СЕРГЕЙ КАШАЕВ

1С:Предприятие 8.2

Программирование и визуальная разработка

НА ПРИМЕРАХ

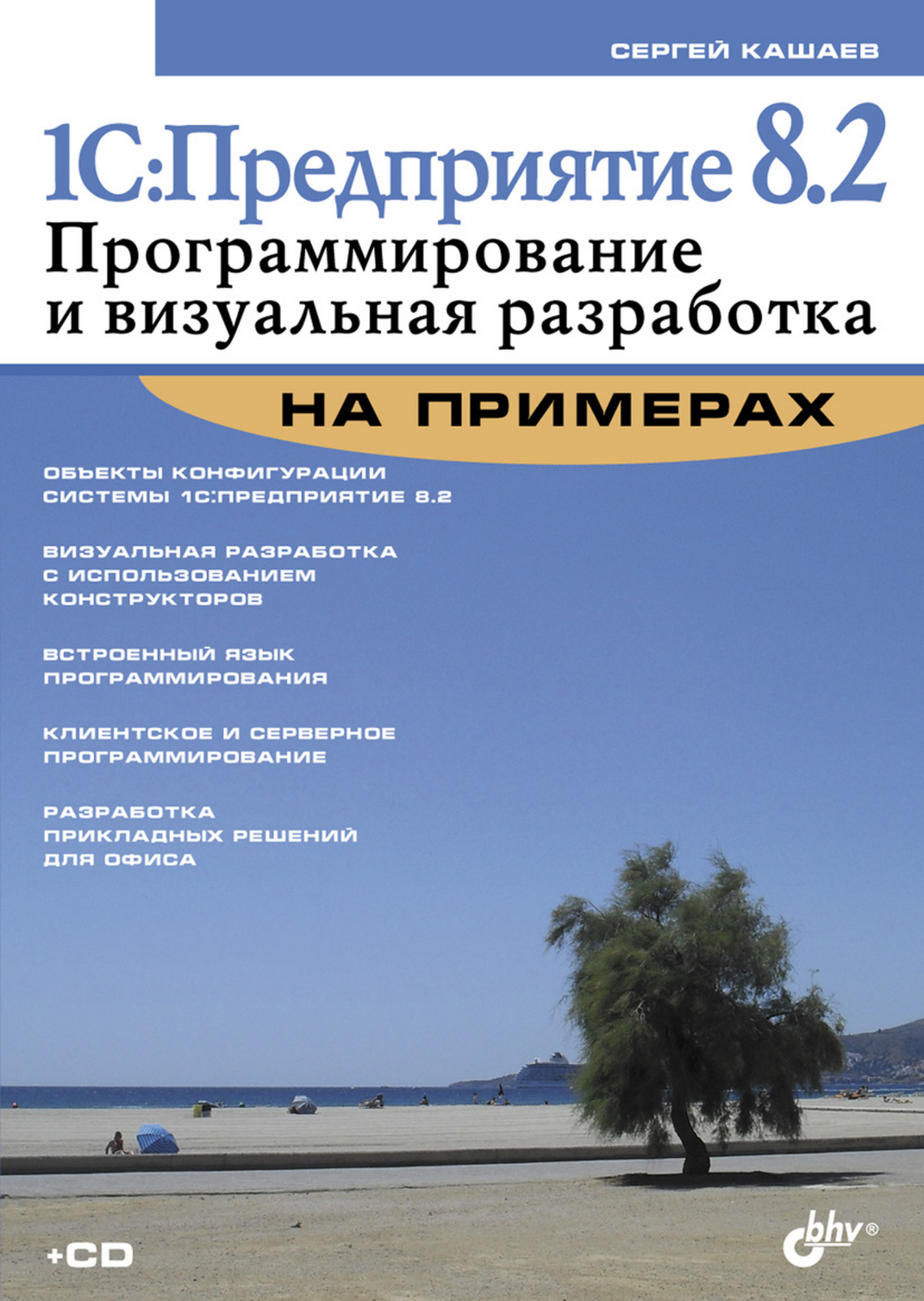
ОБЪЕКТЫ КОНФИГУРАЦИИ СИСТЕМЫ 1С:ПРЕДПРИЯТИЕ 8.2

ВИЗУАЛЬНАЯ РАЗРАБОТКА С ИСПОЛЬЗОВАНИЕМ КОНСТРУКТОРОВ

ВСТРОЕННЫЙ ЯЗЫК ПРОГРАММИРОВАНИЯ

КЛИЕНТСКОЕ И СЕРВЕРНОЕ ПРОГРАММИРОВАНИЕ

РАЗРАБОТКА ПРИКЛАДНЫХ РЕШЕНИЙ ДЛЯ ОФИСА



+CD

bhv®

Сергей Кашаев

1С:Предприятие 8.2
Программирование и визуальная разработка
НА ПРИМЕРАХ

Санкт-Петербург
«БХВ-Петербург»
2011

УДК 681.3.06
ББК 32.973.26-018.2
К31

Кашаев С. М.

К31 1С:Предприятие 8.2. Программирование и визуальная разработка на примерах. — СПб.: БХВ-Петербург, 2011. — 320 с.: ил. + CD-ROM

ISBN 978-5-9775-0722-6

Обучение разработке конфигураций и программированию на платформе 1С:Предприятие 8.2 построено на практических примерах различной сложности. Основной упор сделан на подробное описание технологии разработки конфигураций с использованием конструкторов и встроенного языка программирования, включая конструирование управляемого интерфейса, разработку управляемых форм, использование новых возможностей настройки отчетов, настройку командного интерфейса, клиентское и серверное программирование и ряд других ресурсов. Большое внимание уделено языку запросов и построению разнообразных отчетов. Тематика примеров в основном связана с учетом денежных средств, в частности, рассмотрены задачи автоматизации учета денежных поступлений для организаций, предоставляющих платные образовательные услуги. Компакт-диск содержит разработанные в книге конфигурации, а также упоминаемые документы, созданные в приложении MS Word.

Для начинающих 1С-программистов и опытных 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	*Римма Смоляк*
Компьютерная верстка	*Ольги Сергиенко*
Корректор	*Зинаида Дмитриева*
Дизайн серии	*Игоря Цырульникова*
Оформление обложки	*Елены Беляевой*
Зав. производством	*Николай Тверских*

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 30.06.11.
Формат 70×100 1/16. Печать офсетная. Усл. печ. л. 25,8.
Тираж 2000 экз. Заказ №
"БХВ-Петербург", 190005, Санкт-Петербург, Измайловский пр., 29.

Санитарно-эпидемиологическое заключение на продукцию
№ 77.99.60.953.Д.005770.05.09 от 26.05.2009 г. выдано Федеральной службой
по надзору в сфере защиты прав потребителей и благополучия человека.

Отпечатано с готовых диапозитивов
в ГУП "Типография "Наука"
199034, Санкт-Петербург, 9 линия, 12

ISBN 978-5-9775-0722-6

© Кашаев С. М., 2011
© Оформление, издательство "БХВ-Петербург", 2011

Оглавление

Введение ... **7**
Общий взгляд на систему 1С:Предприятие ... 8
Краткое содержание книги ... 10
Компакт-диск ... 11
От автора ... 12

Глава 1. Знакомство с системой 1С:Предприятие 8.2 ... **13**
Архитектура системы 1С:Предприятие ... 15
Постановка задачи ... 17
Окно запуска ... 17
Создание новой информационной базы ... 18
Назначение объектов конфигурации ... 22
Перечисления ... 23
Справочники ... 27
Справочник *Должности* ... 28
Справочник *Специалисты* ... 32
Справочник *Курсы* ... 42
Печать и настройка формы списка справочника ... 47
Справочник *Клиенты* ... 48
Печать элемента справочника *Клиенты* ... 51
Внесение изменений в печатную форму ... 60
Документы ... 63
Документ *ВнесениеОплаты* ... 63
Заполнение вкладки *Данные* ... 63
Форма документа ... 65
Программирование формы документа ... 68
Создание печатной формы документа *ВнесениеОплаты* ... 73
Подсистемы ... 76
Информационная панель ... 81
Системные команды и главное меню ... 83
Особенности режима *1С:Предприятие* ... 84
Варианты клиентских приложений ... 86
Итоги знакомства с конфигуратором ... 86

Глава 2. Документы и регистры накопления 89
Объект конфигурации *документ* 89
Регистры накопления 91
Создание конфигурации 93
Регистр накопления остатков 95
Изменения в документе *Поступление* 97
Расход по регистру накопления 103
Оборотный регистр накопления 105
Итоги 109

Глава 3. Регистры сведений 111
Учет при организации работы филиалов 111
Подготовка информационной базы 112
Непериодический регистр сведений 113
Документ *Поступление* 115
Процедура установки ответственного в поле документа 117
Создание периодического регистра сведений *Цены* 121
Использование регистра *Цены* в документе *Поступление* 123
Пример информационной базы для учета квитанций 126
Представление информации о тарифах 128
Вспомогательный справочник *ВидТарифа* 128
Регистр сведений *Тарифы* 129
Контроль ошибок в форме регистра *Тарифы* 131
Справочник *Договоры* 132
Регистр сведений *Квитанции* 134
Документ *КвитанцияПоДоговору* 135
Формирование печатной формы 139
Доработка документа по фиксированию квитанций 143
Формирование документа Microsoft Word 146
Внесение дополнений в бланк квитанции 146
Разработка процедуры заполнения бланка 149
Заполнение полного бланка с квитанцией 152
Итоги 155

Глава 4. Запросы 157
Взгляд на язык запросов 159
Извлечение информации о выписанных квитанциях 160
Использование табличного документа для вывода результатов выполнения запроса 164
Псевдонимы и упорядочивание в запросе 168
Отбор максимальных (минимальных) значений 171
Ключевое слово *ИЕРАРХИЯ* 172
Исключение в отборе повторяющейся информации 175
Ключевое слово *ГДЕ* 177
Использование логических операторов в запросе 178
Агрегатные функции в запросах 181
Использование параметров в запросах 184
Ключевое слово *МЕЖДУ* 187
Ключевое слово *ИМЕЮЩИЕ* 189
Функция расчета итогов в запросах 191

Объединение результатов нескольких запросов ... 194
Более сложные конструкции языка запросов ... 197
Отбор специалистов по указанной должности ... 197
Отбор курсов преподавателя ... 202
Соединение таблиц при организации запроса ... 204
Левое внешнее соединение ... 205
Правое внешнее соединение ... 208
Полное внешнее соединение ... 209
Итоги ... 211

Глава 5. Отчеты и система компоновки данных ... 213
Отчеты по *главе 2* ... 215
Отчет по остаткам товаров ... 215
Использование конструкторов настроек ... 223
Отчет в виде таблицы ... 227
Отчет в виде диаграммы ... 231
Формирование отчета по поступлениям с итоговой информацией ... 235
Формирование в отчете двух таблиц ... 238
Отчет по документам *Заявка* ... 238
Отчет по заявкам в интервале дат ... 242
Отчеты по информационной базе *главы 3* ... 247
Отчет по квитанциям ... 247
Отчет по итоговой оплате договоров ... 250
Отчет по квитанциям с группировкой ... 253
Отчеты по информационной базе *главы 1* ... 255
Отчет по учащимся ... 255
Отчет по курсам ... 258
Итоги ... 260

Глава 6. Программирование форм в прикладных задачах ... 261
Программная организация форм ... 261
Учет начислений преподавателям ... 262
Информация о тарифах ... 263
Вспомогательные справочники ... 265
Документ *Учет* ... 267
Процедуры обработки событий на форме документа ... 268
Просмотр актуальных тарифов ... 273
Оборотный регистр накопления ... 275
Движения по регистру *Начисление* ... 277
Просмотр информации по начислениям ... 279
Программное добавление элементов в справочник ... 282
Подготовка конфигурации ... 282
Разработка сервиса для просмотра ... 284
Учет бонусов и штрафов при организации оплаты ... 287
Справочник *Организации* ... 288
Справочник *Учащиеся* ... 289
Регистр накопления по оплате ... 291
Документ *Оплата* ... 293

Конструктор движений ... 294
Планы по доработке формы документа ... 295
Программирование формы документа ... 297
Документ *Возврат* ... 302
Просмотр прихода в документе *Возврат* ... 303
Отчет по оплатам ... 306
Итоги ... 309

Приложение. Описание прилагаемого компакт-диска ... 311

Литература ... 313

Предметный указатель ... 315

Введение

В настоящее время информационные ресурсы широко используются в деятельности практически всех организаций разного профиля. Это приводит к повышению потребности в специалистах, которые профессионально владеют разнообразными современными компьютерными технологиями. При этом среди наиболее частых требований и пожеланий со стороны работодателей — навыки работы с программным продуктом 1С:Предприятие. Эта информационная система является самой популярной на территории нашей страны программой автоматизации различных участков офисной работы. Дело в том, что система 1С:Предприятие позволяет существенно повысить эффективность экономической и организационной деятельности предприятий.

Если посмотреть на спектр специалистов, работающих с рассматриваемым программным продуктом, то мы увидим, что он достаточно широк — менеджеры, бухгалтеры, программисты, разработчики, руководители подразделений и т. д. Всех их можно разделить на две условные категории — разработчиков и пользователей.

Первые создают прикладные разработки для дальнейшего использования пользователями в конкретной области деятельности (бухгалтерия, торговля, склад, управление персоналом и др.). Для этого в систему 1С:Предприятие заложены разнообразные ресурсы, важное место среди которых занимают *визуальные средства разработки* и *встроенный программный язык*. И построение любого прикладного решения на платформе 1С:Предприятие связано с их использованием.

Другая, бóльшая по численности категория специалистов, связанных с рассматриваемым программным продуктом, — пользователи. Это те, кто работает с готовыми разработками (с тем, что уже создано разработчиками). И часто у пользователя, имеющего некоторый опыт работы с одним из прикладных решений на платформе 1С:Предприятие 8, создается иллюзия, что он уже овладел всеми секретами мастерства. Однако для внесения даже небольших изменений в готовое решение (которые часто требуются в связи с тем, что в работе каждой организации есть свои специфические особенности) необходимы другие знания и практические навыки — представление об объектах, составляющих систему 1С:Предприятие, знание встроенного языка программирования, опыт разработчика и многое другое. Перед пользователями, овладевшими этими знаниями, открываются широкие возможности как для внесения изменений в готовые решения, так и для создания собственных приклад-

ных разработок. Именно для читателей, которые на практических примерах хотят быстрее овладеть секретами мастерства разработчика, написана эта книга.

К настоящему времени выпущено несколько версий платформы 1С:Предприятие 8. Последняя на сегодняшний день — 1С:Предприятие 8.2. Она отличается наиболее принципиальными изменениями архитектуры платформы. Эти изменения включают конструирование управляемого интерфейса, разработку управляемых форм, использование новых возможностей настройки отчетов, настройку командного интерфейса и ряд других ресурсов.

Один из первых вопросов, который сразу возникает, — это вопрос об уровне предварительных знаний для успешного усвоения информации, представленной в книге. Практические занятия с различными категориями учащихся показали, что освоение приведенных в книге разработок вполне по силам и тем, кто не имеет навыков в области программирования, и тем, кто до этого не сталкивался с системой 1С:Предприятие. Приведенные в книге примеры использовались в реальном учебном процессе. На этих занятиях присутствовали учащиеся как различных возрастных категорий, так и с различным уровнем начальной подготовки.

Практика показала, что приобрести навыки для самостоятельных разработок на платформе 1С:Предприятие 8.2 можно за относительно небольшой срок.

Общий взгляд на систему 1С:Предприятие

Программный продукт 1С:Предприятие позволяет существенно автоматизировать деятельность как определенного подразделения, так и всей организации в целом.

Основной отличительной чертой рассматриваемой системы является ее *способность подстройки* под конкретную область деятельности. Это достигается за счет *конфигурируемости*, которая заложена в данный программный продукт. Вообще, функционирование системы разделяется на два процесса — конфигурирование (работа в режиме **Конфигуратор**) и непосредственно пользовательскую работу с тем, что создано на этапе конфигурирования.

В режиме **Конфигуратор** создаются необходимые для решения прикладной задачи *объекты* (точнее — объекты конфигурации). При этом определяются формы ввода информации, алгоритмы обработки данных, а также необходимые пользователю отчеты. В целом результатом конфигурирования является конкретная *конфигурация*, которая отражает модель предметной области определенного подразделения или целой организации.

Сама по себе деятельность разработчика в конфигураторе может быть разделена на две составляющие. Как и в любой другой современной инструментальной программной среде, в системе 1С:Предприятие 8.2 большое место занимают визуальные средства разработки. Они предлагают работу с диалоговыми окнами, размещение необходимых элементов управления и задание их свойств. В этом случае практическая сторона при разработке фактически сводится к управлению объектами с помощью мыши, а также к формированию необходимой информации в полях вво-

да, флажках, переключателях и т. д. Вторую же (более трудоемкую) составляющую работы в режиме **Конфигуратор** представляет непосредственное программирование на встроенном языке.

При создании конфигурации, направленной на решение конкретной прикладной задачи, разработчик использует такие заложенные в систему объекты конфигурации, как справочник, документ, форма, регистр сведений, регистр накопления и многие другие. На базе указанных объектов разработчик для определенной прикладной области создает уже специфические объекты, составляющие разрабатываемую конфигурацию. Например, это могут быть справочники "Номенклатура" или "Сотрудники", первый из которых предназначен для хранения информации о товарах, а второй — для представления информации о лицах, работающих в организации.

Аналогично, используя базовый объект конфигурации — *документ*, можно, например, разработать прикладной документ "Приходная накладная".

Стоит заметить, что в конфигураторе создаются не конкретные документы, справочники либо отчеты, а определяется лишь их вид или, можно сказать, структура (еще лучше подходит слово шаблон). Например, в режиме **Конфигуратор** можно определить, что в создаваемом документе должны присутствовать следующие поля:

- название склада;
- название фирмы-поставщика;
- список товаров с указанием количества.

После создания подобной структуры в работу может включаться пользователь, который в режиме **1С:Предприятие** вносит данные в создаваемые системой документы, имеющие разработанную структуру. Таким образом, для сотрудников, не знакомых с конфигуратором и технологией создания объектов конфигурации, видна только пользовательская составляющая деятельности.

Важно отметить, что система 1С:Предприятие позволяет создавать программные объекты, которые являются очень близкой моделью реальных, используемых в конкретной сфере деятельности. В этом случае пользователи, впервые столкнувшиеся с этой информационной системой, отмечают удобство при переходе от бумажной документации к электронной.

Фактически для последующей работы пользователя в режиме **1С:Предприятие** разработчиком в режиме **Конфигуратор** создается необходимая схема (алгоритм действий). Результаты конфигурирования обеспечивают сотрудникам организации комфорт при работе с информационной базой, созданной системой на основании разработанной конфигурации.

В качестве технологических средств разработки ранее был отмечен встроенный язык программирования, заложенный в рассматриваемую систему. Это наиболее важный ресурс для разработчика, но не единственный. Приведем еще несколько наиболее востребованных.

В системе 1С:Предприятие присутствует большое количество конструкторов — инструментов, которые позволяют разработчику избежать выполнения разнообраз-

ных рутинных действий. В частности, конструкторы предоставляют возможность автоматически создавать необходимые формы для ввода данных, а также печатные формы. И главное, что все это происходит в результате несложного диалога, в результате чего и формируется программный код.

Также в программный продукт 1С:Предприятие заложен механизм запросов, который в основном используется для построения разнообразных отчетов, являющихся непременным атрибутом автоматизации в любой предметной области. Механизм запросов основан на еще одном языке, заложенном в систему, — *языке запросов*, который обеспечивает выборку интересующей информации из базы данных.

Имеющийся текстовый редактор используется для создания программных модулей на встроенном языке. Для удобства разработчика в этом редакторе заложено контекстное выделение цветом синтаксических конструкций. Кроме того, при работе программист может пользоваться подсказками и шаблонами.

Для создания макетов выходных форм в системе 1С:Предприятие предусмотрен редактор табличных документов. Формат табличного документа является универсальным для документов, справочников, отчетов и других объектов.

Средством, которое существенно облегчает труд разработчика, является *отладчик*. Он позволяет отслеживать выполнение программных модулей, замерять время выполнения и просматривать содержимое переменных.

Этот перечень составляет только часть многочисленных технологических средств, имеющихся в рассматриваемой информационной системе. Для получения исчерпывающей информации следует обратиться к справочной документации, прилагаемой к программному продукту 1С:Предприятие 8.2.

Краткое содержание книги

Книга состоит из шести глав, которые наполнены разнообразными практическими примерами. Организация материала обусловлена желанием сделать восприятие информации удобным для читателя. Необходимые справочные сведения по используемым визуальным средствам и встроенному языку программирования приводятся по ходу изложения примеров. Наилучший эффект при чтении книги будет достигаться при параллельном выполнении приведенных разработок на компьютере. Именно для этого все технические шаги в книге описаны достаточно подробно.

В *главе 1* читатели познакомятся с такими основными объектами конфигурации, как *перечисление*, *справочник* и *документ*. На практических примерах будет рассмотрена технология разработки электронных форм, являющихся обязательным атрибутом интерфейса пользователя. Первые шаги в плане программирования будут выполнены с использованием конструкторов — замечательных средств для облегчения труда разработчика. После этого самостоятельное программирование (без использования конструкторов) начнется с небольших процедур, которые будут понятны читателям, не встречавшимся до этого с системой 1С:Предприятие 8.2 либо работавшим в ней только в качестве пользователей. Вы увидите, как легко с по-

мощью несложного программного кода можно обеспечить необходимый сервис для последующей пользовательской работы сотрудников организации.

В целом стратегическая цель *главы 1* заключается в том, чтобы заложить у читателя базовые навыки по работе с конфигуратором, который представляет собой один из режимов работы системы 1С:Предприятие. Результатом труда разработчика в этом режиме является конфигурация, с которой пользователь работает в режиме **1С:Предприятие** (это другой важный режим работы данной системы).

В *главе 2* рассматриваются приемы работы с документами и регистрами накопления. На практических примерах разбирается процесс проведения документов и организация движений по регистрам.

В *главе 3* демонстрируются возможности относительно нового объекта системы 1С:Предприятие 8 — регистра сведений (в очень популярной версии 7.7 он отсутствовал). В данной главе рассматривается создание конфигурации, связанной с решением задачи, которая возникла в ходе управления деятельностью подразделением высшего учебного заведения. Эта разработка касается учета оплаты за обучение.

В целом рассмотренный пример конфигурации показывает, как легко с помощью разнообразных объектов системы, их свойств и методов можно создавать удобные пользовательские решения для автоматизации офисной работы.

Глава 4 целиком отведена *языку запросов*, который позволяет обеспечить удобный доступ к имеющимся в системе данным. Причем содержание главы не представляет собой чисто справочную информацию: для каждой синтаксической конструкции языка запросов приводятся примеры, результаты, выполнения которых продемонстрированы иллюстрациями, отражающими работу в режиме **1С:Предприятие.**

В *главе 5* рассматривается *система компоновки данных*. Показано, каким образом без знания программного кода можно эффективно создавать разнообразные отчеты. Система компоновки данных — это новое средство, предназначенное для создания отчетов на основе декларативного описания. Механизм компоновки данных позволяет дать целостное описание отчета, используя только визуальные средства.

В *главе 6* на примерах демонстрируются разнообразные возможности платформы для программирования форм и расположенных на них элементах. На практических примерах показано, каким образом можно обеспечить необходимый сервис для работы пользователя.

Компакт-диск

Прилагающийся к книге компакт-диск содержит все разработки, приведенные в книге. Основная составляющая диска — это различные конфигурации, описание разработки которых и составляет содержательную часть издания. Кроме того, в *главе 3* рассматривается пример, касающийся интеграции работы системы 1С:Предприятие 8.2 и популярного офисного приложения Microsoft Word. Упоминаемые в издании автоматизированные документы Microsoft Word также содержатся на компакт-диске. Все файлы распределены по тематическим папкам, а подробная инструкция работы с ними приводится в конце книги.

От автора

Во-первых, мне бы хотелось выразить благодарность всем читателям, которые познакомились с моей книгой. Издания по тематике, связанной с офисным программированием ([9—12]) были отмечены положительными отзывами со стороны читателей. Фактически это явилось для меня поводом подготовки новой книги, касающейся на этот раз платформы 1С:Предприятие 8.2.

При подготовке нового издания большую помощь мне оказали супруга и дочь (это касается подготовки многочисленных представленных примеров).

Большую благодарность хочу выразить издательству "БХВ-Петербург" за сотрудничество в плане подготовки к изданию моих книг на протяжении последних лет. Особенно хочу поблагодарить Рыбакова Евгения Евгеньевича за ценные советы.

Что касается тематики разработок, приведенных в этой книге, то все они связаны с Институтом дистанционного обучения, который входит в состав Нижегородского государственного лингвистического университета им. Н. А. Добролюбова. Директору Института дистанционного обучения Александру Николаевичу Сверчкову и заместителю директора Юрию Васильевичу Воронкову я также хочу выразить свою признательность. Значительная часть материала появилась как результат их эффективного управления организацией учебного процесса. Поставленные ими задачи фактически явились фундаментом данного издания.

В заключение еще раз замечу, что наибольший эффект будет наблюдаться в том случае, если вы параллельно с чтением будете выполнять практические примеры непосредственно в системе 1С:Предприятие 8.2.

И если, несмотря на все старания, в книге все же обнаружатся неточности, то заранее приношу читателям свои извинения.

Связаться с автором книги можно по электронной почте **mail@bhv.ru**. Также вы можете обратиться по адресу **www.bhv.ru**, чтобы выразить свое отношение к книге.

ГЛАВА 1

Знакомство с системой 1С:Предприятие 8.2

В настоящее время популярность программного продукта 1С:Предприятие 8 стремительно растет и, соответственно, увеличивается потребность в специалистах, знакомых с этой информационной системой. Часто уровень квалификации в данной области является определяющим фактором при подборе кандидатуры на то или иное вакантное место. При этом в большинстве ситуаций от потенциального сотрудника требуются, в основном, навыки уверенного пользователя при работе с одной из наиболее популярных на сегодняшний день конфигураций системы 1С:Предприятие 8. В этом случае специалист занимается профессиональной деятельностью в своей предметной области (например, бухгалтерия или сфера складского учета) и использует в работе современные технологии автоматизации. А большинство сложных задач, связанных с внесением изменений в имеющуюся конфигурацию (например, разработка нового документа или отчета), решает, как правило, программист или подразделение программистов.

Распространена и другая ситуация. Так, в небольших организациях пользователям и программистам, которые не сталкивались до этого с программными продуктами фирмы 1С, приходится самостоятельно решать многие практические вопросы, связанные с внесением изменений в имеющуюся конфигурацию. Для *этой категории читателей* мы и попробуем на примерах разобрать основные приемы, касающиеся программирования и визуальных ресурсов для разработки прикладных решений на платформе 1С:Предприятие 8.2.

На сегодняшний день фирмой 1С выпущено несколько версий программного продукта 1С:Предприятие 8 — это 8.0, 8.1 и 8.2, которая является последней на данный момент. При этом продукт 1С:Предприятие 8.2 представляет собой *наиболее принципиальное изменение архитектуры платформы* 1С:Предприятие 8, начиная с момента ее выхода.

> ***Примечание***
>
> На протяжении последних лет специалистами фирмы 1С выпущено большое количество изданий [1—8], адресованных как профессиональным разработчикам, так и начинающим программистам по разработке прикладных решений на платформе 1С:Предприятие 8.

Учитывая качественную новизну платформы 1С:Предприятие 8.2 большая часть приемов разработки, характерных для версий 8.0 и 8.1, уже не подходит при создании прикладных решений на новой платформе. В связи с этим для читателей полезно рассмотреть технологические ресурсы для создания разработок в системе 1С:Предприятие 8.2, начиная с самых первых практических шагов. На протяжении глав книги именно это и будет сделано на многочисленных примерах разработок прикладных решений.

Будет правильно, если мы начнем с рассмотрения достаточно простых ситуаций, которые должны быть понятны читателям, начинающим знакомиться со средствами разработки, имеющимися в программных продуктах фирмы 1С. Разумеется, при этом сразу возникает вопрос: а что нужно предварительно знать для успешного восприятия изложенного в книге материала? Конечно, желательны навыки в области программирования, а также приветствуется любой опыт работы с информационной системой 1С:Предприятие. Все это придаст читателю больше уверенности при работе с книгой, увеличит скорость рассмотрения примеров и позволит ему быстрее перейти к решению собственных задач. Однако можно обойтись и без этого.

Главное, что необходимо, — это наличие на своем компьютере установленной программы 1С:Предприятие 8.2. Для той категории читателей, которая не располагает данной программой, хочу порекомендовать *учебную версию* 1С:Предприятие 8.2. Она существенно дешевле полноценного варианта, но при этом поддерживает практически все функциональные возможности.

При наличии программного продукта 1С:Предприятие 8.2 оптимальная технология работы с книгой заключается в параллельном выполнении рассматриваемых разработок на своем компьютере. Понятно, что объем справочной информации по программированию в среде 1С:Предприятие 8.2 огромен, и уже в середине этой главы нам потребуется познакомиться с рядом справочных сведений. В процессе изложения материала книги подробная информация будет приводиться по мере необходимости, что весьма удобно для читателей, поэтому рассмотрение примеров не предполагает использования дополнительных источников информации. Хотя если после прочтения книги вы захотите познакомиться с более сложными ресурсами системы 1С:Предприятие 8.2, то от себя хочу порекомендовать ряд изданий специалистов фирмы 1С [6—8].

Для работы нам не потребуется наличие стандартных конфигураций, разработанных для платформы 1С:Предприятие 8.2, — мы будем создавать свои собственные и на их примере рассматривать различные технические приемы, необходимые для выполнения прикладных разработок. Конечно, эти конфигурации не будут такими сложными, как разработки специалистов фирмы 1С, однако итогом рассмотрения примеров для читателей станут навыки, которые они будут использовать в практической работе с данной системой. После рассмотрения приводимых конфигураций читатели смогут создавать собственные разработки, способные автоматизировать деятельность их организаций.

Архитектура системы 1С:Предприятие

В системе 1С:Предприятие можно выделить две основные составляющие:

- технологическую платформу;
- прикладные решения автоматизации различных участков деятельности, которые создаются с помощью технологической платформы.

Подобная организация системы обеспечивает достаточную гибкость прикладных решений. В этом случае программист, даже не участвовавший в создании конкретной разработки, располагает всеми необходимыми ресурсами для внесения в нее изменений. Следует заметить, что в настоящее время имеется большое количество стандартных и узкоспециализированных конфигураций, которые постоянно дорабатываются и совершенствуются. Такая открытость системы для возможной модификации делает ее очень удобной для программистов, поскольку они могут дорабатывать и развивать существующие типовые разработки.

Ничто не мешает каждой организации, располагающей небольшим составом программистов (даже одним специалистом), настроить большинство стандартных прикладных решений под себя. В этом случае не требуется использовать какие-либо отдельные программные продукты — все средства разработки входят в состав технологической платформы.

В технологической платформе выделяются две составляющие:

- среда исполнения;
- среда разработки.

Среда исполнения называется режимом **1С:Предприятие**, а среда разработки открывается пользователю при запуске системы в режиме **Конфигуратор**.

В среде разработки используется технология *метаданных*, которые представляют собой структуру объектов, описывающую конкретное прикладное решение. Среда исполнения выполняет (проигрывает) метаданные. Здесь можно провести некую аналогию с любой системой, выполняющей программный код.

При работе с метаданными доступно широкое использование визуальных средств разработки. В этом случае программисту не требуется писать программный код для добавления нового объекта метаданных. Все выполняется с помощью щелчков мыши и установки необходимых параметров в списках, переключателях и других элементах управления. Результат же этих действий автоматически трансформируется системой в программный код.

Работа программиста в среде разработки приводит к построению конкретной конфигурации. И такая конфигурация реализуется (проигрывается) в среде выполнения (в режиме **1С:Предприятие**). Важно заметить, что платформа 1С:Предприятие содержит фиксированный набор *основных объектов конфигурации*. Условно их можно назвать прототипами. Например, имеются прототипы справочника, документа, регистра сведений, отчета и еще ряд других. Когда в среде разработки создается новый объект конфигурации, то он наследует используемый при его по-

строении прототип (разрабатываемый объект конфигурации строится на основе одного из основных объектов).

Таким образом, несмотря на то, что каждая прикладная конфигурация обладает собственной индивидуальностью, она создана на основании заранее разработанных специалистами фирмы 1С объектов конфигурации.

Встроенный язык программирования системы 1С:Предприятие, а также язык запросов используются для описания специфических алгоритмов прикладного решения. Встроенный язык имеет много общих черт с другими языками программирования, такими как Visual Basic, JavaScript и др. Для построения запросов в систему 1С:Предприятие включен еще один язык — язык запросов, который похож на известный (по работе с базами данных) язык построения запросов SQL. Важной особенностью языка запросов в системе 1С:Предприятие является то, что он представляет собой доступ к данным только на чтение. Для записи же информации используются интерактивные средства и программные методы объектов системы 1С:Предприятие.

При создании прикладных решений применяются как визуальные средства разработки, так и программирование на встроенном языке.

Заметим, что любое новое прикладное решение — это совокупность метаданных. Однако когда вы начинаете новую разработку, то начинаете ее не с нуля. Платформа при создании новой конфигурации автоматически создает некую базовую структуру метаданных. Она уже сама по себе представляет собой разработку, которая может быть выполнена на платформе 1С:Предприятие. Конечно, практической пользы от этого нет, т. к. необходимой функциональности в нее еще не заложено. Задача разработчика как раз и заключается в том, чтобы эту функциональность создать путем доработки структуры метаданных. В дальнейшем после такой доработки при переходе в пользовательский режим работы (режим **1С:Предприятие**) система увидит эти изменения в структуре метаданных и самостоятельно реализует необходимую функциональность.

Набор основных объектов метаданных ограничен (справочник, документ, регистр накопления и еще ряд других), и *разработчик не может создать свой, принципиально новый объект конфигурации*. Он может только добавлять в разрабатываемое прикладное решение новые объекты конфигурации, которые базируются на уже имеющихся основных объектах конфигурации.

Разумеется, практические задачи требуют разработки различных алгоритмов. Эти алгоритмы являются специфическими при решении каждой задачи, и для реализации алгоритмов предназначен встроенный язык системы 1С:Предприятие. Размещение текстов программ на встроенном языке производится в различных модулях (в книге мы встретимся с модулями форм, модулями команд, модулями менеджеров объектов и рядом других). Платформа 1С:Предприятие 8.2 построена так, что большинство программных модулей вызываются в определенные моменты работы прикладного решения.

Постановка задачи

В этой главе мы создадим несколько прикладных объектов конфигурации для решения несложной практической задачи, после чего поработаем в качестве пользователей с информационной базой, построенной системой на основании разработанной конфигурации. Все технические моменты, связанные с работой в конфигураторе, будут подробно комментироваться, а читатель легко сможет повторить описанные действия на своем компьютере. При желании можно обратиться к компакт-диску, прилагаемому к книге, где приведена разработанная в главе конфигурация (как, впрочем, и все другие конфигурации, описываемые в книге).

Условимся считать, что наша организация предлагает платные услуги, связанные с учебно-преподавательской деятельностью. Это может быть одно из подразделений в высшем или среднем учебном заведении, а также просто учебный центр, занимающийся коммерческой деятельностью. Мы, как разработчики, попробуем помочь нашим сотрудникам в автоматизации некоторых участков работы с учащимися (слушателями курсов), для обозначения которых будем использовать более общее понятие — "клиенты". Каждый клиент может записаться на тот или иной платный курс, после чего внести плату за обучение (сразу либо частично), а затем начать посещать занятия. Понятно, что руководство учебного центра должно максимально эффективно управлять данным процессом.

В целом наша задача выглядит так: используя возможности платформы 1С:Предприятие 8.2 необходимо построить информационную систему для внесения, хранения и извлечения данных, которая позволит автоматизировать работу персонала учебного центра. Для этого нам потребуется создать объекты конфигурации, которые будут отражать реальную ситуацию в организации учебного процесса. Фактически мы создадим модель предметной области, которую затем реализуем на практике, используя возможности платформы 1С:Предприятие 8.2.

Все действия будут выполняться в конфигураторе, который, как уже говорилось, является одним из режимов работы информационной системы 1С:Предприятие. По сути, в конфигураторе и были созданы все известные прикладные разработки фирмы 1С для платформы 1С:Предприятие.

Окно запуска

Итак, после выбора в стартовом меню Windows раздела **1С Предприятие 8.2** и далее пункта **1С Предприятие** перед нами открывается стартовое окно информационной системы 1С:Предприятие 8.2 (рис. 1.1).

В центральной части данного окна перечислены уже имеющиеся информационные базы, каждая из которых представляет собой конкретную разработку, направленную на автоматизацию участка работы, либо определенного подразделения, либо организации в целом. В нашей ситуации мы начинаем решение новой прикладной задачи, поэтому от нас потребуется создание новой информационной базы. Эта база будет построена на основе разрабатываемых далее объектов конфигурации.

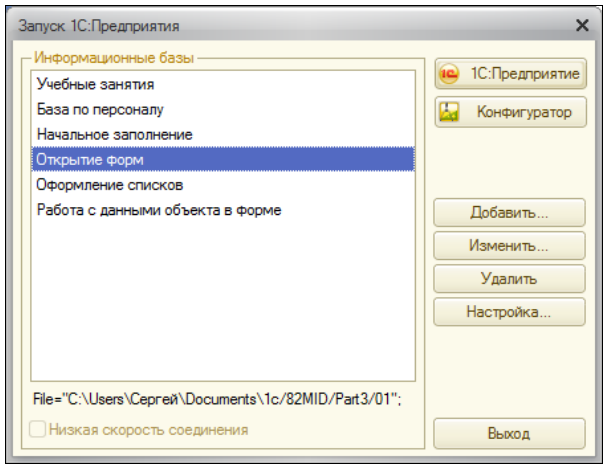


Рис. 1.1. Стартовое окно информационной системы 1С:Предприятие 8.2

Примечание

В системе 1С:Предприятие возможны два режима работы — режим **Конфигуратор** и режим **1С:Предприятие**. Режим **1С:Предприятие** для пользователей является основным, и в нем они вводят информацию в систему, получают необходимые отчеты и т. д. Режим **Конфигуратор** используется разработчиками и программистами. В этом режиме производится создание новых прикладных решений, а также изменение уже имеющихся. Фактически в данной книге большее внимание уделяется как раз режиму конфигуратора.

Создание новой информационной базы

Технические действия для создания новой информационной базы достаточно просты. Сначала следует щелкнуть по кнопке **Добавить** (см. рис. 1.1), что приведет к появлению на экране следующего окна (рис. 1.2), в котором необходимо выбрать один из двух сценариев дальнейшего развития событий. Мы пойдем по пути разработки новой информационной базы и, нажав кнопку **Далее**, перейдем к окну, в котором следует задать способ создания будущей разработки (рис. 1.3). Здесь перед нами открывается возможность формирования новой информационной базы на основе одного из имеющихся шаблонов, однако сейчас мы поступим по-другому и создадим пустую информационную базу.

Примечание

Шаблоны, присутствующие в окне на рис. 1.3, будут отражены, если они предварительно установлены на компьютере. Например, к книгам [6,7] приложены компакт-диски с шаблонами конфигураций для платформы 1С:Предприятие 8.2, которые вы можете самостоятельно установить на свой компьютер. В этом случае перед вами открывается возможность начать разработку не с нуля, а с заготовки, созданной специалистами.

Очередное окно данного диалога показано на рис. 1.4. Здесь необходимо дать информационной базе имя, и в нашем случае вместо варианта, предложенного систе-

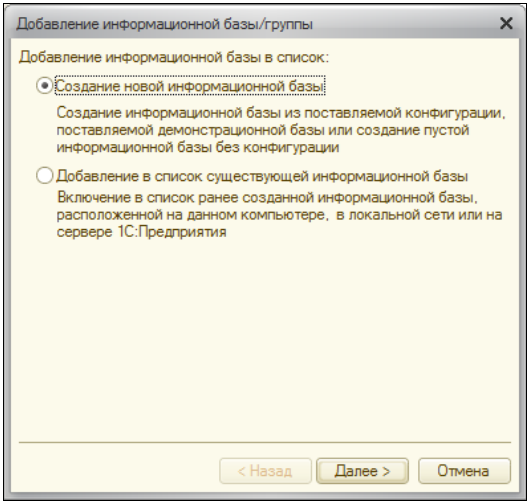


Рис. 1.2. Окно для добавления новой (существующей) информационной базы

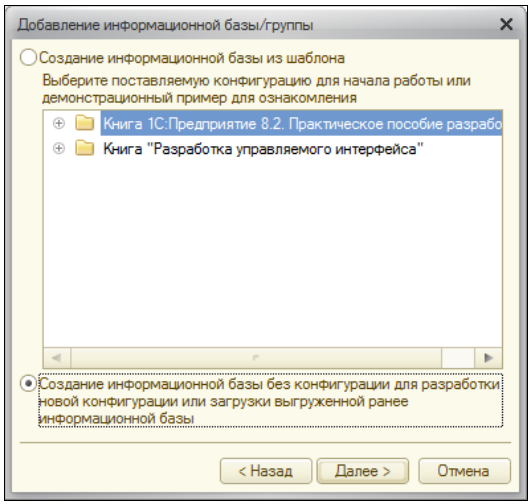


Рис. 1.3. Окно выбора способа создания информационной базы

мой по умолчанию, лучше установить свое название, например `Глава 1`. Если этот вариант не нравится, то можно выбрать любое другое словосочетание.

После щелчка по кнопке **Далее** перед нами открывается следующее окно (рис. 1.5), в котором требуется указать каталог, где наша разработка будет располагаться. Для этого необходимо создать пустую папку в любом месте на компьютере либо в сети, а в диалоговом окне указать ее месторасположение. В этой папке и будет храниться наша конфигурация, которую мы разработаем в данной главе.

Теперь после щелчка по кнопке **Далее** на экране отображается окно (рис. 1.6) для установки параметров и режима запуска. Здесь относительно установок, сделанных

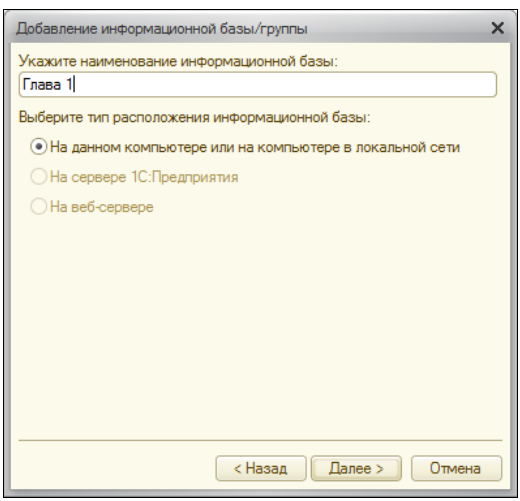


Рис. 1.4. Окно для указания наименования создаваемой информационной базы

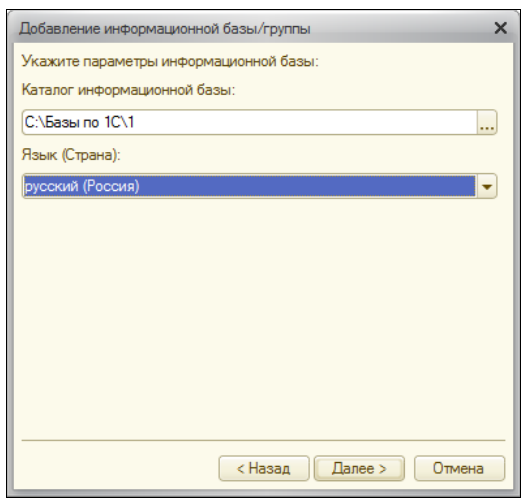


Рис. 1.5. Окно для указания каталога создаваемой информационной базы

системой по умолчанию, мы ничего менять не будем и просто щелкнем по кнопке **Готово**.

В результате проделанных действий на экране повторно появляется окно, которое мы ранее видели (см. рис. 1.1), но уже с новой строкой, содержащей название только что созданной информационной базы (рис. 1.7). Таким образом, все предварительные шаги выполнены, и мы можем перейти к работе по разработке конфигурации. Для перехода в режим конфигуратора предназначена кнопка **Конфигуратор**, которая позволяет перейти к созданию объектов конфигурации. На основании разработанных объектов конфигурации системой 1С:Предприятие 8.2 будет построена информационная база данных, с которой уже может работать пользователь в режиме **1С:Предприятие**.

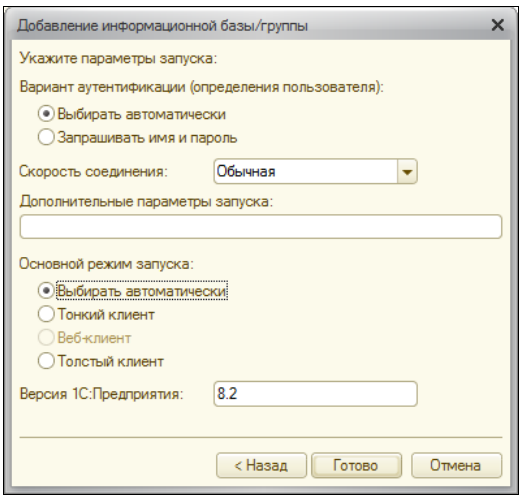


Рис. 1.6. Окно для указания параметров и режима запуска создаваемой информационной базы

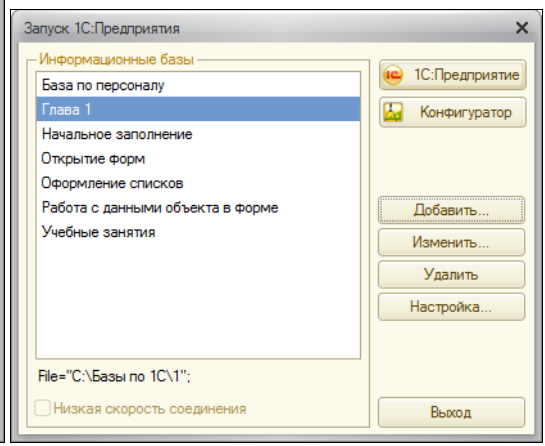


Рис. 1.7. Включение базы **Глава 1** в список информационных баз

Примечание

В дальнейшем (при последующих сеансах работы на компьютере), когда вы решите продолжить созданную разработку, от вас потребуется лишь выбрать название информационной базы в списке (см. рис. 1.7) и нажать кнопку **Конфигуратор** или кнопку **1С:Предприятие**. Вторая из этих кнопок вызывает режим **1С:Предприятие**, предназначенный для непосредственной работы по внесению данных в информационную базу.

Итак, в настоящий момент мы остановились на том, что с помощью кнопки **Конфигуратор** (см. рис. 1.7) открыли окно режима с аналогичным названием (рис. 1.8). Это и есть та инструментальная среда, в которой будем создавать объекты конфигурации. Важно заметить, что все прикладные решения для автоматизации различных участков деятельности созданы разработчиками именно с использованием конфигуратора.

Следующее, что от нас потребуется, — это выбор в меню **Конфигурация** (см. рис. 1.8) пункта **Открыть конфигурацию**. В результате на экране появится

окно с *деревом объектов конфигурации* (рис. 1.9). Дерево объектов конфигурации является одним из основных инструментов, с которым работает разработчик. Вся основная информация о содержимом конфигурации отражена в дереве объектов конфигурации. Сейчас в нем присутствуют только заголовки, а в дальнейшем создаваемые нами объекты конфигурации будут автоматически отображаться в этом дереве.

Примечание

Каждая ветвь древовидной структуры описывает определенную тематическую группу конфигурации. Например, при создании новых *справочников* они будут группироваться в ветви **Справочники**, а в ветви **Документы** найдут свое место объекты конфигурации типа *документ*. Технология работы с деревом конфигурации аналогична работе с подобными древовидными представлениями в операционной системе Microsoft Windows. Для открытия той или иной ветви (например, для обзора имеющихся в конфигурации справочников или документов) достаточно щелкнуть по обозначению соответствующего знака "плюс".

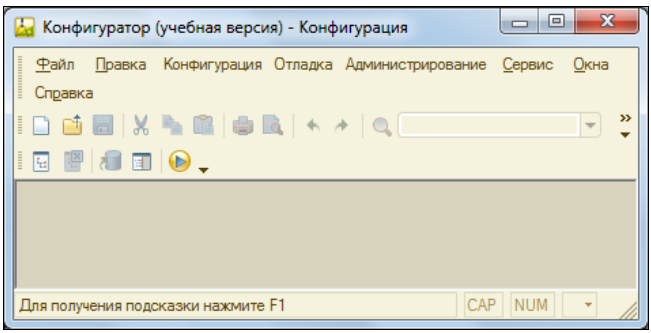


Рис. 1.8. Окно системы 1С:Предприятие 8.2 в режиме конфигуратора

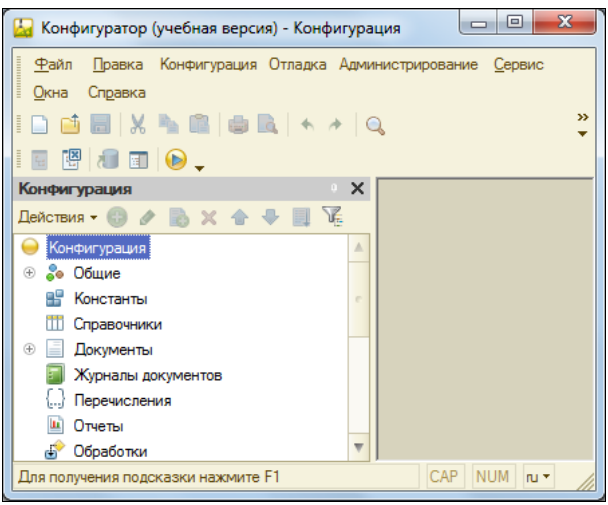


Рис. 1.9. Окно, включающее дерево объектов конфигурации

Назначение объектов конфигурации

В системе 1С:Предприятие имеются *базовые* объекты конфигурации, которые созданы разработчиками фирмы 1С для того, чтобы обеспечить решение любой задачи по автоматизации учета. Так, любая офисная деятельность связана с перечнями данных, с документами, отчетами, механизмами количественного учета и т. д. Чтобы разработчику эффективно и стандартизированно обеспечить решение прикладной задачи в системе 1С:Предприятие, имеются *базовые объекты конфигурации*. Их много, перечислим ряд объектов, с которыми мы встретимся в ближайших разделах книги:

- перечисление;
- справочник;
- документ;
- отчет;
- регистр накопления;
- регистр сведений.

Данные объекты конфигурации условно можно назвать "заготовками" или "деталями", на основании которых строятся прикладные объекты конкретной конфигурации. Так, используя базовый объект конфигурации *справочник*, можно в режиме конфигуратора разработать конкретный справочник "Номенклатура" (или другой справочник "Сотрудники"). При этом справочник "Номенклатура" будет включать как функциональность основного объекта конфигурации *справочник*, так и новые особенности (для справочников уникальность, как правило, заключается в их реквизитах). Также можно на основании основного объекта конфигурации *документ* разработать прикладной документ "Приходная накладная". Важно отметить: в слова *разработать* или *создать* вкладывается формирование структуры и обеспечение необходимой функциональности объекта конфигурации.

Далее платформа 1С:Предприятие 8.2 на основании созданной конфигурации формирует информационную базу, с которой может работать пользователь (добавлять в справочники конкретные элементы, формировать документы с заполненными полями и т. д.).

Следует отметить, что кроме перечисленных видов "сложных" объектов конфигурации (справочник, документ и т. д.) в системе 1С:Предприятие имеются и более простые объекты, входящие в состав сложных. Например, каждый реквизит справочника является, в свою очередь, также объектом конфигурации. В большинстве документов и ряде справочников (у элементов справочников) имеются табличные части, которые также представляют собой объекты конфигурации.

Таким образом, в целом любую конфигурацию можно представить в виде совокупности стандартных компонентов. Это очень удобно для поддержки разработок программистами, которые не участвовали в создании прикладных решений.

Перечисления

Среди базовых объектов конфигурации можно выделить категорию, отличающуюся высокой сложностью. Эти объекты являются *основными* объектами конфигурации. Первым из них, с которым мы познакомимся, будет *перечисление*. Создаваемые пользователями перечисления строятся на основании основного объекта конфигурации *перечисление.* В целом, перечисления представляют собой статические списки данных. Эти списки создаются на этапе разработки (в конфигураторе) и в дальнейшем просто используются без возможности их изменения или дополнения при работе в режиме **1С:Предприятие.**

В конфигурации перечисления, как правило, используются не самостоятельно, а в совокупности с другими типами данных. Так, перечисления применяются для ввода ограниченного количества значений реквизитов справочников и документов. Например, в справочнике "Клиенты" имеется поле **Статус**, в которое по логике работы можно вносить только одно из нескольких фиксированных значений. При этом никаких новых или произвольных данных в данном поле быть не должно (система не разрешит выполнить это пользователю). В результате для данной ситуации заранее созданное и заполненное значениями перечисление **Статус** подходит лучше всего.

> ***Примечание***
>
> Основная цель использования перечислений — исключить неоднозначность вводимой пользователем информации.

Первым объектом конфигурации, который мы создадим, будет перечисление с именем **Статус**, отражающее статус каждого учащегося. Использование здесь перечисления связано с тем, что статус учащегося не подразумевает какого-то большого разнообразия возможных значений. Кроме того, набор значений статуса учащихся, как правило, продолжительное время не изменяется (если происходит добавление какого-то нового значения в качестве статуса, то это случается очень редко). Фактически для подобных ситуаций основной объект конфигурации *перечисление* и был разработан программистами фирмы 1С. В нашей ситуации в качестве значений статуса учащегося примем следующие:

- Зачислен;
- Отчислен;
- Сертификат.

Перейдем теперь к техническим действиям, позволяющим реализовать сказанное на практике. В режиме конфигуратора для создания нового перечисления необходимо в дереве объектов конфигурации (см. рис. 1.9) сначала щелкнуть правой кнопкой мыши на элементе **Перечисления**. В результате на экране откроется контекстное меню, состоящее из единственного пункта **Добавить**, который и следует выбрать, что позволит перейти к работе по конструированию перечисления в *окне редактирования объекта конфигурации* (рис. 1.10). Оно предназначено для разработки и редактирования объектов конфигурации, которые создаются разработчиком на основании основных объектов конфигурации. В данном случае мы разраба-

тываем перечисление **Статус**, которое будет наследовать все свойства основного объекта конфигурации *перечисление*.

Примечание

Для выполнения определенного технического действия в системе 1С:Предприятие, как правило, существует несколько способов. Это весьма характерно для современных программных продуктов. В документации, входящей в комплект программного продукта 1С:Предприятие, а также в издании [6] приводятся различные способы для достижения целей, которые возникают в процессе разработки прикладных решений. Здесь же (и далее в книге) для выполнения определенного действия мы, как правило, будем ограничиваться одним из способов (который, по мнению автора, наиболее удобен).

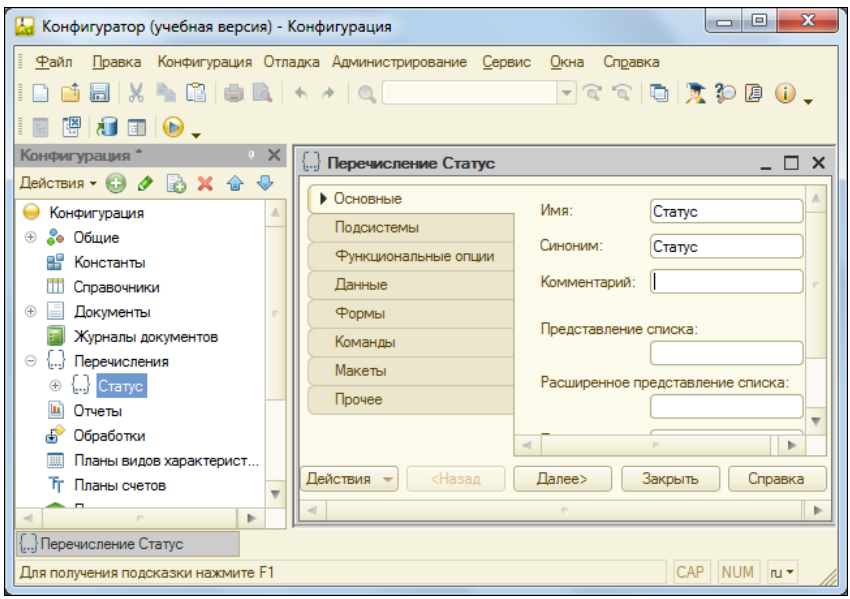


Рис. 1.10. Окно редактирования объекта конфигурации

Окно редактирования объекта конфигурации содержит несколько тематических вкладок, что обеспечивает удобство работы разработчика.

В системе 1С:Предприятие для редактирования свойств основных объектов конфигурации (а также для настройки взаимодействия между объектами) предназначено *окно редактирования объекта конфигурации*. Редактируемые свойства располагаются на нескольких вкладках, и каждая вкладка предназначена для настройки свойств конкретной категории. Сами вкладки (их название и содержание) меняются в зависимости от типа объекта конфигурации (так, вкладки, используемые для редактирования перечислений и, скажем, справочников отличаются). Все основные объекты конфигурации, с которыми мы встретимся в книге, содержат вкладку **Основные**. Поясним назначение полей этой вкладки.

- **Имя** — это ключевое поле, значение которого представляет собой идентификатор объекта конфигурации.

- **Синоним** — поле, используемое в различных ситуациях в режиме **1С:Предприятие** для обозначения такого названия объекта, которое понятно для пользователя. В частности, поле **Синоним** служит в качестве названия раздела меню, связанного с открытием конкретного справочника, документа, отчета и др. Дело в том, что на имя любого объекта накладывается ряд ограничений, например, в нем не должно быть пробелов. В этом случае при использовании сочетаний из нескольких слов имя объекта (например, документ "ПриходнаяНакладная") смотрится непривычно для пользователей. И чтобы при работе в режиме **1С:Предприятие** объект ассоциировался с названием в более традиционной форме, следует использовать раздел **Синоним** для замены имени в меню (на синоним подобных ограничений не накладывается).
- **Комментарий** — поле, которое носит вспомогательный характер и представляет собой произвольную строку. Смысл этого поля заключается в расшифровке и пояснении предназначения данного объекта. В последующих примерах заполнять поле комментария мы не будем. Однако в сложных разработках это имеет смысл делать, поскольку даже автор конфигурации может через какое-то время забыть собственную логику, используемую при разработке.

Сейчас от нас требуется на вкладке **Основные** внести значение **Статус** в поле **Имя**.

Таким образом, несложные действия привели к тому, что мы легко создали первый объект нашей конфигурации, который, правда, пока пуст. Для того чтобы внести в перечисление **Статус** необходимые значения (фиксированный перечень значений), следует в окне редактирования объекта конфигурации перейти на вкладку **Данные**. Для добавления очередного значения следует воспользоваться пиктографическим меню в верхней части окна, в частности, необходимо щелкнуть по кнопке, на которой изображен знак "плюс". В этом случае открывается еще одно важное окно для разработчика с палитрой свойств для поля **Зачислен** (рис. 1.11).

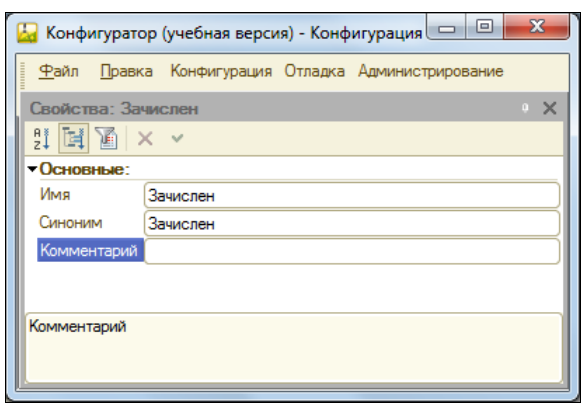


Рис. 1.11. Окно свойств поля **Зачислен**

Палитра свойств — это окно, содержащее набор свойств, которые можно определить для *выбранного* объекта конфигурации. При этом состав свойств зависит от

типа редактируемого объекта конфигурации (для справочников состав свойств один, для документов другой и т. д.).

> ***Примечание***
>
> Важно отметить, что значение перечисления также является объектом конфигурации, только этот объект относится к категории *подчиненных*. Подчиненный объект конфигурации зависит от основного.

Палитру свойств можно открыть и для основного объекта конфигурации. Для этого в дереве объектов конфигурации необходимо выделить интересующий объект и с помощью правой кнопки мыши вызвать контекстное меню, в котором следует обратиться к пункту **Свойства**. На рис. 1.12 показана палитра свойств созданного перечисления **Статус**.

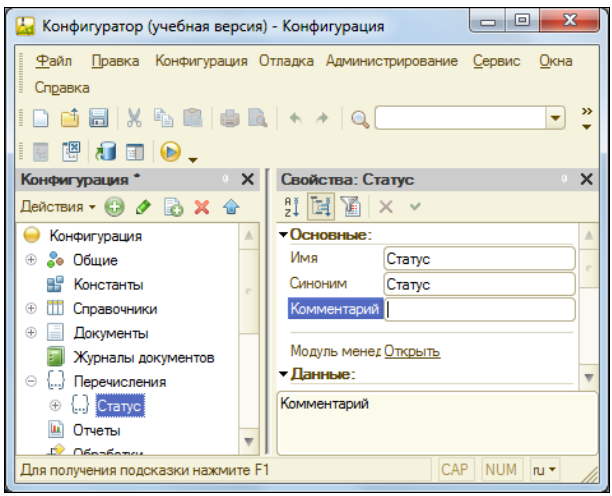


Рис. 1.12. Палитра свойств перечисления **Статус**

Вернемся к нашей разработке, в которой на вкладке **Данные** следует внести несколько значений перечисления **Статус** (рис. 1.13).

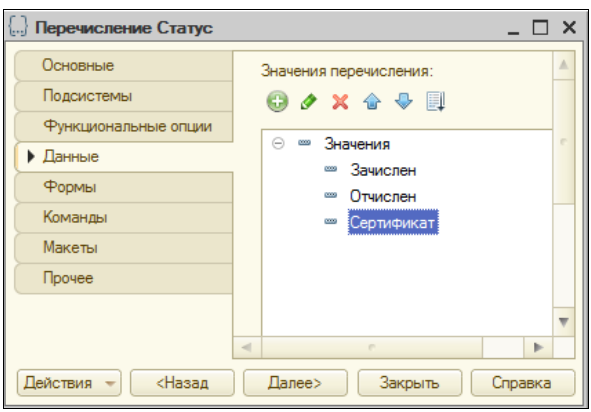


Рис. 1.13. Внесение значений перечисления **Статус**

В пиктографическом меню на вкладке **Данные** присутствуют кнопки, с помощью которых можно не только добавлять, но также корректировать, переставлять и удалять значения перечисления.

Обсудим теперь практическую сторону выполненных шагов. Так, у читателя наверняка возникает вопрос: как перечисление **Статус** использовать? В этой главе мы создадим еще один объект конфигурации — справочник "Клиенты", в котором введем реквизит с именем **Статус**. Он будет принимать варианты только из набора значений перечисления **Статус**, что позволит в этом случае пользователю избежать ввода других данных, кроме трех уже указанных.

> ***Примечание***
>
> В практической работе, кроме исключения ошибок, использование заранее определенных списков данных заметно ускоряет работу пользователя в режиме **1С:Предприятие**. Наряду с перечислениями для этого часто используются еще и справочники. Заметим, что справочники — это существенно более сложные (в плане программирования и функциональных возможностей) объекты по сравнению с перечислениями, и далее в этой главе мы с ними познакомимся.

В заключение этого раздела вернемся к двум инструментальным ресурсам, которые нам здесь встретились. Это окно редактирования объекта конфигурации и палитра свойств, позволяющие работать с объектами конфигурации. При этом окно редактирования объекта конфигурации используется в основном для создания новых объектов. Также оно удобно и при редактировании свойств уже существующего объекта.

Однако следует отметить, что окно редактирования объекта конфигурации является уникальным для каждого объекта и при переходе от одного объекта к другому окно редактирования объекта приходится открывать уже для другого объекта. В принципе никакого ограничения функциональности в этом нет и работать подобным образом вполне удобно. Альтернативой является палитра свойств, достоинство которой связано с тем, что она не привязана к определенному объекту конфигурации. Содержимое палитры свойств меняется в зависимости от того, какой объект вами выбран. Важно отметить, что при выборе в палитре свойств определенного свойства происходит его фиксация, и если мы выберем другой объект конфигурации, то данное свойство останется активным.

Справочники

Для работы с перечнями данных, которые могут формироваться и редактироваться пользователями в режиме **1С:Предприятие**, в системе имеется основной объект конфигурации под названием *справочник*. На основании этого объекта разработчик может создавать свои объекты конфигурации — справочники определенной прикладной направленности.

Система 1С:Предприятие по описанию объекта конфигурации (конкретного справочника) создает информационную структуру, в которую пользователь в режиме **1С:Предприятие** может вносить данные. Эту структуру также принято называть

справочник. Далее в зависимости от контекста должно быть понятно, о чем идет речь: об объекте конфигурации или об информационной структуре с данными.

Упрощенно справочник можно считать совокупностью строк, причем в каждой строке присутствует несколько колонок (количество колонок одинаково для каждой строки). Строки есть не что иное, как имеющиеся в справочнике *записи* (или, точнее говоря, *элементы справочника*), а колонки определяют структуру этих записей. В отличие от перечислений, в справочники пользователь при работе в режиме **1С:Предприятие** может вносить информацию. Так, можно добавлять новые элементы справочника, удалять ненужные и редактировать существующие.

Как уже говорилось, справочники используются в ситуациях, когда необходимо исключить неоднозначный ввод информации. При этом в отличие от перечислений, в справочники пользователь может вносить (и изменять) данные. Аналогом справочника является список данных, который может корректироваться и дополняться в процессе работы. Например, это могут быть списки номенклатуры, организаций или сотрудников.

Система 1С:Предприятие 8.2 позволяет создавать практически любое количество справочников.

Справочник *Должности*

Продолжим разработку нашей конфигурации созданием простого справочника "Должности", в котором будут содержаться названия должностей специалистов условного учебного центра (в качестве должностей внесем: системный администратор, преподаватель и др.). Вместо неформального слова "колонки" (отражающего представление справочника в виде таблицы) в системе 1С:Предприятие используется более общий термин — *реквизиты* (колонок столько, сколько у справочника реквизитов).

При этом в любом справочнике всегда есть два *обязательных* реквизита (присутствующих в любом справочнике независимо от наших действий), которыми являются **Код** и **Наименование**. Другими словами, в каждой строке любого справочника всегда есть поле кода и поле наименования.

Начальные технические действия при разработке справочников аналогичны тем, которые мы ранее выполняли при создании перечислений. Чтобы создать новый справочник, необходимо в дереве объектов конфигурации (см. рис. 1.12) щелкнуть правой кнопкой мыши на пункте **Справочники**, после чего в появившемся контекстном меню следует выбрать пункт **Добавить**. В результате на экране откроется окно редактирования объекта конфигурации (рис. 1.14), только на этот раз не перечисления, а справочника. Здесь на вкладке **Основные** следует ввести имя, которое мы собираемся дать создаваемому справочнику — **Должности.**

Часть параметров (**Представление объекта**, **Расширенное представление объекта** и т. д.) в окне редактирования объекта конфигурации не имеет существенного значения при разработках учебных конфигураций (т. е. для тех, на которых мы только учимся). Разумеется, когда вы будете заниматься профессиональными раз-

работками, то указанные параметры следует использовать. Подробно они описаны в [6].

Здесь для представления о назначении данных параметров прокомментируем лишь поле **Представление объекта**. Этот параметр определяет представление объекта в единственном числе и используется в названии стандартной команды. Например, для созданного нами объекта **Должности** это может быть **Должность:создать**. Если мы не зададим значение в поле **Представление объекта**, то вариант команды для данного случая будет выглядеть так: **Должности:создать**. Понятно, что если для учебных примеров это вполне допустимо, то при разработке реальных проектов подобные команды будут смотреться немного вычурно. Таким образом, можно сказать, что **Представление объекта** стоит задавать тогда, когда синоним объекта конфигурации задан во множественном числе.

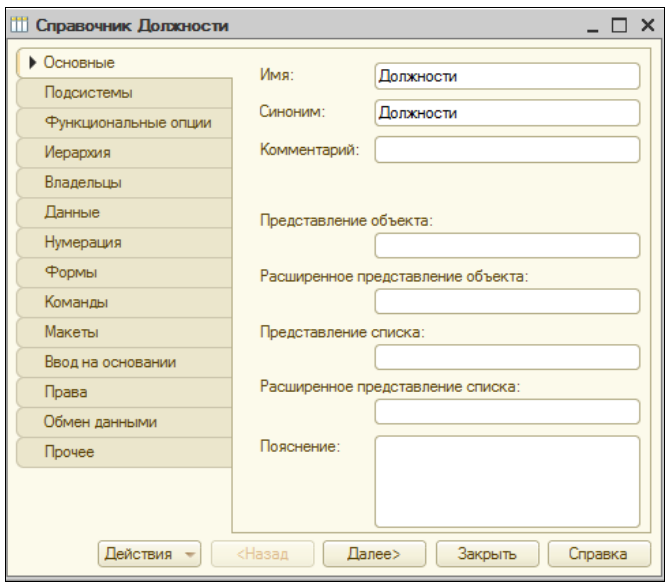


Рис. 1.14. Окно редактирования справочника "Должности"

Если подвести некий итог, то к настоящему моменту, не выполнив особо сложных действий, мы, тем не менее, создали новый объект конфигурации — справочник "Должности", с которым уже можно работать, поскольку никаких дополнительных реквизитов в данном справочнике не требуется.

Следующий новый технический момент — заполнение созданного справочника при работе в режиме **1С:Предприятие**. Для этого в меню **Отладка** (см. рис. 1.12) следует выбрать пункт **Начать отладку**, что позволит нам перейти в режим **1С:Предприятие** (или, можно сказать, в рабочую среду для пользователей системы 1С:Предприятие) (рис. 1.15). В этом случае мы можем работать с информационной базой, которая построена системой на основании созданной конфигурации.

Само окно приложения и его элементы, разумеется, требуют комментария, но пока на этом останавливаться не будем и проверим созданный справочник "Должности"

в работе. Так, в левой части окна приложения располагается *панель навигации*, на которой присутствует гиперссылка, соответствующая единственному созданному на настоящий момент справочнику "Должности". После щелчка мышью по данной гиперссылке произойдет открытие формы списка данного справочника (рис. 1.16).

Справа от панели навигации располагается *рабочая область основного окна приложения*. Рабочая область предназначена для отображения форм представления хранимой информации. Как правило, в рабочей области окна расположена одна форма (см. рис. 1.16).

Примечание

В подавляющем большинстве случаев команды панели навигации позволяют открывать в рабочей области окна формы списков данных.

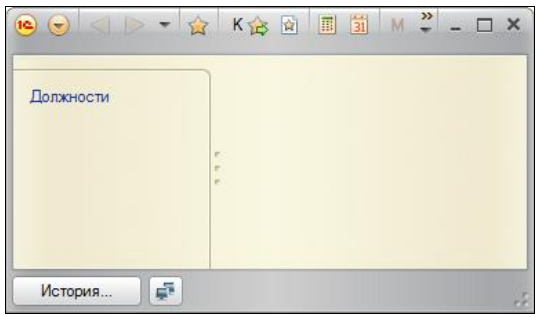


Рис. 1.15. Окно приложения в режиме **1С:Предприятие**

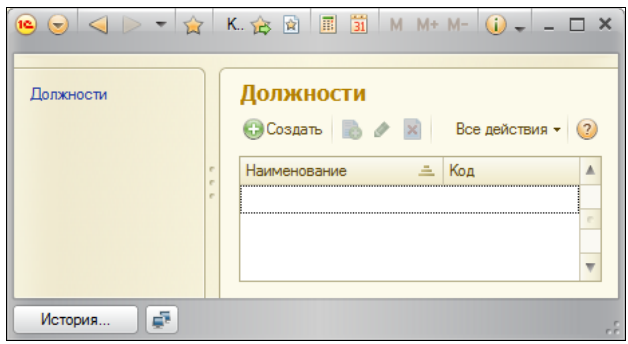


Рис. 1.16. Форма списка справочника "Должности"

Сама по себе форма справочника представляет собой также объект конфигурации. Она является важным связующим звеном между информационной базой и пользователем. Так, пользователь может с помощью форм просматривать и корректировать имеющиеся в информационной базе данные. Также с помощью форм пользователи вносят в базу данных новые сведения.

Примечание

Форма обычно включает в себя панель заголовка формы, командную панель формы и рабочую область формы. В рабочей области отображаются данные. На командной

панели располагаются команды (пиктограммы команд), предназначенные для обработки отображаемых данных.

О формах далее мы поговорим подробно, а сейчас необходимо заполнить справочник "Должности" названиями нескольких должностей. Кроме формирования необходимого практического навыка, заполненный информацией справочник "Должности" потребуется нам для дальнейшей работы.

При заполнении справочника следует воспользоваться пиктограммой **Создать**, которая располагается в пиктографическом меню командной панели формы списка справочника (см. рис. 1.16). После этого на экране откроется форма (рис. 1.17) для добавления нового элемента в справочник. Это так называемая *форма элемента*. Обратим внимание на то, что заполнять поле **Код** не требуется (оно заполняется системой автоматически), а после внесения названия должности в поле **Наименование** следует просто щелкнуть по кнопке **Записать и закрыть**. В результате в форме списка справочника "Должности" мы увидим новую строку (в самом же справочнике будет зафиксирован новый элемент — информация о новой должности). Аналогичным образом следует внести в справочник "Должности" еще несколько названий (рис. 1.18).

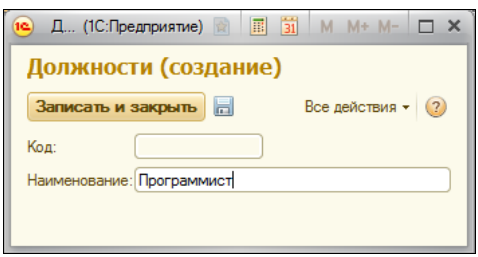


Рис. 1.17. Форма для добавления нового элемента в справочник

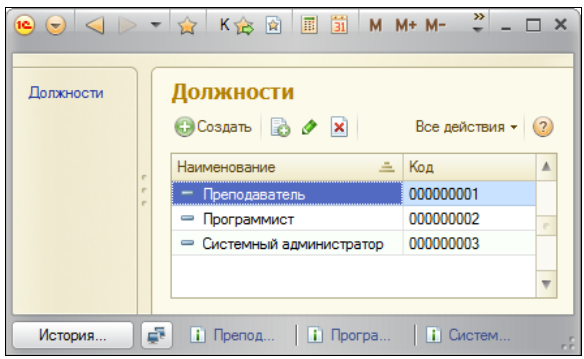


Рис. 1.18. Заполненная форма списка справочника "Должности"

Таким образом, в режиме конфигуратора для любого справочника в окне редактирования объекта конфигурации можно ввести только имя на вкладке **Основные** (см. рис. 1.14), после чего сразу перейти в режим **1С:Предприятие** и начать работу по внесению информации в базу данных.

Вернемся к формам, с которыми мы встретились в этом разделе. В процессе работы в режиме **1С:Предприятие** пользователи просматривают и вводят информацию в справочники с помощью электронных форм (это справедливо не только для справочников). При этом если мы в конфигураторе не совершаем никаких действий для создания формы справочника или какого-либо другого объекта, то платформа автоматически генерирует стандартные формы (это происходит в режиме **1С:Предприятие**, когда у платформы в формировании формы появляется необходимость). В частности, экранная *форма списка* справочника, которую мы видели на рис. 1.18, генерируется системой по умолчанию. Аналогично автоматически платформой

создается форма элемента (см. рис. 1.17), с помощью которой производится ввод новых элементов в справочник.

Технология генерации форм по умолчанию позволяет существенно ускорить процесс разработки, когда в прикладном решении не требуется включения дополнительных сервисных функций помимо тех, которые имеются в формах, создаваемых системой по умолчанию.

Примечание

Форма — это объект системы 1С:Предприятие, который предназначен для ввода и просмотра информации. Форма представляет собой некоего посредника между пользователем и информационной базой. Форма может быть эффективно использована для организации работы с данными. При открытии определенной формы справочника или документа система формирует на ней необходимую информацию. В дальнейшем при вводе данных в элементы, расположенные на форме, они сохраняются в соответствующих основных объектах (тех же справочниках, документах, регистрах и т. д.). Программный продукт 1С:Предприятие 8.2 предоставляет пользователю возможность при желании отказаться от самостоятельного создания форм, а использовать те, которые автоматически генерируются системой в режиме **1С:Предприятие**. В этом случае от разработчика требуется меньше действий при создании прикладных решений.

Справочник *Специалисты*

После создания в какой-то степени символического справочника "Должности" (он будет использоваться в дальнейшем для подстановки названий должностей) начнем разработку другого, более важного для рассматриваемой задачи справочника "Специалисты".

Заполнив в окне редактирования объекта конфигурации вкладку **Основные**, перейдем к вкладке — **Иерархия**. Дело в том, что в большинстве случаев удобнее хранить данные по определенным категориям. Например, если речь идет о товарах, то категориями товаров могут быть следующие группы:

- Обувь;
- Одежда;
- Сумки.

А внутри каждой из этих групп уже располагаются товары с указанием их стоимости и прочих свойств. Вкладка **Иерархия** как раз и предназначена для такой организации справочников.

Вернемся к нашей задаче, в которой специалистов учебного центра будем классифицировать по группам (по профилю специализации). Для этого на вкладке **Иерархия** следует установить флажок **Иерархический справочник** (рис. 1.19).

Иерархия в справочниках может быть двух видов:

- иерархия групп и элементов;
- иерархия элементов.

Описанный выше случай с категориями товаров как раз является примером *иерархии групп и элементов*. В этом случае любой элемент, представляющий группу, яв-

ляется *родителем* для всех элементов и групп, входящих в данную группу. В приведенном выше примере группа "Обувь" является родителем для всех названий (моделей) обуви, а также для групп, входящих в ее состав (например, обувь может дополнительно группироваться по странам-производителям).

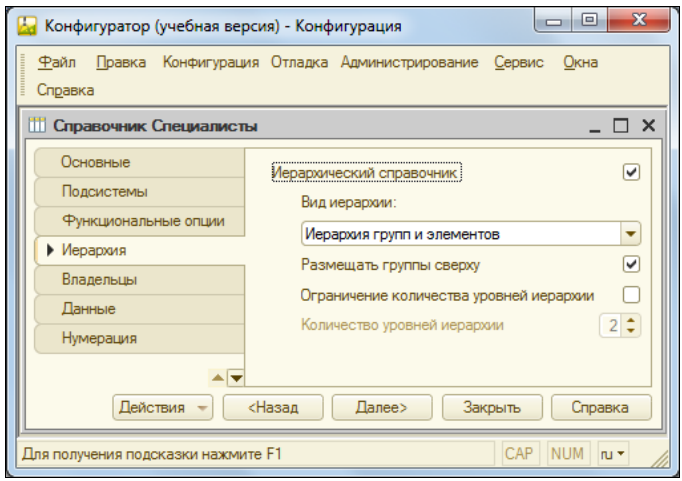


Рис. 1.19. Вкладка **Иерархия** окна редактирования справочника "Специалисты"

Другой вид иерархии — *иерархия элементов*. В этом случае в качестве *родителя* выступает один из элементов справочника. Например, справочник "Сотрудники" можно организовать в иерархическом виде. Тогда в данном справочнике будет представлена информация о том, кто из сотрудников кому подчиняется.

В нашем созданном справочнике "Специалисты" установим **Вид иерархии** — **Иерархия групп и элементов**.

> ***Примечание***
>
> Количество уровней иерархии в справочнике можно ограничить, что делается с помощью установки флажка **Ограничение количества уровней иерархии** и выбора необходимого значения в поле **Количество уровней иерархии** (см. рис. 1.19). Если же данный флажок не установлен, то максимальное количество вложенности справочника реально не ограничено. В рассматриваемом примере при создании иерархического справочника сбросим данный флажок.

Флажок **Размещать группы сверху** позволяет обеспечить традиционную группировку информации в справочнике (и его, как правило, всегда устанавливают).

Обратимся теперь к вкладке **Данные** (рис. 1.20), где определяются реквизиты справочника. Здесь перед нами открывается форма, содержащая несколько элементов управления. В правой верхней части этой формы указаны размеры полей, отводимые для кода и наименования. Эти значения при создании нового справочника устанавливаются системой по умолчанию, но при необходимости их можно изменить. Что касается длины кода, то в нашем случае достаточно оставить значение, предложенное по умолчанию, а для наименования (фактически для фамилии, имени и отчества специалиста) можно увеличить число символов до 50.

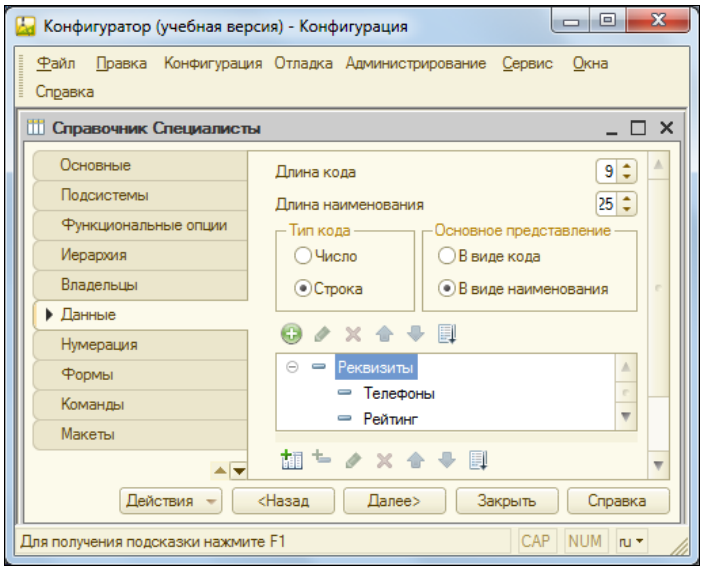


Рис. 1.20. Вкладка **Данные** окна редактирования справочника "Специалисты"

Поле **Код** используется для идентификации элементов справочника (записей справочника). Содержание этого поля должно быть уникальным для каждого элемента справочника. В принципе, пользователю можно не задумываться о заполнении поля **Код** — система при работе в режиме **1С:Предприятие** сама обеспечивает автоматическое внесение уникальных кодов при создании новых элементов в справочнике. Однако при желании можно самостоятельно заполнять поле кода, исходя из определенных соображений. Например, при использовании некоторой систематизации содержание поля кода помогает пользователю быстро идентифицировать записи (в поле кода при желании можно заложить необходимую информацию об элементах справочника).

Поле **Наименование** отражает основную информацию об элементе справочника (например, если рассматривается справочник городов, то поле **Наименование** предназначено для названия города). Для него можно отвести различное число символов, однако существуют рекомендации фирмы 1С (или, можно даже сказать, норматив) для числа символов в наименовании [4].

В центре вкладки **Данные** окна редактирования объекта конфигурации (см. рис. 1.20) расположено поле **Реквизиты**, в котором можно добавлять дополнительные реквизиты в справочник. Это выполняется с помощью самой левой кнопки расположенного выше пиктографического меню. В результате на экране открывается уже знакомая палитра свойств (рис. 1.21), где следует установить имя реквизита — **Телефоны**. Оно предназначено для хранения информации о контактных телефонах, по которым можно связаться со специалистом.

Для каждого реквизита должен быть определен тип данных или, иначе говоря, тип значений, который можно заносить в реквизит при работе в режиме **1С:Предприятие**. Этот параметр также следует установить в *палитре свойств*. В качестве

типа данных для реквизита **Телефоны** можно указать строку не более 50 символов (см. рис. 1.21), поскольку номеров телефонов может быть несколько.

Примечание

Палитру свойств можно открыть с помощью двойного щелчка мыши по имени реквизита на вкладке **Данные** (см. рис. 1.21).

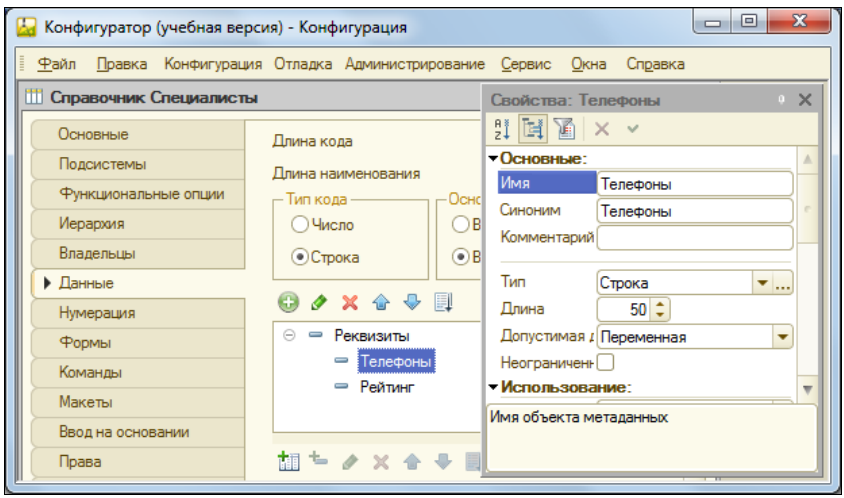


Рис. 1.21. Палитра свойств реквизита **Телефоны**

Добавим в справочник "Специалисты" еще один реквизит — **Рейтинг** (тип данных — **Число**) для характеристики квалификации специалиста. Таким способом при необходимости можно добавлять разнообразные реквизиты в создаваемые справочники, делая их более информативными.

Примечание

С помощью кнопок панели инструментов, расположенной над перечнем реквизитов (они являются подчиненными объектами), производится их добавление, удаление и упорядочивание.

Следует заметить, что в отличие от системы 1С:Предприятие 7.7, программные продукты версий, начиная с 1С:Предприятие 8.0, позволяют создавать для элементов справочников табличные части.

Примечание

Табличные части также являются объектами конфигурации.

Так, для каждого элемента справочника можно создать одну или несколько подчиненных табличных частей (таблиц). Здесь мы сразу воспользуемся этой возможностью и добавим в справочник "Специалисты" табличную часть с именем **МестаРаботы** (рис. 1.22).

Дело в том, что преподаватели (особенно высокой квалификации) часто одновременно работают в нескольких местах. При этом у одного специалиста дополни-

тельных мест трудовой деятельности может быть три, а у другого даже пять. Табличная часть справочника для хранения информации подобного плана в этом случае идеально подходит. В противном случае мы должны были бы создавать несколько реквизитов вне табличной части, причем их количество заранее предусмотреть достаточно сложно.

Далее уже в табличной части реквизита **МестаРаботы** создадим реквизит **Организация** (это выполняется с помощью второй слева кнопки пиктографического меню, относящегося к табличным частям), в качестве типа данных которого следует указать строку длиной не более 50 символов. Кроме того, добавим реквизит **Должность** (тип данных — **СправочникСсылка.Должности**). В этом случае, при работе пользователя в режиме **1С:Предприятие** для заполнения поля **Должность** будет автоматически открываться форма выбора справочника "Должности".

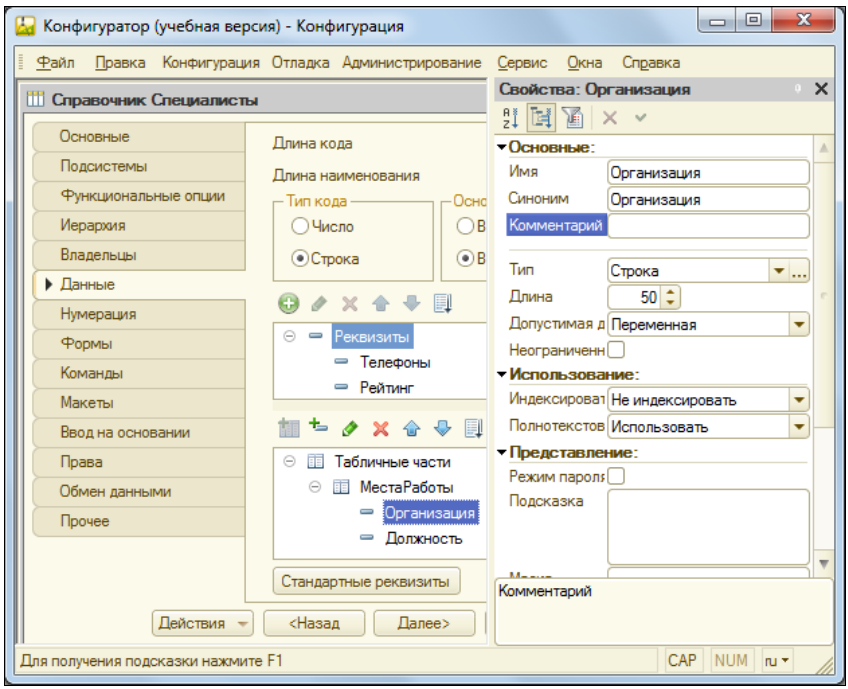


Рис. 1.22. Добавление табличной части в справочник "Специалисты"

Таким образом, мы разработали структуру нового объекта конфигурации — справочника "Специалисты", и теперь следует позаботиться о том, как пользователь будет с ним работать. Как уже говорилось, в системе 1С:Предприятие имеются разнообразные формы справочников, формируемые платформой по умолчанию. Кроме того, инструментальные средства легко позволяют пользователю самому создавать удобные формы.

Способ создания формы по умолчанию очень удобен, когда после разработки объекта конфигурации необходимо как можно быстрее начать работу по внесению данных в информационную базу (либо продолжить работу в конфигураторе по созданию других объектов).

У справочника наиболее часто используется *форма списка*, в которой информация представлена в виде таблицы, где на каждую запись отводится одна строка (это стандартный вид любого справочника). Основным достоинством подобной формы является то, что в этом случае на экране видно сразу несколько элементов справочника.

С другой стороны, недостаток формы списка обнаруживается, когда в справочнике присутствует много колонок (большое количество реквизитов), и хотелось бы, чтобы все они одновременно были видны пользователю. В этом случае форму списка часто приходится прокручивать, чтобы увидеть необходимую информацию. Выход в данной ситуации заключается в использовании еще одной стандартной формы — *формы элемента*, которая позволяет отображать на экране информацию только об одном элементе справочника. При этом даже если реквизитов в справочнике много, все они видны на экране.

У формы элемента имеется еще одно существенное достоинство. Если в справочнике присутствует табличная часть (как в только что созданном справочнике "Специалисты"), то в форме списка она не отображается, а в форме элемента, наоборот, видна.

Посмотрим теперь, как выглядят описанные формы справочника в режиме **1С:Предприятие**. Для этого следует перейти в этот режим (через меню **Отладка**) и открыть справочник "Специалисты".

Ранее при разработке мы указали, что справочник "Специалисты" будет иерархическим, и сейчас создадим в нем две группы: **Техническое направление** и **Гуманитарное направление** (рис. 1.23). В этих группах будут размещаться специалисты разного направления.

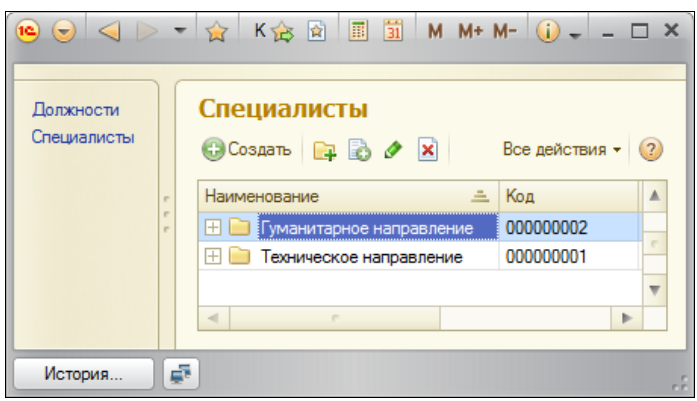


Рис. 1.23. Создание двух групп в справочнике "Специалисты"

Примечание

Для создания группы в справочнике следует воспользоваться второй (слева) пиктограммой, расположенной на командной панели формы списка справочника (см. рис. 1.23). При наведении мыши на пиктограмму на экране появляется подсказка **Создание новой группы**.

Примечание

Для перехода в режим **1С:Предприятие** из режима **Конфигуратор** можно воспользоваться меню **Сервис**, где следует выбрать пункт **1С:Предприятие**. В процессе разработки конфигурации удобнее пользоваться пунктом **Начать отладку** (**Продолжить отладку**) из меню **Отладка**.

Далее в группе **Техническое направление** создадим три дополнительные группы в соответствии с рис. 1.24 для детализации данной категории специалистов. При этом в форме группы (как, впрочем, и в форме элемента) присутствует реквизит **Родитель**. Он определяет исходную группу, куда будет входить создаваемая группа или создаваемый элемент справочника. В данном случае при создании дополнительной группы в группе **Техническое направление** реквизит **Родитель** принимает значение **Техническое направление**.

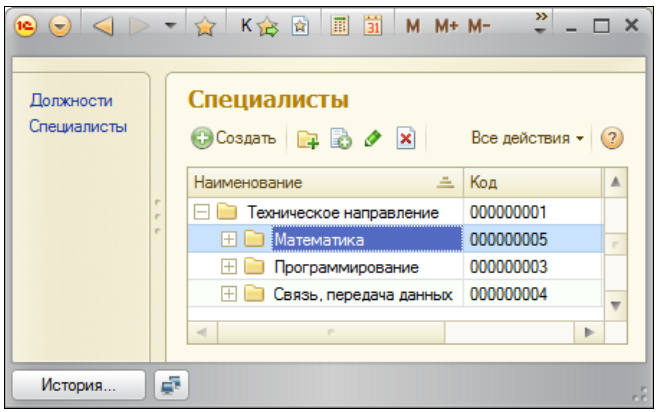


Рис. 1.24. Создание трех вложенных групп в справочнике "Специалисты"

Откроем теперь группу **Программирование**, в которую внесем преподавателя (это уже будет элемент справочника) в соответствии с информацией, представленной на рис. 1.25.

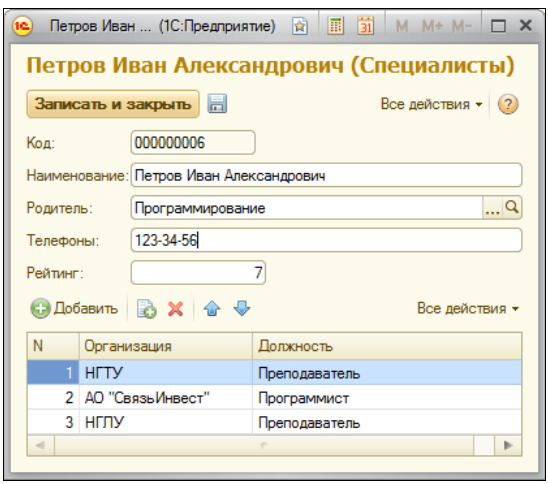


Рис. 1.25. Заполнение формы элемента справочника "Специалисты"

Примечание

Для создания элемента в справочнике следует воспользоваться кнопкой **Создать**, расположенной в пиктографическом меню формы списка справочника (см. рис. 1.24). При этом новый элемент будет создан в той группе, которая в настоящий момент открыта.

