◊ подчиненный 402, 443
◊ рекурсивный 234
◊ с параметрами 236
◊ перекрестный 389
Защита:
◊ администратор рабочих групп 1017
◊ администрирование задачи 1006
◊ базы данных паролем 1014
◊ владелец 887
◊ встроенные учетные записи 886
◊ данных на SQL Server 1012
◊ кода VBA
  ▫ паролем 894
◊ на уровне пользователей 885
  ▫ группа 885
  ▫ мастер 889
  ▫ смена пароля пользователя 1022
  ▫ снятие 892
  ▫ управление учетными записями 1020
  ▫ установка 889
◊ от несанкционированного доступа 33
◊ политика 884
◊ права:
  ▫ административные 887
  ▫ на владение базой данных 1026
  ▫ на владение объектом 887, 1026
  ▫ назначение 892, 1023
  ▫ неявные 885
  ▫ явные 885
◊ представления в проекте Access шифрованием 1012
◊ проектов Access 1012
  ▫ управление учетными записями 1013
◊ рабочая группа 885
  ▫ файл MDW 885
◊ способы 1011
◊ файл ADE 895
◊ файл MDE 895
◊ шифрованием 1010
Значение по умолчанию 83

И

Идентификаторы 244
Изменение связей 107
Импорт баз данных 1003
Импорт объектов:
◊ необходимые права 1028
Индекс 102
Интернет 538
Интранет 538

К

Класс 669
Клиент автоматизации 817

Ключевое поле 70, 99
Кнопки навигации 67
Кнопочная форма 777
Код репликации 81
Команда 150, 155, 156, 196
◊ вписать диаграмму в окно 932
◊ вставить 54
◊ выбрать помощника 64
◊ вырезать 54
◊ выстроить значки 51
◊ добавить в группу 56
◊ добавить связанные столбцы 930
◊ добавить столбец 99
◊ добавить строки 98
◊ добавить таблицу 105
◊ закрепить столбцы 133
◊ импорт 156
◊ конструктор 69
◊ копировать 54
◊ масштаб 932
◊ мотор 64
◊ настройка 624
◊ новая группа 56
◊ новая запись 113
◊ новая таблица 924
◊ новое название 935
◊ открыть 66
◊ параметры 62
◊ переименовать 53
◊ переименовать столбец 73, 98
◊ печать 134, 243
◊ предварительный просмтр 136
◊ проверка условий 115
◊ свойства 55, 116, 426, 917, 927
◊ связь 156
◊ скрыть 62
◊ скрыть таблицу 105, 936
◊ схема макета 930
◊ схема разметки 930
◊ триггеры 922
◊ удалить 53
◊ удалить запись 114
◊ удалить столбец 99
◊ удалить строки 98
◊ удалить таблицу 936
◊ упорядочить значки 52
◊ фильтр по выделенному 145
◊ шрифт 935
Команда меню:
◊ вид:
▫ панели инструментов 112, 782
▫ режим sql 231, 239, 399
▫ свойства 55, 426
◊ вставка:
▫ модуль 700
▫ новая запись 113
▫ подтаблица 127
▫ поле подстановок 125
▫ столбец 99
▫ строки 98, 223
◊ записи:
▫ удалить фильтр 140
▫ фильтр 145, 239
◊ запрос:
▫ добавить таблицу 221
▫ добавление 382
▫ обновление 384
▫ создание таблицы 379
◊ запуск:
▫ по шагам 623, 633
◊ правка:
▫ буфер обмена Office 123
▫ вставить, вырезать 54
▫ добавить из буфера 123
▫ найти 141, 142
▫ очистить бланк 224
▫ удалить 53, 98, 114
▫ удалить запись 114
▫ удалить строки 98, 223
◊ связи:
▫ изменить связь 106
◊ сервис:
▫ макрос:
• выполнить макрос 622, 623
• преобразовать макросы 645
• преобразовать макросы формы 643
• создать меню из макроса 626
• создать панель инструментов из макроса 625
▫ параметры 34, 71, 129, 139, 759
▫ параметры автозамены 120
▫ связи с Office:
• слияние с Microsoft Office Word 835
▫ служебные программы:
• диспетчер кнопочных форм 777
▫ схема данных 104
▫ элементы управления ActiveX 848
◊ создать:
▫ текстовый документ 904
◊ справка 64
◊ схема:
▫ изменить настраиваемое представление 933
▫ просмотреть разрывы страниц 936
▫ шрифт для надписи 935