Обратим внимание на то, что при редактировании формы элемента справочника становится доступной для работы и табличная часть справочника. В форме же списка справочника табличная часть элементов на экране не отображается.

Важно отметить, что при использовании в справочнике иерархии (как в нашем случае) автоматически формируется дополнительный реквизит **Родитель**, который как раз и содержит указание на группу, в которой размещается текущий элемент. В созданном нами элементе (см. рис. 1.25) реквизит **Родитель** содержит указание на группу **Программирование**.

Примечание

Каждый справочник может иметь практически неограниченное число табличных частей.

Далее требуется внести в группу **Программирование** еще несколько элементов в соответствии с рис. 1.26.

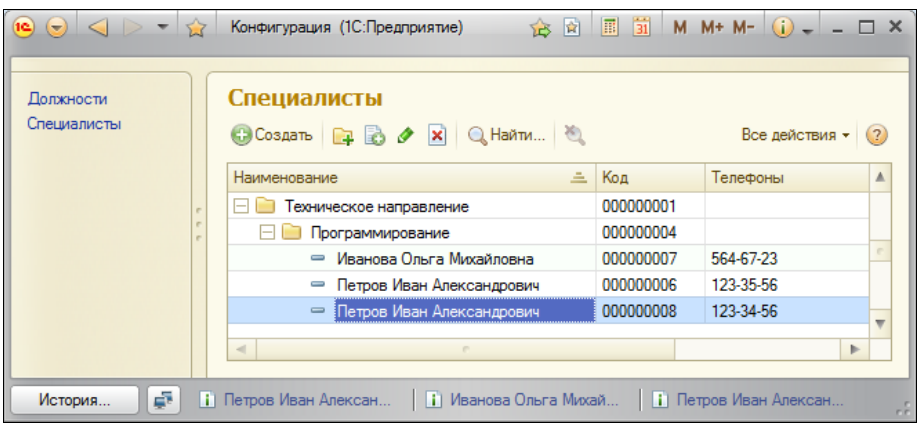


Рис. 1.26. Заполненная форма списка справочника "Специалисты"

Важный практический прием касается переноса элементов из одной группы в другую. Для этого можно воспользоваться пунктом **Переместить в группу**, который включен в раздел **Все действия** (см. рис. 1.26). В этом случае на экране открывается новое окно для выбора группы, куда выделенный элемент (или целиком группу, если выделена группа) следует перенести.

Для иерархического справочника платформа 1С:Предприятие 8.2 предлагает несколько вариантов просмотра. Если вы параллельно выполняете описанные действия на своем компьютере, то форма списка справочника, которую видите, может отличаться от варианта, представленного на рис. 1.26. Это связано с тем, что существует несколько режимов просмотра иерархического справочника:

- иерархический справочник;
- список;
- дерево.

Эти режимы можно установить, если воспользоваться пунктом **Режим просмотра** из выпадающего списка **Все действия** (см. рис. 1.26).

Таким образом, на данном этапе мы познакомились с технологией разработки структуры справочников (объектов конфигурации) в режиме конфигуратора, а также узнали, каким образом заполнять справочники данными с помощью электронных форм, создаваемых системой по умолчанию. На этапе разработки и отладки прикладных решений такой вариант (использование форм по умолчанию) применяется достаточно часто.

Взглянем теперь немного критически на форму списка справочника "Специалисты" (см. рис. 1.26). Так, информация о телефонах и рейтинге используется не часто и ее просмотр лучше ограничить формой элемента (см. рис. 1.25), когда мы обращаемся к сведениям по конкретному специалисту. Кроме того, эти реквизиты для групп вообще не имеют смысла. Поэтому лучше разработать другую форму списка, которую будем использовать вместо генерируемой системой по умолчанию.

Для создания формы списка вернемся в окно редактирования справочника "Специалисты" и перейдем на вкладку **Формы** (рис. 1.27). Здесь в одноименном фрейме перечислены те виды форм, которые можно создать для справочника.

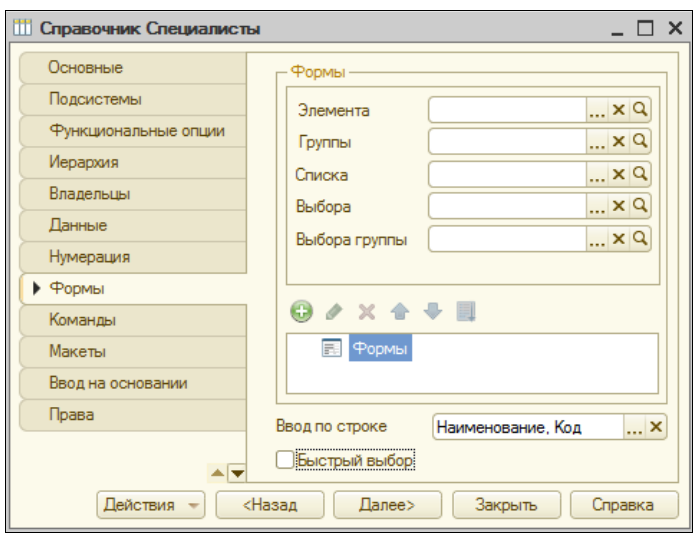


Рис. 1.27. Вкладка **Формы** окна редактирования справочника "Специалисты"

Наша задача заключается в создании формы списка, поэтому щелкнем по изображению лупы напротив соответствующей надписи (**Списка**). В результате мы перейдем к работе с конструктором форм (рис. 1.28). В данном окне конструктора все установки уже выполнены системой по умолчанию (главная опция — установка

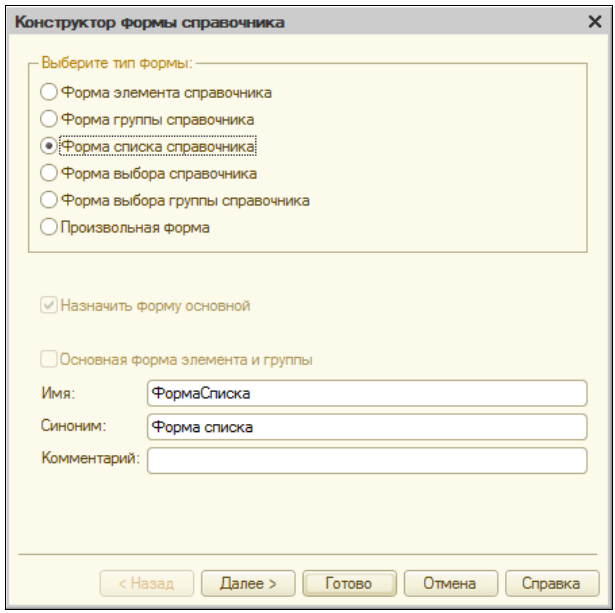


Рис. 1.28. Окно конструктора формы справочника

переключателя **Форма списка справочника**) и нам ничего менять не требуется, поэтому просто щелкнем по кнопке **Далее**.

В следующем окне (рис. 1.29) оставим в качестве реквизитов формы списка только **Код** и **Наименование**, после чего щелкнем по кнопке **Готово**. В результате мы

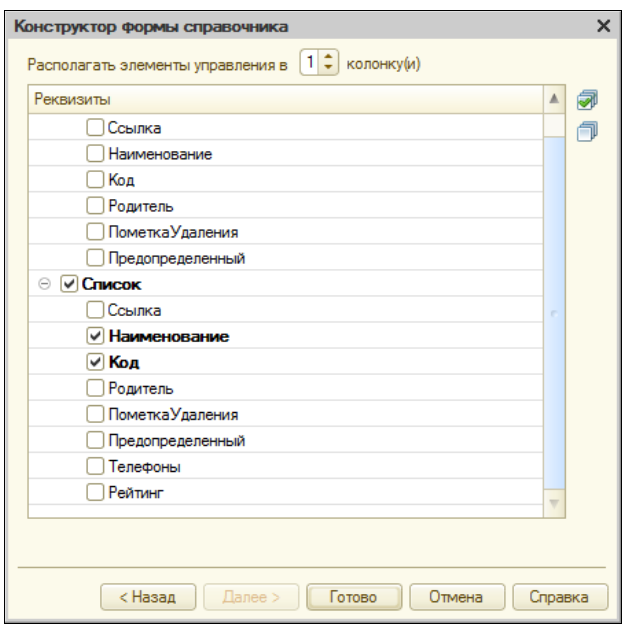


Рис. 1.29. Окно конструктора формы справочника для включения реквизитов в форму списка

увидим на экране итог работы конструктора — форму списка в окне редактора форм (рис. 1.30). *Редактор форм* является очень сложным инструментом, и далее в книге мы будем весьма активно с ним работать. Сейчас же от нас требуется просто перейти в режим **1С:Предприятие** и посмотреть результат выполненных действий (рис. 1.31).

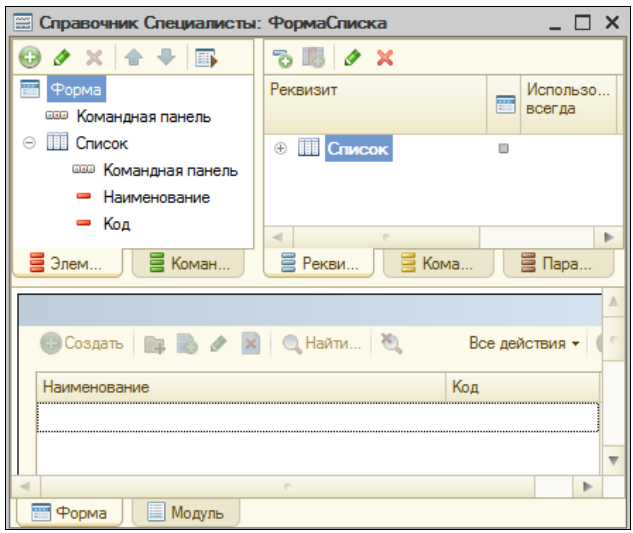


Рис. 1.30. Форма списка справочника "Специалисты" в окне редактора форм

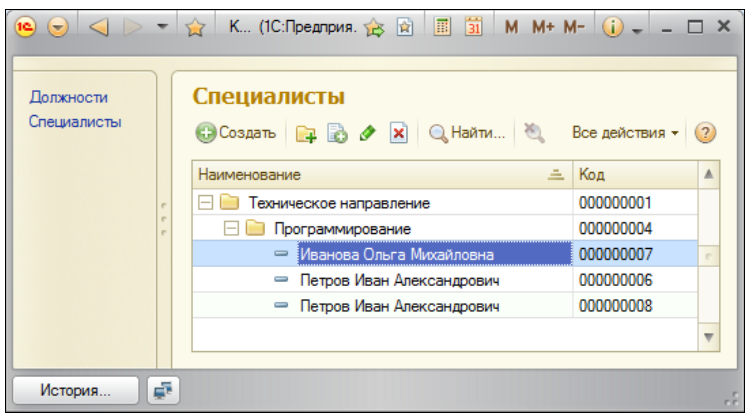


Рис. 1.31. Пользовательская форма списка справочника "Специалисты"

Справочник *Курсы*

Следующая практическая задача заключается в создании справочника, который будет хранить информацию об учебных курсах, предлагаемых учащимся нашего учебного центра. В этом же справочнике в категории дополнительных реквизитов поместим информацию о стоимости курсов и их продолжительности. В функцио-

нальном плане на примере данного справочника мы познакомимся с рядом новых ресурсов разработки. Так, предусмотрим автоматический контроль корректности вводимых данных со стороны системы 1С:Предприятие 8.2.

> ***Примечание***
>
> При параллельном выполнении описанных в книге действий на компьютере желательно соблюдать точность в названиях. Это связано с использованием приводимых обозначений в последующих программных процедурах.

Вернемся в конфигуратор и приступим к созданию еще одного справочника. Техническая работа во многом будет аналогичной ранее описанным действиям, и поэтому подчеркнем лишь существенную информацию для выполнения практических шагов в конфигураторе.

Создаваемый справочник назовем "Курсы", а кроме обязательных реквизитов (**Код** и **Наименование**) на вкладке **Данные** окна редактирования объекта конфигурации добавим еще четыре реквизита (рис. 1.32):

- **ДатаНачала** — для записи даты начала занятий по данному курсу (очевидно, что в качестве типа данных следует выбрать значение **Дата**);
- **Продолжительность** — для определения продолжительности занятий в днях (тип данных — **Число**);
- **Стоимость** — для указания стоимости обучения (тип данных — **Число**);
- **Преподаватель** — для определения специалиста, который будет эту дисциплину вести (тип данных — **СправочникСсылка.Специалисты**).

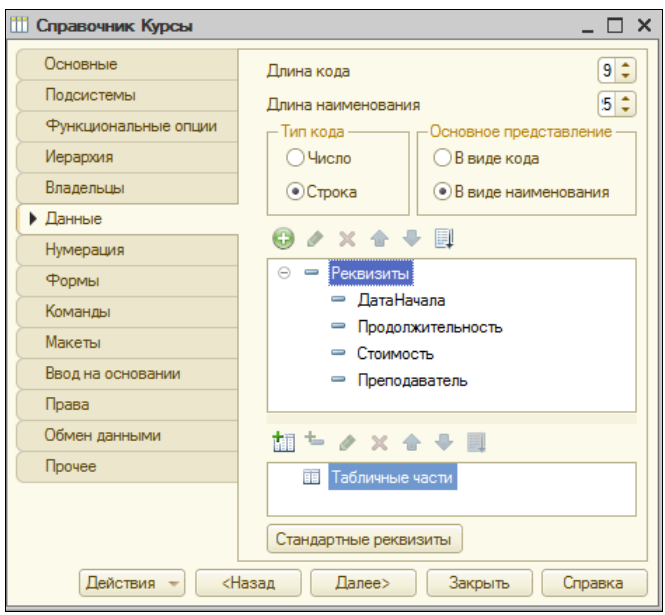


Рис. 1.32. Вкладка **Данные** окна редактирования справочника "Курсы"

Примечание

При создании справочников (как, впрочем, и других объектов) системой автоматически генерируются новые типы данных. Так, после разработки справочника "Специалисты" в системе появился новый тип данных — **СправочникСсылка.Специалисты**. Это позволяет хранить в указанном выше поле **Преподаватель** ссылку на конкретный элемент (он также является объектом) справочника "Специалисты". Подобные типы данных будут часто встречаться в книге при разработке объектов конфигурации.

Теперь наша задача заключается в установке допустимого интервала (по дням) продолжительности курсов. Для этого у числовых реквизитов справочника имеются свойства:

- **Минимальное значение**;
- **Максимальное значение**.

Установка свойств объектов конфигурации, как мы уже знаем, производится с помощью палитры свойств. Таким образом, необходимо выполнить двойной щелчок мышью по реквизиту **Продолжительность** в окне (см. рис. 1.32). В результате на экране откроется палитра свойств данного реквизита (рис. 1.33). Здесь требуется установить значения выше упоминаемых свойств так, чтобы система самостоятельно контролировала продолжительность каждого курса от 3-х до 12-ти дней.

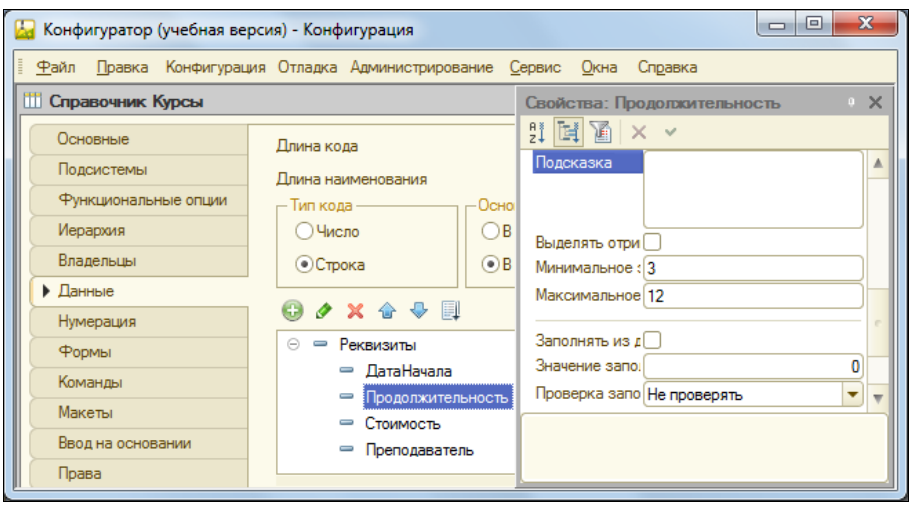


Рис. 1.33. Палитра свойств реквизита **Продолжительность**

Теперь перейдем в режим **1С:Предприятие** и попробуем ввести в справочник "Курсы" данные согласно рис. 1.34. В этом случае система самостоятельно блокирует работу пользователя при внесении значения вне допустимого диапазона в поле **Продолжительность**.

Для работы по созданию прикладного решения в рамках данной главы нам потребуется несколько названий курсов, которые и следует внести в созданный справочник в режиме **1С:Предприятие** (рис. 1.35).

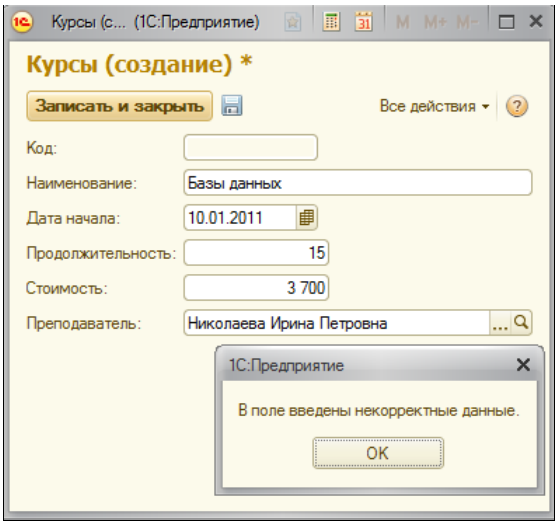


Рис. 1.34. Контроль вводимых значений в форме элемента справочника "Курсы"

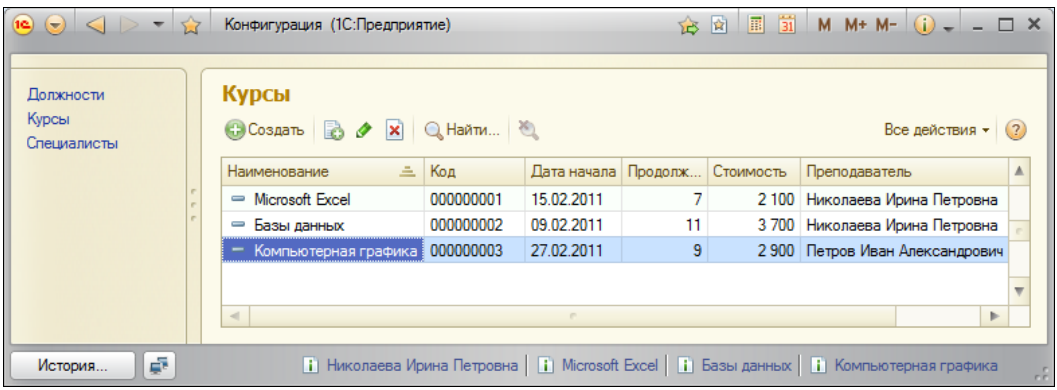


Рис. 1.35. Форма списка справочника "Курсы"

Выполним еще одно изменение в справочнике. Так в форме списка на рис. 1.35 названия курсов отражаются в столбце **Наименование**. Наша задача заключается в том, чтобы заменить это стереотипное название столбца более осмысленным — **Название курса**. Для этого сначала необходимо в дереве объектов конфигурации выделить объект — справочник **Курсы** (рис. 1.36). Затем с помощью правой кнопки мыши следует вызвать контекстное меню, где воспользоваться разделом **Стандартные реквизиты**. В результате на экране откроется окно с аналогичным названием (см. рис. 1.36). Здесь требуется с помощью правой кнопки мыши открыть палитру свойств реквизита **Наименование** (рис. 1.37). В этом окне осталось внести необходимый текст в поле **Синоним**. Выберем вариант **Название курса**, после чего можно перейти в режим **1С:Предприятие** и посмотреть на результат проделанных изменений (рис. 1.38). В результате выполненных шагов изменился внешний вид формы списка справочника, а именно изменилась подпись для реквизита **Наименование**. При этом само имя данного реквизита осталось прежним.

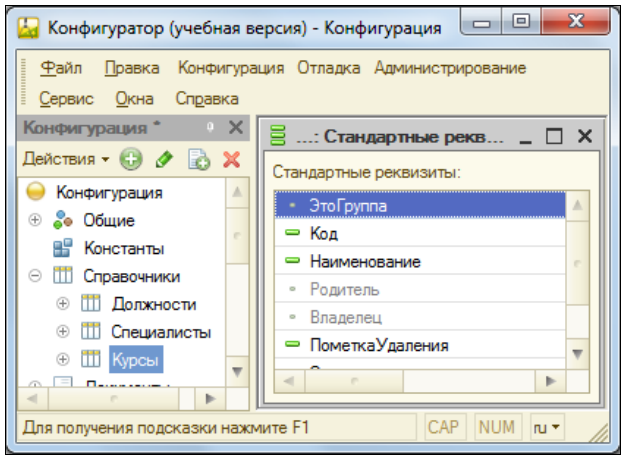


Рис. 1.36. Дерево объектов конфигурации и окно стандартных реквизитов справочника "Курсы"

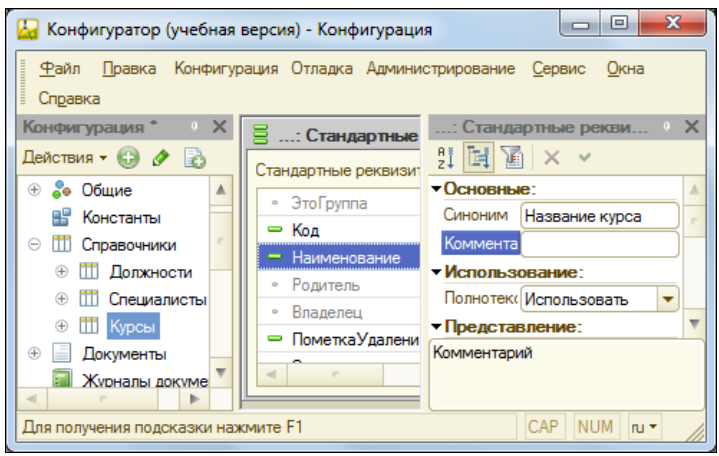


Рис. 1.37. Изменение текста шапки в форме списка справочника "Курсы"

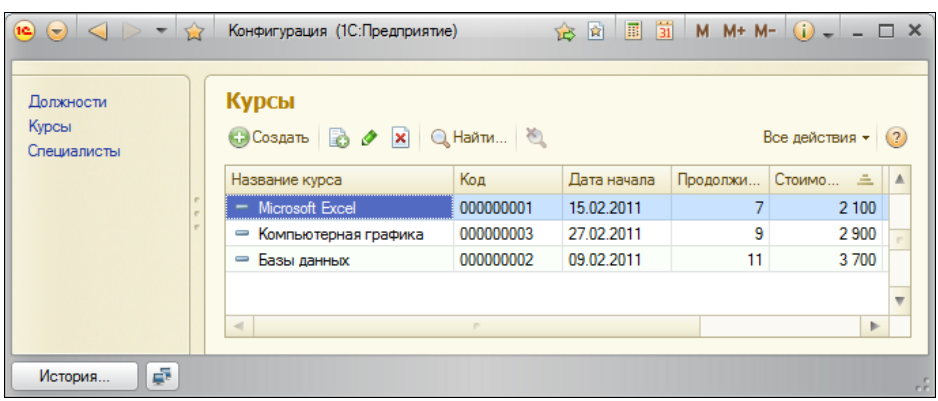


Рис. 1.38. Форма списка справочника "Курсы" с измененной подписью над первым столбцом

Печать и настройка формы списка справочника

Кроме электронной формы списка справочника в реальной работе часто требуется и его печатная форма. Платформа 1С:Предприятие 8.2 позволяет это легко реализовать. Рассмотрим возможности печати для справочника "Курсы". Так, воспользуемся кнопкой **Все действия** в форме списка справочника "Курсы" (рис. 1.39). В результате перед нами откроется выпадающий список с перечнем возможных команд.

В данном случае нас интересует пункт **Вывести список**, с помощью которого на экране открывается форма настройки списка (рис. 1.40). Выберем вывод информации в табличный документ и установим флажки против всех имеющихся реквизитов. После этого щелчком по кнопке **ОК** можно сформировать *табличный документ*, содержащий информацию об имеющихся элементах справочника "Курсы" (рис. 1.41).

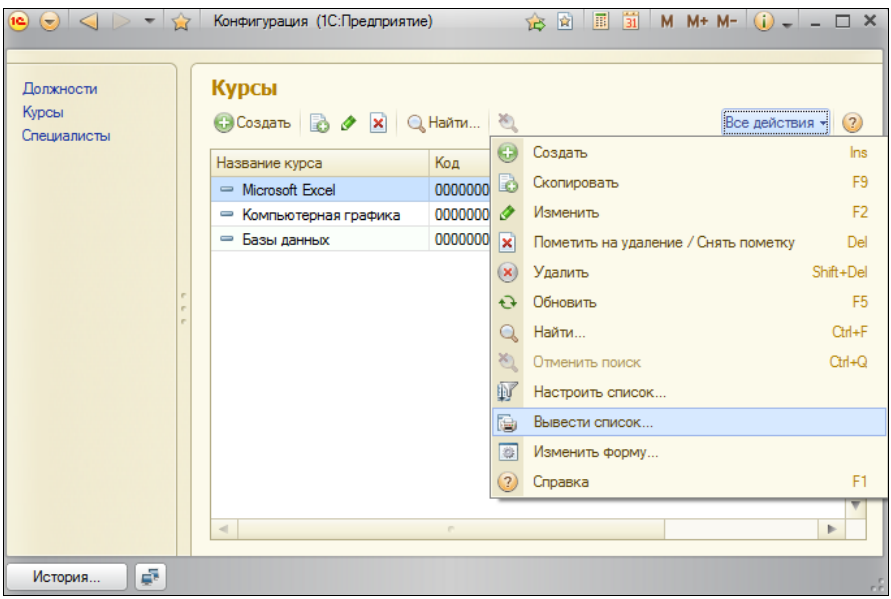


Рис. 1.39. Состав меню **Все действия**

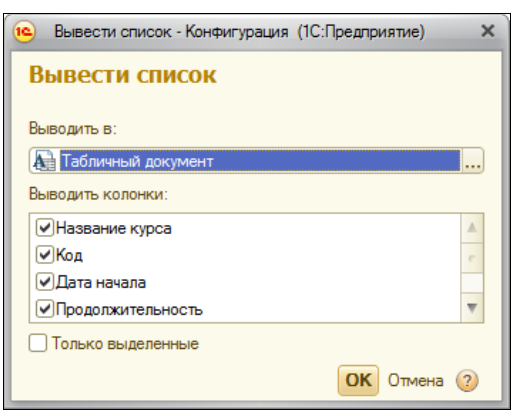
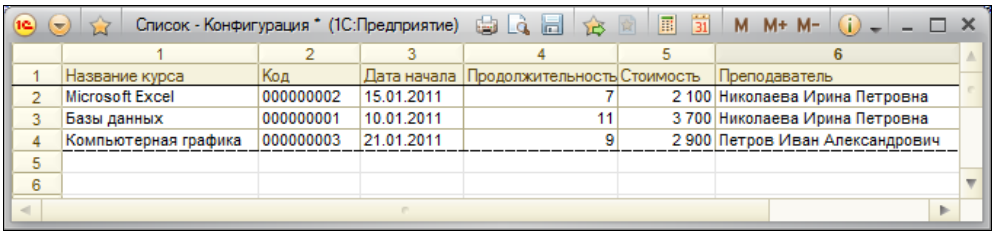


Рис. 1.40. Окно настройки списка для вывода информации в табличный документ



	1	2	3	4	5	6
1	Название курса	Код	Дата начала	Продолжительность	Стоимость	Преподаватель
2	Microsoft Excel	000000002	15.01.2011	7	2 100	Николаева Ирина Петровна
3	Базы данных	000000001	10.01.2011	11	3 700	Николаева Ирина Петровна
4	Компьютерная графика	000000003	21.01.2011	9	2 900	Петров Иван Александрович
5						
6						

Рис. 1.41. Табличный документ со списком справочника "Курсы"

Теперь если вы хотите данный табличный документ напечатать, то следует воспользоваться *главным меню приложения*, которое открывается щелчком по кнопке с изображением стрелки (рис. 1.42), расположенной в верхней части окна. После этого в разделе **Файл** следует выбрать пункт **Печать**.

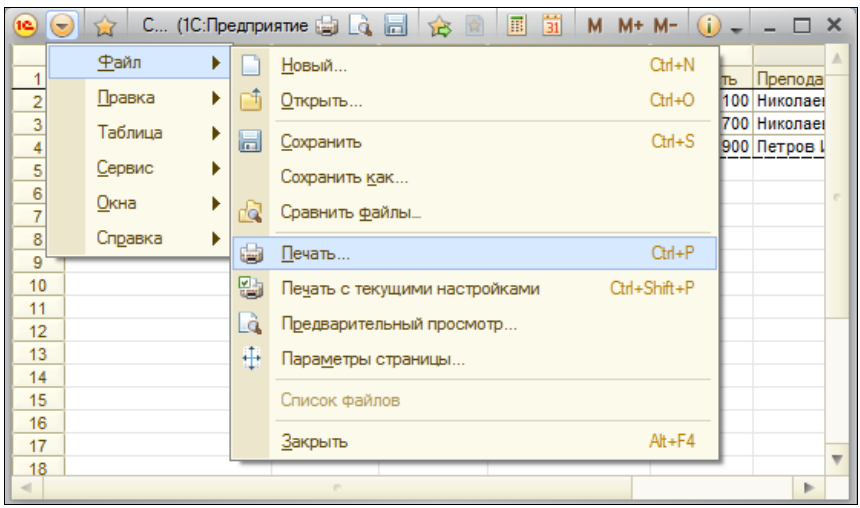


Рис. 1.42. Главное меню приложения

Справочник *Клиенты*

Добавим в нашу информационную базу еще один справочник, который назовем "Клиенты". В нем будут отражены учащиеся или слушатели курсов. Для продолжения практических действий вернемся в конфигуратор и начнем уже известным способом создание нового справочника (нового объекта конфигурации).

Во-первых, на вкладке **Данные** окна редактирования объекта конфигурации необходимо внести информацию о реквизитах справочника в соответствии с рис. 1.43. Для реквизитов справочника **Адрес** и **Телефоны** в качестве типа данных укажем строку длиной не более 50 символов.

Во-вторых, с помощью соответствующей кнопки пиктографического меню, расположенного в нижней части окна редактирования объекта конфигурации, создадим в элементе справочника табличную часть с именем **ИзучаемыеКурсы**.

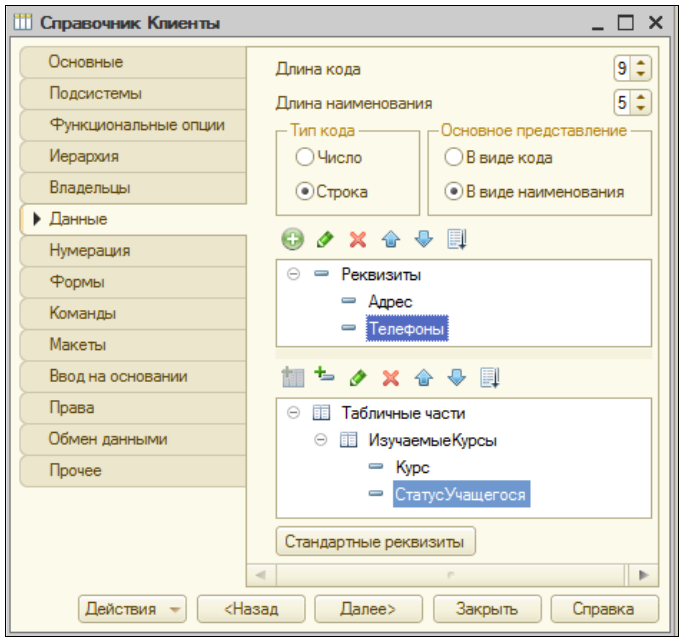


Рис. 1.43. Окно редактирования справочника "Клиенты"

В-третьих, в созданной табличной части разместим два реквизита (см. рис. 1.43): **Курс** (тип данных — **СправочникСсылка.Курсы**) и **СтатусУчащегося** (тип данных — **ПеречислениеСсылка.Статус**). В этом случае при заполнении данных полей в режиме **1С:Предприятие** будут открываться, соответственно, форма выбора справочника "Курсы" и форма выбора перечисления **Статус**.

> ***Примечание***
>
> Поскольку мы эти формы в конфигураторе не создавали, то платформа будет самостоятельно их генерировать непосредственно в режиме **1С:Предприятие**.

Создадим теперь форму элемента справочника "Клиенты". Это необходимо в связи с тем, что мы собираемся далее расширить функционал формы по сравнению с тем, который имеется в форме, генерируемой по умолчанию.

Практические действия для этого следует выполнить на вкладке **Формы**, где во фрейме с таким же названием обратимся к полю для формы элемента. Здесь требуется щелкнуть мышью по пиктограмме с изображением лупы, что позволит перейти к работе с конструктором форм. В первом открывшемся окне конструктора необходимо с помощью соответствующего переключателя подтвердить, что мы собираемся создавать форму элемента справочника, после чего следует щелкнуть по кнопке **Далее**. В результате на экране откроется следующее окно конструктора формы (рис. 1.44), где необходимо сделать отметку флажками для включения соответствующих реквизитов в форму элемента справочника.

Теперь осталось нажать кнопку **Готово** в нижней части окна, после чего на экране откроется созданная форма элемента справочника в окне редактора форм

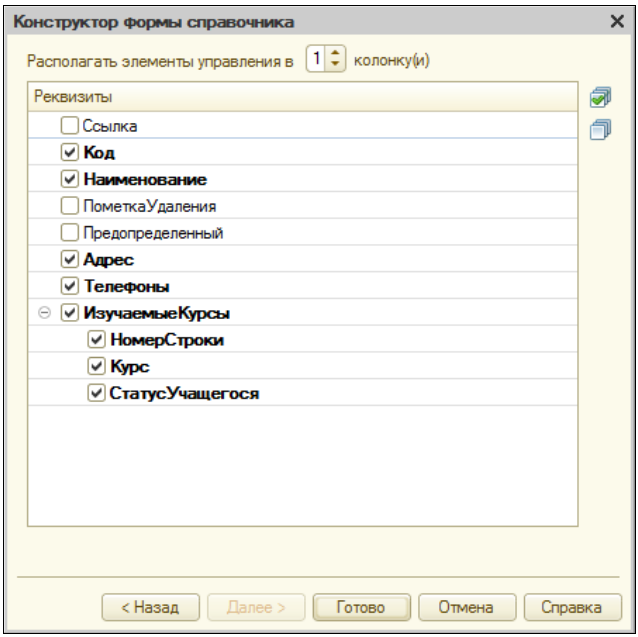


Рис. 1.44. Окно конструктора для добавления необходимых реквизитов в форму

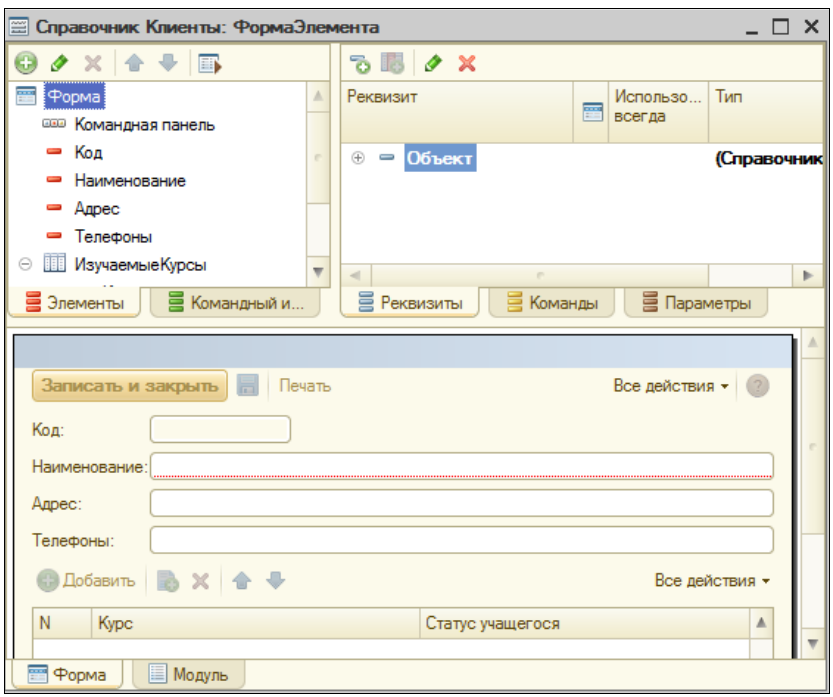


Рис. 1.45. Форма элемента справочника "Клиенты" в окне редактора форм

(рис. 1.45). Редактор форм включает много инструментальных ресурсов, о которых мы поговорим в последующих главах книги.

Сейчас же следует перейти в режим **1С:Предприятие** и внести нескольких учащихся в справочник "Клиенты" с помощью созданной формы элемента. Один из вариантов заполнения формы элемента приведен на рис. 1.46.

Создание пользовательской формы элемента справочника обусловлено тем, что сейчас мы собираемся разместить на форме кнопку **Печать** для формирования печатной формы элемента справочника. В случае же использования формы, генерируемой системой по умолчанию, такое дополнение невозможно.

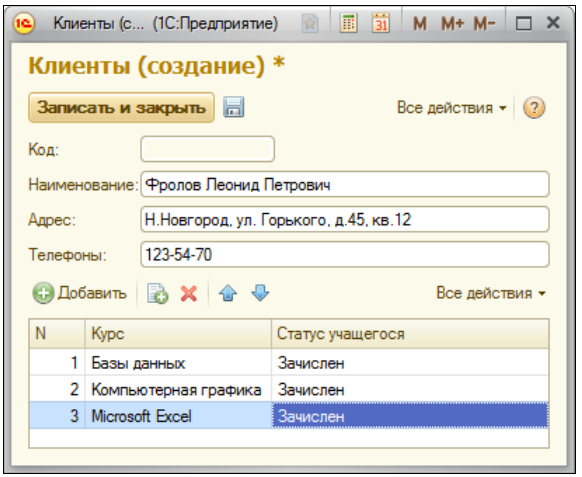


Рис. 1.46. Форма элемента справочника "Клиенты" в режиме **1С:Предприятие**

Печать элемента справочника *Клиенты*

Перейдем к созданию печатного варианта формы элемента справочника "Клиенты". Это будет табличный документ, отражающий информацию, расположенную на форме. Для этого воспользуемся возможностями конструктора печати. Вызвать конструктор печати можно в окне редактирования справочника "Клиенты" (см. рис. 1.43) на вкладке **Макеты**, где в нижней части окна располагается выпадающее меню **Конструкторы**, в котором имеется единственный пункт **Конструктор печати**. Им мы и воспользуемся.

> ***Примечание***
>
> В системе имеются различные конструкторы, облегчающие труд разработчика. Это конструкторы форм, конструкторы движений регистров, конструкторы печатных форм и др.

Первое окно диалога с конструктором печати представлено на рис. 1.47. Сам по себе диалог с конструктором печати несложный, и в данном окне от нас требуется только согласиться с предложением создать новую команду с именем **Печать**, после чего следует нажать кнопку **Далее.**

Следующее окно конструктора печати (рис. 1.48) позволяет определиться с реквизитами шапки, которые мы собираемся расположить на печатной форме элемента

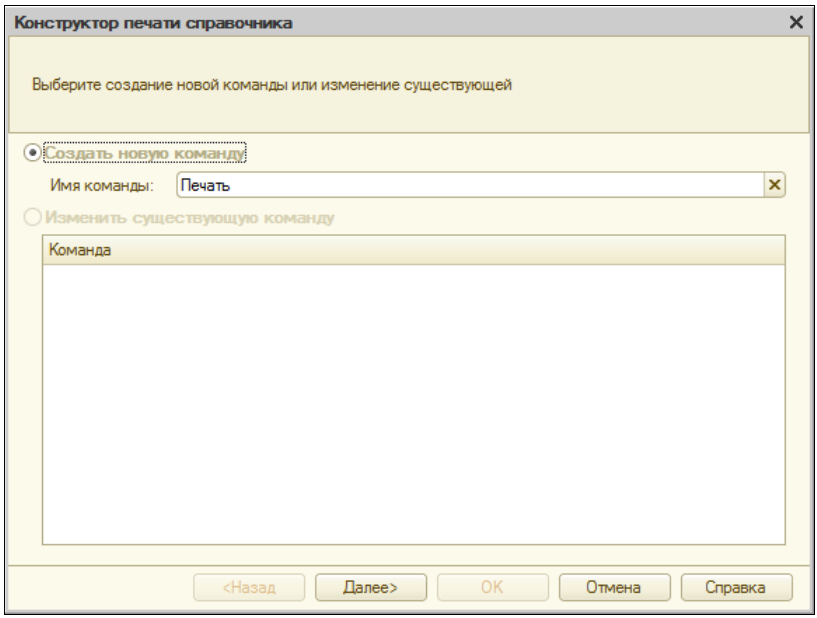


Рис. 1.47. Окно конструктора печати для создания команды печати

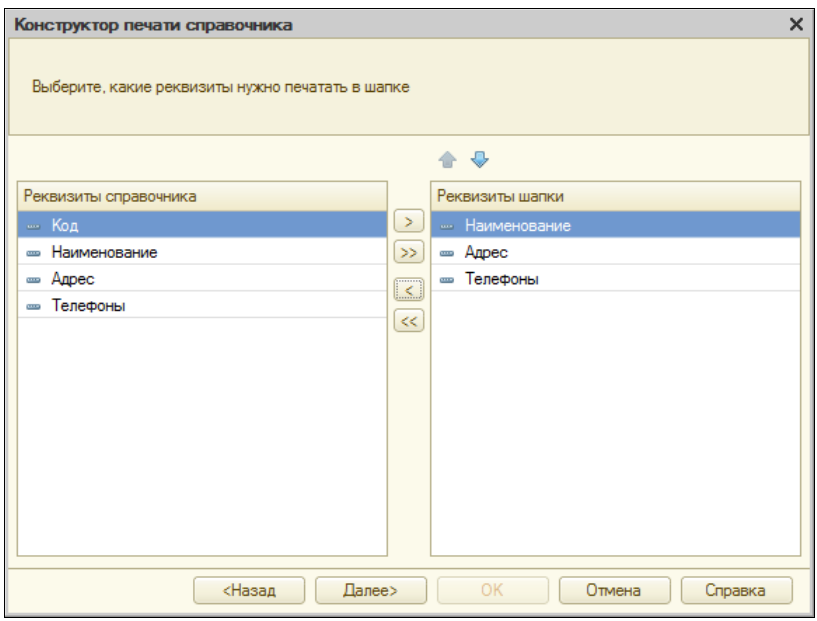


Рис. 1.48. Формирование шапки печатной формы с помощью конструктора печати

справочника. Здесь следует исключить только поле **Код**, присутствие которого в табличном документе будет лишним.

После этого в очередном окне (рис. 1.49) следует задать содержание табличной части печатной формы (включим все предлагаемые поля, кроме поля **НомерСтроки**).

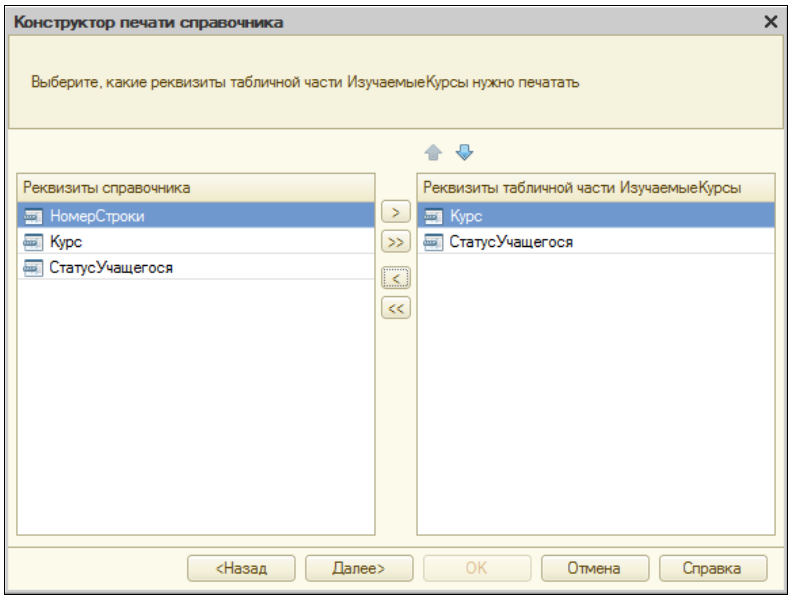


Рис. 1.49. Формирование табличной части печатного документа

Заполнение нижней части печатной формы элемента справочника (подвала) в рассматриваемом примере не понадобится (рис. 1.50).

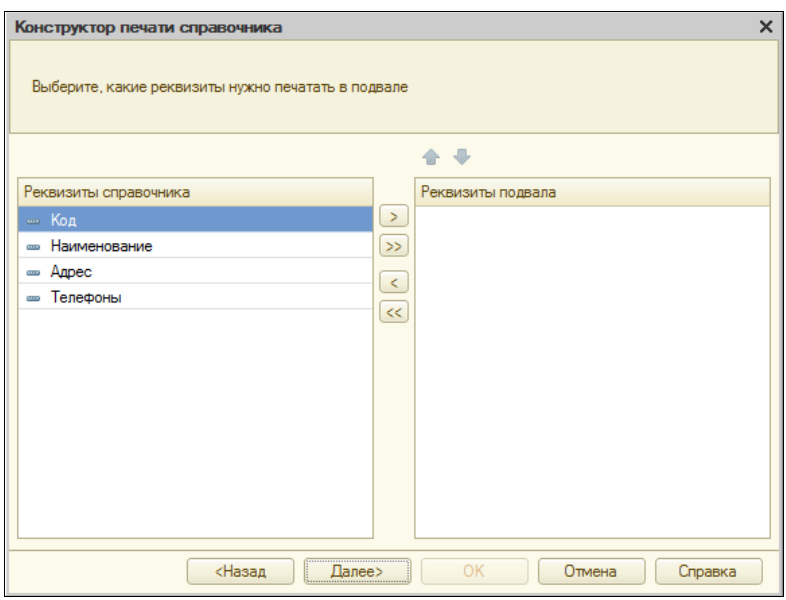


Рис. 1.50. Формирование подвала печатного документа

После щелчка по кнопке **Далее** открывается завершающее окно конструктора печати для указания группы, куда будет входить команда формирования печатной формы (рис. 1.51).

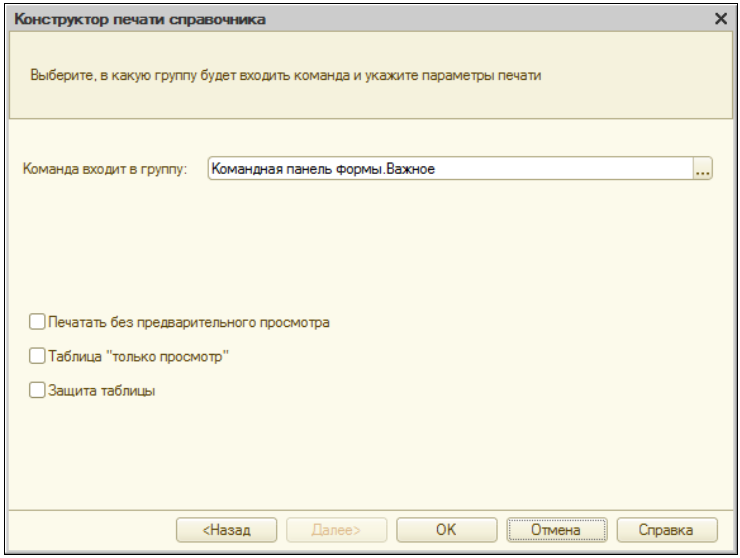


Рис. 1.51. Окно конструктора печати для указания местоположения команды печати

Примечание

Здесь речь идет об интерфейсе — о форме элемента справочника, где кнопка для формирования печатной формы будет располагаться.

Теперь осталось щелкнуть по кнопке **OK**, что приведет к завершению работы конструктора. В результате на экране мы увидим макет печатной формы, а также автоматически созданный набор программных процедур (рис. 1.52).

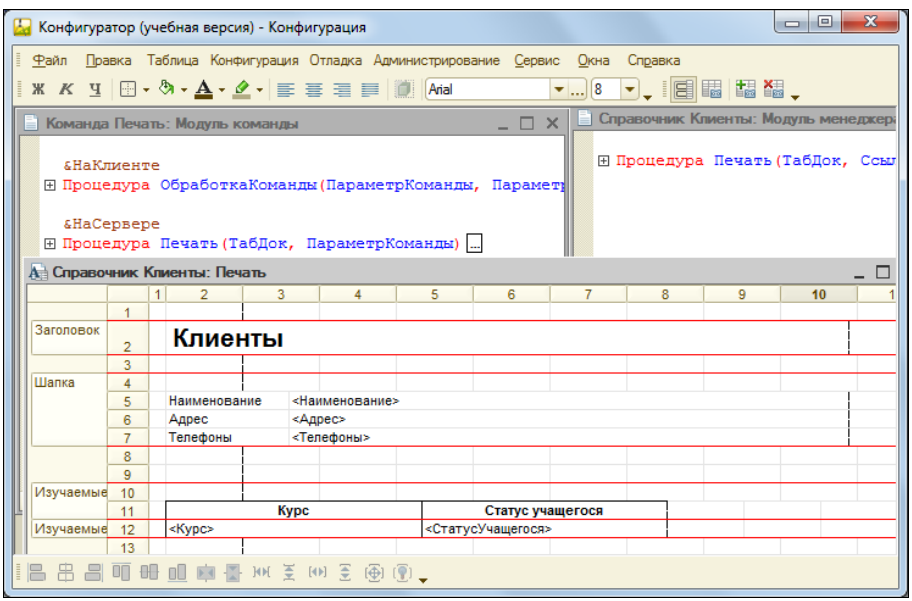


Рис. 1.52. Результат работы конструктора печати

Перед тем как проанализировать итоги работы конструктора, посмотрим на результат разработки в режиме **1С:Предприятие**. Для этого воспользуемся уже знакомым меню **Отладка**.

После обращения к форме списка справочника "Клиенты" видно, что на командной панели появилась новая кнопка **Печать** (рис. 1.53), щелчком по которой формируется печатная форма выделенного элемента справочника (рис. 1.54). Таким образом действия, выполняемые в диалоге с конструктором печати, достаточно несложные, а итог, как мы видим, весьма эффективный.

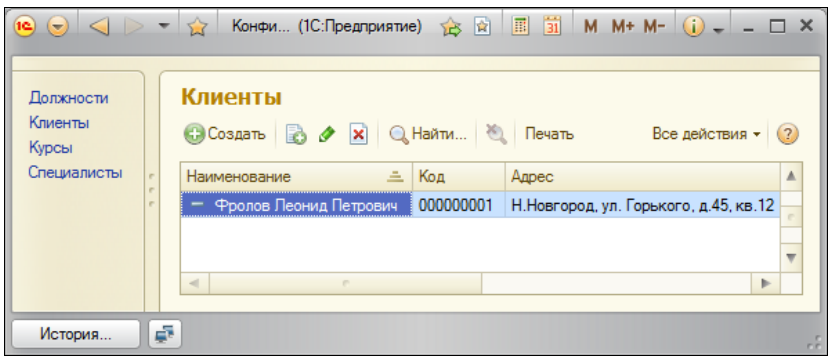


Рис. 1.53. Форма списка справочника "Клиенты" с включением кнопки **Печать**

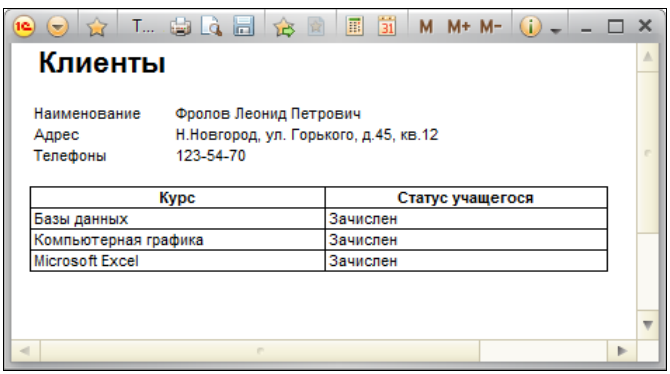


Рис. 1.54. Печатная форма элемента справочника "Клиенты"

Вернемся теперь немного назад и обсудим результаты работы конструктора печати. На рис. 1.52 показан *макет печатной формы*, представляющий собой заготовку (шаблон) для печатного документа, который будет формироваться системой при работе в режиме **1С:Предприятие**. Макет печатной формы доступен для разработчика — его всегда можно открыть и посмотреть (а при необходимости и изменить) на вкладке **Макеты** окна редактирования объекта конфигурации. В нашем случае на данной вкладке мы увидим пиктограмму нового объекта конфигурации — макета **Печать**.

Примечание

Для проектирования макета используется редактор табличного документа, входящий в систему 1С:Предприятие. В процессе создания макета в ячейки вводится текст (это может быть выполнено не только в результате работы конструктора, но и непосредственно разработчиком). При этом можно задавать различные параметры форматирования для ячеек. Печатные формы, как правило, содержат несколько типичных областей: заголовочную область, таблицу результатов, место для итоговых показателей и др. Это связано с тем, что информация, выводимая в отчеты, обычно неоднородная — есть табличные данные, заголовок, область размещения подписей. Макет представляет собой совокупность прямоугольных областей, каждая из которых служит для формирования какой-то части печатной формы. Заметим, что табличный документ (это более правильное название, чем печатная форма, но условно их можно считать синонимами) и макет — это различные объекты. Макет представляет собой некий эталон для табличного документа в плане того, как необходимо создавать печатную форму. Фактически в конфигураторе мы определяем вид печатной формы, которую собираемся затем увидеть при работе в режиме **1С:Предприятие**.

Другим результатом работы конструктора печати является набор программных процедур, которые выполняются при работе в режиме **1С:Предприятие** и взаимодействуют с макетом. В данном случае конструктором печати созданы три процедуры:

- `ОбработкаКоманды` — процедура, содержащая программный код обработки команды **Печать**, которая была также создана конструктором печати;
- `Печать` — процедура, выполняемая на сервере;
- `Печать` — процедура, расположенная в модуле менеджера справочника "Клиенты".

Примечание

Модуль — это контейнер для текста программ на встроенном языке. В конфигурации существует большое количество модулей, которые расположены в различных местах конфигурации. Тексты программ, содержащихся в модулях, используются платформой в заранее определенные моменты работы системы 1С:Предприятие.

Рассмотрим сначала процедуру (листинг 1.1), которая выполняется при обработке команды **Печать** (фактически она выполняется, когда пользователь в режиме **1С:Предприятие** производит щелчок по кнопке **Печать** в форме списка справочника "Клиенты").

Листинг 1.1. Процедура обработки команды Печать

```
&НаКлиенте
Процедура ОбработкаКоманды(ПараметрКоманды, ПараметрыВыполненияКоманды)
  //{{_КОНСТРУКТОР_ПЕЧАТИ(Печать)
  ТабДок = Новый ТабличныйДокумент;
  Печать(ТабДок, ПараметрКоманды);
  ТабДок.ОтображатьСетку = Ложь;
  ТабДок.Защита = Ложь;
  ТабДок.ТолькоПросмотр = Ложь;
  ТабДок.ОтображатьЗаголовки = Ложь;
  ТабДок.Показать();
  //}}
КонецПроцедуры
```

Процедура, представленная в листинге 1.1, располагается в модуле команды **Печать**. Модуль команды предназначен для того, чтобы описать в нем на встроенном языке те действия, которые должна выполнить платформа при вызове команды. Модуль команды может включать набор процедур и функций. Процедура `ОбработкаКоманды` обязательно должна в качестве начальной строки включать директиву компиляции `&НаКлиенте`, что означает выполнение последующего программного кода на стороне клиента.

Примечание

Подробнее об этих директивах компиляции мы поговорим далее в книге.

При рассмотрении листинга 1.1 бросается в глаза то, что несколько строк начинаются с двух наклонных черт. Это комментарий, который программно не выполняется, а предназначен лишь для пояснения разработки. В строке

```
ТабДок = Новый ТабличныйДокумент;
```

создается новый программный объект типа `ТабличныйДокумент`, а переменная `ТабДок` в данной процедуре будет с ним далее ассоциироваться. Упрощенно объект типа `ТабличныйДокумент` представляет собой таблицу, но с разнообразными свойствами и методами работы с ней. Далее в тексте процедуры `ОбработкаКоманды` производится вызов серверной процедуры `Печать` (листинг 1.2). Данная процедура в качестве начальной строки включает директиву компиляции `&НаСервере`, информирующую систему о том, что последующий программный код следует выполнять на сервере.

Листинг 1.2. Серверная процедура Печать в модуле команды

```
&НаСервере
Процедура Печать(ТабДок, ПараметрКоманды)
  Справочники.Клиенты.Печать(ТабДок, ПараметрКоманды);
КонецПроцедуры
```

Содержание процедуры в листинге 1.2 фактически представляет собой вызов другой процедуры `Печать` (листинг 1.3), расположенной в модуле менеджера справочника "Клиенты". Данный вызов отправляет последующее выполнение программного кода в модуль менеджера.

Листинг 1.3. Процедура Печать в модуле менеджера справочника

```
Процедура Печать(ТабДок, Ссылка) Экспорт
//{{_КОНСТРУКТОР_ПЕЧАТИ(Печать)
Макет = Справочники.Клиенты.ПолучитьМакет("Печать");
Запрос = Новый Запрос;
Запрос.Текст =
"ВЫБРАТЬ
  |     Клиенты.Адрес,
  |     Клиенты.Наименование,
```

```
  |     Клиенты.Телефоны,
  |     Клиенты.ИзучаемыеКурсы.(
  |            Курс,
  |            СтатусУчащегося
  |     )
  | ИЗ
  |     Справочник.Клиенты КАК Клиенты
  |ГДЕ
  |     Клиенты.Ссылка В (&Ссылка)";
  Запрос.Параметры.Вставить("Ссылка", Ссылка);
  Выборка = Запрос.Выполнить().Выбрать();
  ОбластьЗаголовок = Макет.ПолучитьОбласть("Заголовок");
  Шапка = Макет.ПолучитьОбласть("Шапка");
  ОбластьИзучаемыеКурсыШапка =
              Макет.ПолучитьОбласть("ИзучаемыеКурсыШапка");
 ОбластьИзучаемыеКурсы = Макет.ПолучитьОбласть("ИзучаемыеКурсы");
 ТабДок.Очистить();
 ВставлятьРазделительСтраниц = Ложь;
 Пока Выборка.Следующий() Цикл
    Если ВставлятьРазделительСтраниц Тогда
      ТабДок.ВывестиГоризонтальныйРазделительСтраниц();
    КонецЕсли;
    ТабДок.Вывести(ОбластьЗаголовок);
    Шапка.Параметры.Заполнить(Выборка);
    ТабДок.Вывести(Шапка, Выборка.Уровень());
    ТабДок.Вывести(ОбластьИзучаемыеКурсыШапка);
    ВыборкаИзучаемыеКурсы = Выборка.ИзучаемыеКурсы.Выбрать();
    Пока ВыборкаИзучаемыеКурсы.Следующий() Цикл
      ОбластьИзучаемыеКурсы.Параметры.Заполнить(ВыборкаИзучаемыеКурсы);
      ТабДок.Вывести(ОбластьИзучаемыеКурсы,
             ВыборкаИзучаемыеКурсы.Уровень());
    КонецЦикла;
    ВставлятьРазделительСтраниц = Истина;
   КонецЦикла;
//}}
КонецПроцедуры
```

В тексте процедуры, представленной в листинге 1.3, используется метод `Получить-Макет`, в качестве входного параметра которого указывается имя макета печатной формы, который был создан конструктором печати. Далее с помощью переменной `Макет` производится формирование табличного документа на основании нашего макета.

> ***Примечание***
>
> Более точно табличный документ является объектом, который предназначен в разработках для формирования печатных форм основных объектов (справочников, документов, регистров и др.), а также для форм, в которых информация представляется в табличном виде. Используя свойства и методы этого объекта, можно формировать необходимый для пользователя вид таблицы в режиме **1С:Предприятие**.

В тексте процедуры в листинге 1.3 используется новый объект — *запрос*, который предназначен для извлечения информации из базы данных. Этот объект имеет большое значение при разработке прикладных решений, и в последующих главах книги будут подробно рассмотрены конструкции языка запросов. В данном случае запрос используется для извлечения информации об элементе справочника "Клиенты".

Далее в процедуре листинга 1.3 производятся действия по заполнению табличного документа на основании автоматически созданного конструктором макета. Сам макет имеет имя `Печать` и включает несколько областей, одна из которых содержит заголовочную информацию и называется — `Заголовок` (см. рис. 1.52). В следующей строке программного кода в табличный документ выводится первая секция в соответствии с макетом:

```
ТабДок.Вывести(ОбластьЗаголовок);
```

Далее аналогичным образом производится заполнение последующих фрагментов табличного документа. Таким способом происходит *формирование табличного документа на основании макета*. Содержание областей макета и их названий, сформированное по умолчанию, мы в принципе можем изменить, однако в данном случае в этом нет необходимости.

Учитывая вид макета (см. рис. 1.52), можно сказать, что основное содержание печатной формы документа представляет собой совокупность одинаковых по структуре строк, подписи для которых располагаются в секции `ИзучаемыеКурсыШапка`. Метод `Выбрать`, применяемый к результату выполнения запроса, позволяет подготовить дальнейшее извлечение информации с помощью стандартных программных конструкций.

Последовательное извлечение всех элементов табличной части справочника "Курсы" производится с помощью цикла:

```
Пока ВыборкаИзучаемыеКурсы.Следующий() Цикл
```

В результате каждого прохода цикла мы получаем информацию, содержащуюся в очередной строке табличной части элемента справочника "Клиенты". Данный цикл перебора строк документа производит заполнение табличного документа информацией на основании области макета `ИзучаемыеКурсы`. Для этого используется метод `Вывести`. В результате выполнения рассматриваемого цикла в табличном документе создается количество строк, соответствующее числу строк табличной части элемента справочника "Клиенты".

Теперь вернемся к листингу 1.1 и поясним ряд программных конструкций. После выполнения серверной процедуры печати дальнейшая "доводка" табличного документа производится на стороне клиента (см. листинг 1.1). Так настройка табличного документа начинается со строки

```
ТабДок.ОтображатьСетку = Ложь;
```

в которой указывается, что выводить сетку не надо. В качестве альтернативы для свойства табличного документа `ОтображатьСетку` можно указать значение `Истина`.

Тогда для всего табличного документа (в режиме **1С:Предприятие**) будет установлена сетка.

Другое свойство `Защита` позволяет при установке значения `Истина` защитить табличный документ от редактирования и копирования. В нашем примере мы этого не сделали:

```
ТабДок.Защита = Ложь.
```

Однако если для данного свойства установить значение `Истина`, то в режиме **1С:Предприятие** скопировать табличный документ в буфер обмена не удастся.

Свойство `ТолькоПросмотр` позволяет обеспечить два варианта работы с табличным документом:

- только просмотр информации (при значении `Истина`), в этом случае будут недоступны для редактирования ячейки и рисунки, у которых установлено свойство `Защита` (производится в окне свойств ячейки, которое открывается с помощью контекстного меню);
- возможность изменения информации (при значении `Ложь`) в табличном документе при работе в режиме **1С:Предприятие**.

Свойство `ОтображатьЗаголовки` содержит признак отображения заголовков колонок и строк таблицы. Если для данного свойства установлено значение `Истина`, то в табличном документе будут выведены номера строк и колонок в верхней и левой части таблицы.

И наконец, обратим внимание на последний оператор процедуры листинга 1.1:

```
ТабДок.Показать(),
```

в которой используется метод `Показать` для отображения табличного документа на экране.

После автоматически выполненных действий конструктора осталось лишь перейти в режим работы **1С:Предприятие**, открыть форму списка справочника "Клиенты" и щелкнуть по кнопке **Печать**. На рис. 1.54 мы уже видели заполненную печатную форму одного из элементов справочника.

Теперь при желании печатную форму можно скорректировать (изменить информацию, содержащуюся в ячейках), а также сохранить как рабочую книгу Microsoft Excel (для этого следует в меню **Файл** выбрать раздел **Сохранить как** и далее указать вариант рабочей книги Microsoft Excel).

Внесение изменений в печатную форму

Попробуем самостоятельно немного изменить вид табличного документа, полученного в предыдущем разделе (см. рис. 1.54). Как известно, в печатных формах большинства документов имеется место для подписей. Наша задача заключается в оформлении нижней части печатной формы элемента справочника "Клиенты" в виде, представленном на рис. 1.55. Здесь в измененном варианте имеется нижняя часть печатной формы с местом для подписи двух руководителей.

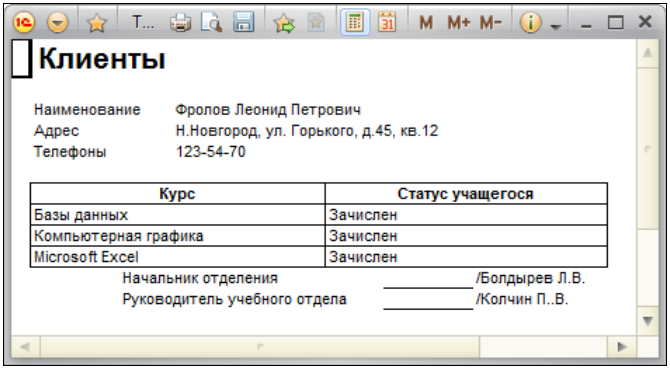


Рис. 1.55. Печатная форма элемента справочника "Клиенты" с местом для подписей

Для внесения изменений в печатную форму следует в конфигураторе открыть уже знакомый нам макет `Печать`, в котором присутствуют все поля, выводимые в табличный документ. Здесь требуется создать новую секцию табличного документа под названием **Подписи** (рис. 1.56). Для этого следует выделить две строки в нижней части макета (замаркировать заголовки строк мышью). Далее необходимо обратиться к меню **Таблица**, в котором выбрать раздел **Имена** и далее воспользоваться пунктом **Назначить имя**. В результате на экране откроется новое окно, в котором следует ввести название области — **Подписи**. В созданной области введем текст в соответствии с рис. 1.56.

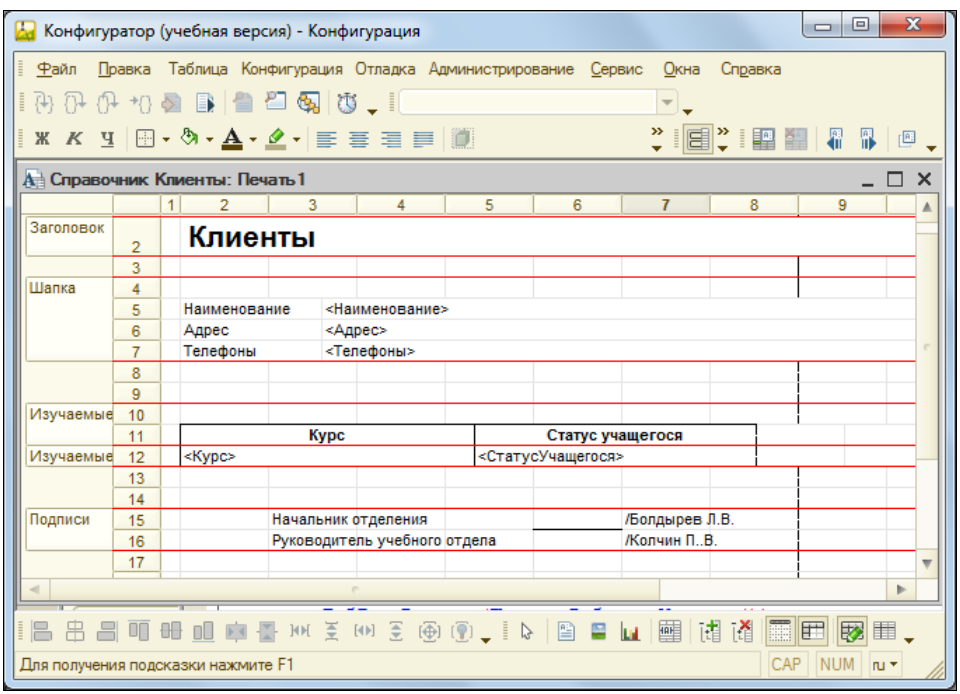


Рис. 1.56. Внесение изменений в макет табличного документа

Далее нам необходимо внести изменения в текст процедуры `Печать`, расположенной в модуле менеджера справочника "Клиенты". Измененное содержание этой процедуры представлено в листинге 1.4, где новые строки выделены более жирным шрифтом.

Листинг 1.4. Измененная процедура Печать в модуле менеджера справочника Клиенты

```
Процедура Печать(ТабДок, Ссылка) Экспорт
//{{_КОНСТРУКТОР_ПЕЧАТИ(Печать)
  Макет = Справочники.Клиенты.ПолучитьМакет("Печать");
  Запрос = Новый Запрос;
  Запрос.Текст ="ВЫБРАТЬ
  |     Клиенты.Адрес,
  |     Клиенты.Наименование,
  |     Клиенты.Телефоны,
  |     Клиенты.ИзучаемыеКурсы.(
  |            Курс,
  |            СтатусУчащегося
  |     )
  |ИЗ
  |     Справочник.Клиенты КАК Клиенты
  |ГДЕ
  |     Клиенты.Ссылка В (&Ссылка)";
  Запрос.Параметры.Вставить("Ссылка", Ссылка);
  Выборка = Запрос.Выполнить().Выбрать();
  ОбластьЗаголовок = Макет.ПолучитьОбласть("Заголовок");
  Шапка = Макет.ПолучитьОбласть("Шапка");
  ОбластьИзучаемыеКурсыШапка =
       Макет.ПолучитьОбласть("ИзучаемыеКурсыШапка");
  ОбластьИзучаемыеКурсы = Макет.ПолучитьОбласть("ИзучаемыеКурсы");
  ОбластьПодписи = Макет.ПолучитьОбласть("Подписи");
  ТабДок.Очистить();
  ВставлятьРазделительСтраниц = Ложь;
  Пока Выборка.Следующий() Цикл
     Если ВставлятьРазделительСтраниц Тогда
       ТабДок.ВывестиГоризонтальныйРазделительСтраниц();
      КонецЕсли;
  ТабДок.Вывести(ОбластьЗаголовок);
  Шапка.Параметры.Заполнить(Выборка);
  ТабДок.Вывести(Шапка, Выборка.Уровень());
  ТабДок.Вывести(ОбластьИзучаемыеКурсыШапка);
  ВыборкаИзучаемыеКурсы = Выборка.ИзучаемыеКурсы.Выбрать();
  Пока ВыборкаИзучаемыеКурсы.Следующий() Цикл
    ОбластьИзучаемыеКурсы.Параметры.Заполнить(ВыборкаИзучаемыеКурсы);
    ТабДок.Вывести(ОбластьИзучаемыеКурсы,
    ВыборкаИзучаемыеКурсы.Уровень());
  КонецЦикла;
```

```
    ТабДок.Вывести(ОбластьПодписи);
    ВставлятьРазделительСтраниц = Истина;
КонецЦикла;
//}}
КонецПроцедуры
```

После внесения указанных изменений в макет и текст процедуры следует перейти в режим **1С:Предприятие** и получить новый вариант печатной формы (см. рис. 1.55) элемента справочника "Клиенты".

Документы

Сразу отметим, что документы относятся к наиболее сложным объектам конфигурации. Как уже говорилось, они предназначены для фиксирования информации о происходящих событиях в определенном подразделении или в организации в целом. Сейчас мы познакомимся с основными особенностями объекта конфигурации *документ*, а в *главе 2* документам будет уделено большее внимание.

Документ *ВнесениеОплаты*

Наша задача в этом разделе — рассмотреть структуру документов и технические приемы их разработки в конфигураторе. Здесь мы создадим документ, который будет фиксировать информацию о текущей оплате за обучение. И первое уже знакомое действие на этом пути заключается в щелчке правой кнопкой мыши на строке **Документы** в окне **Конфигурация** и выборе пункта **Добавить** в открывшемся контекстном меню. После этого начинается несложная работа в окне редактирования объекта конфигурации. В качестве имени создаваемого объекта конфигурации введем **ВнесениеОплаты** (рис. 1.57). Этот документ будет предназначаться для фиксации внесенной оплаты за обучение конкретным учащимся за набор курсов, которые он посещает.

> ***Примечание***
>
> *Документ* — одно из основных понятий системы 1С:Предприятие [6]. Каждый объект данного типа содержит информацию о конкретной хозяйственной операции и характеризуется своим номером, датой и временем. Дата и время создания документа позволяют установить строгую временную последовательность совершения операций. В режиме **Конфигуратор** создается только структура документа, а конкретные экземпляры документов вводятся пользователем в информационную базу при работе в режиме **1С:Предприятие**. В дальнейшем для краткости вместо словосочетания *"структура документа"* будет использоваться просто слово *документ*.

Заполнение вкладки *Данные*

На рис. 1.58 показана вкладка **Данные** окна редактирования объекта конфигурации, на которой необходимо определить поля создаваемого документа. Заметим, что любой документ имеет шапку — общую или заголовочную часть, а также чаще всего одну или несколько табличных частей. В шапке размещается информация,

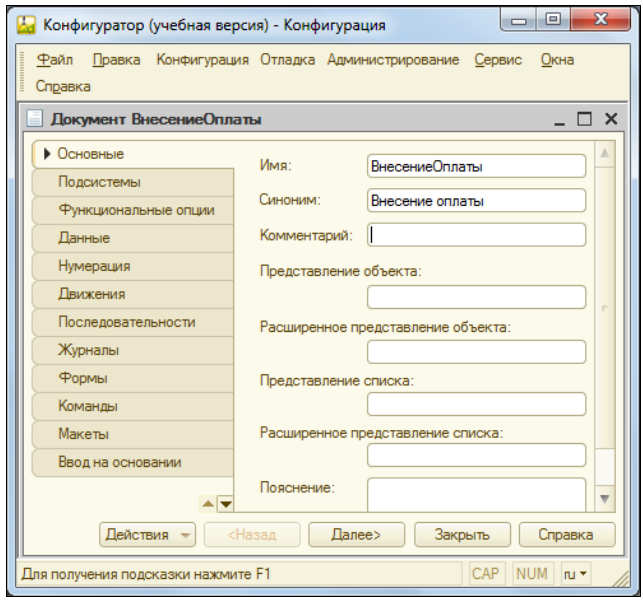


Рис. 1.57. Окно редактирования документа "ВнесениеОплаты"

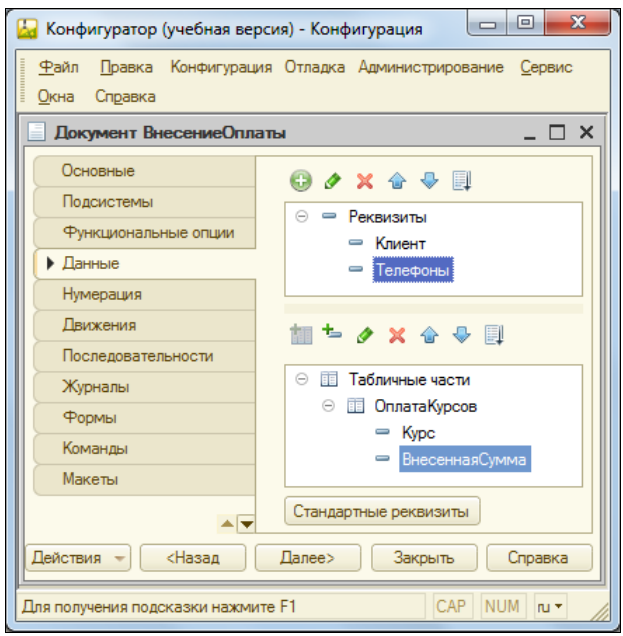


Рис. 1.58. Структура документа "ВнесениеОплаты"

характерная для всего документа в целом. В табличной части размещается перечень однотипных данных (перечень товаров, заявок, услуг и т. д.)

Для создаваемого документа "ВнесениеОплаты" в области шапки разместим два реквизита — **Клиент** и **Телефоны**. В качестве типа данных первого из них устано-

вим **СправочникСсылка.Клиенты**, а для другого выберем строковый тип данных — строку не более 50 символов.

С помощью приведенной далее процедуры (см. листинг 1.5) программным способом будет обеспечиваться автоматическое заполнение поля **Телефоны**, когда пользователь в режиме **1С:Предприятие** вносит информацию в поле **Клиент** (т. е. выбирает необходимого клиента из справочника "Клиенты").

А теперь перейдем к разработке табличной части документа. Следует заметить, что большинство реальных документов содержат один или несколько перечней каких-либо данных (строк или элементов). Это может быть список сотрудников, список номенклатуры или что-то другое. Для отображения подобной информации в объект конфигурации *документ* заложена возможность создания табличных частей.

> ***Примечание***
>
> *Табличные части* — это также объекты конфигурации, но они являются подчиненными основным объектам. С ними мы уже встречались при работе со справочниками. И в том, и в этом случае табличные части подчинены основным объектам (элементу справочника или документу). В плане программирования для объекта *табличная часть* существует многообразие свойств и методов работы с ними.

На практике для документов с несложной структурой используется одна табличная часть, но в типовых конфигурациях, созданных программистами фирмы 1С, вы найдете немало документов с включением нескольких таких частей.

После такого комментария вернемся к практике. В разрабатываемом документе "ВнесениеОплаты" создадим табличную часть, которую назовем **ОплатаКурсов**, и разместим в ней два реквизита:

- **Курс** (тип данных — **СправочникСсылка.Курсы**);
- **ВнесеннаяСумма** (тип данных — **Число**).

Форма документа

После определения структуры документа перейдем к разработке интерфейса пользователя для работы с документом, т. е. к созданию его экранной формы. Для этого в окне редактирования объекта конфигурации (рис. 1.59) выберем вкладку **Формы** и приступим к созданию формы документа. Дело в том, что в дальнейшем нам придется добавлять для расширения функционала документа собственные программные процедуры, поэтому и необходимо создать пользовательскую форму документа. Для этого щелкнем по изображению лупы в группе **Основные формы** напротив надписи **Документа** (см. рис. 1.59) и перейдем к работе с конструктором документа.

В первом появившемся окне диалога конструктора следует сделать установки (реально они должны быть выполнены по умолчанию) в соответствии с рис. 1.60. После этого, нажав кнопку **Далее**, мы увидим следующее окно конструктора (рис. 1.61), позволяющее отобразить (либо исключить) на форме определенные реквизиты документа.

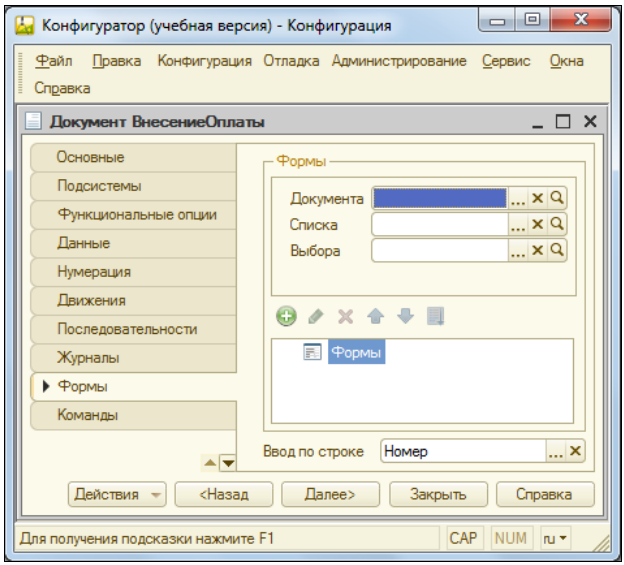


Рис. 1.59. Вкладка **Формы** окна редактирования документа

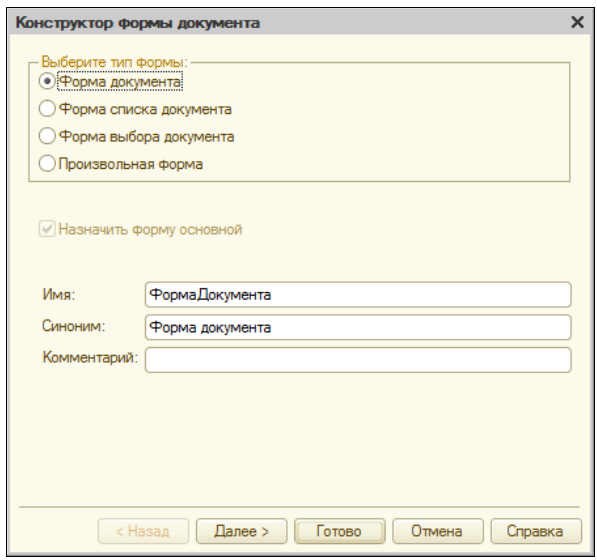


Рис. 1.60. Окно конструктора формы документа для выбора типа формы

Работа в режиме диалога с конструктором завершается нажатием кнопки **Готово**, что приводит к отображению на экране формы документа с автоматически созданными на ней элементами управления (рис. 1.62).

На этом работа с конструктором формы документа в конфигураторе завершена, и теперь требуется перейти в режим **1С:Предприятие**, где и следует заполнить несколько документов созданного вида. Один из вариантов внесения информации в документ "ВнесениеОплаты" показан на рис. 1.63.

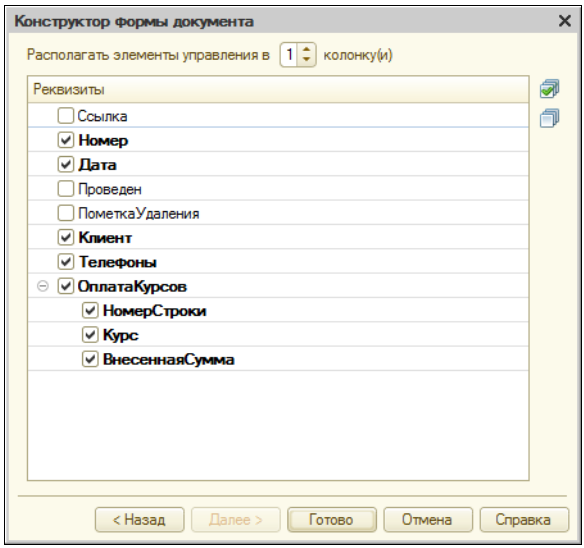


Рис. 1.61. Окно конструктора формы для определения отображаемых на форме реквизитов

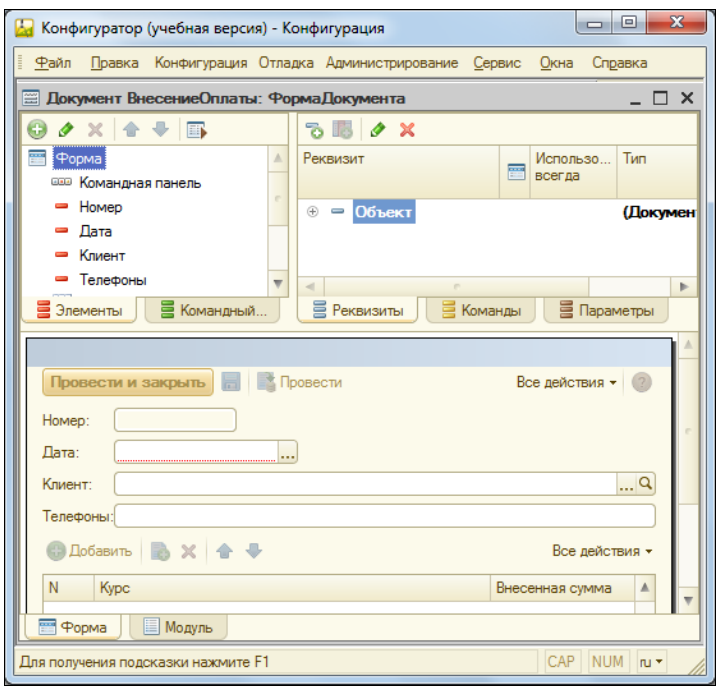


Рис. 1.62. Форма документа "ВнесениеОплаты" в окне редактора форм

После заполнения полей документа его следует сохранить и закрыть (все эти действия выполняются с помощью кнопки **Провести и закрыть**). В дальнейшем данный документ (как и все другие, созданные таким же образом) можно всегда найти (при работе в режиме **1С:Предприятие**) в списке (рис. 1.64), который открывается при выборе на панели навигации названия документа — **Внесение оплаты**.

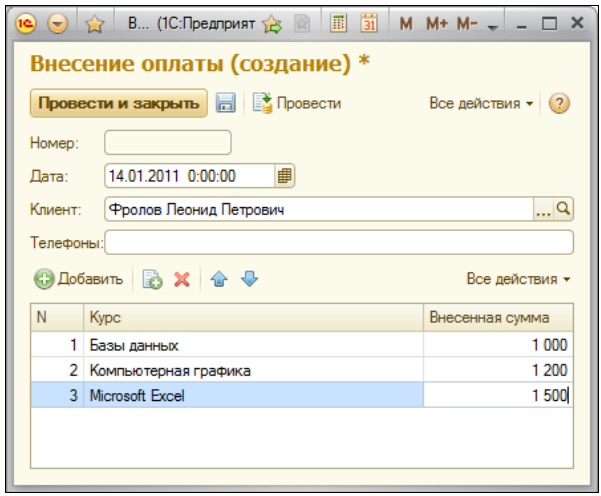


Рис. 1.63. Форма документа "ВнесениеОплаты" в режиме **1С:Предприятие**

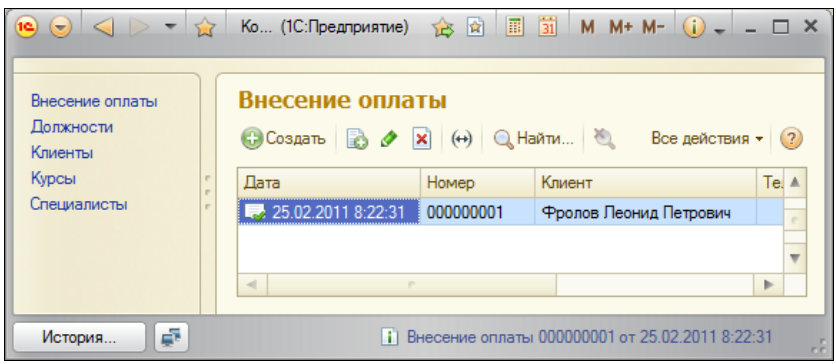


Рис. 1.64. Форма списка документа "ВнесениеОплаты" в режиме **1С:Предприятие**

Программирование формы документа

Обеспечим небольшую автоматизацию работы пользователя в режиме **1С:Предприятие**, о которой мы уже упоминали. В созданном документе "ВнесениеОплаты" два поля, **Клиент** и **Телефоны**, связаны между собой. А именно: при заполнении поля **Клиент** значением из справочника "Клиенты" соответствующий телефон должен автоматически фиксироваться в поле **Телефоны**. Наличие телефона в шапке рассматриваемого документа удобно, т. к. в этом случае по конкретному документу сразу видно, как с человеком связаться. Кроме того, автоматизация заполнения поля **Телефоны** приведет к ускорению работы пользователя и уменьшению ошибок ввода. Для такого сервиса мы напишем короткую процедуру.

Итак, перейдем непосредственно к разработке. В окне редактирования формы документа необходимо двойным щелчком по полю **Клиент** (рис. 1.65) открыть палитру его свойств. Здесь напротив имени события `ПриИзменении` следует щелкнуть

по пиктограмме с изображением лупы, в результате чего мы попадем в окно программного кода процедуры, автоматически выполняемой при выборе учащегося в поле **Клиент**. Это фактически пока еще только заготовка будущей процедуры, которую мы напишем. В дальнейшем ее всегда можно найти на вкладке **Модуль**, где размещаются все программные процедуры, работающие с экранной формой документа.

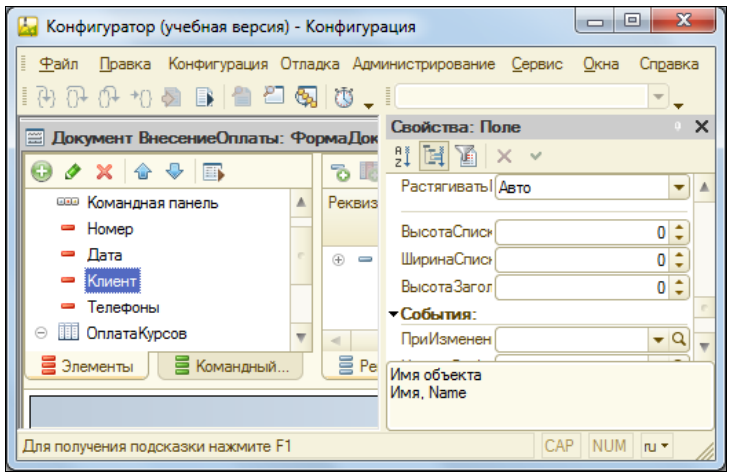


Рис. 1.65. Палитра свойств поля **Клиент**

Сам текст процедуры, которую необходимо написать для решения поставленной задачи, приведен в листинге 1.5.

Листинг 1.5. Процедура, выполняемая при заполнении поля Клиент

```
&НаКлиенте
Процедура КлиентПриИзменении(Элемент)
  Объект.Телефоны=ПолучитьТелефоны();
КонецПроцедуры
```

С помощью реквизита формы `Объект` мы можем обращаться к полям документа для извлечения и внесения информации. При этом в контексте клиента (что определяется директивой `&НаКлиенте`) мы ограничены в использовании программных конструкций языка. В частности, для того чтобы извлечь из справочника "Клиенты" номера телефонов учащегося, необходимо выполнить программный код в контексте сервера.

> ***Примечание***
>
> Выбор учащегося в поле **Клиент** выполняется в клиентской части платформы. Однако в клиентской части платформы нельзя получить значение реквизита этой ссылки.

В связи с этим мы должны написать функцию `ПолучитьТелефоны`, которая будет выполняться на сервере, и ее текст приведен в листинге 1.6.

Листинг 1.6. Функция ПолучитьТелефоны, выполняемая в контексте сервера

```
&НаСервере
Функция ПолучитьТелефоны()
   Возврат Объект.Клиент.Телефоны;
КонецФункции
```

Примечание

Функция `ПолучитьТелефоны` располагается в модуле формы документа "ВнесениеОплаты" (рядом с процедурой `КлиентПриИзменении`).

Теперь если мы перейдем в режим **1С:Предприятие** и начнем создавать новый документ "ВнесениеОплаты", то телефоны учащегося будут внесены в соответствующее поле автоматически после выбора конкретного учащегося (рис. 1.66).

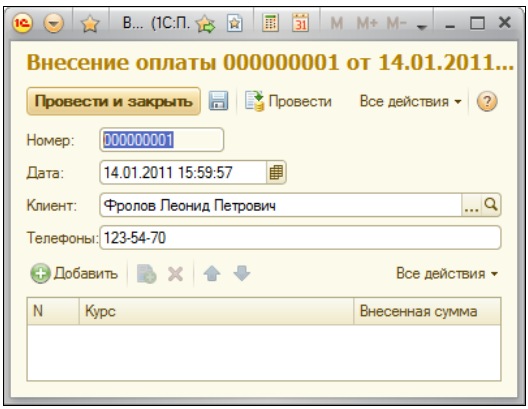


Рис. 1.66. Заполнение поля **Телефоны** после выбора учащегося

Можно предложить другой вариант заполнения поля **Телефоны**. В этом случае все действия и по извлечению информации, и по заполнению поля **Телефоны** будут производиться на сервере. Текст клиентской процедуры для этого варианта приведен в листинге 1.7.

Листинг 1.7. Процедура, выполняемая при заполнении поля Клиент (вариант 2)

```
&НаКлиенте
Процедура КлиентПриИзменении(Элемент)
  ЗаполнитьТелефоны();
КонецПроцедуры
```

Теперь вместо функции мы должны написать процедуру `ЗаполнитьТелефоны`, выполняемую на сервере (листинге 1.8).

Листинг 1.8. Процедура ЗаполнитьТелефоны, выполняемая в контексте сервера

```
&НаСервере
Процедура ЗаполнитьТелефоны()
   Объект.Телефоны=Объект.Клиент.Телефоны;
КонецПроцедуры
```

Особенностью приведенных примеров (листинги 1.6 и 1.8) является передача контекста формы от клиента на сервер. Сам контекст достаточно емкий и поэтому для передачи требуется время.

Можно реализовать взаимодействие клиентской и серверной части более эффективно. Данный вариант представлен в листингах 1.9 (клиентская часть) и 1.10 (серверная часть). Здесь мы воспользовались другой директивой `НаСервереБезКонтекста`, что позволяет выполнить программный код в контексте сервера, но сам контекст формы на сервер при этом не передается.

Листинг 1.9. Процедура, выполняемая при заполнении поля Клиент (вариант 3)

```
&НаКлиенте
Процедура КлиентПриИзменении(Элемент)
  Объект.Телефоны=ПолучитьТелефоны(Объект.Клиент);
КонецПроцедуры
```

Листинг 1.10. Функция ЗаполнитьТелефоны, выполняемая на сервере

```
&НаСервереБезКонтекста
Функция ПолучитьТелефоны(Учащийся)
   Возврат Учащийся.Телефоны;
КонецФункции
```

Таким образом, практически всегда можно реализовать программирование форм несколькими способами. Предпочтительным всегда является наиболее эффективный вариант. В [7] подробно разбираются различные приемы эффективного программирования форм.

Продолжим дальше работу по усовершенствованию нашего документа. Следующая цель — обеспечить автоматическое заполнение его табличной части информацией о курсах после выбора клиента (автоматически внести в табличную часть названия курсов, на которые записался клиент).

Для этого необходимо внести в уже написанную процедуру (точнее в одну из написанных) `КлиентПриИзменении` новые программные конструкции. В листинге 1.11 показана измененная процедура `КлиентПриИзменении`.

Листинг 1.11. Измененная процедура, выполняемая при заполнении поля Клиент

```
&НаКлиенте
Процедура КлиентПриИзменении(Элемент)
   Объект.Телефоны=ПолучитьТелефоны(Объект.Клиент);
   ЗаполнениеТабЧасти();
КонецПроцедуры
```

Здесь мы использовали вызов дополнительной процедуры `ЗаполнениеТабЧасти`, которую нам также необходимо создать. Данная процедура выполняется в контексте сервера, текст приведен в листинге 1.12.

Листинг 1.12. Процедура для заполнения табличной части документа ВнесениеОплаты

```
&НаСервере
Процедура ЗаполнениеТабЧасти()
  Объект.ОплатаКурсов.Очистить();
  Для Каждого ТекСтрока Из Объект.Клиент.ИзучаемыеКурсы Цикл
      НоваяСтрока=Объект.ОплатаКурсов.Добавить();
      НоваяСтрока.Курс=ТекСтрока.Курс;
  КонецЦикла
КонецПроцедуры
```

На рис. 1.67 показан результат выполнения новых процедур `КлиентПриИзменении` и `ЗаполнениеТабЧасти` в режиме **1С:Предприятие** — после выбора учащегося в поле **Клиент** его телефон переносится в шапку документа, а табличная часть документа автоматически заполняется названиями посещаемых им курсов.

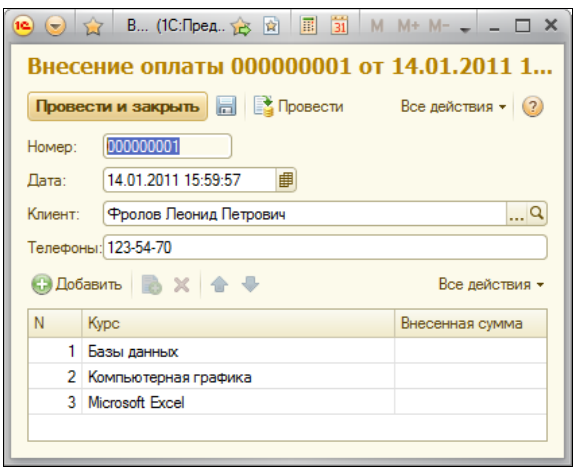


Рис. 1.67. Автоматическое заполнение телефона и табличной части документа после выбора клиента

Разберем теперь непосредственно текст листинга 1.12. В начале процедуры к табличной части документа ("ОплатаКурсов") применяется метод `Очистить`. Если этого не сделать, то табличная часть будет дополняться названиями курсов при последовательных щелчках по полю **Клиент**. Таким образом, в процедуре листинга 1.12 она сначала очищается, а затем заполняется вновь. В цикле, начинающемся со строки

```
Для Каждого ТекСтрока Из Объект.Клиент.ИзучаемыеКурсы Цикл,
```

последовательно перебираются все строки табличной части элемента справочника "Клиенты", и при каждом проходе цикла в табличную часть документа добавляется

один из курсов, на которые записался учащийся. Для этого используется метод `Добавить`. В результате пользователь, выбрав в форме документа конкретного учащегося, автоматически получает заполненную названиями посещаемых им курсов табличную часть документа, и остается только внести данные об оплаченных суммах (рис. 1.68).

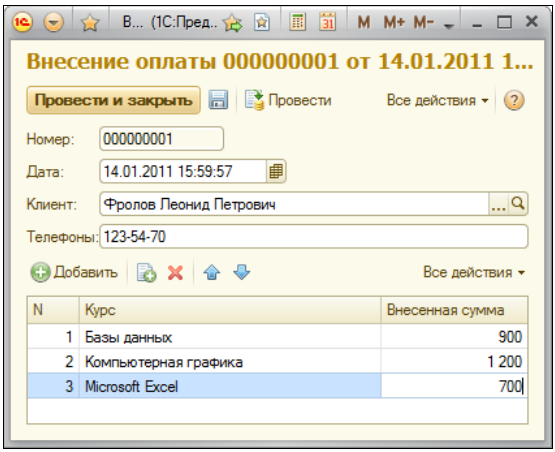


Рис. 1.68. Заполненный документ "ВнесениеОплаты" в режиме **1С:Предприятие**

Примечание

При необходимости в автоматически заполненную табличную часть документа "ВнесениеОплаты" можно добавить дополнительно еще одну или несколько строк. Это может быть выполнено с помощью стандартной кнопки пиктографического меню формы документа.

Создание печатной формы документа *ВнесениеОплаты*

В предыдущих разделах рассматривалось создание печатной формы элемента справочника "Клиенты" с помощью конструктора печати. Этот же конструктор мы используем при разработке печатной формы документа "ВнесениеОплаты".

Перейдем на вкладку **Макеты**, где воспользуемся знакомой последовательностью кнопок **Конструкторы**, а затем **Конструктор печати**. Последующие окна диалога с конструктором нам уже знакомы (по разработке печатной формы элемента справочника "Клиенты"), поэтому комментарий не будет таким детальным.

Так, в первом окне конструктора печати следует согласиться с именем команды **Печать**. Далее на экране откроется окно (рис. 1.69), в котором следует определиться с реквизитами шапки печатной формы документа. После этого сформируем табличную часть печатной формы в соответствии с рис. 1.70. Подвал (нижняя часть) табличного документа нам не понадобится и в завершающем окне конструктора печати следует определиться с месторасположением кнопки печати (рис. 1.71). На этом работа с конструктором печати завершена и щелчком по кнопке **ОК** мы получаем на экране результаты:

- макет табличного документа;
- программные процедуры;

- команду печати;
- кнопку на форме документа для реализации печати.

Результат разработки табличного документа представлен на рис. 1.72, а на рис. 1.73 приведена печатная форма документа в режиме **1С:Предприятие**.

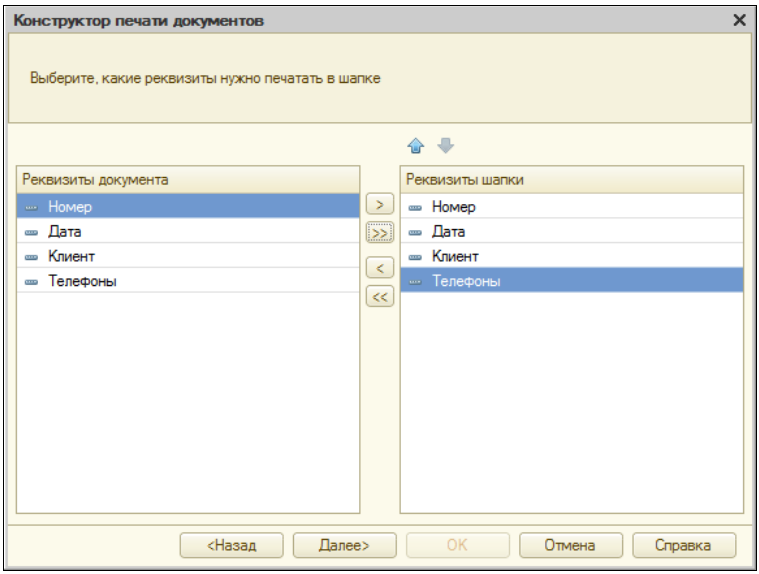


Рис. 1.69. Формирование шапки печатной формы документа "ВнесениеОплаты"

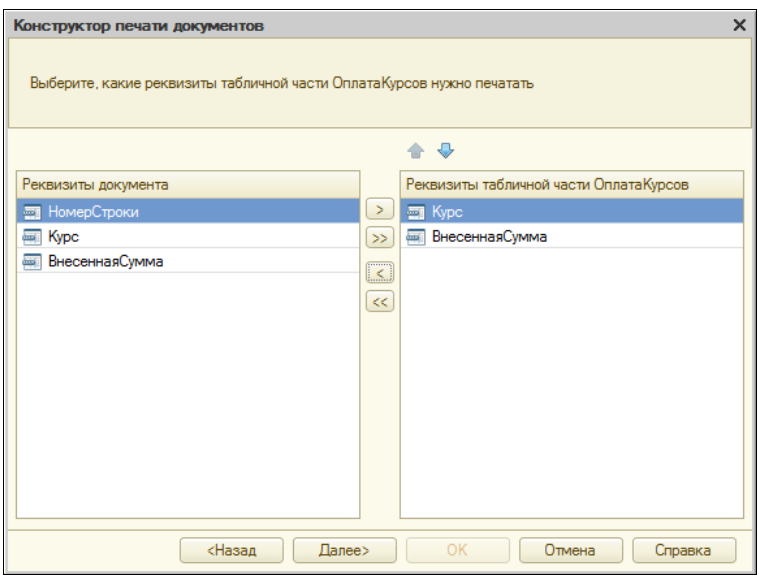


Рис. 1.70. Формирование табличной части печатной формы документа "ВнесениеОплаты"

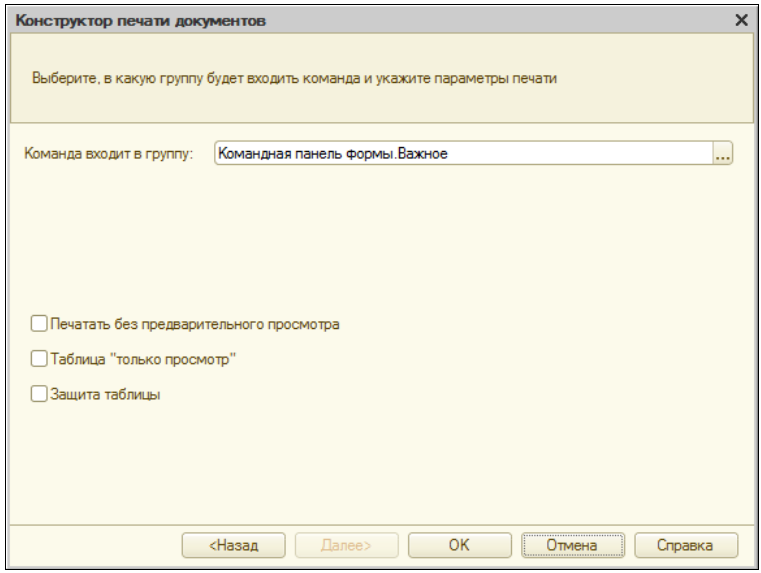


Рис. 1.71. Задание месторасположения кнопки печати

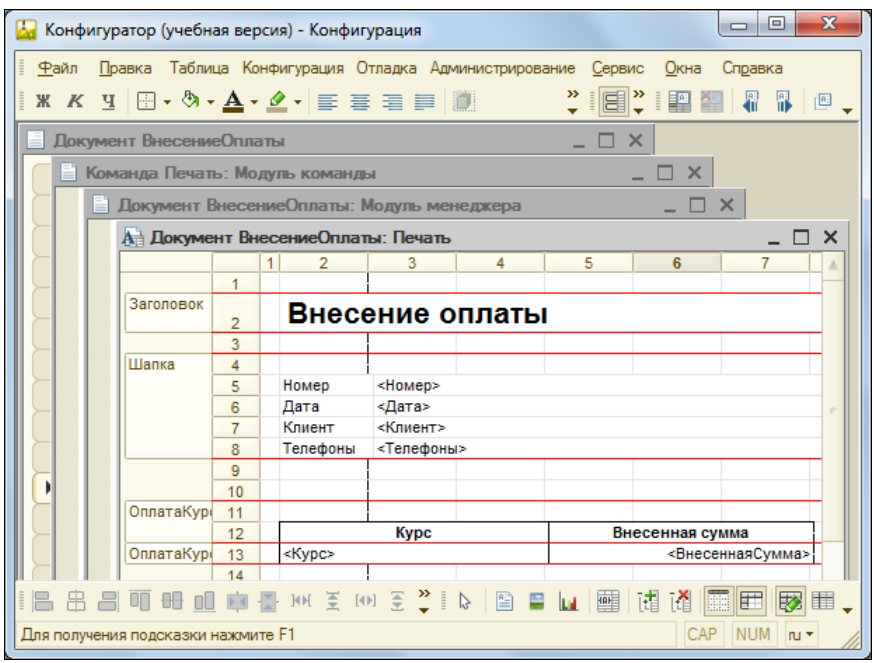


Рис. 1.72. Результат работы конструктора печати

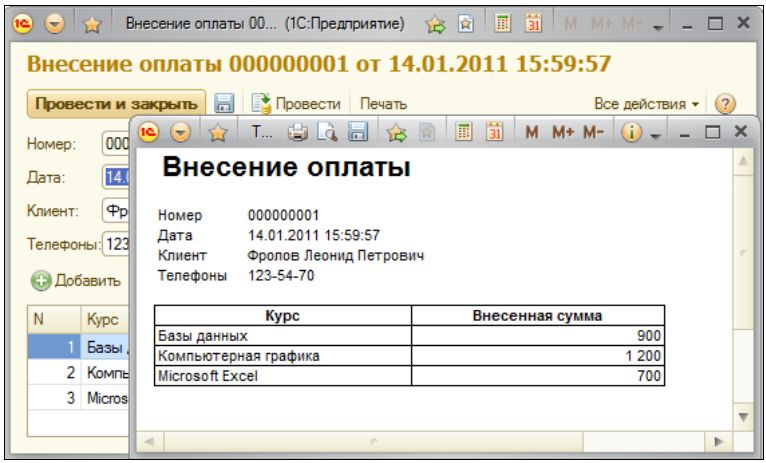


Рис. 1.73. Печатная форма документа "ВнесениеОплаты"

Подсистемы

К настоящему моменту созданные нами прикладные объекты при работе в режиме **1С:Предприятие** располагались в одном месте. Это вполне удобно, когда количество объектов небольшое и с текущей информационной базой работает один сотрудник. Понятно, что когда количество таких объектов значительно увеличивается, то возникает проблема придания прикладному решению некой структурности. Также если с информационной базой работает несколько человек (разных по профилю деятельности), то желательна оптимизация информационной системы для каждого специалиста. Для решения такой задачи в версии 1С:Предприятие 8.2 имеются подсистемы.

Подсистема представляет собой основной элемент построения интерфейса информационной системы 1С:Предприятие. Как правило, любое более или менее серьезное прикладное решение подразумевает его использование для различных категорий пользователей. Это могут быть менеджеры, бухгалтеры, управленческий персонал и т. д. В этом случае для каждой такой категории пользователей программа должна быть удобна в плане интерфейса. Удобство подразумевает включение в ассортимент только того, что требуется конкретному пользователю. Так, менеджеру не нужны объекты, с которыми работает только бухгалтер, и наоборот. Для организации подобного сервиса и предназначены подсистемы. Разумеется, если конфигурация очень простая, то ее разбивки на подсистемы не требуется. Однако практически все реальные разработки становятся более удобными для пользователей при использовании деления на подсистемы.

Рассмотрим небольшую модернизацию уже разработанной конфигурации с включением в ее состав двух подсистем:

- **НачальникУчебногоОтдела;**
- **МенеджерПоОплатам.**

Таким образом, нашей задачей сейчас является разработка этих двух подсистем. Для этого в конфигураторе откроем уже знакомое окно **Конфигурация** и обратимся к ветке дерева объектов конфигурации под названием **Общие** (рис. 1.74). Для этого достаточно лишь распахнуть ее щелчком мыши по изображению знака "плюс".

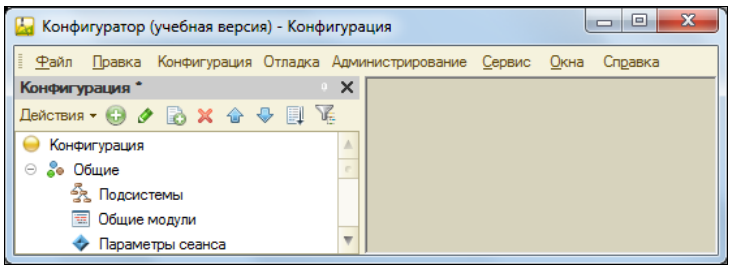


Рис. 1.74. Дерево объектов конфигурации

Технические действия по созданию новой подсистемы достаточно несложные. Для начала необходимо выделить ветвь **Подсистемы** и щелкнуть на ней правой кнопкой мыши. В результате на экране появится контекстное меню, где следует выбрать пункт **Добавить**. После этого вы увидите *окно редактирования объекта конфигурации* (рис. 1.75), которое в данном случае предназначено для установки необходимых параметров подсистемы.

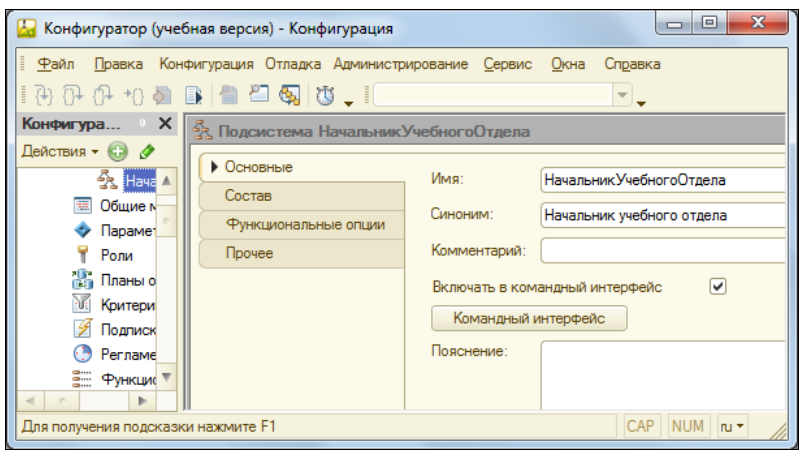


Рис. 1.75. Окно редактирования объекта конфигурации

От нас требуется ввести словосочетание **НачальникУчебногоОтдела** в поле **Имя**. Аналогичным образом создадим еще одну подсистему (рис. 1.76), которая требуется в соответствии с поставленной задачей.

Таким образом, первая подготовительная часть модернизации нашей разработки выполнена — необходимые подсистемы созданы.

Далее необходимо соотнести уже разработанные объекты конфигурации (перечисление, справочники и документ) к этим двум подсистемам. Для начала обратимся

к справочнику "Должности". На рис. 1.77 показана вкладка **Подсистемы** окна редактирования данного объекта конфигурации. Здесь необходимо с помощью флажков установить те подсистемы, в которых мы хотим обеспечить отображение данного объекта конфигурации. В рассматриваемом случае по логике работы учебного центра установим флаг напротив надписи **НачальникУчебногоОтдела**. Аналогичным образом следует "распределить" остальные созданные объекты конфигурации по двум подсистемам. Определим следующее соответствие:

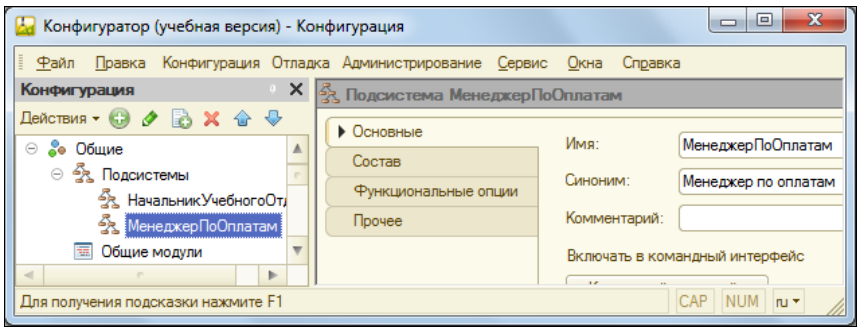


Рис. 1.76. Создание подсистемы для менеджера по оплатам

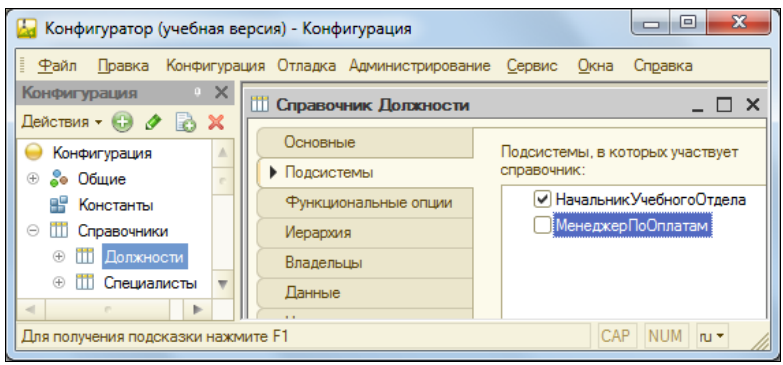


Рис. 1.77. Вкладка **Подсистемы** окна редактирования справочника "Должности"

- справочник "Клиенты" — подсистемы **НачальникУчебногоОтдела** и **МенеджерПоОплатам**;
- справочник "Курсы" — подсистема **НачальникУчебногоОтдела**;
- справочник "Специалисты" — подсистема **НачальникУчебногоОтдела**;
- документ "ВнесениеОплаты" — подсистема **МенеджерПоОплатам**;
- перечисление **Статус** — подсистема **НачальникУчебногоОтдела**.

Технически сама установка данных соответствий производится с помощью включения тех или иных флажков (рис. 1.77).

Теперь перейдем в режим **1С:Предприятие** и посмотрим, как выглядит наше приложение при использовании в конфигурации двух созданных подсистем (рис. 1.78).

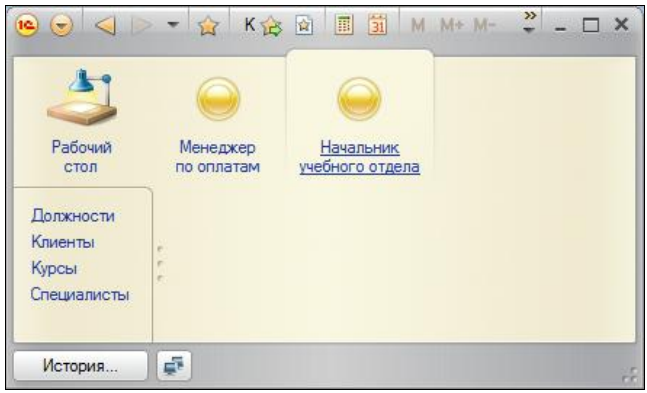


Рис. 1.78. Окно приложения с использованием двух подсистем

Видно, что изменения заметные и их необходимо пояснить. Сразу под главным меню располагается *панель разделов приложения*, где и нашли свое место созданные нами подсистемы. Разделы приложения организованы в виде гиперссылок, с помощью которых можно получить доступ к объектам, входящим в эти разделы.

Надо отметить, что на панели разделов приложения нашли отражение подсистемы верхнего уровня. Кроме них в конфигурации можно создать подчиненные подсистемы. Для них разделы приложения не создаются, а формируются подразделы, которые отражаются на панели навигации.

Примечание

Представленное на рис. 1.78 окно является основным окном приложения. С его помощью пользователь осуществляет навигацию по всему прикладному решению. Кроме основного окна, при работе на экране отображается вспомогательное окно приложения, которое обеспечивает работу с данными (просмотр и ввод документов, отчетов и т. д.).

Рядом с созданными нами подсистемами на рис. 1.78 присутствует еще и пиктограмма с подписью **Рабочий стол**. Этот раздел формируется системой автоматически и его следует использовать для размещения наиболее часто используемых документов и отчетов.

Кроме рабочего стола, остальные выше перечисленные разделы являются специфическими для решаемой прикладной задачи. В данном случае два раздела являются следствием разделения разработанных объектов конфигурации для двух потенциальных пользователей разработки. При необходимости можно создать столько разделов, сколько требуется.

Примечание

Рабочий стол представляет собой стандартный раздел приложения, содержащий часто используемые объекты (это могут быть справочники, документы или отчеты).

Подсистемы позволяют выделить в прикладном решении специфические функциональные ресурсы для различных категорий пользователей. Так, для директора в работе требуются определенные отчеты, а бухгалтер работает со своими документа-

ми. В связи с этим объект конфигурации *подсистема* занимает в платформе 1С:Предприятие 8.2 ключевое место.

В левой части на рис. 1.78 отображен важный элемент нового интерфейса — *панель навигации*. На этой панели отражены объекты, которыми располагает пользователь при выбранном интерфейсе. Фактически содержание навигации представляет собой совокупность команд (команды в данном контексте — это действия, которые может выполнять пользователь). При использовании на панели навигации команд открытия списков автоматически отображаются соответствующие управляемые формы списков в рабочей области окна приложения. На рис. 1.79 рабочую область окна приложения занимает форма списка справочника "Клиенты".

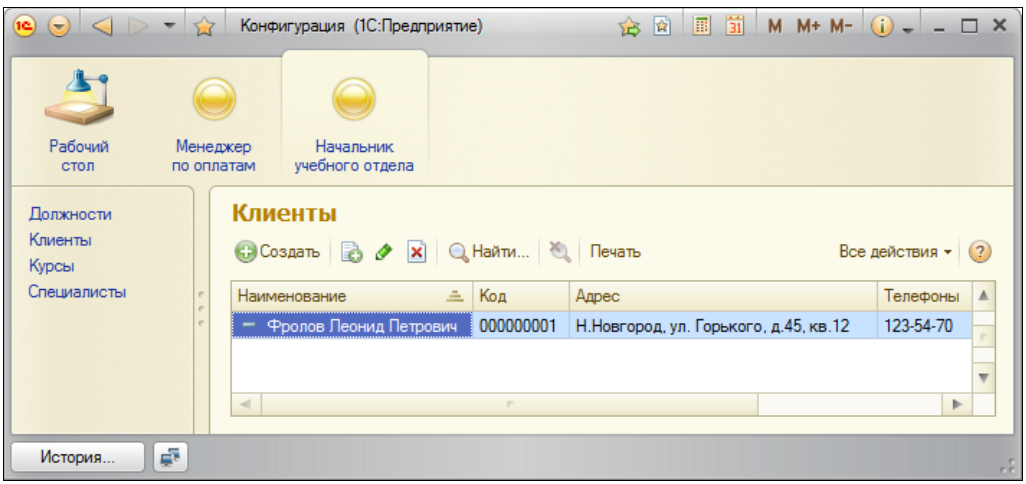


Рис. 1.79. Форма списка справочника "Клиенты" в рабочей области окна приложения

Кроме панели разделов имеется еще панель действий, которая включает наиболее популярные действия со стороны пользователя. Для того чтобы создать элементы на панели действий, вернемся в конфигуратор. В дереве объектов конфигурации обратимся к подсистеме **МенеджерПоОплатам**. Здесь с помощью правой кнопки мыши вызовем контекстное меню данной подсистемы и откроем пункт **Открыть командный интерфейс** (рис. 1.80). В открывшемся окне следует установить следующие флажки:

- **Внесение оплаты: создать**;
- **Клиенты: создать**.

После этого перейдем в режим **1С:Предприятие** и обратимся к разделу **МенеджерПоОплатам** (рис. 1.81). В результате в центральной части окна (рабочей области) мы увидим панель действий, содержащую две команды.

Команды, размещенные на панели действий, позволяют быстро создать элемент справочника, добавить новый документ, сформировать отчет. Это ускоряет работу пользователя, сокращая технические действия до минимума.

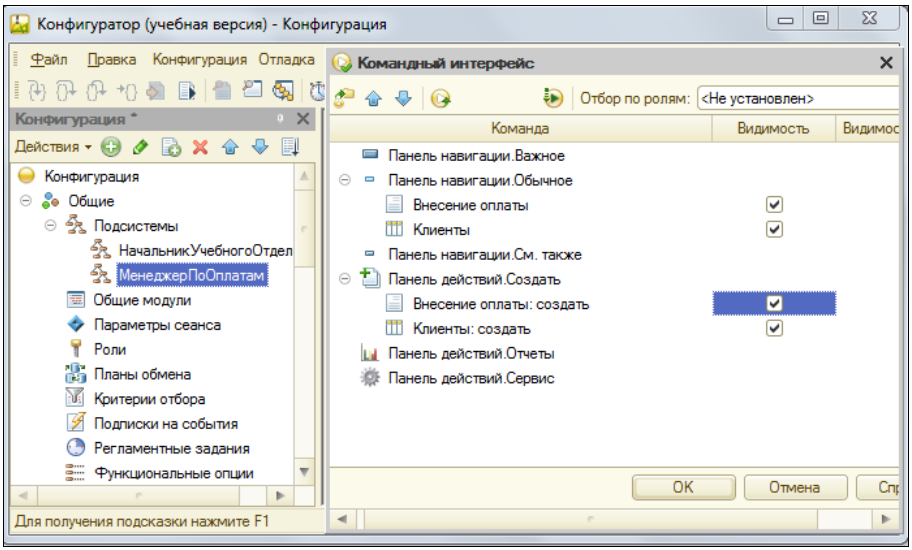


Рис. 1.80. Командный интерфейс подсистемы **МенеджерПоОплатам**

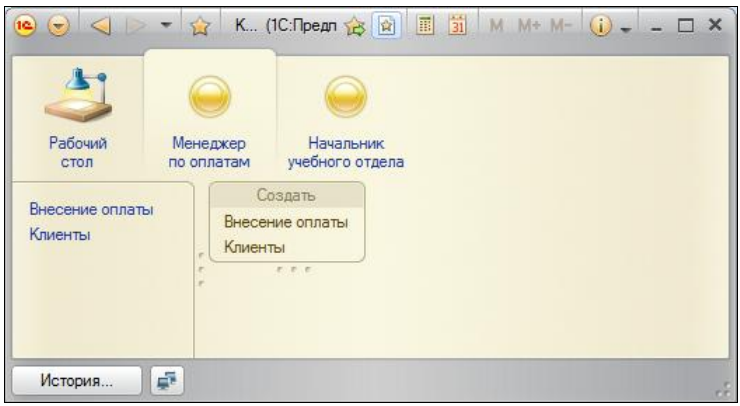


Рис. 1.81. Изменения в разделе **МенеджерПоОплатам**

Информационная панель

Рассмотрим окно режима **1С:Предприятие** на рис. 1.82, где в нижней части окна приложения присутствует строка с информацией о фамилиях, а также о названии одного из курсов. Эта строка представляет собой еще один элемент интерфейса — *информационную панель*. Основное назначение информационной панели заключается в формировании сведений о последних действиях, выполненных в системе. Если выполненное действие было связано с записью данных, то, щелкнув по гиперссылке оповещения, можно открыть форму, содержащую данные. Применительно к рис. 1.82 с помощью первой (справа) гиперссылки оповещения производится открытие формы элемента справочника "Клиенты" с внесенной записью о Маслове Дмитрии Николаевиче.

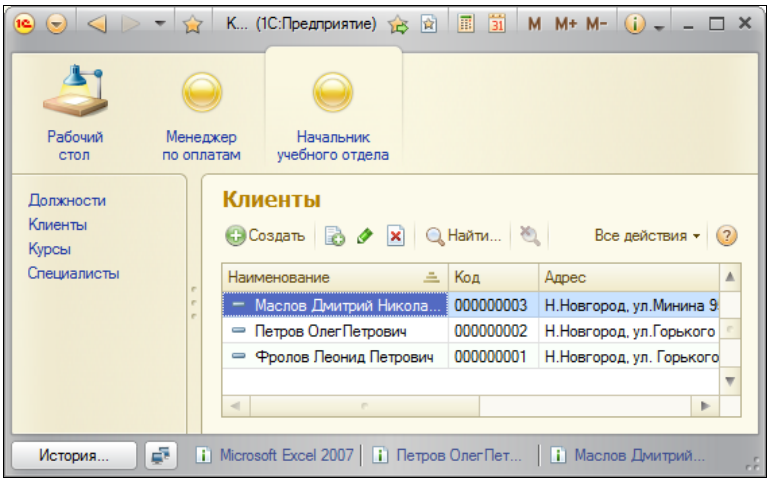


Рис. 1.82. Содержание информационной панели с отражением последних действий

Примечание

Последние оповещения об изменении объектов информационной базы отображаются в правой части информационной панели для быстрого доступа к ним.

В левой части информационной панели присутствует важная кнопка **История**, с помощью которой можно открыть историю работы пользователя с программой (рис. 1.83). Это является особенностью новой версии системы. Важно отметить, что такое запоминание не просто пассивное — система позволяет выполнить возврат к событиям из окна **История**. Например, щелчок по гиперссылке **Внесение оплаты 000000002 от 16.02.2011** приводит к открытию формы соответствующего документа. Таким образом всегда можно вернуться и проверить внесенные данные.

Если один конкретный объект информационной базы изменялся несколько раз, то в истории будет храниться единственная запись о последнем изменении. Важно отметить, что окно **История** хранит не более 200 записей и поэтому при внесении новых данных старые удаляются.

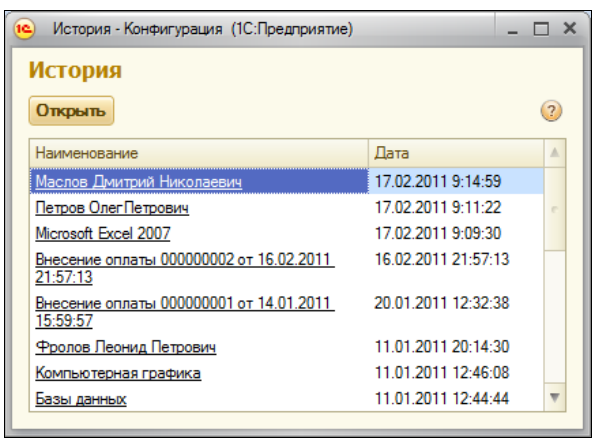


Рис. 1.83. История работы пользователя с программой

Примечание

В окно **История** попадают только действия, которые пользователь выполнил интерактивно. Если объекты системы были изменены программно, то подобные события в истории не фиксируются.

Системные команды и главное меню

Обратимся теперь к заголовку окна приложения (рис. 1.84), где присутствует область *системных команд*. В левой части данной области располагается кнопка (для перехода к главному меню). Правее находятся команды для перемещения по истории работы в основном окне: кнопка (**Вперед**), кнопка (**Назад**) и меню (**Избранное**).

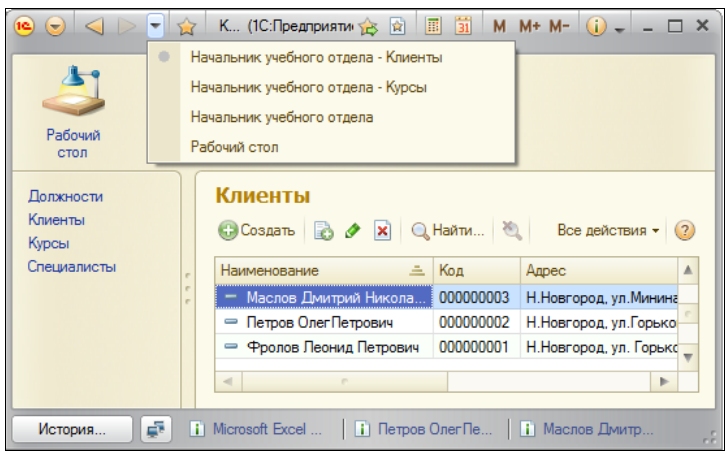


Рис. 1.84. Список с разделами, использованными пользователем

В главное меню входят типовые команды, характерные для приложения в целом. Эти команды не зависят от конкретной конфигурации.

Команды **Вперед** и **Назад** позволяют легко перемещаться по истории работы в текущем пользовательском сеансе. Если необходимо, то в выпадающем списке (сам список открывается с помощью щелчка мышью по пиктограмме с изображением стрелки) можно наблюдать пункты программы, которые открывал пользователь (см. рис. 1.84). Далее правее, в области заголовка, располагается пиктограмма, обеспечивающая переход к избранным ссылкам пользователя (в состав избранных ссылок можно добавить необходимый отчет или текущую точку навигации программы).

В правой части заголовка окна приложения располагаются пиктограммы, соответствующие вспомогательным командам:

- — переход по ссылке;
- — калькулятор;

- — календарь;
- — информация о системе.

Особенности режима *1С:Предприятие*

Одной из особенностей версии 8.2 является отсутствие меню **Операции** при работе пользователя в режиме **1С:Предприятие**. В предыдущих версиях программы этим разделом меню пользователи активно пользовались в процессе своей работы. В результате в версиях 8.0 и 8.1 сразу обеспечивался доступ к объектам, созданным системой на основании разработанных объектов конфигурации. Такая организация с одной стороны весьма удобна. Однако не очень хорошо, если пользователю невысокой квалификации легко доступен подобный ресурс. В этом случае, открыв меню **Операции**, он получал доступ даже к тем объектам, которые ему не предназначены.

По логике организации системы команда, аналогичная команде **Операции**, не должна быть доступна автоматически всем категориям пользователей. В версии 8.2 подобная команда автоматически не отображается в меню. Для включения доступа ко всем объектам необходимо в главном меню выбрать пункт **Сервис**, после чего в открывшемся списке щелкнуть мышью на пункте **Параметры**. Затем в открывшемся окне (рис. 1.85) следует установить флажок **Отображать команду "Все функции"**. Как результат этой установки, в главном меню приложения появится команда **Все функции** (рис. 1.86). Если ею воспользоваться, то увидим окно (рис. 1.87) с перечнем объектов.

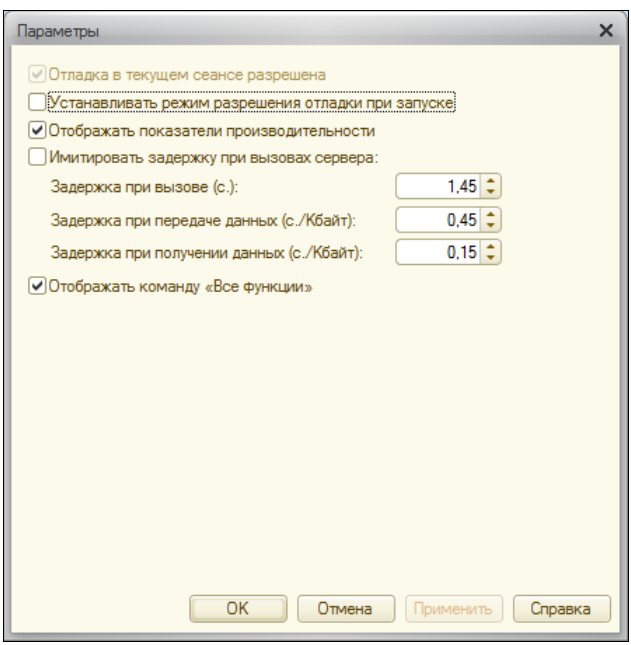


Рис. 1.85. Окно для установки параметров системы

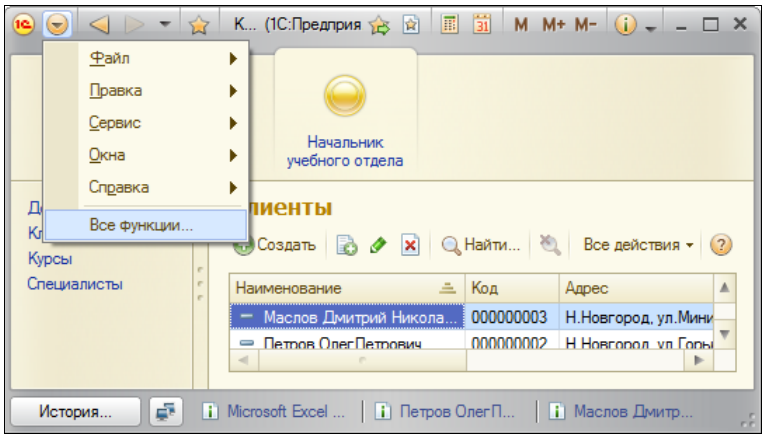


Рис. 1.86. Главное меню приложения

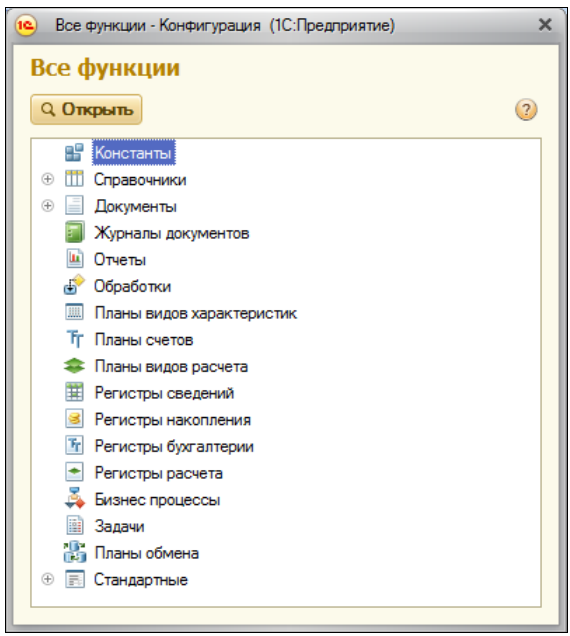


Рис. 1.87. Окно **Все функции**

В меню **Операции** предыдущей версии программы существовал набор системных команд типа **Удаление помеченных объектов**, **Проведение документов** и т. д. Сейчас эти команды собраны в уже знакомом окне **Все функции** (ветка **Стандартные**).

При разработках справочников в версиях 8.0 и 8.1 активно использовалось свойство **Способ редактирования**, позволяющее установить для режима **1С:Предприятие** тот или иной вариант редактирования справочников. Так, можно было остановиться на редактировании в диалоге или в списке (кроме того, имелась возможность сохранения выбора за пользователем в режиме **1С:Предприятие**, где он

мог самостоятельно переключать формы справочников). Теперь свойства **Способ редактирования** в платформе 1С:Предприятие 8.2 нет, что связано с тем, что не поддерживается режим редактирования ссылочных данных в списке. Данные теперь всегда редактируются в отдельной форме.

Варианты клиентских приложений

В версии 8.1 существовал один вариант платформы — мы устанавливали программу 1cv8.exe с одного дистрибутива, который включал все компоненты. Данная программа обеспечивала интерактивное взаимодействие пользователя с компьютером.

Версия 8.2 включает в себя три различных приложения:

- толстый клиент — 1cv8.exe;
- тонкий клиент — 1cv8c.exe;
- веб-клиент.

Толстый клиент представляет собой программу в определенной степени идентичную той, с которой мы встречались в версии 8.1. Отличие заключается в том, что толстый клиент может работать в режиме управляемого приложения. В этом случае разработчику и пользователю доступен управляемый интерфейс и управляемые формы. Несколько вычурное название данного приложения объясняется его функциональностью. Это приложение реализует всю функциональность встроенного языка.

Тонкий клиент представляет собой новое клиентское приложение. Оно работает с информационной базой только в новом режиме управляемого приложения (управляемый интерфейс и управляемые формы).

Тонкий клиент требует существенно меньших аппаратных ресурсов. Это приложение может общаться по локальной сети и через Интернет. Название "тонкий" объясняется тем, что данное приложение умеет исполнять ограниченный набор функциональности встроенного языка.

Веб-клиент выполняется в среде интернет-браузера. И для работы с веб-клиентом необходимо лишь открыть браузер на своем компьютере и указать адрес веб-сервера, на котором выставлена информационная база. В этом случае веб-клиент автоматически начнет выполняться.

Итоги знакомства с конфигуратором

Итак, данная глава завершена. Стоит оглянуться на выполненные разработки и подвести основные итоги. Наверное, главный результат заключается в том, что мы узнали, как выглядит технология разработки конфигураций для информационной системы 1С:Предприятие 8.2. Возможно, что до этого вы работали только с готовыми конфигурациями как пользователь. Теперь же можно взглянуть на эту сложную информационную систему и глазами разработчика (программиста) конфигураций.

Удобное комбинирование работы в режиме конфигуратора и режиме 1С:Предприятие позволяет существенно облегчить работу по созданию прикладных решений на платформе 1С:Предприятие 8.2.

На примере использования конструктора печати мы познакомились с возможностями, которые позволяют разработчику существенно ускорить процесс создания программного кода. В этом случае от него требуется только включение небольших собственных фрагментов.

В ходе рассмотрения примеров этой главы мы получили практические навыки работы с наиболее популярными прикладными объектами системы 1С:Предприятие 8.2 — перечислениями, справочниками, документами, формами и табличными документами.

В целом же итог данной главы заключается в том, что рассмотренные примеры являются фундаментом для более сложных разработок, которые встретятся в последующих главах книги.

ГЛАВА 2

Документы и регистры накопления

В этой главе основное внимание будет уделено практической работе с документами и регистрами накопления. Хотя это два совершенно различных объекта конфигурации, однако в задачах учета они часто взаимосвязаны.

Что касается документов, то с ними мы уже встречались в *главе 1*. Можно сказать, что объект конфигурации *документ* предназначен для фиксирования информации о событиях, происходящих в организации.

Документы в задачах учета используются вместе с другими объектами конфигурации — *регистрами накопления*. Основное назначение регистров накопления просматривается уже из названия — они предназначены для реализации механизма количественного учета информации. Этим двум объектам и будет уделено основное внимание в данной главе.

Объект конфигурации *документ*

Один из примеров в *главе 1* был связан с разработкой в конфигураторе документа, фиксирующего денежные поступления от клиентов. Мы познакомились с электронной формой документа, а также с его печатным вариантом (печатной формой). Рассмотренные примеры можно считать некой базой для данной и последующих глав книги, где работе с документами будет уделено значительное внимание.

С практической точки зрения документ можно считать прикладным объектом, назначением которого является ввод и фиксация информации о событиях, происходящих в организации. Эти события, как правило, влияют на количественные показатели в рассматриваемой предметной области (чаще всего учет связан с движением денежных средств и материальных ценностей). В качестве примеров можно предложить факт выписки денежных сумм сотрудникам, передачу товаров и материалов с одного склада на другой и т. д. Именно для того чтобы отражать подобные факты в системе учета, объект конфигурации *"документ"* и был разработан.

Практически все события, происходящие в организации, отличаются привязкой ко времени, в связи с чем параметр, характеризующий точное время, является обязательным реквизитом любого документа. Мы уже видели, что действия при разра-

ботке документов начинаются в конфигураторе с создания структуры документа — набора полей с указанием их типов данных. Кроме того, с помощью программных процедур реализуются необходимые действия при работе пользователя с документом. Результатом разработки документа (объекта конфигурации) в конфигураторе является формирование информационной структуры, которая позволяет пользователю в режиме **1С:Предприятие** работать с документами данного типа (создавать, вносить в них информацию, изменять, удалять).

Основной чертой, отличающей документы от других объектов, является их способность *проведения* (возможность формировать проводки, являющиеся результатом информации, присутствующей в документах). Это очень важная особенность, и ее необходимо пояснить. В информационной системе 1С:Предприятие имеются структуры данных (к ним относятся и регистры накопления, рассматриваемые в данной главе), которые описывают количественное состояние дел в организации (например, стоимостный и количественный учет разнообразных товаров). Проведение документов означает факт внесения изменений в эти структуры (причина изменений, разумеется, связана с информацией, содержащейся в документах). Например, поступление по конкретному документу определенного товара изменяет суммарное количество данного товара на складе.

До тех пор, пока документ не проведен, состояние учета неизменно. В этом случае функциональность документа не реализована, но сам он, тем не менее, зарегистрирован в базе данных.

С другой стороны, в информационной системе при необходимости должна существовать и возможность отмены проведения (возвращения информации в первоначальное состояние). Например, при отмене проведения документа "Поступление товара" должна быть скорректирована информация в регистре, который используется для учета количества товаров на складах. Подобная возможность *отмены проведения* документов в системе 1С:Предприятие также реализована.

Важно подчеркнуть, что документы, разработанные для различных стандартных конфигураций системы 1С:Предприятие, являются аналогами реальных документов, которые используются в работе большинства организаций.

Многочисленные документы, используемые в реальном документообороте, различны по своей структуре. Например, основное содержание одних документов могут составлять таблицы (скажем, списки номенклатуры, услуг или сотрудников), а в других они могут отсутствовать. Существуют также документы с наличием нескольких табличных частей (например, одна табличная часть отводится для списка сотрудников, работающих на постоянной основе, а другая — для списка совместителей). В связи с этим объект конфигурации *документ* системы 1С:Предприятие должен обладать хорошими возможностями для конструирования.

Использование рассматриваемого объекта конфигурации при разработке прикладных решений должно позволять легко создавать различные документы — приходную накладную, счет или документ о перемещении товаров со склада на склад. В системе 1С:Предприятие 8.2 ситуация выглядит именно так — все конструктивные возможности во власти разработчика.

Опишем теперь основные параметры объекта конфигурации *документ*. Так, в любом документе (как в созданном в системе 1С:Предприятие, а также в реальном) присутствует информация, которая характеризует его в целом. Это дата и время его создания, а также порядковый номер (в последовательности документов одного типа).

> ***Примечание***
>
> Поскольку документ вносит изменения в состояние учета, он всегда привязан к конкретному моменту времени.

Кроме того, в любом документе присутствует общая информация и другого плана. Например, организация, подразделение и ответственное лицо. Для подобных параметров в объекте конфигурации *документ* отводится область, называемая *шапкой*. Ее фактически составляют *реквизиты*, характеризующие документ в целом. Реквизиты документа, как мы уже упоминали, являются также объектами конфигурации, только относятся к категории подчиненных.

Кроме общей области, документ может содержать список (и не один) данных, схожих по структуре. Например, список сотрудников, организаций или товаров. Каждый список включает в себя еще уточняющую и дополнительную информацию. Например, вместе с названием товара указываются количество и цена. Для отражения такой информации в разрабатываемый объект конфигурации следует включить один или несколько подчиненных объектов, которые называются *табличными частями*.

В большинстве ситуаций у документов установлено свойство **Автонумерация**. В этом случае каждому вновь введенному документу номер будет присваиваться автоматически. Заметим, что автоматически присвоенный номер пользователь может исправить.

В ряде случаев требуется установить сквозную нумерацию для документов разных типов. Для этого предназначены *нумераторы*, которые также являются объектами конфигурации.

Регистры накопления

Основное назначение *регистров накопления* просматривается уже из названия — они предназначены для реализации механизма количественного учета информации. Вся информация, которая вводится с помощью документов, по логике работы должна быть зафиксирована в регистрах накопления. А уже после этого сведения из регистров накопления можно использовать для формирования отчетов.

Регистр накопления представляет собой объект конфигурации, предназначенный для хранения *движений* — изменений, которые в нем происходят при проведении документа. Для работы с регистром накопления в первую очередь необходимо определить его структуру — в каких разрезах следует накапливать данные с целью последующего максимального эффективного извлечения информации.

Для формирования представления о регистре накопления рассмотрим следующий пример. Допустим, в нашей организации имеется несколько филиалов, а сама орга-

низация ведет торговую деятельность. По логике работы необходимо учитывать различные товары на разных филиалах. Фактически в этом случае сотрудникам организации необходимо в процессе работы получать ответы на вопросы следующего типа: сколько товаров конкретного наименования находится на определенном складе или сколько всего товаров конкретного наименования находится на всех складах и т. д.

Данные, хранящиеся в регистрах накопления, подразделяются на *измерения* и *ресурсы*. Накопление количественной информации может производиться в разрезе одного или нескольких измерений, которые описываются при разработке конфигураций. Например, в качестве измерений могут выступать склад и номенклатура товаров.

Виды числовой информации, аккумулируемой регистром накопления, называются *ресурсами*. Так, регистр накопления может накапливать информацию о количестве различных товаров на каждом из имеющихся складов.

Примечание

Измерения и ресурсы являются подчиненными объектами конфигурации.

Можно сказать, что регистр накопления представляет собой многомерную систему координат, а ресурс регистра накопления является функцией. Оси этой системы координат называются измерениями регистра накопления, а значения функции являются значениями ресурса регистра накопления.

Изменение состояния регистра накопления происходит обычно при проведении документа. Процедура проведения документа располагается в модуле документа. В данной процедуре определяются движения регистра накопления. Важно отметить, что движения регистров содержат только приращения (со знаком "плюс" или "минус") значений ресурсов регистра.

В конкретной конфигурации можно создать неограниченное количество регистров накопления.

Кроме измерений и ресурсов, у регистра накопления может быть создан набор *реквизитов*. С помощью реквизитов можно включать в движения регистра дополнительную информацию (например, фамилию человека, который произвел отгрузку товара).

Когда в конфигураторе мы создаем новый регистр накопления (далее в данной главе это будут конкретные регистры под именами `ИнформацияОТоварах` и `Заявки`), то в памяти вычислительной системы создается структура данных для реализации механизма накопления численных показателей.

Ручное внесение информации в регистры накопления, как правило, исключается. Они не предназначены для интерактивного редактирования.

Примечание

Вообще говоря, редактирование содержимого регистров накопления возможно, однако стандартная работа с регистрами накопления заключается в модификации их содержимого с помощью программно реализуемых алгоритмов.

Основной режим работы по добавлению данных (а также модификации и удалению) в регистры накопления связан с действиями, выполняемыми в программных процедурах, относящихся к другим объектам. Часто (даже можно сказать в основном) этими объектами являются документы.

Далее в этой главе рассматривается разработка конфигурации, в которой мы познакомимся с программными приемами внесения информации в регистры накопления с использованием документов.

Создание конфигурации

Для демонстрации функциональности документов и регистров накопления создадим несложную конфигурацию. Технология разработки конфигурации подробно описана в начале *главы 1*. Новая конфигурация будет содержать ряд справочников, документов и регистров накопления. При этом добавление объектов конфигурации будем производить поэтапно по ходу изложения.

В заключительной части *главы 1* мы познакомились с подсистемами и на реальном примере увидели, как с помощью подсистем можно обеспечить структурность прикладного решения. Поэтому здесь в качестве первого действия после создания новой конфигурации мы сформируем две подсистемы, которые условно назовем:

- **ЗаведующийФилиалом;**
- **Менеджер.**

Это связано с тем, что дальнейший "функционал" будет разнесен на две данные категории, исходя из логики работы сотрудников нашей условной организации.

Технические действия, касающиеся создания подсистемы, нам знакомы по *главе 1* и они производятся в дереве объектов конфигурации (рис. 2.1).

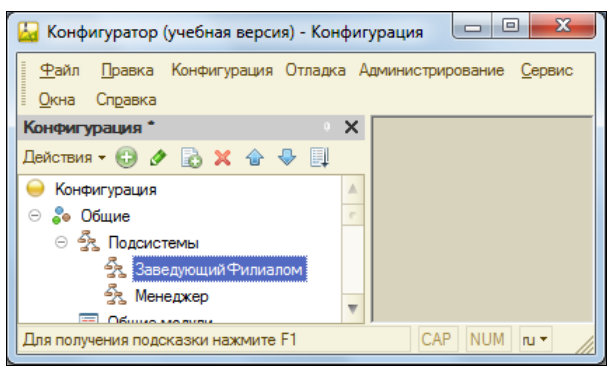


Рис. 2.1. Дерево объектов конфигурации

На следующем этапе требуется создать справочники "Фирмы" и "Номенклатура", которые не будут иметь дополнительных реквизитов. Технология их разработки полностью основывается на тех приемах, которые были использованы в *главе 1*. На рис. 2.2 представлена форма списка справочника "Фирмы", который при его разработке в конфигураторе следует отнести к подсистеме **ЗаведующийФилиалом**.

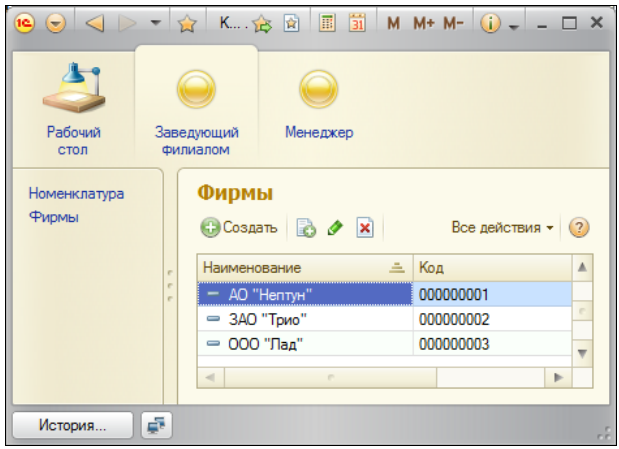


Рис. 2.2. Форма списка справочника "Фирмы"

После этого создадим справочник "Номенклатура", отнесем его сразу к двум имеющимся подсистемам, а в режиме **1С:Предприятие** внесем в данный справочник несколько элементов (рис. 2.3).

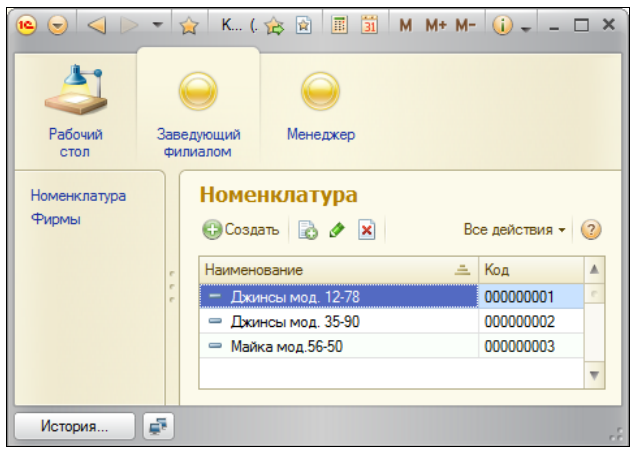


Рис. 2.3. Форма списка справочника "Номенклатура"

Далее нам понадобится документ "Поступление". В связи с тем, что технология разработки документов в конфигураторе рассматривалась в *главе 1*, подчеркнем только наиболее информативные моменты для выполнения практических действий при создании нового документа. После установки имени документа на вкладке **Основные** в окне редактирования объекта конфигурации перейдем на вкладку **Данные**. В области шапки документа (рис. 2.4) разместим один дополнительный реквизит — **Поставщик** (тип данных — **СправочникСсылка.Фирмы**). Для табличной части документа выберем имя **Перечень** и внесем в нее два реквизита:

- **Товар** (тип данных — **СправочникСсылка.Номенклатура**);
- **Количество** (тип данных — **Число**) для внесения количества поступивших товаров.

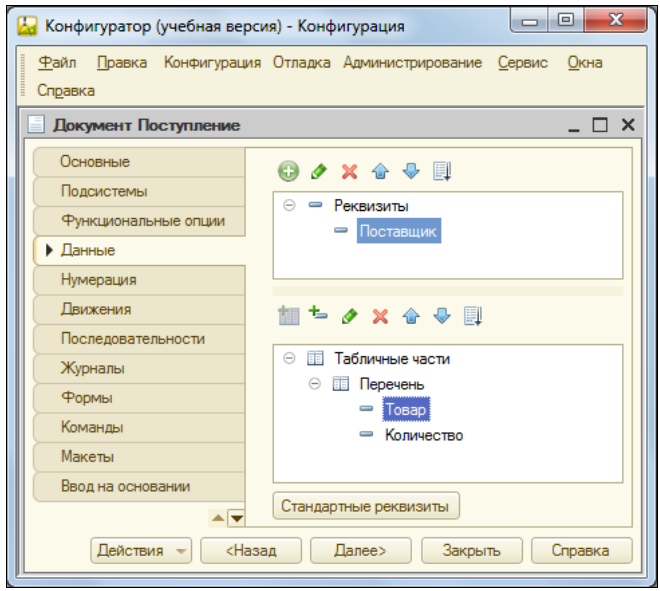


Рис. 2.4. Вкладка **Данные** окна редактирования документа "Поступление"

Включим разработанный документ "Поступление" в подсистему **Менеджер**, перейдем в режим **1С:Предприятие** и заполним несколько документов. Один из вариантов заполненного документа "Поступление" приведен на рис. 2.5.

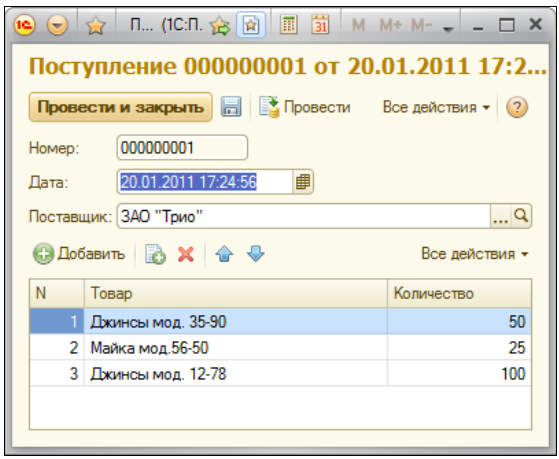


Рис. 2.5. Форма документа "Поступление"

Регистр накопления остатков

После создания необходимых справочников и документа перейдем к разработке регистра накопления, который назовем **ИнформацияОТоварах**. Для этого в дереве объектов конфигурации щелкнем правой кнопкой по элементу **Регистры накопления**. Далее в открывшемся контекстном меню выберем пункт **Добавить** и перей-

дем к работе в окне редактирования объекта конфигурации. На вкладке **Основные** установим имя объекта (рис. 2.6), а также обратим внимание на список **Вид регистра**. Здесь следует установить вид регистра накопления — **Остатки.**

Примечание

Что касается вариантов установки параметра **Вид регистра**, то далее следует комментарий как использованного вида **Остатки**, так и другого возможного — **Обороты**.

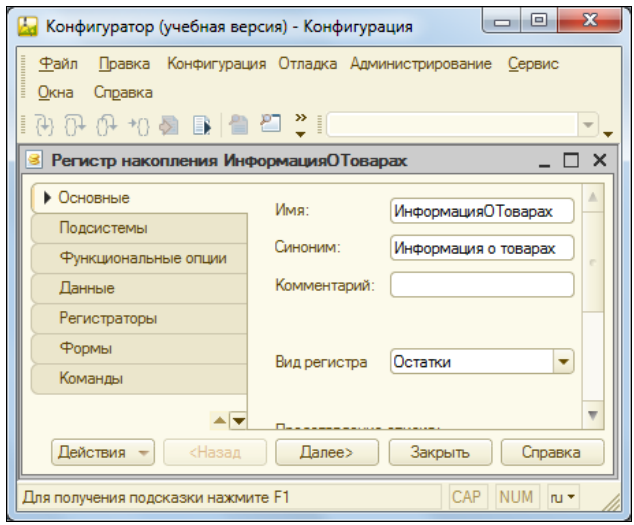


Рис. 2.6. Вкладка **Основные** регистра накопления **ИнформацияОТоварах**

После этого в окне редактирования объекта конфигурации необходимо перейти на вкладку **Данные** и сформировать параметры для этого регистра:

- **Измерения: Организации** (тип данных — **СправочникСсылка.Фирмы**);
- **Измерения: Товар** (тип данных — **СправочникСсылка.Номенклатура**);
- **Ресурсы: Количество** (в качестве типа данных выберем **Число**).

На рис. 2.7 показано заполнение вкладки **Данные** окна редактирования объекта конфигурации в соответствии с описанными параметрами регистра накопления. Однако сразу после этого переход в режим **1С:Предприятие** для работы с созданным регистром не имеет смысла — необходимые *действия в конфигураторе еще не завершены*. Дело в том, что создание регистра накопления без документа, который с ним взаимодействует (формирует движения по регистру), не принимается информационной системой. В этом случае при попытке сохранения конфигурации вы увидите на экране соответствующее диагностическое сообщение системы 1С:Предприятие.

Для устранения этой недоработки следует внести коррективы в документ "Поступление", в котором с помощью программной процедуры нужно реализовать выполнение движений по созданному регистру **ИнформацияОТоварах**. В этом случае все записи в регистр, фиксирующие заказы, будут выполняться для пользователя

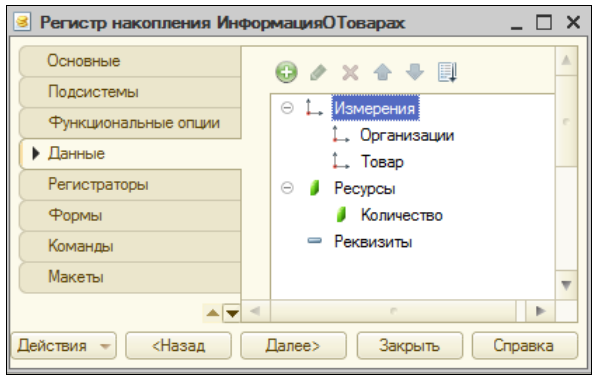


Рис. 2.7. Структура регистра накопления **ИнформацияОТоварах**

незаметно, т. е. одновременно (точнее последовательно) с записью документа "Поступление".

> ***Примечание***
>
> При записи документа осуществляется и его проведение — формирование движений по регистру. Таким образом, для пользователя каких-либо новых действий в режиме **1С:Предприятие** не требуется — все обеспечивает традиционный щелчок по кнопке **Провести и закрыть** в окне формы документа.

Изменения в документе *Поступление*

Ранее созданный документ "Поступление" нам сейчас потребуется доработать в плане формирования необходимых проводок. Для этого в режиме конфигуратора откроем окно редактирования объекта конфигурации и займемся изменениями в рассматриваемом документе, а именно перейдем на вкладку **Движения** (рис. 2.8), где от нас требуется определить, в каких регистрах будут происходить изменения при проведении документа "Поступление". В разделе **Регистры накопления** отражаются все имеющиеся регистры накопления, по которым можно организовать движения. В данном случае в нашей конфигурации мы располагаем только одним регистром накопления — **ИнформацияОТоварах**, и только он присутствует в этом окне. От нас требуется поставить флажок напротив его имени.

> ***Примечание***
>
> Система 1С:Предприятие позволяет организовать движения в документе по нескольким регистрам [6].

В результате мы подошли к следующему этапу разработки, и теперь можно пойти двумя путями: либо самостоятельно написать процедуру, которая будет формировать движения при проведении подобных документов, либо воспользоваться возможностями *конструктора движений*. Соответствующая кнопка (**Конструктор движений**) активизируется сразу после указания конкретного регистра (см. рис. 2.8). Выберем второй путь и, используя конструктор движений, автоматически сформируем необходимую программную процедуру в результате несложного диа-

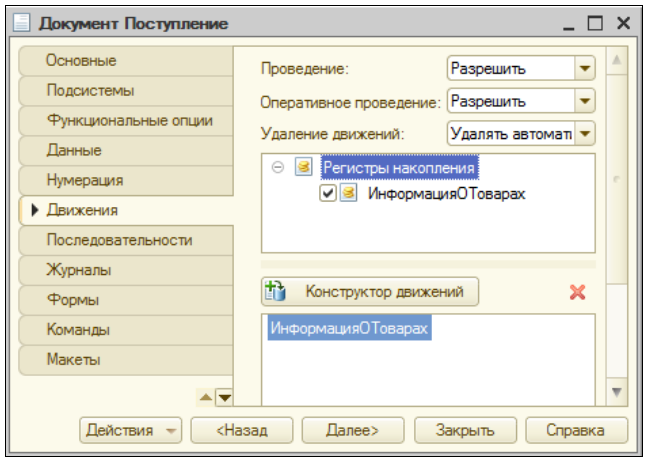


Рис. 2.8. Вкладка **Движения** окна редактирования документа "Поступление"

лога. Подобным образом в предыдущих главах мы уже несколько раз создавали печатные формы (элемента справочника и документа).

В режиме диалога с конструктором движений ключевым (для построения программной процедуры) является окно, представленное на рис. 2.9, которое стоит прокомментировать. В разделе **Регистры** указаны те регистры, по которым можно формировать движения в документе. В данном случае регистр у нас один — **ИнформацияОТоварах**, который и выбран для дальнейших действий конструктора.

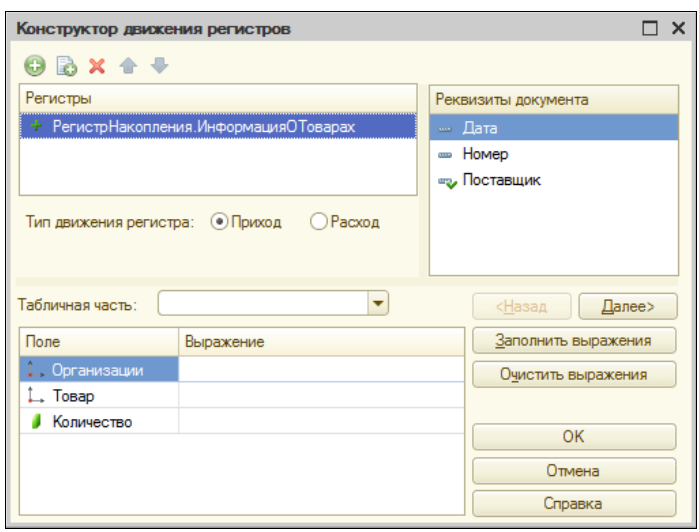


Рис. 2.9. Окно конструктора движений регистра накопления

В правое верхнее поле уже автоматически внесены реквизиты шапки данного документа. Это связано с тем, что в указанное поле включаются параметры документа, которые могут участвовать в движениях.

Но кроме шапки, в документе "Поступление" имеется еще и табличная часть, и для добавления реквизитов табличной части в поле **Реквизиты документа** следует выполнить несложную операцию. В центральной части рис. 2.9 присутствует поле со списком (с соответствующей подписью — **Табличная часть**), где перечислены имеющиеся табличные части в документе (в нашем документе табличная часть одна — **Перечень**). Для перенесения в поле **Реквизиты документа** реквизитов табличной части необходимо просто выбрать ее имя в списке табличных частей (рис. 2.10).

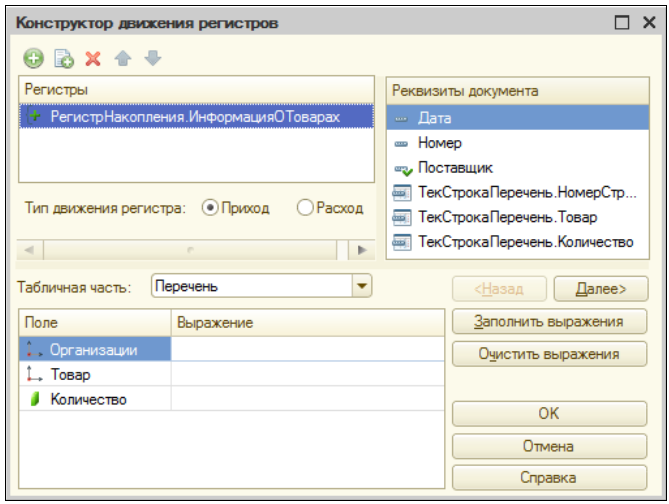


Рис. 2.10. Отображение реквизитов табличной части в окне конструктора движений

На этом работа с данным окном еще не завершена, и нам следует перейти к нижней левой части окна конструктора движений. Здесь необходимо указать соответствие полей регистра накопления и реквизитов документа. Технически это можно сделать с помощью двойного щелчка мышью по необходимому реквизиту документа в правом верхнем разделе на рис. 2.10. Также можно воспользоваться кнопкой **Заполнить выражения**. В результате напротив названий полей регистра появятся соответствующие выражения (рис. 2.11).

Теперь все действия в окне конструктора движений регистра выполнены и осталось щелкнуть по кнопке **ОК**. Это приведет к генерации конструктором программной процедуры (листинг 2.1), которая формирует движения по регистру при проведении документа "Поступление". В дальнейшем если потребуется эту процедуру найти (скажем, для коррекции или просто для просмотра), то в окне редактирования объекта конфигурации следует перейти на вкладку **Прочее**. Здесь после нажатия кнопки **Модуль объекта** мы увидим текст, представленный в листинге 2.1.

Обратим внимание еще на одну важную деталь в окне конструктора движений регистров на рис. 2.11. В качестве значения параметра **Тип движения регистра** по умолчанию с помощью переключателя установлен вариант **Приход**. Это соответствует добавлению ресурсов регистра при проведении документа. Альтернативный вариант **Расход** приводит к уменьшению ресурсов.

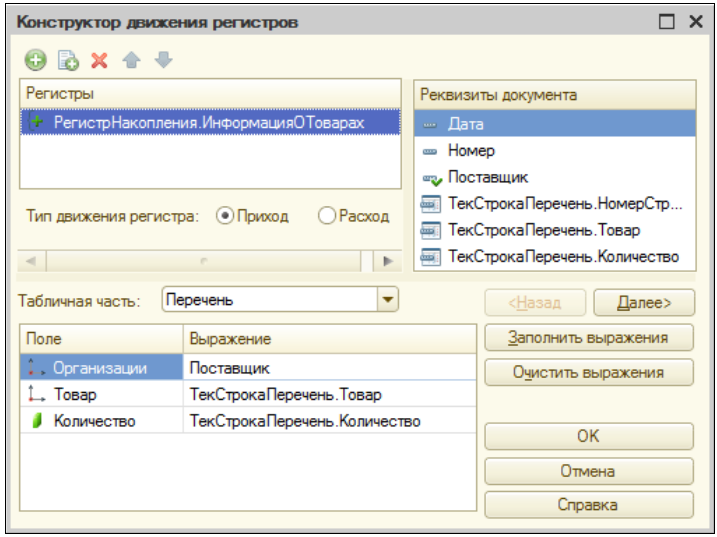


Рис. 2.11. Заполнение выражений в окне конструктора движений регистра

Листинг 2.1. Процедура, выполняемая при проведении документа Поступление

```
Процедура ОбработкаПроведения(Отказ, Режим)
  //{{__КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
  // Данный фрагмент построен конструктором.
  Движения.ИнформацияОТоварах.Записывать = Истина;
  Для Каждого ТекСтрокаПеречень Из Перечень Цикл
    // регистр ИнформацияОТоварах Приход
    Движение = Движения.ИнформацияОТоварах.Добавить();
    Движение.ВидДвижения = ВидДвиженияНакопления.Приход;
    Движение.Период = Дата;
    Движение.Организации = Поставщик;
    Движение.Товар = ТекСтрокаПеречень.Товар;
    Движение.Количество = ТекСтрокаПеречень.Количество;
  КонецЦикла;
  //}}__КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
КонецПроцедуры
```

Основное содержание процедуры `ОбработкаПроведения` составляет цикл, в котором перебираются строки табличной части **Перечень** документа "ИнформацияОТоварах":

```
Для Каждого ТекСтрокаПеречень Из Перечень Цикл
```

При каждом проходе цикла с помощью метода `Добавить` создается очередная запись в регистре накопления. В последующих строках процедуры устанавливаются значения полей регистра в соответствии с данными из очередной строки табличной части документа.

На этом работа с конструктором движений завершена, и можно подчеркнуть главный результат, заключающийся в том, что вследствие несложных визуальных действий в документе "Поступление" реализовано формирование движений по регистру накопления (`ИнформацияОТоварах`). Автоматически созданную программную процедуру `ОбработкаПроведения` в дальнейшем можно при необходимости скорректировать либо дополнить.

Сейчас стоит перейти в режим **1С:Предприятие** и проверить результат выполненных изменений (протестировать созданный документ "ИнформацияОТоварах"). Так, обратимся к подсистеме **Менеджер**, в которой выберем документ "Поступление". В результате на экране откроется форма списка данного документа, содержащая пиктографическое меню для выполнения возможных действий с документом. Так можно создать новый документ или открыть форму (элемента) уже созданного документа (рис. 2.12).

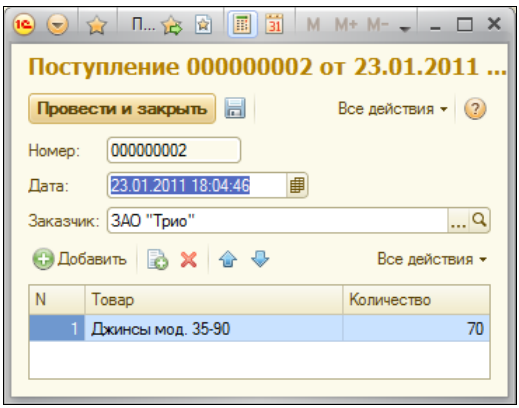


Рис. 2.12. Форма документа "Поступление"

Для проведения открытого документа следует щелкнуть по кнопке **Провести и закрыть**, в результате чего платформой будет выполнена запись информации в регистр накопления **ИнформацияОТоварах**.

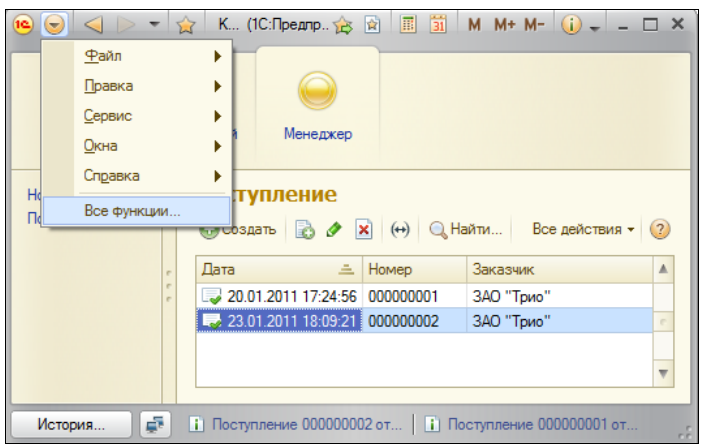


Рис. 2.13. Главное меню приложения

Чтобы просмотреть содержание регистра накопления, следует обратиться к разделу **Все функции** главного меню приложения (рис. 2.13). В результате на экране откроется окно (рис. 2.14), в котором требуется обратиться к регистру накопления **Информация о товарах**. Мы увидим форму списка данного регистра накопления (рис. 2.15), в которой отражены внесенные записи (в поле **Регистратор** указано, с какими документами они связаны).

Данный набор записей называется *движениями* по регистру. Каждая запись содержит информацию о регистраторе (это документ, который вызвал движение). Также каждая запись содержит значения измерений и значения *приращений* ресурсов.

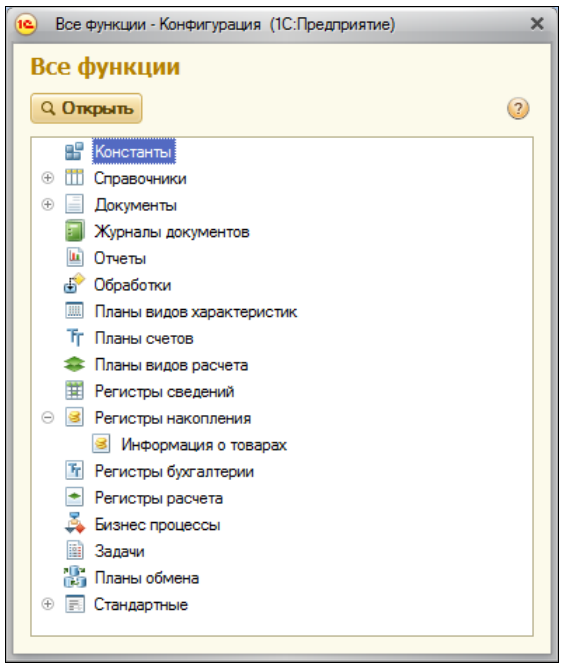


Рис. 2.14. Содержание окна **Все функции**

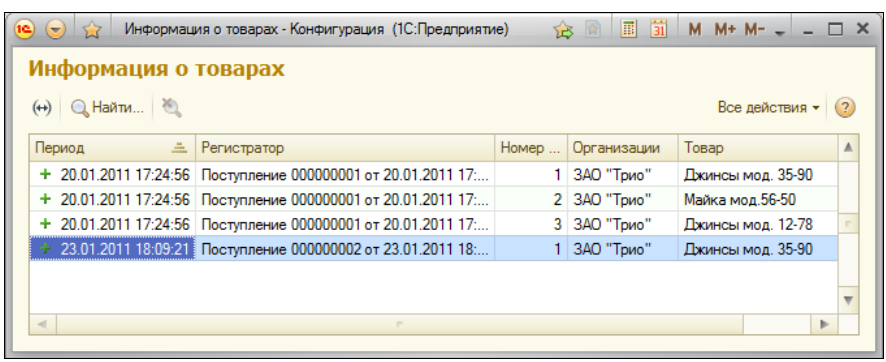


Рис. 2.15. Форма списка регистра **ИнформацияОТоварах**

Расход по регистру накопления

В предыдущем разделе мы рассмотрели ситуацию, когда документ обеспечивает добавление ресурса регистра накопления. Для полноты картины создадим документ "Отгрузка", который будет уменьшать ресурс регистра **ИнформацияОТоварах**. Данный документ (рис. 2.16) будет похож по своей структуре на уже разработанный нами документ "Поступление".

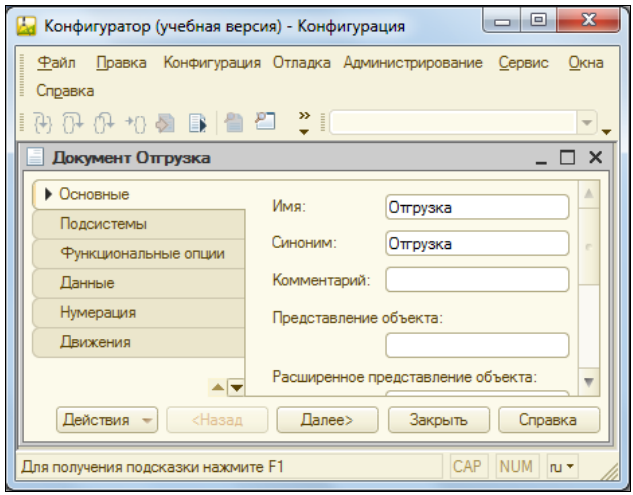


Рис. 2.16. Вкладка **Основные** окна редактирования документа "Отгрузка"

На вкладке **Данные** определим реквизиты шапки и содержание табличной части нового документа (рис. 2.17). При этом типы данных полей нового документа полностью соответствуют типам полей документа, связанного с поступлениями товаров.

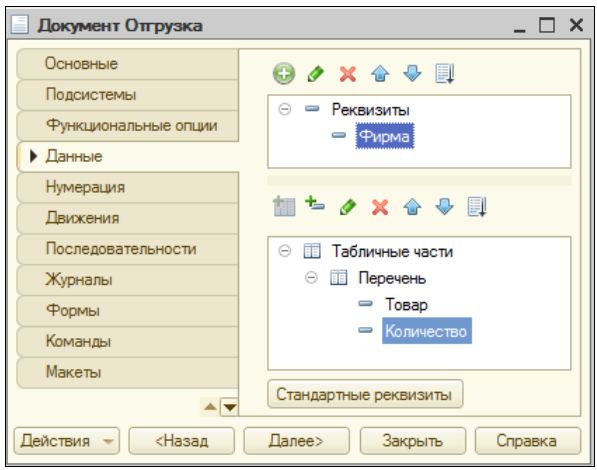


Рис. 2.17. Вкладка **Данные** окна редактирования документа "Отгрузка"

После формирования структуры документа "Отгрузка" перейдем на вкладку **Движения** и здесь с помощью конструктора (рис. 2.18) нам необходимо сформировать соответствие реквизитов документа и полей регистра. Важно отметить, что в данном случае в качестве типа движения регистра указан **Расход**.

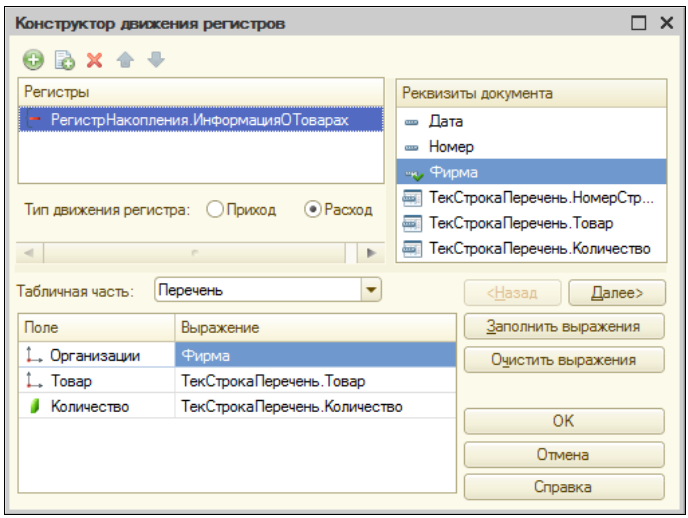


Рис. 2.18. Окно конструктора движений документа "Отгрузка"

После щелчка по кнопке **ОК** автоматически формируется процедура обработки проведения данного документа (листинг 2.2).

Листинг 2.2. Процедура, выполняемая при проведении документа Отгрузка

```
Процедура ОбработкаПроведения(Отказ, Режим)
  //{{__КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
  // Данный фрагмент построен конструктором.
  Движения.ИнформацияОТоварах.Записывать = Истина;
  Для Каждого ТекСтрокаПеречень Из Перечень Цикл
  // регистр ИнформацияОТоварах Расход
     Движение = Движения.ИнформацияОТоварах.Добавить();
     Движение.ВидДвижения = ВидДвиженияНакопления.Расход;
     Движение.Период = Дата;
     Движение.Организации = Фирма;
     Движение.Товар = ТекСтрокаПеречень.Товар;
     Движение.Количество = ТекСтрокаПеречень.Количество;
  КонецЦикла;
  //}}__КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
КонецПроцедуры
```

На рис. 2.19 показана форма документа "Отгрузка", а на рис. 2.20 форма списка регистра накопления **ИнформацияОТоварах** с наличием внесенной записи, связанной с данным экземпляром документа "Отгрузка".

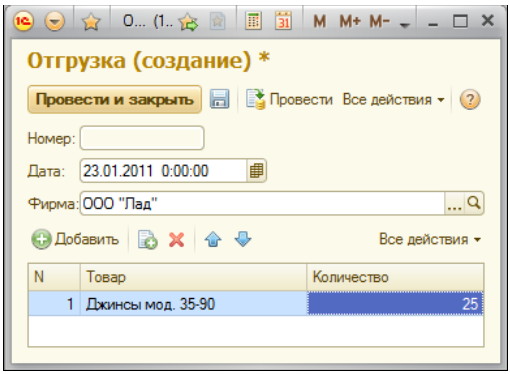


Рис. 2.19. Форма документа "Отгрузка"

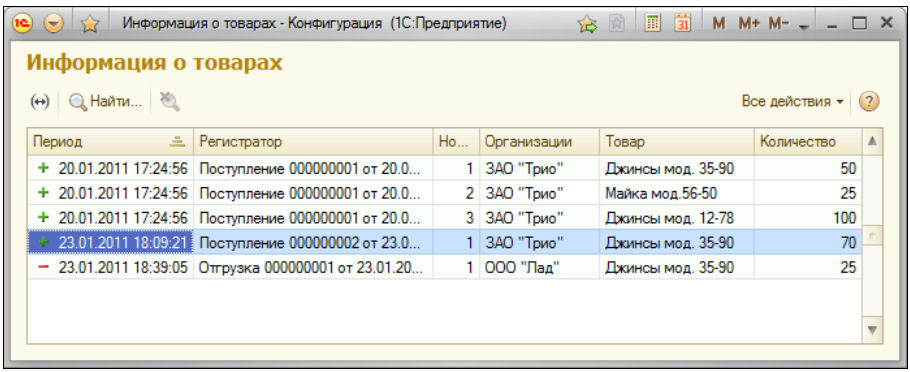


Рис. 2.20. Форма списка регистра **ИнформацияОТоварах**

Оборотный регистр накопления

В ряде ситуаций нет необходимости учитывать остатки. Например, в случае подачи заявок на продукцию представляет интерес подсчет заявок за определенный интервал. Или, скажем, другой пример — наша организация предоставляет образовательные услуги. В этом случае важны услуги в денежном выражении, оказанные за определенный период. Для подобных ситуаций предназначены *оборотные регистры накопления*.

В общем, оборотные регистры накопления и регистры остатков отличаются не очень сильно. Оборотные регистры накапливают обороты, а остатки в них не учитываются. В этом случае система создает только одну *виртуальную таблицу* — таблицу оборотов. Фактически это основное отличие оборотного регистра.

> ***Примечание***
>
> Виртуальные таблицы регистров создаются платформой автоматически и содержат итоги (по приходу, расходу, оборотам, остаткам).

Для демонстрации работы оборотного регистра создадим еще один документ с именем "Заявка", используя который мы рассмотрим особенности регистров оборо-

тов. На вкладке **Основные** внесем имя документа, после чего перейдем на вкладку **Данные** (рис. 2.21). В шапке документа добавим один дополнительный реквизит **ФирмаПоставщик** (тип данных — **СправочникСсылка.Фирмы**). Для табличной части документа выберем имя **ПереченьТоваров** и внесем в нее два реквизита:

- **Номенклатура** (тип данных — **СправочникСсылка.Номенклатура**);
- **Количество** (тип данных — **Число**).

Отнесем созданный документ к подсистеме **ЗаведующийФилиалом**, после чего перейдем в режим **1С:Предприятие** и заполним несколько документов указанного типа.

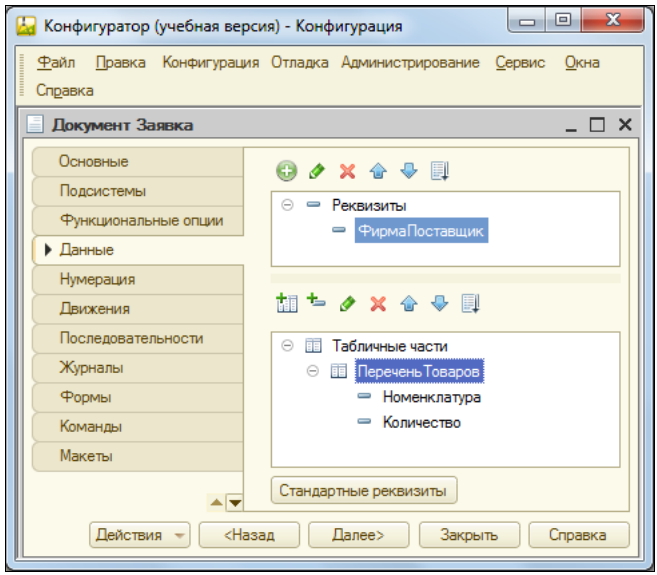


Рис. 2.21. Вкладка **Данные** окна редактирования документа "Заявка"

Приступим далее к разработке нового регистра накопления. После установки имени регистра **ИнформацияОЗаявках** на вкладке **Основные** необходимо выбрать значение **Обороты** в поле **Вид регистра** (рис. 2.22).

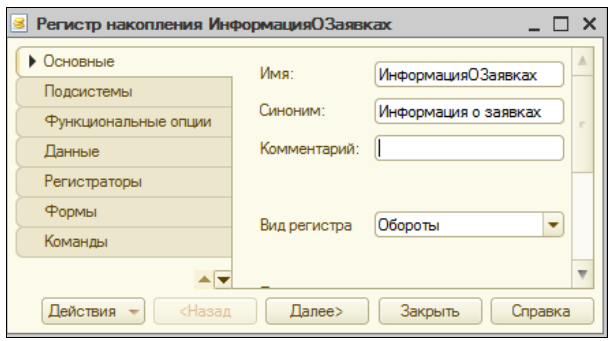


Рис. 2.22. Вкладка **Основные** окна редактирования регистра **ИнформацияОЗаявках**

После этого в окне редактирования данного объекта конфигурации необходимо перейти на вкладку **Данные** и сформировать следующие параметры регистра накопления:

- **Измерения: Фирма** (тип данных — **СправочникСсылка.Фирмы**);
- **Измерения: Товар** (тип данных — **СправочникСсылка.Номенклатура**);
- **Ресурсы: Количество** (в качестве типа данных выберем **Число**).

На рис. 2.23 показано заполнение вкладки **Данные** окна редактирования создаваемого регистра накопления в соответствии с указанной структурой.

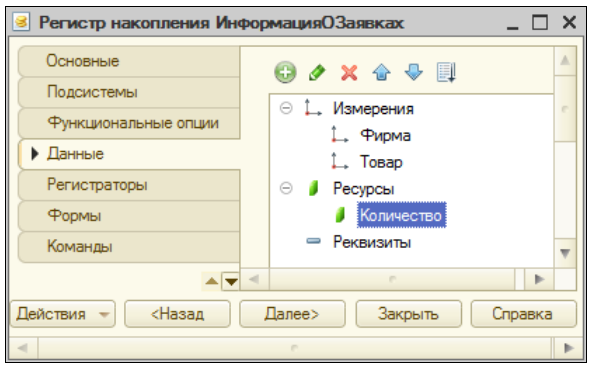


Рис. 2.23. Структура регистра **ИнформацияОЗаявках**

Перейдем к следующему этапу разработки, который связан с формированием движений по регистру **ИнформацияОЗаявках**. Эти движения будут формироваться в результате проведения документа "Заявка". Поэтому откроем окно редактирования данного объекта конфигурации и перейдем на вкладку **Движения** (рис. 2.24). Здесь требуется с помощью флажка отметить регистр **ИнформацияОЗаявках**, по которому будут формироваться движения.

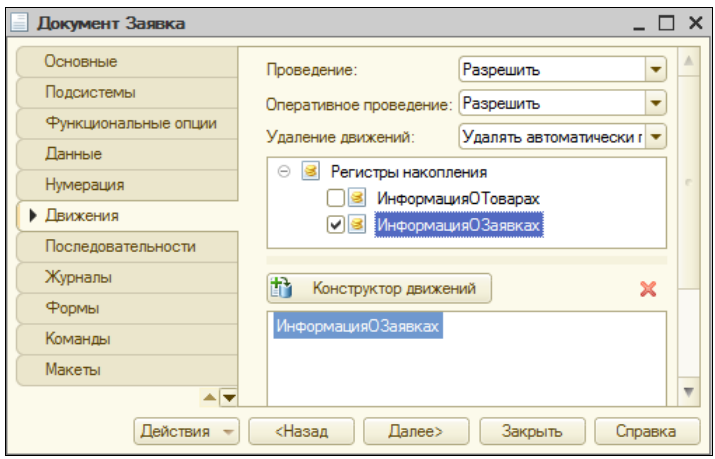


Рис. 2.24. Вкладка **Движения** окна редактирования документа "Заявка"

Как и ранее, для разработки процедуры `ОбработкаПроведения` воспользуемся конструктором движений. Поэтому в окне на рис. 2.24 следует щелкнуть по кнопке **Конструктор движений**, после чего в открывшемся окне конструктора необходимо установить соответствие полей регистра и реквизитов документа (рис. 2.25).

Примечание

Для заполнения поля **Выражения** следует выполнить двойной щелчок мышью по имени необходимого реквизита.

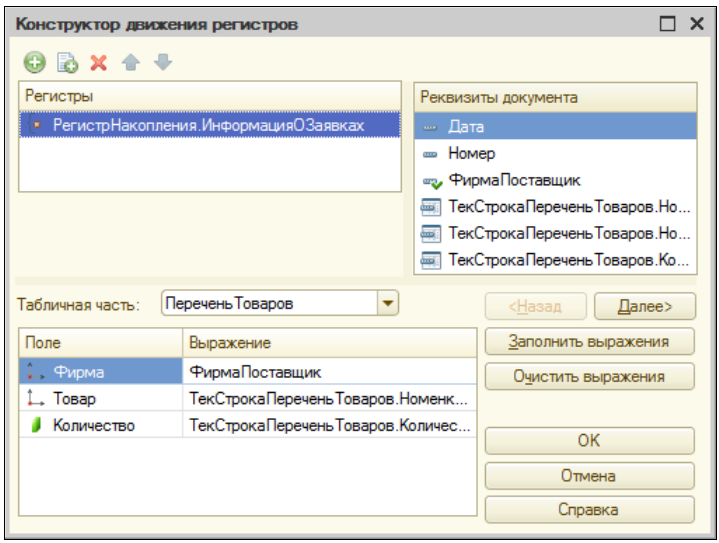


Рис. 2.25. Окно конструктора движений регистра **ИнформацияОЗаявках**

После этого с помощью щелчка по кнопке **ОК** конструктор автоматически формирует текст необходимой процедуры проведения документа "Заявка" (листинг 2.3).

Листинг 2.3. Процедура, выполняемая при проведении документа Заявка

```
Процедура ОбработкаПроведения(Отказ, Режим)
  //{{__КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
  // Данный фрагмент построен конструктором.
  Движения.ИнформацияОЗаявках.Записывать = Истина;
  Для Каждого ТекСтрокаПереченьТоваров Из ПереченьТоваров Цикл
  // регистр ИнформацияОЗаявках
    Движение = Движения.ИнформацияОЗаявках.Добавить();
    Движение.Период = Дата;
    Движение.Фирма =  ФирмаПоставщик;
    Движение.Товар = ТекСтрокаПереченьТоваров.Номенклатура;
    Движение.Количество = ТекСтрокаПереченьТоваров.Количество;
  КонецЦикла;
  //}}__КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
КонецПроцедуры
```

Отнесем созданный регистр к подсистеме **ЗаведующийФилиалом**. Далее следует в режиме **1С:Предприятие** провести одну или несколько заявок (один из экземпляров документа "Заявка" представлен на рис. 2.26) и посмотреть на содержание регистра **ИнформацияОЗаявках** (рис. 2.27).