◊ удалить столбец 99
◊ файл 34
  ▫ загрузить из запроса 238
  ▫ открыть 34
  ▫ отправить 148
  ▫ параметры страницы 135
  ▫ печать 134, 243, 936
  ▫ подключение 904, 910
  ▫ предварительный просмотр 136
  ▫ создать 38, 908
  ▫ сохранить 225, 922
  ▫ сохранить как 646
  ▫ сохранить как запрос 238
  ▫ экспорт 147, 204
◊ формат:
  ▫ высота строки 131
  ▫ закрепить столбцы 133
  ▫ освободить все столбцы 133
  ▫ отобразить столбцы 132
  ▫ переименовать столбец 99
  ▫ подтаблица 126, 128
  ▫ режим таблицы 130
  ▫ скрыть столбцы 132
  ▫ ширина столбца 131
Компонент 817
Константа 244
◊ встроенная 657
◊ объявление 656
Контейнер OLE 797
Контекстное меню 781, 790
Контроллер автоматизации 817
Контрольное выражение 720

M

Макет таблицы 129
Макрокоманда 610
Макрос 610
◊ преобразование 643
Маска ввода 83, 94
Массив 658, 659
Мастер:
◊ автостраниц 561
◊ автоформ для многотабличной формы 472
◊ защиты на уровне пользователей 889
◊ подключения данных 571
◊ подчиненной формы/отчета 475, 511
◊ преобразования в формат SQL Server 984, 987
◊ разделения баз данных 985
◊ страниц доступа к данным 564
◊ форм 258
◊ для многотабличных форм 474
◊ элемента Группа 452
◊ элемента Кнопка 463
◊ элемента Поле со списком 480
◊ элемента Список 457, 576
◊ элементов 447
Масштабируемость 983
  ▫ Меню 48
Метод 669
Миграция 980
Модуль 610, 651, 696
◊ класса 697
◊ компиляция 706

H

Надпись, свойства по умолчанию 526
Нормализация 30

O

Область:
◊ действия 660
  ▫ переменной 660
  ▫ процедур, функций 661
◊ задач 37
◊ строк 404
◊ фильтра 404
Обновляемый статический набор записей 953
Обозреватель объектов 713
Обработка ошибок 706
Объект 669
◊ Access 51
◊ OLE 797
◊ OLE DB:
  ▫ источники данных 902
  ▫ команды 903
  ▫ сеансы 902
  ▫ транзакции 903
◊ автоматизации 818
◊ внедренный 797
◊ глобальный 692
◊ связанный 797
◊ ссылки 692
Объектная модель 670
◊ Microsoft Excel 821
◊ Microsoft Word 829

Окно 46, 47
Окно свойств:
◊ ввод значений 338
◊ отображение 282
◊ отчета 336
◊ переключение между объектами 337
◊ построитель 338
◊ раздела отчета 336
◊ элемента управления 336
Операнды 245
Операторы 245
◊ ветвления 663
◊ выбора 663, 664
Орфография 769
Отладка меню 716
Отношение 28
Отчет 31
◊ вывод итогов 508
◊ выделение 334, 335
◊ высота 331
◊ вычисления 531
◊ вычисляемое поле 350
◊ границы 331
◊ группировка 522
◊ группировка данных 352
◊ масштабирование в режиме просмотра 355
◊ нумерация страниц 348
◊ область выделения 335
◊ окно свойств 336
◊ отправка по e-mail 323
◊ оформление по шаблону 353
◊ перекрестный 517, 747
  ▫ динамический 518
  ▫ статический 517
◊ печать 355
  ▫ поля страницы 354
  ▫ в несколько колонок 520
◊ подчиненный:
  ▫ поля связи 516
  ▫ применение 506
  ▫ создание 510
◊ преобразование 321
  ▫ в документ Word 322
  ▫ в снимок отчета 323
  ▫ в страницу доступа к данным 313
  ▫ в электронную таблицу Excel 322
  ▫ форматы 321, 322
  ▫ экспорт 322, 323
◊ публикация снимков отчета 326
◊ разделы 330
◊ разметка 331
◊ разрыв страницы 348
◊ режимы работы 326
◊ сетка 344
◊ снимок отчета 323
◊ создание:
  ▫ автоматическое 312
  ▫ в режиме Конструктора 327
  ▫ с помощью Мастера 313
◊ сортировка 523
  ▫ данных 352
◊ составной 506
◊ сохранение 313
◊ страницы:
  ▫ нумерация 348
  ▫ поля 354
◊ структура 330
◊ разрыв 348
◊ фильтрация данных 535
◊ форматирование 334