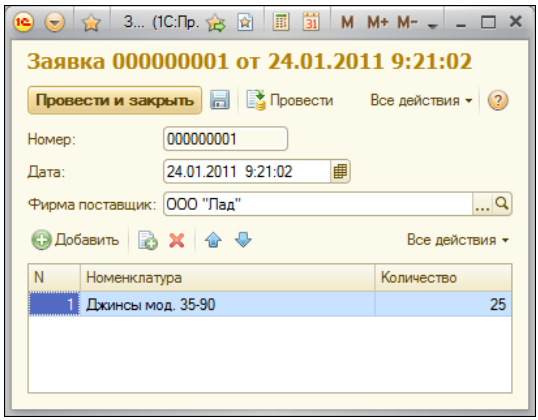


Рис. 2.26. Форма документа "Заявка"

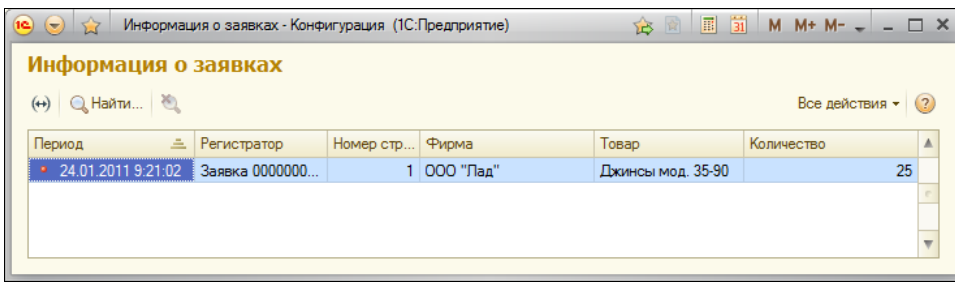


Рис. 2.27. Список зарегистрированных заявок

Итоги

В этой главе мы познакомились с новым объектом конфигурации — *регистром накопления*, а также более подробно рассмотрели ранее встречавшийся объект — *документ*. Совместное рассмотрение этих различных объектов конфигурации связано с тем, что в практических задачах они бывают часто взаимосвязаны. На примерах разработки двух конфигураций мы разобрали взаимодействие данных объектов.

При проведении документов производится формирование движений регистров накопления. Это и является основным механизмом регистрации и учета информации о событиях, привязанных ко времени.

ГЛАВА 3

Регистры сведений

Регистры сведений — это относительно новые объекты конфигурации, которые появились в версии программного продукта 1С:Предприятие 8 (в широко распространенной в нашей стране системе 1С:Предприятие 7.7 их не было). Примеры данной главы преследуют цель познакомить читателей с практическими приемами работы с данными объектами.

Как и ранее, в процессе выполнения практических действий по тексту главы будет приводиться необходимая справочная информация, дающая представление об используемых программных конструкциях на встроенном языке 1С.

В этой главе будут рассмотрены примеры двух разработок, в которых *регистры сведений* играют ключевую роль. В этих конфигурациях мы продемонстрируем визуальные и программные действия с регистрами сведений.

Учет при организации работы филиалов

Рассмотрим ситуацию, когда в организации имеется несколько филиалов, каждый из которых включает ряд подразделений. И в каждом конкретном подразделении филиала определено ответственное лицо. Содержание табл. 3.1 отражает данную ситуацию.

Комбинация названия филиала и номера подразделения является уникальной, и она представляет собой *ключ* при записи либо при извлечении информации об ответственном за поставки. Определенной комбинации названия филиала и номера подразделения соответствует не более чем один ответственный за поставки (это функциональная зависимость, если проводить аналогию с математикой).

Таблица 3.1. *Информация об ответственных лицах*

Филиал	Подразделение	Ответственный за поставки
Окский	1	Петров Л. В.
Окский	2	Иванов О. В.
Волжский	1	Сидорова О. М.
Волжский	3	Николаев Л. Т.

Подготовка информационной базы

Начнем работу с создания новой конфигурации. Для хранения информации о филиалах сформируем перечисление с соответствующим названием — **Филиалы**. Создание объекта конфигурации данного типа (на этот раз не справочника) связано с тем, что перечень имеющихся филиалов в любой организации является неизменным на протяжении длительного времени.

Технически в окне редактирования объекта конфигурации следует сначала внести имя объекта на вкладке **Основные**, а затем на вкладке **Данные** добавить несколько значений в созданное перечисление (рис. 3.1).

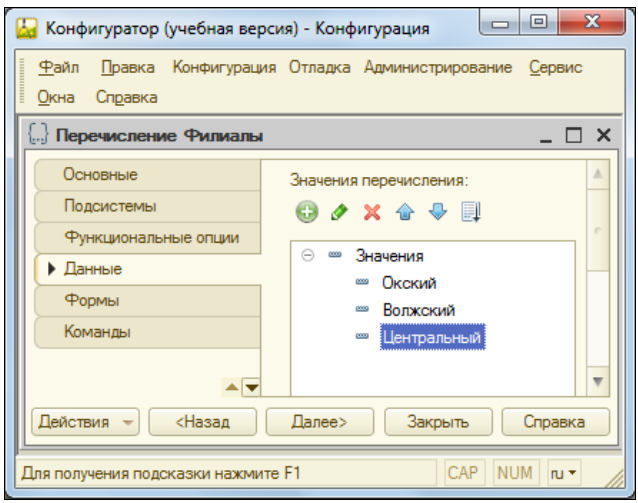


Рис. 3.1. Окно редактирования перечисления **Филиалы**

После этого внесем в нашу конфигурацию справочник "Сотрудники", который не будет содержать дополнительных реквизитов, а затем в режиме **1С:Предприятие** внесем в него несколько фамилий (рис. 3.2).

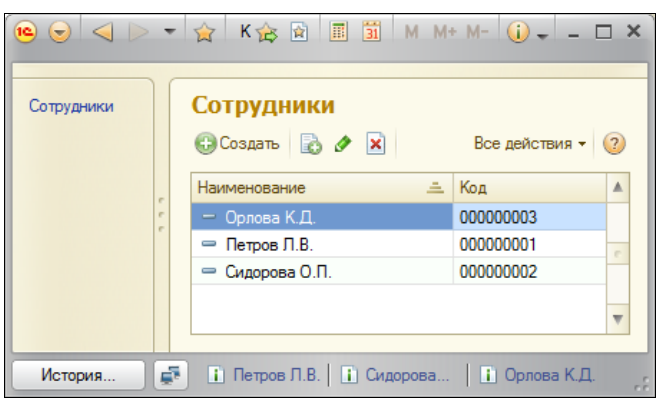


Рис. 3.2. Форма списка справочника "Сотрудники"

Для дальнейшей работы нам потребуется еще один вспомогательный справочник, который будет содержать названия товаров. Его назовем "Номенклатура". Дополнительных реквизитов в данный справочник вводить не будем, а что касается напрашивающегося реквизита **Цена**, то для хранения цен организуем периодический регистр сведений **Цены**. Это связано с тем, что по логике работы нам необходимо хранить информацию обо всех изменениях цен на товары. Так, на практике часто приходится выполнять ряд действий "задним числом". В этом случае знать значение цены товара на определенную дату необходимо. Следовательно, такой фактический параметр товара, как цена, должен храниться не в справочнике, а в другой информационной структуре (в регистре сведений).

Действия по созданию регистра сведений выполним в следующем разделе, а пока, после создания в конфигураторе справочника "Номенклатура", следует перейти в режим **1С:Предприятие** и внести в него несколько названий товаров (рис. 3.3).

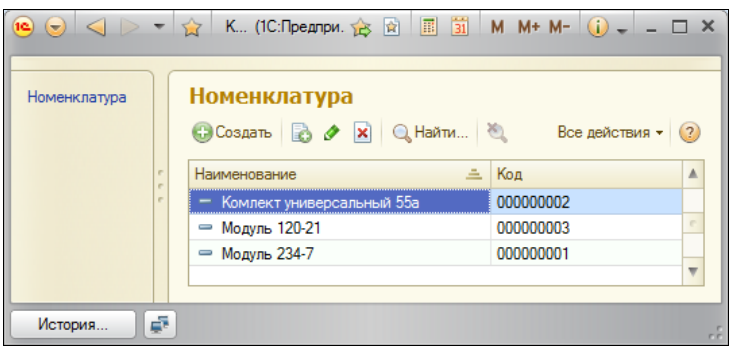


Рис. 3.3. Внесение информации в справочник "Номенклатура"

Непериодический регистр сведений

После создания справочников и перечисления перейдем к разработке *регистра сведений*. Основная задача регистра сведений заключается в хранении информации, состав которой развернут по определенной комбинации значений и при необходимости развернут во времени.

Регистр сведений представляет собой многомерный массив данных, необходимый для реализации функции, которая выдает нужную информацию по определенному набору аргументов. Эти аргументы называются *измерениями*, а результат функции — *ресурсами*.

Кроме измерений и ресурсов, в регистр сведений может быть добавлен набор *реквизитов*. Реквизиты позволяют включать в записи регистра сведений дополнительную информацию. И они, как правило, используются для анализа записей регистра сведений.

Перейдем к практическим действиям и добавим в конфигурацию новый объект. Технические действия по созданию объекта конфигурации хорошо знакомы — щелчок правой кнопкой мыши в дереве объектов конфигурации (в данном случае)

на элементе **Регистры сведений**. В результате на экране открывается контекстное меню, в котором следует выбрать пункт **Добавить**.

На вкладке **Основные** (рис. 3.4) кроме имени регистра сведений (**ОтветственныеЛица**) следует указать, что он является *непериодическим*. Это означает, что *время* в качестве дополнительного аргумента в создаваемом регистре использоваться не будет. В дальнейшем мы рассмотрим и пример создания периодического регистра сведений, однако сейчас нам это не требуется.

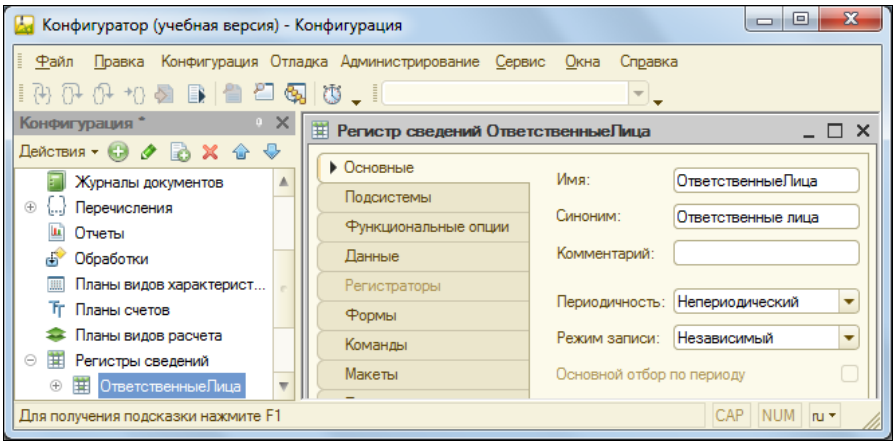


Рис. 3.4. Окно редактирования регистра сведений **ОтветственныеЛица**

На той же вкладке **Основные** (см. рис. 3.4) укажем в разделе **Режим записи** вариант — **Независимый**, что позволит напрямую производить внесение информации в регистр сведений в режиме **1С:Предприятие**. В противоположность этому при указании режима **Подчинение регистратору** появляется возможность формирования записей в регистре сведений автоматически как результат обработки информации из документов.

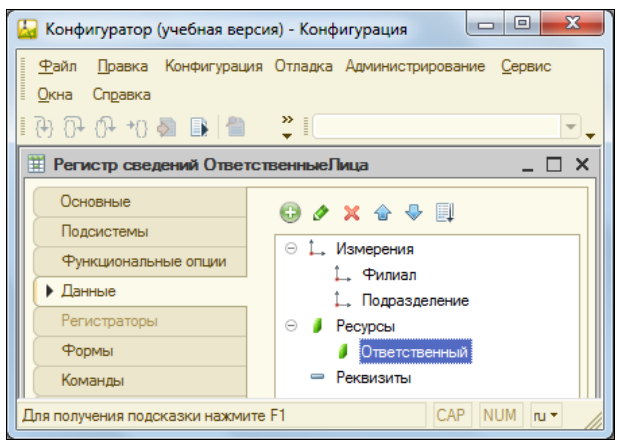


Рис. 3.5. Вкладка **Данные** окна редактирования регистра сведений **ОтветственныеЛица**

Перейдем на вкладку **Данные** окна редактирования объекта конфигурации, где укажем измерения и ресурсы регистра в соответствии со структурой табл. 3.1. Первое измерение регистра сведений — **Филиал**, для которого определим тип данных — **ПеречислениеСсылка.Филиалы**. Для второго измерения (**Подразделение**) выберем числовой тип данных, а имя ресурса регистра определим — **Ответственный** (тип данных — **СправочникСсылка.Сотрудники**).

В результате вкладка **Данные** окна редактирования рассматриваемого объекта конфигурации будет выглядеть так, как показано на рис. 3.5. На этом действия в конфигураторе по разработке регистра сведений **ОтветственныеЛица** завершены. Теперь можно перейти в режим **1С:Предприятие** и внести в созданный регистр несколько записей (рис. 3.6).

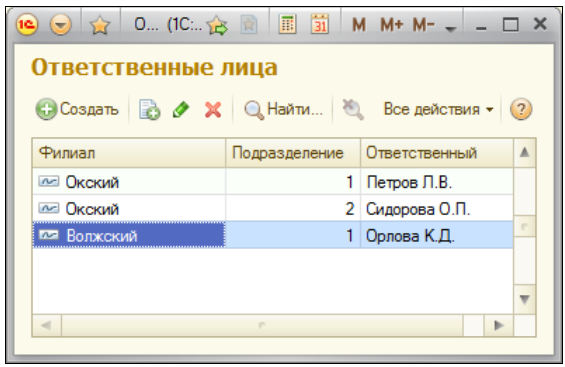


Рис. 3.6. Внесение информации в регистр сведений **ОтветственныеЛица**

Документ *Поступление*

Кроме регистра сведений в конфигурации, нам потребуется документ "Поступление". На рис. 3.7 показана вкладка **Данные** окна редактирования рассматриваемого объекта конфигурации.

В качестве типов данных дополнительных реквизитов документа "Поступление" установим:

- **Филиал** (тип данных — **ПеречислениеСсылка.Филиалы**);
- **Подразделение** (тип данных — **Число**);
- **Ответственный** (тип данных — **СправочникСсылка.Сотрудники**).

В табличной части **Перечень** данного документа определим набор реквизитов и соответствующие типы данных:

- **Название** (тип данных — **СправочникСсылка.Номенклатура**);
- **Цена** (тип данных — **Число**);
- **Количество** (тип данных — **Число**);
- **Сумма** (тип данных — **Число**).

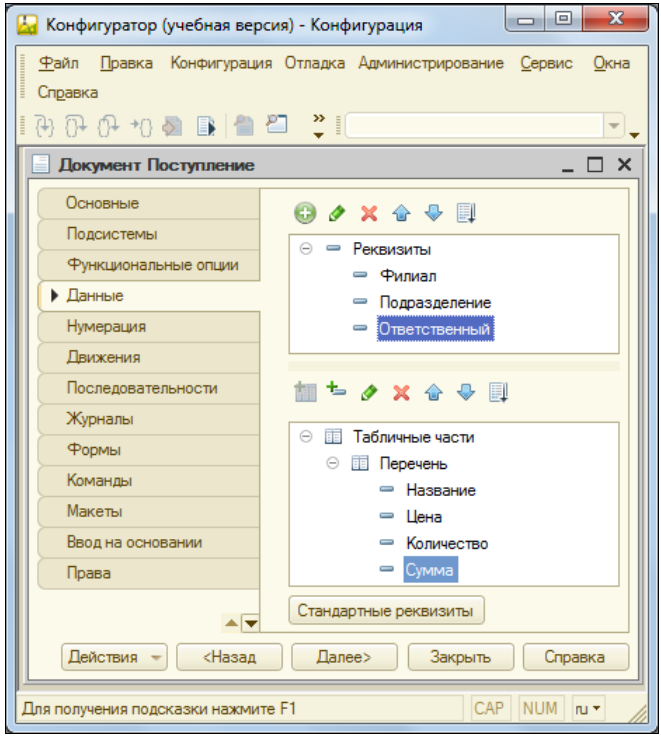


Рис. 3.7. Вкладка **Данные** окна редактирования документа "Поступление"

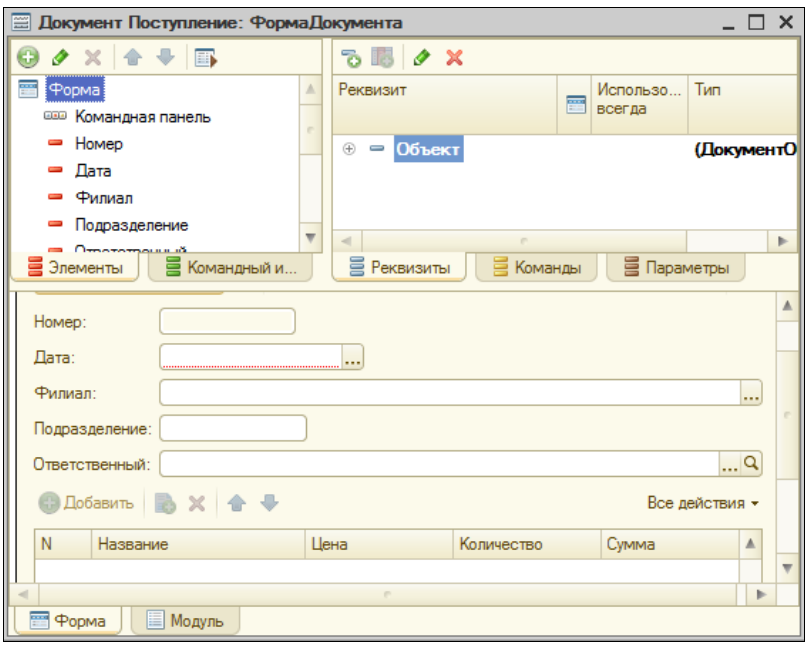


Рис. 3.8. Форма документа "Поступление"

Учитывая, что далее в этом разделе мы будем программировать события, связанные с формой документа, то необходимо создать пользовательскую форму документа. Технические действия по разработке формы основного объекта конфигурации нам знакомы по *главе 1*. Для этого следует перейти на вкладку **Формы** и воспользоваться *конструктором форм*. В результате несложных визуальных действий на экране откроется форма документа "Поступление" в окне редактора форм (рис. 3.8). В следующем разделе мы рассмотрим создание ряда сервисных возможностей для пользователя при работе с данной формой документа в режиме **1С:Предприятие**.

Процедура установки ответственного в поле документа

Во-первых, необходимо обеспечить автоматическое заполнение поля **Ответственный** после выбора названия филиала и номера подразделения в шапке документа "Поступление". Предпосылка для такого заполнения основывается на информации, содержащейся в регистре сведений **ОтветственныеЛица**, где для определенной комбинации филиала и номера подразделения указано определенное ответственное лицо.

Таким образом, мы должны обеспечить заполнение поля **Ответственный** после выбора филиала и номера подразделения в документе "Поступление".

Для определенности сначала разберем выполнение необходимого действия при вводе в поле **Подразделение** номера подразделения.

Для этого двойным щелчком по полю **Подразделение** (см. рис. 3.8) откроем палитру его свойств (рис. 3.9) и обратимся к разделу **События**. Здесь мы должны соответствующим образом оформить обработку события `ПриИзменении`. Для этого следует щелкнуть по изображению лупы напротив надписи данного события. В результате мы попадем в заготовку программной процедуры.

Теперь необходимо разработать программный код, в котором должен производиться запрос к регистру сведений **ОтветственныеЛица** для извлечения необходимой информации. Эти действия могут быть выполнены только на сервере, поэтому программный код будет состоять из двух компонентов:

- процедуры `ПодразделениеПриИзменении`, которая выполняется в контексте клиента;
- функции `ПолучитьОтветственного`, которая выполняется в контексте сервера и вызывается из клиентской процедуры `ПодразделениеПриИзменении`.

Текст процедуры `ПодразделениеПриИзменении` приведен в листинге 3.1. В ней мы фактически только обращаемся к серверной функции `ПолучитьОтветственного`. При этом в данной функции используются значения двух реквизитов документа:

- **Филиал**;
- **Подразделение**.

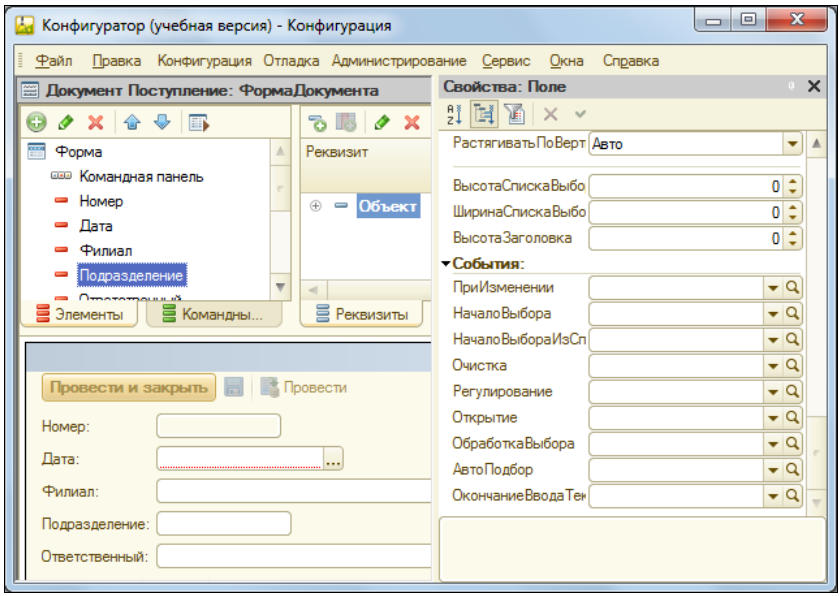


Рис. 3.9. Палитра свойств поля **Подразделение**

Листинг 3.1. Процедура, выполняемая при выборе номера подразделения

```
&НаКлиенте
Процедура ПодразделениеПриИзменении(Элемент)
  Объект.Ответственный=ПолучитьОтветственного();
КонецПроцедуры
```

Текст функции `ПолучитьОтветственного` приведен в листинге 3.2. Там производится запрос к регистру сведений **ОтветственныеЛица**:

```
Запрос.Текст = "ВЫБРАТЬ *
| ИЗ  РегистрСведений.ОтветственныеЛица
| ГДЕ Филиал=&Филиал И
| Подразделение =&Подразделение";
```

> ***Примечание***
>
> Запросы представляют собой очень важные объекты, им отведена *глава 4* книги. Там синтаксические конструкции запросов будут рассмотрены подробнее.

Параметрами запроса являются **Филиал** и **Подразделение**. При определении параметров в тексте запроса используется специальный символ `&`. Для установки значений параметров запроса во внутреннем языке 1С имеется метод `УстановитьПараметр`.

В результате выполнения рассматриваемого запроса возможны две ситуации:

- имеется единственная запись об интересующем нас человеке, который является ответственным в определенном отделе конкретного филиала;
- записи с указанными измерениями нет.

В первом случае мы возвращаем извлеченную информацию об ответственном в исходную процедуру `ПодразделениеПриИзменении`. Далее фамилия ответственного лица подставляется в поле **Ответственный**. Во втором же случае никакого заполнения поля **Ответственный** не производится.

Листинг 3.2. Серверная функция ПолучитьОтветственного

```
&НаСервере
Функция ПолучитьОтветственного()
  Запрос = Новый Запрос;
  Запрос.Текст = "ВЫБРАТЬ *
  | ИЗ  РегистрСведений.ОтветственныеЛица
  | ГДЕ Филиал=&Филиал И
  | Подразделение =&Подразделение";
  Запрос.УстановитьПараметр("Филиал",
        Объект.Филиал);
  Запрос.УстановитьПараметр("Подразделение",
        Объект.Подразделение);
  Результат = Запрос.Выполнить().Выбрать();
  Если Результат.Следующий() > 0 Тогда
    Возврат Результат.Ответственный;
  КонецЕсли;
КонецФункции
```

Теперь перейдем в режим **1С:Предприятие** и убедимся в работоспособности разработанного программного кода (рис. 3.10).

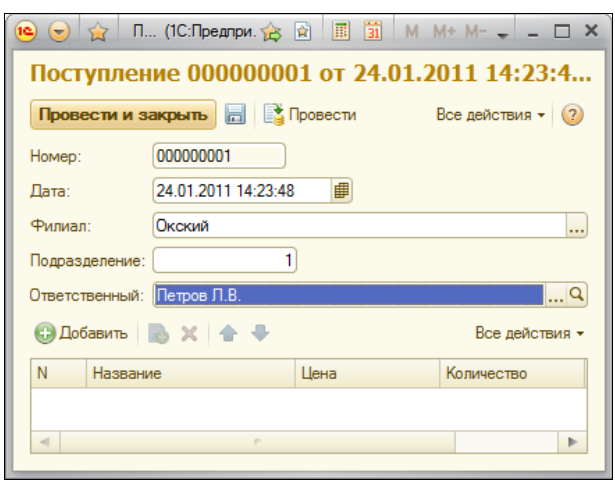


Рис. 3.10. Заполнение поля документа информацией из регистра сведений

В данном случае взаимодействие процедуры `ПодразделениеПриИзменении` и функции `ПолучитьОтветственного` организовано так, что в функцию необходимо передавать

весь контекст формы (это связано с тем, что в тексте функции мы используем переменные Объект.Филиал и Объект.Подразделение).

Примечание

Имя Объект представляет собой имя основного реквизита формы документа, с помощью которого можно работать с реквизитами документа.

Можно реализовать более эффективный вариант клиент-серверного взаимодействия. Новые реализации рассматриваемых процедуры и функции приведены соответственно в листингах 3.3 и 3.4. Здесь мы использовали для функции директиву &НаСервереБезКонтекста, что в результате приводит к экономии при передаче данных (за счет того, что на сервер не передается контекст формы).

Листинг 3.3. Процедура, выполняемая при выборе номера подразделения (вариант 2)

```
&НаКлиенте
Процедура ПодразделениеПриИзменении(Элемент)
Объект.Ответственный=ПолучитьОтветственного(Объект.Филиал,
        Объект.Подразделение);
КонецПроцедуры
```

Листинг 3.4. Серверная функция ПолучитьОтветственного (вариант 2)

```
&НаСервереБезКонтекста
Функция ПолучитьОтветственного(НашФилиал,НашОтветственный)
  Запрос = Новый Запрос;
  Запрос.Текст = "ВЫБРАТЬ *
  | ИЗ  РегистрСведений.ОтветственныеЛица
  | ГДЕ Филиал=&Филиал И
  | Подразделение =&Подразделение";
  Запрос.УстановитьПараметр("Филиал",
           НашФилиал);
  Запрос.УстановитьПараметр("Подразделение",
           НашОтветственный);
  Результат = Запрос.Выполнить().Выбрать();
  Если Результат.Следующий() > 0 Тогда
     Возврат Результат.Ответственный;
  КонецЕсли;
КонецФункции
```

Для придания данной разработке логической завершенности аналогичную процедуру необходимо выполнять и при выборе филиала, поскольку мы не можем гарантировать конкретный порядок заполнения рассматриваемых полей пользователем. Для этого в редакторе формы документа откроем палитру свойств поля **Филиал** и обратимся к событию ПриИзменении. Текст процедуры, которую необходимо создать, представлен в листинге 3.5.

Листинг 3.5. Процедура, выполняемая при выборе филиала

```
&НаКлиенте
Процедура ФилиалПриИзменении(Элемент)
    Объект.Ответственный=ПолучитьОтветственного(Объект.Филиал,
        Объект.Подразделение);
КонецПроцедуры
```

Таким образом, что касается автоматизации при заполнении шапки документа "Поступление", то поставленная задача выполнена. Теперь обеспечим удобный сервис для табличной части документа. Однако при этом необходимо сформировать цены в периодическом регистре сведений, что мы и сделаем в следующем разделе.

Создание периодического регистра сведений *Цены*

Периодический регистр сведений обладает способностью хранить информацию в привязке ко времени. В этом случае в регистре сведений находятся и актуальные (последние) значения, и все предшествующие. Периодичность регистра сведений можно определить одним из следующих значений:

- в пределах секунды;
- в пределах дня;
- в пределах месяца;
- в пределах квартала;
- в пределах года;
- в пределах регистратора (если установлен режим записи — **Подчинение регистратору**).

Периодический регистр сведений содержит поле **Период**, добавляемое системой 1С:Предприятие автоматически. Тип данных поля **Период** — **Дата**, а назначение данного поля связано с соотнесением конкретной записи в регистре с определенным периодом. Важно отметить, что когда производится запись в регистр, то автоматически осуществляется ее привязка к началу периода, в который она попадает.

> ***Примечание***
>
> Свойство **Период** позволяет указать, с какой периодичностью регистр сведений должен сохранять значения ресурсов. Для непериодического регистра можно получить только последнее введенное значение (предыдущие значения не сохраняются). Для периодического регистра существует единственное значение ресурса внутри периода. Если указан период — *месяц*, то в регистре фиксируется одно значение для каждого месяца. Вариант периодичности *секунда* реально не накладывает ограничений при вводе записей в любой периодический регистр сведений.

Система самостоятельно автоматически контролирует уникальность записи в периодический регистр сведений. При этом ключом записи является совокупность значений измерений регистра и периода.

Продемонстрируем работу с регистром сведений на конкретном примере. Наша задача заключается в том, чтобы создать периодический регистр сведений **Цены** (рис. 3.11). В качестве значения периодичности укажем — **В пределах секунды**. После этого на вкладке **Данные** создадим измерение **Номенклатура** (тип данных — **СправочникСсылка.Номенклатура**). Разумно указать, что данное поле является ведущим (рис. 3.12). В этом случае при удалении из справочника "Но-

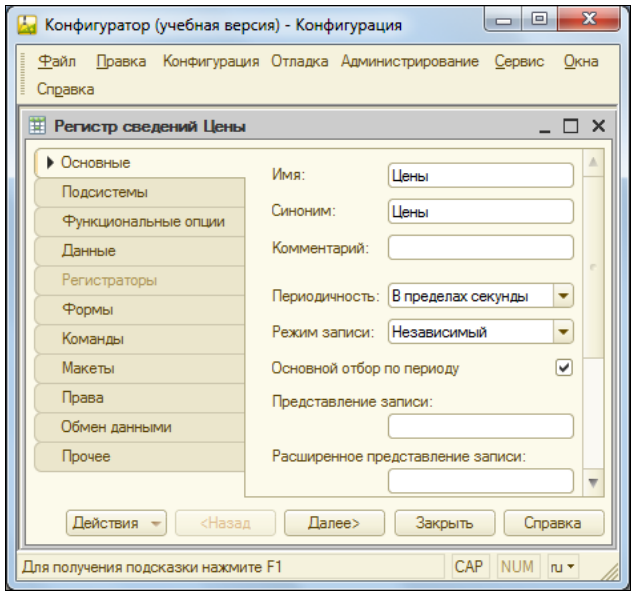


Рис. 3.11. Вкладка **Основные** регистра сведений **Цены**

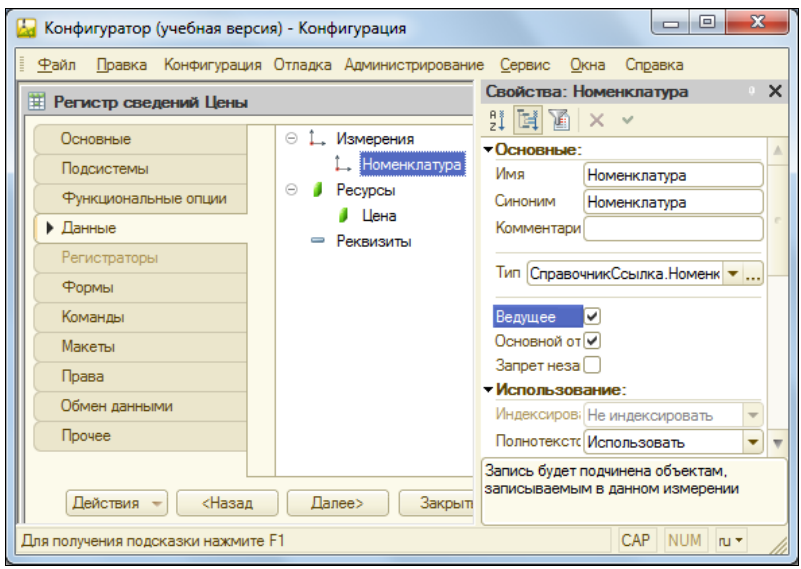


Рис. 3.12. Вкладка **Данные** регистра сведений **Цены**

менклатура" определенного элемента автоматически будут удалены все записи из регистра сведений, в которых указан данный элемент справочника. Также создадим в рассматриваемом регистре ресурс **Цена** (тип данных — **Число**). После этого перейдем в режим **1С:Предприятие** и внесем в созданный регистр сведений несколько записей о ценах (рис. 3.13).

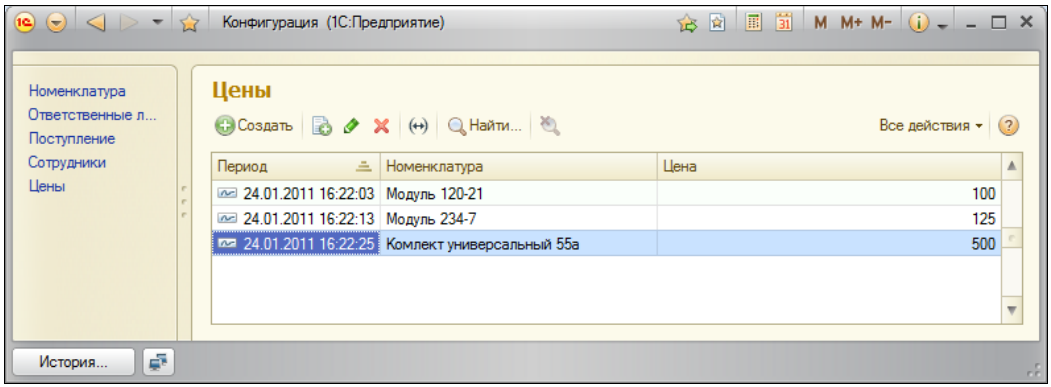


Рис. 3.13. Внесение информации о ценах в регистр сведений **Цены**

Использование регистра *Цены* в документе *Поступление*

Сформулируем теперь задачу, которую нам следует решить. В документе "Поступление" необходимо после выбора товара обеспечить автоматическую подстановку цены на указанную дату. Для этого следует обратиться к форме данного документа в окне редактора форм.

Наша задача заключается в том, чтобы написать процедуру, выполняемую при выборе названия товара в табличной части документа. Для этого в палитре свойств поля `ПереченьНазвание` следует перейти к обработке события `ПриИзменении` (рис. 3.14).

Первая программа, которую нам необходимо разработать, представлена в листинге 3.6. Это процедура `ПереченьНазваниеПриИзменении`, которая автоматически выполняется при выборе названия товара в поле `ПереченьНазвание`. Здесь сначала с помощью переменной `СтрТаб` мы получаем доступ к текущей редактируемой строке табличной части документа. При этом мы воспользовались свойством `Элементы` объекта `УправляемаяФорма`. Каждый элемент формы можно получить, указав его имя в качестве свойства этого объекта, т. е. через точку от него. В данном случае мы обращаемся к табличной части нашего документа: `Элементы.Перечень`. Для получения редактируемой строки мы воспользовались свойством `ТекущиеДанные`.

После этого производится обращение к дополнительной функции `ПолучитьЦену`, которую нам также необходимо создать (листинг 3.7). Заметим, что данная функция выполняется на сервере, но без передачи контекста формы.

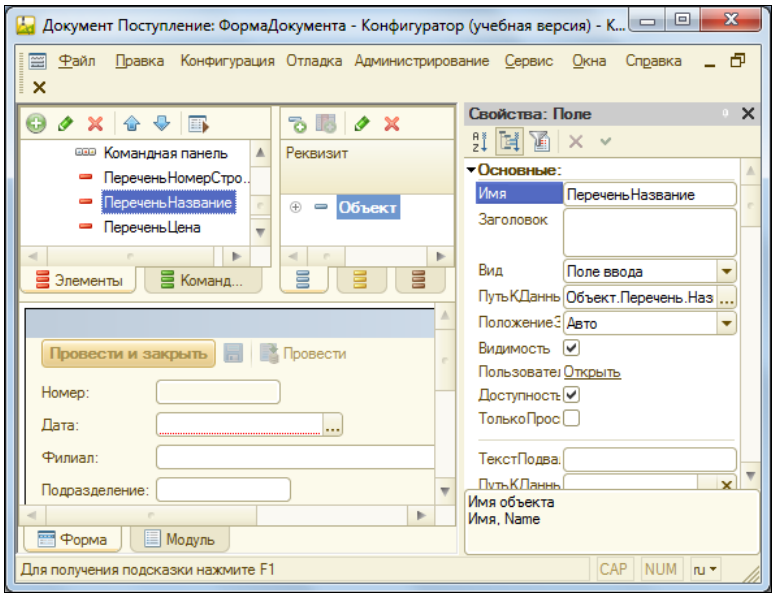


Рис. 3.14. Палитра свойств поля `ПеречеьНазвание`

Листинг 3.6. Процедура, выполняемая при выборе названия товара

```
&НаКлиенте
Процедура ПеречеьНазваниеПриИзменении(Элемент)
  СтрТаб = Элементы.Перечень.ТекущиеДанные;
  СтрТаб.Цена=ПолучитьЦену(СтрТаб.Название,Объект.Дата);
КонецПроцедуры
```

Листинг 3.7. Серверная функция ПолучитьЦену

```
&НаСервереБезКонтекста
Функция ПолучитьЦену(Товар,ТекДата)
  ОтборЦены = Новый Структура("Номенклатура", Товар);
  ЗначениеРесурсов =
     РегистрыСведений.Цены.ПолучитьПоследнее(ТекДата, ОтборЦены);
   Возврат ЗначениеРесурсов.Цена;
КонецФункции
```

В функции `ПолучитьЦену` используется метод `ПолучитьПоследнее`, требующий некоторого комментария. Вообще для получения информации о значении ресурса регистра сведений можно использовать метод `Получить`. В этом случае мы можем выбрать конкретное значение ресурса, соответствующее набору измерений (в данной ситуации у нас одно измерение) и значению периода. Например, это можно выполнить с помощью следующего программного кода:

```
ЗначениеРесурсов=РегистрыСведений.Цены.Получить(ТекДата,ОтборЦены);
```

Использование подобной конструкции вполне допустимо, но при условии, что для каждой даты в регистре сведений занесено значение тарифа. Разумеется, в реальной ситуации это не так. Поэтому если на указанную дату информации в регистре сведений нет, то метод `Получить` вернет пустое значение.

Используемый в нашем случае метод `ПолучитьПоследнее` позволяет извлечь из регистра сведений последнюю по времени внесения цену (не позднее указанной даты) для выбранного товара. Как раз это нам и требуется для рассматриваемой задачи.

Таким образом, функция, представленная в листинге 3.7, вносит правильное значение цены на дату документа. После этого в форме документа пользователю необходимо ввести количество, что приведет к автоматическому расчету суммы. Для этого требуется соответствующим образом оформить процедуру `ПриИзменении` для поля `ПереченьКоличество` (листинг 3.8).

Листинг 3.8. Процедура заполнения поля Сумма после внесения количества товаров

```
Процедура ПереченьКоличествоПриИзменении(Элемент)
  Товар = Элементы.Перечень.ТекущиеДанные;
  Товар.Сумма = Товар.Цена * Товар.Количество;
КонецПроцедуры
```

Теперь разработка готова, далее следует перейти в режим **1С:Предприятие** и проверить созданную функциональность (рис. 3.15).

Таким образом, мы на конкретном примере разобрали ресурсы для обеспечения необходимого сервиса пользователю. В следующем разделе будет рассмотрен еще один пример конфигурации на тему использования регистров сведений.

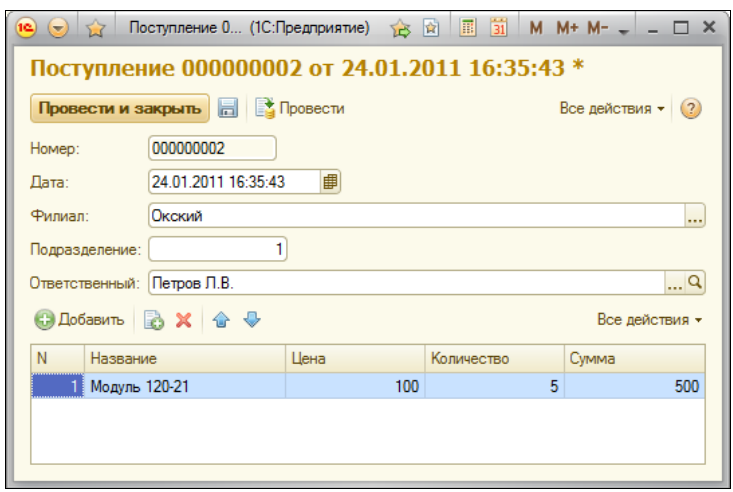


Рис. 3.15. Результат заполнения табличной части документа "Поступление"

Пример информационной базы для учета квитанций

Рассматриваемая здесь прикладная задача отражает ситуацию, которая часто встречается при предоставлении различными организациями платных образовательных услуг. В практической работе в большинстве случаев используется автоматизация подобного учета, и для этого применяются разнообразные программные продукты — Microsoft Excel, Microsoft Access и др. Однако часто после автоматизации большинства участков работы все равно остаются еще и ручные действия (например, заполнение квитанций для оплаты в отделении банка).

В данном случае нам требуется спроектировать конфигурацию, позволяющую эффективно контролировать внесение оплаты за обучение. Созданная разработка для платформы 1С:Предприятие 8.2 будет обеспечивать выписку квитанций на оплату и помогать вести учет сделанных учащимися платежей. На рассматриваемом далее примере мы попробуем показать, как можно практически полностью разгрузить офисного работника от рутинных вычислений и ручного заполнения бланков квитанций.

В качестве схемы оплаты взят вариант, используемый в одном из высших учебных заведений. Для определенности будем считать, что наша организация устанавливает помесячную индексацию оплаты за обучение. Поясним, как это реально выглядит на примере. Допустим, что для зачисленного (поступившего) учащегося срок обучения по выбранной им специальности составляет 36 месяцев, но месячная стоимость обучения при этом возрастает с течением времени (точнее, от месяца к месяцу). И если в сентябре 2011 г. стоимость за месяц составляла 2000 рублей, то, заплатив именно в сентябре 2011 г. по этому тарифу всю сумму (72 000 рублей), учащийся больше ничего учебному заведению не должен. Если же он по своему желанию прекратит обучение, то учебное заведение возвратит ему сумму за не полностью предоставленную услугу (за оставшиеся месяцы после прекращения учебы).

В реальной ситуации учащиеся платят не за весь период, а только за несколько месяцев вперед, и технически это выглядит следующим образом. Учащемуся предварительно выписывается квитанция, которая фиксируется в неком реестре (в информационной базе). Оплатив данную квитанцию в отделении банка и отметив это в соответствующем отделе учебного заведения, он получает следующую квитанцию (на один или несколько следующих месяцев), оплачивает ее и т. д.

Наша задача заключается в создании удобной технологии для ведения описанного учета сотрудниками учебного заведения. Автоматизация должна привести к сокращению времени выполнения технических операций и минимизации количества ошибок в работе. Для более точного восприятия ситуации в табл. 3.2 представлен фрагмент информации о тарифах месячных оплат.

В представленной таблице содержание колонки "Вид тарифа" определяет различные формы обучения — полная учебная программа, ускоренная, сокращенная, индивидуальная и т. д. Каждому учащемуся в зависимости от выбранной им формы

обучения устанавливается определенный вид тарифа. Конкретная уникальная комбинация номера месяца, года и вида тарифа по табл. 3.2 позволяет узнать стоимость обучения в определенном месяце в указанном учебном году по выбранной учащимся форме обучения. Эта комбинация и является ключом при записи либо при извлечении информации о месячной стоимости обучения.

Таблица 3.2. *Информация о тарифах месячных оплат*

Номер месяца	Год	Вид тарифа	Стоимость
2	2011	Основной	2000
3	2011	Основной	2020
2	2011	Сокращенная форма	2100
3	2011	Сокращенная форма	2150

Другие важные данные для создания системы учета оплаты представлены в табл. 3.3. Это необходимые сведения об учащихся для ведения учета оплаты. В приведенной таблице каждая строка представляет собой запись о конкретном человеке. При зачислении учащегося с ним заключается договор, информация о котором и отражается в первой колонке таблицы. В последующие поля заносятся данные об учащемся и плательщике (часто оплату вносит один из родственников), а в графу "Число оплачиваемых месяцев" вносится количество месяцев обучения по выбранной форме. Таким образом, это и есть то количество месяцев по договору, которые человек должен оплатить. Оплатить он может сразу при поступлении (тогда индексации не будет), а при постепенном внесении платежей должно учитываться изменение месячной стоимости с течением времени.

Таблица 3.3. *Информация о договорах*

Номер договора	ФИО студента	ФИО плательщика	Число оплачиваемых месяцев	Вид тарифа
100/234	Петров Петр Петрович	Петров Иван Иванович	36	Основной
101/234	Иванов Иван Иванович	Иванов Петр Иванович	36	Основной
102/234	Сидорова Ольга Николаевна	Сидорова Ольга Николаевна	32	Сокращенная форма
103/234	Сидоров Олег Петрович	Сидоров Олег Петрович	32	Сокращенная форма

Таким образом, цель автоматизации сформулирована. И теперь можно перейти к разработке конфигурации, которая позволит организовать на платформе 1С:Пред-

приятие 8.2 удобную систему для ведения учета оплаты. Информация, приведенная в табл. 3.2 и 3.3, будет рассматриваться в качестве исходных данных для создаваемой системы учета.

Для отражения информации, содержащейся в каждой из упоминаемых таблиц, создадим соответствующий программный аналог. После этого, добавив необходимые компоненты интерфейса и запрограммировав алгоритмы обработки, мы получим удобную информационную систему для организации учета оплаты за обучение.

Представление информации о тарифах

Вернемся к табл. 3.2. Приведенные в ней данные с точки зрения математического представления можно трактовать как функцию от трех переменных, в качестве которых выступают год, месяц и вид тарифа. Для каждого уникального набора этих данных в таблице имеются не более одного значения функции — стоимости месяца обучения.

Первой практической задачей в режиме конфигуратора системы 1С:Предприятие, которую мы решим, будет создание регистра сведений под названием **Тарифы**, предназначенного для хранения информации об имеющихся тарифах. Разумеется, эту информацию пользователю будет разрешено вносить в регистр, также как и просматривать. Если взглянуть на табл. 3.2, то видно, что содержимое колонки "Вид тарифа" представляет собой совокупность значений из достаточно ограниченного набора — форм обучения реально немного (как правило, не более пяти). В этом случае для заполнения колонки "Вид тарифа" удобнее создать вспомогательный справочник с аналогичным названием. Тогда, заполнив его один раз названиями тарифов, далее при внесении информации в колонку "Вид тарифа" мы будем просто выбирать значения из данного справочника.

Итак, теперь можно считать вступительную часть, касающуюся данной задачи, завершенной. И со следующего раздела мы перейдем к практическим действиям по созданию новой конфигурации.

Вспомогательный справочник *ВидТарифа*

Как и в предыдущих главах, начнем работу в режиме конфигуратора с пустой информационной базы. Первым объектом новой конфигурации будет справочник "ВидТарифа" (рис. 3.16).

Данный справочник достаточно прост, и нам не потребуется создавать экранные формы для работы с ним — вполне достаточно воспользоваться формами элемента и списка, генерируемыми системой по умолчанию. На рис. 3.17 показан фрагмент заполнения этого справочника возможными видами тарифов в режиме **1С:Предприятие**.

После этого, имея список названий видов тарифов, можно перейти в следующем разделе к созданию регистра сведений **Тарифы**.

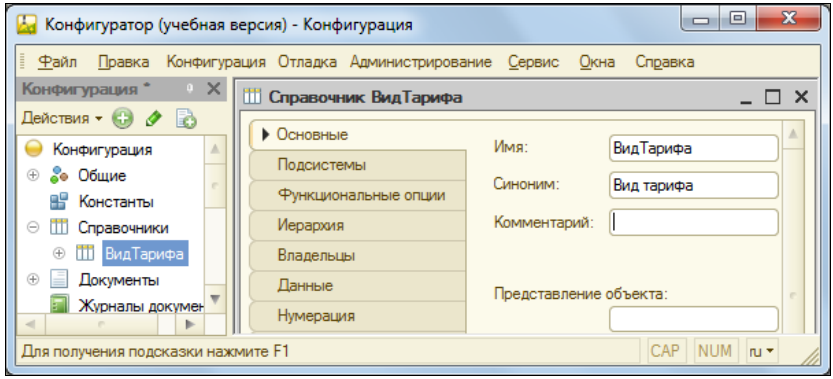


Рис. 3.16. Окно редактирования справочника "ВидТарифа"

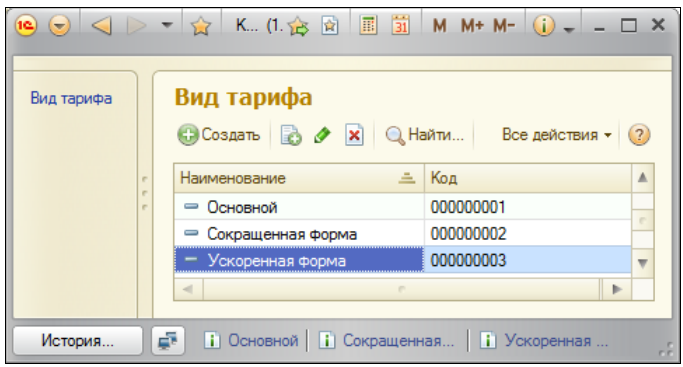


Рис. 3.17. Справочник "ВидТарифа" в режиме **1С:Предприятие**

Регистр сведений *Тарифы*

Начальные действия по созданию объекта конфигурации такого типа нам знакомы — щелчок правой кнопкой мыши на названии **Регистры сведений** в окне, содержащем дерево объектов конфигурации, и последующий выбор пункта **Добавить** в открывшемся контекстном меню. В результате перед нами откроется окно редактирования объекта конфигурации, где на вкладке **Основные** (рис. 3.18) кроме имени регистра необходимо указать: **Непериодический**. Это означает, что *время* в качестве дополнительного аргумента в создаваемом регистре использоваться не будет.

На той же вкладке в процессе создания данного регистра в поле **Режим записи** укажем **Независимый** — в этом случае мы можем напрямую производить запись информации в регистр сведений. В противоположность этому при указании режима **Подчинение регистратору** появляется возможность формировать записи в регистре автоматически в результате обработки информации из документов. В следующем разделе мы рассмотрим и эту возможность, однако для данного регистра сведений будем использовать независимый режим записи.

Перейдем на вкладку **Данные** (рис. 3.19), где укажем измерения и ресурсы регистра в соответствии с табл. 3.2. Первое измерение — **НомерМесяца**, для которого выберем числовой тип данных. Аналогичные установки выполним и для следующего измерения — **Год**. В соответствии с табл. 3.2 третьим измерением является **ВидТарифа**, в качестве типа данных которого следует установить **Справочник-Ссылка.ВидТарифа**. Кроме того, отметим с помощью соответствующего флажка в палитре свойств, что данное измерение является *ведущим*.

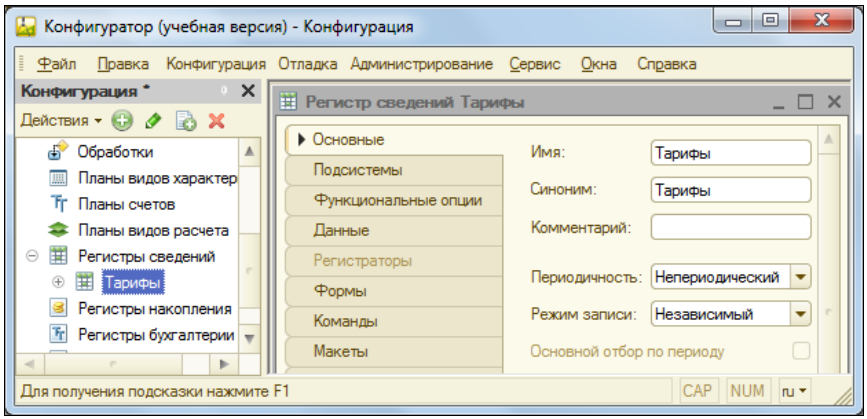


Рис. 3.18. Окно редактирования регистра сведений **Тарифы**

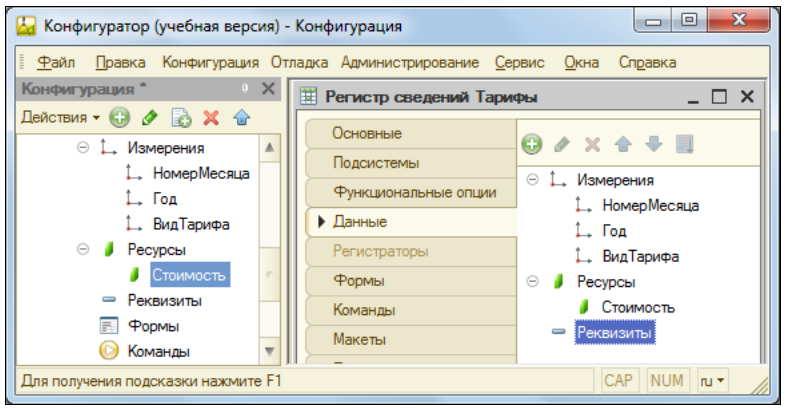


Рис. 3.19. Вкладка **Данные** окна редактирования регистра сведений **Тарифы**

Ресурсом регистра сведений **Тарифы** назначим **Стоимость** (числовой тип данных). Теперь можно сказать, что структура регистра для хранения значений тарифов создана. Далее следует перейти в режим **1С:Предприятие** и заполнить регистр сведений **Тарифы** значениями нескольких тарифов. Вполне достаточно для этого воспользоваться формой ввода по умолчанию. На рис. 3.20 приведен фрагмент заполнения регистра сведений **Тарифы** данными о тарифах.

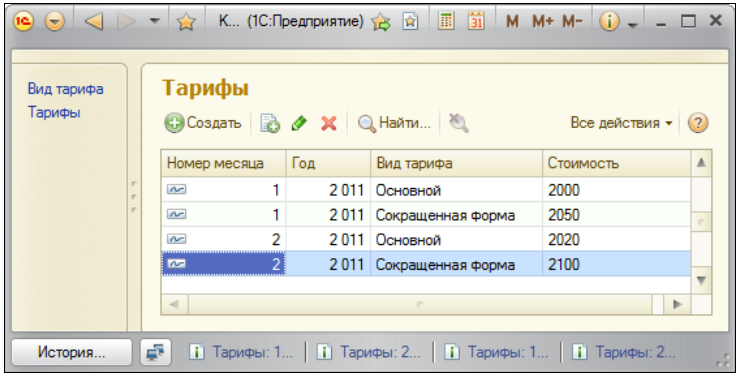


Рис. 3.20. Форма списка регистра сведений **Тарифы**

Контроль ошибок в форме регистра *Тарифы*

Сделаем еще одну доработку, связанную с автоматической проверкой возможных ошибок при вводе информации. Так, к настоящему моменту форма списка регистра сведений **Тарифы** не обеспечивает контроля вводимых данных. Наша цель — установить интервал целых чисел от 1 до 12 при вводе номера месяца, а также задать разумные интервалы при указании года (скажем от 2011 до 2020). Все это уменьшит возможные технические неточности при работе пользователя.

Для этого следует обратиться к палитре свойств измерения **НомерМесяца** (рис. 3.21). Здесь используем свойства **Максимальное значение** и **Минимальное значение**, которые установим в соответствии с рис. 3.21. После этого аналогичные действия выполним для поля **Год** (рис. 3.22).

Таким образом, в результате проделанной работы мы получили электронный аналог табл. 3.2, но при этом добавили в нее автоматический контроль ошибок при

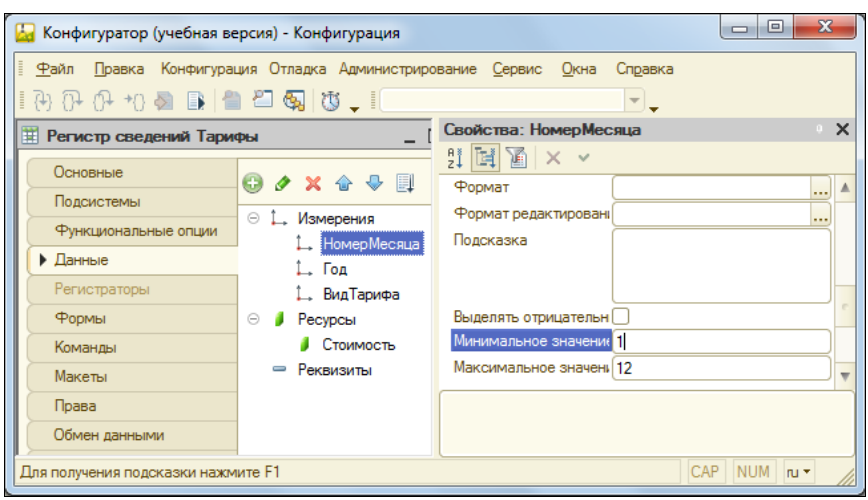


Рис. 3.21. Палитра свойств поля **НомерМесяца**

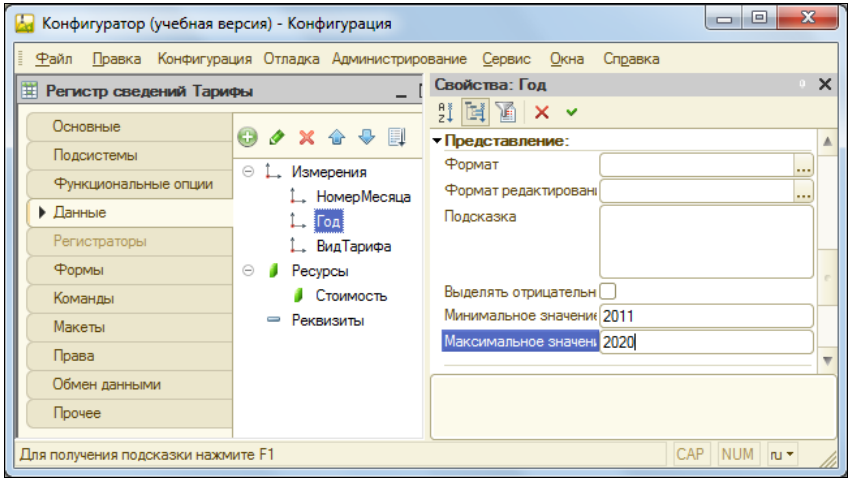


Рис. 3.22. Палитра свойств поля **Год**

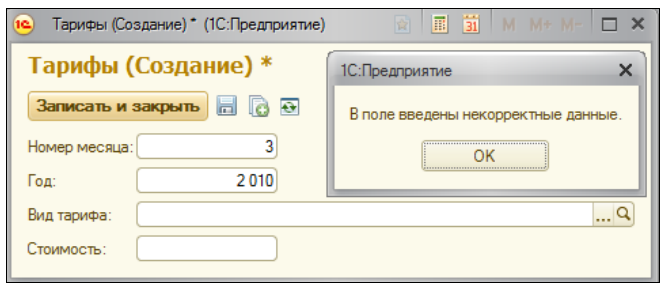


Рис. 3.23. Проверка корректности вводимых данных в поле **Год**

вводе. На рис. 3.23 представлена реакция платформы при внесении "некорректного" года.

Справочник *Договоры*

Следующая задача заключается в создании электронного варианта табл. 3.3. Содержащаяся в ней информация достаточно статична, и поэтому для ее хранения лучше всего подходит справочник.

Так что на очереди у нас создание справочника "Договоры", который будет соответствовать структуре табл. 3.3. При его разработке, кроме обязательных полей кода и наименования, добавим следующие реквизиты (рис. 3.24):

- **ФИОучащегося** (тип данных — строка не более 100 символов);
- **ФИОплательщика** (тип данных — строка не более 100 символов);
- **ДатаДоговора** (тип данных — **Дата**);
- **ВидТарифа** (тип данных — **СправочникСсылка.ВидТарифа**);
- **КоличествоОплачиваемыхМесяцев** (тип данных — **Число**).

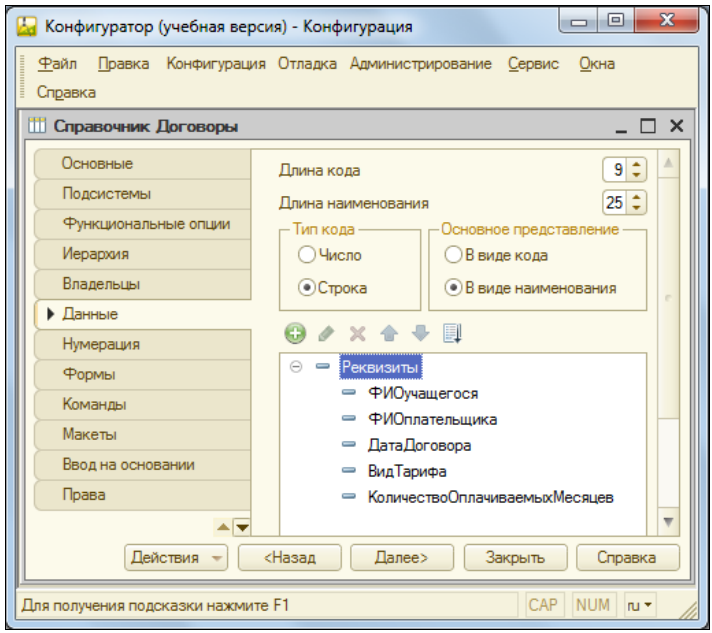


Рис. 3.24. Вкладка **Данные** окна редактирования справочника "Договоры"

Значение параметра **КоличествоОплачиваемыхМесяцев** определяет, сколько месяцев обучения студент должен оплатить по заключаемому договору. После создания указанных реквизитов каких-либо еще действий для доработки справочника не требуется — при работе в режиме **1С:Предприятие** мы воспользуемся формами, генерируемыми системой по умолчанию. Пример заполнения формы элемента справочника "Договоры" приведен на рис. 3.25.

Таким образом, теперь у нас есть объекты конфигурации для хранения информации о стоимости тарифов за обучение и для фиксации сведений о договорах. Следующая задача заключается в создании механизма выписки и регистрации квитанций. Это как раз и составляет основную содержательную часть разработки.

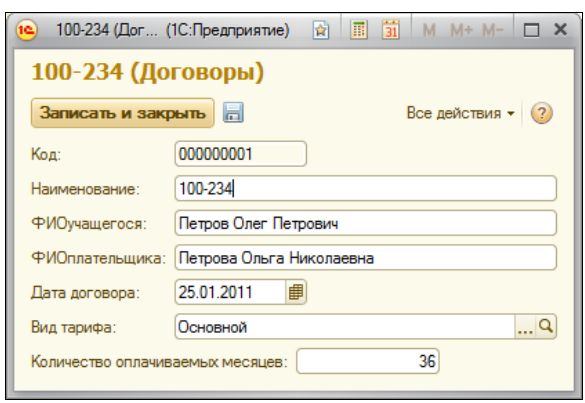


Рис. 3.25. Форма элемента справочника "Договоры" в режиме **1С:Предприятие**

Регистр сведений *Квитанции*

Перейдем к разработке регистра сведений **Квитанции**, в котором будет храниться информация о выписанных квитанциях, а также отмечаться факт их оплаты через отделение банка. Данный регистр сделаем периодическим, что и отметим на вкладке **Основные** в поле **Периодичность** (рис. 3.26). Для решаемой нами задачи разумно указать в поле **Периодичность** значение **В пределах секунды**, что позволит хранить практически любое количество квитанций по конкретному договору. Также на этой же вкладке установим **Подчинение регистратору** в качестве режима записи в рассматриваемый регистр.

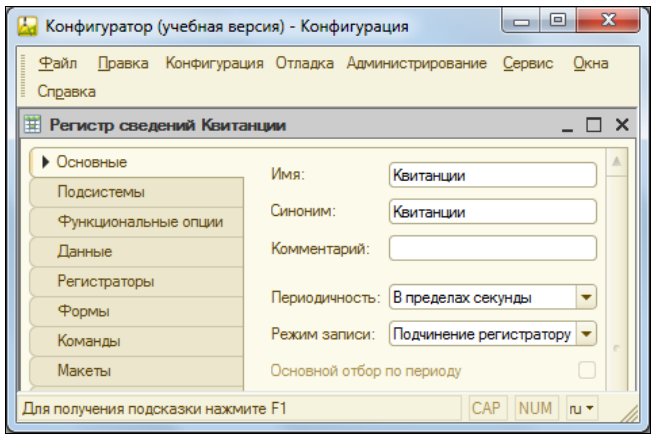


Рис. 3.26. Окно редактирования регистра **Квитанции**

Теперь перейдем на вкладку **Данные**, где от нас требуется задать структуру полей регистра (рис. 3.27). Измерений в регистре **Квитанции** будет два — **Договор** и **Период** (второе измерение, как уже говорилось, автоматически добавляется платформой для всех периодических регистров).

Для поля **Договор** в качестве типа данных укажем **СправочникСсылка.Договоры**, а также отметим, что данное поле является ведущим. В этом случае при удалении объекта информационной системы автоматически производится удаление записей регистра сведений, в которых имеется ссылка на удаляемый объект. Например, записи квитанций по конкретному договору будут автоматически удалены системой при удалении договора.

Для ресурса **СуммаПоКвитанции** следует указать числовой тип данных. Каждая запись в регистре сведений информирует о конкретной квитанции, вот для отражения денежной суммы выписанной квитанции и отводится данный ресурс.

Реквизиты регистра сведений несут уточняющую информацию по каждой записи в нем. Так, для поля **Оплачено** следует установить булевый тип данных, после чего в указанном поле можно отмечать, произведена ли оплата по данной квитанции. Для других реквизитов (**МесяцНачала**, **ГодНачала**, **ОплаченныеМесяцы**) установим числовой целочисленный тип данных.

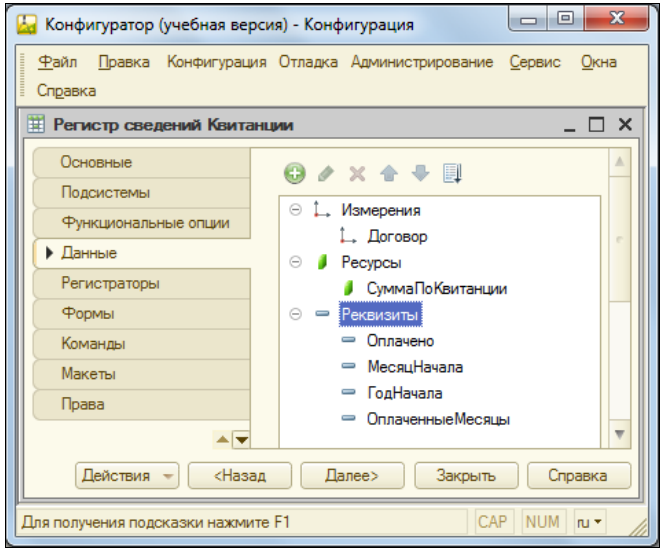


Рис. 3.27. Структура регистра сведений **Квитанции**

Теперь можно подвести небольшой промежуточный итог. Мы создали необходимые регистры (**Тарифы** и **Квитанции**), а также справочники — "ВидТарифа" и "Договоры".

Примечание

Свойства измерений, ресурсов и реквизитов редактируются при помощи уже знакомого окна под названием палитра свойств. В целом свойства регистров сведений аналогичны свойствам других объектов конфигурации. Однако есть и ряд свойств, характерных именно для регистров. Так, установка свойства **Ведущее** используется, если у рассматриваемого измерения тип данных представляет собой ссылку на объект конфигурации. В этом случае запись регистра сведений имеет смысл, только пока существует сам объект конфигурации. При удалении объекта записи по нему удаляются системой автоматически. Среди свойств измерений присутствует флажок **Запрет незаполненных значений**, который запрещает запись регистра с пустым значением измерения. Установка свойства **Индексировать** приводит к созданию отдельного индекса, что в ряде случаев увеличивает производительность работы с регистром.

Документ *КвитанцияПоДоговору*

Наша задача заключается в разработке документа, который будет осуществлять фиксирование выписанных квитанций в регистре **Квитанции**. Поэтому перейдем к разработке нового документа. В окне редактирования нового объекта конфигурации укажем в поле имя документа — "КвитанцияПоДоговору", после чего перейдем на вкладку **Данные** (рис. 3.28).

Определим реквизиты создаваемого документа:

- **Договор** (тип данных — **СправочникСсылка.Договоры**);
- **Учащийся** (тип данных — строка не более 100 символов);

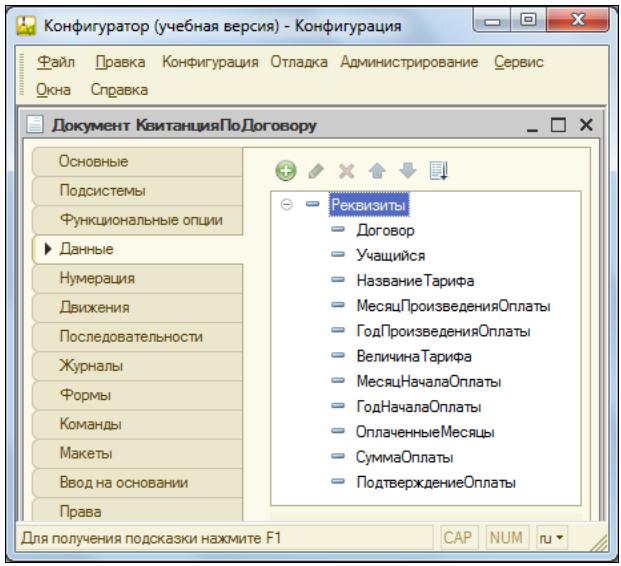


Рис. 3.28. Вкладка **Данные** окна редактирования документа "КвитанцияПоДоговору"

- **НазваниеТарифа** (тип данных — **СправочникСсылка.ВидТарифа**);
- **МесяцПроизведенияОплаты** (тип данных — **Число**);
- **ГодПроизведенияОплаты** (тип данных — **Число**);
- **ВеличинаТарифа** (тип данных — **Число**);
- **МесяцНачалаОплаты** (тип данных — **Число**);
- **ГодНачалаОплаты** (тип данных — **Число**);
- **ОплаченныеМесяцы** (тип данных — **Число**);
- **СуммаОплаты** (тип данных — **Число**);
- **ПодтверждениеОплаты** (тип данных — булевый).

Далее перейдем на вкладку **Движения** (рис. 3.29) и с помощью соответствующего флажка выберем регистр **Квитанции**.

После этого воспользуемся конструктором движений (кнопка **Конструктор движений**) и сформируем соответствие полей регистра сведений **Квитанции** и реквизитов документа "КвитанцияПоДоговору" (рис. 3.30). После этого с помощью кнопки **ОК** следует сформировать текст процедуры, реализующей обработку проведения документа "КвитанцияПоДоговору" (листинг 3.9).

Функционал документа еще не готов, но процесс его проведения подготовлен и это можно проверить в работе. Поэтому перейдем в режим **1С:Предприятие** и приступим к созданию нового экземпляра документа "КвитанцияПоДоговору". На рис. 3.31 приведен вариант частично заполненного документа "КвитанцияПоДоговору", чего вполне хватает для проверки выполненной разработки.

Далее следует созданный документ записать и провести (для этого предназначена кнопка **Провести и закрыть**). Чтобы убедиться в факте внесения информации в

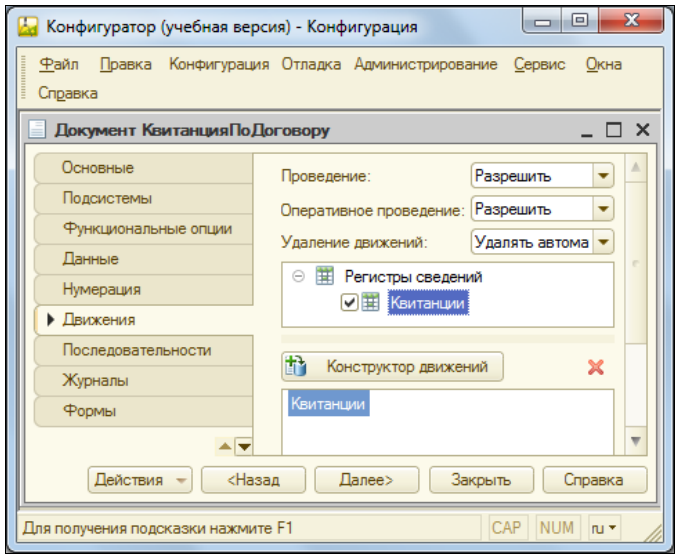


Рис. 3.29. Вкладка **Движения** окна редактирования документа "КвитанцияПоДоговору"

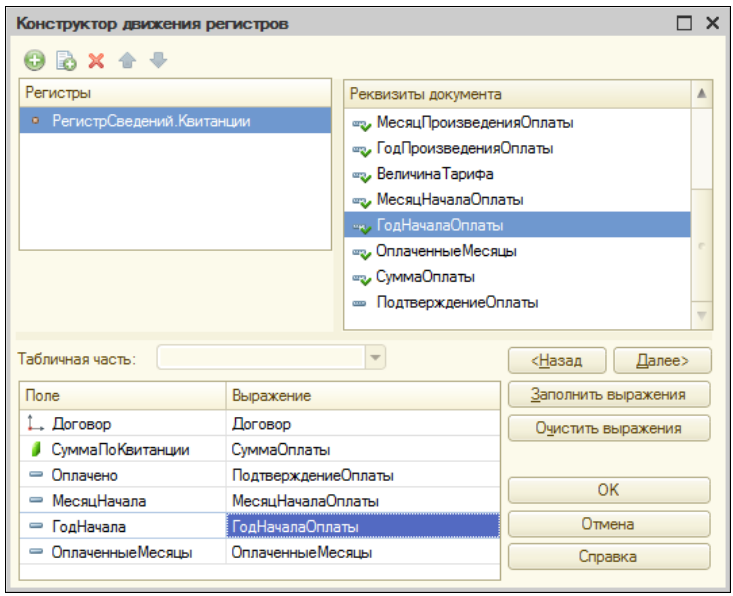


Рис. 3.30. Окно конструктора движений регистра сведений **Квитанции**

регистр сведений **Квитанции**, необходимо обратиться к главному меню приложения, где следует воспользоваться пунктом **Все функции**. В результате на экране откроется окно с соответствующим названием (рис. 3.32).

Выберем в данном окне пункт **Квитанции**, что приведет к отображению на экране формы списка данного регистра сведений (рис. 3.33). Это и есть результат проведения созданного документа.

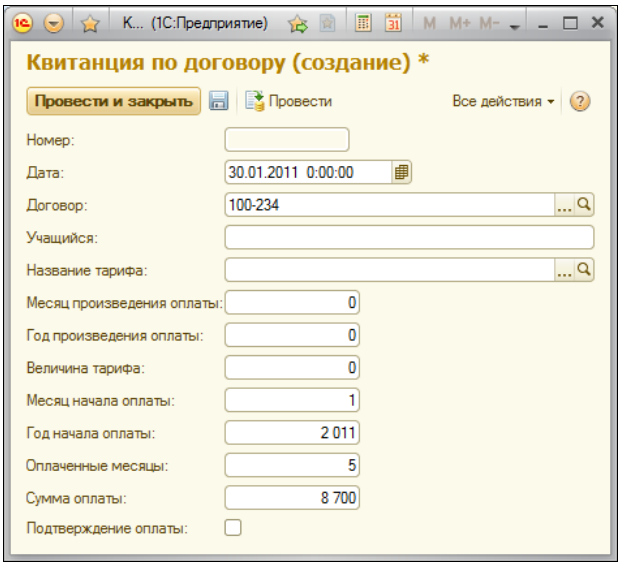


Рис. 3.31. Форма документа "КвитанцияПоДоговору"

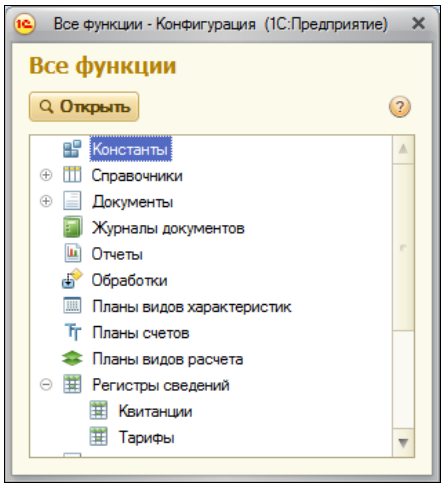


Рис. 3.32. Окно **Все функции**

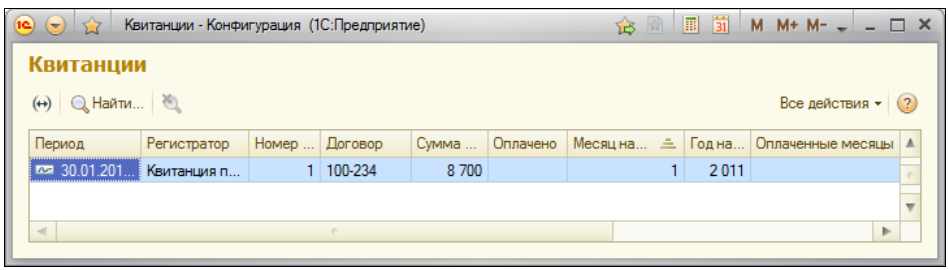


Рис. 3.33. Содержание регистра **Квитанции**

Листинг 3.9. Процедура обработки проведения документа КвитанцияПоДоговору

```
Процедура ОбработкаПроведения(Отказ, Режим)
  //{{__КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
  // Данный фрагмент построен конструктором.
  Движения.Квитанции.Записывать = Истина;
  Движение = Движения.Квитанции.Добавить();
  Движение.Период = Дата;
  Движение.Договор = Договор;
  Движение.СуммаПоКвитанции = СуммаОплаты;
  Движение.Оплачено = ПодтверждениеОплаты;
  Движение.МесяцНачала = МесяцНачалаОплаты;
  Движение.ГодНачала = ГодНачалаОплаты;
  Движение.ОплаченныеМесяцы = ОплаченныеМесяцы;
  //}}__КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
КонецПроцедуры
```

Итак, мы рассмотрели, каким образом фиксируются в регистре сведений выписанные квитанции. Если учащийся приносит оплаченную квитанцию, то следует выполнить соответствующую отметку в информационной базе. Так, сотрудник учебного центра должен открыть исходный документ "КвитанцияПоДоговору", установить флажок **ПодтверждениеОплаты** (рис. 3.34) и повторно провести данный документ.

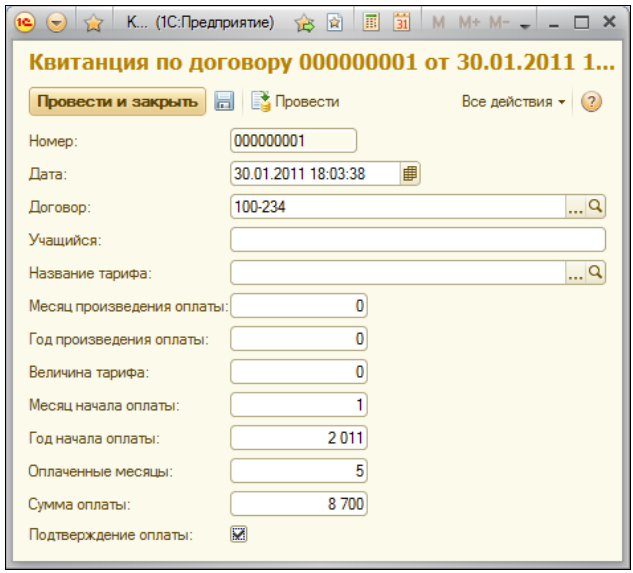


Рис. 3.34. Отметка оплаты в документе "КвитанцияПоДоговору"

Формирование печатной формы

В рассматриваемой разработке осталось еще несколько не рассмотренных моментов. Одним из них является формирование печатной формой квитанции. Для этого нам необходимо разработать макет табличного документа, необходимые программные процедуры и разместить на форме списка документа "КвитанцияПоДоговору" командную кнопку для формирования печатной формы.

Начнем с разработки макета, для чего перейдем на вкладку **Макеты** и воспользуемся кнопкой **Добавить**. В результате мы перейдем к разработке необходимого макета с помощью предназначенного для этого конструктора.

В первом открывшемся окне конструктора (рис. 3.35) следует с помощью переключателя установить тип макета (сейчас нам необходим **Табличный документ**). После этого на экране откроется редактор для разработки макета. На рис. 3.36 представлен макет квитанции, который нам необходимо разработать. В макете разместим две области:

- **Извещение;**
- **Кассир.**

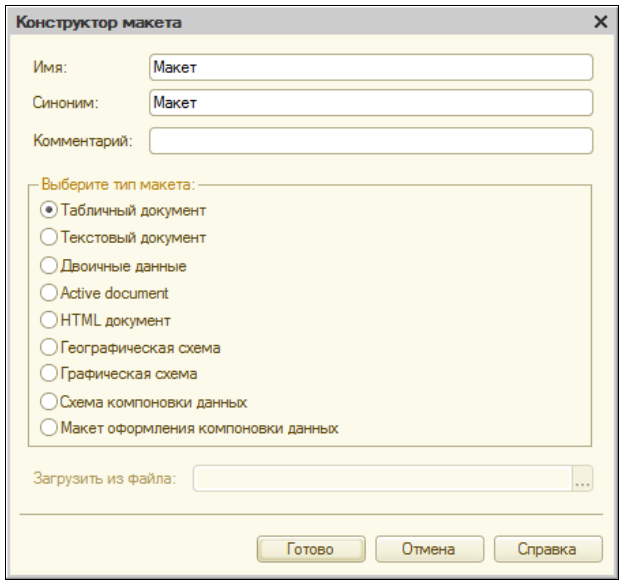


Рис. 3.35. Окно конструктора макета

Наряду с чисто текстовой информацией в макете следует разместить несколько полей, для которых в палитре свойств необходимо в разделе **Заполнение** сделать установку **Параметр**. На рис. 3.36 ячейки, для которых установлено значение **Параметр**, выделены угловыми скобками.

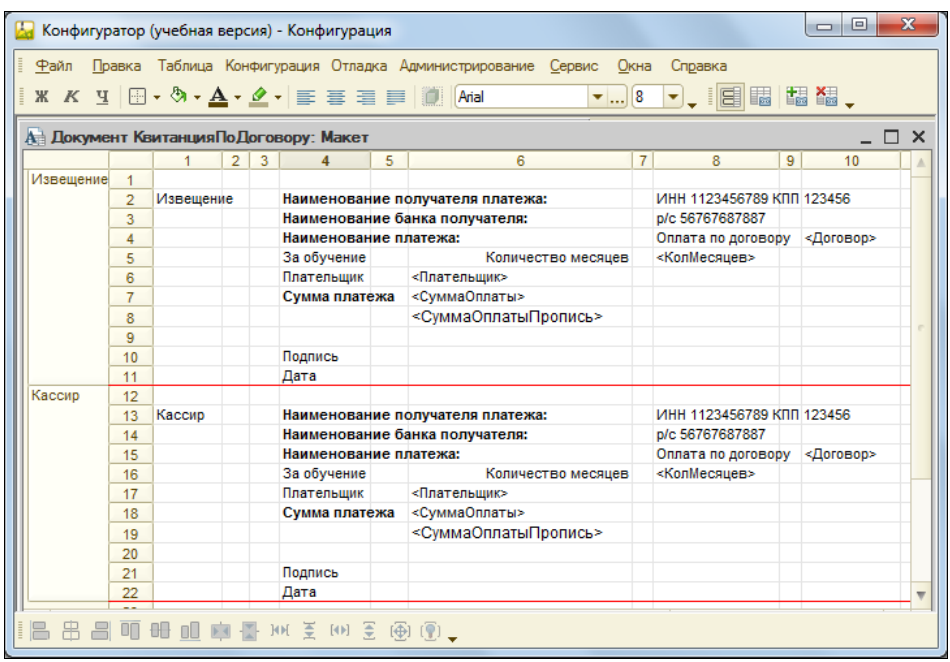


Рис. 3.36. Макет табличного документа для квитанции

Фактически в этом случае в макете задаются имена переменных, которые будут видны из программной процедуры.

Теперь перейдем к разработке программных процедур. Для этого активизируем вкладку **Формы** и создадим форму документа (рис. 3.37). Здесь нам необходимо иметь новую команду (для печати), поэтому перейдем на дополнительную вкладку **Команды** и с помощью кнопки пиктографического меню создадим новую команду **Печать**.

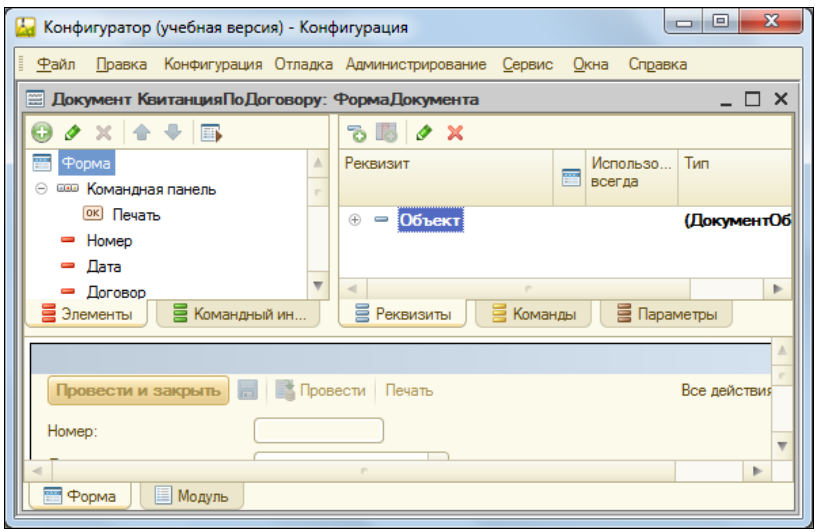


Рис. 3.37. Форма документа "КвитанцияПоДоговору"

В палитре свойств созданной команды обратимся к пункту **Действие** и оформим процедуру обработки команды в варианте, представленном в листинге 3.10. В процессе выполнения данной процедуры производится последующий серверный вызов процедуры `ПечатьКвитанции` (листинг 3.11).

Для завершения разработки создадим кнопку на командной панели, необходимую для выполнения команды. Для этого достаточно мышью перетащить пиктограмму команды **Печать** на командную панель (рис. 3.38).

Листинг 3.10. Процедура обработки команды Печать

```
Процедура Печать(Команда)
    ТабДок = Новый ТабличныйДокумент;
    ПечатьКвитанции(ТабДок);
    ТабДок.ОтображатьСетку = Ложь;
    ТабДок.Защита = Ложь;
    ТабДок.ТолькоПросмотр = Ложь;
    ТабДок.ОтображатьЗаголовки = Ложь;
    ТабДок.Показать();
КонецПроцедуры
```

Листинг 3.11. Серверная процедура ПечатьКвитанции

```
&НаСервере
Процедура ПечатьКвитанции(ТабДок)
  ТабДок.Очистить();
  Макет = Документы.КвитанцияПоДоговору.ПолучитьМакет("Макет");
  Извещение = Макет.ПолучитьОбласть("Извещение");
  Извещение.Параметры.КолМесяцев=Объект.ОплаченныеМесяцы;
  Извещение.Параметры.Договор=Объект.Договор;
  Извещение.Параметры.Плательщик=Объект.Договор.ФИОПлательщика;
  Извещение.Параметры.СуммаОплаты=Объект.СуммаОплаты;
  ПараметрыПредметаИсчисления = "рубль,рубля,рублей";
  ФорматнаяСтрока = "Л=ru_RU";
  ВыводимоеЧисло = Объект.СуммаОплаты;
  Извещение.Параметры.СуммаОплатыПропись =
     ЧислоПрописью(ВыводимоеЧисло,
     ФорматнаяСтрока, ПараметрыПредметаИсчисления);
  ТабДок.Вывести(Извещение);
  Кассир = Макет.ПолучитьОбласть("Кассир");
  Кассир.Параметры.КолМесяцев=Объект.ОплаченныеМесяцы;
  Кассир.Параметры.Договор=Объект.Договор;
  Кассир.Параметры.Плательщик=Объект.Договор.ФИОПлательщика;
  Кассир.Параметры.СуммаОплаты=Объект.СуммаОплаты;
  ПараметрыПредметаИсчисления = "рубль,рубля,рублей";
  ФорматнаяСтрока = "Л=ru_RU";
  ВыводимоеЧисло = Объект.СуммаОплаты;
  Кассир.Параметры.СуммаОплатыПропись = ЧислоПрописью(ВыводимоеЧисло,
        ФорматнаяСтрока, ПараметрыПредметаИсчисления);
  ТабДок.Вывести(Кассир);
КонецПроцедуры
```

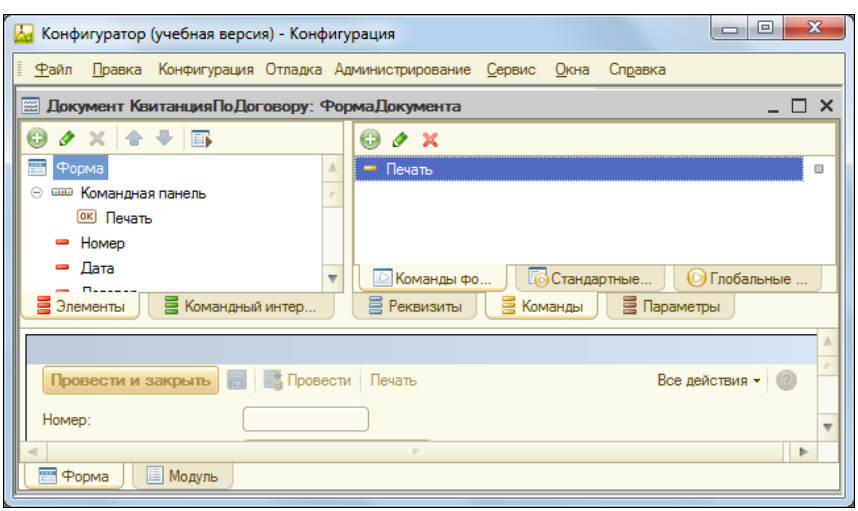


Рис. 3.38. Размещение командной кнопки на форме

Теперь все готово, осталось перейти в режим **1С:Предприятие**, обратиться к документу "КвитанцияПоДоговору", после чего щелкнуть по кнопке **Печать**. В результате на экране будет отображена печатная форма документа (рис. 3.39).

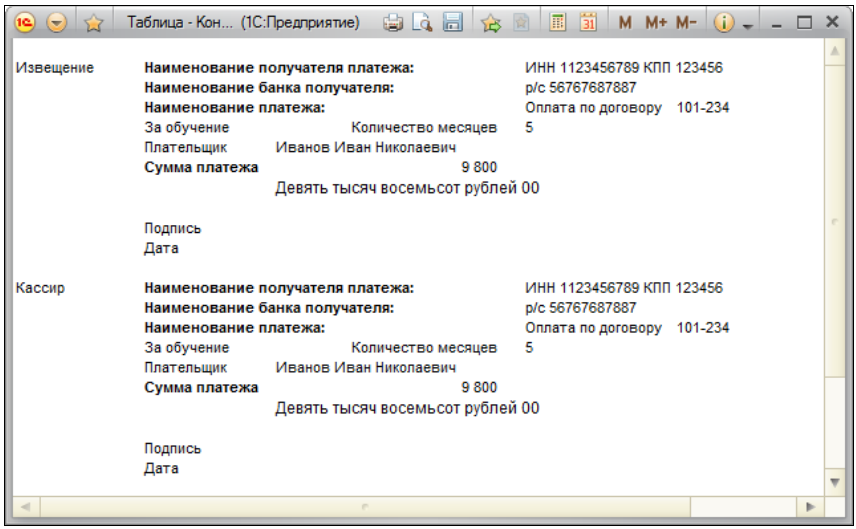


Рис. 3.39. Печатная форма квитанции по оплате

Доработка документа по фиксированию квитанций

Если вернуться к разработанному документу "КвитанцияПоДоговору", то мы увидим недоработки функционала:

- при выборе договора не заполняется поле **Учащийся**;
- при выборе договора не заполняется поле **НазваниеТарифа**, зафиксированное в договоре;
- не заполняется поле **ВеличинаТарифа**.

Эти недоработки в данном разделе мы устраним.

Для начала реализуем внесение фамилии учащегося в поле **Учащийся** после выбора договора. Для этого необходимо в редакторе форм обратиться к палитре свойств поля **Договор**. Здесь мы должны оформить процедуру обработки события `ПриИзменении`. Необходимый текст представлен в листинге 3.12. В приведенной процедуре используется обращение к ряду дополнительных функций:

- `ПолучитьУчащегося` — для внесения фамилии учащегося по выбранному договору (листинг 3.13);
- `ПолучитьВидТарифаУчащегося` — для внесения названия тарифа по выбранному договору (листинг 3.14);
- `ЗаполнитьМесяцГод` — для внесения в соответствующие поля текущих месяца и года по выбранному договору (листинг 3.15);

- ПолучитьЗначениеТарифа — для внесения величины тарифа по выбранному договору (листинг 3.16).

Листинг 3.12. Процедура обработки события ПриИзменении в поле Договор

```
&НаКлиенте
Процедура ДоговорПриИзменении(Элемент)
Объект.Учащийся=ПолучитьУчащегося(Объект.Договор);
Объект.НазваниеТарифа=ПолучитьВидТарифаУчащегося(Объект.Договор);
ЗаполнитьМесяцГод();
Объект.Тариф=ПолучитьЗначениеТарифа(Объект.Договор,
 Объект.МесяцПроизведенияОплаты,Объект.ГодПроизведенияОплаты);
КонецПроцедуры
```

Листинг 3.13. Серверная функция ПолучитьУчащегося

```
&НаСервереБезКонтекста
Функция ПолучитьУчащегося(ДоговорУчащегося)
Возврат ДоговорУчащегося.ФИОучащегося;
КонецФункции
```

Листинг 3.14. Серверная функция ПолучитьВидТарифаУчащегося

```
&НаСервереБезКонтекста
Функция ПолучитьВидТарифаУчащегося(ДоговорУчащегося)
Возврат ДоговорУчащегося.ВидТарифа;
КонецФункции
```

Листинг 3.15. Серверная процедура ЗаполнитьМесяцГод

```
&НаСервере
Процедура ЗаполнитьМесяцГод()
Объект.МесяцПроизведенияОплаты = Месяц(ТекущаяДата());
Объект.ГодПроизведенияОплаты = Год(ТекущаяДата());
КонецПроцедуры
```

Листинг 3.16. Серверная функция ПолучитьЗначениеТарифа

```
&НаСервереБезКонтекста
Функция ПолучитьЗначениеТарифа(ДоговорУчащегося,Месяц,Год)
Запрос = Новый Запрос;
Запрос.Текст = "ВЫБРАТЬ *
| ИЗ РегистрСведений.Тарифы
| ГДЕ ВидТарифа=&НазваниеТарифа И
| НомерМесяца =&Месяц И
| Год = &Год";
```

```
Запрос.УстановитьПараметр("НазваниеТарифа",
    ДоговорУчащегося.ВидТарифа);
  Запрос.УстановитьПараметр("Месяц",Месяц);
  Запрос.УстановитьПараметр("Год",Год);
  Результат = Запрос.Выполнить().Выбрать();
  Если Результат.Следующий() > 0 Тогда
   Возврат Результат.Стоимость;
  КонецЕсли;
КонецФункции
```

Для автоматического вычисления суммы после внесения количества оплачиваемых месяцев следует разработать еще одну программную процедуру. Для этого необходимо в редакторе форм обратиться к палитре свойств поля `КоличествоОплачиваемыхМесяцев`. Здесь мы должны оформить процедуру обработки события `ПриИзменении`. Необходимый текст представлен в листинге 3.17.