П

Панель инструментов 781
◊ веб-узел 545
◊ конструктор отчетов 331, 332
◊ конструктор форм 265, 268
◊ панель элементов 330, 334
◊ предварительный просмотр 327
◊ режим формы 303
◊ формат (форма/отчет) 265, 270
◊ форматирование 331
Панель команд 781
◊ буфер обмена 365
◊ поиск файла 363
Панель элементов 264, 265, 446, 573
Первичный ключ 99
Передача параметров 650
Переменная:
◊ объектная 657, 693
◊ объявление 652
◊ статическая 661
Печать:
◊ заголовка группы 528
◊ нумерация записей в отчете 532
◊ отчета 355
◊ параметры печати группы 524, 528
◊ скрытие повторяющихся данных 529
Поле 28, 67
◊ вычисляемое 225, 493
  ▫ вывод условий отбора данных 534

- вычисление процентов 532
- итоговое 527
- объединение данных 533
- сумма с накоплением 531

◊ объекта OLE 81
◊ подстановки 82, 462
◊ свойства по умолчанию 526
Полоса прокрутки 67
Построитель 447
Потребители данных 901
Представление:
◊ группировка 940
◊ объединение таблиц 939
◊ свойства 943
◊ создание 938
Преобразование 322
◊ объекта базы данных:
- в Web-страницы 546
- в страницу доступа к данным 558
- в форматы Web 547
- импорт 552
- экспорт 550

◊ таблиц 322
Преобразование баз данных:
◊ в формат Access 97
- ограничения 1004

◊ варианты форматов 1002
◊ интеграция с SQL Server 983
- выполнение 990
- завершение 995
- подготовка 987

◊ совместимость форматов 996
Присоединение 154
Присоединенная рамка объекта 799
Провайдеры данных 901
Программа просмотра Интернета 540
Проект 651, 702
Проект Access:
◊ свойства подключения 1013
◊ создание из базы данных 990
Процедура 649
◊ вызов 649
◊ обработки событий 650, 670, 744
◊ определение 649
Процесс:
◊ клиент 898
◊ сервер 898
Публикация 539, 603
◊ на Web-сервере 606
◊ на сервере FTP 608
◊ по электронной почте 607

P

Рабочая группа 1016
Разделение баз данных 984
Расширенный фильтр 239
Редактор сценариев 594
Режим:
◊ выполнения 52
◊ конструктора 52
◊ сводной диаграммы 66, 70
◊ сводной таблицы 66, 70
◊ таблицы 66
◊ открытия базы данных 37
Результирующий набор 143
Реплика 957, 966
◊ защита 976
◊ набор реплик 966
◊ область видимости 966
◊ основная 966, 967
◊ преобразование в обычную базу данных 976
◊ приоритет 966
◊ синхронизация 975
◊ создание 967
◊ удаление из набора 977
◊ частичная 966, 969, 970
Репликация 957
◊ базы данных 958, 966
◊ в Internet Explorer 978
◊ в Windows 978
◊ на сервере Web 958, 978
◊ портфель:
- забрать 963
- просмотр 962
- синхронизация 963
- создание 960

◊ портфельная 958
◊ программная 958
◊ проекта Access 958, 977
◊ синхронизация 957, 966
- косвенная 974
- прямая 973
- устранение конфликтов 975
- через Интернет 974