Листинг 3.17. Процедура обработки в поле КоличествоОплачиваемыхМесяцев

```
&НаКлиенте
Процедура КоличествоОплачиваемыхМесяцевПриИзменении(Элемент)
 Объект.СуммаОплаты=Объект.ОплаченныеМесяцы*Объект.ВеличинаТарифа;
КонецПроцедуры
```

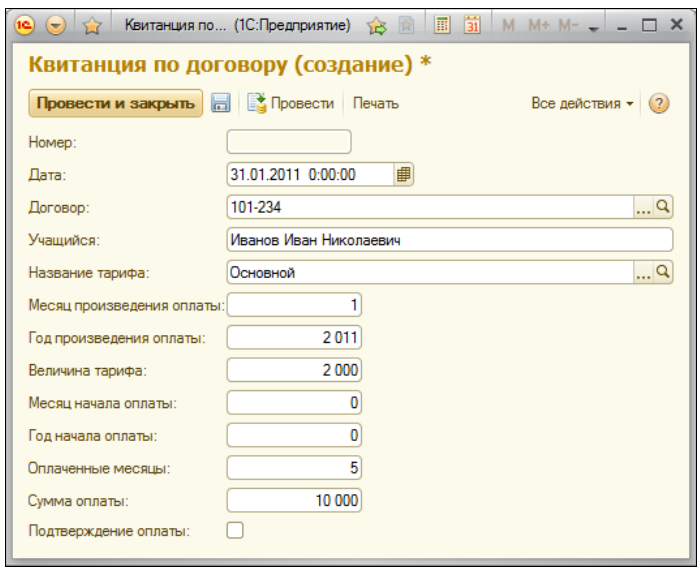


Рис. 3.40. Форма документа по учету квитанций

Таким образом, теперь необходимый "функционал" готов и можно в режиме **1С:Предприятие** провести несколько документов по оплате и посмотреть на результат в регистре сведений **Квитанции** (рис. 3.40).

Формирование документа Microsoft Word

В настоящее время интеграция между различными приложениями в практической работе используется достаточно часто. Кроме информационной системы 1С:Предприятие 8.2 в офисной работе очень популярны программы Microsoft Word и Microsoft Excel. Создание большинства офисных документов выполняется именно в этих приложениях.

Здесь мы рассмотрим практическую задачу, связанную с автоматическим созданием документа Microsoft Word непосредственно из процедуры, выполняемой в среде 1С:Предприятие 8.2.

Будем считать, что мы располагаем файлом, созданным в приложении Microsoft Word, который содержит бланк квитанции об оплате (рис. 3.41). По традиционной технологии этот бланк распечатывается на принтере, а затем заполняется вручную сотрудниками офиса. Наша задача заключается в разработке технологии автоматизированного заполнения полей бланка из процедуры, выполняемой в системе 1С:Предприятие. Мы уже разработали систему регистрации квитанций об оплате, а теперь необходимо обеспечить их распечатку в определенном формате.

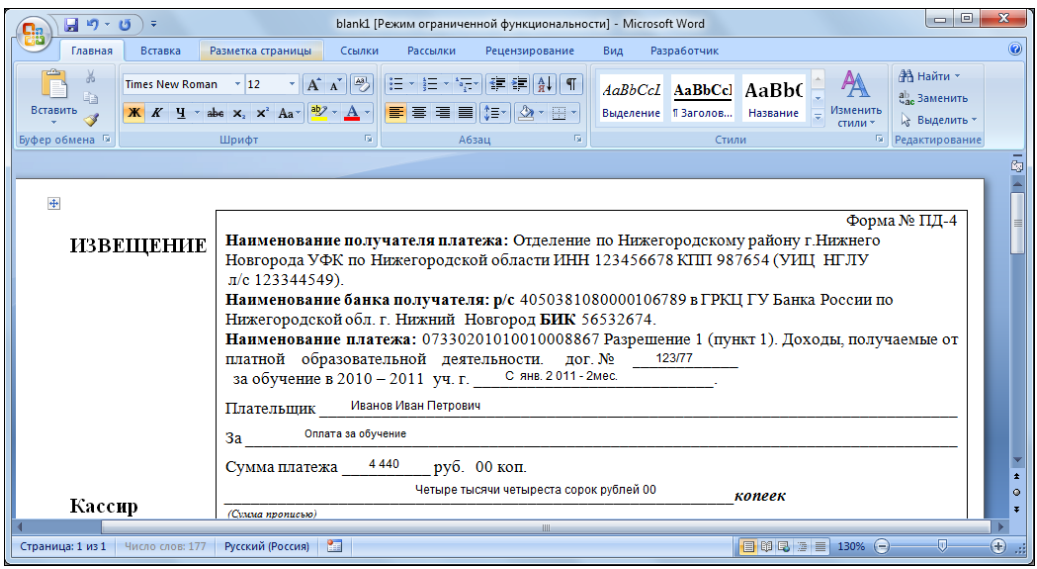


Рис. 3.41. Файл Microsoft Word, содержащий бланк квитанции

Внесение дополнений в бланк квитанции

Итак, для данной разработки потребуется файл, содержащий квитанцию (см. рис. 3.41), — это документ, созданный в приложении Microsoft Word 2007. Фактически это обыкновенный документ с необходимым форматированием.

Теперь мы должны данный документ дополнить элементами управления, в которые информация будет передаваться непосредственно из приложения 1С:Предприятие.

Часть действий для этого будет производиться в приложении Microsoft Word, а часть — в программе 1С:Предприятие.

Рассмотрим сначала необходимые шаги в программе Microsoft Word. Так, в приложении Microsoft Word 2007 нам потребуется работать с элементами управления, поэтому необходимо *на ленте* открыть вкладку **Разработчик**.

С помощью кнопки **Режим конструктора** можно перейти в конструктор, что позволяет размещать в документе необходимые интерфейсные элементы (аналогичные тем, с которыми мы уже сталкивались при работе по конструированию форм в режиме конфигуратора).

В документе, содержащем бланк квитанции, нам потребуется создать несколько элементов управления типа *поле* (более правильное название — *текстовое поле*). Для этого в группе **Элементы управления** воспользуемся кнопкой **Инструменты из предыдущих версий**. И здесь в разделе **Элементы ActiveX** щелкнем по пиктограмме **Поле**. Функциональное назначение элементов *поле* аналогично ранее встречавшимся в системе 1С:Предприятие элементам *поле ввода*.

После выбора элемента *поле* с помощью мыши следует поместить его в нашем документе (рис. 3.42).

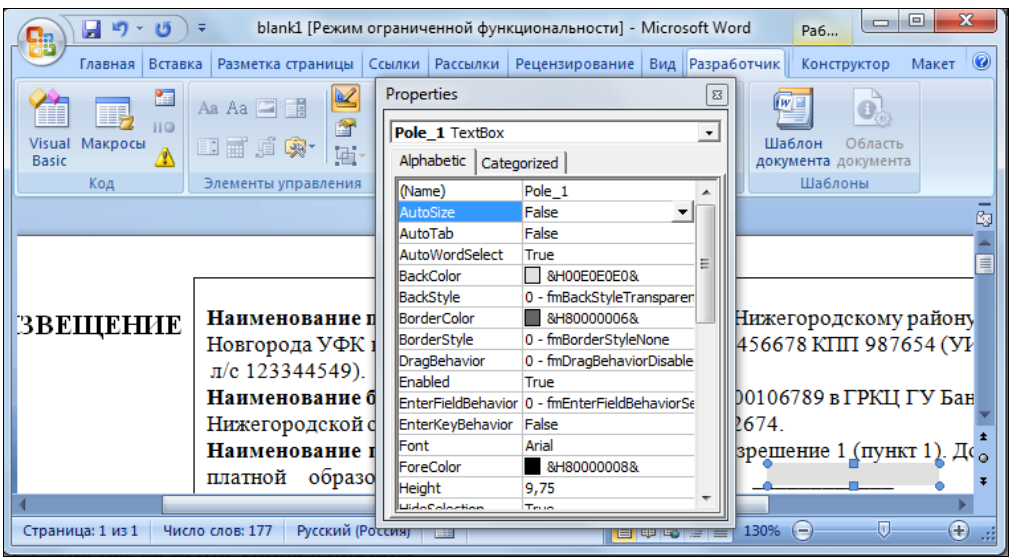


Рис. 3.42. Окно свойств элемента управления *поле*

Чтобы внешний вид исходного бланка не нарушался, необходимо обеспечить его размещение перед текстом. Для этого следует щелкнуть правой кнопкой мыши по созданному в документе элементу управления и в открывшемся контекстном меню выбрать пункт **Формат объекта**. В появившемся диалоговом окне на вкладке **Положение** следует установить формат **перед текстом** (рис. 3.43).

В результате проделанных действий внешний вид бланка не изменится, но в нем появится дополнительное поле для внесения информации. С помощью маркеров,

окружающих элемент управления (см. рис. 3.42), мы можем выбрать его размер и месторасположение в документе.

На ленте кнопка **Свойства** позволяет открыть окно свойств активного (выделенного с помощью мыши) элемента управления, которое отражается в центральной части рис. 3.42. Установим для свойства **Name** этого элемента значение **Pole_1**. Кроме того, с помощью свойства **Font** следует подобрать желаемый шрифт для отображения текста в данном поле. На рис. 3.42 текстовое поле выделено серым оттенком, что обеспечивается установкой цвета в свойстве **BackColor**. Однако реально этого делать не требуется — в данном случае цель заключалась просто в придании большей наглядности описываемым действиям.

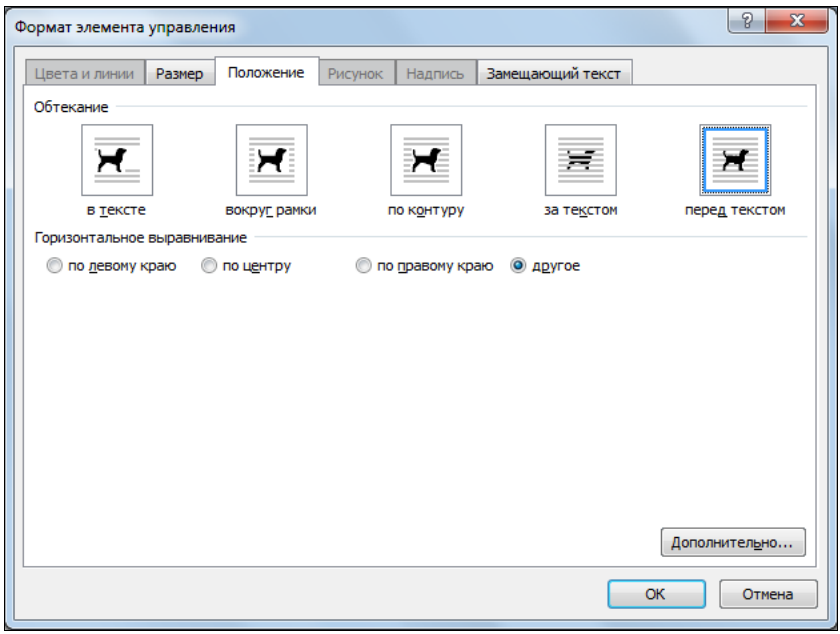


Рис. 3.43. Окно для установки формата объекта *поле*

Аналогичным образом создадим еще семь элементов (рис. 3.44). Для правильной работы дальнейшей программной процедуры необходимо установить у них значения свойства **Name**, соответственно (сверху вниз), следующими: **Pole_2**, **Pole_3**, **Pole_4**, **Pole_5**, **Pole_6**, **Pole_7**, **Pole_8**.

После этого нужно выйти из режима конструктора ("отжать" соответствующую кнопку на ленте) и сохранить документ на диске, полное имя для которого следует выбрать *C:\Бланки\blank1.docx*. Дело в том, что далее в программной процедуре потребуется указание полного пути месторасположения файла.

> ***Примечание***
>
> Можно изменить имя и месторасположение файла, но тогда нужно и скорректировать соответствующую строку в программной процедуре.

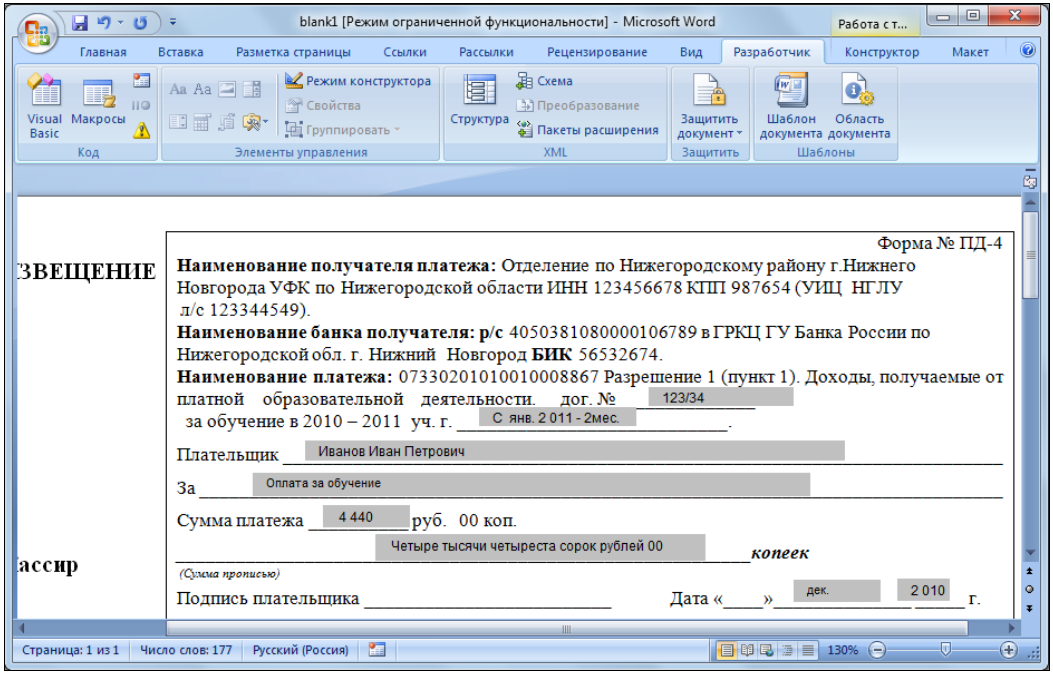


Рис. 3.44. Элементы управления,
расположенные в документе приложения Microsoft Word

Итак, наша конструкторская работа в приложении Microsoft Word завершена — документ для выписки квитанции готов. Рассматриваемая в следующем разделе программная процедура на встроенном языке 1С будет вносить в квитанцию необходимую информацию.

Разработка процедуры заполнения бланка

Теперь вернемся из приложения Microsoft Word в режим конфигуратора системы 1С:Предприятие. Откроем снова конфигурацию, созданную в данном разделе. Здесь нам потребуется обратиться к форме документа "КвитанцияПоДоговору" и разместить в ней новую команду **ПередачаBWord** (рис. 3.45).

После этого с помощью щелчка по изображению лупы (в окне палитры свойств) напротив надписи **Действие** (см. рис. 3.45) перейдем в редактор программного кода и оформим клиентскую процедуру обработки команды так, как показано в листинге 3.18. Основные вычисления, как и ранее, производятся на сервере (листинг 3.19).

Листинг 3.18. Процедура обработки команды ПередачаBWord

```
&НаКлиенте
Процедура ПередачаBWord(Команда)
ПередачаBWordПечать();
КонецПроцедуры
```

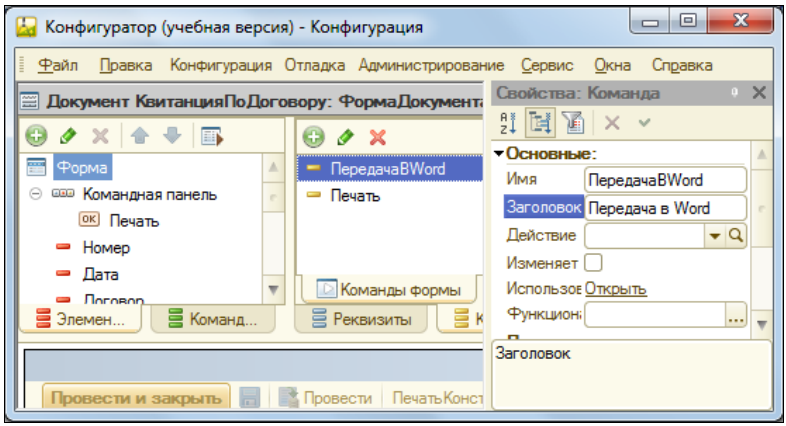


Рис. 3.45. Создание команды **ПередачаBWord** в форме документа "КвитанцияПоДоговору"

Листинг 3.19. Серверная процедура ПередачаBWordПечать

```
&НаСервере
Процедура ПередачаBWordПечать ()
  МассивМесяцев = Новый Массив();
  МассивМесяцев.Вставить(1,"янв.");
  МассивМесяцев.Вставить(2,"февр.");
  МассивМесяцев.Вставить(3,"март");
  МассивМесяцев.Вставить(4,"апр.");
  МассивМесяцев.Вставить(5,"май");
  МассивМесяцев.Вставить(6,"июнь");
  МассивМесяцев.Вставить(7,"июль");
  МассивМесяцев.Вставить(8,"авг.");
  МассивМесяцев.Вставить(9,"сент.");
  МассивМесяцев.Вставить(10,"окт.");
  МассивМесяцев.Вставить(11,"ноябр.");
  МассивМесяцев.Вставить(12,"дек.");
  МесяцНачала = Объект.МесяцНачалаОплаты;
  ГодНачала = Объект.ГодНачалаОплаты;
  Колич = Объект.ОплаченныеМесяцы;
  appWD = Новый COMОбъект("Word.Application");
  appWD.Visible = Истина;
  appWD.Documents.Open("C:\Бланки/blank1.doc");
  appWD.ActiveDocument.Pole_1.Text =
              Строка(Объект.Договор);
  appWD.ActiveDocument.Pole_2.Text = "С  " +
              МассивМесяцев[МесяцНачала] +  " " +
              Строка(ГодНачала) + " — " +  Строка(Колич)+ "мес.";
  appWD.ActiveDocument.Pole_3.Text = Объект.Учащийся;
  appWD.ActiveDocument.Pole_4 = "Оплата за обучение";
  appWD.ActiveDocument.Pole_5 = Строка(Объект.СуммаОплаты);
```

```
  ПараметрыПредметаИсчисления = "рубль,рубля,рублей";
  ФорматнаяСтрока = "Л=ru_RU";
  ВыводимоеЧисло = Объект.СуммаОплаты;
  appWD.ActiveDocument.Pole_6 = ЧислоПрописью(ВыводимоеЧисло,
        ФорматнаяСтрока, ПараметрыПредметаИсчисления);
  appWD.ActiveDocument.Pole_7 =
        МассивМесяцев[Объект.МесяцПроизведенияОплаты];
  appWD.ActiveDocument.Pole_8 =
        Строка(Объект.ГодПроизведенияОплаты);
КонецПроцедуры
```

Процедура `ПередачаВWordПечать` включает несколько моментов, которые отличают ее от ранее рассмотренных, поэтому мы познакомимся с ее программными конструкциями подробнее. В начале процедуры создается массив элементов и переменной `МассивМесяцев` присваивается ссылка на него. Вообще, *массив* является объектом, предназначенным для работы с временными наборами данных в течение текущего сеанса работы пользователя. Чаще всего использование массивов требуется для некоторых промежуточных действий. В данном случае мы создаем набор соответствий номеров месяцев и их сокращенных названий. Для этого можно было поступить по-другому — создать стационарный объект, сохраняемый между различными сеансами на энергонезависимом носителе (диске). Однако выбранный путь связан с целью знакомства с массивами на конкретном примере.

В общем, массив можно считать набором значений произвольного типа. При этом каждый элемент массива имеет свой индекс — по значению этого индекса к элементу массива можно обратиться (записать или извлечь информацию). Например, `Массив1[5]` — это значение пятого элемента массива с именем `Массив1`.

Для создания массива в процедуре используется оператор `Новый` в следующей редакции:

```
МассивМесяцев = Новый Массив();
```

Здесь `МассивМесяцев` — имя переменной, которая будет ассоциироваться с создаваемым объектом типа `Массив`. В тексте процедуры после создания массива мы добавляем в него значения строкового типа с помощью метода `Вставить`, у которого два параметра. Первый параметр определяет индекс элемента массива, а второй — значение элемента массива.

Таким образом, в нашей процедуре массив заполняется сокращенными названиями месяцев. В дальнейшем это потребуется при выводе дат в поля квитанции, а именно: в элемент управления `Pole_2` заносится название месяца начала оплаты и количество оплачиваемых месяцев. Кроме того, элемент `Pole_7` также заполняется строковым названием месяца, в течение которого учащийся собирается произвести оплату.

В процедуре, представленной в листинге 3.19, использована технология OLE (связывание и внедрение объектов). Она предоставляет возможность в одних приложениях работать с объектами, созданными в других приложениях.

В этой технологии существует несколько ключевых понятий. Так, OLE-объектом называется объект, созданный в приложении, отличном от текущего, у которого сохранена связь со своим "родным" приложением. В офисной работе очень популярны приложения Microsoft Word и Microsoft Excel, и, соответственно, используемые документы Microsoft Word и рабочие книги Microsoft Excel могут быть OLE-объектами. Сервером приложения OLE считается приложение, создающее объекты, которые можно поместить в документ-контейнер, создаваемый в другом приложении.

В приведенной в листинге 3.19 процедуре приложение 1С:Предприятие 8.2 используется как контейнер, а приложение Microsoft Word — как сервер. При таком взаимодействии Microsoft Word предоставляет свои объекты другим приложениям, чем мы и воспользовались.

В строке

```
appWD = Новый COMОбъект("Word.Application");
```

мы формируем в переменной `appWD` ссылку на приложение Microsoft Word. При этом приложение загружается в оперативную память компьютера, но на экране не отображается. Для его показа применяется следующая строка:

```
appWD.Visible = Истина;
```

Для дальнейших действий требуется знание месторасположения в компьютере документа Microsoft Word с квитанцией. Как уже говорилось, созданный бланк квитанции мы расположили в папке *Бланки* на диске C:. В качестве имени файла выбрано — *blank1.docx*. Далее мы пользуемся методом `Open` для открытия необходимого документа:

```
appWD.Documents.Open ("C:\Бланки/blank1.docx")
```

После этого можно начать работу с открытым документом, используя его внутренние объекты, методы и свойства. В последующем программном коде процедуры в объекты документа Microsoft Word (в элементы управления, которые мы сами разместили) переносится информация с формы обработки.

После действий в конфигураторе системы 1С:Предприятие (которые свелись к размещению кнопки на форме и созданию программной процедуры, обслуживающей нажатие кнопки) можно перейти в режим **1С:Предприятие**, открыть форму документа "КвитанцияПоДоговору" и щелкнуть по кнопке передачи информации в Microsoft Word. В результате мы получим заполненную квитанцию в этом приложении.

Заполнение полного бланка с квитанцией

Мы рассмотрели заполнение бланка, содержащего только один фрагмент квитанции. Реально требуется заполнение двух областей бланка. Для этого в уже знакомой форме документа "КвитанцияПоДоговору" создадим соответствующую команду **ПередачаBWordПолногоБланка** (рис. 3.46).

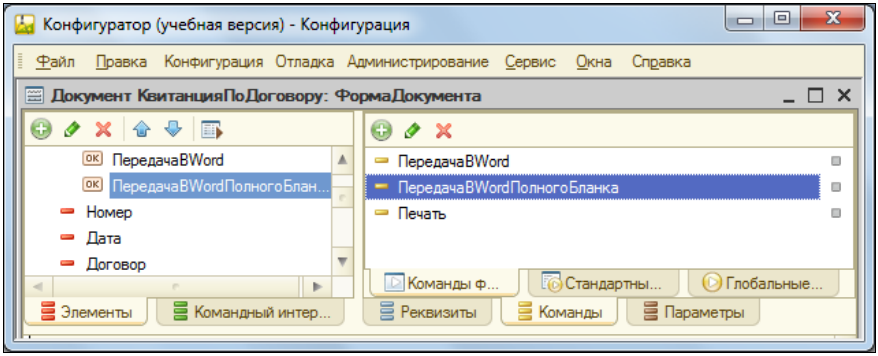


Рис. 3.46. Добавление команды для печати двух квитанций

Листинг 3.20. Процедура обработки команды ПередачаBWordПолногоБланка

```
&НаКлиенте
Процедура ПередачаBWordПолногоБланка(Команда)
 ПередачаBWordПечатьПолногоБланка();
КонецПроцедуры
```

Для вычислительных действий разработана соответствующая серверная процедура (листинг 3.21). Она предполагает некоторую модификацию созданного ранее бланка, а именно дублирование текста и создание наряду с восемью текстовыми полями для первой квитанции еще восьми аналогичных элементов управления для второго бланка. Для имен этих новых элементов в процедуре предполагаются следующие имена (порядок элементов сверху вниз, как и в первой квитанции): `Pole_1_2`, `Pole_2_2`, `Pole_3_2`, `Pole_4_2`, `Pole_5_2`, `Pole_6_2`, `Pole_7_2`, `Pole_8_2`. Созданный бланк следует сохранить на диске C: в папке *Бланки* под именем *blank2.docx*. Заполненный программной процедурой документ Word показан на рис. 3.47.

Листинг 3.21. Процедура для заполнения бланка с двумя квитанциями

```
&НаСервере
Процедура ПередачаBWordПечать2Квитанции ()
  МассивМесяцев = Новый Массив();
  МассивМесяцев.Вставить(1,"янв.");
  МассивМесяцев.Вставить(2,"февр.");
  МассивМесяцев.Вставить(3,"март");
  МассивМесяцев.Вставить(4,"апр.");
  МассивМесяцев.Вставить(5,"май");
  МассивМесяцев.Вставить(6,"июнь");
  МассивМесяцев.Вставить(7,"июль");
  МассивМесяцев.Вставить(8,"авг.");
  МассивМесяцев.Вставить(9,"сент.");
  МассивМесяцев.Вставить(10,"окт.");
  МассивМесяцев.Вставить(11,"ноябр.");
```

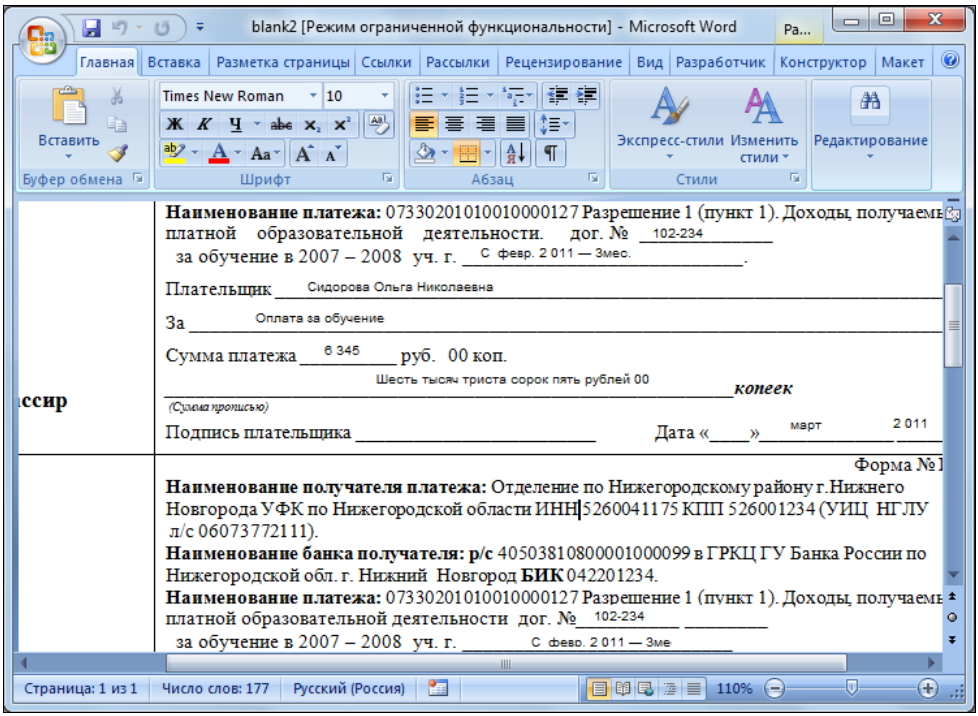


Рис. 3.47. Заполненный бланк с двумя квитанциями

```
  МассивМесяцев.Вставить(12,"дек.");
  МесяцНачала = Объект.МесяцНачалаОплаты;
  ГодНачала = Объект.ГодНачалаОплаты;
  Колич = Объект. ОплаченныеМесяцы;
  appWD = Новый COMОбъект("Word.Application");
  appWD.Visible = Истина;
  appWD.Documents.Open("C:\Бланки/blank2.doc");
  appWD.ActiveDocument.Pole_1.Text =
          Строка(Объект.Договор);
  appWD.ActiveDocument.Pole_1_2.Text =
          Строка(Объект.Договор);
appWD.ActiveDocument.Pole_2.Text = "С  " +
        МассивМесяцев[МесяцНачала] +  " " +
        Строка(ГодНачала) + " — " +  Строка(Колич)+ "мес.";
  appWD.ActiveDocument.Pole_2_2.Text = "С  " +
        МассивМесяцев[МесяцНачала] +  " " +
        Строка(ГодНачала) + " — " +  Строка(Колич)+ "мес.";
  appWD.ActiveDocument.Pole_3.Text = Объект.Учащийся;
  appWD.ActiveDocument.Pole_3_2.Text = Объект.Учащийся;
  appWD.ActiveDocument.Pole_4 = "Оплата за обучение";
  appWD.ActiveDocument.Pole_4_2 = "Оплата за обучение";
  appWD.ActiveDocument.Pole_5 = Строка(Объект.СуммаОплаты);
  ПараметрыПредметаИсчисления = "рубль,рубля,рублей";
```

```
  appWD.ActiveDocument.Pole_5_2 = Строка(Объект.СуммаОплаты);
  ПараметрыПредметаИсчисления = "рубль,рубля,рублей";
  ФорматнаяСтрока = "Л=ru_RU";
  ВыводимоеЧисло = Объект.СуммаОплаты;
  appWD.ActiveDocument.Pole_6 = ЧислоПрописью(ВыводимоеЧисло,
        ФорматнаяСтрока, ПараметрыПредметаИсчисления);
  appWD.ActiveDocument.Pole_6_2 = ЧислоПрописью(ВыводимоеЧисло,
        ФорматнаяСтрока, ПараметрыПредметаИсчисления);
  appWD.ActiveDocument.Pole_7 =
        МассивМесяцев[Объект.МесяцПроизведенияОплаты];
  appWD.ActiveDocument.Pole_7_2 =
        МассивМесяцев[Объект.МесяцПроизведенияОплаты];
  appWD.ActiveDocument.Pole_8 =
        Строка(Объект.ГодПроизведенияОплаты);
  appWD.ActiveDocument.Pole_8_2 =
        Строка(Объект.ГодПроизведенияОплаты);
КонецПроцедуры
```

Итоги

В этой главе мы рассмотрели технологию работы с важным объектом конфигурации — регистром сведений. Приведены примеры, касающиеся практического использования как периодических, так и непериодических регистров сведений. Разработки двух конфигураций продемонстрировали их роль в построении информационных баз данных по учету материальных и денежных средств. В рассмотренных примерах используется как ручной ввод информации в регистры сведений, так внесение данных в результате проведения документов.

ГЛАВА 4

Запросы

Если в целом оценить примеры, приведенные в предыдущих главах, то можно сказать, что мы работали в основном с первичной информацией, сосредоточенной в разнообразных объектах — справочниках, регистрах и документах. И если бы нас интересовали данные только в такой исходной форме, то больше в принципе ничего и не нужно. Так, всегда можно открыть необходимый справочник, просмотреть содержимое регистра или документ из списка документов.

Однако для управляющего звена организации представляет интерес информация в существенно обработанном виде (отбор по параметрам, итоговые и сводные данные). Например, в *главе 1* мы создали справочник "Специалисты", в котором присутствует табличная часть, отражающая различные места работы сотрудников (рассматривается ситуация, что у определенного специалиста может быть не одно место работы). И, скажем, может возникнуть необходимость получения перечня названий всех сторонних организаций, где наши сотрудники дополнительно работают.

Конечно, можно было бы при разработке данного справочника пойти по пути предварительного создания вспомогательного (подстановочного) справочника со списком организаций. Да, действительно, именно так и стоило бы поступить, если бы на этапе разработки планировалось, что подобный справочник будет нам необходим в дальнейшем. В данной же ситуации (когда конфигурация разработана и информационная база заполнена данными) для получения отчета требуется проанализировать различные организации исходя из информации, располагающейся в табличных частях элементов справочника "Специалисты". Один из вариантов интересующего нас отчета приведен в табл. 4.1.

Таблица 4.1. *Список организаций*

Название организации
АО "Связьинвест"
НМСИ
АО "Меда"
МТКИ

Технически для составления подобного отчета мы должны перебрать все элементы справочника "Специалисты" и изучить их табличные части. Понятно, что без автоматизации данную работу выполнять не очень интересно.

Другая аналогичная задача может возникнуть при работе со справочником "Курсы", также созданным в *главе 1*. В нем представлена информация о дате начала занятий, их продолжительности в днях, стоимости обучения и преподавателе. При работе с подобным справочником может возникнуть, например, следующий вопрос — как составить список курсов продолжительностью не менее пяти дней, стоимость обучения на которых не превышает 1500 рублей.

Для решения подобных задач в системе 1С:Предприятие используется специальный объект, который называется *запросом*. По существу любой запрос представляет собой требование к системе извлечь выборку с необходимой информацией из базы данных. Но кроме обыкновенного отбора запросы предоставляют еще много полезных возможностей — группировка отобранных данных, вычисление итогов и др. Запросы удобно использовать, когда требуется сформировать сложную выборку данных, сгруппированную и отсортированную необходимым образом.

В конфигурации, разработанной в *главе 3*, мы создали регистр сведений **Квитанции**. Используя запрос с группировкой по учащимся, можно легко подсчитать, кто какую сумму уже оплатил за обучение. Также с помощью запроса можно легко сформировать список тех учащихся, кто имеет неоплаченные квитанции.

С запросами мы уже эпизодически встречались в предыдущих главах книги. При этом те примеры запросов являлись результатом работы конструктора печати, который автоматически формировал необходимый программный код, включающий и текст запроса. Фактически подобные примеры являются неким анонсом данной главы. Здесь технология составления запросов будет рассмотрена, начиная с основных положений. На разнообразных задачах вы получите практические навыки по формированию учетной и сводной информации.

Но кроме запросов, в этой главе мы встретимся с еще одним важным элементом любой прикладной задачи количественного учета — *отчетом*. Это связано с тем, что при разработке решений, связанных с количественным учетом, ключевым компонентом являются отчеты. Здесь возможны варианты. Так, можно создать отчет без использования специально предназначенных для этого средств автоматизации. В этом случае требуется формировать макеты табличных документов и создавать программные процедуры, которые в режиме **1С:Предприятие** приводят к заполнению табличных документов в соответствии со структурой макетов.

Однако одним из основных достоинств системы 1С:Предприятие 8.2 являются эффективные возможности построения отчетов с минимальными усилиями со стороны разработчика. Наличие обеспечивающих это средств автоматизации повышает скорость разработки прикладных решений, а также предоставляет возможность самостоятельного формирования необходимых отчетов пользователям, не знакомым со встроенным языком программирования системы 1С:Предприятие. Этим ресурсам отведена *глава 5*, где будет рассматриваться система компоновки данных. В этой же главе основное внимание сосредоточено на запросах, а что касается от-

четов, то для вывода информации будут использоваться табличные документы и окно сообщений.

Взгляд на язык запросов

В первых главах книги в плане рассмотрения приемов программирования мы познакомились с рядом конструкций встроенного языка системы 1С:Предприятие 8.2. Так были рассмотрены программные объекты, их свойства и методы. Как и в большинстве подобных языков программирования, действия разработчика сводятся к определению точной последовательности шагов, направленных на достижение конкретной цели, и записи этих шагов в понятном компьютеру виде. Все начинается с формулировки задачи, затем составляется алгоритм действий, а далее подбираются соответствующие программные конструкции языка (этап кодирования).

В концепцию запроса заложена другая идея — мы лишь формулируем задачу для исполняющей системы на понятном ей языке (языке запросов). При построении запроса необходимо максимально точно описать, какой результат нам нужно получить. Сами же действия, требуемые для извлечения нужной информации, определяются и осуществляются исполняющей системой самостоятельно.

Мы познакомимся с основными конструкциями языка запросов на различных примерах. Однако при этом не ограничимся только справочной информацией, а проиллюстрируем почти все приведенные запросы практическими примерами.

В качестве иллюстрации идеи построения запросов рассмотрим текст запроса для извлечения информации, расположенной в полях **Договор** и **СуммаПоКвитанции** регистра сведений **Квитанции** (речь пойдет о конфигурации, разработанной в *главе 3*). Так, нам необходимо получить список выписанных квитанций с номерами договоров и указанными в квитанциях суммами. Такой несложный запрос выглядит следующим образом:

```
ВЫБРАТЬ Договор, СуммаПоКвитанции
ИЗ  РегистрСведений.Квитанции
```

В результате выполнения этого запроса мы получим таблицу из набора строк, в каждой из которых присутствуют две колонки. Число строк в таблице соответствует числу имеющихся записей в регистре сведений **Квитанции**.

В приведенном запросе использовано ключевое слово ВЫБРАТЬ, позволяющее указать список полей для данного запроса. Другое ключевое слово ИЗ позволяет определить объекты конфигурации, участвующие в рассматриваемом запросе. Так, в данной ситуации используется регистр сведений **Квитанции**, из которого извлекается информация о двух его полях (измерении **Договор** и ресурсе **СуммаПоКвитанции**).

В принципе язык запросов позволяет обходиться и без указания ключевого слова ИЗ, но в этом случае в конструкции запроса требуется указать полные имена полей, участвующих в запросе. Рассмотренный выше текст можно было оформить и по-другому:

```
ВЫБРАТЬ РегистрСведений.Квитанции.Договор,
        РегистрСведений.Квитанции.СуммаПоКвитанции
```

Существует много разнообразных синтаксических конструкций запросов. И далее мы познакомимся с ними, а также увидим, каким образом можно выполнять подобные запросы в системе 1С:Предприятие 8.2. При этом почти во всех примерах, кроме конструкций языка запросов, потребуется и уже в некоторой степени знакомый нам встроенный язык программирования системы 1С:Предприятие 8.2.

Извлечение информации о выписанных квитанциях

Обратимся к справочнику "Договоры", который был разработан нами в составе конфигурации *(см. главу 3)*. Будем считать, что нам, как разработчикам, необходимо обеспечить вывод информации о выписанных квитанциях при работе пользователя со справочником "Договоры", а именно требуется разместить в форме списка справочника "Договоры" кнопку, с помощью которой в окно сообщений (это одно из служебных окон приложения) будет выводиться необходимая информация. Эти сведения будут представлять собой перечень строк сообщений, в каждой из которых будет указано название договора и сумма по выписанной квитанции.

Напомним, что при работе в режиме **1С:Предприятие** все внесенные квитанции фиксируются в информационной структуре, созданной системой на основании объекта конфигурации **Квитанции** (регистра сведений).

Начнем практическую работу и откроем окно редактирования объекта конфигурации для справочника "Договоры" (рис. 4.1).

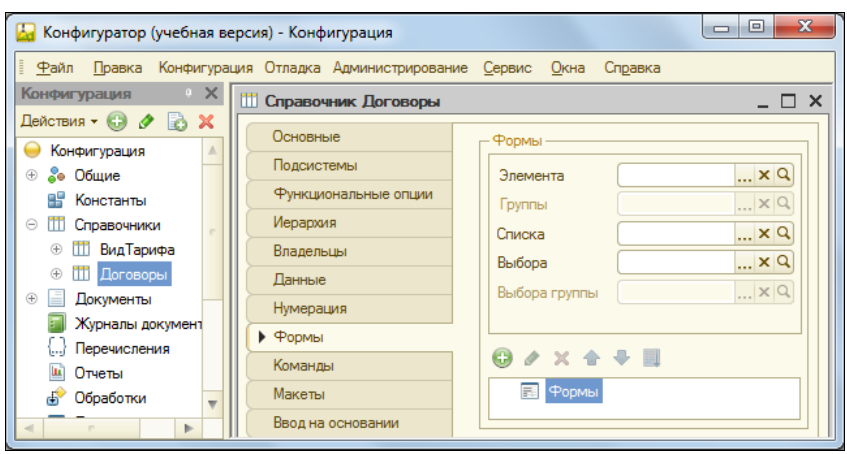


Рис. 4.1. Окно редактирования справочника "Договоры"

Мы пока не располагаем пользовательской формой списка, поэтому ее необходимо создать. Технический процесс создания формы для основного объекта конфигурации нам хорошо знаком из предыдущих глав. Так, на вкладке **Формы** необходимо

щелкнуть по пиктограмме с изображением лупы напротив раздела **Формы списка** (см. рис. 4.1). После этого в результате несложного диалога с конструктором формы перед нами откроется форма списка справочника "Договоры" в окне редактора форм (рис. 4.2).

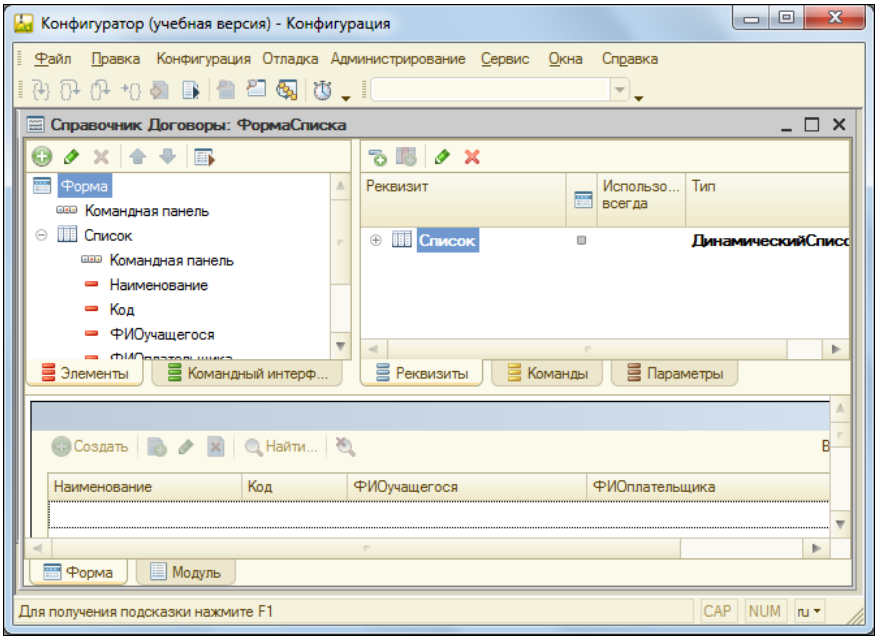


Рис. 4.2. Форма списка справочника "Договоры" в окне редактора форм

Далее в окне редактора форм требуется перейти на вкладку **Команды**, где с помощью соответствующей пиктограммы (на ней нанесен знак "плюс") следует добавить в разработанную форму команду **ДанныеПоКвитанциям** (рис. 4.3).

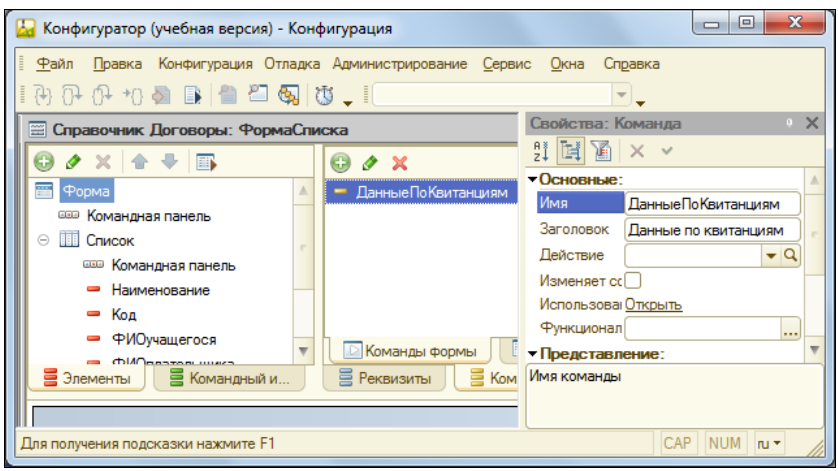


Рис. 4.3. Палитра свойств команды **ДанныеПоКвитанциям**

После этого в палитре свойств команды **ДанныеПоКвитанциям** (см. рис. 4.3) необходимо обратиться к разделу **Действие**, где с помощью щелчка мышью по пиктограмме с изображением лупы происходит переход в окно процедуры обработки данной команды. В качестве программного кода команды **ДанныеПоКвитанциям** (листинг 4.1) поместим единственную строку вызова серверной процедуры `ПолучитьИнформацию`, которую мы также разработаем.

Выделение отдельной серверной процедуры `ПолучитьИнформацию` связано с ее программными конструкциями, которые можно выполнить только на сервере, а не на клиенте. Текст данной процедуры представлен в листинге 4.2.

Листинг 4.1. Обработка команды ДанныеПоКвитанциям

```
&НаКлиенте
Процедура ДанныеПоКвитанциям(Команда)
ПолучитьИнформацию();
КонецПроцедуры
```

Листинг 4.2. Серверная процедура ПолучитьИнформацию

```
&НаСервере
Процедура ПолучитьИнформацию()
  Запрос = Новый Запрос;
  Запрос.Текст = "ВЫБРАТЬ  Договор,
  | СуммаПоКвитанции
  |  ИЗ  РегистрСведений.Квитанции";
  Результат = Запрос.Выполнить().Выбрать();
  Сообщение=Новый СообщениеПользователю;
  Пока Результат.Следующий() > 0 Цикл
     Сообщение.Текст=Строка(Результат.Договор) +
           "   " + Строка(Результат.СуммаПоКвитанции) + " руб. ";
     Сообщение.Сообщить();
  КонецЦикла;
КонецПроцедуры
```

В первой выполняемой строке процедуры `ПолучитьИнформацию` создается новый объект типа `Запрос`, а далее с помощью его свойства `Текст` непосредственно формируется содержание запроса. Это текст того запроса, который мы уже рассмотрели в начале данной главы. Заметим, что конструкция как данного, так и большинства запросов, занимает более одной строки. В этом случае для обозначения продолжения строки в тексте программной процедуры (или функции) используется символ вертикальной черты (`|`).

В следующей программной конструкции

```
Результат = Запрос.Выполнить().Выбрать()
```

сначала используется метод `Выполнить`, что приводит к выполнению запроса, а к результату выполнения этого метода применяется другой метод — `Выбрать`, который

позволяет произвести выборку информации из выполненного запроса. Фактически метод `Выбрать` подготавливает работу еще одного метода — `Следующий`, который осуществляет выборку очередной строки из результата выполнения запроса. Это позволяет последовательно перебирать строки в полученной выборке.

Далее в рассматриваемой процедуре `ПолучитьИнформацию` создается новое сообщение для пользователя и в цикле (по числу записей, извлеченных в результате выполнения запроса) выводится информация о выписанных квитанциях. В результате извлеченные с помощью запроса данные отображаются в окне сообщений.

> ***Примечание***
>
> `Строка()` — это функция преобразования значений. Она преобразует входной параметр в значение типа `Строка`, при этом входной параметр может иметь произвольный тип данных.

В процедуре листинга 4.2 мы использовали директиву `НаСервере`, однако контекст формы на сервере в данном случае не нужен — в серверной процедуре просто выполняются вычисления. В связи с этим, можно заменить указанную директиву на другую — `НаСервереБезКонтекста`.

Таким образом, команда (с необходимой дополнительной серверной процедурой) для выполнения запроса подготовлена. Далее необходимо на форме списка справочника "Договоры" создать кнопку, которая позволит выполнить разработанную команду.

Для этого следует в окне редактора форм (рис. 4.4) мышью перетащить пиктограмму команды **ДанныеПоКвитанциям** на пиктограмму **Командная панель**. В результате на форме списка справочника добавится новый интерфейсный элемент — кнопка, которую мы затем увидим в режиме **1С:Предприятие** при открытии формы списка справочника "Договоры".

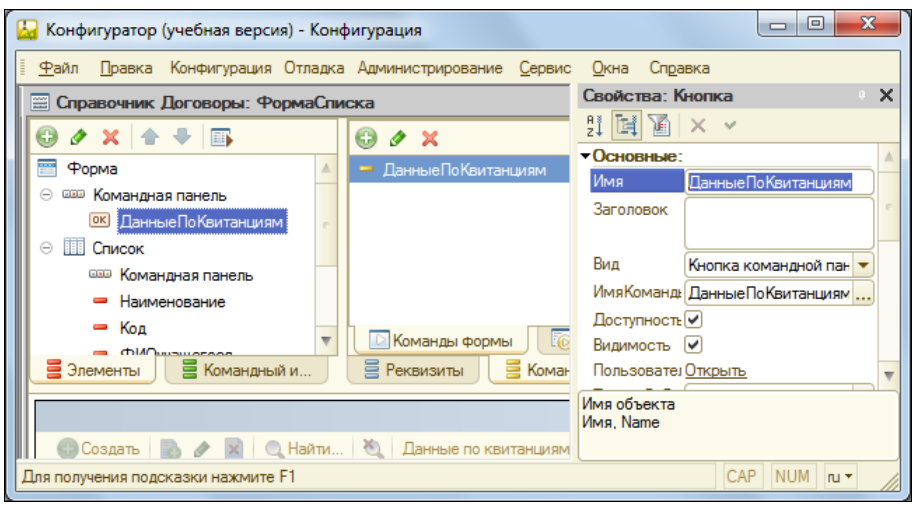


Рис. 4.4. Создание кнопки для выполнения команды в форме справочника "Договоры"

Теперь следует перейти в режим **1С:Предприятие** и обратиться к форме списка справочника "Договоры", где с помощью кнопки **Данные по квитанциям** в окне сообщений можно сформировать результат выполнения запроса (рис. 4.5).

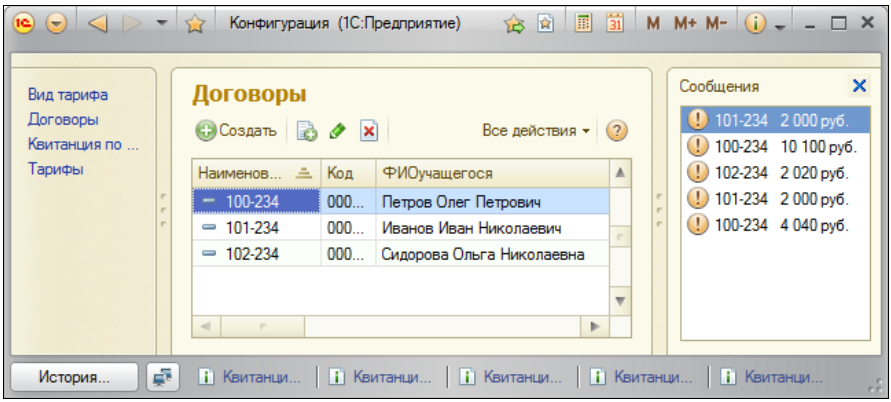


Рис. 4.5. Вывод результата запроса о квитанциях в окне сообщений

Примечание

В зависимости от ширины формы при работе в режиме **1С:Предприятие** созданные нами кнопки на форме могут автоматически убираться платформой. В этом случае следует обратиться к выпадающему списку **Все действия** (см. рис. 4.5), где управляющие кнопки всегда можно найти.

В следующем разделе мы рассмотрим выполнение описанного здесь запроса по квитанциям, но с использованием других технических приемов.

Использование табличного документа для вывода результатов выполнения запроса

В первой главе мы уже встречались с табличными документами. Они используются для формирования печатных форм справочников, документов, отчетов и т. д. Разработаем вариант выполнения запроса из предыдущего раздела для извлечения информации о выписанных квитанциях, но при этом используем для отображения информации табличный документ.

В процессе разработки нам понадобится макет табличного документа, поэтому в окне редактирования справочника "Договоры" перейдем на вкладку **Макеты** (рис. 4.6). Здесь для разработки нового макета воспользуемся соответствующей пиктограммой в пиктографическом меню. В результате на экране откроется окно конструктора макета (рис. 4.7), где с помощью переключателя необходимо указать тип макета — **Табличный документ.**

Несложный диалог с конструктором макета приводит к открытию на экране окна редактора макета (рис. 4.8). Здесь можно вводить текст и параметры в ячейки макета, а также определять свойства ячеек макета (шрифт, цвет, обводку и т. д.). Разра-

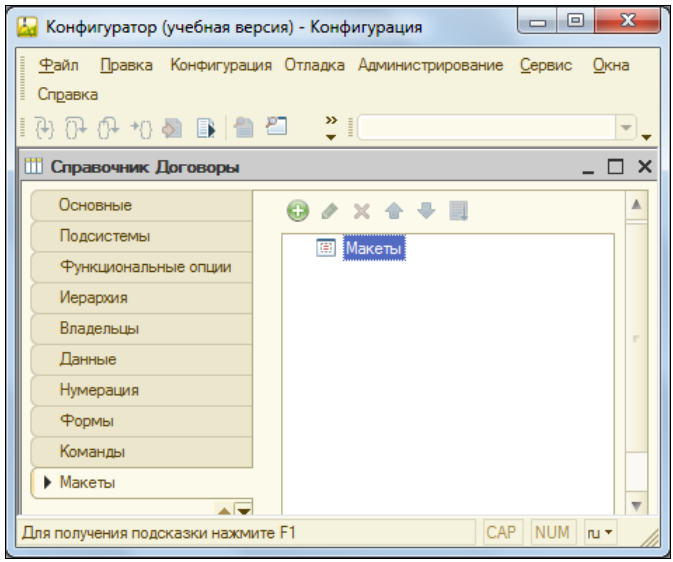


Рис. 4.6. Вкладка **Макеты** окна редактирования объекта конфигурации

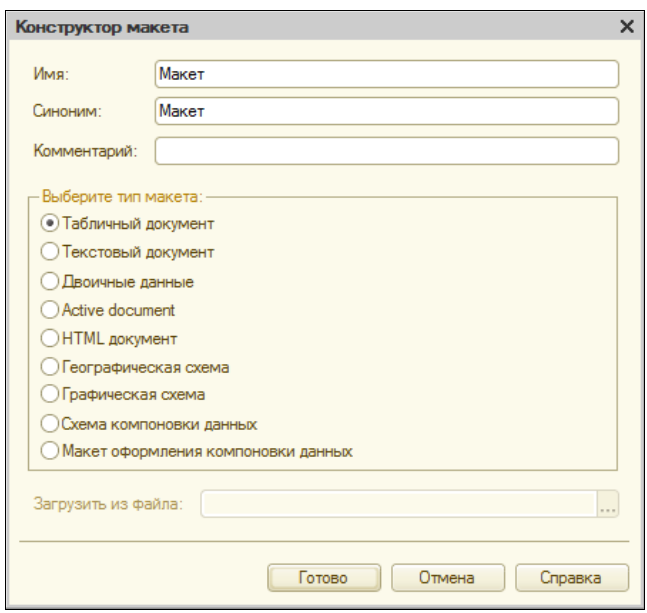


Рис. 4.7. Окно конструктора макета

ботанный макет платформа далее будет использовать при создании табличного документа в режиме **1С:Предприятие**.

Разрабатываемый табличный документ будет содержать перечень выписанных квитанций, поэтому создадим в макете две горизонтальные секции в соответствии с рис. 4.9. Одна секция требуется для вывода шапки табличного документа, а вторая — для перечня строк результата выполнения запроса.

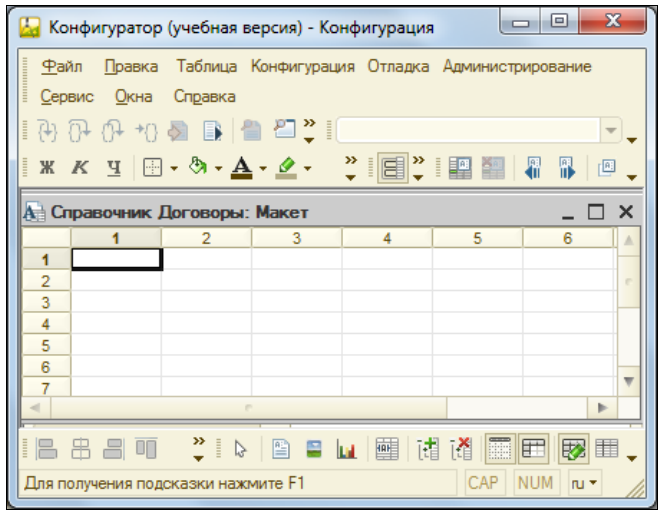


Рис. 4.8. Окно редактора макета

Важно отметить, что для двух заполненных ячеек в секции **Шапка** следует в палитре свойств определить **Текст** в качестве значения свойства **Заполнение**, а для двух других ячеек (из секции **Строка**) — **Параметр**.

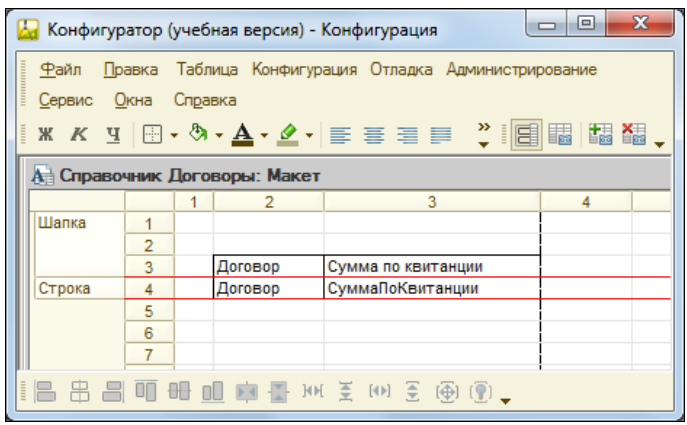


Рис. 4.9. Создание двух секций в макете

После формирования макета табличного документа перейдем к разработке программных процедур, которые позволят в режиме **1С:Предприятие** создать табличный документ на основании макета.

Откроем форму списка справочника "Договоры" в редакторе форм и создадим новую команду **ПечатьВТабДок** (рис. 4.10).

В качестве обработки данной команды введем процедуру, представленную в листинге 4.3. Другую часть программного кода формирования табличного документа придется выполнить на сервере (листинг 4.4).

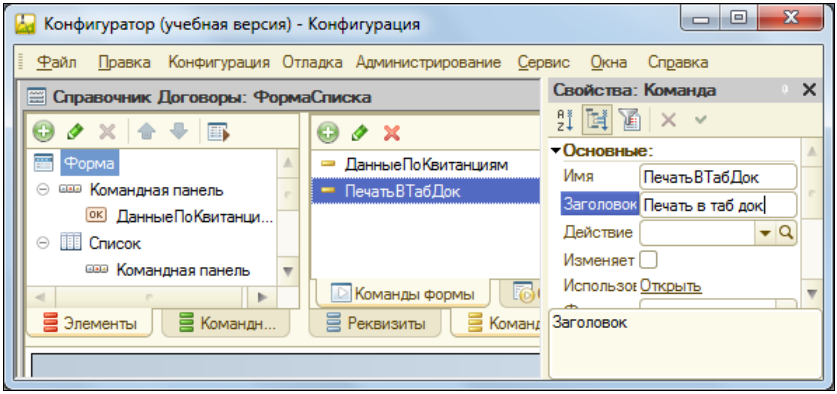


Рис. 4.10. Добавление команды **ПечатьВТабДок** в форме списка справочника "Договоры"

Листинг 4.3. Процедура обработки команды ПечатьВТабДок

```
&НаКлиенте
Процедура ПечатьВТабДок(Команда)
  ТабДок = Новый ТабличныйДокумент;
  ПечатьВДокумент(ТабДок);
  ТабДок.ОтображатьСетку = Ложь;
  ТабДок.Защита = Ложь;
  ТабДок.ТолькоПросмотр = Ложь;
  ТабДок.ОтображатьЗаголовки = Ложь;
  ТабДок.Показать();
КонецПроцедуры
```

Листинг 4.4. Серверная процедура ПечатьВДокумент

```
&НаСервереБезКонтекста
Процедура ПечатьВДокумент(ТабДок)
  ТабДок.Очистить();
  Макет = Справочники.Договоры.ПолучитьМакет("Макет");
  Шапка = Макет.ПолучитьОбласть("Шапка");
  ТабДок.Вывести(Шапка);
  Запрос = Новый Запрос;
  Запрос.Текст = "ВЫБРАТЬ  Договор, СуммаПоКвитанции
  | ИЗ  РегистрСведений.Квитанции";
  Результат = Запрос.Выполнить().Выбрать();
  Пока Результат.Следующий() > 0 Цикл
    Область = Макет.ПолучитьОбласть("Строка");
    Область.Параметры.Договор = Результат.Договор;
    Область.Параметры.СуммаПоКвитанции = Результат.СуммаПоКвитанции;
    ТабДок.Вывести(Область);
  КонецЦикла;
КонецПроцедуры
```

Далее требуется создать на форме списка справочника "Договоры" кнопку для выполнения разработанной команды **ПечатьВТабДок**. Технически для этого достаточно мышью перетащить пиктограмму данной команды на командную панель (рис. 4.11).

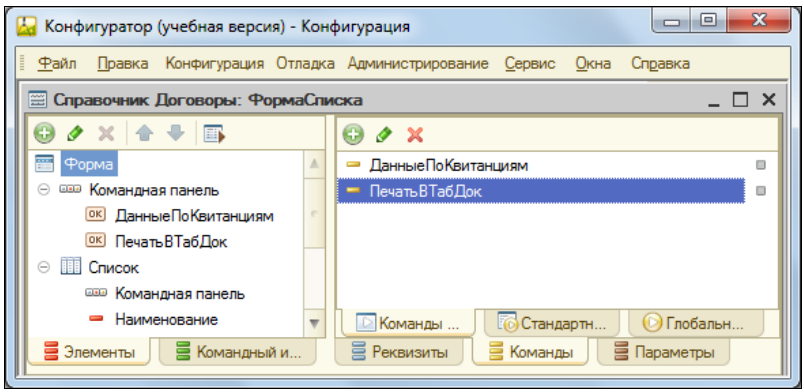
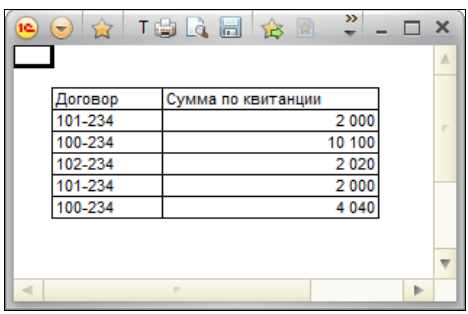


Рис. 4.11. Добавление кнопки для выполнения команды **ПечатьВТабДок**

Теперь после перехода в режим **1С:Предприятие** следует открыть форму списка справочника "Договоры" и воспользоваться созданной кнопкой **ПечатьВТабДок** для формирования табличного документа (рис. 4.12). Информация о квитанциях выведена без дополнительного упорядочивания — в той последовательности, как они были введены в информационную базу.



Договор	Сумма по квитанции
101-234	2 000
100-234	10 100
102-234	2 020
101-234	2 000
100-234	4 040

Рис. 4.12. Отображение результата запроса о квитанциях в табличном документе

Псевдонимы и упорядочивание в запросе

Если поля или названия таблиц, участвующих в запросе, являются достаточно длинными, то для них удобнее назначить псевдонимы с помощью ключевого слова КАК. В результате это позволяет обеспечить более компактную запись программных конструкций.

Для упорядочивания информации в запросе предназначена конструкция УПОРЯДОЧИТЬ ПО, после которой следует указать поле сортировки. Кроме того, для определения порядка сортировки используются следующие ключевые слова:

- УБЫВ — для упорядочивания в порядке убывания;
- ВОЗР — для упорядочивания в порядке возрастания.

Например, чтобы информация о квитанциях была отсортирована в порядке убывания сумм, указанных в квитанциях, следует использовать следующую конструкцию:

```
УПОРЯДОЧИТЬ ПО СуммаПоКвитанции УБЫВ
```

> ***Примечание***
>
> В предложении УПОРЯДОЧИТЬ ПО через запятую перечисляются условия, в соответствии с которыми необходимо упорядочить результат запроса. Выбранные данные упорядочиваются сначала по первому условию, затем по второму и т. д. В результате извлеченная запросом информация становится структурированной и удобной для восприятия пользователями.

Рассмотрим использование псевдонима и применение упорядочивания на конкретном примере. Будем считать, что нам нужно упорядочить перечень информации о квитанциях сначала по договорам (по алфавиту в порядке возрастания), а затем по суммам (в порядке убывания). Результат выполнения этого запроса следует вывести в окно сообщений.

Для выполнения разработки откроем форму списка справочника "Договоры" в окне редактора форм. Здесь создадим новую команду с именем ПорядокИПсевдонимы, процедура обработки которой приведена в листинге 4.5.

Листинг 4.5. Процедура обработки команды ПорядокИПсевдонимы

```
&НаКлиенте
Процедура ПорядокИПсевдонимы(Команда)
  ПолучитьИнфоПорядок();
КонецПроцедуры
```

Далее необходимо разработать серверную процедуру ПолучитьИнфоПорядок (листинг 4.6) для выполнения необходимых вычислительных действий.

Листинг 4.6. Серверная процедура ПолучитьИнфоПорядок

```
&НаСервереБезКонтекста
Процедура ПолучитьИнфоПорядок()
  Запрос = Новый Запрос;
  Запрос.Текст = "ВЫБРАТЬ  Договор,
  | СуммаПоКвитанции
  | КАК Сумма
  |  ИЗ  РегистрСведений.Квитанции
  | УПОРЯДОЧИТЬ ПО Договор ВОЗР,Сумма УБЫВ";
  Результат = Запрос.Выполнить().Выбрать();
  Сообщение=Новый СообщениеПользователю;
```

```
  Пока Результат.Следующий() > 0 Цикл
     Сообщение.Текст=Строка(Результат.Договор) +
        "    " + Строка(Результат.Сумма) + " руб. ";
     Сообщение.Сообщить();
   КонецЦикла;
КонецПроцедуры
```

Теперь необходимо создать в форме списка справочника "Договоры" кнопку, с помощью которой можно будет выполнить разработанную команду в режиме **1С:Предприятие**. Технически для этого следует в окне редактора форм перенести созданную команду **ПорядокИПсевдонимы** на элемент **Командная панель**. В результате на командной панели появится новая кнопка для выполнения разработанной команды (рис. 4.13).

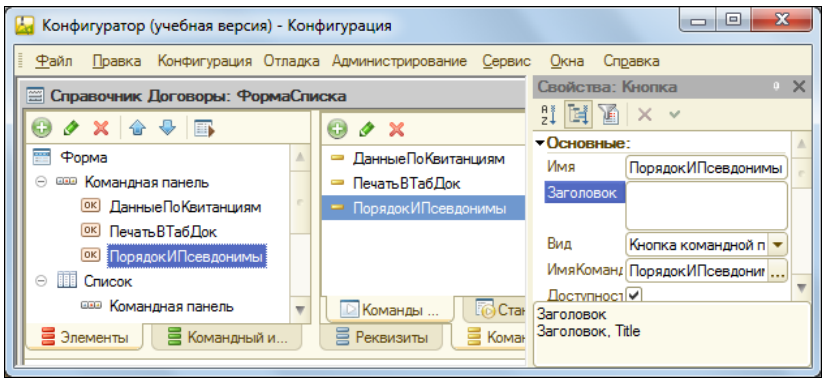


Рис. 4.13. Добавление кнопки **ПорядокИПсевдонимы** в форму списка справочника "Договоры"

На этом необходимые действия в режиме конфигуратора завершены и теперь следует перейти в режим **1С:Предприятие** и открыть форму списка справочника "Договоры". После щелчка по добавленной кнопке **ПорядокИПсевдонимы** производится выполнение разработанного запроса и вывод результата в окно сообщений (рис. 4.14). Видно, что информация о квитанциях упорядочена по договорам, а внутри договоров по суммам (в порядке убывания).

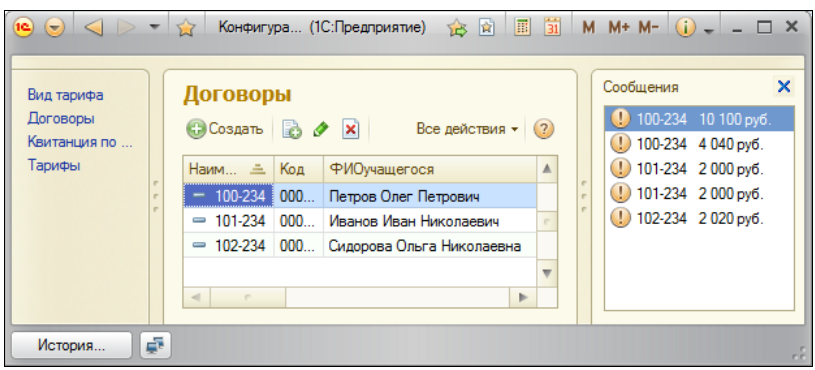


Рис. 4.14. Упорядочивание квитанций по договорам и суммам

Отбор максимальных (минимальных) значений

При большом объеме информации в результате выполненного запроса часто требуется просмотреть только несколько наиболее больших или, наоборот, маленьких значений. Например, это может понадобиться для получения списков самых дорогих услуг или товаров.

В этом случае в конструкцию `ВЫБРАТЬ` добавляется ключевое слово `ПЕРВЫЕ`. Продемонстрируем его использование на примере. Будем считать, что нас интересуют две квитанции с самыми большими выписанными суммами.

Как и в предыдущих примерах, реализуем отбор информации с помощью дополнительной кнопки в форме списка справочника "Договоры". На первом этапе создадим новую команду `ОтборMax`, процедура обработки которой приведена в листинге 4.7. Как и в предыдущем примере, для выполнения запроса нам понадобится серверная процедура `ПолучитьИнфоМакс`, которая отражена в листинге 4.8.

Далее требуется разместить на форме списка справочника "Договоры" соответствующую кнопку (рис. 4.15), щелчком по которой разработанную команду можно выполнить в режиме **1С:Предприятие**.

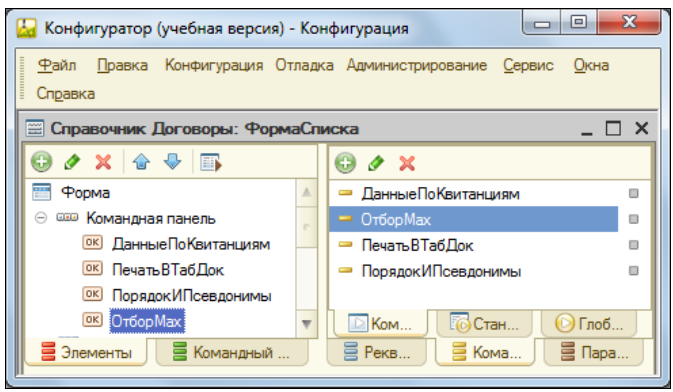


Рис. 4.15. Добавление кнопки **ОтборMax** в форму списка справочника "Договоры"

Теперь уже разработка готова и на рис. 4.16 отражен результат работы этой процедуры в режиме **1С:Предприятие**: показана информация (названия договоров и соответствующие суммы) о двух максимальных по сумме квитанциях.

Листинг 4.7. Процедура обработки команды отбора максимальных квитанций

```
&НаКлиенте
Процедура ОтборMax(Команда)
    ПолучитьИнфоМакс();
КонецПроцедуры
```

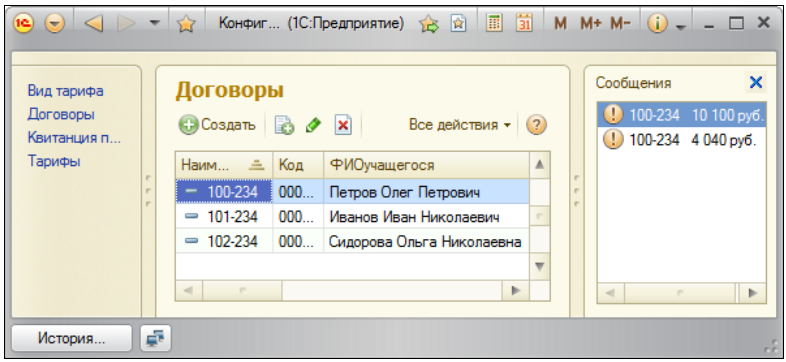


Рис. 4.16. Отображение результата запроса для извлечения максимальных сумм по квитанциям

Листинг 4.8. Серверная процедура ПолучитьИнфоМакс

```
&НаСервереБезКонтекста
Процедура ПолучитьИнфоМакс()
    Запрос = Новый Запрос;
    Запрос.Текст = "ВЫБРАТЬ ПЕРВЫЕ 2  Договор,
    | СуммаПоКвитанции
    | КАК Сумма
    | ИЗ  РегистрСведений.Квитанции
    | УПОРЯДОЧИТЬ ПО Сумма УБЫВ";
    Результат = Запрос.Выполнить().Выбрать();
    Сообщение=Новый СообщениеПользователю;
    Пока Результат.Следующий() > 0 Цикл
        Сообщение.Текст=Строка(Результат.Договор) +
            "    " + Строка(Результат.Сумма) + " руб. ";
        Сообщение.Сообщить();
    КонецЦикла;
КонецПроцедуры
```

Если бы нас интересовали, наоборот, две квитанции на минимальные суммы, то в тексте листинга 4.8 необходимо было бы изменить только конструкцию, связанную с упорядочиванием полученных данных. В этом случае текст запроса должен выглядеть следующим образом:

```
Запрос.Текст = "ВЫБРАТЬ ПЕРВЫЕ 2 Договор,СуммаПоКвитанции
| КАК Сумма
| ИЗ РегистрСведений.Квитанции
| УПОРЯДОЧИТЬ ПО Сумма ВОЗР";
```

Ключевое слово *ИЕРАРХИЯ*

В предыдущих примерах мы рассмотрели упорядочивание, которое в данной языковой конструкции весьма эффективно для объектов без групп (без иерархии). Пе-

ред рассмотрением следующей разработки внесем небольшие изменения в справочник "Договоры".

Для начала необходимо указать, что справочник будет иметь иерархическую структуру, что выполняется на вкладке **Иерархия** в окне редактирования объекта конфигурации (рис. 4.17). Как видно, здесь мы установили флажок **Иерархический справочник.**

После выполненных изменений следует перейти в режим **1С:Предприятие** и создать в справочнике "Договоры" две группы (рис. 4.18). Ранее внесенные договоры учащихся Петрова и Иванова перенесем в группу **Филиал 1**, а в группу **Филиал 2** запишем Сидорову.

Примечание

Технически для переноса определенного элемента справочника в группу следует его сначала выделить, а затем с помощью правой кнопки мыши вызвать контекстное меню, в котором воспользоваться пунктом **Переместить в группу**.

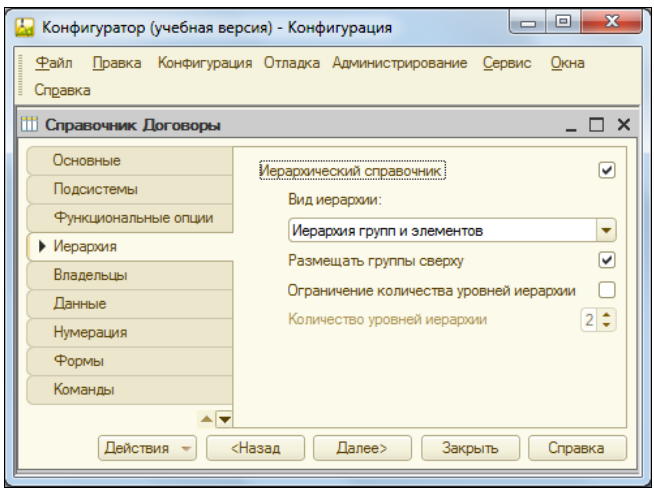


Рис. 4.17. Вкладка **Иерархия** окна редактирования справочника "Договоры"

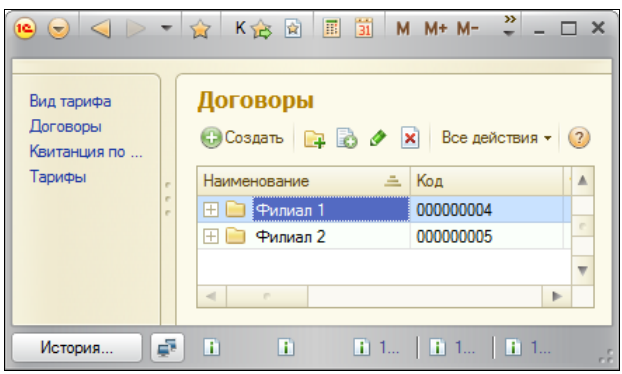


Рис. 4.18. Создание групп в справочнике "Договоры" в режиме **1С:Предприятие**

Примечание

Для иерархических справочников системой предоставляется несколько вариантов для просмотра. Чтобы указать конкретный вариант, следует обратиться к выпадающему меню **Все действия** (см. рис. 4.18). Здесь присутствует пункт **Режим просмотра**, в котором можно указать один из нескольких вариантов (**Иерархический список**, **Список**, **Дерево**).

Теперь сконструируем запрос, который позволит выбрать из справочника "Договоры" названия договоров и соответствующие фамилии плательщиков, но при этом записи результата выполнения запроса должны группироваться по филиалам.

Подобное упорядочивание в запросе достигается за счет использования ключевого слова ИЕРАРХИЯ.

Создадим новую команду ПолучитьИерархию в форме списка справочника "Договоры", текст процедуры которой приведен в листинге 4.9. Необходимая серверная процедура ОтборПоИерархии для выполнения данной команды представлена в листинге 4.10.

Листинг 4.9. Процедура обработки команды ПолучитьИерархию

```
&НаКлиенте
Процедура ПолучитьИерархию (Команда)
ОтборПоИерархии ();
КонецПроцедуры
```

Листинг 4.10. Серверная процедура ОтборПоИерархии

```
&НаСервереБезКонтекста
Процедура ОтборПоИерархии ()
   Запрос = Новый Запрос;
   Запрос.Текст = "ВЫБРАТЬ Наименование КАК Договор,
   | ФИОПлательщика
   |  ИЗ  Справочник.Договоры
   | УПОРЯДОЧИТЬ ПО Договор ИЕРАРХИЯ";
   Результат = Запрос.Выполнить().Выбрать();
   Сообщение=Новый СообщениеПользователю;
   Пока Результат.Следующий() > 0 Цикл
      Сообщение.Текст=Строка(Результат.Договор) +
          "   " + Строка(Результат.ФИОПлательщика);
      Сообщение.Сообщить();
   КонецЦикла;
КонецПроцедуры
```

Теперь программные процедуры готовы и осталось разместить на форме списка справочника кнопку **ПолучитьИерархию** для выполнения разработанной команды. Технические действия для этого аналогичны тем, которые использованы в предыдущих разделах.

Далее перейдем в режим **1С:Предприятие** и в форме справочника "Договоры" щелкнем по созданной кнопке **ПолучитьИерархию**. Результат выполнения разработанного запроса приведен на рис. 4.19.

Примечание

Если какая-то созданная кнопка для выполнения необходимой команды в форме списка не отражается, то обратиться к ней можно с помощью выпадающего списка **Все действия** (см. рис. 4.19).

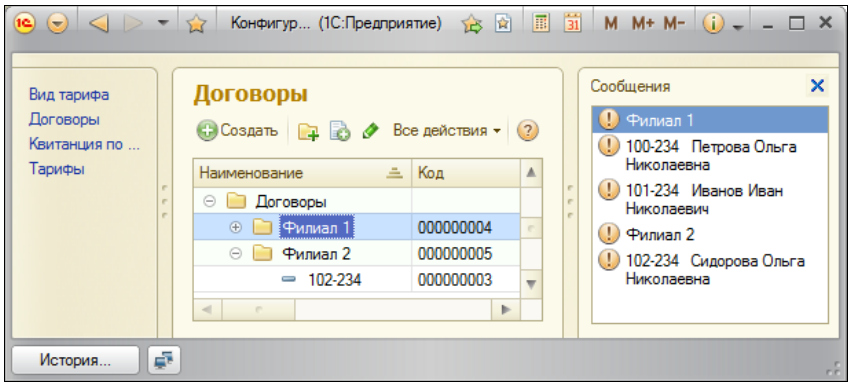


Рис. 4.19. Результат отбора плательщиков по иерархии

Исключение в отборе повторяющейся информации

Ранее в листинге 4.6 был приведен текст процедуры с запросом, который позволяет извлечь информацию о выписанных квитанциях (из регистра сведений **Квитанции**).

В этом случае по результату запроса мы видим перечень, в котором отражены договоры и указанные в квитанциях суммы. Однако если задача заключается в получении списка всех договоров, по которым выписывалась хотя бы одна квитанция, то рассмотренная процедура не подходит.

В этом разделе нам необходимо отделить договоры, по которым квитанции выдавались плательщикам, от тех, по которым они еще не выписывались. При этом повторений в перечне договоров быть не должно.

В языке запросов существует ключевое слово `РАЗЛИЧНЫЕ`, позволяющее отобрать только отличающиеся строки. Иначе говоря, в выборке результата выполнения запроса, включающего это ключевое слово, повторений не будет.

Организуем извлечение договоров из регистра **Квитанции** с использованием ключевого слова `РАЗЛИЧНЫЕ`. Для этого создадим в форме списка справочника "Договоры" новую команду `ОтборБезПовторений`. Ее процедура обработки приведена в листинге 4.11. Необходимая серверная процедура `ПолучитьИнфоБезПовторений` представлена в листинге 4.12.

После размещения программных процедур требуется создать командную кнопку для выполнения разработанной команды `ОтборБезПовторений` на форме списка справочника "Договоры" (рис. 4.20). Технические действия для этого нам уже знакомы.

Теперь разработка готова, и на рис. 4.21 показан результат выполнения рассмотренного запроса в режиме работы **1С:Предприятие**. Видно, что в списке нет названий договоров, по которым еще не выписывались квитанции (в данном случае это договор 100-99).

Листинг 4.11. Процедура обработки команды ОтборБезПовторений

```
&НаКлиенте
Процедура ОтборБезПовторений(Команда)
    ПолучитьИнфоБезПовторений();
КонецПроцедуры
```

Листинг 4.12. Серверная процедура ПолучитьИнфоБезПовторений

```
&НаСервереБезКонтекста
Процедура ПолучитьИнфоБезПовторений()
  Запрос = Новый Запрос;
  Запрос.Текст = "ВЫБРАТЬ РАЗЛИЧНЫЕ  Договор
  |  ИЗ  РегистрСведений.Квитанции
  | УПОРЯДОЧИТЬ ПО Договор ВОЗР";
  Результат = Запрос.Выполнить().Выбрать();
  Сообщение=Новый СообщениеПользователю;
  Пока Результат.Следующий() > 0 Цикл
    Сообщение.Текст=Строка(Результат.Договор);
    Сообщение.Сообщить();
 КонецЦикла;
КонецПроцедуры
```

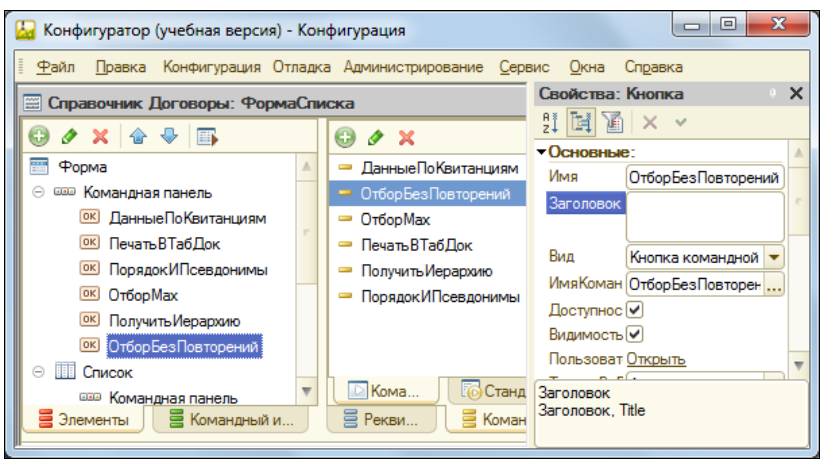


Рис. 4.20. Добавление кнопки в форму списка справочника "Договоры"

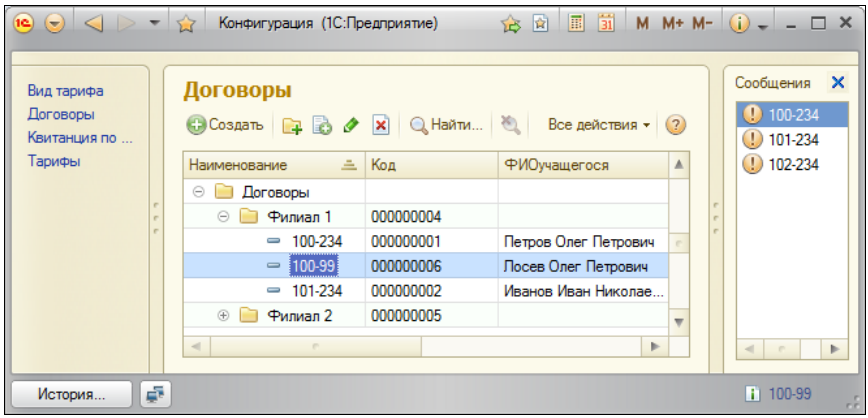


Рис. 4.21. Отбор договоров (без повторений), по которым выписывались квитанции

Ключевое слово *ГДЕ*

До этого момента нам еще не встречалось ключевое слово языка запросов `ГДЕ`. Оно позволяет задать условие отбора данных из исходных таблиц. В результате в запросе окажутся отобранными только те записи, для которых выполняется заданное условие. Используем данную возможность для того, чтобы отобрать только те договоры, по которым были оплаты (у квитанций выписанных по этим договорам реквизит **Оплачено** установлен в значение **Истина**).

Создадим новую команду в форме списка справочника "Договоры" и разработаем уже традиционную для этой главы процедуру обработки данной команды (листинг 4.13). После этого в том же окне редактора программного кода поместим серверную процедуру (рис. 4.14), в которой фактически и будет реализовано выполнение запроса.

Напомним, что для отметки оплаты квитанций в регистр **Квитанции** ранее мы включили реквизит **Оплачено**, который может принимать значения **Истина** или **Ложь**.

> ***Примечание***
>
> При проведении (или при перепроведении) документа "КвитанцияПоДоговору" осуществляется установка значения реквизита **Оплачено** в регистре **Квитанции**.

И в запросе листинга 4.14 конструкция с использованием ключевого слова `ГДЕ` позволяет оставить в отобранных данных только те договоры, у которых рассматриваемый реквизит установлен в значение **Истина**.

Листинг 4.13. Процедура обработки команды КлючевоеСловоГДЕ

```
&НаКлиенте
Процедура КлючевоеСловоГДЕ(Команда)
  ОтборСоСловомГДЕ();
КонецПроцедуры
```

Листинг 4.14. Серверная процедура ОтборСоСловомГДЕ

```
&НаСервереБезКонтекста
Процедура ОтборСоСловомГДЕ ()
  Запрос = Новый Запрос;
  Запрос.Текст = "ВЫБРАТЬ РАЗЛИЧНЫЕ  Договор
  |  ИЗ  РегистрСведений.Квитанции
  |  ГДЕ Оплачено=Истина
  |  УПОРЯДОЧИТЬ ПО Договор ВОЗР";
  Результат = Запрос.Выполнить().Выбрать();
  Сообщение=Новый СообщениеПользователю;
  Пока Результат.Следующий() > 0 Цикл
    Сообщение.Текст=Строка(Результат.Договор);
    Сообщение.Сообщить();
 КонецЦикла;
КонецПроцедуры
```

После размещения процедур добавим в форму списка справочника "Договоры" уже обязательную кнопку для выполнения команды (на этот раз — **КлючевоеСловоГДЕ**). Перейдем в режим **1С:Предприятие** и протестируем созданную разработку. Результат выполнения команды **КлючевоеСловоГДЕ** продемонстрирован на рис. 4.22, где показано окно сообщений с результатами отбора.

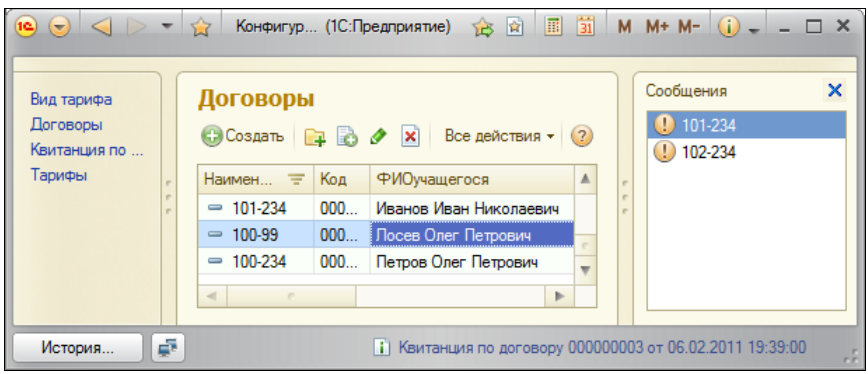


Рис. 4.22. Отбор договоров с оплаченными квитанциями

Использование логических операторов в запросе

В конструкции `ГДЕ` можно использовать логические операторы (`И`, `ИЛИ`, `НЕ`), которые позволяют построить достаточно сложные логические выражения. Скажем, применительно к предыдущему примеру нас могут интересовать договоры, по которым имеются оплаченные квитанции на сумму более 3000 рублей.

Создадим новую команду в форме списка справочника "Договоры" и разработаем традиционную для данной главы процедуру обработки команды (листинг 4.15). По-

сле этого нам понадобится серверная процедура (листинг 4.16), предназначенная непосредственно для выполнения запроса.

Далее следует разместить на форме списка справочника новую кнопку, с помощью которой в режиме **1С:Предприятие** можно будет выполнить разработанную команду `КомбинацияДвухУсловий`.

Теперь перейдем в пользовательский режим, откроем форму списка справочника "Договоры" и щелкнем по новой кнопке **КомбинацияДвухУсловий**. Как видно, в список отобранных данных (окно **Сообщения** на рис. 4.23) за счет указанной комбинации условий попал только один договор.

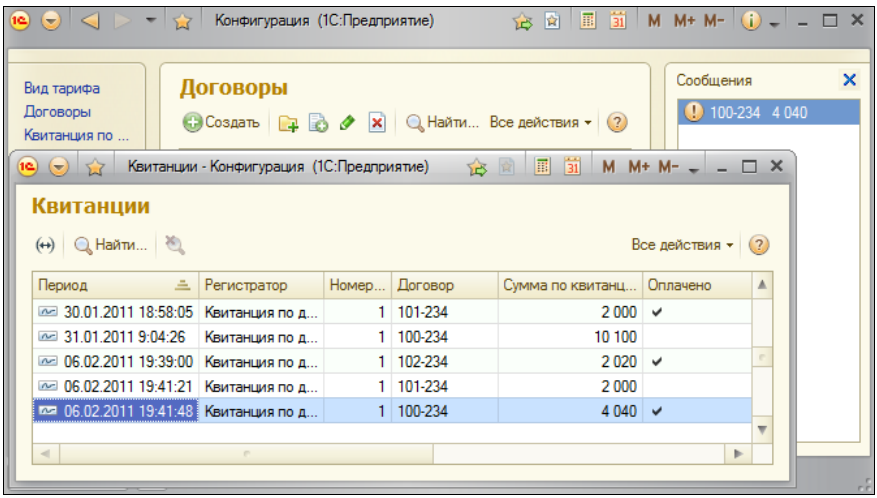


Рис. 4.23. Результат выполнения запроса с использованием логического оператора

Листинг 4.15. Процедура обработки команды КомбинацияДвухУсловий

```
&НаКлиенте
Процедура КомбинацияДвухУсловий(Команда)
    ИспользованиеДвухУсловий();
КонецПроцедуры
```

Листинг 4.16. Серверная процедура ИспользованиеДвухУсловий

```
&НаСервереБезКонтекста
Процедура ИспользованиеДвухУсловий ()
    Запрос = Новый Запрос;
    Запрос.Текст = "ВЫБРАТЬ РАЗЛИЧНЫЕ  Договор, СуммаПоКвитанции
    |  ИЗ  РегистрСведений.Квитанции
    |  ГДЕ Оплачено=Истина И СуммаПоКвитанции > 3000
    |  УПОРЯДОЧИТЬ ПО Договор ВОЗР";
    Результат = Запрос.Выполнить().Выбрать();
    Сообщение=Новый СообщениеПользователю;
```

```
    Пока Результат.Следующий() > 0 Цикл
        Сообщение.Текст=Строка(Результат.Договор) +
            "   " + Строка(Результат.СуммаПоКвитанции);
        Сообщение.Сообщить();
    КонецЦикла;
КонецПроцедуры
```

Приведем еще один пример запроса, в котором будут комбинироваться конструкции ИЛИ и И. Скажем, нас интересуют все квитанции, которые выписывались на сумму более 2000 рублей для оплаты за один или два месяца. В этом случае для нас представляют интерес два реквизита из регистра сведений **Квитанции**:

- **ОплаченныеМесяцы**;
- **СуммаПоКвитанции**.

В листинге 4.17 приведена процедура обработки команды КомбинацияНескольких-Условий, а в листинге 4.18 показана серверная процедура, реализующая необходимую выборку информации с помощью запроса.

Листинг 4.17. Процедура обработки команды КомбинацияНесколькихУсловий

```
&НаКлиенте
Процедура КомбинацияНесколькихУсловий(Команда)
ИспользованиеНесколькихУсловий();
КонецПроцедуры
```

Листинг 4.18. Серверная процедура ИспользованиеНесколькихУсловий

```
&НаСервереБезКонтекста
Процедура ИспользованиеНесколькихУсловий()
Запрос = Новый Запрос;
Запрос.Текст = "ВЫБРАТЬ РАЗЛИЧНЫЕ Договор, СуммаПоКвитанции
| ИЗ РегистрСведений.Квитанции
| ГДЕ (ОплаченныеМесяцы = 1 ИЛИ ОплаченныеМесяцы = 2)
| И СуммаПоКвитанции > 2000
| УПОРЯДОЧИТЬ ПО Договор ВОЗР";
Результат = Запрос.Выполнить().Выбрать();
Сообщение=Новый СообщениеПользователю;
Пока Результат.Следующий() > 0 Цикл
 Сообщение.Текст=Строка(Результат.Договор) +
   "   " + Строка(Результат.СуммаПоКвитанции);
 Сообщение.Сообщить();
КонецЦикла;
КонецПроцедуры
```

Осталось в форме списка справочника "Договоры" создать кнопку для выполнения команды, после чего следует перейти в режим **1С:Предприятие** и получить в окне **Сообщения** информацию об отобранных квитанциях (рис. 4.24).

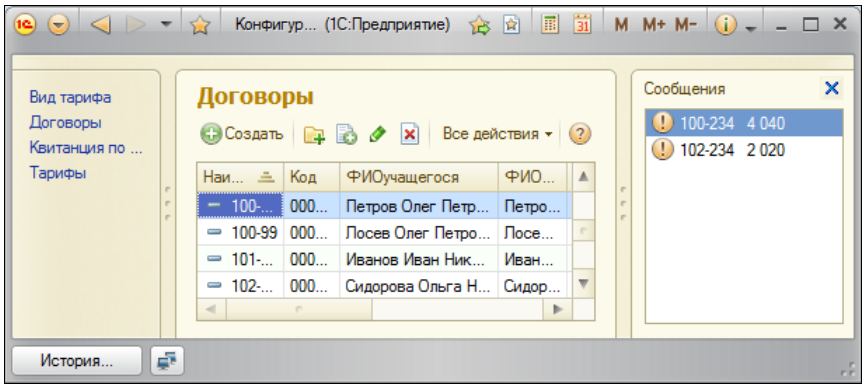


Рис. 4.24. Результат выполнения запроса с использованием нескольких логических операторов

Примечание

При написании сложных логических конструкций следует обращать внимание на скобки, которые, как и в арифметических операциях, указывают порядок, в котором производится проверка выражения.

Агрегатные функции в запросах

Таким образом, мы на несложных примерах познакомились с технологией выборки информации из базы данных. Однако часто на практике требуется получить и разнообразную сводную информацию. В частности, по данным регистра **Квитанции** управленческому персоналу могут понадобиться ответы на такие вопросы:

- Какова итоговая сумма оплаты, произведенная каждым учащимся?
- Какова средняя сумма оплаты учащимися?
- Какова наибольшая сумма оплаты по каждому учащемуся?

Для реализации подобных запросов следует воспользоваться агрегатными функциями и группировками.

Примечание

В язык запросов включены агрегатные функции, которые используются при группировке результатов запросов, а также при подсчете итогов.

Любая агрегатная функция принимает в качестве аргумента какой-либо столбец, а возвращает единственное значение. В табл. 4.2 перечислены имеющиеся в системе 1С:Предприятие 8.2 агрегатные функции.

Таблица 4.2. *Агрегатные функции*

Название	Описание
СУММА	Вычисляет сумму всех значений, содержащихся в указанном столбце. В качестве параметра функции можно указывать только поле, содержащее числовое значение

Таблица 4.2 *(окончание)*

Название	Описание
МАКСИМУМ	Находит наибольшее значение в указанном столбце
МИНИМУМ	Находит наименьшее значение в указанном столбце
СРЕДНЕЕ	Вычисляет среднее арифметическое значение по указанному столбцу
КОЛИЧЕСТВО	Подсчитывает количество значений, содержащихся в указанном столбце

В данной ситуации для демонстрации запроса с использованием агрегатных функций более подходит табличный документ. Поэтому обеспечим формирование соответствующего табличного документа при работе со справочником "Договоры".

Для начала разработаем макет табличного документа (рис. 4.25). В качестве имени макета укажем **МакетДляСводнойИнформации**.

Макет будет содержать две секции — **ЗаголовокИШапка** (для названия отчета и вывода поясняющей текстовой информации перед табличными данными) и **Строка** (в которой будут выводиться значения переменных, определенных в тексте дальнейшей процедуры). Кроме заголовка отчета, для всех ячеек с информацией (за исключением названия отчета), расположенных в секциях **ЗаголовокИШапка** и **Строка** установим обводку линиями. Для этого (как мы уже знаем) следует обратиться к палитре свойств ячеек.

Для заполненных ячеек секции **ЗаголовокИШапка** рассматриваемого макета в качестве значения свойства **Заполнение** укажем **Текст**.

У заполненных ячеек секции макета **Строка** в качестве значения свойства **Заполнение** укажем **Параметр**.

После этого (как и в предыдущих примерах) добавим в форму списка справочника "Договоры" команду, которую назовем `СводнаяИнформацияПоДоговорам`. Клиентская процедура обработки данной команды представлена в листинге 4.19, а серверная часть программного кода приведена в листинге 4.20.

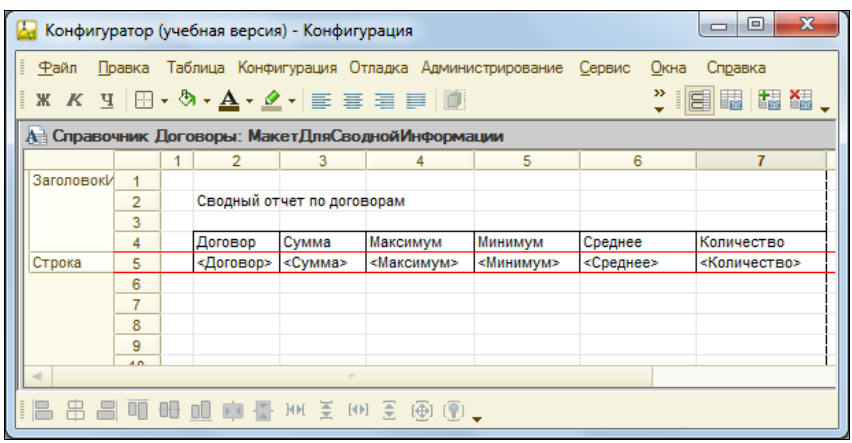


Рис. 4.25. Макет табличного документа

Листинг 4.19. Процедура обработки команды СводнаяИнформацияПоДоговорам

```
&НаКлиенте
Процедура СводнаяИнформацияПоДоговорам(Команда)
ТабДок = Новый ТабличныйДокумент;
ПечатьВСводнуюТаблицу(ТабДок);
ТабДок.ОтображатьСетку = Ложь;
ТабДок.Защита = Ложь;
ТабДок.ТолькоПросмотр = Ложь;
ТабДок.ОтображатьЗаголовки = Ложь;
ТабДок.Показать();
КонецПроцедуры
```

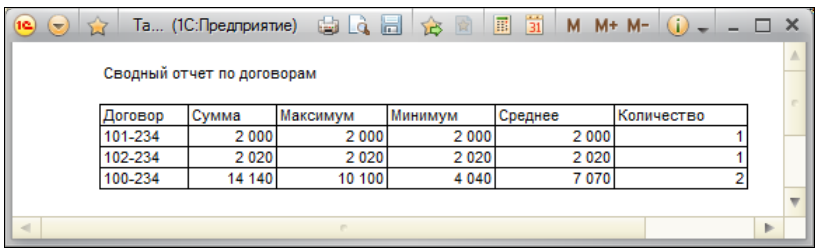
Листинг 4.20. Процедура ПечатьВСводнуюТаблицу

```
&НаСервереБезКонтекста
Процедура ПечатьВСводнуюТаблицу(ТабДок)
ТабДок.Очистить();
Макет = Справочники.Договоры.ПолучитьМакет
    ("МакетДляСводнойИнформации");
ЗаголовокИШапка = Макет.ПолучитьОбласть("ЗаголовокИШапка");
ТабДок.Вывести(ЗаголовокИШапка);
Запрос = Новый Запрос;
Запрос.Текст = "ВЫБРАТЬ Договор КАК Договор,
| СУММА(СуммаПоКвитанции) КАК Сумма,
| МАКСИМУМ(СуммаПоКвитанции) КАК Максимум,
| МИНИМУМ(СуммаПоКвитанции) КАК Минимум,
| СРЕДНЕЕ(СуммаПоКвитанции) КАК Среднее,
| КОЛИЧЕСТВО(*) КАК Количество
| ИЗ
| РегистрСведений.Квитанции
| ГДЕ Оплачено=Истина
| СГРУППИРОВАТЬ ПО Договор";
Выборка = Запрос.Выполнить().Выбрать();
Пока Выборка.Следующий() > 0 Цикл
 Область = Макет.ПолучитьОбласть("Строка");
 Область.Параметры.Договор = Выборка.Договор;
 Область.Параметры.Сумма = Выборка.Сумма;
 Область.Параметры.Минимум = Выборка.Минимум;
 Область.Параметры.Максимум = Выборка.Максимум;
 Область.Параметры.Среднее = Выборка.Среднее;
 Область.Параметры.Количество = Выборка.Количество;
 ТабДок.Вывести(Область);
КонецЦикла;
КонецПроцедуры
```

В тексте программной процедуры `ПечатьВСводнуюТаблицу` мы воспользовались конструкцией языка запросов — `СГРУППИРОВАТЬ ПО`. При ее использовании можно не просто выбирать записи из таблицы, но еще и группировать их определенным образом.

> ***Примечание***
>
> Под словом *"сгруппировать"* подразумевается компоновка информации по группировочным полям и вычисление агрегатных функций по каждой группе.

Для внесенных в базу данных квитанций заполненный вариант отчета при работе в режиме **1С:Предприятие** приведен (вместе со списком квитанций) на рис. 4.26.



Договор	Сумма	Максимум	Минимум	Среднее	Количество
101-234	2 000	2 000	2 000	2 000	1
102-234	2 020	2 020	2 020	2 020	1
100-234	14 140	10 100	4 040	7 070	2

Рис. 4.26. Сводный отчет по квитанциям и состояние квитанций

Использование параметров в запросах

Во всех ранее рассмотренных запросах данной главы пользователю не требовалось устанавливать какие-либо параметры, а условия для отбора формировались непосредственно в тексте запроса. Если же посмотреть на любой из отчетов, разработанных специалистами фирмы 1С, то увидим, что параметры многочисленных условий пользователь может указывать в диалоге.

Для реализации подобной возможности запрос строится с использованием параметров. В качестве примера рассмотрим разработку, где от пользователя требуется внесение в поле ввода максимальной суммы для отбора интересующих квитанций. В этом случае в полученной выборке будут присутствовать все квитанции на суммы, не превышающие максимальную, указанную пользователем в поле ввода.

Необходимый функционал реализуем для формы списка регистра сведений **Квитанции**. Для начала следует требуемую форму создать, поэтому перейдем на вкладку **Формы** (рис. 4.27) окна редактирования регистра сведений **Квитанции**. Здесь следует щелкнуть по пиктограмме с изображением лупы напротив раздела **Формы списка**.

После несложных (знакомых по предыдущим примерам) действий в диалоге с конструктором форм мы попадаем в окно редактора формы. Здесь требуется перейти на вкладку **Реквизиты** и добавить в форму реквизит **МаксимальнаяСумма** (рис. 4.28), в качестве типа данных которого следует указать численный тип.

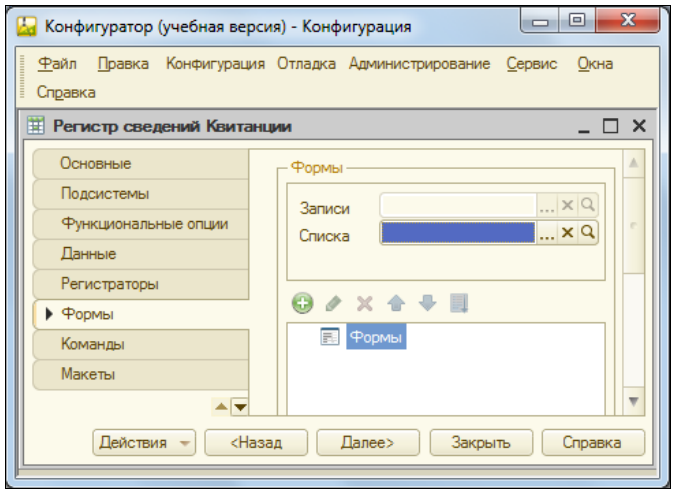


Рис. 4.27. Вкладка **Формы** регистра сведений **Квитанции**

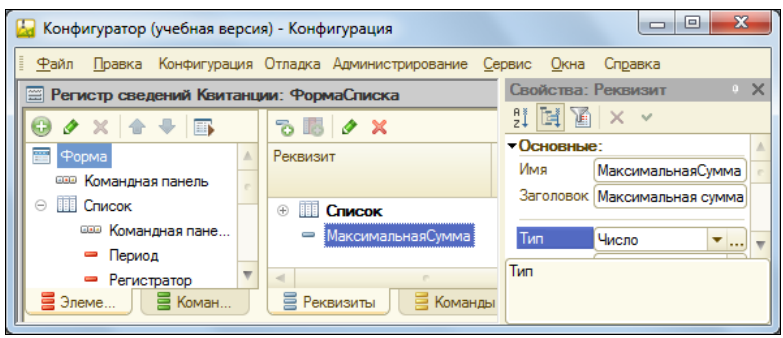


Рис. 4.28. Создание реквизита в форме списка регистра **Квитанции**

После этого необходимо разместить на форме элемент управления (поле) для того, чтобы иметь возможность интерактивно работать с созданным реквизитом **МаксимальнаяСумма** в режиме **1С:Предприятие**. Технически для этого достаточно перетащить мышью пиктограмму созданного реквизита **МаксимальнаяСумма** на пиктограмму **Командная панель** (рис. 4.29).

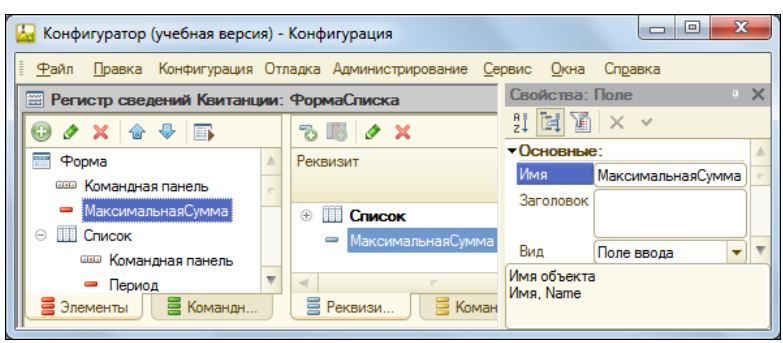


Рис. 4.29. Размещение поля **МаксимальнаяСумма** в форме списка регистра **Квитанции**

Далее требуется создать новую команду для реализации намеченного нами действия (отбора информации). Для этого на вкладке **Команды** воспользуемся знакомой пиктограммой для добавления новой команды **ОтборКвитанций**.

Процедура обработки данной команды представлена в листинге 4.21. Фактически вся основная вычислительная часть программного кода будет выполняться на сервере с помощью процедуры `ОтобратьКвитанции` (листинг 4.22).

Листинг 4.21. Процедура обработки команды ОтборКвитанций

```
&НаКлиенте
Процедура ОтборКвитанций(Команда)
 ОтобратьКвитанции();
КонецПроцедуры
```

Листинг 4.22. Серверная процедура ОтобратьКвитанции

```
&НаСервере
Процедура ОтобратьКвитанции()
Запрос = Новый Запрос;
Запрос.Текст = "ВЫБРАТЬ Договор, СуммаПоКвитанции
| ИЗ РегистрСведений.Квитанции
| ГДЕ СуммаПоКвитанции <= &МаксимальнаяСумма";
Запрос.УстановитьПараметр("МаксимальнаяСумма",
   МаксимальнаяСумма);
Результат = Запрос.Выполнить().Выбрать();
Сообщение=Новый СообщениеПользователю;
Пока Результат.Следующий() > 0 Цикл
 Сообщение.Текст=Строка(Результат.Договор) +
   "  " + Строка(Результат.СуммаПоКвитанции);
 Сообщение.Сообщить();
КонецЦикла;
КонецПроцедуры
```

Обратите внимание на синтаксис указания в параметрическом запросе (листинг 4.22) параметра:

```
Запрос.Текст = "ВЫБРАТЬ  Договор, СуммаПоКвитанции
| ИЗ  РегистрСведений.Квитанции
| ГДЕ СуммаПоКвитанции <= &МаксимальнаяСумма";
```

В серверной процедуре `ОтобратьКвитанции` используется метод `УстановитьПараметр`, который позволяет определить значение параметра для запроса. При этом первый параметр метода представляет собой название параметра запроса, а второй — значение, присваиваемое этому параметру.

В данном случае второй параметр метода берется равным значению в поле ввода `МаксимальнаяСумма` (которое пользователь заполняет перед выполнением запроса).

На рис. 4.30 приведена информация, отобранная в результате выполнения запроса при работе в режиме **1С:Предприятие**.

Примечание

Для установки параметра запроса используется метод `УстановитьПараметр(ИмяПараметра, ЗначениеПараметра)`. Для параметра `ИмяПараметра` тип данных — строка, а параметр `ЗначениеПараметра` имеет произвольный тип данных. С помощью этого метода можно передавать переменные в запрос.

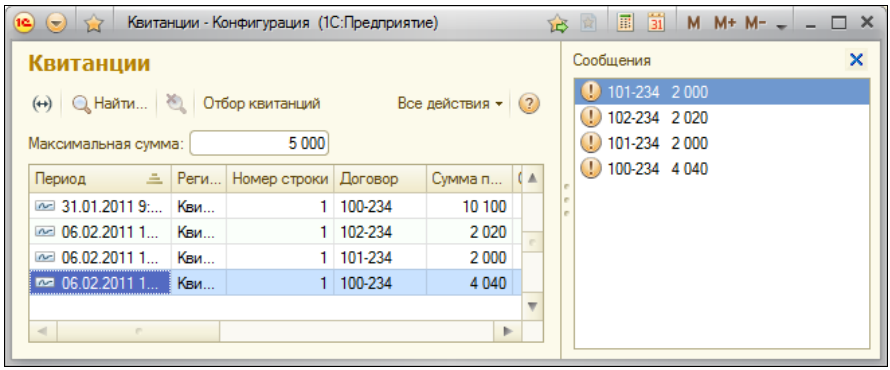


Рис. 4.30. Параметрический запрос по квитанциям и состояние квитанций

Ключевое слово *МЕЖДУ*

В языке запросов имеется вспомогательное ключевое слово `МЕЖДУ` для задания интервалов. Рассмотрим пример, в котором требуется выбрать квитанции из одноименного регистра сведений в интервале дат:

- **ДатаНачала**;
- **ДатаОкончания**.

Для этого в форме списка регистра сведений **Квитанции** на вкладке **Реквизиты** нам понадобится создать два соответствующих реквизита (тип данных — **Дата**).

После этого перейдем на вкладку **Команды** и добавим в форму новую команду **ИспользованиеМежду**, процедура обработки которой приведена в листинге 4.23.

Листинг 4.23. Процедура обработки команды ИспользованиеМежду

```
&НаКлиенте
Процедура ИспользованиеМежду(Команда)
  ПримерСМежду();
КонецПроцедуры
```

Как и в предыдущих примерах, для выполнения вычислений нам понадобится серверная процедура, текст которой представлен в листинге 4.24.

Листинг 4.24. Серверная процедура ПримерСМежду

```
&НаСервере
Процедура ПримерСМежду()
  Запрос = Новый Запрос;
  Запрос.Текст = "ВЫБРАТЬ  Договор,
  | СуммаПоКвитанции КАК Сумма
  | ИЗ  РегистрСведений.Квитанции
  | ГДЕ  РегистрСведений.Квитанции.Период МЕЖДУ  &Дата1 И  &Дата2";
   Запрос.УстановитьПараметр("Дата1",
        ДатаНачала);
   Запрос.УстановитьПараметр("Дата2",
        ДатаОкончания);
   Результат = Запрос.Выполнить().Выбрать();
   Сообщение=Новый СообщениеПользователю;
   Пока Результат.Следующий() > 0 Цикл
      Сообщение.Текст=Строка(Результат.Договор) +
          "   " + Строка(Результат.Сумма);
      Сообщение.Сообщить();
   КонецЦикла;
КонецПроцедуры
```

Обратимся теперь к вкладке **Элементы**, на которой нам потребуется разместить три дополнительных элемента управления: два текстовых поля (для ввода дат) и кнопку для выполнения запроса. Размещение этих элементов следует произвести перемещением пиктограмм созданных реквизитов (**ДатаНачала** и **ДатаОкончания**) и разработанной команды на элемент **Командная панель**. В результате в окне редактора форм произойдут соответствующие изменения (рис. 4.31).

Учитывая, что элементы управления, созданные для предыдущего примера (**ОтборКвитанций** и **МаксимальнаяСумма**), нам здесь не нужны, то лучше их сде-

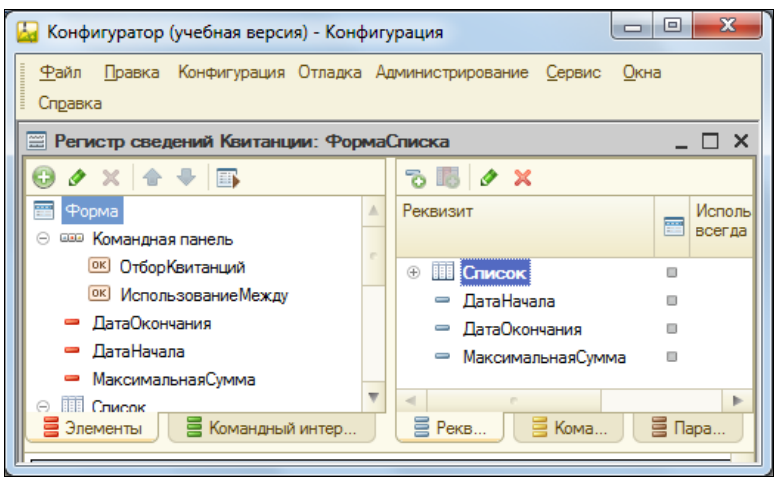


Рис. 4.31. Создание элементов управления в форме регистра **Квитанции**

лать невидимыми. Для этого следует воспользоваться палитрой свойств (рис. 4.32), где необходимо снять флажок у свойства **Видимость**.

Теперь разработка готова. Далее следует перейти в режим **1С:Предприятие** и воспользоваться разработанным функционалом (рис. 4.33).

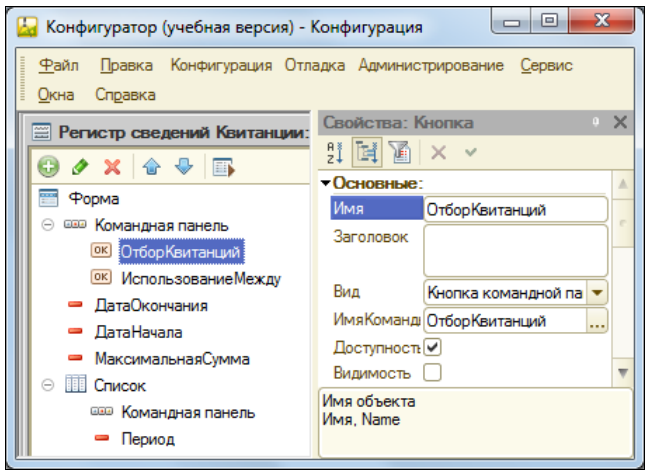


Рис. 4.32. Изменение значения свойства **Видимость** в палитре свойств

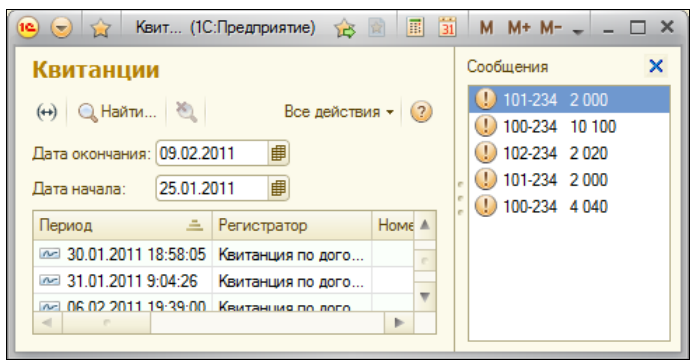


Рис. 4.33. Отбор по квитанциям с использованием ключевого слова МЕЖДУ

Заметим, что в конструкции рассматриваемого запроса (листинг 4.24) используется параметр Период (более конкретно — измерение), который отражает дату и время внесения информации.

Ключевое слово *ИМЕЮЩИЕ*

В языке запросов существует ключевое слово ИМЕЮЩИЕ, позволяющее накладывать условия на значения агрегатных функций. В других конструкциях языка запросов использовать агрегатные функции в условиях нельзя. Однако и в условии отбора со словом ИМЕЮЩИЕ можно использовать агрегатные функции только для полей, по которым осуществляется группировка.

Выполним следующие действия в форме списка регистра **Квитанции**:

1. На вкладке **Элементы** отключим видимость элементов **ОтборКвитанций**, **ИспользованиеМежду**, **ДатаНачала** и **ДатаОкончания**.
2. На вкладке **Команды** добавим в форму списка новую команду — **ИспользованиеИмеющие**.
3. Разработаем процедуру обработки созданной команды **ИспользованиеИмеющие**.
4. Разместим на форме новую кнопку **ИспользованиеИмеющие** для выполнения одноименной команды.

В листинге 4.25 приведена клиентская процедура обработки команды `ИспользованиеИмеющие`, включающая только вызов серверной процедуры `ПримерСИмеющие` (листинг 4.26).

Листинг 4.25. Процедура обработки команды ИспользованиеИмеющие

```
&НаКлиенте
Процедура ИспользованиеИмеющие(Команда)
  ПримерСИмеющие();
КонецПроцедуры
```

Листинг 4.26. Серверная процедура ПримерСИмеющие

```
&НаСервере
Процедура ПримерСИмеющие()
  Запрос = Новый Запрос;
  Запрос.Текст = "ВЫБРАТЬ  Договор,
  | СУММА(СуммаПоКвитанции)  КАК Сумма
  | ИЗ  РегистрСведений.Квитанции
  | ГДЕ Оплачено=Истина
  | СГРУППИРОВАТЬ ПО Договор
  | ИМЕЮЩИЕ
  | СУММА(СуммаПоКвитанции) <= &МаксимальнаяСумма";
   Запрос.УстановитьПараметр("МаксимальнаяСумма",
        МаксимальнаяСумма);
  Результат = Запрос.Выполнить().Выбрать();
  Сообщение=Новый СообщениеПользователю;
  Пока Результат.Следующий() > 0 Цикл
     Сообщение.Текст=Строка(Результат.Договор) +
          "    " + Строка(Результат.Сумма);
     Сообщение.Сообщить();
  КонецЦикла;
КонецПроцедуры
```

Цель рассматриваемого запроса — отобрать договоры, для которых итоговая сумма по оплаченным квитанциям не превышает значения **МаксимальнаяСумма** (его мы вводим в качестве параметра с клавиатуры).

Перейдем в режим **1С:Предприятие** и воспользуемся разработанным функционалом (рис. 4.34)

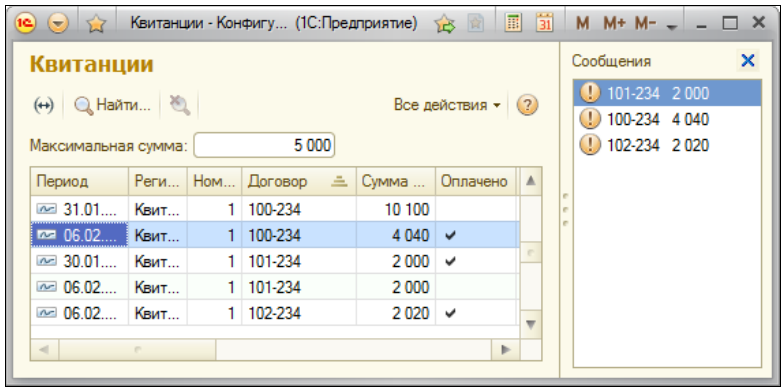


Рис. 4.34. Результат выполнения запроса с использованием ключевого слова ИМЕЮЩИЕ

Функция расчета итогов в запросах

Формирование итоговой информации является одной из наиболее востребованных функций в любой информационной системе. Язык запросов системы 1С:Предприятие 8.2 включает в себя очень мощные возможности расчета итогов. В этом случае пользователь может включить в результат запроса дополнительные строки, содержащие общие и промежуточные итоги по заданным полям и группировкам.

Создадим новую команду `ОтобратьИнформацию`. В листинге 4.27 приведена клиентская процедура обработки данной команды. Также нам потребуется серверная процедура (листинг 4.28), реализующая выполнение запроса, который позволяет получить общие итоги по выписанным квитанциям.

Листинг 4.27. Процедура обработки команды ОтобратьИнформацию

```
&НаКлиенте
Процедура ОтобратьИнформацию(Команда)
  ОтборИнформации();
КонецПроцедуры
```

Листинг 4.28. Серверная процедура ОтборИнформации

```
&НаСервереБезКонтекста
Процедура ОтборИнформации ()
Запрос = Новый Запрос;
```

```
Запрос.Текст = "ВЫБРАТЬ Договор,
| СуммаПоКвитанции КАК Сумма, Период
| ИЗ РегистрСведений.Квитанции
| ИТОГИ СУММА(Сумма) ПО Общие";
Результат = Запрос.Выполнить().Выбрать();
Сообщение=Новый СообщениеПользователю;
Пока Результат.Следующий() > 0 Цикл
Сообщение.Текст = Строка(Результат.Период) +
    "  " + Строка(Результат.Договор) +
    "  " + Строка(Результат.Сумма) + " руб.";
        Сообщение.Сообщить();
КонецЦикла;
КонецПроцедуры
```

Далее создадим кнопку в форме списка регистра сведений **Квитанции** для выполнения команды `ОтобратьИнформацию`. После этого перейдем в режим **1С:Предприятие**, откроем форму списка регистра **Квитанции** и щелкнем по кнопке **Отобрать информацию**. В результате в окне сообщений мы получим результат выполнения запроса (рис. 4.35). Как видно, в результате выполнения запроса появляется дополнительная строка с итоговой суммой.

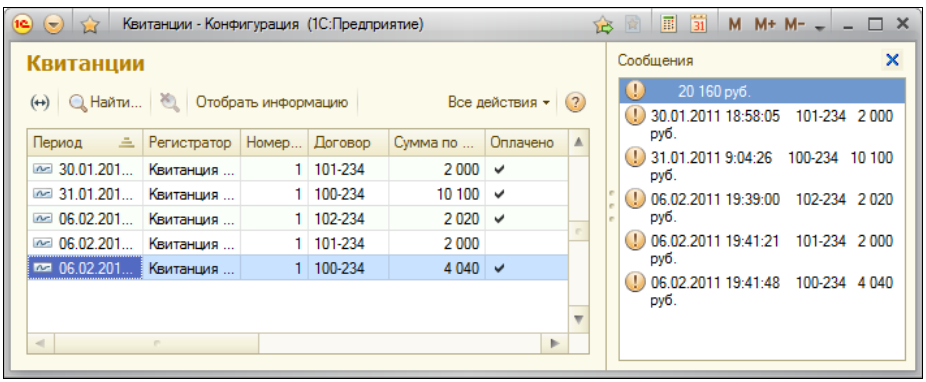


Рис. 4.35. Результат выполнения запроса с функцией расчета итогов

При обходе результата запроса итоговые строки можно отличить от обычных с помощью метода `ТипЗаписи`. Используем его для небольшой модификации выполненной разработки, а именно вместе с итоговой суммой выведем еще и поясняющее сообщение. Учитывая незначительность изменений относительно предыдущего примера, не будем писать новую команду, а лишь поменяем серверную процедуру (листинг 4.29).

Листинг 4.29. Серверная процедура ОтборИнформации (второй вариант)

```
&НаСервереБезКонтекста
Процедура ОтборИнформации ()
  Запрос = Новый Запрос;
```

```
  Запрос.Текст = "ВЫБРАТЬ  Договор,
  | СуммаПоКвитанции КАК Сумма, Период
  | ИЗ  РегистрСведений.Квитанции
  | ИТОГИ СУММА(Сумма) ПО Общие";
  Результат = Запрос.Выполнить().Выбрать();
  Сообщение=Новый СообщениеПользователю;
  Пока Результат.Следующий() > 0 Цикл
    Если Результат.ТипЗаписи() = ТипЗаписиЗапроса.ОбщийИтог Тогда
         Сообщение.Текст = "Суммарный итог "+ Строка(Результат.Сумма);
               Сообщение.Сообщить();
    Иначе
         Сообщение.Текст = Строка(Результат.Период) +
            "     " + Строка(Результат.Договор) +
            "    " + Строка(Результат.Сумма) + " руб.";
         Сообщение.Сообщить();
     КонецЕсли;
  КонецЦикла;
КонецПроцедуры
```

Несложно получить и итоги по группировкам. В этом случае вычисляются значения агрегатных функций по выборкам с одинаковыми значениями полей, по которым производится группировка.

Для демонстрации группировки создадим новую команду `ОтборСГруппировкой`. В листинге 4.30 приведена клиентская процедура обработки данной команды. Также нам потребуется серверная процедура `ОтобратьСГруппировкой` (листинг. 4.31), использующая запрос для получения итогов по группировкам.

Листинг 4.30. Процедура обработки команды ОтборСГруппировкой

```
&НаКлиенте
Процедура ОтборСГруппировкой(Команда)
  ОтобратьСГруппировкой();
КонецПроцедуры
```

Листинг 4.31. Серверная процедура ОтобратьСГруппировкой

```
&НаСервереБезКонтекста
Процедура ОтобратьСГруппировкой()
 Запрос = Новый Запрос;
 Запрос.Текст = "ВЫБРАТЬ Договор,
 | СуммаПоКвитанции КАК Сумма, Период
 | ИЗ РегистрСведений.Квитанции
 | ГДЕ Оплачено=Истина
 | ИТОГИ СУММА(Сумма) ПО Договор";
```

```
  Результат = Запрос.Выполнить().Выбрать();
        Сообщение=Новый СообщениеПользователю;
  Пока Результат.Следующий() > 0 Цикл
    Сообщение.Текст=Строка(Результат.Период) +
     " "+Строка(Результат.Договор) +
     "   " + Строка(Результат.Сумма);
    Сообщение.Сообщить();
  КонецЦикла;
КонецПроцедуры
```

Создадим на форме кнопку **ОтборСГруппировкой** для выполнения разработанной команды. Далее следует перейти в режим **1С:Предприятие** и выполнить запрос (рис. 4.36).

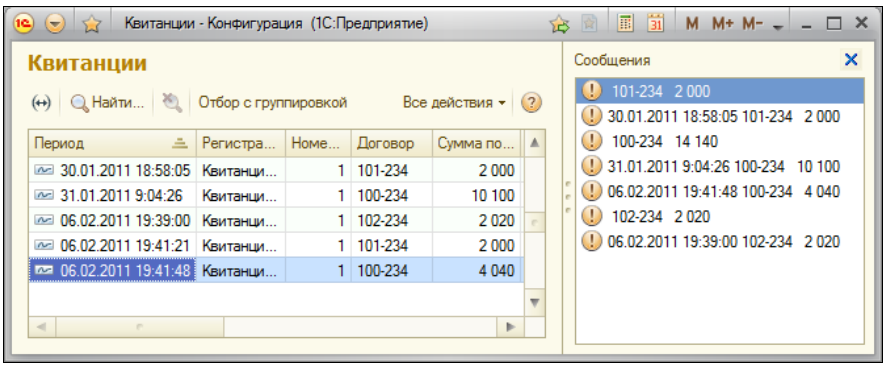


Рис. 4.36. Отчет с включением итогов по группировкам

Запрос с итогами по группировкам отличается от обычной группировки (свертки) с использованием предложения `СГРУППИРОВАТЬ ПО` с агрегатными функциями. В последнем случае в результат запроса не включаются исходные записи, а остаются только итоговые строки.

Объединение результатов нескольких запросов

В языке запросов имеется возможность объединять результаты нескольких запросов, при этом записи, полученные с помощью каждого из объединяемых запросов, будут собраны в один результат. При объединении каждый запрос собирает данные независимо, а такие операции, как упорядочивание результатов и расчет итогов, выполняются уже над результатом объединения запросов.

Поля результата запроса будут называться так, как описано в списке полей выборки первого из объединяемых запросов. Поля выборки остальных запросов сопоставляются с полями результата в соответствии с порядком их следования в списке полей выборки. Объединяемые запросы должны иметь одинаковое количество полей в списке полей выборки.

Рассмотрим ситуацию, когда кроме оплаты наличным путем через квитанции возможно еще и безналичное перечисление. В этом случае нам понадобится еще один регистр сведений — **БезналичнаяОплата**. Таким образом, в конфигураторе создадим новый объект конфигурации. Это будет независимый периодический (с периодом — одна секунда) регистр сведений. На вкладке **Данные** окна редактирования объекта конфигурации сформируем одно измерение **Договор** (тип данных — **СправочникСсылка.Договоры**) и один ресурс **ПеречисленнаяСумма** (рис. 4.37).

После этого перейдем в режим **1С:Предприятие** и внесем информацию об одной безналичной оплате (рис. 4.38).

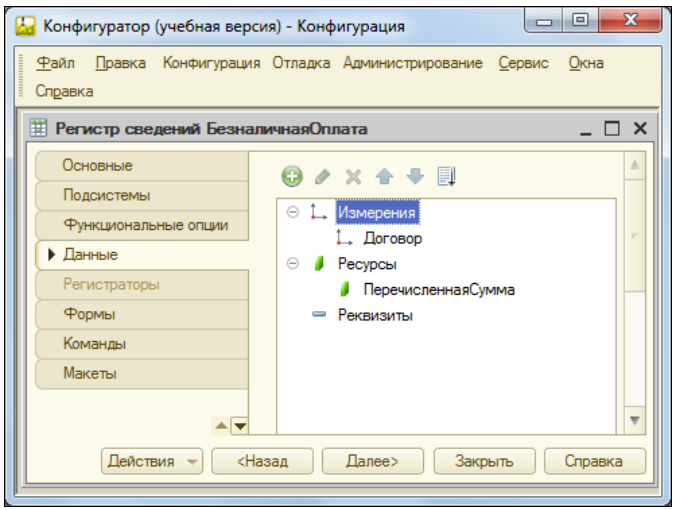


Рис. 4.37. Окно редактирования регистра **БезналичнаяОплата**

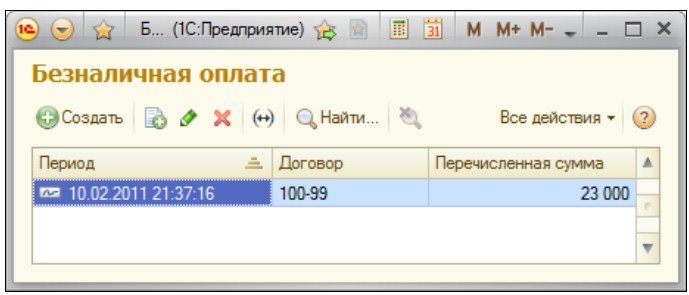


Рис. 4.38. Форма списка регистра **БезналичнаяОплата**

Теперь мы должны построить запрос для извлечения оплаченных квитанций и банковских перечислений в задаваемом пользователем интервале дат.

Реализуем выполнение запроса в форме списка справочника "Договоры". Для этого в редакторе формы добавим новую команду **ПросмотрОплат** в имеющуюся форму списка справочника "Договоры". В листинге 4.32 приведена процедура обработки данной команды.

Листинг 4.32. Процедура обработки команды ПросмотрОплат

```
&НаКлиенте
Процедура ПросмотрОплат(Команда)
  НаличнаяИБезналичная();
КонецПроцедуры
```

Для выполнения команды ПросмотрОплат нам потребуется серверная процедура НаличнаяИБезналичная (листинг 4.33).

При объединении запросов ключевым является слово ОБЪЕДИНИТЬ. Перед ним должен быть расположен один запрос, а после него — другой.

Листинг 4.33. Серверная процедура НаличнаяИБезналичная

```
&НаСервере
Процедура НаличнаяИБезналичная()
  Запрос = Новый Запрос;
  Запрос.Текст = "ВЫБРАТЬ  Договор,
  | СуммаПоКвитанции КАК Сумма,Период
  | ИЗ  РегистрСведений.Квитанции
  | ГДЕ  (Период МЕЖДУ &Дата1 И &Дата2 ) И Оплачено=Истина
  |   ОБЪЕДИНИТЬ
  | ВЫБРАТЬ  Договор, ПеречисленнаяСумма КАК Сумма, Период
  | ИЗ  РегистрСведений.БезналичнаяОплата
  | ГДЕ  Период МЕЖДУ &Дата1 И &Дата2";
  Запрос.УстановитьПараметр("Дата1",
         ДатаНачала);
  Запрос.УстановитьПараметр("Дата2",
         ДатаОкончания);
  Результат = Запрос.Выполнить().Выбрать();
  Сообщение=Новый СообщениеПользователю;
  Пока Результат.Следующий() > 0 Цикл
    Сообщение.Текст=Строка(Результат.Период)+ "    " +
        Строка(Результат.Договор) +
        "    " + Строка(Результат.Сумма) + " руб.";
    Сообщение.Сообщить();
 КонецЦикла;
КонецПроцедуры
```

Нам потребуются два поля для ввода данных (**ДатаНачала** и **ДатаОкончания**), а также кнопка для выполнения разработанной команды. На рис. 4.39 представлена информация, полученная из регистров **Квитанции** и **БезналичнаяОплата**.

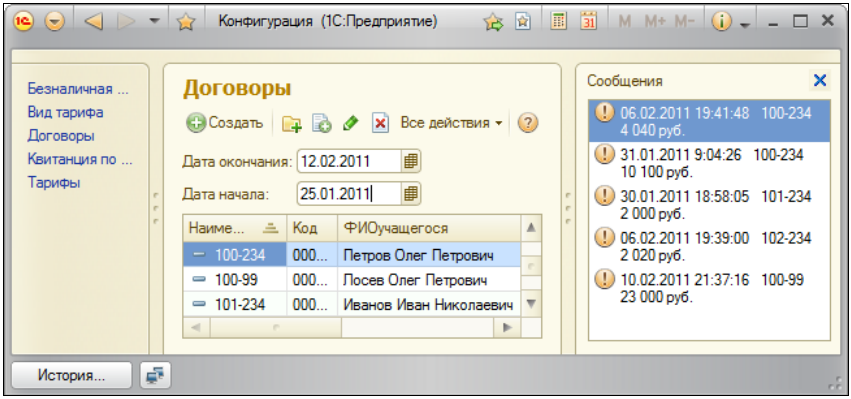


Рис. 4.39. Отбор по оплатам за обучение

Более сложные конструкции языка запросов

В этом разделе мы познакомимся с рядом новых конструкций для построения запросов. Объединяющим моментом всех приведенных далее примеров является то, что они связаны с информационной базой из *главы 1*. В целом для формирования запросов нам будет достаточно ранее созданных объектов конфигурации, однако для непосредственной реализации запросов мы воспользуемся объектом конфигурации *обработка*. Визуализация результатов потребует также создания нескольких интерфейсных форм с необходимыми элементами управления.

Итак, если вы выполняете описанные в книге действия на компьютере, то лучше всего в режиме **Конфигуратор** открыть созданную информационную базу, разработанную в *главе 1*.

Отбор специалистов по указанной должности

Приступим к первому запросу, связанному с разработанной конфигурацией из *главы 1*. В ней мы создали объект конфигурации — справочник "Специалисты", содержащий список преподавателей нашего учебного центра. В этом справочнике имеется табличная часть **МестаРаботы**, где перечислены организации, в которых дополнительно работает каждый сотрудник нашего центра. Вместе с указанием организации, в каждой строке табличной части справочника определяется еще и должность специалиста в этой организации.

Будем считать, что цель запроса заключается в том, чтобы отобрать специалистов (с указанием места работы), имеющих выбранную в элементе интерфейса (поле ввода) должность. Подобная задача вполне может возникнуть и в другой области деятельности.

Итак, для выполнения запроса предварительно создадим обработку (объект конфигурации нового вида). Для этого в дереве объектов конфигурации следует обра-

титься к элементу **Обработки**. Здесь с помощью правой кнопки мыши необходимо вызвать контекстное меню, где воспользоваться пунктом **Добавить**. В результате на экране откроется окно редактирования объекта конфигурации (рис. 4.40). Здесь на вкладке **Данные** требуется указать имя обработки — **Обработка1**. Фактически основные технические действия по разработке развернутся с формой данного объекта конфигурации, поэтому перейдем на вкладку **Формы** (рис. 4.41).

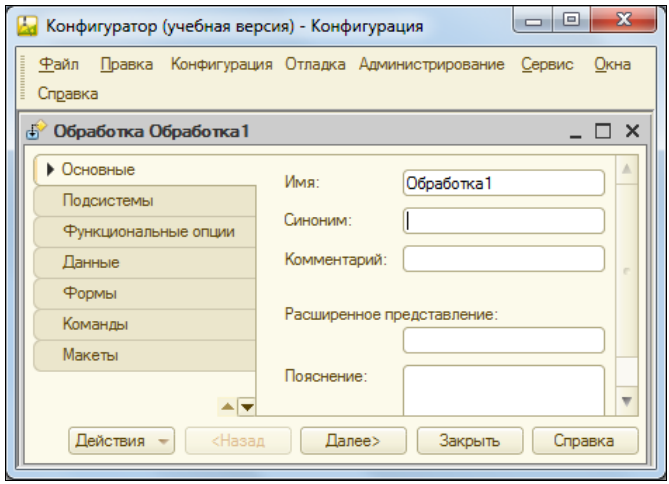


Рис. 4.40. Окно редактирования объекта **Обработка1**

На вкладке **Формы** следует создать форму обработки. Для этого щелкнем правой кнопкой мыши по пиктограмме с изображением лупы во фрейме **Форма обработки** (см. рис. 4.41). В результате мы перейдем к работе с конструктором формы, и в первом окне диалога с конструктором следует сделать установку (с помощью переключателя) для создания произвольной формы. После этого с помощью кнопки **Готово** перейдем в окно редактора форм, где будем наполнять форму обработки необходимыми нам деталями.

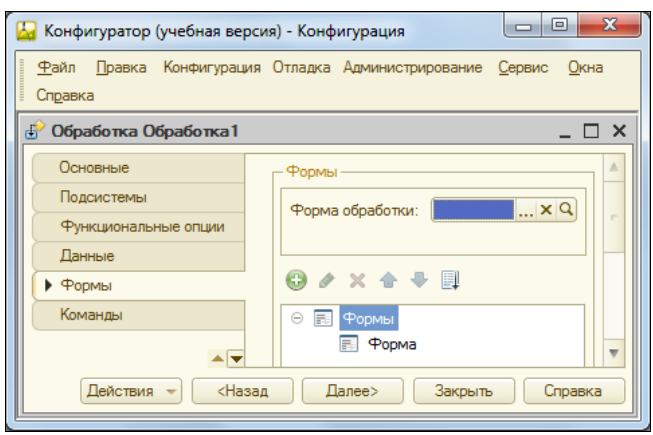


Рис. 4.41. Вкладка **Формы** окна редактирования объекта **Обработка1**

Добавим в форму реквизит **Должность** (рис. 4.42), в качестве типа данных которого укажем **СправочникСсылка.Должности**.

После этого потребуется сформировать на форме поле ввода для ввода должности. Технически для этого проще всего мышью перетащить созданный реквизит **Должность** на командную панель (расположена на вкладке **Элементы**).

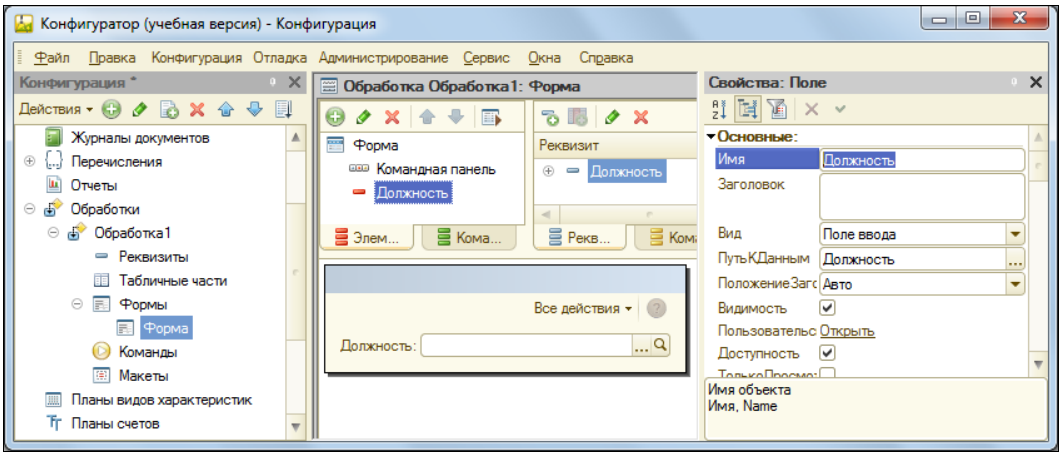


Рис. 4.42. Форма объекта **Обработка1** в окне редактора форм

Далее добавим в форму обработки новую команду `ПолучитьСведения`, программный код которой приведен в листинге 4.34. Это традиционная клиентская процедура, в которой производится серверный вызов `ОтборПоДолжности` (листинг 4.35).

Листинг 4.34. Процедура обработки команды ПолучитьСведения

```
&НаКлиенте
Процедура ПолучитьСведения(Команда)
ОтборПоДолжности();
КонецПроцедуры
```

Листинг 4.35. Серверная процедура ОтборПоДолжности

```
&НаСервере
Процедура ОтборПоДолжности ()
Запрос = Новый Запрос;
Запрос.Текст = "ВЫБРАТЬ Ссылка.Наименование
| КАК Наименование,
| Организация, Должность
| ИЗ Справочник.Специалисты.МестаРаботы
| ГДЕ Должность = &Должность";
Запрос.УстановитьПараметр("Должность",
     Должность);
Результат = Запрос.Выполнить().Выбрать();
Сообщение=Новый СообщениеПользователю;
```

```
Пока Результат.Следующий() > 0 Цикл
  Сообщение.Текст=Строка(Результат.Наименование) +
   "  " + Строка(Результат.Организация) +
   "  " + Строка(Результат.Должность);
  Сообщение.Сообщить();
 КонецЦикла;
КонецПроцедуры
```

При осуществлении запроса к табличной части справочника обращение к обычным (вне таблицы) реквизитам производится через поле **Ссылка**. Хотя запрос (листинг 4.35) обращается к табличной части **МестаРаботы** справочника "Специалисты", основная информация о специалисте заключена в реквизите **Наименование**, который располагается вне табличной части. И для получения фамилии специалиста в тексте запроса используется конструкция **Ссылка.Наименование**.

После этого требуется разместить на форме обработки кнопку **ПолучитьСведения** для выполнения разработанной одноименной команды. Это следует выполнить с помощью перетаскивания мышью пиктограммы созданной команды на элемент **Командная панель**.

Учитывая, что в конфигурации, разработанной в *главе 1*, мы использовали деление на две подсистемы, то следует отнести созданный объект конфигурации (**Обработка1**) к одной из подсистем. Для определенности выберем подсистему — **Начальник учебного отдела**.

Теперь все необходимые действия в конфигураторе выполнены и можно перейти в режим **1С:Предприятие**.

Действия пользователя заключаются в предварительном выборе в поле ввода интересующей должности. После этого нажатие кнопки **Получить сведения** приведет к заполнению списка интересующей нас информацией о сотрудниках. На рис. 4.43 показан результат отбора сотрудников по должности.

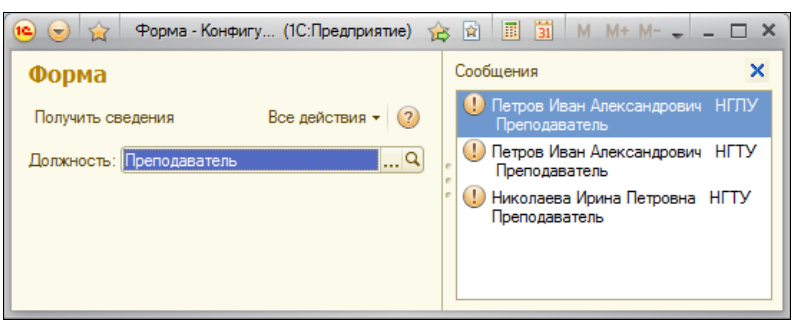


Рис. 4.43. Отбор специалистов по должности

В запросе, приведенном в листинге 4.35, в выборку отобранных данных один и тот же специалист может попасть два и более раз, если он работает в нескольких организациях на одинаковых должностях. Именно такая ситуация отражена на

рис. 4.43: Иван Александрович Петров работает преподавателем в двух организациях (**НГТУ** и **НГЛУ**).

Рассмотрим теперь организацию запроса, при котором в окно сообщений будет выведен перечень только различных специалистов (без повторов), имеющих указанную пользователем должность.

Реализуем данный запрос с использованием новой обработки — **Обработка2**. Не будем описывать технические шаги, которые аналогичны выполненным при разработке объекта **Обработка1**.

В листинге 4.36 приведена процедура обработки команды `ПолучитьСведенияБезПовтора`. Необходимая серверная процедура, в которой фактически производится выполнение запроса, представлена в листинге 4.37.

Листинг 4.36. Процедура обработки команды ПолучитьСведенияБезПовтора

```
&НаКлиенте
Процедура ПолучитьСведенияБезПовтора(Команда)
   ОтборПоДолжностиБезПовтора();
КонецПроцедуры
```

Листинг 4.37. Серверная процедура ОтборПоДолжностиБезПовтора

```
&НаСервере
Процедура ОтборПоДолжностиБезПовтора ()
   Запрос = Новый Запрос;
   Запрос.Текст = "ВЫБРАТЬ РАЗЛИЧНЫЕ Ссылка.Наименование
| КАК Наименование,
|  Должность
| ИЗ  Справочник.Специалисты.МестаРаботы
| ГДЕ Должность = &Должность";
Запрос.УстановитьПараметр("Должность",
        Должность);
   Результат = Запрос.Выполнить().Выбрать();
   Сообщение=Новый СообщениеПользователю;
   Пока Результат.Следующий() > 0 Цикл
      Сообщение.Текст=Строка(Результат.Наименование) +
        "   " +          "   " + Строка(Результат.Должность);
        Сообщение.Сообщить();
   КонецЦикла;
КонецПроцедуры
```

Результат выполнения запроса для имеющихся в информационной базе данных приведен на рис. 4.44.

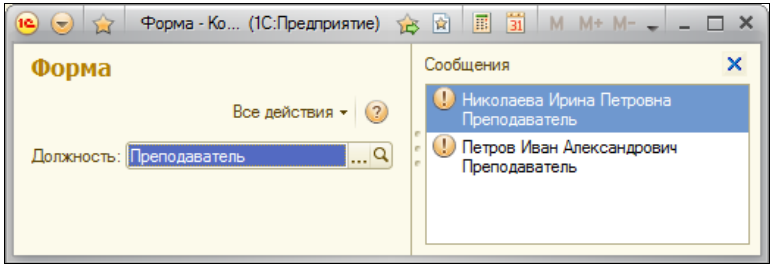


Рис. 4.44. Результат отбора специалистов с указанной должностью

Отбор курсов преподавателя

При работе с рассматриваемой информационной базой нам может понадобиться получить в выборке курсы с указанием их стоимости, которые ведет конкретный преподаватель (фамилия преподавателя аналогично должности должна выбираться из списка значений).

Создадим новую обработку (выберем имя — **Обработка3**). В окне редактирования объекта конфигурации перейдем на вкладку **Формы** и разработаем форму обработки.

В окне редактора форм добавим в форму реквизит **Преподаватель** (тип данных — **СправочникСсылка.Специалисты**). Далее на вкладке **Команды** добавим новую команду, которую назовем **ОтборИнформации**. В листинге 4.38 приведена процедура обработки данной команды. Основные вычисления производятся в серверной процедуре, представленной в листинге 4.39.

Листинг 4.38. Процедура обработки команды ОтборИнформации

```
&НаКлиенте
Процедура ОтборИнформации(Команда)
    ОтборДанных();
КонецПроцедуры
```

Листинг 4.39. Серверная процедура ОтборДанных

```
&НаСервере
Процедура ОтборДанных ()
  Запрос = Новый Запрос;
  Запрос.Текст = "ВЫБРАТЬ Наименование,
  | Стоимость, Преподаватель
  | ИЗ  Справочник.Курсы
  | ГДЕ Преподаватель = &Преподаватель";
  Запрос.УстановитьПараметр("Преподаватель",
        Преподаватель);
  Результат = Запрос.Выполнить().Выбрать();
  Сообщение=Новый СообщениеПользователю;
```

```
  Пока Результат.Следующий() > 0 Цикл
     Сообщение.Текст=Строка(Результат.Наименование) +
        "    " + Строка(Результат.Преподаватель) +
        "    " + Строка(Результат.Стоимость);
        Сообщение.Сообщить();
  КонецЦикла;
КонецПроцедуры
```

Далее требуется создать кнопку на форме для выполнения разработанной команды (рис. 4.45).

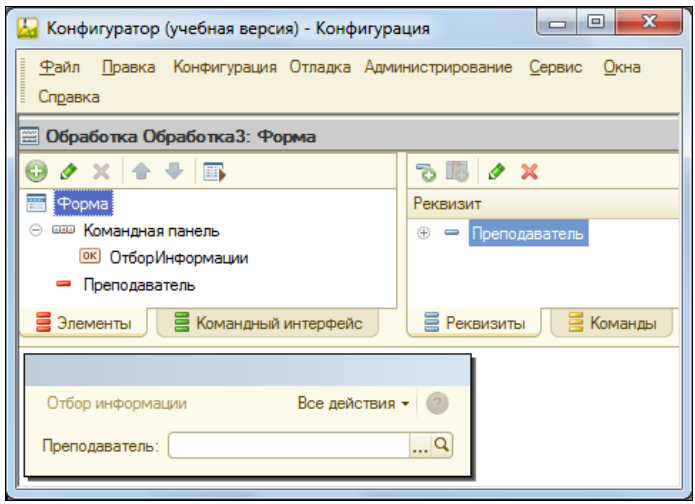


Рис. 4.45. Форма объекта **Обработка3**

Осталось отнести разработанный объект конфигурации (**Обработка3**) к одной из подсистем (для определенности выберем **Начальник учебного отдела**). Теперь можно перейти в режим **1С:Предприятие** и выполнить разработанный запрос. Этот результат продемонстрирован на рис. 4.46.

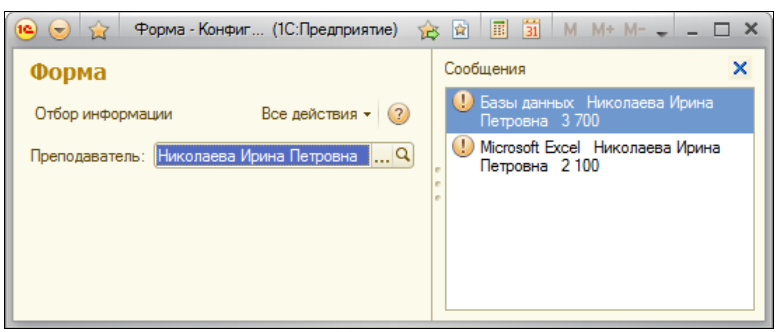


Рис. 4.46. Отбор дисциплин указанного преподавателя

Соединение таблиц при организации запроса

До настоящего времени во всех использованных запросах выборка извлекалась из одной таблицы. Важной особенностью языка запросов системы 1С является возможность обращения сразу к нескольким таблицам. При этом их можно соединять определенным образом. Например, при работе может потребоваться выбрать названия всех курсов с указанием стоимости, преподавателя и рейтинга, который имеет данный преподаватель. В этом случае часть информации располагается в таблице справочника "Курсы", а другая часть в таблице справочника "Специалисты".

В конструкции подобного запроса (листинг 4.40) используется ключевое слово СОЕДИНЕНИЕ, позволяющее отобрать из обоих справочников только те записи, в которых преподаватели совпадают.

Листинг 4.40. Текст запроса с использованием соединения двух таблиц

```
ВЫБРАТЬ СпрКурсы.Наименование КАК Курс,
        СпрКурсы.Преподаватель  КАК Преподаватель,
        СпрКурсы.Стоимость КАК Стоимость,
        СпрСпециалисты.Рейтинг КАК РейтингПреподавателя
        ИЗ  Справочник.Курсы КАК СпрКурсы
        СОЕДИНЕНИЕ
        Справочник.Специалисты КАК СпрСпециалисты
        ПО СпрКурсы.Преподаватель= СпрСпециалисты.Ссылка
```

Для практической реализации данного запроса создадим новую обработку (выберем имя — **Обработка4**). Добавим в объект конфигурации **Обработка4** форму, в которой создадим команду СоединениеТаблиц. Процедура для выполнения данной команды приведена в листинге 4.41. Как и ранее, вычислительную часть программного кода вынесем в серверную процедуру ОтобратьИнформацию (листинг 4.42).

Листинг 4.41. Процедура обработки команды СоединениеТаблиц

```
&НаКлиенте
Процедура СоединениеТаблиц(Команда)
ОтобратьИнформацию();
КонецПроцедуры
```

Листинг 4.42. Серверная процедура ОтобратьИнформацию

```
&НаСервере
Процедура ОтобратьИнформацию ()
  Запрос = Новый Запрос;
  Запрос.Текст = "ВЫБРАТЬ СпрКурсы.Наименование КАК Курс,
  | СпрКурсы.Преподаватель  КАК Преподаватель,
  | СпрКурсы.Стоимость КАК СтоимостьКурса,
```

```
  | СпрСпециалисты.Рейтинг КАК РейтингПреподавателя
  | ИЗ  Справочник.Курсы КАК СпрКурсы
  | СОЕДИНЕНИЕ
  | Справочник.Специалисты КАК СпрСпециалисты
  | ПО СпрКурсы.Преподаватель=СпрСпециалисты.Ссылка";
  Результат = Запрос.Выполнить().Выбрать();
  Сообщение=Новый СообщениеПользователю;
  Пока Результат.Следующий() > 0 Цикл
      Сообщение.Текст=Строка(Результат.Курс) +
        "   " + Строка(Результат.Преподаватель) + "   "
        + Строка(Результат.СтоимостьКурса) + "руб.  Рейтинг: "
        + Строка(Результат.РейтингПреподавателя);
      Сообщение.Сообщить();
 КонецЦикла;
КонецПроцедуры
```

Далее необходимо создать на форме обработки кнопку, с помощью которой разработанную команду можно выполнить в режиме **1С:Предприятие**. Это реализуется перемещением пиктограммы **СоединениеТаблиц** на пиктограмму **Командная панель**.

Теперь следует определить объект **Обработка4** к определенной подсистеме (укажем — **НачальникУчебногоОтдела**) и перейти в режим **1С:Предприятие**. На рис. 4.47 представлен результат выполнения запроса: мы видим в каждой строке интересующую нас информацию по каждому курсу.

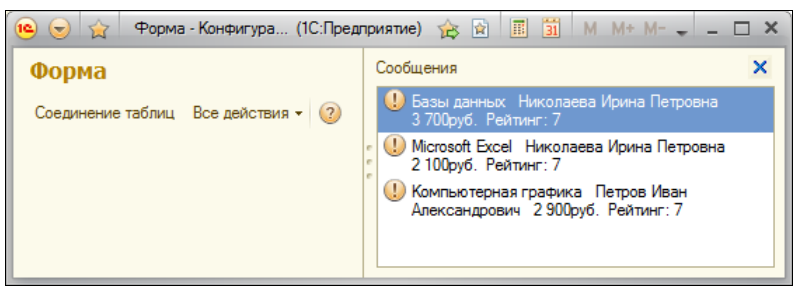
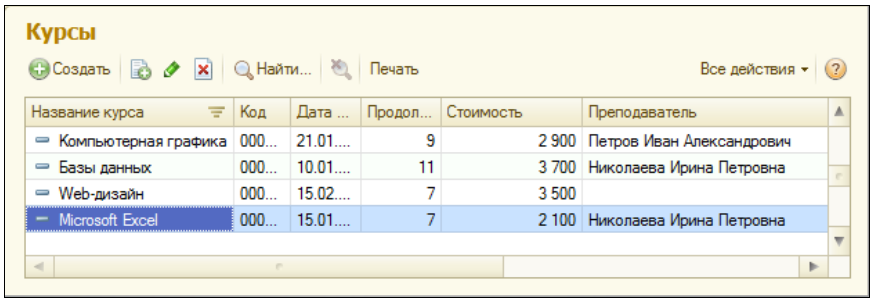


Рис. 4.47. Результат внутреннего соединения двух таблиц в запросе

Левое внешнее соединение

При выполнении предыдущего запроса в результат отбора может попасть только часть интересующей нас сводной информации. Этот недостаток связан с тем, что в результат отбора не попадают курсы, для которых не указан преподаватель. Например, в справочнике "Курсы" создадим курс (**Web-дизайн**), в котором не укажем преподавателя (рис. 4.48). После этого выполним команды из предыдущего раздела. В результате информация в окне сообщений (см. рис. 4.47) не изменится, т. е. данного курса мы не увидим.

Соединение с таким поведением относится к классу *внутренних*. В этом случае в результат запроса попадают комбинации записей из обеих исходных таблиц, которые соответствуют указанному условию.



Название курса	Код	Дата ...	Продол...	Стоимость	Преподаватель
Компьютерная графика	000...	21.01....	9	2 900	Петров Иван Александрович
Базы данных	000...	10.01....	11	3 700	Николаева Ирина Петровна
Web-дизайн	000...	15.02....	7	3 500	
Microsoft Excel	000...	15.01....	7	2 100	Николаева Ирина Петровна

Рис. 4.48. Результат внутреннего соединения двух таблиц в запросе

В языке запросов системы 1С:Предприятие 8.2 существует возможность еще и *внешних* соединений, которые могут быть *левыми*, *правыми* и *полными*.

Конструкция `ЛЕВОЕ [ВНЕШНЕЕ] СОЕДИНЕНИЕ` означает, что в результат запроса надо включить комбинации записей из обеих исходных таблиц, которые соответствуют указанному условию. Но, в отличие от внутреннего соединения, в результат запроса попадают и записи из первого источника (указанного *слева* от слова `СОЕДИНЕНИЕ`), для которых не найдено соответствующих условию записей из второго источника.

Пусть, например, необходимо выбрать названия всех курсов с указанием стоимости, преподавателя и рейтинга, который имеет данный преподаватель, однако при этом в результирующую таблицу необходимо включить курсы, для которых не указан преподаватель.

Выполним запрос с использованием левого соединения в той же обработке (**Обработка4**). Добавим в имеющуюся форму новую команду `ЛевоеСоединение`, программный код которой представлен в листинге 4.43.

Листинг 4.43. Процедура обработки команды ЛевоеСоединение

```
&НаКлиенте
Процедура ЛевоеСоединение (Команда)
ОтобратьИнформациюЛевоеСоединение();
КонецПроцедуры
```

Вычислительные действия произведем в серверной процедуре `ОтобратьИнформациюЛевоеСоединение` (листинг 4.44).

Листинг 4.44. Серверная процедура с использованием левого соединения

```
&НаСервере
Процедура ОтобратьИнформациюЛевоеСоединение ()
  Запрос = Новый Запрос;
```

```
  Запрос.Текст = "ВЫБРАТЬ СпрКурсы.Наименование КАК Курс,
  | СпрКурсы.Преподаватель  КАК Преподаватель,
  | СпрКурсы.Стоимость КАК СтоимостьКурса,
  | СпрСпециалисты.Рейтинг КАК РейтинтПреподавателя
  | ИЗ  Справочник.Курсы КАК СпрКурсы
  | ЛЕВОЕ СОЕДИНЕНИЕ
  | Справочник.Специалисты КАК СпрСпециалисты
  | ПО СпрКурсы.Преподаватель=СпрСпециалисты.Ссылка";
  Результат = Запрос.Выполнить().Выбрать();
  Сообщение=Новый СообщениеПользователю;
  Пока Результат.Следующий() > 0 Цикл
      Сообщение.Текст=Строка(Результат.Курс) +
        "    " + Строка(Результат.Преподаватель) + "    "
        + Строка(Результат.СтоимостьКурса) + "руб.  Рейтинг: "
        + Строка(Результат.РейтинтПреподавателя);
      Сообщение.Сообщить();
 КонецЦикла;
КонецПроцедуры
```

После этого создадим новую кнопку на форме для выполнения разработанной команды `ЛевоеСоединение`. На рис. 4.49 показан результат выполнения запроса с отражением курса, для которого не указан преподаватель.

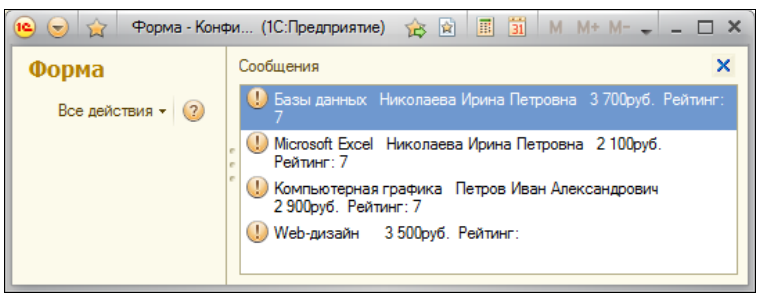


Рис. 4.49. Результат левого внешнего соединения при построении запроса

Таким образом, в результат запроса с использованием левого соединения включены все записи из первого источника. Они соединены с записями из второго источника при выполнении указанного условия. Строки результата запроса, для которых не найдено соответствующих условию записей из второго источника, содержат специальное значение **NULL** в полях, формируемых на основании записей из этого источника.

Примечание

Значения **NULL** являются специальными маркерами, обозначающими неуказанные (отсутствующие) значения или значения, не имеющие смысла.

Правое внешнее соединение

Конструкция ПРАВОЕ [ВНЕШНЕЕ] СОЕДИНЕНИЕ означает, что в результат запроса надо включить комбинации записей из обеих исходных таблиц, которые соответствуют указанному условию. Кроме того, в результат запроса надо включить еще и записи из второго источника (указанного справа от слова СОЕДИНЕНИЕ), для которых не найдено соответствующих условию записей из первого источника.

Таким образом, в результат запроса будут включены все записи из второго источника; они будут соединены с записями из первого источника при выполнении указанного условия. Строки результата запроса, для которых не найдено соответствующих условию записей из первого источника, будут содержать значение **NULL** в полях, формируемых на основании записей из этого источника. Правое внешнее соединение полностью аналогично левому, за исключением того, что таблицы поменялись местами.

Пусть, например, необходимо выбрать названия всех курсов с указанием их стоимости, преподавателя и рейтинга, который имеет данный преподаватель. Однако при этом в результирующую таблицу необходимо включить и преподавателей, для которых не указан курс (еще не определена дисциплина, которую он будет вести).

Выполним запрос с использованием левого соединения в той же обработке (**Обработка4**). Добавим в имеющуюся форму новую команду ПравоеСоединение, программный код которой представлен в листинге 4.45.

Листинг 4.45. Процедура обработки команды ПравоеСоединение

```
&НаКлиенте
Процедура ПравоеСоединение (Команда)
ОтобратьИнформациюПравоеСоединение();
КонецПроцедуры
```

Вычислительные действия произведем в серверной процедуре ОтобратьИнформацию-ПравоеСоединение (листинг 4.46). Решение поставленной задачи реализуется с помощью конструкции ПРАВОЕ [ВНЕШНЕЕ] СОЕДИНЕНИЕ.

Листинг 4.46. Серверная процедура с использованием правого соединения

```
&НаСервере
Процедура ОтобратьИнформациюПравоеСоединение ()
  Запрос = Новый Запрос;
  Запрос.Текст = "ВЫБРАТЬ СпрКурсы.Наименование КАК Курс,
  | СпрСпециалисты.Наименование КАК ПреподавательУЦ,
  | СпрКурсы.Ссылка КАК СпрСсылка1,СпрСпециалисты.Ссылка КАК СпрСсылка2,
  | СпрКурсы.Стоимость КАК СтоимостьКурса,
  | СпрСпециалисты.Рейтинг КАК РейтингПреподавателя
```

```
  | ИЗ  Справочник.Курсы КАК СпрКурсы
  |     ПРАВОЕ СОЕДИНЕНИЕ
  | Справочник.Специалисты КАК СпрСпециалисты
  | ПО СпрКурсы.Преподаватель=СпрСпециалисты.Ссылка";
  Результат = Запрос.Выполнить().Выбрать();
  Сообщение=Новый СообщениеПользователю;
  Пока Результат.Следующий() > 0 Цикл
     Если Результат.СпрСсылка2.ЭтоГруппа = Ложь Тогда
        Сообщение.Текст=Строка(Результат.Курс) +
          "   " + Строка(Результат.ПреподавательУЦ) + "   "
          + Строка(Результат.СтоимостьКурса) + "руб.  Рейтинг: "
          + Строка(Результат.РейтингПреподавателя);
        Сообщение.Сообщить();
      КонецЕсли;
 КонецЦикла;
КонецПроцедуры
```

Для исключения попадания в результаты отбора названий категорий специалистов (техническое и гуманитарное направление) мы воспользовались в этой процедуре свойством `ЭтоГруппа`.

После этого создадим новую кнопку на форме для выполнения разработанной команды `ЛевоеСоединение`.

На рис. 4.50 показан результат выполнения запроса, учитывающего преподавателя, для которого не указан курс.

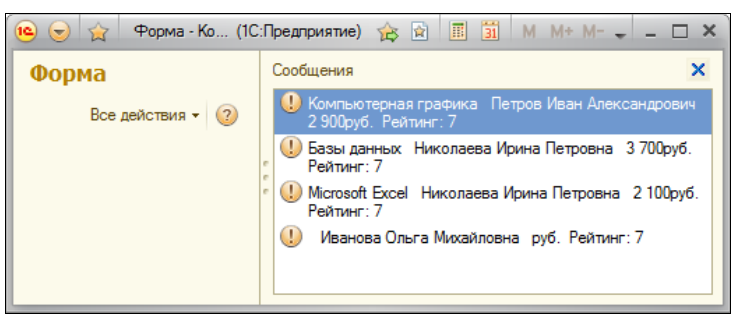


Рис. 4.50. Пример выполнения запроса с правым внешним соединением

Полное внешнее соединение

Конструкция `ПОЛНОЕ [ВНЕШНЕЕ] СОЕДИНЕНИЕ` означает, что в результат запроса надо включить комбинации записей из обеих исходных таблиц, которые соответствуют указанному условию. Кроме того, в результат запроса нужно включить также еще и те записи из обоих источников, для которых не найдено соответствия.

Таким образом, в результат запроса будут включены все записи из обоих источников и они будут соединены друг с другом при выполнении указанного условия.

Выполним запрос с использованием полного соединения в той же обработке (**Обработка4**). Добавим в имеющуюся форму новую команду ПолноеСоединение, программный код которой представлен в листинге 4.47.

Листинг 4.47. Процедура обработки команды ПолноеСоединение

```
&НаКлиенте
Процедура ПолноеСоединение(Команда)
 ОтобратьИнформациюПолноеСоединение();
КонецПроцедуры
```

Вычислительные действия произведем в серверной процедуре ОтобратьИнформацию-ПолноеСоединение (листинг 4.48). Решение поставленной задачи реализуется с помощью конструкции ПОЛНОЕ [ВНЕШНЕЕ] СОЕДИНЕНИЕ.

Листинг 4.48. Серверная процедура с использованием полного соединения

```
&НаСервере
Процедура ОтобратьИнформациюПолноеСоединение ()
  Запрос = Новый Запрос;
  Запрос.Текст = "ВЫБРАТЬ СпрКурсы.Наименование КАК Курс,
  |  СпрСпециалисты.Наименование КАК ПреподавательУЦ,
  | СпрКурсы.Ссылка КАК СпрСсылка1,СпрСпециалисты.Ссылка КАК СпрСсылка2,
  | СпрКурсы.Стоимость КАК СтоимостьКурса,
  | СпрСпециалисты.Рейтинг КАК РейтингПреподавателя
  |  ИЗ  Справочник.Курсы КАК СпрКурсы
  |     ПОЛНОЕ СОЕДИНЕНИЕ
  | Справочник.Специалисты КАК СпрСпециалисты
  | ПО СпрКурсы.Преподаватель=СпрСпециалисты.Ссылка";
  Результат = Запрос.Выполнить().Выбрать();
  Сообщение=Новый СообщениеПользователю;
  Пока Результат.Следующий() > 0 Цикл
      Если Результат.СпрСсылка1 = NULL Тогда
         Если Результат.СпрСсылка2.ЭтоГруппа = Ложь Тогда
            Сообщение.Текст= Строка(Результат.ПреподавательУЦ) +
              "   Рейтинг: "   +
             + Строка(Результат.РейтингПреподавателя);
                    Сообщение.Сообщить();
          КонецЕсли;
      Иначе
         Если Результат.СпрСсылка2 = NULL Тогда
              Сообщение.Текст= Строка(Результат.Курс) +
                     Строка(Результат.СтоимостьКурса) + "руб. ";
                      Сообщение.Сообщить();
          Иначе
             Сообщение.Текст=Строка(Результат.Курс) +
              "   " + Строка(Результат.ПреподавательУЦ) + "   "
```

```
                    + Строка(Результат.СтоимостьКурса) + "руб.  Рейтинг: "
                    + Строка(Результат.РейтинтПреподавателя);
                    Сообщение.Сообщить();
                КонецЕсли;
            КонецЕсли;
        КонецЦикла;
КонецПроцедуры
```

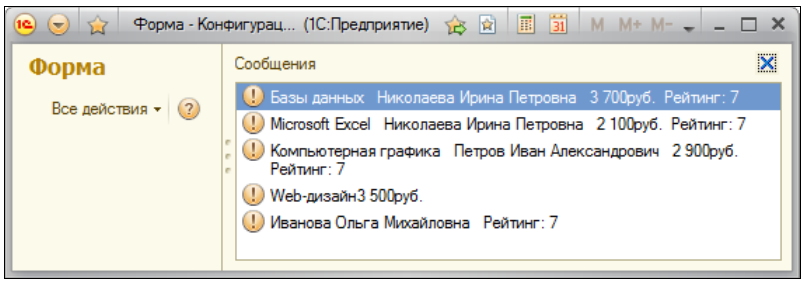


Рис. 4.51. Результат выполнения запроса с полным внешним соединением

На рис. 4.51 показан результат выполнения запроса с полным внешним соединением. Здесь в зависимости от полноты строки, выбранной с помощью запроса информации, изменяется формат отображения в элементе **РезультатыОтбора**. Например, при извлечении строки с информацией только о дисциплине (еще не указан преподаватель) не выводится лишнее обозначение относительно рейтинга специалиста. Аналогичная ситуация и для преподавателей.

Итоги

В этой главе мы на примерах познакомились с языком запросов. Разнообразные практические задачи продемонстрировали этот важный инструмент, используемый разработчиками для извлечения данных. Взяв за исходный материал ранее созданные информационные базы, был сформирован интерфейс для отбора и просмотра информации. При выводе информации использованы окно сообщений и табличный документ. С языком запросов мы еще встретимся в последующих главах книги, где значительное внимание будет уделено созданию отчетов.

ГЛАВА 5

Отчеты и система компоновки данных

При решении любых задач, связанных с количественным учетом, ключевым результатом являются отчеты. В предыдущих главах на ряде примеров было рассмотрено создание отчетов без использования специально предназначенных для этого средств автоматизации или с их минимальным использованием. Так, мы сами формировали макеты для табличных документов и создавали программные процедуры, которые в режиме **1С:Предприятие** приводили к заполнению табличных документов в соответствии со структурой макетов. Одним из основных достоинств системы 1С:Предприятие 8.2 являются эффективные возможности построения отчетов с минимальными усилиями со стороны разработчика. Наличие обеспечивающих это средств автоматизации повышает скорость разработки прикладных решений, а также предоставляет возможность самостоятельного формирования необходимых отчетов пользователям, не знакомым со встроенным языком программирования системы 1С:Предприятие.

Эта глава будет касаться технологии использования системы компоновки данных. Такая технология отсутствовала в версии 1С:Предприятие 8.0, а ее появление, начиная с программного продукта 1С:Предприятие 8.1, позволяет пользователю разрабатывать отчеты, не внедряясь в программный код на встроенном языке.

Система компоновки данных — это новое средство, которое предназначено для создания отчетов на основе декларативного описания. Механизм компоновки данных позволяет дать целостное описание отчета, используя только визуальные средства. Это отличает систему компоновки данных от построителя отчетов (основного средства автоматизации при создании отчетов в системе 1С:Предприятие 8.0).

При работе с системой компоновки данных на этапе визуального конструирования создается основа отчета — *схема компоновки данных*. Также создаются и настройки по умолчанию. В дальнейшем при интерактивной работе с отчетом в режиме **1С:Предприятие** пользователь может изменить имеющиеся настройки отчета, а также создать собственные.

Система компоновки данных позволяет реализовывать новые возможности, отсутствующие у построителя отчета. Все они приведены в документации, прилагаемой к рассматриваемому программному продукту.

Отметим основные черты системы компоновки данных:

- создание отчета без программирования;
- использование автоматически генерируемых форм отчета;
- настройка структуры отчета;
- совмещение в отчете нескольких таблиц;
- создание вложенных отчетов.

Прежде чем начать практические шаги, посмотрим на систему компоновки данных в целом. Она является набором элементов, каждому из которых отводится конкретное место при построении отчета. В итоге процесс создания отчета представляет собой переход от одного элемента к другому. Перечислим основные элементы системы компоновки данных:

- *схема компоновки данных* определяет суть данных, которые используются отчетом (откуда следует получать данные и как управлять их компоновкой);
- *настройки компоновки данных* представляют собой описание всего, что может быть настроено в установленной схеме компоновки данных (т. е. отбор, упорядочивание, параметры получения данных, параметры вывода данных и др.);
- *макет компоновки данных* задает готовое описание того, как должен формироваться отчет (он представляет собой результат применения конкретных настроек к схеме компоновки);
- *процессор компоновки данных*, который выбирает данные;
- *элемент результата компоновки данных* (выделение элемента результата компоновки данных связано с тем, что единого компонента результата компоновки данных нет, а существуют только его элементы).

Процесс компоновки данных состоит из нескольких этапов. Начальный этап называется созданием схемы компоновки данных. Этот этап чаще всего выполняется с использованием визуальных средств (конструктора схемы компоновки данных). Далее в книге мы будем рассматривать именно эти средства. Кроме того, можно использовать и программный текст на языке XML. Также возможно и применение программных объектов встроенного языка 1С:Предприятие 8.2.

Следующие этапы процесса компоновки данных связаны с редактированием настроек компоновки данных, подготовкой к исполнению (процесс формирования макета компоновки данных), исполнением компоновки данных, выводом результата компоновки данных (отчет может быть выведен в различных форматах).

Особенностью отчета, формируемого с использованием системы компоновки данных, является то, что он может включать различное сочетание таких элементов, как:

- группировка;
- таблица;
- диаграмма;
- вложенный отчет.

Теперь, после взгляда на систему компоновки данных в целом, рассмотрим ее на практических примерах. При этом в качестве исходной информации будем использовать разработки, уже созданные в предыдущих главах.

Отчеты по *главе 2*

Для начала обратимся к конфигурации, которая была нами разработана в *главе 2*. Эта информационная база связана с фиксированием поступления товаров.

Отчет по остаткам товаров

В *главе 2* для хранения информации о товарах мы создали регистр накопления **ИнформацияОТоварах**. Данный регистр предназначен для отражения поступления и отгрузки товаров. При работе в режиме **1С:Предприятие** информационная структура, созданная системой на основании данного объекта конфигурации, фиксирует все движения о поступлении и отгрузке товаров.

Итак, в режиме **Конфигуратор** откроем информационную базу, разработанную в *главе 2*. Первые шаги хорошо знакомы: необходимо создать новый объект конфигурации — отчет. Для этого нам потребуется обратиться к дереву объектов конфигурации.

На вкладке **Основные** введем имя отчета — **Отчет5_1** (рис. 5.1). Такой выбор названия связан с тем, что далее в этой главе будет создан ряд отчетов, и подобный стиль названий отражает номер главы в совокупности с порядковым номером отчета в главе. Разумеется, в реальных разработках рекомендуется давать отчетам более информативные названия.

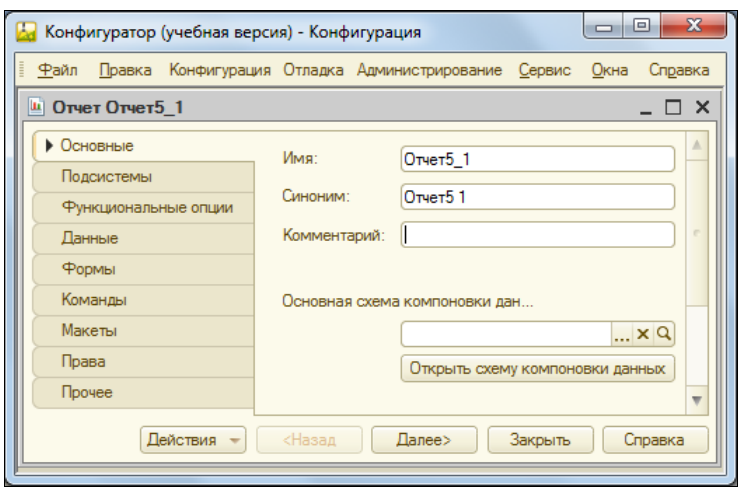


Рис. 5.1. Вкладка **Основные** окна редактирования объекта конфигурации **Отчет5_1**

Теперь на той же вкладке воспользуемся кнопкой **Открыть схему компоновки данных**. Для этого действия можно также использовать кнопку с изображением

лупы напротив поля ввода с подписью **Основная схема компоновки данных**. Это является первым шагом процесса создания схемы компоновки (ранее при создании отчетов мы ее не использовали).

В результате перед нами откроется окно конструктора макета (рис. 5.2). Установленный переключатель **Схема компоновки данных** отражает тип макета, который в данном случае потребуется. Заметим, что при выполнении практических действий значения параметров в большинстве подобных окон, как правило, изменять не требуется — по умолчанию они установлены верно.

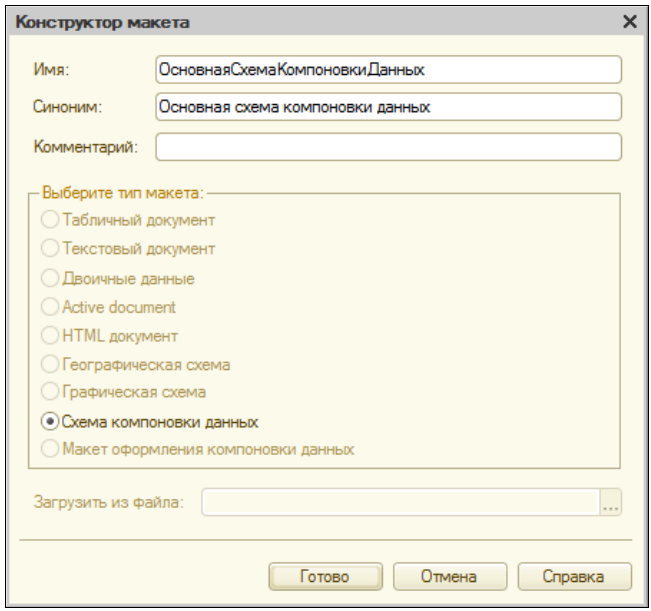


Рис. 5.2. Окно конструктора макета для схемы компоновки данных

После нажатия на кнопку **Готово** (см. рис. 5.2) перед нами откроется *окно конструктора схемы компоновки данных*, которое содержит много вкладок. На рис. 5.3 показана вкладка **Наборы данных**. С помощью данного конструктора можно создавать или редактировать источники данных. Также можно создавать и редактировать наборы данных, в качестве которых могут выступать:

- запрос (используется язык запросов и инструкции специального расширения языка запросов для системы компоновки данных);
- произвольный источник данных;
- объединение, механизм которого аналогичен объединению запросов средствами языка запросов.

Продолжим практические действия по созданию отчета. После нажатия на кнопку (**Добавить набор данных**) (можно также воспользоваться контекстным меню, открывающимся по щелчку правой кнопкой мыши по пиктограмме **Наборы данных**), выберем вариант **Добавить набор данных — запрос**. В результате окно кон-

структора схемы компоновки изменится (рис. 5.4), и в его правой части мы увидим кнопку **Конструктор запроса**.

Теперь следует создать необходимый запрос для получения количества имеющихся товаров. С помощью щелчка по кнопке **Конструктор запроса** перейдем к визуальной разработке запроса (ранее мы самостоятельно писали его текст на языке запросов, а теперь сформируем текст в результате визуальных действий).

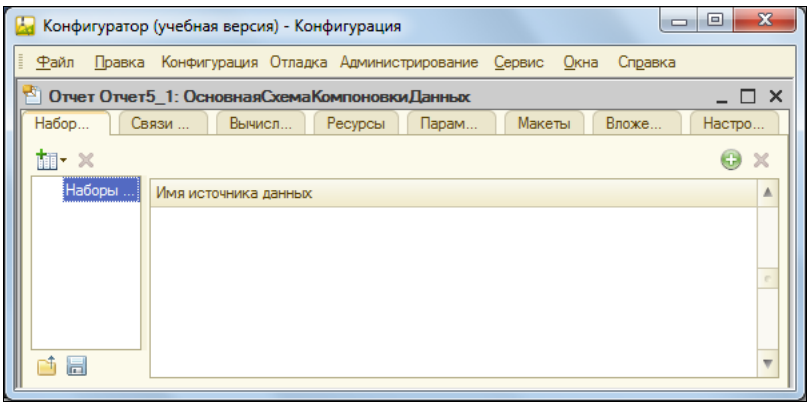


Рис. 5.3. Вкладка **Наборы данных** окна конструктора схемы компоновки данных

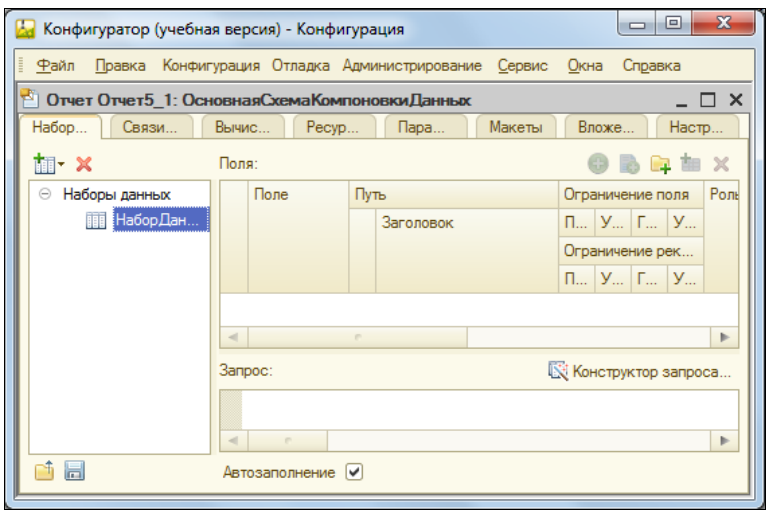


Рис. 5.4. Результат добавления набора данных в схему компоновки

На рис. 5.5 представлено окно конструктора запроса, в котором присутствуют три раздела:

- **База данных**;
- **Таблицы**;
- **Поля**.

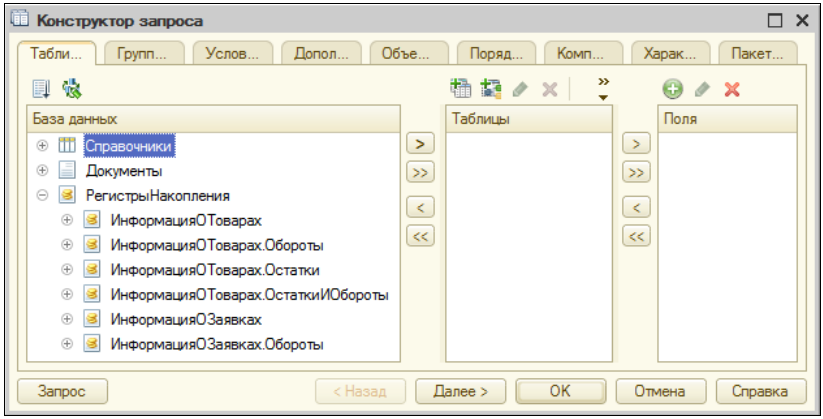


Рис. 5.5. Окно конструктора запроса

В разделе **База данных** представлены имеющиеся таблицы для формирования запросов. Эти таблицы отражают все данные, присутствующие в информационной базе. На основании информации, содержащейся в этих таблицах, и строятся разнообразные отчеты.

Если раскрыть ветку **РегистрыНакопления**, то кроме созданных нами (в *главе 2*) регистров **ИнформацияОТоварах** и **ИнформацияОЗаявках** мы увидим (см. рис. 5.5) еще несколько *виртуальных* таблиц, связанных с данными регистрами. Так, три виртуальные таблицы самостоятельно формируются системой на основании информации, имеющейся в регистре **ИнформацияОТоварах**:

- **ИнформацияОТоварах.Остатки**;
- **ИнформацияОТоварах.Обороты**;
- **ИнформацияОТоварах.ОстаткиИОбороты**.

В данный момент нас интересуют остатки товаров. Поэтому перенесем в раздел **Таблицы** таблицу **ИнформацияОТоварах.Остатки**. Далее для запроса необходимо выбрать используемые в нем поля. В рассматриваемой ситуации следует перенести два поля (**Товар**, **КоличествоОстаток**) в раздел **Поля** (рис. 5.6).

Окно конструктора запроса имеет еще ряд вкладок, позволяющих формировать необходимые указания для запроса. Так, можно создавать группировки, задавать условия, назначать псевдонимы полям, задавать порядок и т. д. В последующих примерах с этими действиями мы встретимся, а пока, после того как необходимые поля для запроса выбраны, можно просто нажать кнопку **ОК**. В результате проделанные визуальные действия приведут к формированию запроса, текст которого приведен в листинге 5.1.

Листинг 5.1. Запрос для получения информации о количестве товаров

```
ВЫБРАТЬ
  ИнформацияОТоварахОстатки.Товар,
  ИнформацияОТоварахОстатки.КоличествоОстаток
```

```
ИЗ
    РегистрНакопления.ИнформацияОТоварах.Остатки
    КАК ИнформацияОТоварахОстатки
```

При этом на вкладке **Наборы данных** произойдут изменения. На рис. 5.7 показано заполнение полей, которое в данном случае выполнено автоматически.