◊ создание резервной копии 977

C

Свободная рамка объекта 799
Сводная диаграмма 420
Сводная таблица 403

Свойство 669
◊ задание значения в процедуре VBA 482
Свойства поля 82, 83
Связывание 154
Связь 28, 1015
Семейство 693, 667, 670
Сервер:
◊ OLE 797
◊ автоматизации 817
Сервисы 902
Синхронизация форм 745
Система управления базами данных 28
Системные объекты 759
Скрытые объекты 759
Снимки отчетов 323
Событие 628, 670, 725
◊ данных 725
◊ клавиатуры 730
◊ мыши 731
◊ окна 735
◊ ошибок 737
◊ печати 733
◊ таймера 737
◊ текущая запись 725
◊ удаление 725
◊ фильтра 734
◊ фокуса 729
Соединение 227, 228, 232
Сортировка данных в отчете 352
Составные документы 797
Список подстановок 124
Ссылка 715
◊ на подчиненную форму или отчет 636
◊ на форму или отчет 635
◊ на элемент управления 636
Ссылочная целостность 107
Статистические функции 395
◊ Стек вызовов 723
Страница доступа к данным 32, 559
◊ Web-компоненты 593
◊ вычисляемое поле 580
◊ гиперссылка 588, 589
◊ группировка 564, 579, 580
◊ кнопки перехода по записям 567
◊ маркер развертывания группы 567, 580
◊ несвязанный раздел 574
◊ оформление 590
◊ подключение 869
◊ панель элементов 573
◊ параметры группы 581
◊ параметры подключения 570
◊ подключение 569
◊ режим Конструктора 560, 572
◊ режим Просмотра 560
◊ рисунок 587, 588
◊ свойства 585
◊ связанный текст 580
◊ создание 561, 562
▫ автостраница 561
▫ конструктор 572
▫ мастер 564
◊ сортировка 579, 582
◊ сохранение 562
◊ список полей 570, 574
◊ тема 590
◊ фильтрация 580
◊ элемент ActiveX 592
◊ элемент группировки 580
◊ элемент управления 573
▫ добавление 576
▫ источник данных 581
▫ кнопки перехода 578
▫ свойства 583
▫ удаление 576
◊ ярлык 559
▫ настройка 870
▫ переименование 568
▫ подключение к странице 569
▫ удаление 568
Строка:
◊ меню 781
◊ состояния 47
Схемная привязка 951
Счетчик цикла 666

T

Таблица:
◊ базовая 216
◊ главная 30
◊ подчиненная 30
◊ связанные 30
◊ ошибок импорта 168
Тегированное значение 594
Тип данных 28
◊ гиперссылка 82
◊ дата/время 81
◊ денежный 81
◊ логический 81
◊ поле MEMO 80

◊ счетчик 81
◊ текстовый 80
◊ числовой 80
Точка останова 718
Транзакция 883, 982
Триггер 989

У

Указатель текущей записи 67
Унифицированный локатор ресурсов 538
Уровни безопасности 646, 647
Условие на значение 83

Ф

Файл:
◊ ADE 895
◊ MDE 895
◊ блокировок 861
◊ объектов данных 984
◊ объектов приложения 984
◊ рабочей группы 1016
▫ восстановление 1019
▫ подключение 886, 891, 1017
▫ создание 891, 1018
Фильтр 212
◊ серверный 952
Фоновый рисунок 799
Форма 256
◊ автоформат 257, 291
◊ всплывающая 491
◊ выделение 282
◊ вычисления 483, 493
◊ вычисляемое поле 493
◊ данные:
▫ добавление записи 305
▫ изменение записи 308
▫ новая запись 305
▫ поиск и замена 305
▫ пустая запись 305
▫ сортировка 305
▫ удаление записи 309
▫ фильтрация 305
◊ диалоговое окно 491
◊ иерархическая 473
◊ кнопки перехода по записям 307
◊ многостраничная 467, 468, 484
▫ с набором вкладок 487
▫ с переводом страниц 484
◊ многотабличная:
▫ создание 471, 472, 474
◊ модальная 491
◊ орфография 305
◊ основная 471
◊ оформление 290
▫ в едином стиле 503
▫ советы 499
▫ шаблон 272
◊ параметры 272
◊ переход на определенную запись 480
◊ переход по элементам 282, 308
▫ настройка 501
◊ печать 264, 304
◊ подчиненная 471
▫ добавление в форму 475
▫ связующие поля 479
▫ связывание с главной 478
◊ последовательность перехода 282
◊ примеры 479
◊ разделы 266
▫ выделение 282
▫ цвет 293
◊ режим Конструктора 264
▫ сетка 267, 286
◊ режим Предварительного просмотра 264
◊ режим работы 262
◊ режим Таблицы 262, 302
◊ режим Формы 262, 302
◊ рисунок в фоне 294
◊ свойства 481, 491, 493
◊ создание:
▫ автоматическое 257
▫ в режиме Конструктора 265
▫ с помощью Мастера 259
◊ сохранение 301
◊ структура 266
◊ фильтрация 481
◊ цвет 293, 502
◊ экранная 30
Формат БД по умолчанию 767
Функция 245, 649
◊ Avg 497
◊ CCur 495
◊ Count 497
◊ Date 494
◊ DAvg 498
◊ DCount 498
◊ DLookup 498

Продолжение рубрики см. на с. 1068

Функция *(прод.)*:
◊ DMax 498
◊ DMin 498
◊ DSum 497
◊ Max 497
◊ Min 497
◊ Nz 497
◊ Sum 495, 497
◊ встроенная 250, 950
◊ определение 649
◊ определяемая пользователем 949
◊ статистическая 497
◊ скалярная 950
◊ табличная 950