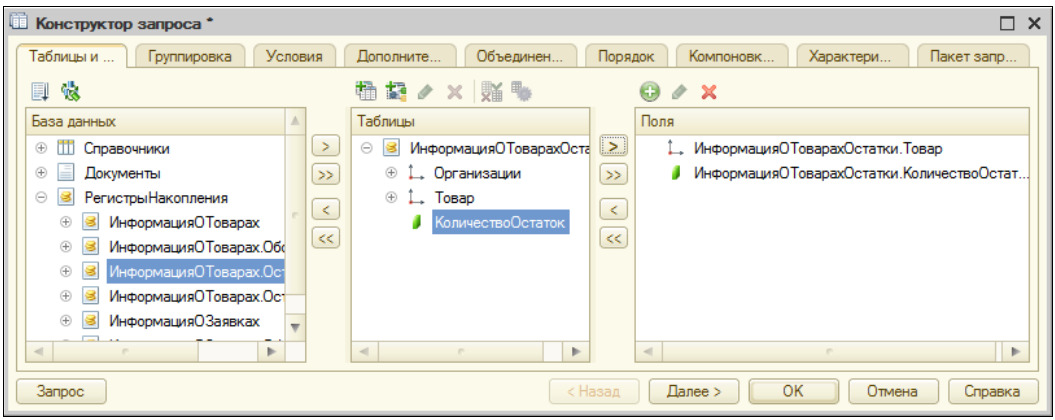


Рис. 5.6. Отбор полей для запроса по остаткам товаров

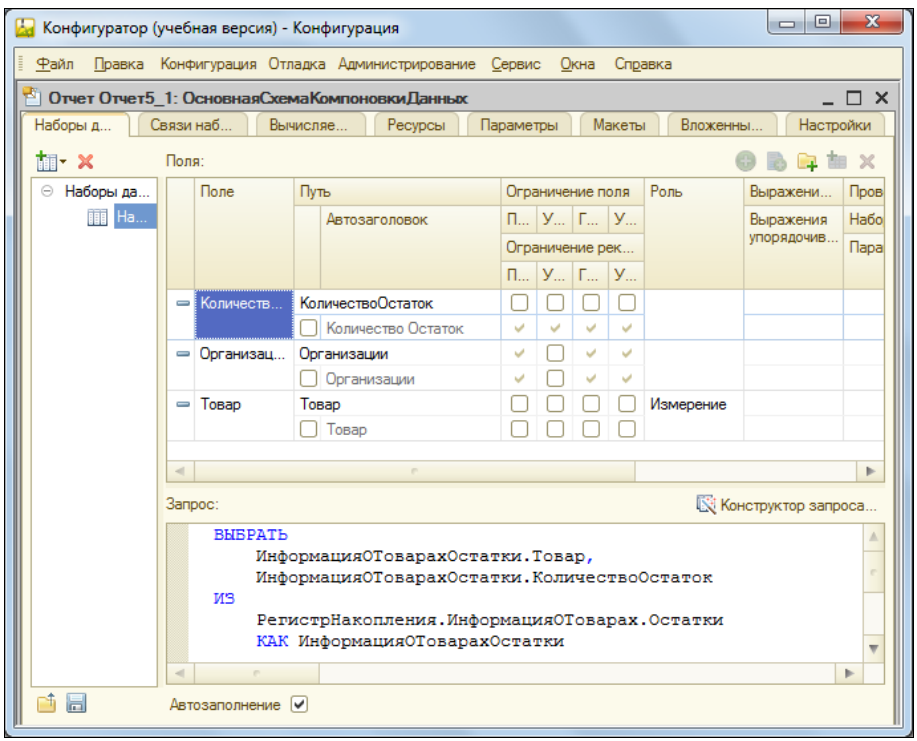


Рис. 5.7. Заполнение полей в схеме компоновки данных

Примечание

Поля для текущего набора данных будут сформированы системой автоматически только в том случае, если в конструкторе схемы компоновки данных установлен флажок **Автозаполнение**.

Прокомментируем раздел **Поля** (см. рис. 5.7).

Первый столбец с информацией называется **Поле**, в нем отражается имя поля набора данных, для которого формируется описание. Учитывая, что для получения набора данных мы воспользовались запросом, содержание поля определяется именем поля в тексте запроса. Важно отметить, что информация в столбце **Поле** не допускает редактирования.

Путь — это имя поля набора данных, под которым к нему можно обращаться в других разделах конструктора схемы компоновки данных. Это значение можно устанавливать самостоятельно.

Заголовок — еще одно имя поля набора данных, под которым оно будет фигурировать в настройках в режиме **1С:Предприятие**, а также в названии данного поля в шапке самого отчета. Значение в этом поле также может быть назначено разработчиком самостоятельно.

Настройка в столбце **Роль** показывает, каким образом поле будет использовано в дальнейшем в системе компоновки данных. Информация о том, что поле является измерением, используется при расчете итогов.

Сейчас наша задача заключается в том, чтобы получить информацию об остатках товаров. Для этого перейдем на вкладку **Настройки**, где необходимо задать настройки отчета по умолчанию, а именно следует щелкнуть правой кнопкой мыши на элементе **Отчет** и в появившемся контекстном меню выбрать пункт **Новая группировка**.

Примечание

Вкладка **Настройки** в нижней части окна имеет еще ряд дополнительных вкладок, дающих разработчику разнообразные возможности влиять на формирование отчета.

Элемент **Группировка** позволяет отображать в отчете не только сгруппированную определенным образом информацию, но и детальные записи, полученные системой в результате обработки текста запроса. И если мы хотим отображать в отчете детальные записи, то в окне **Группировка** (рис. 5.8) следует раздел **Поле** оставить пустым.

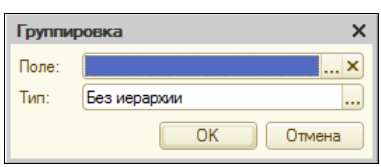


Рис. 5.8. Окно для указания группировочного поля

Перейдем теперь на дополнительную вкладку **Выбранные поля** (рис. 5.9), где в число выбранных полей следует добавить два поля: **Товар** и **КоличествоОстаток**. Добавление полей в раздел **Поля** проще всего произвести, дважды щелкнув по

нужному полю в разделе **Доступные поля**. Также для этого можно воспользоваться перетаскиванием названий из раздела **Доступные поля**. Третий способ для добавления — использование кнопки **Добавить** из пиктографического меню, расположенного в правой части окна.

Учитывая, что при разработке рассматриваемой конфигурации были созданы две подсистемы, то и созданный отчет следует включить в одну из подсистем (выберем для этого и всех последующих отчетов — **ЗаведующийФилиалом**).

После этого можно проверить формирование отчета (рис. 5.10). Для этого следует перейти в режим **1С:Предприятие**. На панели действий в группе **Отчеты** присутствует пиктограмма созданного отчета **Отчет5_1**. Выбрав его и нажав кнопку **Сформировать**, в следующем окне мы получим готовый отчет по остаткам товаров (см. рис. 5.10).

Использование системы компоновки данных позволяет расширить возможности формирования отчетов без внесения изменений в конфигурацию. Рассмотрим, как

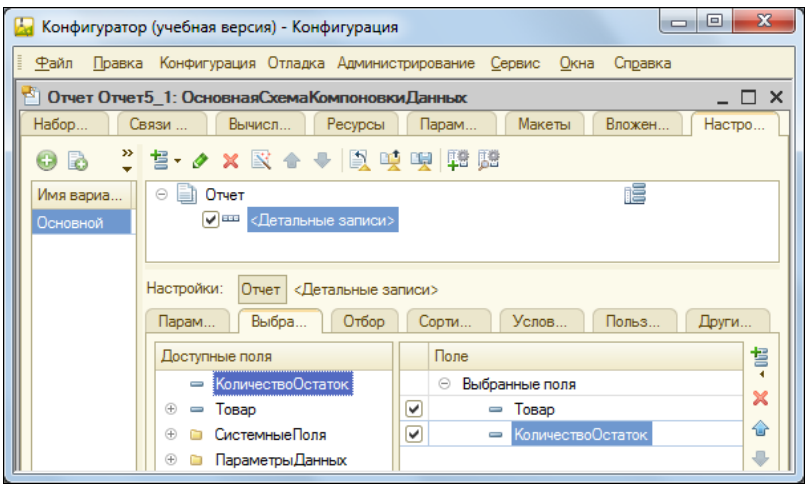


Рис. 5.9. Заполнение дополнительной вкладки **Выбранные поля**

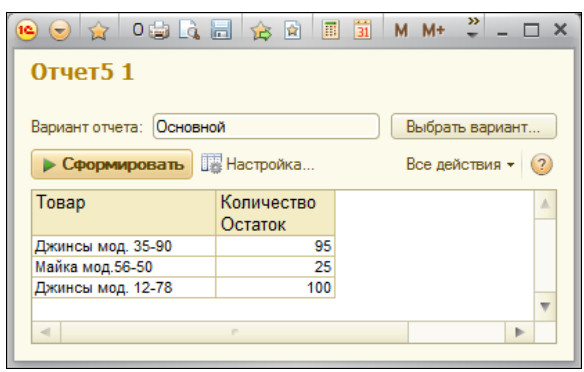


Рис. 5.10. Отчет по остаткам товаров

это выглядит на практике. Будем считать, что в нашем отчете необходимо просмотреть суммарную информацию о том, сколько всего в настоящий момент имеется товаров (необходимо просуммировать остатки по всем товарам). Для этого не требуется переходить в режим конфигуратора и перерабатывать созданный отчет. Необходимо лишь воспользоваться кнопкой **Все действия** (см. рис. 5.10) в форме отчета.

В результате открывается выпадающий список команд, в котором следует воспользоваться пунктом **Изменить вариант**. После этого на экране появится новое окно (рис. 5.11), в котором необходимо перейти на дополнительную вкладку **Поля**. Здесь требуется снять флажок напротив поля **Товар**. Больше никаких действий не требуется и следует просто нажать кнопку **Завершить редактирование**.

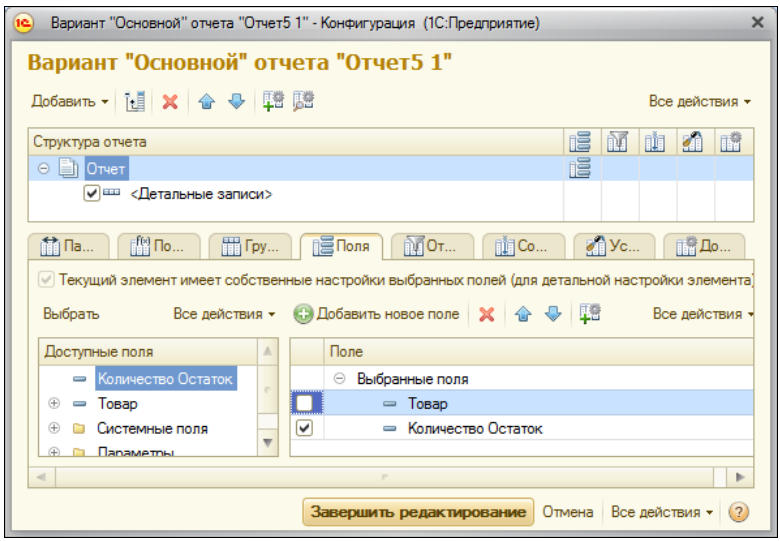


Рис. 5.11. Настройка отчета для получения общего остатка по товарам

В результате мы вернемся в исходную форму отчета (см. рис. 5.10), для его повторного формирования (с новыми созданными настройками) осталось щелкнуть по кнопке **Сформировать**. После этого на экране появится уже новая версия отчета (рис. 5.12) с интересующим нас общим итогом по товарам.

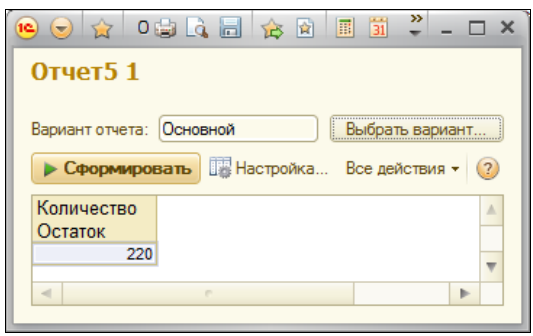


Рис. 5.12. Отчет по суммарным остаткам товаров

Важно отметить, что изменения в формировании отчета (изменение вида отчета) мы выполнили непосредственно в режиме **1С:Предприятие**. На практике с помощью подобных действий из готовых отчетов можно получать дополнительную информацию.

Использование конструкторов настроек

Если охарактеризовать проделанную работу по созданию предыдущего отчета с помощью системы компоновки данных, то можно выделить в разработке две составляющие:

- создание набора данных для отчета было сделано с помощью конструктора запросов;
- при формировании необходимых настроек для отчета применялись ручные действия.

Однако кроме ручного формирования настроек можно воспользоваться и *конструктором настроек*. В этом разделе мы создадим настройки для отчета по товарам (на этот раз нас будут интересовать кроме остатков товаров еще и приход с расходом), но при этом воспользуемся конструктором настроек.

Начнем с того, что в дереве объектов конфигурации сформируем новый отчет с именем **Отчет5_2**. Начальные технические действия по его созданию аналогичны тем, которые были выполнены при разработке предыдущего отчета.

Первый момент, отличающий новый отчет, связан с визуальным формированием запроса с помощью конструктора (рис. 5.13). Здесь мы воспользовались виртуальной таблицей **ИнформацияОТоварах.ОстаткиИОбороты**, из которой в раздел **Поля** с помощью соответствующей пиктограммы необходимо перенести четыре поля:

- **ИнформацияОТоварахОстаткиИОбороты.Товар**;
- **ИнформацияОТоварахОстаткиИОбороты.КоличествоКонечныйОстаток**;
- **ИнформацияОТоварахОстаткиИОбороты.КоличествоПриход**;
- **ИнформацияОТоварахОстаткиИОбороты.КоличествоРасход**.

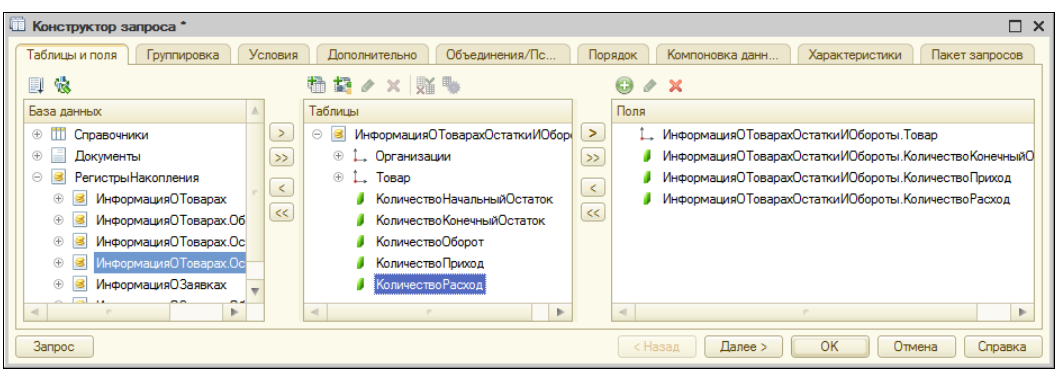


Рис. 5.13. Окно конструктора запроса для объекта **Отчет5_2**

Теперь следует с помощью щелчка по кнопке **ОК** завершить визуальное формирование запроса и вернуться в окно конструктора схемы компоновки данных (рис. 5.14).

В результате проделанных нами визуальных действий платформа сформировала синтаксически правильный текст запроса (листинг 5.2).

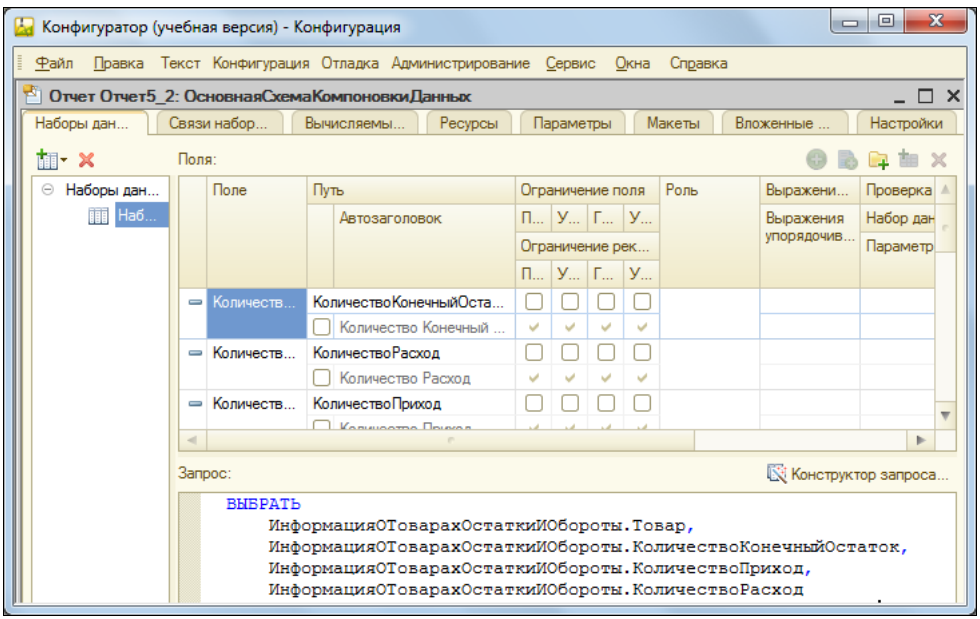


Рис. 5.14. Окно конструктора схемы компоновки данных для объекта **Отчет5_2**

Листинг 5.2. Запрос для получения информации по товарам

```
ВЫБРАТЬ
  ИнформацияОТоварахОстаткиИОбороты.Товар,
  ИнформацияОТоварахОстаткиИОбороты.КоличествоКонечныйОстаток,
  ИнформацияОТоварахОстаткиИОбороты.КоличествоПриход,
  ИнформацияОТоварахОстаткиИОбороты.КоличествоРасход
ИЗ
  РегистрНакопления.ИнформацияОТоварах.ОстаткиИОбороты
  КАК ИнформацияОТоварахОстаткиИОбороты
```

Теперь на вкладке **Настройки** окна конструктора схемы компоновки данных следует обратиться к пиктографическому меню, расположенному в верхней части окна, и воспользоваться пиктограммой **Открыть конструктор настроек** (рис. 5.15).

В результате перед нами откроется окно конструктора настроек, показанное на рис. 5.16. На первом шаге требуется выбрать тип отчета, который мы собираемся создать (в данном случае необходим вариант **Список**).

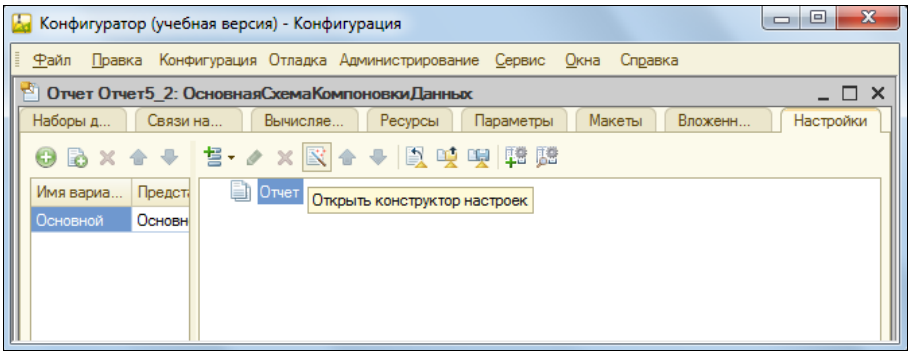


Рис. 5.15. Вкладка **Настройки** окна конструктора схемы компоновки данных

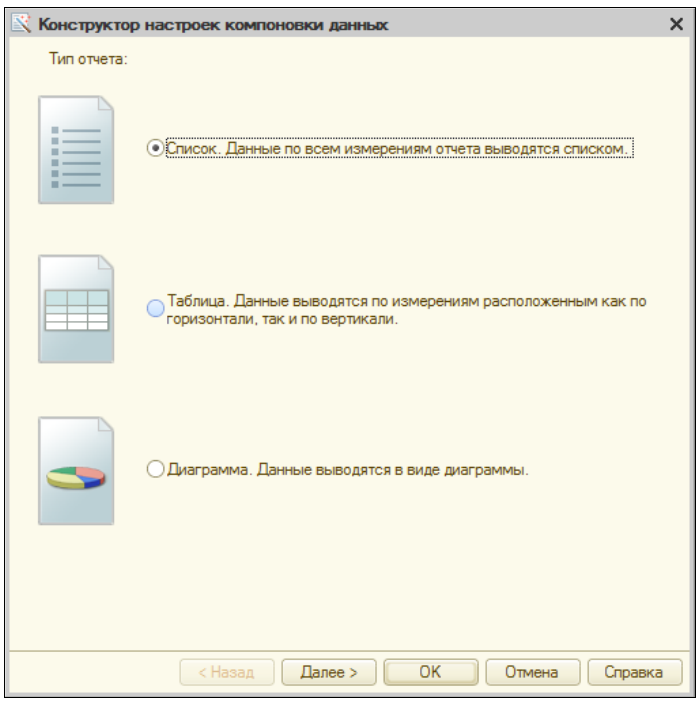


Рис. 5.16. Диалог с конструктором настроек (шаг 1)

Примечание

Примеры других вариантов отчета (**Таблица** и **Диаграмма**) будут рассмотрены далее в этой главе. Надо заметить, что в предыдущем разделе, когда создание настроек производилось вручную, отчет был также сформирован в виде списка.

Итак, отметив вариант **Список** (см. рис. 5.16), нажмем кнопку **Далее**. Следующее окно, которое увидим на экране, представлено на рис. 5.17. Здесь от нас требуется определить поля, выводимые в отчет. Из левого списка доступных полей следует перенести четыре поля: **Товар**, **КоличествоПриход**, **КоличествоРасход**, **КоличествоКонечныйОстаток**.

Примечание

Для изменения порядка расположения полей в отчете следует пользоваться кнопками **Вверх** и **Вниз**.

Выполнив данный шаг, нажмем кнопку **Далее** (см. рис. 5.17), что приведет к появлению на экране следующего окна, показанного на рис. 5.18. На этом шаге нам предоставляется возможность сформировать в отчете поля группировки. Например, если в базе данных имеется несколько записей, касающихся одного и того же товара, то в отчете можно получить суммарные данные по этому товару. Но поскольку таблица **ИнформацияОТоварах.Остатки** является виртуальной, а информация по товарам уже сгруппирована, то этот этап создания отчета следует пропустить — сразу нажмем кнопку **Далее.**

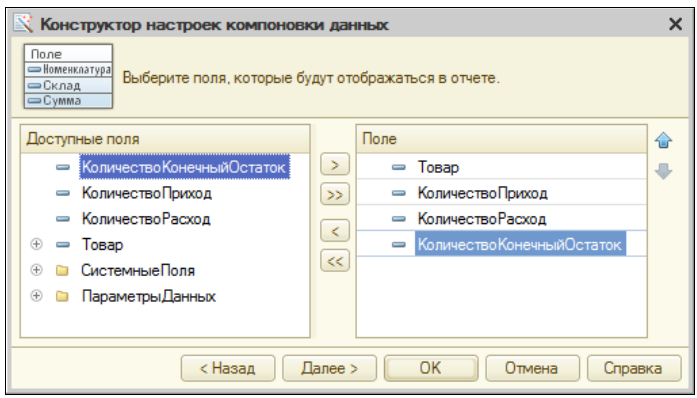


Рис. 5.17. Диалог с конструктором настроек (шаг 2)

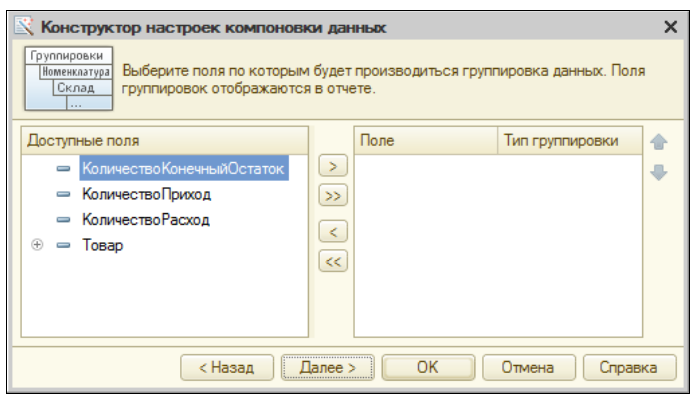


Рис. 5.18. Диалог с конструктором настроек (шаг 3)

В результате на экране появится следующее окно, предназначенное для сортировки данных (рис. 5.19). Здесь при необходимости упорядочивания данных в создаваемом отчете требуется указать конкретные поля.

В данном случае разумно упорядочить информацию по товарам в порядке возрастания. Для этого достаточно перенести в правый список (из левого) поле **Товар**.

Как видно, кнопка **Далее** неактивна, что говорит о том, что данное окно диалога последнее и остается только щелкнуть по кнопке **ОК**. В результате мы получим настройки отчета, но на этот раз они сделаны конструктором.

Необходимо отнести созданный отчет к одной из подсистем (выберем — **ЗаведующийФилиалом**), после чего можно перейти в режим **1С:Предприятие** и получить готовый отчет (рис. 5.20).

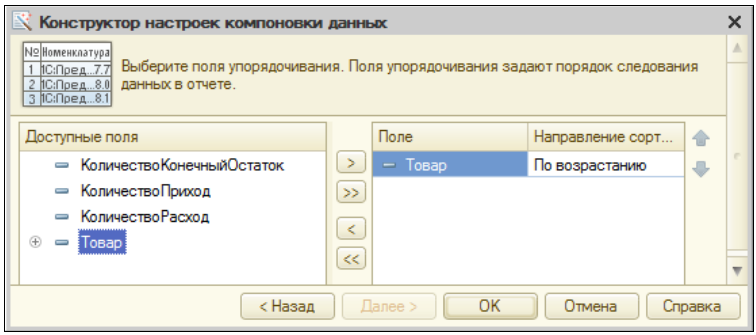


Рис. 5.19. Диалог с конструктором настроек (шаг 4)

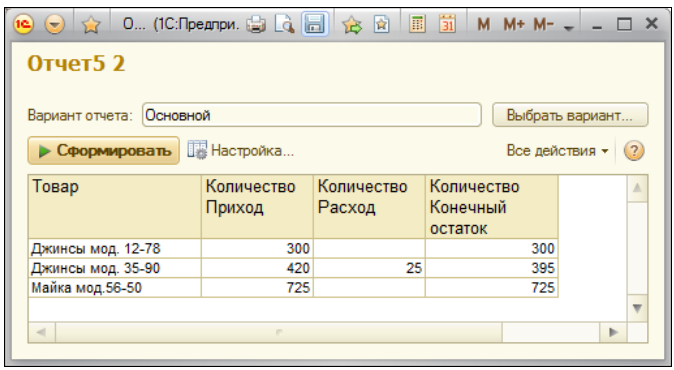


Рис. 5.20. Отчет5_2 в режиме **1С:Предприятие**

Отчет в виде таблицы

Предыдущие примеры касались использования одного варианта отчета — вывода данных в виде списка. Рассмотрим теперь формирование отчета в виде таблицы. В этом случае интересующая пользователя информация находится на пересечении определенного столбца и конкретной строки. Будем считать, что нас интересуют сведения о поступлениях товаров от организаций. По горизонтали в отчете сформируем перечень организаций, а по вертикали — перечень товаров. Содержательная часть таблицы будет касаться количества поступивших товаров от определенных организаций.

Продолжим работу с информационной базой из *главы 2* и создадим новый отчет — **Отчет5_3**. При этом, как обычно, воспользуемся кнопкой **Открыть схему компо-**

новки данных. На первом этапе необходимо сформировать запрос для получения информации о поступивших товарах. Фактически нас интересует приход товаров. Поэтому перенесем в раздел **Таблицы** таблицу **ИнформацияОТоварах.Обороты** и выберем необходимые поля (рис. 5.21):

- **Организации**;
- **Товар**;
- **КоличествоПриход**.

Больше никаких установок в окне конструктора запросов не требуется. В результате проделанные визуальные действия привели к формированию запроса, текст которого показан в листинге 5.3.

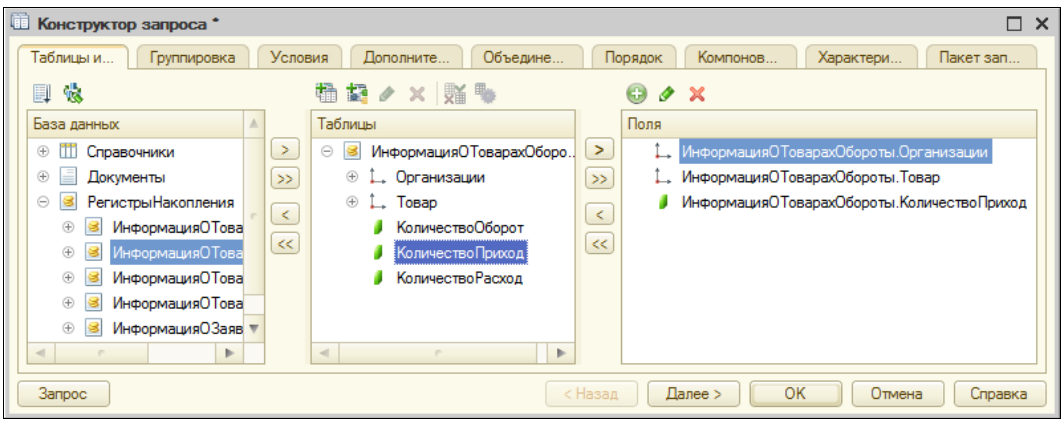


Рис. 5.21. Окно конструктора запроса

Листинг 5.3. Запрос для получения информации о поступлении товаров

```
ВЫБРАТЬ
  ИнформацияОТоварахОбороты.Организации,
  ИнформацияОТоварахОбороты.Товар,
  ИнформацияОТоварахОбороты.КоличествоПриход
ИЗ
  РегистрНакопления.ИнформацияОТоварах.Обороты
   КАК ИнформацияОТоварахОбороты
```

Для задания группировки информации в отчете предназначена вкладка **Ресурсы**. И для включения в отчет конкретного ресурса необходимо перенести его в раздел **Поле** (рис. 5.22). Для этого достаточно выполнить двойной щелчок мышью по необходимому полю в разделе **Доступные поля**. Можно также воспользоваться кнопкой, расположенной на разделительной полосе окна. В колонке **Выражение** содержится выражение, по которому будет вычисляться значение ресурса. По

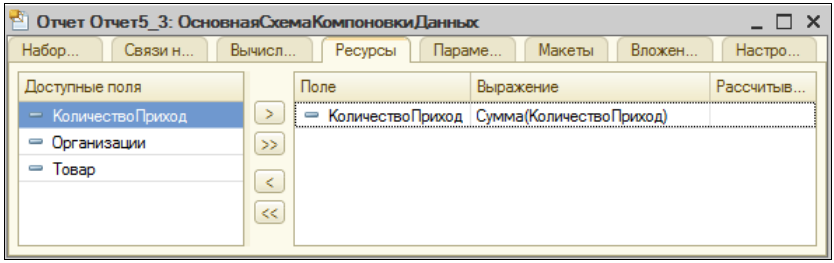


Рис. 5.22. Вкладка **Ресурсы** окна конструктора схемы компоновки данных

умолчанию используется выражение **Сумма**, которое нас устраивает, но доступны и другие функции работы с данными.

Следующий этап заключается в создании настроек отчета, для чего воспользуемся конструктором настроек. В данном случае нам требуется указать вариант **Таблица** (рис. 5.23), после чего в следующем окне отметить все имеющиеся поля для отчета (рис. 5.24).

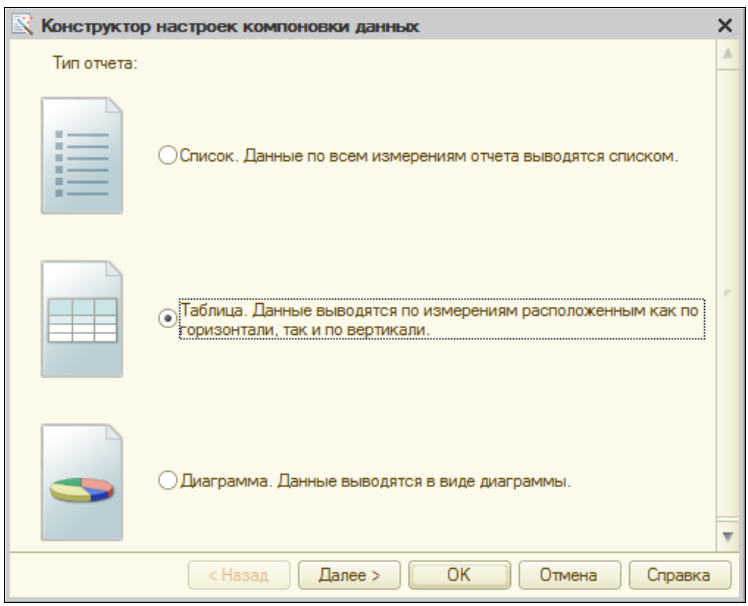


Рис. 5.23. Диалог с конструктором настроек (шаг 1)

В очередном окне (рис. 5.25) определим поля для группировки данных в отчете. После этого на следующем шаге следует указать упорядочивание по товару.

В результате проделанных действий на экране отобразится окно конструктора схемы компоновки данных с автоматически созданными настройками (рис. 5.26).

Необходимо отнести созданный отчет к одной из подсистем, после чего разработка готова и можно перейти в режим **1С:Предприятие** для формирования отчета по имеющимся данным (рис. 5.27).

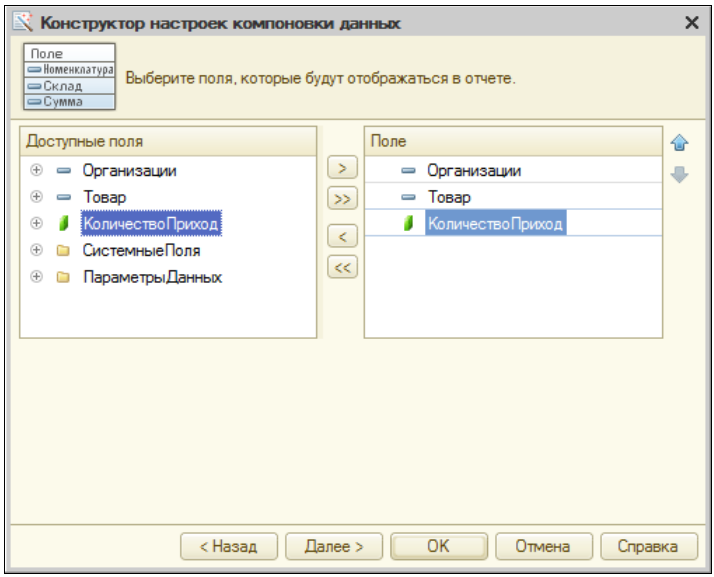


Рис. 5.24. Диалог с конструктором настроек (шаг 2)

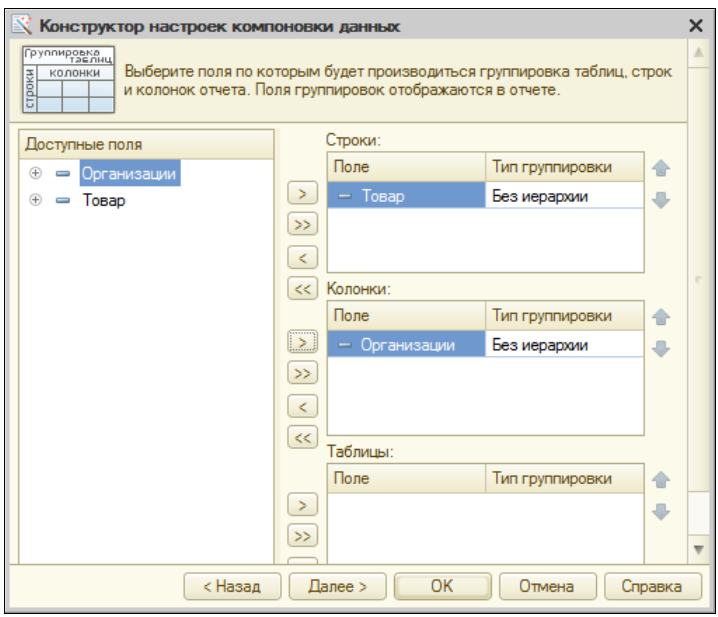


Рис. 5.25. Диалог с конструктором настроек (шаг 3)

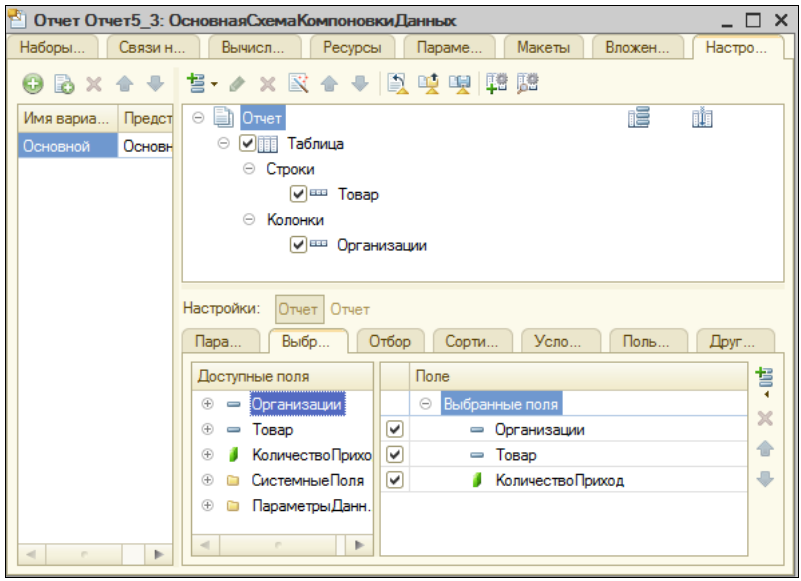


Рис. 5.26. Автоматическое формирование настроек отчета

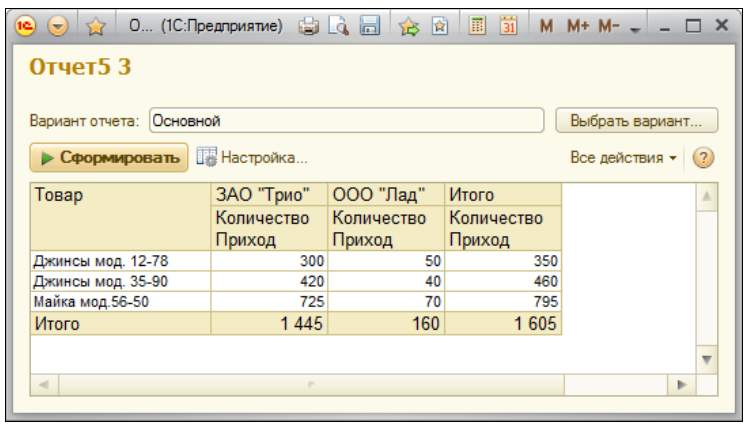


Рис. 5.27. Отчет в виде таблицы

Отчет в виде диаграммы

После рассмотрения отчетов в виде списка и таблицы осталось привести пример использования в отчете диаграммы. Разработаем отчет (он будет иметь имя **Отчет5_4**) в варианте, представленном на рис. 5.28.

Начальные шаги по созданию нового отчета не требуют комментария, т. к. они вполне аналогичны предыдущим отчетам данной главы. Первый ключевой момент связан с формированием запроса для получения информации о поступлениях товаров. На рис. 5.29 представлены необходимые действия, которые следует выполнить в окне конструктора запросов.

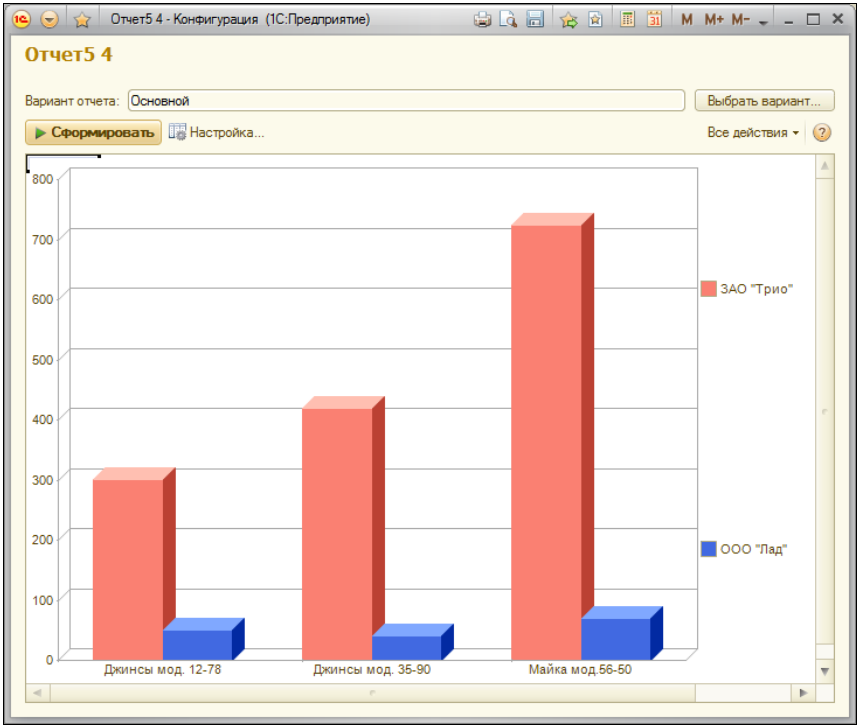


Рис. 5.28. Отчет в виде диаграммы

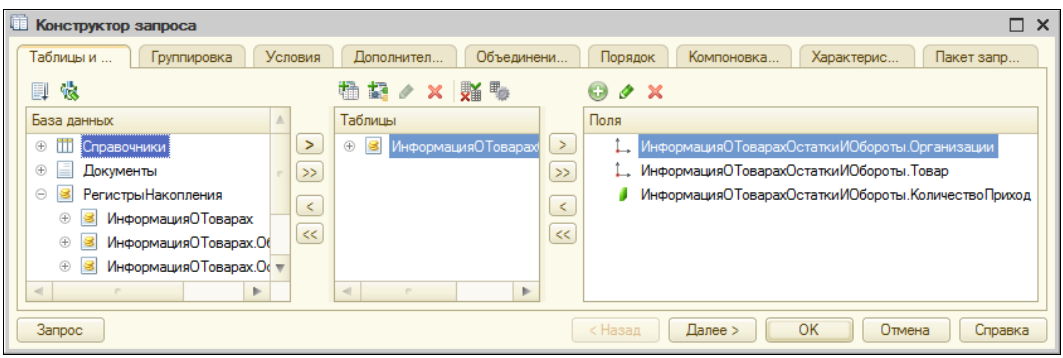


Рис. 5.29. Окно конструктора запроса

Дальнейших действий по конструированию запроса не требуется и щелчок по кнопке **ОК** приводит к заполнению вкладки **Наборы данных** окна конструктора схемы компоновки данных.

Далее требуется установить ресурс для схемы компоновки, после чего следует перейти на вкладку **Ресурсы** и выбрать в качестве ресурса поле **КоличествоПриход** (рис. 5.30).

Затем в окне конструктора схемы компоновки данных перейдем на вкладку **Настройки** и воспользуемся конструктором настроек (кнопка **Открыть конструктор**

настроек в пиктографическом меню). В первом окне диалога с конструктором следует с помощью соответствующего переключателя указать в качестве типа создаваемого отчета **Диаграмма** (рис. 5.31).

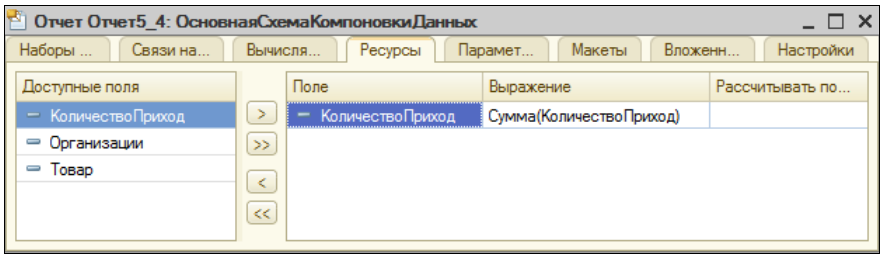


Рис. 5.30. Вкладка **Ресурсы** окна конструктора схемы компоновки данных

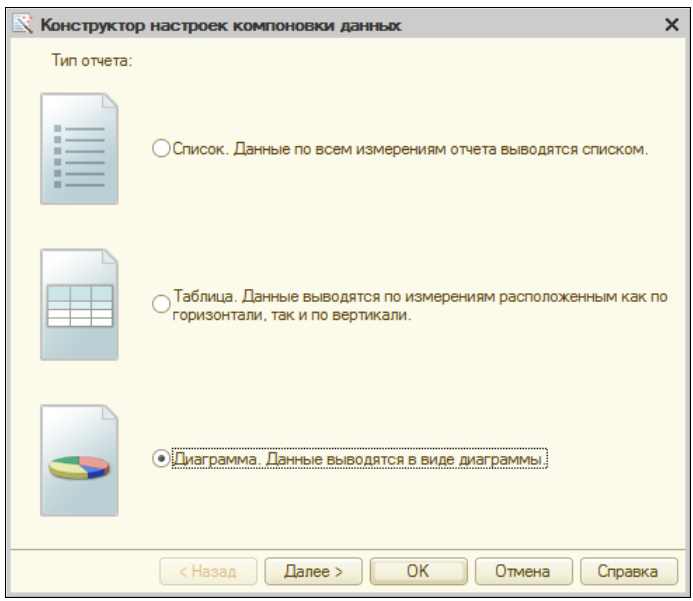


Рис. 5.31. Окно конструктора настроек для выбора типа отчета

В следующем окне конструктора настроек схемы компоновки данных (рис. 5.32) необходимо указать поля для создаваемого отчета.

После этого на экране откроется новое окно конструктора, которое специфично именно для отчета с использованием диаграммы. Так, здесь необходимо выбрать поля для серий и точек диаграммы. Эти установки следует выполнить в соответствии с рис. 5.33.

Далее на экране открывается окно для указания упорядочивания в отчете. Учитывая небольшое количество введенной информации в информационную базу, поля для упорядочивания в данном случае можно не указывать. После этого на экране отображается окно для выбора типа диаграммы (рис. 5.34) — нас устроит вариант **Гистограмма объемная**.

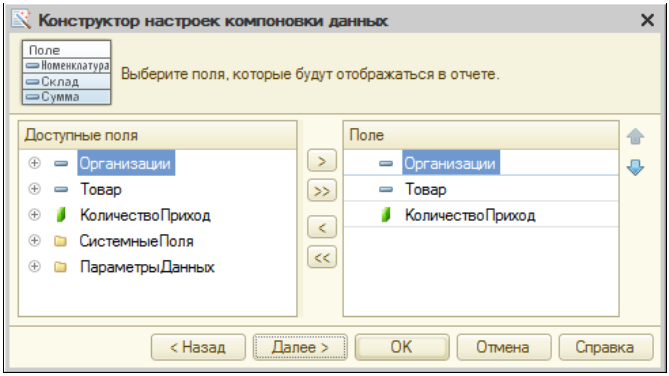


Рис. 5.32. Окно конструктора настроек для выбора полей отчета

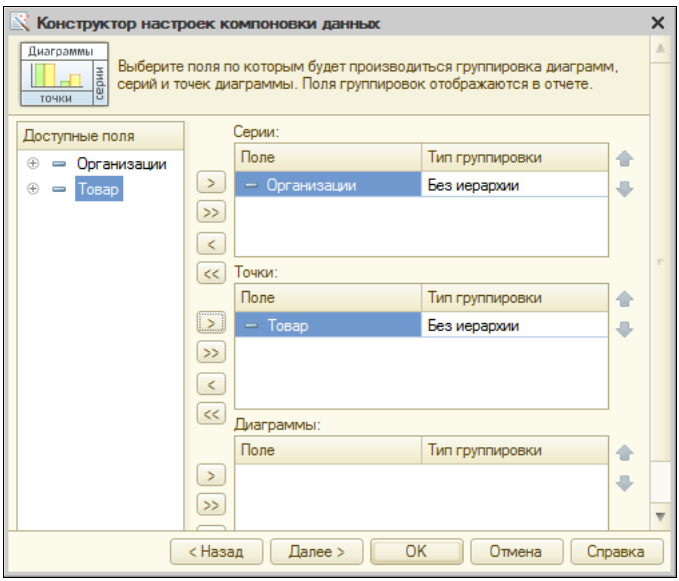


Рис. 5.33. Окно для определения серий и точек диаграммы

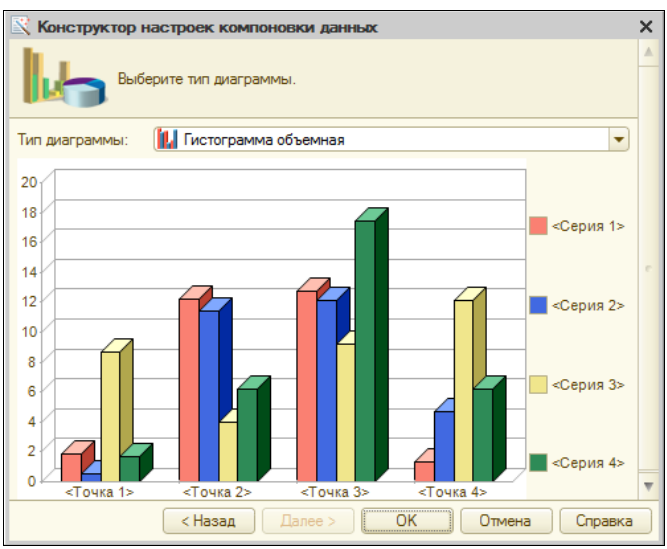


Рис. 5.34. Окно для определения типа диаграммы

Фактически на этом этап разработки отчета завершен, а в режиме **1С:Предприятие** его мы уже видели (см. рис. 5.28).

Примечание

Необходимо включить созданный отчет в одну из подсистем.

Формирование отчета по поступлениям с итоговой информацией

Рассмотрим еще один из вариантов отчета по поступлениям товаров, в котором будут реализованы следующие требования:

- организация отчета в виде списка;
- определение пользователем интервала по дате для отбора информации;
- включение в отчет итоговых показателей по каждому товару;
- упорядочивание информации в отчете.

Приступим к созданию нового отчета, который назовем **Отчет5_5**. Начальные шаги традиционны — щелчок по кнопке **Открыть схему компоновки данных** и добавление нового набора данных (запроса). В окне конструктора запроса перенесем таблицу **ИнформацияОТоварах.Обороты** и выберем из нее три поля: **Организации**, **Товар** и **КоличествоПриход** (рис. 5.35).

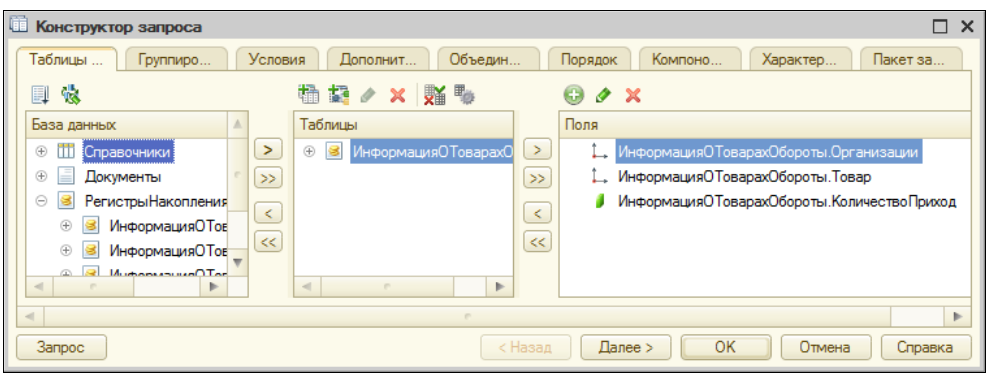


Рис. 5.35. Окно конструктора запроса

Для того чтобы данные в отчете были упорядочены необходимым образом, следует сделать установки на вкладке **Порядок** (рис. 5.36).

Теперь можно щелкнуть по кнопке **ОК**, что позволит сформировать текст запроса на языке запросов (листинг 5.4).

Листинг 5.4. Запрос для получения информации о поступивших товарах

```
ВЫБРАТЬ
    ИнформацияОТоварахОбороты.Организация КАК Организация,
    ИнформацияОТоварахОбороты.Товар КАК Товар,
    ИнформацияОТоварахОбороты.КоличествоПриход КАК КоличествоПриход
```

```
ИЗ
    РегистрНакопления.ИнформацияОТоварах.Обороты
            КАК ИнформацияОТоварахОбороты
УПОРЯДОЧИТЬ ПО
    Товар,
    Организация,
    КоличествоПриход
```

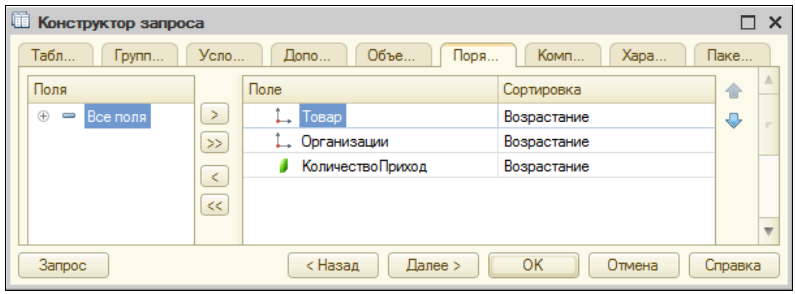


Рис. 5.36. Вкладка для установки упорядочивания результатов запроса

В создаваемый отчет вместе с детальной информацией необходимо включить итоговые показатели по поступлению каждого товара. Для этого в окне конструктора схемы компоновки перейдем на вкладку **Ресурсы** и выберем в качестве ресурса поле **КоличествоПриход**.

Теперь осталось перейти на вкладку **Настройки**, где следует создать группировку по товарам, включающую вложенную группировку для отображения детальных записей (рис. 5.37).

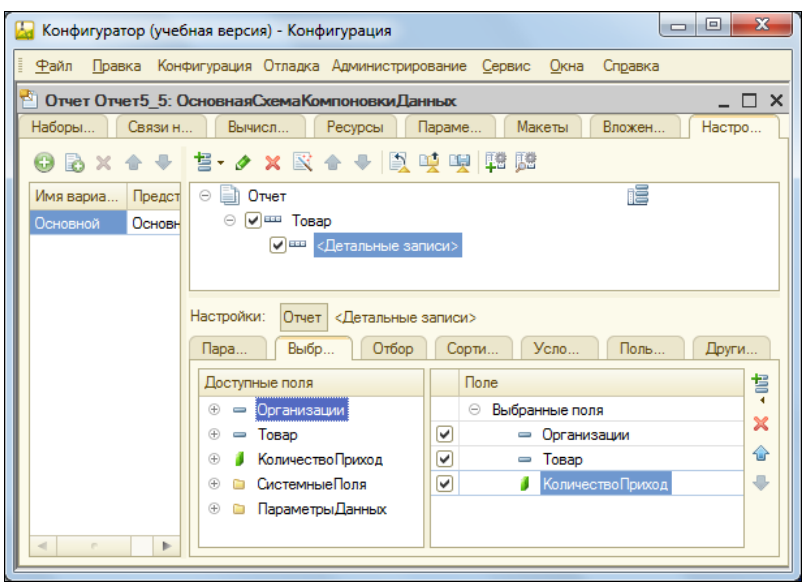


Рис. 5.37. Установки на вкладке **Настройки**

На завершающем этапе разработки отчета его следует отнести к одной из подсистем. В режиме **1С:Предприятие** сформированный отчет по имеющимся данным представлен на рис. 5.38. Получившийся отчет весьма информативен. Так, мы видим, сколько всего различных товаров поступило, при этом в случае необходимости можно увидеть детализацию прихода товаров по конкретным поставщикам.

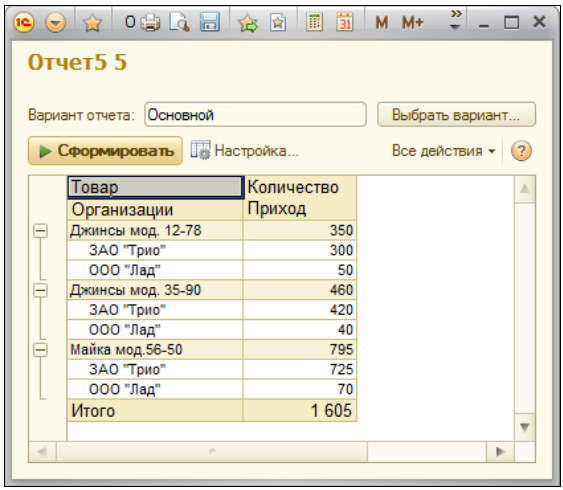


Рис. 5.38. Отчет по поступлениям товаров с группировкой данных

Однако в такой редакции в отчетную информацию попадают сведения обо всех поступлениях. Реально же часто требуется просмотреть информацию в определенном периоде по дате. Для этого в нашем отчете системой автоматически предусмотрены параметры (это обусловлено той виртуальной таблицей, которую мы указали):

- **Начало периода**;
- **Конец периода**.

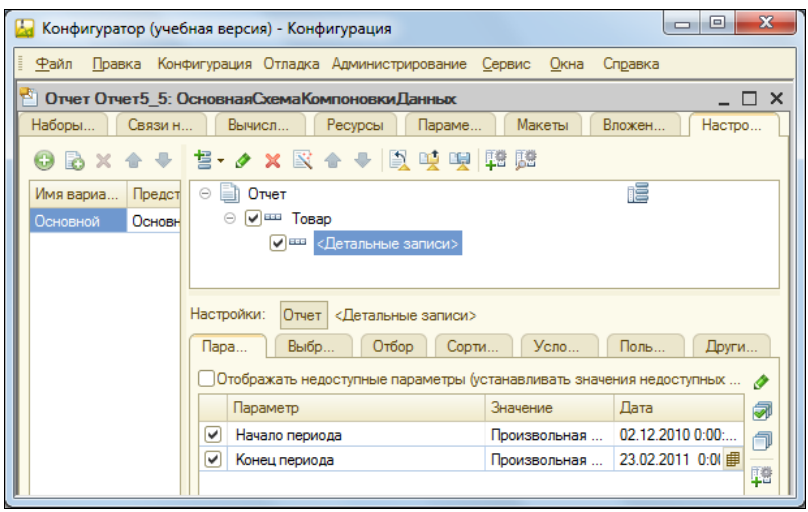


Рис. 5.39. Задание параметров отчета на вкладке **Настройки**

Для того чтобы выполнить их включение в отчет, следует в окне конструктора схемы компоновки данных перейти на вкладку **Настройки** и далее активизировать вкладку **Параметры** (рис. 5.39). Теперь если сформировать отчет (рис. 5.40), то мы получим сведения только по тем поступлениям, которые ограничены указанными датами.

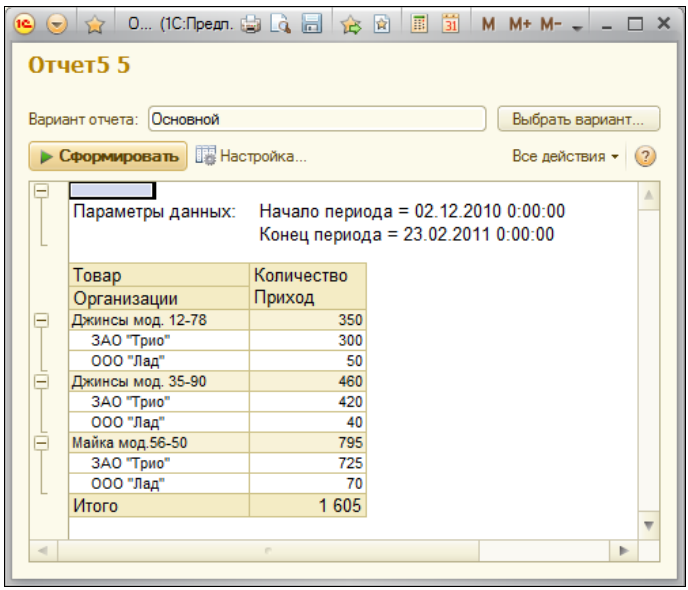


Рис. 5.40. Отчет о поступлениях по интервалу дат

Формирование в отчете двух таблиц

Рассмотрим вариант отчета (**Отчет5_6**) с отображением в нем двух таблиц. Внешний вид создаваемого отчета показан на рис. 5.41, где представлены две сгруппированные таблицы — одна по товарам, а другая по организациям.

При создании нового отчета следует выполнить шаги, аналогичные тем, которые были проделаны при разработке отчета **Отчет5_5**. Изменения касаются вкладки **Настройки** (рис. 5.42). Здесь требуется создать две независимые группировки:

- **Товар**;
- **Организации**.

В результате после перехода в режим **1С:Предприятие** мы увидим отчет в желаемом виде.

Отчет по документам *Заявка*

Разработаем отчет, который будет включать в себя информацию по имеющимся заявкам на товары. Данный отчет (**Отчет5_7**) предназначен для включения в него перечня имеющихся в информационной базе данных документов "Заявка".

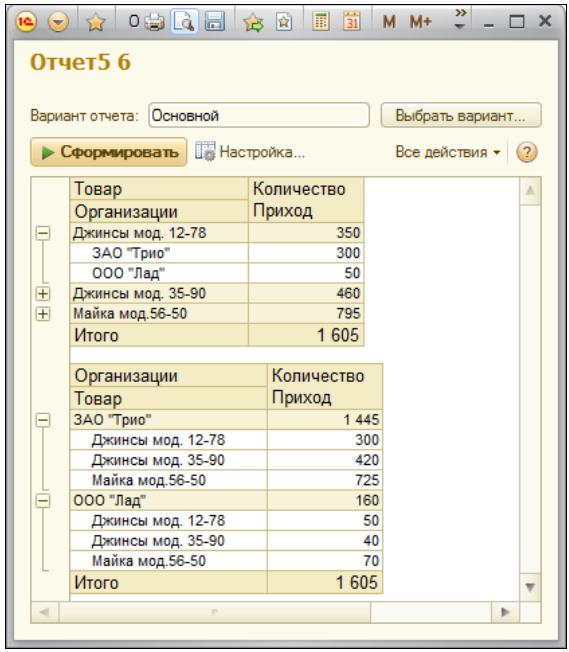


Рис. 5.41. Отчет с использованием двух таблиц

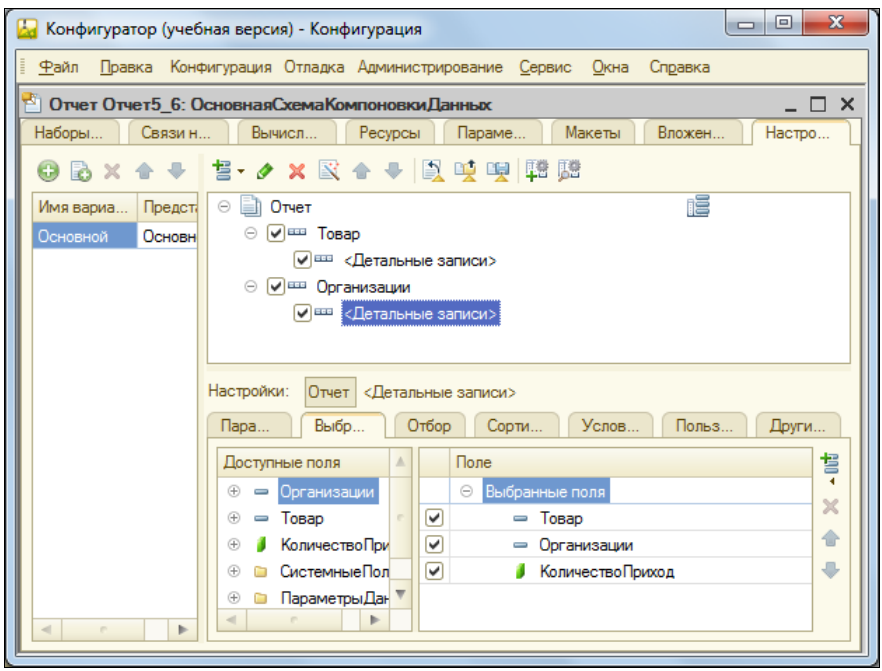


Рис. 5.42. Задание параметров отчета (с использованием двух таблиц) на вкладке **Настройки**

В окне конструктора запроса нам потребуется таблица **Заявка** (рис. 5.43), из которой для запроса необходимо отобрать следующие поля:

- **Ссылка**;
- **Дата**;
- **ФирмаЗаказчик**.

В окне конструктора запроса на вкладке **Объединения/Псевдонимы** установим псевдоним для поля **Ссылка** (рис. 5.44).

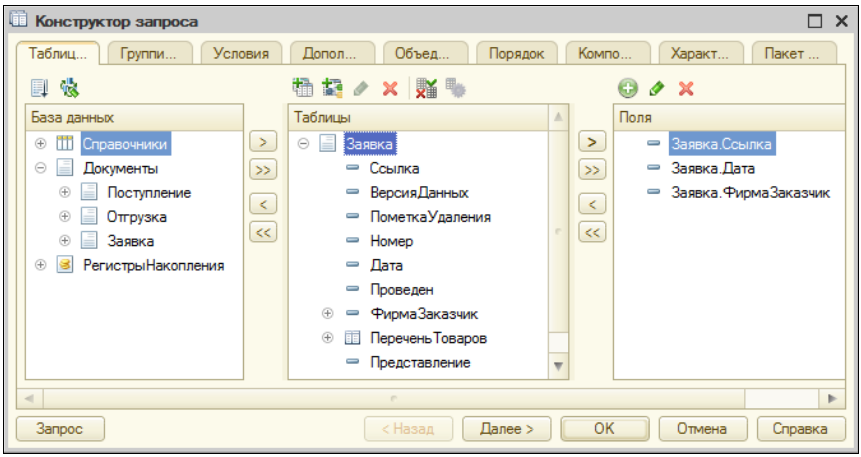


Рис. 5.43. Отбор полей для запроса при формировании отчета по документам "Заявка"

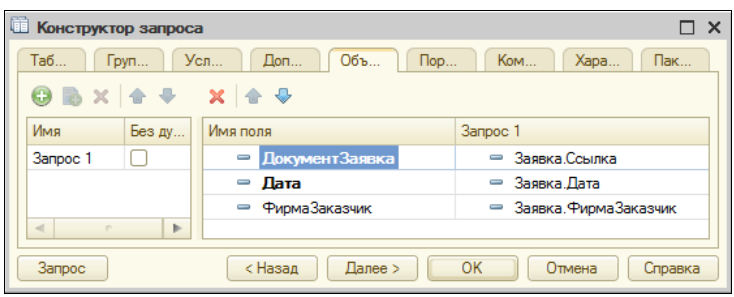


Рис. 5.44. Установка псевдонима для поля **Ссылка**

Далее перейдем на вкладку **Порядок** и укажем, что результат запроса должен быть упорядочен в порядке убывания дат документов (рис. 5.45).

Других установок для запроса делать не будем, и щелчок по кнопке **ОК** приведет к формированию текста необходимого запроса (листинг 5.5).

Листинг 5.5. Запрос для получения информации о документах Заявка

```
ВЫБРАТЬ
  Заявка.Ссылка КАК ДокументЗаявка,
  Заявка.Дата КАК Дата,
  Заявка.ФирмаЗаказчик
```

```
ИЗ
  Документ.Заявка КАК Заявка
УПОРЯДОЧИТЬ ПО
  Дата УБЫВ
```

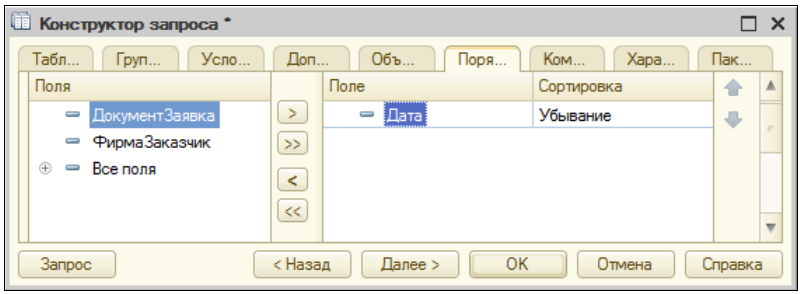


Рис. 5.45. Установка порядка вывода результатов запроса

После формирования текста запроса продолжим подготовку отчета в окне конструктора схемы компоновки данных. Так, перейдем на вкладку **Настройки** и выполним необходимые установки. Укажем, что в группировке будут выводиться детальные записи из информационной базы, и перенесем поля запроса в раздел **Выбранные поля** (рис. 5.46).

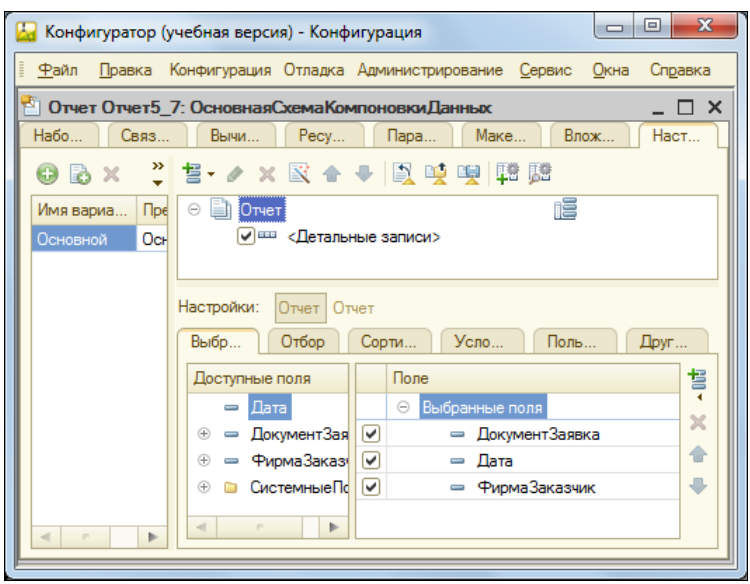


Рис. 5.46. Установки на вкладке **Настройки**
для получения информации о документах

Осталось включить отчет в одну из подсистем, перейти в режим **1С:Предприятие** и сформировать отчет по имеющимся документам "Заявка" в информационной базе (рис. 5.47). Обратим внимание на то, что двойным щелчком мыши по полю в пер-

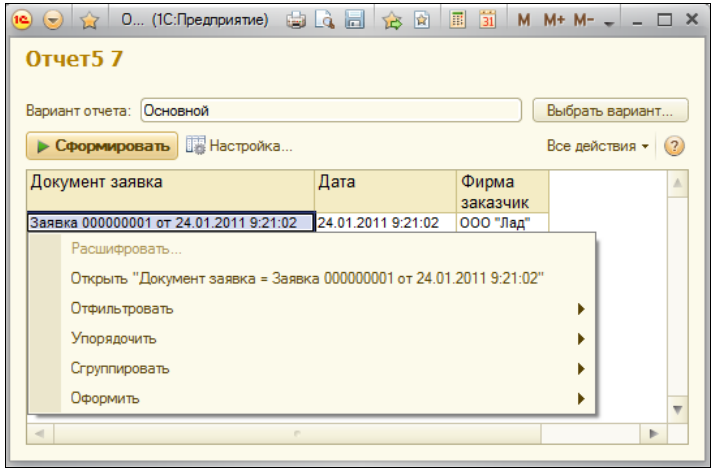


Рис. 5.47. Формирование отчета по заявкам

вой колонке отчета можно вызвать выпадающее меню и открыть интересующий нас документ.

Отчет по заявкам в интервале дат

Рассматриваемый в данном разделе отчет позволит получить информацию о количестве позиций заявок по фирмам-заказчикам. Новый объект конфигурации будет называться **Отчет5_8**. В окне конструктора запроса выберем две таблицы:

- объектную таблицу справочника "Фирмы";
- виртуальную таблицу **ИнформацияОЗаявках.Обороты**.

Теперь из рассматриваемых таблиц в раздел **Поля** перенесем три поля (рис. 5.48):

- **Фирмы.Ссылка**;
- **ИнформацияОЗаявкахОбороты.КоличествоОборот**;
- **ИнформацияОЗаявкахОбороты.Товар**.

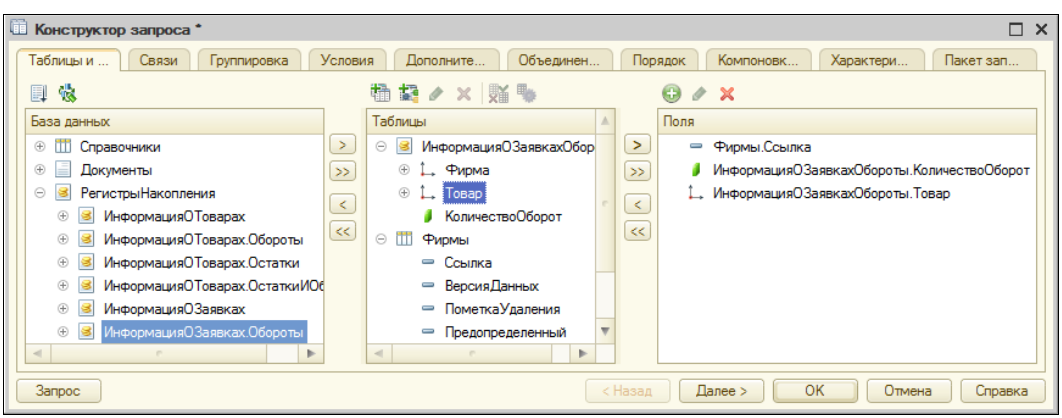


Рис. 5.48. Окно конструктора запроса

В отличие от предыдущих примеров этой главы, сейчас в запросе мы используем две таблицы, которые необходимо связать. Ранее при рассмотрении запросов использовались различные варианты соединения таблиц, а также синтаксические конструкции языка запросов, позволяющих это реализовать. В данном случае подобные установки реализуются визуальными действиями. Так, на вкладке **Связи** следует убедиться в установке необходимой связи между двумя таблицами (рис. 5.49). Значение измерения **Фирма** виртуальной таблицы регистра накопления должно быть равно ссылке на элемент справочника "Фирмы". Обязательно требуется проверить установку флажка **Все** у таблицы справочника. В этом случае из справочника будут выбраны все элементы справочника, и этим элементам будет поставлено в соответствие значение оборота по данной организации из регистра. Таким образом, в результате выполнения запроса в отчете будут присутствовать все фирмы, а для ряда из них (которые делали заявки на товары) будут указаны значения оборотов.

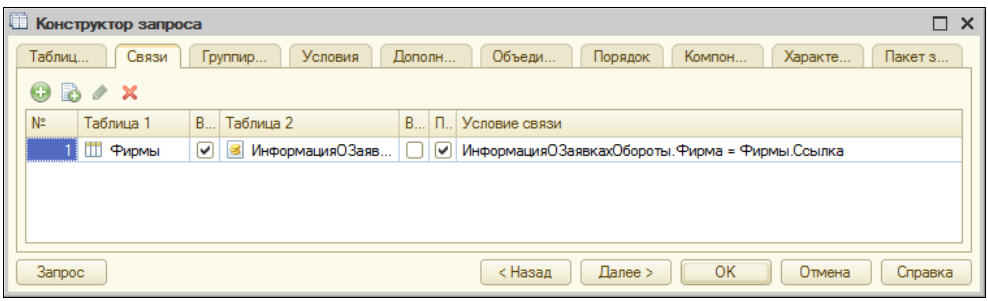


Рис. 5.49. Установка связей между двумя таблицами в запросе

Перейдем на вкладку **Объединения/Псевдонимы**, где установим псевдоним **Организация** вместо **Фирмы.Ссылка** (рис. 5.50), т. к. при работе для пользователя слово "Организация" намного информативнее.

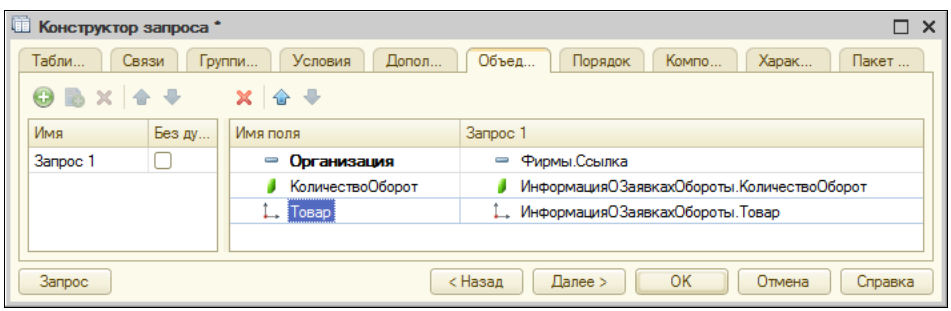


Рис. 5.50. Вкладка **Объединения/Псевдонимы**

На вкладке **Порядок** укажем (рис. 5.51), что результат должен быть отсортирован сначала по фирмам, а затем по значению поля **КоличествоОборот** в порядке убывания.

На этом визуальные действия по конструированию запроса завершены, и осталось щелкнуть по кнопке **ОК**, чтобы получить текст запроса (листинг 5.6).

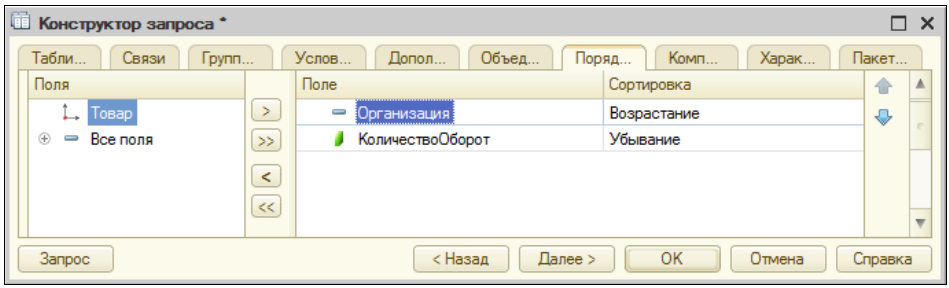


Рис. 5.51. Установка порядка вывода результатов запроса

Листинг 5.6. Запрос для получения информации о сделанных заказах

```
ВЫБРАТЬ
        Фирмы.Ссылка КАК Организация,
        ИнформацияОЗаявкахОбороты.КоличествоОборот КАК КоличествоОборот,
        ИнформацияОЗаявкахОбороты.Товар
ИЗ
        Справочник.Фирмы КАК Фирмы
ЛЕВОЕ СОЕДИНЕНИЕ РегистрНакопления.ИнформацияОЗаявках.Обороты
     КАК ИнформацияОЗаявкахОбороты
ПО ИнформацияОЗаявкахОбороты.Фирма = Фирмы.Ссылка
УПОРЯДОЧИТЬ ПО
        Организация,
        КоличествоОборот УБЫВ
```

В данном случае в запросе используется уже знакомое `ЛЕВОЕ` соединение таблиц, о котором мы говорили в *главе 4*.

Перейдем теперь к установкам в окне конструктора схемы компоновки данных. Так, на вкладке **Параметры** (рис. 5.52) уже указаны два параметра. Это связано с тем, что система компоновки данных самостоятельно анализирует текст запроса и указывает те параметры, которые, скорее всего, будут использоваться в отчете.

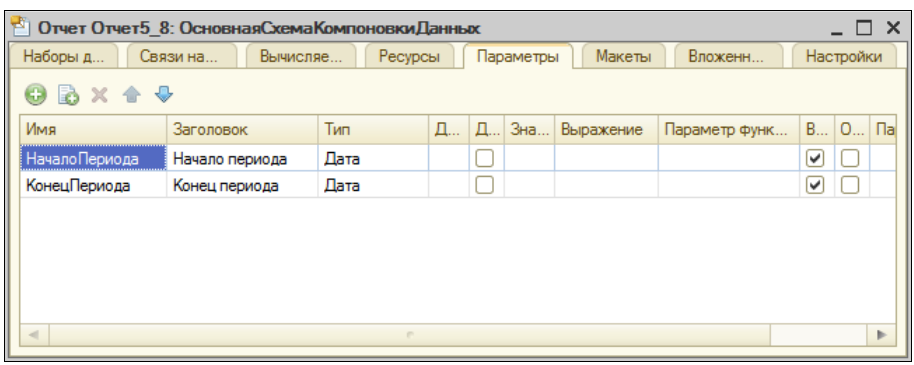


Рис. 5.52. Вкладка для определения параметров отчета

Присутствующие здесь **НачалоПериода** и **КонецПериода** являются параметрами виртуальной таблицы **РегистрНакопления.ИнформацияОЗаказах.Обороты**. Первый параметр представляет собой начало периода расчета итогов, второй — конец периода. В результате исходная таблица будет содержать обороты, рассчитанные в переданном интервале дат.

Важно отметить, что тип данных **Дата**, кроме непосредственно даты, содержит еще и время. Разумеется, при указании интервала дат присутствие времени является излишним. Для необходимой коррекции (исключения конкретного времени в указании даты) нужно дважды щелкнуть в графе **Тип** параметра **НачалоПериода** и с помощью пиктограммы с изображением многоточия открыть окно **Редактирование типа данных** (рис. 5.53). Здесь в качестве состава даты необходимо указать вариант — **Дата.**

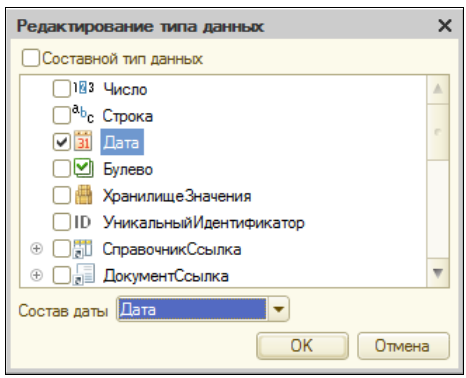


Рис. 5.53. Окно для редактирования типа данных

Особенностью работы системы является то, что по умолчанию время в дате установлено 00:00:00. В связи с этим, если пользователь задаст период расчета, скажем, до 20.02.08, то данные на этот день (время создания которых позже 00:00:00) в расчет не войдут. Поэтому при задании интервала по дате при формировании отчета это надо учитывать.

Примечание

В [6] приведен пример, в котором с помощью программных конструкций реализуется включение в результат информации по указанную дату включительно.

Аналогичные установки выполним и для параметра **КонецПериода.**

Обратим внимание еще на одну важную установку. Так, в окне на рис. 5.52 присутствуют флажки **Ограничения доступности** (для двух параметров). Для того чтобы можно было устанавливать значения рассматриваемых параметров, пользователю эти флажки необходимо снять.

Перейдем теперь на вкладку **Настройки**, где добавим группировку без указания поля группировки. Здесь же на дополнительной вкладке **Выбранные поля** (рис. 5.54) внесем поля: **Организация, Товар** и **КоличествоОборот.**

Дополним отчет установкой интервала по дате. Для этого обратимся к дополнительной вкладке **Параметры** (рис. 5.55). Также включим рассматриваемые пара-

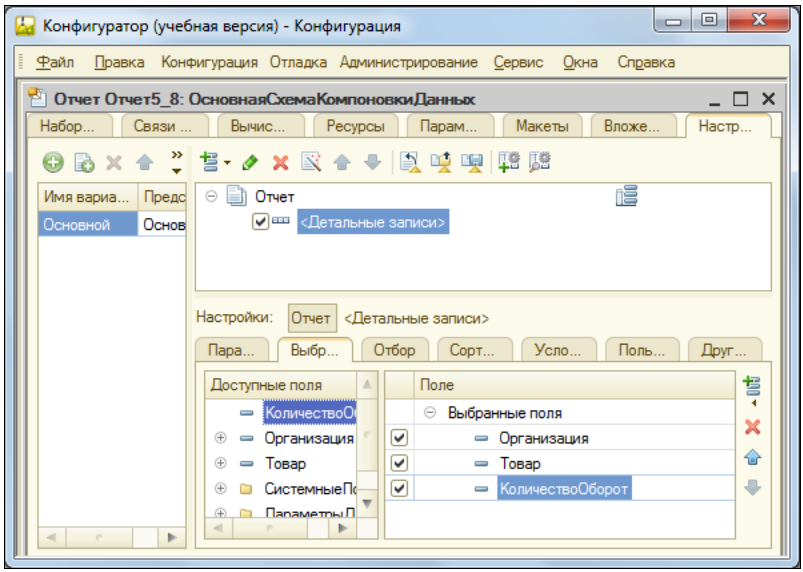


Рис. 5.54. Настройки отчета по заказам

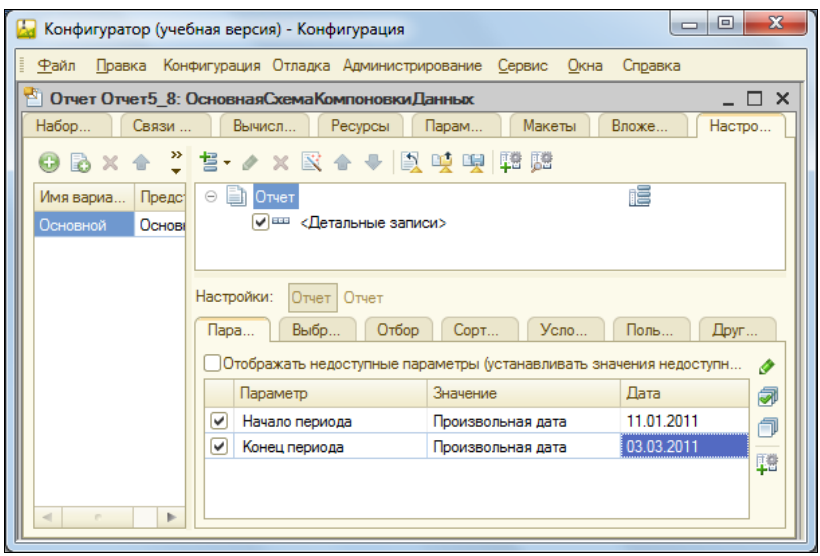


Рис. 5.55. Настройка параметров отчета

метры в пользовательские настройки. Для этого воспользуемся пиктограммой **Свойства элемента пользовательских настроек** (в вертикальном меню она занимает самое нижнее положение). В результате на экране откроется новое окно, где требуется выполнить необходимые установки (рис. 5.56).

После включения отчета в одну из подсистем можно перейти в режим **1С:Предприятие** и посмотреть отчет в работе (рис. 5.57). Теперь пользователь может сам задавать интервал по дате в форме отчета.

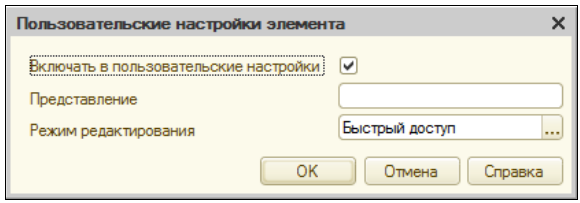


Рис. 5.56. Окно пользовательских настроек элемента

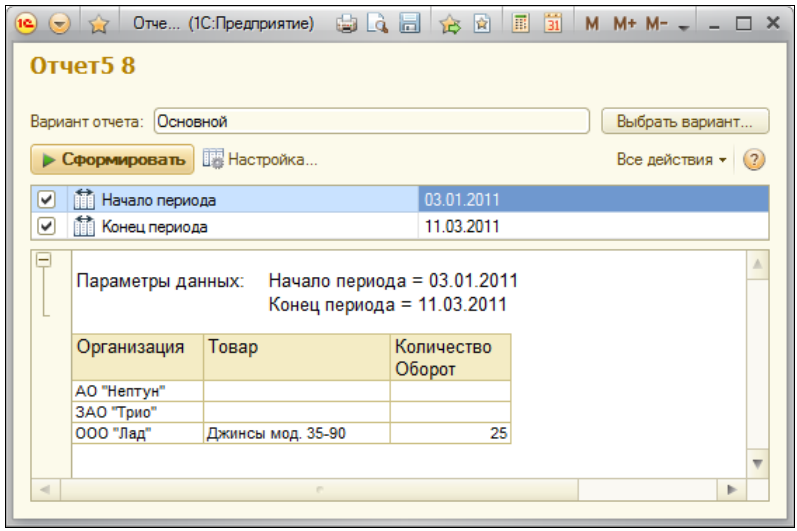


Рис. 5.57. Отчет по заявкам в режиме **1С:Предприятие**

Отчеты по информационной базе *главы 3*

Перейдем теперь к рассмотрению конфигурации, разработанной в *главе 3*. В ней был предложен вариант организации учета оплаты за обучение. В связи с этим последующие действия переносятся в конфигурацию, разработанную в *главе 3*.

Отчет по квитанциям

Для логического продолжения созданной в *главе 3* разработки добавим в нее отчет по квитанциям. Поэтому в режиме **Конфигуратор** откроем знакомую информационную базу и начнем создание нового отчета, который назовем **Отчет5_9** (рис. 5.58).

В окне конструктора запроса (рис. 5.59) воспользуемся таблицей **Квитанции** (из ветки **РегистрыСведений**) и в раздел **Поля** перенесем:

- **Договор**;
- **СуммаПоКвитанции**;
- **МесяцНачала**;

- **ГодНачала**;
- **ОплаченныеМесяцы**;
- **Оплачено**.

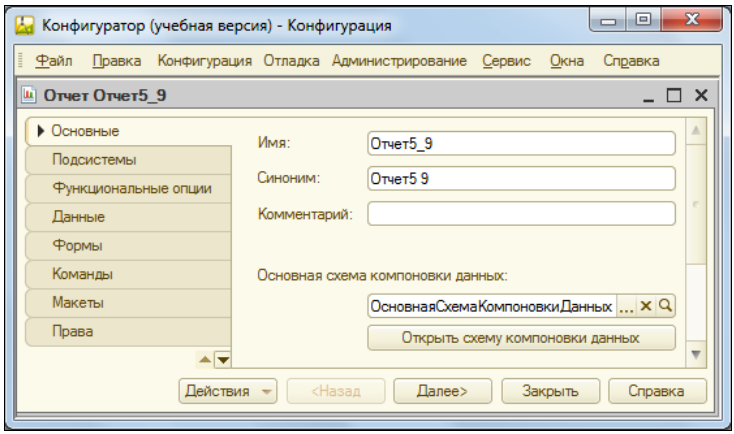


Рис. 5.58. Отчет по квитанциям в режиме **Конфигуратор**

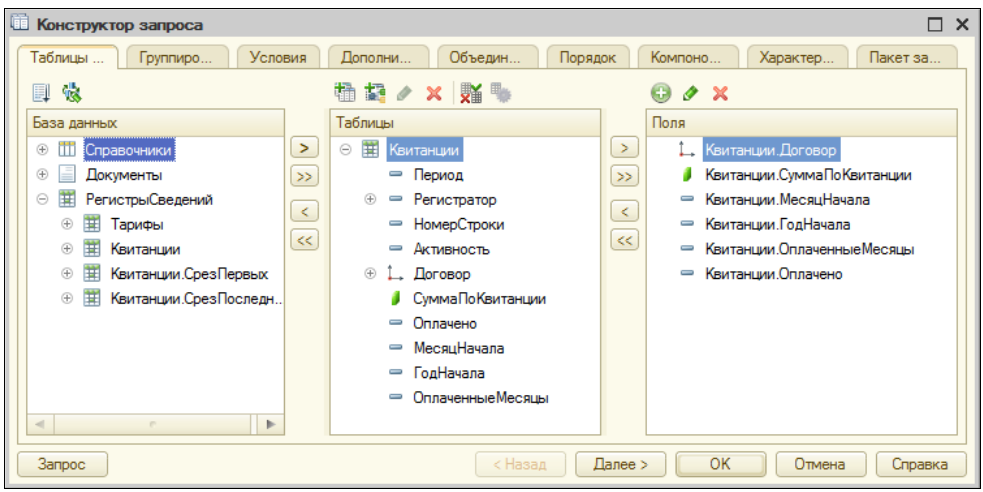


Рис. 5.59. Окно конструктора запроса для выбора полей в запросе

На вкладке **Порядок** установим порядок сортировки сначала по договорам, а затем по сумме в квитанции (рис. 5.60). На этом визуальное проектирование запроса завершено и можно нажать кнопку **ОК**, что приведет к формированию запроса (листинг 5.7).

Листинг 5.7. Запрос для получения информации о квитанциях

```
ВЫБРАТЬ
        Квитанции.Договор КАК Договор,
        Квитанции.СуммаПоКвитанции КАК СуммаПоКвитанции,
```

```
        Квитанции.МесяцНачала,
        Квитанции.ГодНачала,
        Квитанции.ОплаченныеМесяцы,
        Квитанции.Оплачено
ИЗ
        РегистрСведений.Квитанции КАК Квитанции
УПОРЯДОЧИТЬ ПО
        Договор,
        СуммаПоКвитанции
```

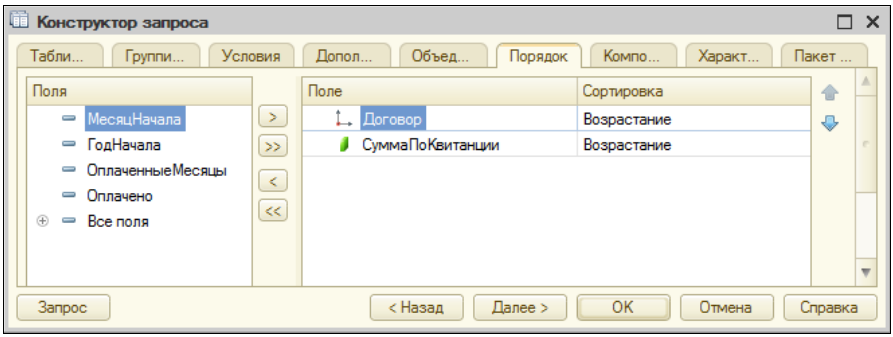


Рис. 5.60. Добавление сортировки в объект **Отчет5_9**

Продолжим доработку отчета в окне конструктора схемы компоновки данных. На вкладке **Ресурсы** установим для нашего отчета ресурс **СуммаПоКвитанции** (рис. 5.61).

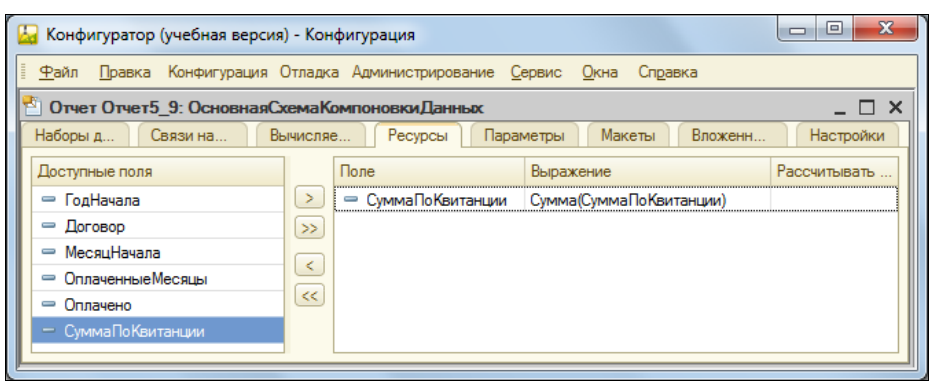


Рис. 5.61. Вкладка **Ресурсы** окна конструктора схемы компоновки данных

Далее следует перейти на вкладку **Настройки** (рис. 5.62), где требуется сформировать группировку по полю **Оплачено**. Таким образом, в отчете мы будем отделять оплаченные квитанции от просто выписанных.

На этом разработку отчета можно считать завершенной, далее перейдем в режим **1С:Предприятие** и сформируем разработанный отчет (рис. 5.63).