X

Хранимая процедура 945

Ц

Целостность 109, 989
Цикл:
◊ с перечислением 666
◊ с условием 665

Ш

Шаблон оформления отчета 353

Э

Элемент управления 256
◊ ActiveX 447, 841
◊ буфер обмена 289
◊ встроенный 447
◊ выделение 282, 285, 344
◊ выключатель 274, 451
◊ выравнивание 286, 287, 344, 345
◊ вычисляемое поле 350
◊ вычисляемый 448, 493
◊ гиперссылка 450
◊ группа 273, 452
  ▫ выделение 452
  ▫ значение 452
  ▫ переключателей 573
  ▫ создание 452
◊ группировка элементов 289
◊ задание значения в процедуре VBA 483
◊ значение по умолчанию 306
◊ изменение высоты 344
◊ изменение размеров 343—346
◊ кнопка 463, 573
  ▫ назначение комбинации клавиш 466
  ▫ отмены действия 467
  ▫ по умолчанию 467
  ▫ создание 463, 578
  ▫ перехода по записям 574, 578
◊ копирование 289
  ▫ формата 347
  ▫ через буфер обмена 347
◊ линия 348, 469
◊ маркер изменения размеров 343
◊ маркер перемещения 343
◊ на заднем плане 288
◊ на переднем плане 288
◊ набор вкладок 468, 484
  ▫ выделение 491
  ▫ добавление/удаление страниц 489
  ▫ свойства 490
◊ надпись 273, 449, 573
  ▫ изменение текста 290
  ▫ советы 501
  ▫ создание 275
◊ объект, созданный другим приложением 467
◊ оформление 296
  ▫ копирование формата 297
  ▫ советы 500
  ▫ формат по условию 298
  ▫ цвет 300
◊ переключатель 274, 451, 573
◊ перемещение 284, 343, 345, 577
◊ подчиненная форма/отчет 469, 475
  ▫ связывание с главной формой 478
  ▫ создание 476
◊ поле 273
  ▫ ввода 573
  ▫ подстановки 462
  ▫ связанное 280
  ▫ со списком 274, 461
  ▫ текстовое 450
◊ преобразование в другой элемент 290, 470
◊ присоединение надписи к нему 449
◊ присоединенная рамка объекта 467
◊ присоединенный 448
◊ прямоугольник 349, 469
◊ размеры:
  ▫ выравнивание 287
  ▫ изменение 284

- размещение 500
- разрыв страницы 348, 467, 484
- раскрывающийся список 573
- рисунок 467
- свободная рамка объекта 467
- свободный 448
- свойство:
  - Id 584
  - автоподстановка 462
  - блокировка 584
  - всплывающая подсказка 584
  - вывод на экран 584
  - данные 452, 458, 494
  - дополнительные сведения 468
  - заглавия столбцов 459
  - значение 273
    - параметра 452
    - по умолчанию 274, 451
  - имя 458
  - индекс перехода по <Tab> 584
  - источник данных 584
  - источник строк 458
  - маска ввода 462
  - несвязное выделение 461
  - ограничиться списком 462
  - основные поля 478
  - от верхнего края 468
  - от левого края 468
  - оформление 469
  - подпись 449
  - подчиненные поля 478
  - по умолчанию 526
  - присоединенный столбец 459, 461
  - рисунок 465
  - текст 450
  - тип границы 469
  - тип источника строк 458
  - тип фона 469
  - только для чтения 584
  - формат поля 462
  - цвет границы 469
  - цвет фона 469
  - число десятичных знаков 462
  - число столбцов 459
  - ширина границы 469
  - ширина столбцов 459
- связанный текст 573
- событие После обновления 480, 482, 483
- создание 275
  - с помощью панели элементов 339
  - с помощью списка полей 342
- список 274, 457, 573
  - аргументы заполняющей функции 460
  - выбор нескольких элементов сразу 461
  - заполнение программно 459
  - заполняемый функцией 459
  - значение 457
  - из нескольких столбцов 457
  - свойства 458
  - создание 276, 339, 340, 576
  - список значений 459
  - список полей 459
  - типы источников строк 459
- текстовое поле 573
- удаление 281, 342
- флажок 274, 451, 573
- цвет 293, 502
- элемент выбора 452

Я

Язык структурированных запросов 29
Язык сценариев 594