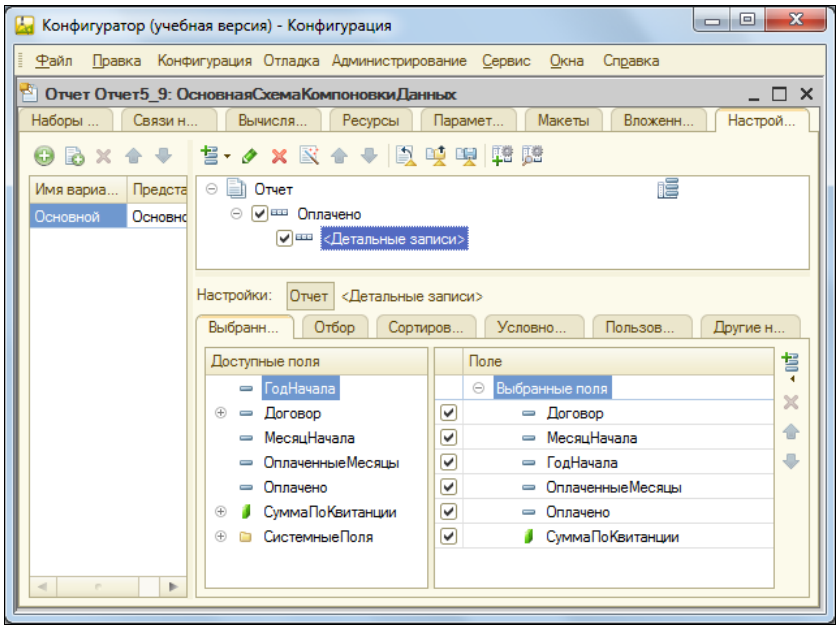
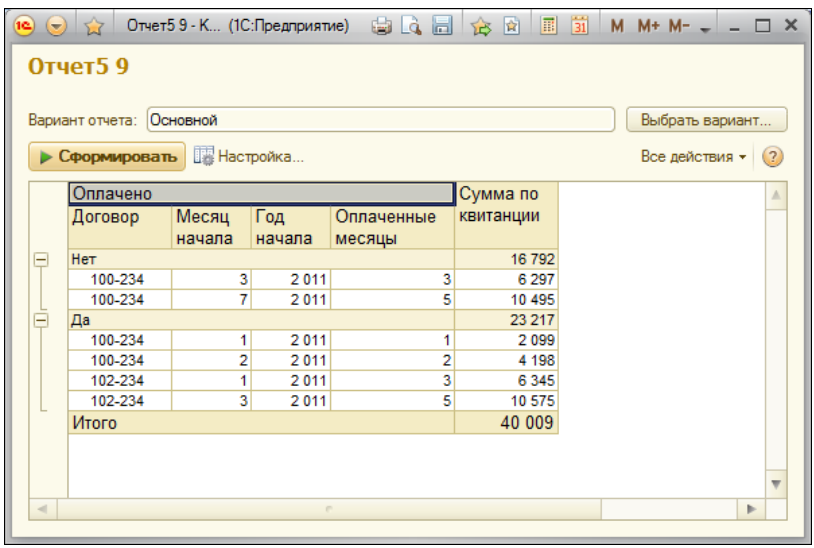


Рис. 5.62. Вкладка **Настройки** окна конструктора схемы компоновки данных



Оплачено				Сумма по квитанции
Договор	Месяц начала	Год начала	Оплаченные месяцы	
Нет				16 792
100-234	3	2 011	3	6 297
100-234	7	2 011	5	10 495
Да				23 217
100-234	1	2 011	1	2 099
100-234	2	2 011	2	4 198
102-234	1	2 011	3	6 345
102-234	3	2 011	5	10 575
Итого				40 009

Рис. 5.63. Отчет по различным категориям квитанций

Отчет по итоговой оплате договоров

Составим отчет для получения информации об итоговой оплате по договорам. Имя отчета выберем — **Отчет5_10**. После обычных начальных действий первые отличительные шаги мы сделаем в окне конструктора запроса (рис. 5.64). В данном случае нам для отчета достаточно одной таблицы — регистра сведений **Квитанции**.

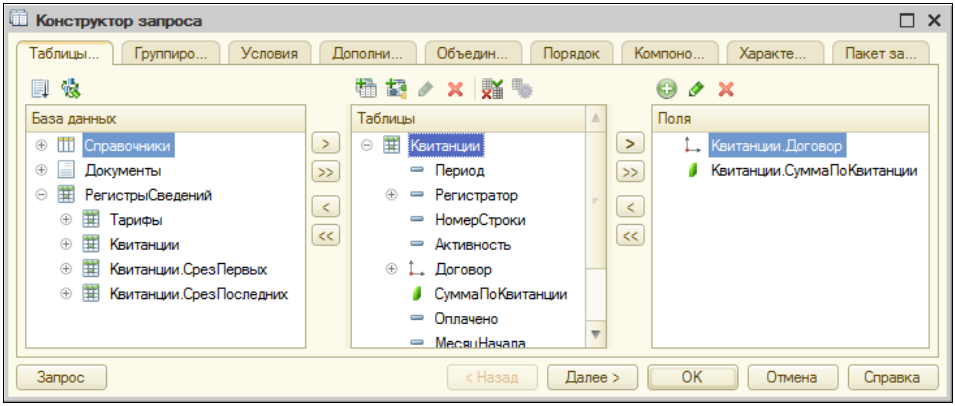


Рис. 5.64. Окно конструктора запроса

На вкладке **Группировка** укажем в качестве группового поля **Квитанции.Договор** (рис. 5.65). Это позволит уже в самом запросе сгруппировать оплаты по каждому договору. Если нам нет необходимости просматривать отдельно квитанции, то такой вариант вполне подходит.

Далее перейдем на вкладку **Условия** и добавим условие для поля **Оплачено** (рис. 5.66). На этом действия в окне конструктора запросов завершены и после щелчка по кнопке **ОК** сформируется текст запроса (листинг 5.8).

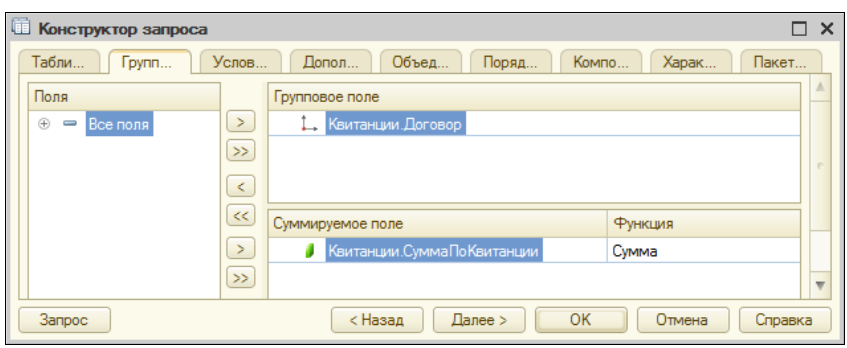


Рис. 5.65. Выбор поля для группировки в запросе

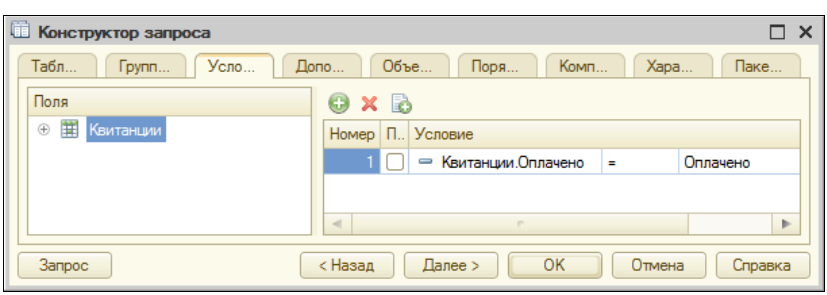


Рис. 5.66. Вкладка **Условия** окна конструктора запроса

Листинг 5.8. Запрос для получения итоговой информации об оплате договоров

```
ВЫБРАТЬ
        Квитанции.Договор,
        СУММА(Квитанции.СуммаПоКвитанции) КАК СуммаПоКвитанции
ИЗ
        РегистрСведений.Квитанции КАК Квитанции
ГДЕ
        Квитанции.Оплачено = &Оплачено
СГРУППИРОВАТЬ ПО
        Квитанции.Договор
```

Продолжим дальнейшие действия по формированию отчета в окне конструктора схемы компоновки данных. Так, на вкладке **Параметры** (рис. 5.67) следует установить условие отбора — только для оплаченных квитанций.

Осталось выполнить стандартные установки на вкладке **Настройки** (рис. 5.68), после чего можно перейти в режим **1С:Предприятие** и получить готовый отчет (рис. 5.69).

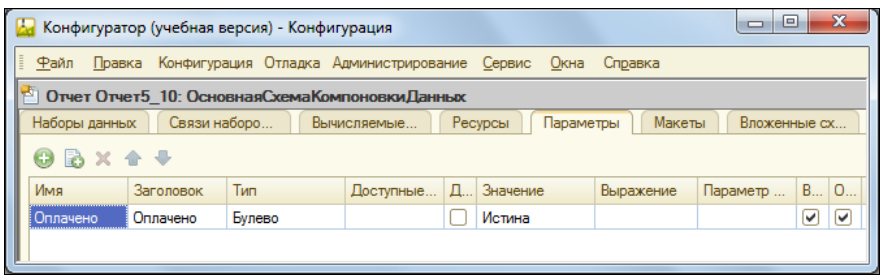


Рис. 5.67. Вкладка **Параметры** окна конструктора схемы компоновки данных

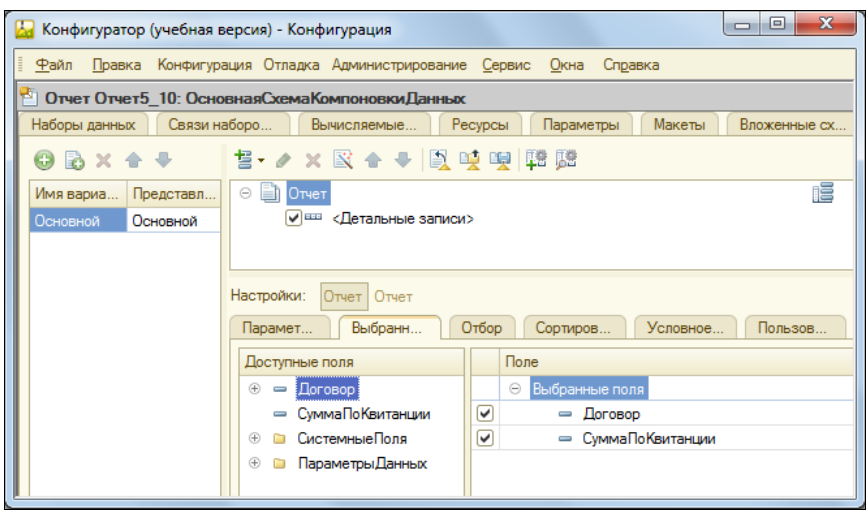


Рис. 5.68. Установки на вкладке **Настройки**

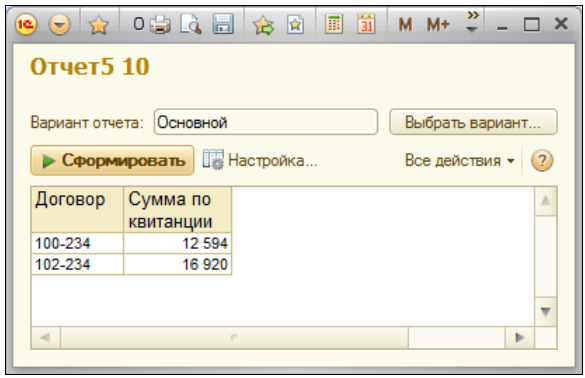


Рис. 5.69. Отчет по итоговой оплате договоров

Отчет по квитанциям с группировкой

Рассмотрим создание отчета, представленного на рис. 5.70. Он позволяет увидеть как суммарную оплату по конкретному договору, так и детализацию оплат за обучение.

Учитывая уже сформированный опыт работы с системой компоновки данных, мы не будем детально описывать этапы создания отчета, а поясним лишь ключевые шаги, которые необходимо выполнить. Так, на рис. 5.71 показано окно конструктора запроса с указанием полей, которые нам потребуются для отчета. Далее перейдем на вкладку **Условия** и включим в запрос условие по полю **Оплачено** (рис. 5.72), а на вкладке **Порядок** определим упорядочивание по договорам. После этого щелчок по кнопке **ОК** приведет к формированию текста запроса (листинг 5.9) и возврату в окно конструктора схемы компоновки данных.

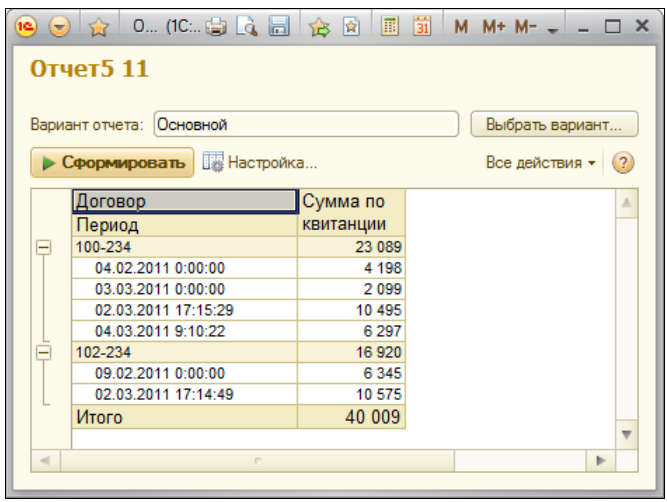


Рис. 5.70. Отчет по квитанциям с группировкой

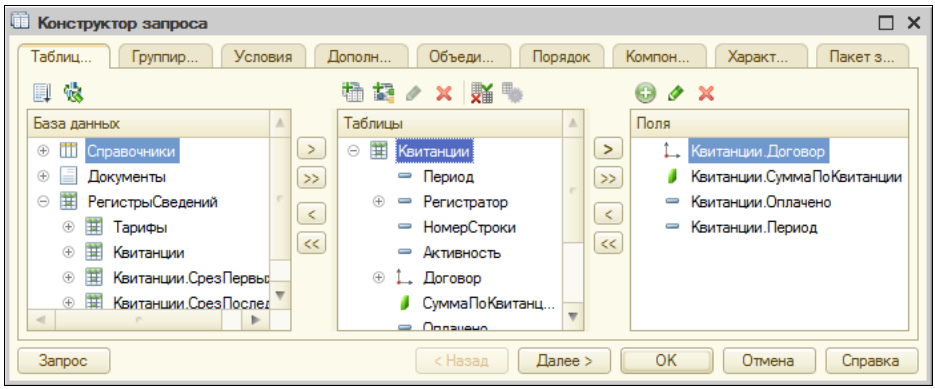


Рис. 5.71. Окно конструктора запроса

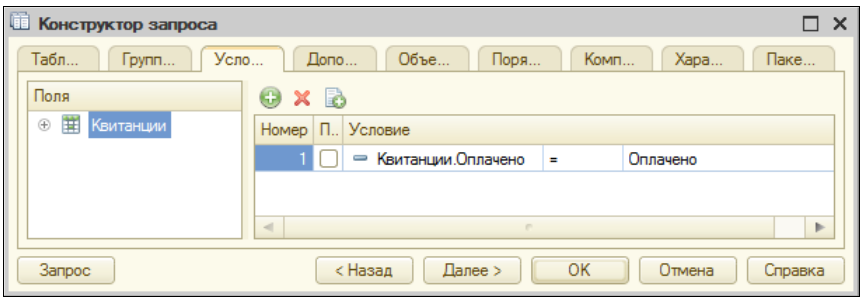


Рис. 5.72. Вкладка **Условия** окна конструктора запроса

Листинг 5.9. Запрос для получения информации об оплате

```
ВЫБРАТЬ
        Квитанции.Договор КАК Договор,
        Квитанции.СуммаПоКвитанции,
        Квитанции.Оплачено,
        Квитанции.Период
ИЗ
        РегистрСведений.Квитанции КАК Квитанции
ГДЕ
        Квитанции.Оплачено = &Оплачено

УПОРЯДОЧИТЬ ПО
        Договор
```

На вкладке **Ресурсы** окна конструктора схемы компоновки данных необходимо включить ресурс **СуммаПоКвитанции** (рис. 5.73). После этого перейдем на вкладку **Настройки**, где выберем поля и установим группировки в соответствии с рис. 5.74. В режиме **1С:Предприятие** разработанный отчет показан на рис. 5.70.

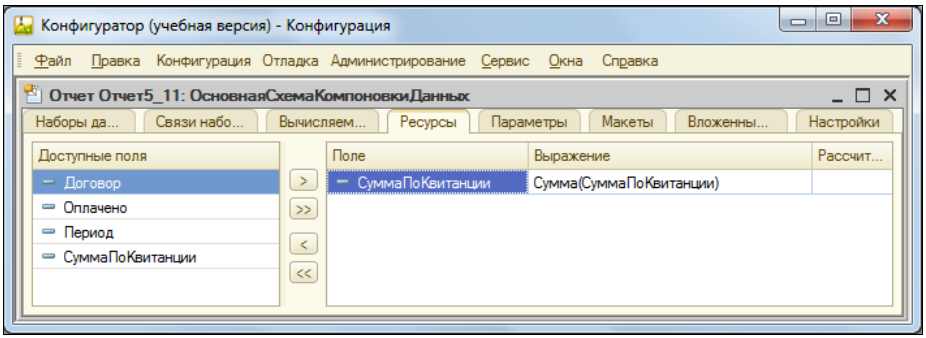


Рис. 5.73. Вкладка **Ресурсы** окна конструктора схемы компоновки данных

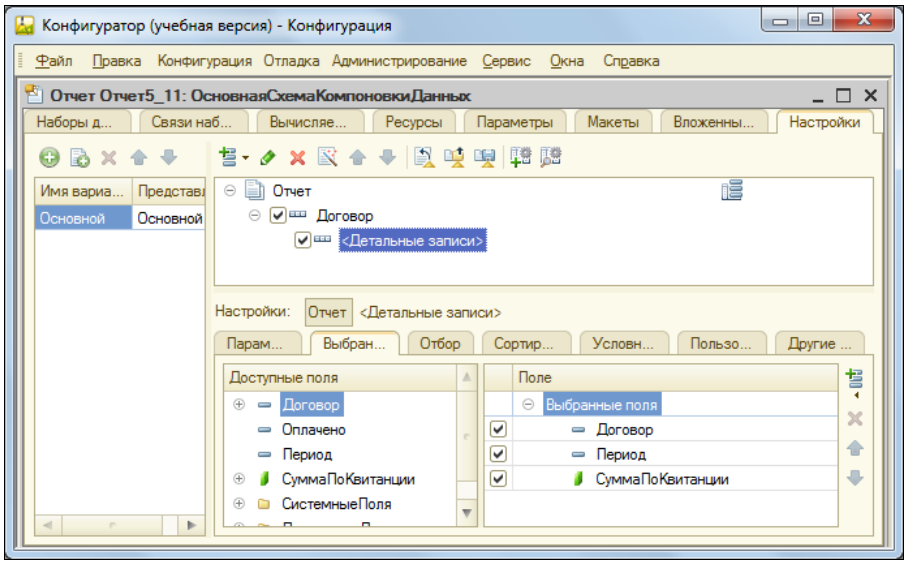


Рис. 5.74. Вкладка окна конструктора схемы компоновки данных для создания настроек отчета

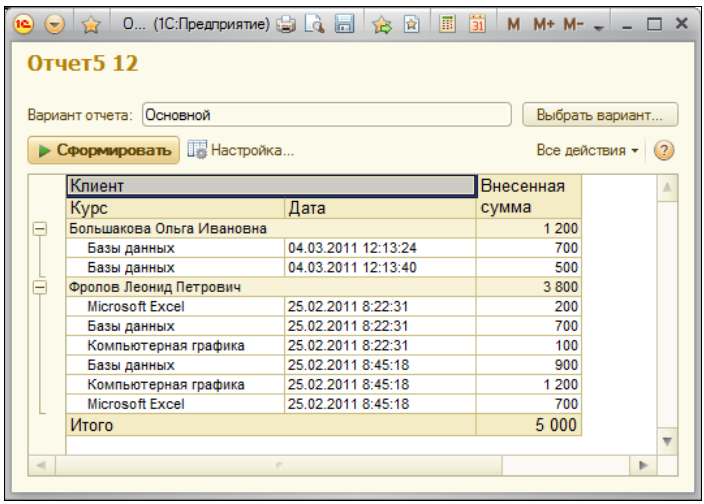
Отчеты по информационной базе *главы 1*

Рассмотрим еще несколько примеров отчетов, которые можно создать с использованием системы компоновки данных. Они будут касаться конфигурации, разработанной в *главе 1*. В ней был рассмотрен учет слушателей учебных курсов. В связи с этим последующие действия переносятся в конфигурацию, разработанную в *главе 1*.

Отчет по учащимся

На рис. 5.75 приведен отчет, который позволяет просмотреть как общие оплаченные суммы, так и увидеть детализацию сделанных оплат. Для начала нам потребуется с помощью конструктора запросов сформировать необходимый запрос (рис. 5.76). Здесь мы воспользуемся двумя таблицами: таблицей документа **Внесе-**

ниеОплаты и табличной частью **ОплатаКурсов** этого же документа. В окне конструктора документа на вкладке **Связи** следует установить внутреннее соединение данных таблиц (рис. 5.77). После этого щелчком по кнопке **ОК** можно сформировать текст запроса (листинг 5.10), который позволит извлечь необходимую информацию.



Клиент / Курс	Дата	Внесенная сумма
Большакова Ольга Ивановна		1 200
Базы данных	04.03.2011 12:13:24	700
Базы данных	04.03.2011 12:13:40	500
Фролов Леонид Петрович		3 800
Microsoft Excel	25.02.2011 8:22:31	200
Базы данных	25.02.2011 8:22:31	700
Компьютерная графика	25.02.2011 8:22:31	100
Базы данных	25.02.2011 8:45:18	900
Компьютерная графика	25.02.2011 8:45:18	1 200
Microsoft Excel	25.02.2011 8:45:18	700
Итого		5 000

Рис. 5.75. Отчет по учащимся

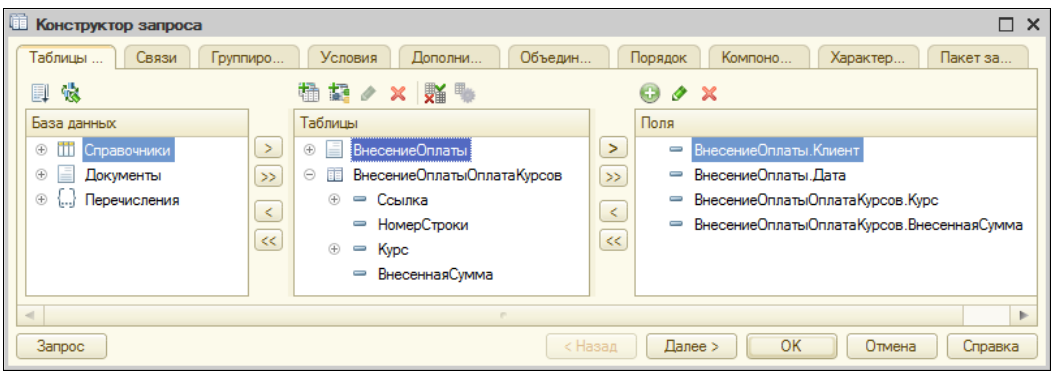


Рис. 5.76. Окно конструктора запроса

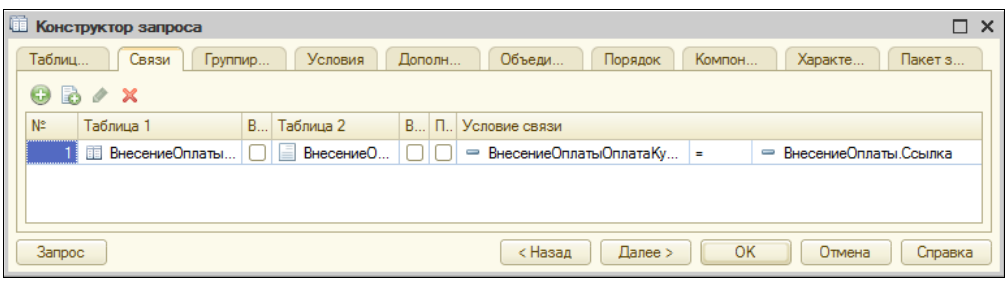


Рис. 5.77. Вкладка **Связи** окна конструктора запроса

Листинг 5.10. Запрос для получения информации о слушателях курсов

```
ВЫБРАТЬ
  ВнесениеОплаты.Клиент,
  ВнесениеОплаты.Дата,
  ВнесениеОплатыОплатаКурсов.Курс,
  ВнесениеОплатыОплатаКурсов.ВнесеннаяСумма
ИЗ
  Документ.ВнесениеОплаты.ОплатаКурсов
    КАК ВнесениеОплатыОплатаКурсов
ВНУТРЕННЕЕ СОЕДИНЕНИЕ Документ.ВнесениеОплаты
   КАК ВнесениеОплаты
ПО ВнесениеОплатыОплатаКурсов.Ссылка = ВнесениеОплаты.Ссылка
```

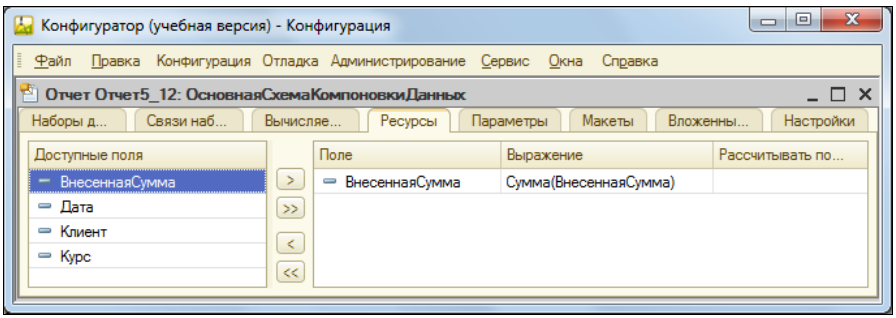


Рис. 5.78. Вкладка **Ресурсы** окна конструктора схемы компоновки данных

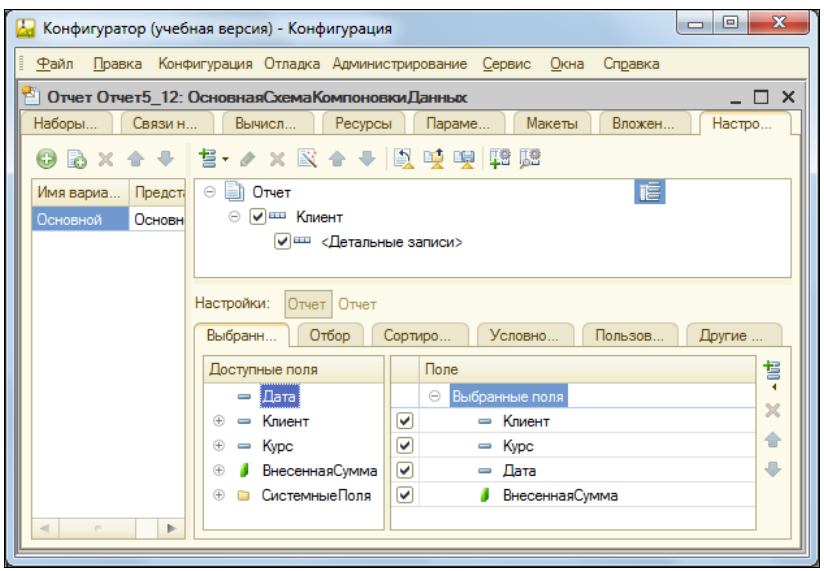
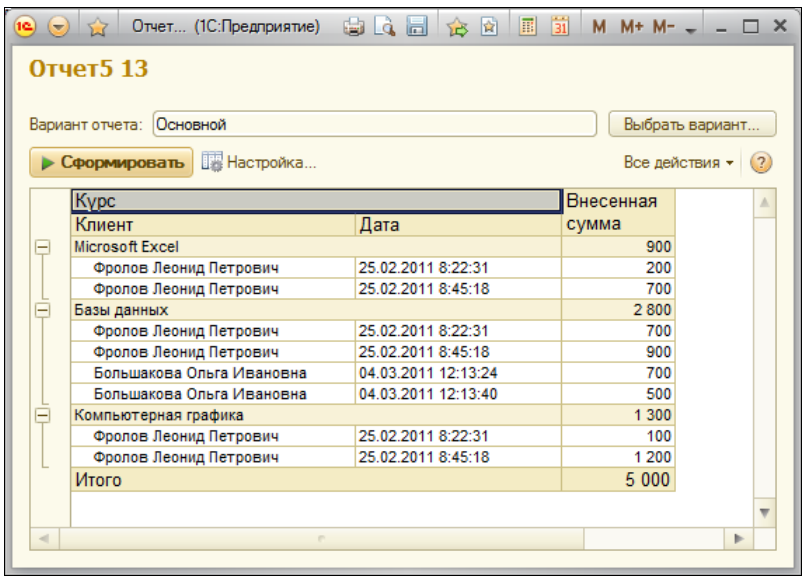


Рис. 5.79. Вкладка **Настройки** окна конструктора схемы компоновки данных

Перейдем теперь к подготовке отчета в окне конструктора схемы компоновки данных. Нам понадобится указать ресурс для отчета (рис. 5.78) и сформировать установки на вкладке **Настройки** (рис. 5.79). Уже после этого можно перейти в режим **1С:Предприятие** и получить отчет по оплатам (см. рис. 5.75).

Отчет по курсам

На рис. 5.80 приведен отчет, который позволяет просмотреть информацию по оплатам за курсы (видно, кто и сколько внес денежных средств). Для начала нам потребуется с помощью конструктора запросов сформировать необходимый запрос (рис. 5.81). Данный запрос аналогичен тому, который нам встретился в предыдущем разделе. Текст запроса приведен в листинге 5.11.



Курс		Внесенная сумма
Клиент	Дата	
Microsoft Excel		900
Фролов Леонид Петрович	25.02.2011 8:22:31	200
Фролов Леонид Петрович	25.02.2011 8:45:18	700
Базы данных		2 800
Фролов Леонид Петрович	25.02.2011 8:22:31	700
Фролов Леонид Петрович	25.02.2011 8:45:18	900
Большакова Ольга Ивановна	04.03.2011 12:13:24	700
Большакова Ольга Ивановна	04.03.2011 12:13:40	500
Компьютерная графика		1 300
Фролов Леонид Петрович	25.02.2011 8:22:31	100
Фролов Леонид Петрович	25.02.2011 8:45:18	1 200
Итого		5 000

Рис. 5.80. Отчет по оплатам за курсы

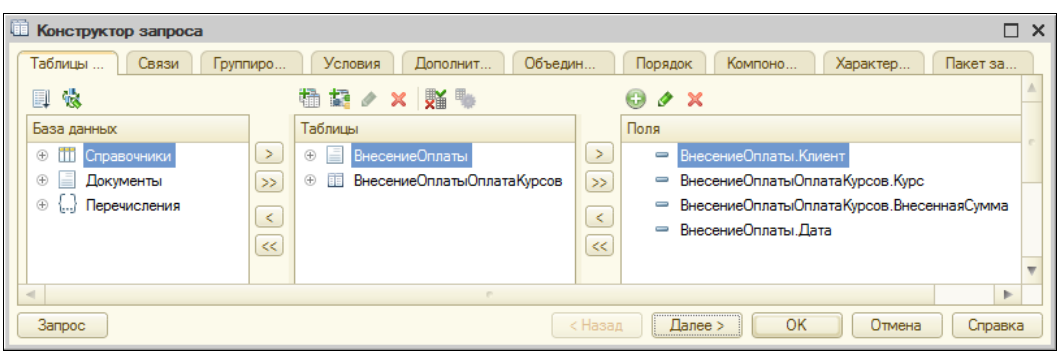


Рис. 5.81. Окно конструктора запроса

Листинг 5.11. Запрос для получения информации об оплатах курсов

```
ВЫБРАТЬ
  ВнесениеОплаты.Клиент,
  ВнесениеОплатыОплатаКурсов.Курс,
  ВнесениеОплатыОплатаКурсов.ВнесеннаяСумма,
  ВнесениеОплаты.Дата
ИЗ
  Документ.ВнесениеОплаты.ОплатаКурсов
  КАК ВнесениеОплатыОплатаКурсов
ВНУТРЕННЕЕ СОЕДИНЕНИЕ Документ.ВнесениеОплаты
  КАК ВнесениеОплаты
  ПО ВнесениеОплатыОплатаКурсов.Ссылка = ВнесениеОплаты.Ссылка
```

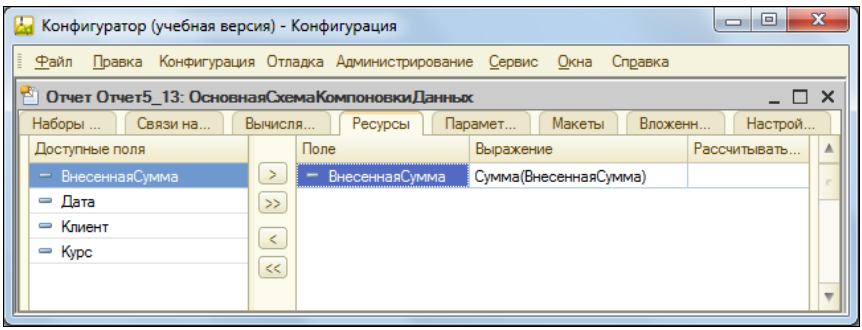


Рис. 5.82. Вкладка **Ресурсы** окна конструктора схемы компоновки данных

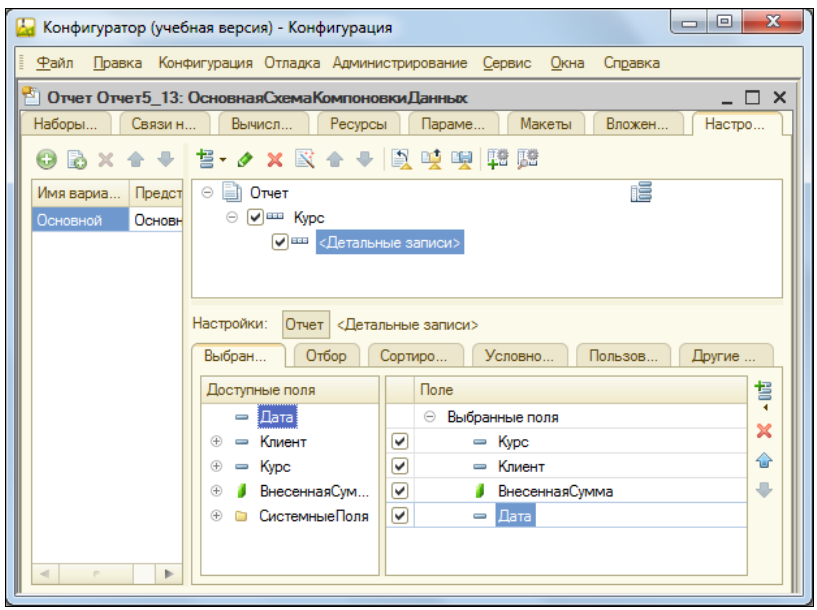


Рис. 5.83. Вкладка **Настройки** окна конструктора схемы компоновки данных

Перейдем теперь к подготовке отчета в окне конструктора схемы компоновки данных. Нам понадобится указать ресурс для отчета (рис. 5.82) и сформировать установки на вкладке **Настройки** (рис. 5.83). Уже после этого можно перейти в режим **1С:Предприятие** и получить отчет по оплатам (см. рис. 5.80).

Итоги

В этой главе мы рассмотрели систему компоновки данных, которая является мощным ресурсом для создания отчетов. Используя данную технологию, можно получать сложные отчеты по информационной базе без написания программного кода на встроенном языке программирования. При этом система компоновки позволяет пользователю выполнять существенную перенастройку отчетов уже в режиме **1С:Предприятие**.

Механизм компоновки данных позволяет дать целостное описание отчета только за счет визуальных средств. Это отличает систему компоновки данных от построителя отчетов (основного средства автоматизации при создании отчетов в системе 1С:Предприятие 8.0).

ГЛАВА 6

Программирование форм в прикладных задачах

К настоящему моменту рассмотрено большое количество практических примеров, дающих представление о технологии разработки конфигураций для платформы 1С:Предприятие 8.2. В предыдущих главах мы работали с такими объектами конфигурации, как перечисления, справочники, документы, отчеты, регистры сведений и регистры накопления. Рассмотрев большое количество программных процедур, мы познакомились с объектами системы 1С:Предприятие, с их свойствами и методами. Большое внимание было уделено подчиненным объектам конфигурации — формам, табличным частям, макетам, командам и др. Также была рассмотрена система компоновки данных — основной инструмент для формирования отчетов. Все это должно сформировать у читателей цельное представление о разработке прикладных решений на платформе 1С:Предприятие 8.2.

Однако в данную систему заложены практически неисчерпаемые ресурсы для разработчиков, и к настоящему моменту уже выпущено несколько изданий, раскрывающих секреты мастерства профессионального разработчика [6—8]. В этой главе мы рассмотрим ряд примеров, объединяющим моментом которых является использование в них программирования форм. Эта тема затрагивалась на протяжении всех глав книги и здесь она получит дальнейшее развитие.

Программная организация форм

Форма представляет собой очень сложный объект в плане внутреннего устройства. Основная особенность формы, как программного объекта, заключается в том, что она присутствует и на сервере, и на клиенте. В связи с этим все процедуры и функции, расположенные в модуле формы, должны включать директиву (указание) о том, где платформа должна выполнять тот или иной фрагмент программного кода (на сервере или на клиенте).

У формы имеется набор реквизитов. Для просмотра имеющихся или добавления новых реквизитов предназначена вкладка **Реквизиты** окна редактора форм.

Во всех рассматриваемых в книге примерах используются формы основных объектов конфигурации. Данные объекта, которые отображаются в форме, всегда нахо-

дятся в основном реквизите формы. Основной реквизит формы содержит набор подчиненных реквизитов, соответствующих реквизитам прикладного объекта. Например, для обращения к реквизиту документа требуется сначала указать имя основного реквизита формы — `Объект`, а затем через точку имя подчиненного реквизита. Это может выглядеть так:

```
Объект.Заказчик
```

Мы уже познакомились на примерах с контекстными и внеконтекстными серверными вызовами при работе с формой. Это является основной составляющей при программировании форм.

При вызове из клиентской процедуры (функции) серверной процедуры (функции) происходит передача всей формы на сервер. В этом случае контекст формы (реквизиты, элементы формы и параметры формы) упаковываются и передаются на сервер. После этого производится выполнение той серверной процедуры (функции), которая была вызвана с клиента. Контекстный вызов формы представляет собой достаточно затратный процесс. Существует много ситуаций, когда необходимо выполнять на сервере определенные действия, но передавать контекст формы не нужно. Для этого используются внеконтекстные серверные вызовы. В этом случае из модуля клиентской части формы вызывается серверная процедура с директивой компиляции `НаСервереБезКонтекста`. В результате выполнение кода сразу же передается на сервер из модуля клиентской части формы во внеконтекстную процедуру (функцию). Внеконтекстный вызов существенно производительней и его рекомендуется использовать всегда, когда это возможно.

Учет начислений преподавателям

Первая задача, которую мы рассмотрим, будет связана с необходимостью учета начислений преподавателям за проведенные занятия. В *главе 3* мы рассматривали задачу учета оплаты за обучение в организации, предоставляющей платные образовательные услуги. Можно сказать, что тематика первой разработки данной главы будет в определенной степени похожа на ту разработку.

Очевидно, что во всех учебных заведениях существует несколько видов (категорий) занятий. В рассматриваемом примере ограничимся для определенности лекциями, практическими занятиями и консультациями (могут быть и другие, но в этом примере остановимся именно на таком наборе).

Эти занятия, как правило, по-разному тарифицируются в плане оплаты (под тарификацией подразумевается часовая стоимость занятия для преподавателя). В этом случае преподаватель, проведя некоторое количество часов учебной нагрузки определенного вида, должен получить сумму, являющуюся результатом умножения количества часов на определенный тариф. Таким образом, в информационной базе следует учитывать различные виды занятий, проведенных сотрудниками.

Разрабатываемые в данном разделе объекты конфигурации, как и прежде, будут предназначаться для ввода исходных данных, а также для получения разнообразной итоговой информации по работе учебного центра. Программирование форм и эле-

ментов управления, расположенных на формах, позволит обеспечить необходимый сервис для пользователя.

Для начала создадим набор необходимых объектов конфигурации, которые затем будут использоваться в примерах программирования форм.

Информация о тарифах

Начнем создание новой конфигурации, которая будет включать в себя необходимые объекты и программные процедуры, что в итоге обеспечит автоматизацию учета в учебном центре.

Конфигурацию разделим на две подсистемы, одну из которых назовем **ВнесениеДанных**, а другую — **Отчеты** (рис. 6.1). Для создания подсистем в дереве объектов конфигурации обратимся к разделу **Подсистемы**, входящему в ветку **Общие**, и воспользуемся контекстным меню.

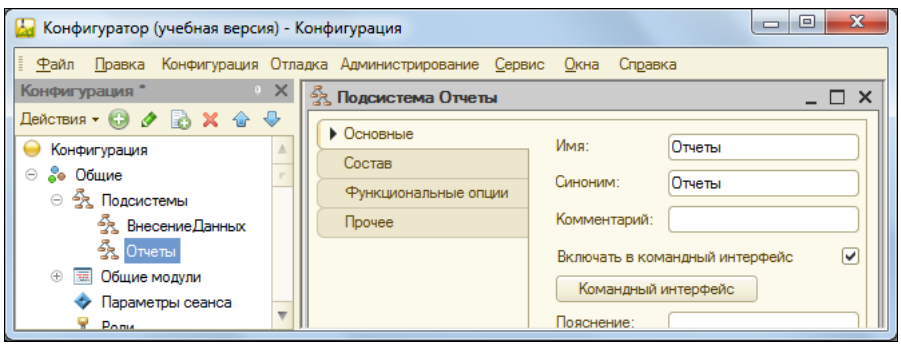


Рис. 6.1. Создание двух подсистем в конфигурации

Нам понадобится описать сведения о тарифах (их названиях и стоимости). Для этого создадим два объекта конфигурации — перечисление **ВидыЗанятий** и регистр сведений **Тарифы**.

Начнем с перечисления **ВидыЗанятий**. Значения, составляющие перечисление **ВидыЗанятий**, определяются на вкладке **Данные** окна редактирования объекта конфигурации (рис. 6.2).

Для задания тарифов занятий (очевидно, что различные виды занятий по-разному тарифицируются) создадим периодический регистр сведений **Тарифы**. Это название уже встречалось в одной из предыдущих глав, но поскольку новая конфигурация независимая, то путаницы здесь возникать не должно. Ранее подразумевалась тарификация оплаты за обучение, а сейчас рассматривается тарификация оплаты преподавателей.

На практике тарифы с течением времени меняются. В нашем случае будем считать, что они изменяются не чаще одного раза в месяц. Чтобы отразить это, необходимо на вкладке **Основные** в окне редактирования объекта конфигурации **Тарифы** (рис. 6.3) выбрать в списке **Периодичность** значение **В пределах месяца**. Здесь же

следует для режима записи в регистр сведений установить вариант **Независимый**. Такая установка позволит нам в режиме **1С:Предприятие** самостоятельно вносить в данный регистр сведений значения тарифов за обучение.

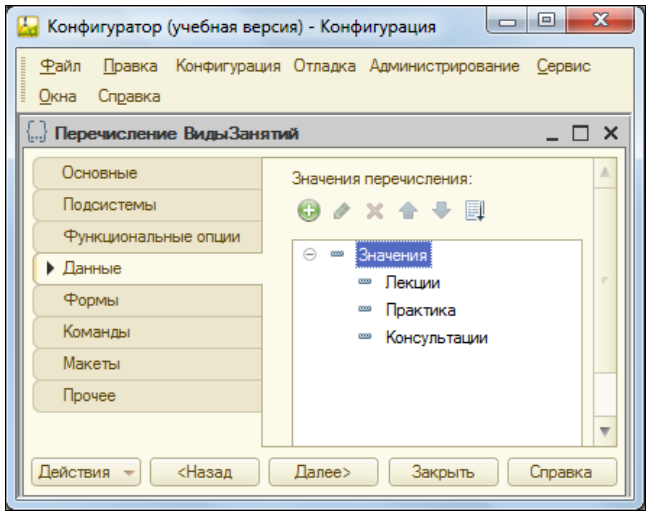


Рис. 6.2. Значения перечисления **ВидыЗанятий**

После этого следует перейти на вкладку **Данные** и определить измерения и ресурсы регистра сведений **Тарифы** (рис. 6.4). Одно измерение будет называться **ВидЗанятия** (тип данных — **ПеречислениеСсылка.ВидыЗанятий**). Другим измерением регистра является **Период**, что связано со сделанной нами установкой для поля **Периодичность** (в этом случае данное измерение создается платформой автоматически). В качестве ресурса регистра сведений примем — **ВеличинаТарифа** (тип данных — **Число**).

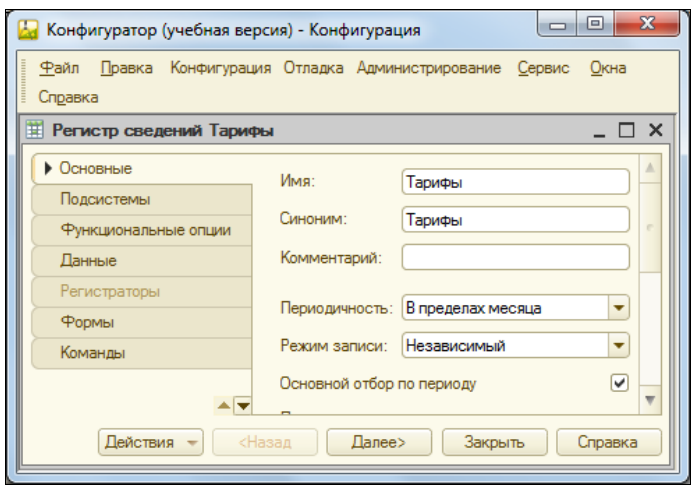


Рис. 6.3. Вкладка **Основные** окна редактирования регистра сведений **Тарифы**

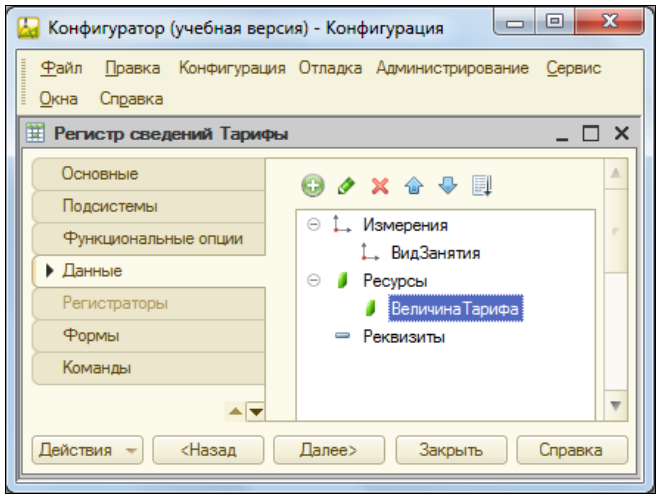


Рис. 6.4. Вкладка **Данные** окна редактирования регистра сведений **Тарифы**

Отнесем созданный регистр сведений **Тарифы** к подсистеме **ВнесениеДанных**, после чего перейдем из конфигуратора в режим **1С:Предприятие**. Здесь с помощью автоматически созданной экранной формы следует внести в регистр **Тарифы** информацию о нескольких тарифах в привязке к месяцу (рис. 6.5).

Примечание

Возможна ситуация, когда требуется организовать изменения в тарифах чаще, чем ежемесячно. В этом случае следует просто изменить значение периодичности в окне редактирования регистра сведений.

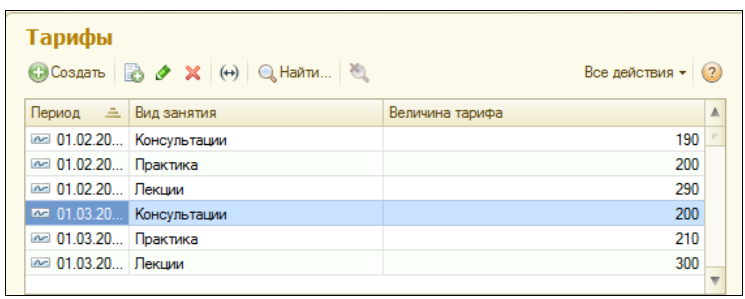


Рис. 6.5. Внесение данных в регистр сведений **Тарифы** в режиме **1С:Предприятие**

Этап подготовительной работы по созданию необходимых объектов конфигурации еще не завершен, и перед разработкой программных процедур нам потребуется сформировать несколько вспомогательных справочников.

Вспомогательные справочники

Во-первых, необходимо создать справочник "Преподаватели", предназначенный для перечисления фамилий преподавателей нашего учебного центра. Этот объект

конфигурации не будет содержать дополнительных реквизитов, и каких-то особенностей при работе с формой справочника в режиме **1С:Предприятие** нам не потребуется. В связи с этим автоматически генерируемых форм будет вполне достаточно. Для дальнейшей работы внесем в справочник несколько преподавателей (рис. 6.6).

Другой справочник, который назовем "Группы", отражает названия учебных групп (рис. 6.7). В нем, кроме кода и наименования, добавлен реквизит **Количество** — для отражения количества учащихся в каждой группе. Как и при заполнении предыдущего справочника, используем форму списка, генерируемую системой по

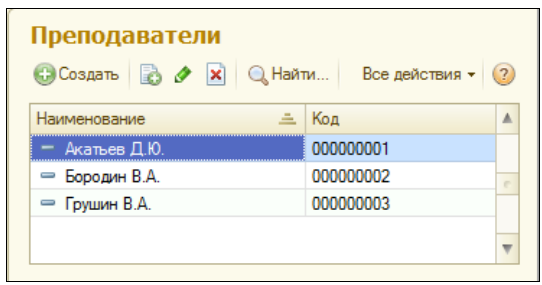


Рис. 6.6. Заполнение справочника "Преподаватели" в режиме **1С:Предприятие**

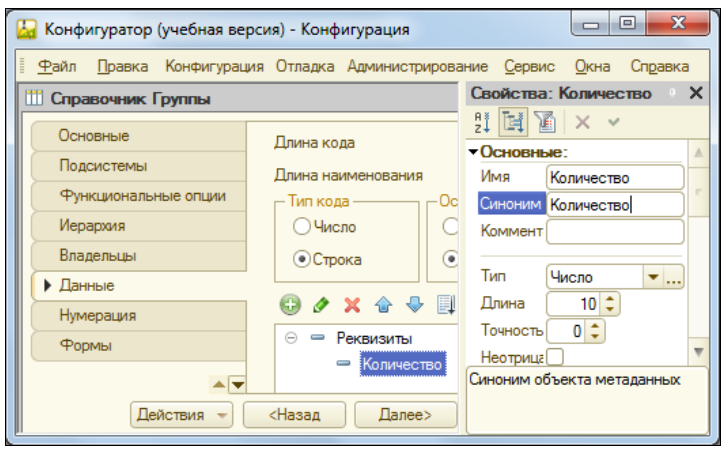


Рис. 6.7. Вкладка **Данные** окна редактирования справочника "Группы"

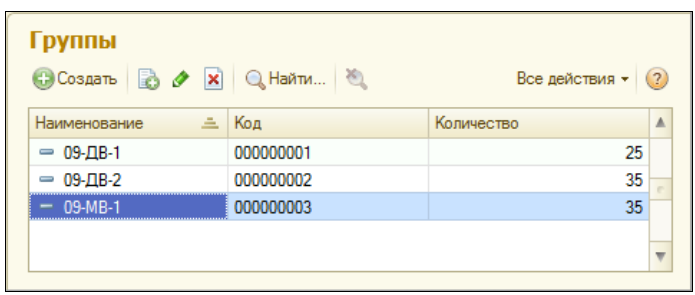


Рис. 6.8. Форма списка справочника "Группы" в режиме **1С:Предприятие**

умолчанию. На рис. 6.8 приведен один из вариантов заполнения справочника "Группы" с названиями групп.

Документ *Учет*

После создания и заполнения информацией справочников перейдем к разработке документа для фиксирования начисленных преподавателям сумм за проведенные занятия. Для этого в режиме конфигуратора создадим новый объект конфигурации — документ "Учет". В области шапки разместим только один дополнительный реквизит **ДатаРасчета**, который будет использоваться для внесения информации о дате, на которую производится расчет начисляемых сумм. Сами суммы будут размещаться в табличной части документа. Это связано с тем, что дата создания документа и дата расчета могут отличаться, а тарифы в нашей ситуации привязаны к дате (в качестве которой примем значение в поле **ДатаРасчета**).

Использование реквизита **ДатаРасчета** позволит нам в дальнейшем (в программной процедуре) извлекать из регистра сведений **Тарифы** информацию о величине тарифа на тот месяц, который указан в этом реквизите.

Для отражения основной информации об оплате в окне редактирования объекта конфигурации (рис. 6.9) необходимо создать табличную часть (с именем **Список**) и разместить в ней несколько реквизитов:

- **Преподаватель** (информация о преподавателе, тип данных — **СправочникСсылка.Преподаватели**);

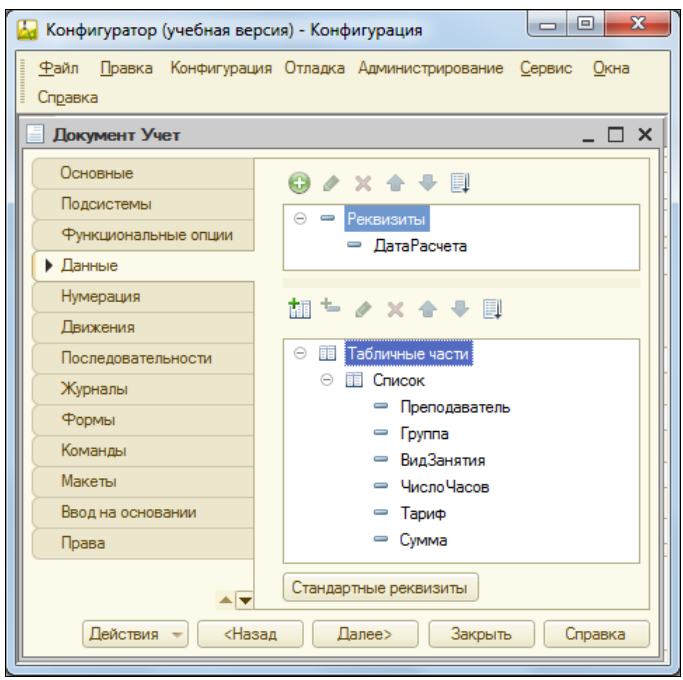


Рис. 6.9. Вкладка **Данные** окна редактирования документа "Учет"

- **Группа** (информация о группе, тип данных — **СправочникСсылка.Группы**);
- **ВидЗанятия** (тип данных — **ПеречислениеСсылка.ВидыЗанятий**);
- **ЧислоЧасов** (тип данных — **Число**);
- **Тариф** (тип данных — **Число**);
- **Сумма** (тип данных — **Число**).

В связи с тем, что в дальнейшем нам придется реагировать на интерактивные действия пользователя в программе, форма документа, генерируемая по умолчанию, нам не подойдет. Поэтому несложными (уже знакомыми) техническими действиями создадим пользовательскую форму для данного документа (рис. 6.10).

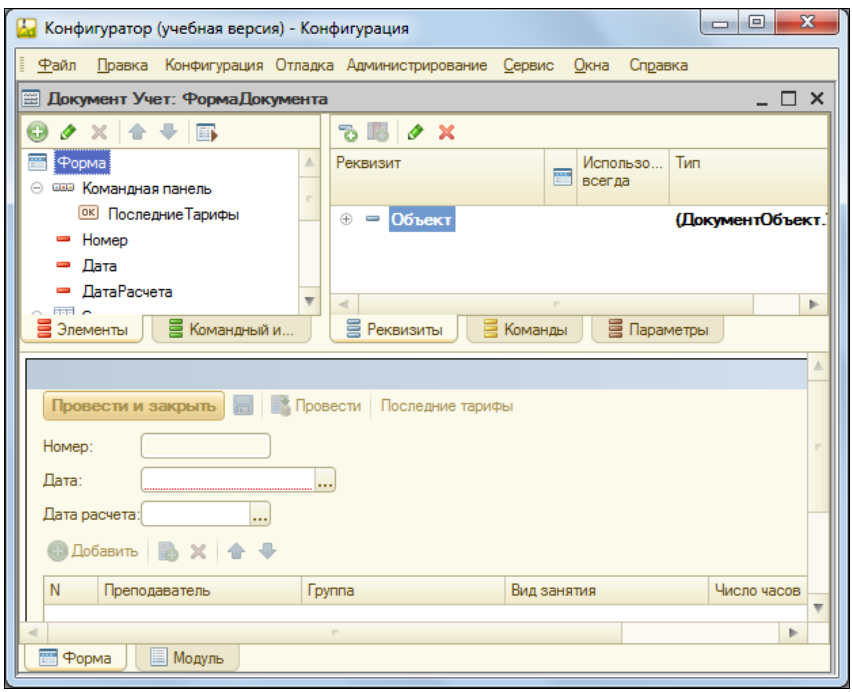


Рис. 6.10. Форма документа "Учет"

Процедуры обработки событий на форме документа

В форму документа необходимо заложить удобный интерфейс для работы пользователя. Поэтому в этом разделе мы рассмотрим программирование событий, связанных с расположенными на форме элементами управления. Но сначала поясним, как будет выглядеть в целом процесс заполнения данного документа пользователем в режиме **1С:Предприятие**.

Сотрудник учебного центра при создании нового документа "Учет" сначала заполняет расположенное в области шапки поле **ДатаРасчета**, которое определяет, на

какую дату мы собираемся производить расчет начислений преподавателям. После этого вносится информация о проведенных занятиях в табличную часть документа "Учет".

В процессе заполнения очередной строки прежде всего необходимо заполнить поле **Преподаватель** (для этого автоматически открывается список преподавателей из соответствующего справочника). Затем заполняется поле **Группа**, для чего используется другой справочник "Группы". Внесение данных в следующее поле **ВидЗанятия** также производится с помощью выбора значения из ранее созданного перечисления **ВидыЗанятий**.

После заполнения этих трех полей требуется реализовать определенную автоматизацию, а именно: при выборе вида занятия в поле **Тариф** должно автоматически подставляться значение тарифа на дату, ближайшую к дате расчета (точнее, на дату не позже даты расчета).

Для этого мы должны оформить соответствующим образом обработку события **ПриИзменении** для поля **СписокВидЗанятия**. Технически для этого необходимо открыть палитру свойств данного поля и обратиться к событию **ПриИзменении** (рис. 6.11). Здесь с помощью щелчка мышью по изображению лупы происходит переход в окно процедуры обработки данного события. В листинге 6.1 приведена клиентская процедура `СписокВидЗанятияПриИзменении`, которую здесь требуется поместить.

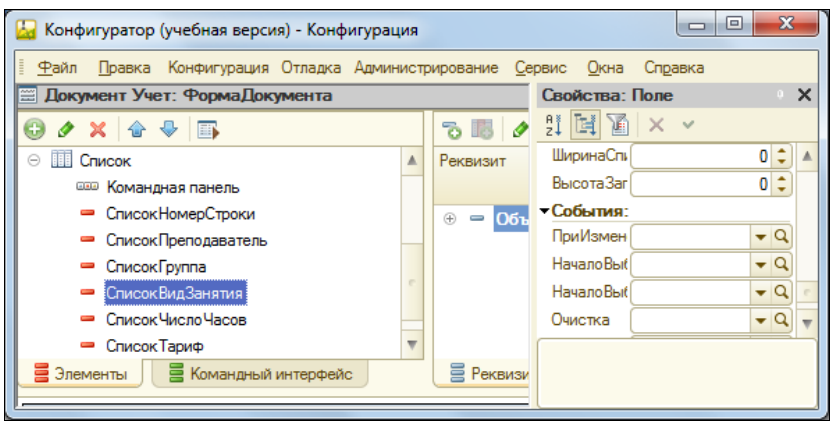


Рис. 6.11. Палитра свойств поля **СписокВидЗанятия**

Листинг 6.1. Процедура, выполняемая при изменении в поле СписокВидЗанятия

```
&НаКлиенте
Процедура СписокВидЗанятияПриИзменении(Элемент)
 СтрТабЧасти=Элементы.Список.ТекущиеДанные;
 СтрТабЧасти.Тариф=ПолучитьТариф(Объект.ДатаРасчета,
   СтрТабЧасти.ВидЗанятия);
КонецПроцедуры
```

Необходимые вычисления тарифа в процедуре СписокВидЗанятияПриИзменении производятся с помощью серверной функции ПолучитьТариф (листинг 6.2). Заметим, что данная функция не требует передачи контекста формы на сервер.

Листинг 6.2. Серверная функция ПолучитьТариф

```
&НаСервереБезКонтекста
Функция ПолучитьТариф(АктуальнаяДата,Занятие)
 Отбор=Новый Структура("ВидЗанятия",Занятие);
 ЗначениеТарифа=РегистрыСведений.Тарифы.ПолучитьПоследнее(АктуальнаяДата,
    Отбор);
 Возврат ЗначениеТарифа.ВеличинаТарифа;
КонецФункции
```

Для отбора ближайшего к дате расчета значения тарифа используется метод ПолучитьПоследнее, который требует некоторого комментария. Вообще для получения информации о значении ресурса регистра сведений можно использовать метод Получить. В этом случае мы можем выбрать конкретное значение ресурса, соответствующее набору указанных измерений (в данной ситуации у нас измерение одно) и значению периода. Например, это можно выполнить с помощью следующего программного кода:

```
ЗначениеТарифа = РегистрыСведений.Тарифы.Получить(АктуальнаяДата, Отбор);
```

Использование подобной конструкции вполне допустимо, но при условии, что для каждого месяца в регистр сведений занесено значение тарифа. Однако в реальной ситуации в течение продолжительного времени тарифы могут не меняться, и в регистр сведений их повторно не вносят. В связи с этим, если на указанный месяц (**ДатаРасчета**) информации в регистре сведений нет, то при использовании метода Получить мы выберем пустое значение. Метод ПолучитьПоследнее позволяет извлечь из регистра последний по времени имеющийся тариф (не позднее указанной в качестве параметра даты) для выбранного вида занятий. Это как раз нам и требуется для рассматриваемой задачи.

Таким образом, в процедуре, представленной в листинге 6.2, вносится правильное значение тарифа на дату расчета, и от пользователя требуется ввести только количество отработанных часов. После этого на форме документа (в табличной части) должна автоматически рассчитываться сумма, которую следует начислить преподавателю. Для этого требуется соответствующим образом оформить процедуру обработки события **ПриИзменении** для поля **СписокЧислоЧасов** (рис. 6.12). В листинге 6.3 представлена эта несложная клиентская процедура.

Листинг 6.3. Процедура для расчета в поле Сумма при внесении количества часов

```
&НаКлиенте
Процедура СписокЧислоЧасовПриИзменении(Элемент)
  ТабличнаяЧасть = Элементы.Список.ТекущиеДанные;
  ТабличнаяЧасть.Сумма = ТабличнаяЧасть.Тариф*ТабличнаяЧасть.ЧислоЧасов;
КонецПроцедуры
```

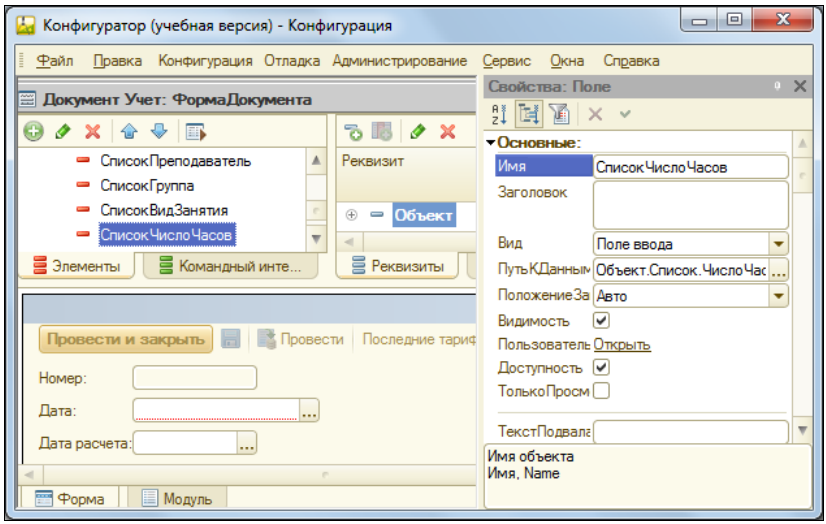


Рис. 6.12. Палитра свойств поля **СписокЧислоЧасов**

Итак, первый этап программирования формы документа завершен. Теперь можно перейти в режим **1С:Предприятие** и заполнить несколько документов "Учет". Один из вариантов заполнения формы документа показан на рис. 6.13. В результате обеспечивается запланированный сервис — значения тарифов на дату расчета заполняются автоматически, а суммы рассчитываются после указания пользователем количества отработанных часов.

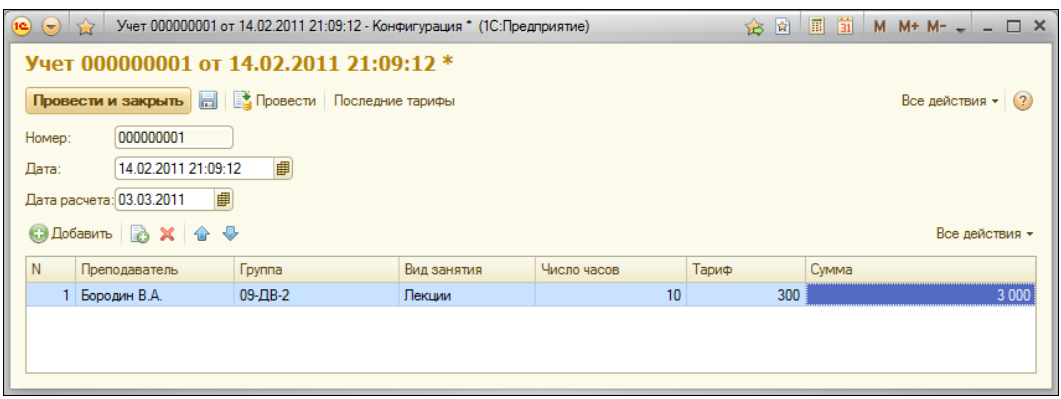


Рис. 6.13. Заполненная форма документа "Учет" в режиме **1С:Предприятие**

Однако функция установки тарифов (см. листинг 6.2) не позволяет реализовать функциональность интерфейса в случае, если при заполненной табличной части мы изменим дату расчета. В этом случае необходимо обеспечить автоматический пересчет информации в табличной части документа, а именно: должны быть внесены новые значения тарифов и пересчитана сумма по каждому преподавателю. Для реализации такого функционала следует оформить соответствующим образом процедуру (листинг 6.4), выполняемую при изменении в поле **ДатаРасчета** (рис. 6.14).

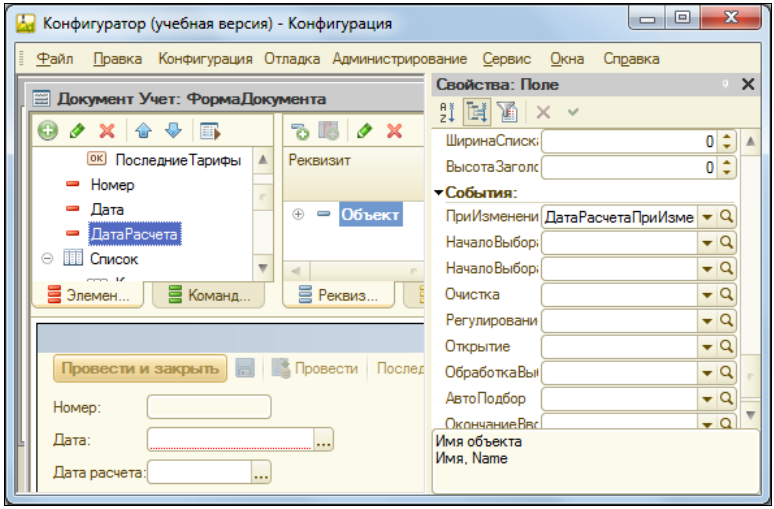


Рис. 6.14. Палитра свойств поля **ДатаРасчета**

При этом основные вычислительные действия будут, как ранее, выполняться на сервере (листинг 6.5).

Листинг 6.4. Процедура, выполняемая при изменении в поле ДатаРасчета

```
&НаКлиенте
Процедура ДатаРасчетаПриИзменении(Элемент)
   ПересчитатьНаСервере();
КонецПроцедуры
```

Листинг 6.5. Серверная процедура для перерасчета табличной части документа

```
&НаСервере
Процедура ПересчитатьНаСервере()
 Для Каждого ТекСтрокаСписок Из Объект.Список Цикл
   СтруктураОтбора = Новый Структура("ВидЗанятия",
         ТекСтрокаСписок.ВидЗанятия);
   Ресурс = РегистрыСведений.Тарифы.
           ПолучитьПоследнее(Объект.ДатаРасчета, СтруктураОтбора);
   ТекСтрокаСписок.Тариф = Ресурс.ВеличинаТарифа;
   ТекСтрокаСписок.Сумма =
         ТекСтрокаСписок.Тариф*ТекСтрокаСписок.ЧислоЧасов;
КонецЦикла;
КонецПроцедуры
```

Теперь, если перейти в режим **1С:Предприятие**, можно убедиться в том, что данные табличной части документа пересчитываются при изменении даты расчета (рис. 6.15).

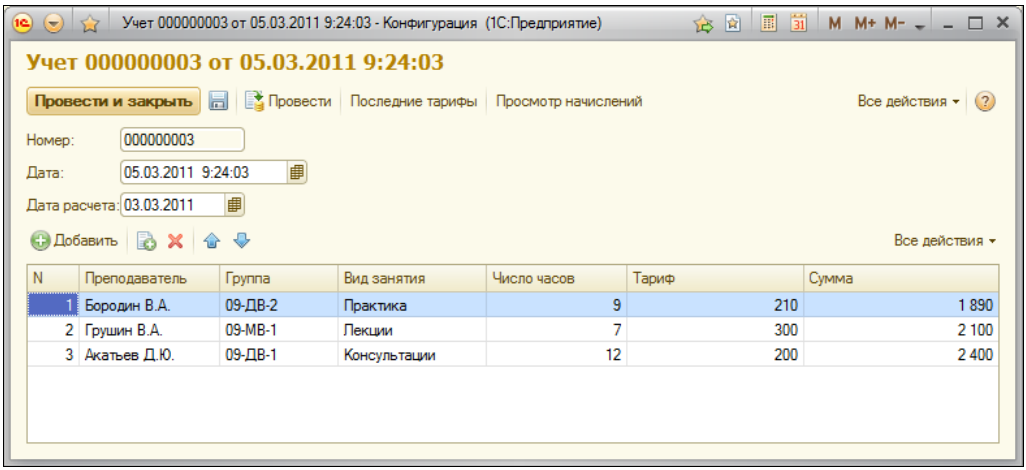


Рис. 6.15. Заполненная форма документа "Учет" в режиме работы **1С:Предприятие**

Просмотр актуальных тарифов

Документ уже достаточно функционален, но сервисные доработки формы пока не завершены. Наша цель в данном разделе — организовать просмотр информации по тарифам. Для этого на форме документа "Учет" разместим элемент управления — кнопку. Этот элемент в данном случае будет предназначен для того, чтобы пользователь прямо на форме документа мог посмотреть последние тарифы (на указанную дату расчета) по всем категориям занятий. Так, при нажатии данной кнопки в окно сообщений будет выводиться актуальная информация из базы данных.

Для начала в форме документа "Учет" создадим новую команду (рис. 6.16). Процедура обработки данной команды (листинг 6.6) содержит только серверный вызов другой процедуры `ОтобразитьПоследниеТарифы` (листинг 6.7).

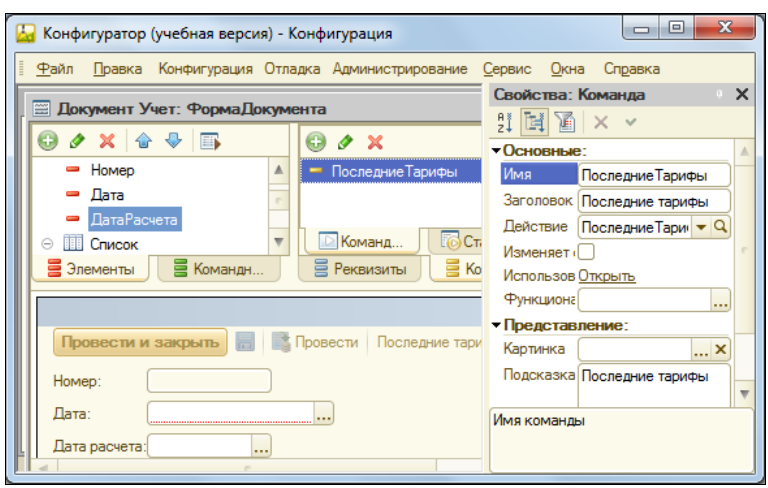


Рис. 6.16. Добавление команды в форму документа "Учет"

Листинг 6.6. Обработка команды ПоследниеТарифы

```
&НаКлиенте
Процедура ПоследниеТарифы(Команда)
ОтобразитьПоследниеТарифы()
КонецПроцедуры
```

Листинг 6.7. Серверная процедура ОтобразитьПоследниеТарифы

```
&НаСервере
Процедура ОтобразитьПоследниеТарифы()
  Запрос = Новый Запрос;
  Запрос.Текст = "ВЫБРАТЬ  Срез.ВидЗанятия КАК Вид,
  | Срез.ВеличинаТарифа   КАК Величина
  |  ИЗ  РегистрСведений.Тарифы.СрезПоследних(&АктуальнаяДата)
  |  КАК Срез";
  Запрос.УстановитьПараметр("АктуальнаяДата",
        Объект.ДатаРасчета);
  Результат = Запрос.Выполнить().Выбрать();
  Сообщение=Новый СообщениеПользователю;
  Пока Результат.Следующий() > 0 Цикл
      Сообщение.Текст=Строка(Результат.Вид) +
         "   " + Строка(Результат.Величина) + " руб. ";
      Сообщение.Сообщить();
   КонецЦикла;
КонецПроцедуры
```

Приведенная процедура содержит некоторые новые детали и поэтому требует небольшого комментария. В системе 1С:Предприятие 8.2 для регистров сведений существует метод `СрезПоследних`, который возвращает таблицу значений, заполненную информацией о последних актуальных значениях ресурсов регистра.

В связи с тем, что получение среза последней по дате информации встречается достаточно часто, разработчиками системы 1С:Предприятие была включена возможность использования виртуальной таблицы для получения такого среза.

В данном случае вместо запроса к основной таблице регистра сведений мы воспользовались запросом к виртуальной таблице:

```
ИЗ  РегистрСведений.Тарифы.СрезПоследних(&АктуальнаяДата).
```

Далее необходимо добавить кнопку в форму документа для выполнения команды **ПоследниеТарифы** (рис. 6.17). Это реализуется путем перетаскивания пиктограммы разработанной команды на командную панель.

Теперь можно перейти в режим **1С:Предприятие** и проверить разработанный функционал (рис. 6.18).

Примечание

Виртуальная таблица среза последних значений измерений не хранится в базе данных, а строится в момент обращения к ней.

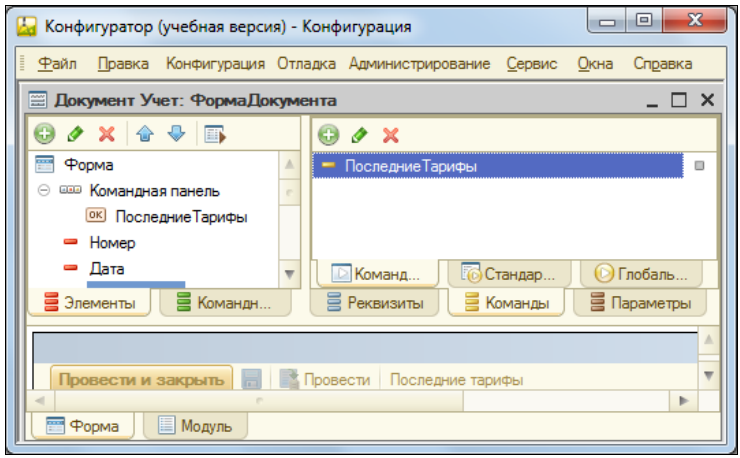


Рис. 6.17. Добавление кнопки в форму документа для выполнения команды

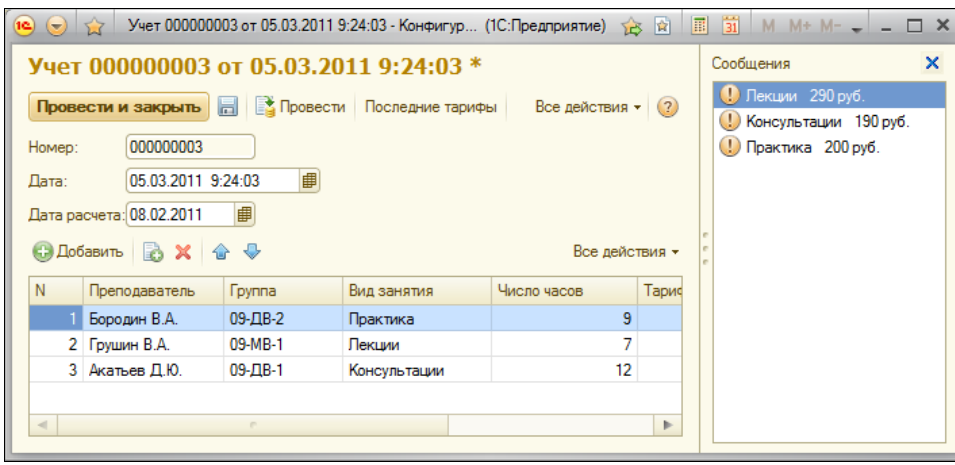


Рис. 6.18. Просмотр тарифов в форме документа "Учет"

Таким образом, документ "Учет" почти готов к использованию, и осталось лишь организовать автоматический учет денежных сумм по данным документам.

Для этого, во-первых, нужно создать регистр накопления, в котором будут регистрироваться внесенные суммы. А во-вторых, требуется обеспечить движения по данному регистру при проведении документа "Учет".

Оборотный регистр накопления

Как мы уже знаем, регистры накопления имеют две разновидности — регистры остатков и регистры оборотов. В предыдущих главах для задач учета количественной информации мы использовали и те и другие регистры. Для регистров остатков система 1С:Предприятие создает несколько виртуальных таблиц — таблицу остатков, таблицу оборотов и совместную таблицу остатков и оборотов.

Оборотный регистр отличается от регистра остатков тем, что для него система создает только одну виртуальную таблицу — таблицу оборотов.

Для начала на вкладке **Основные** в окне редактирования объекта конфигурации необходимо установить начальную информацию (рис. 6.19), определяющую регистр **Начисление**. Обратите внимание на то, что вид данного регистра — **Обороты**.

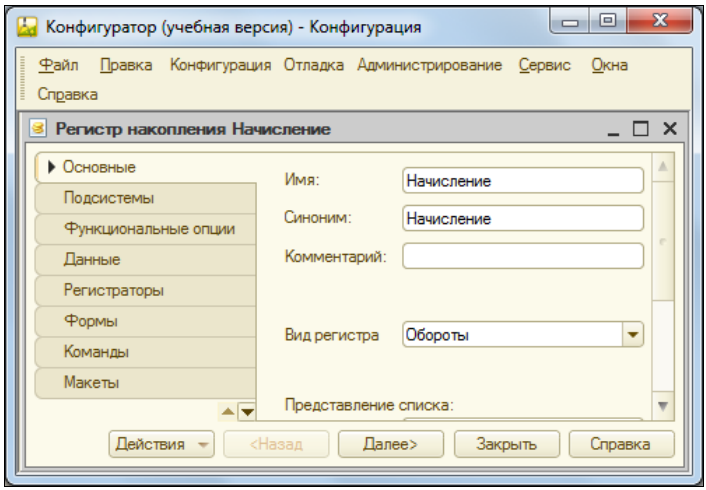


Рис. 6.19. Вкладка **Основные** окна редактирования регистра **Начисление**

Далее на вкладке **Данные** (рис. 6.20) следует ввести измерение **Преподаватель** (тип данных — **СправочникСсылка.Преподаватели**) и ресурс **Сумма** (тип данных — **Число**). Дальнейшая работа с регистром (для записи информации по начислению денежных средств) должна быть организована с помощью документа "Учет".

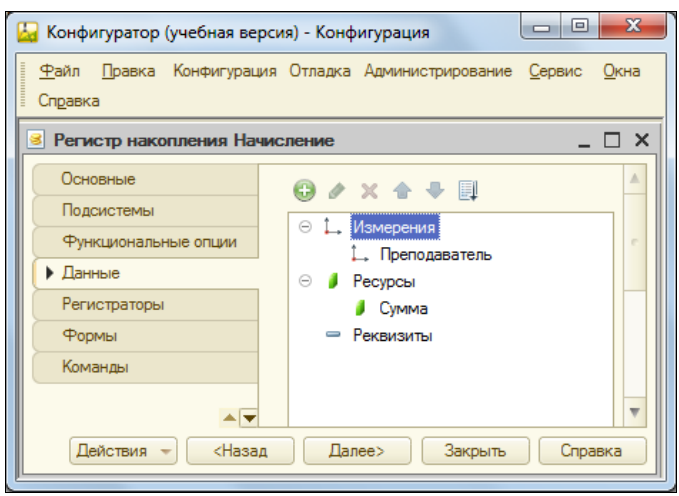


Рис. 6.20. Вкладка **Данные** окна редактирования регистра **Начисление**

Движения по регистру *Начисление*

В режиме **1С:Предприятие** работать с созданным регистром для учета начисленных сумм пока рано, поскольку механизм записи информации в регистр еще не обеспечен. Дело в том, что движения по регистру должны выполняться при проведении документа "Учет".

Поэтому в режиме **Конфигуратор** вернемся в окно редактирования документа "Учет" и перейдем на вкладку **Движения**, где укажем в качестве регистра накопления для движений только что созданный регистр (рис. 6.21). Для формирования движений воспользуемся конструктором (кнопка — **Конструктор движений**). В результате на экране увидим уже знакомое окно (рис. 6.22), где от нас потребуется подставить необходимые выражения для полей регистра (наши инструкции конструктору).

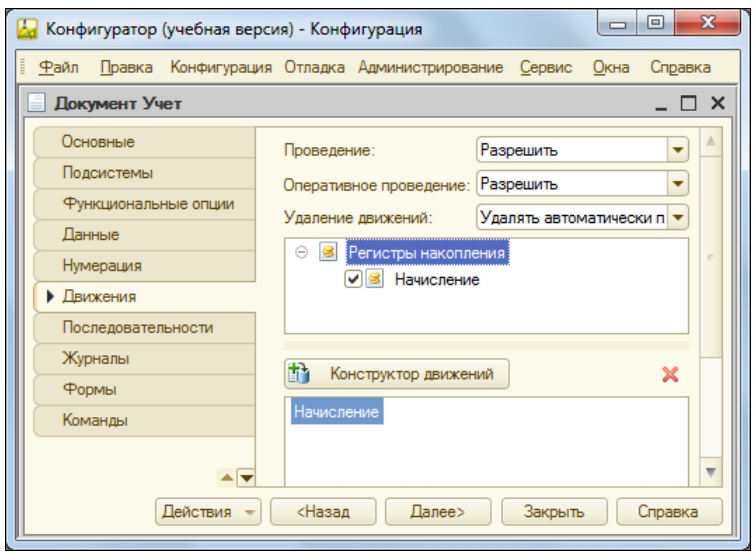


Рис. 6.21. Вкладка **Движения** окна редактирования документа "Учет"

После щелчка по кнопке **ОК** конструктор движений автоматически генерирует необходимый программный код (листинг 6.8).

Листинг 6.8. Процедура формирования движений по регистру Начисление

```
Процедура ОбработкаПроведения(Отказ, Режим)
//{{__КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
// Данный фрагмент построен конструктором.
  Движения.Начисление.Записывать = Истина;
  Для Каждого ТекСтрокаСписок Из Список Цикл
   // регистр Начисление
   Движение = Движения.Начисление.Добавить();
   Движение.Период = Дата;
```

```
    Движение.Преподаватель = ТекСтрокаСписок.Преподаватель;
    Движение.Сумма = ТекСтрокаСписок.Сумма;
  КонецЦикла;
//}}__КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
КонецПроцедуры
```

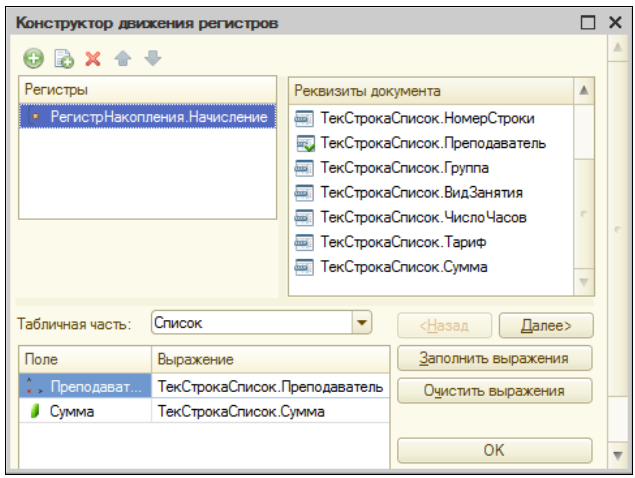


Рис. 6.22. Окно конструктора движений документа "Учет"

Конструктор движений обеспечил завершенность функциональности документа "Учет" и теперь при записи документа в информационную базу формируются движения по регистру. На этом работа в конфигураторе по созданию документа "Учет" завершена. Документ готов к использованию, и, перейдя в режим **1С:Предприятие**, можно сформировать (с проведением при записи) несколько документов типа **Учет**. В результате в регистре **Начисление** будут сформированы соответствующие записи (рис. 6.23). Каждая строка табличной части документа найдет свое отражение в регистре. Теперь можно составить отчет на основании информации, которая содержится в таблицах, формируемых системой для сохранения данных регистра **Начисление**.

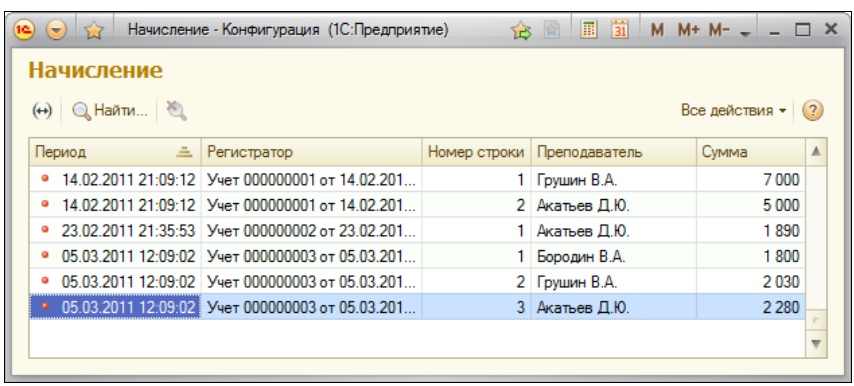


Рис. 6.23. Форма списка регистра **Начисление**

Просмотр информации по начислениям

После разработки электронной регистрации документов необходимо продумать для сотрудников учебного центра и возможность просмотра всех начислений. Реализуем этот функционал в документе "Учет". Здесь мы создадим кнопку **Просмотр начислений** (рис. 6.24), с помощью которой на экране будет открываться табличный документ со всеми начислениями по сотрудникам, указанным в данном документе "Учет".

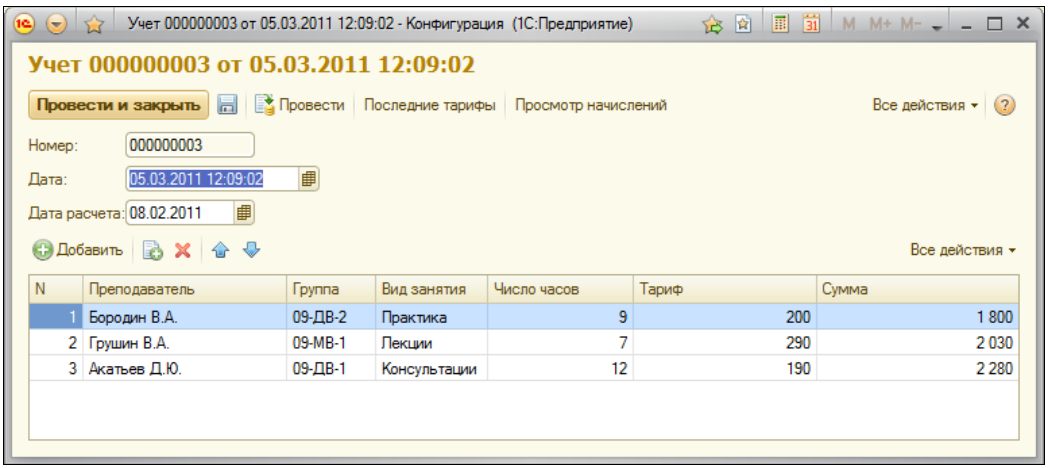


Рис. 6.24. Добавление кнопки **Просмотр начислений** в форму списка документа "Учет"

Для этого в конфигураторе откроем окно редактирования объекта конфигурации для документа "Учет". Здесь требуется перейти на вкладку **Макет** и создать макет для табличного документа (рис. 6.25). На рис. 6.26 представлена форма документа "Учет".

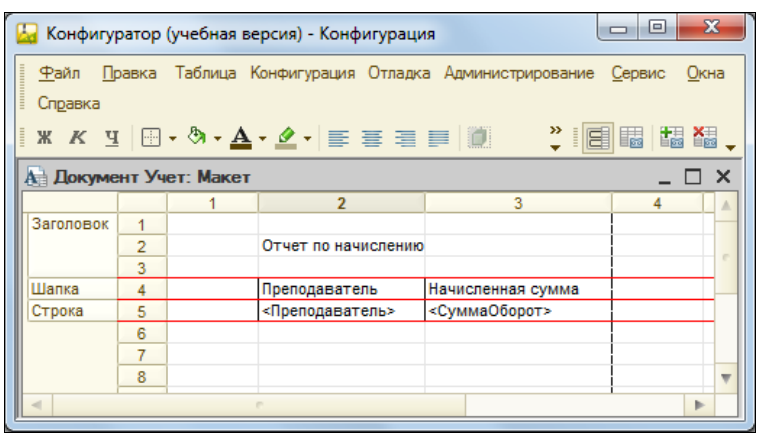


Рис. 6.25. Макет для табличного документа

На следующем этапе необходимо создать в форме новую команду ПросмотрНачислений. Программный код процедуры обработки данной команды представлен в листинге 6.9.

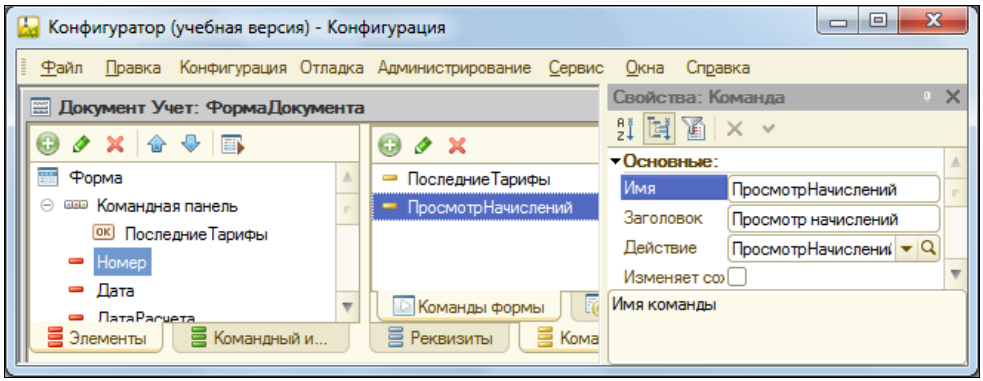


Рис. 6.26. Форма документа "Учет"

Листинг 6.9. Процедура обработки команды ПросмотрНачислений

```
&НаКлиенте
Процедура ПросмотрНачислений (Команда)
  ТабДок = Новый ТабличныйДокумент;
  ПечатьВДокумент(ТабДок);
  ТабДок.ОтображатьСетку = Ложь;
  ТабДок.Защита = Ложь;
  ТабДок.ТолькоПросмотр = Ложь;
  ТабДок.ОтображатьЗаголовки = Ложь;
  ТабДок.Показать();
КонецПроцедуры
```

Основные вычисления производятся в вызываемой серверной процедуре ПечатьВДокумент (листинг 6.10).

Листинг 6.10. Серверная процедура ПечатьВДокумент

```
&НаСервере
Процедура ПечатьВДокумент(ТабДок)
ТабДок.Очистить();
Макет = Документы.Учет.ПолучитьМакет("Макет");
Заг = Макет.ПолучитьОбласть("Заголовок");
ТабДок.Вывести(Заг);
Шапка = Макет.ПолучитьОбласть("Шапка");
ТабДок.Вывести(Шапка);
Для Каждого ТекСтрокаСписок Из Объект.Список Цикл
  Запрос = Новый Запрос;
```

```
  Запрос.Текст = "ВЫБРАТЬ Преподаватель,СуммаОборот
 | ИЗ РегистрНакопления.Начисление.Обороты
 | ГДЕ Преподаватель = &ВыборПреподавателя";
 Запрос.УстановитьПараметр("ВыборПреподавателя",
   ТекСтрокаСписок.Преподаватель);
 Результат = Запрос.Выполнить().Выбрать();
 Пока Результат.Следующий() > 0 Цикл
        Область = Макет.ПолучитьОбласть("Строка");
   Область.Параметры.Преподаватель = Результат.Преподаватель;
   Область.Параметры.СуммаОборот = Результат.СуммаОборот;
   ТабДок.Вывести(Область);
 КонецЦикла;
КонецЦикла;
КонецПроцедуры
```

Осталось поместить кнопку **ПросмотрНачислений** на форме, что производится с помощью перемещения названия команды на командную панель (рис. 6.27). В результате после перехода в режим **1С:Предприятие** и открытия формы одного из имеющихся документов "Учет" можно получить печатную форму отчета (рис. 6.28).

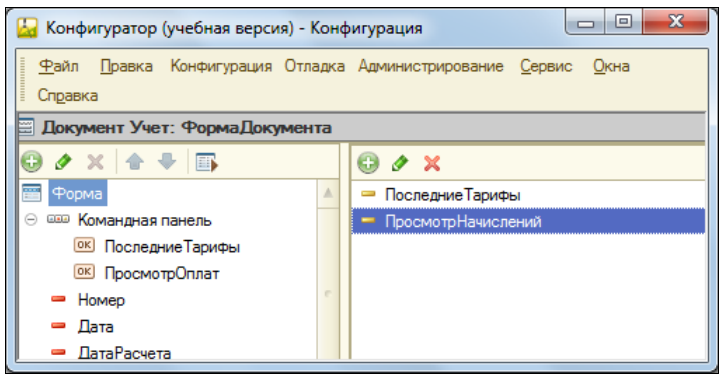


Рис. 6.27. Добавление кнопки в форму документа "Учет"

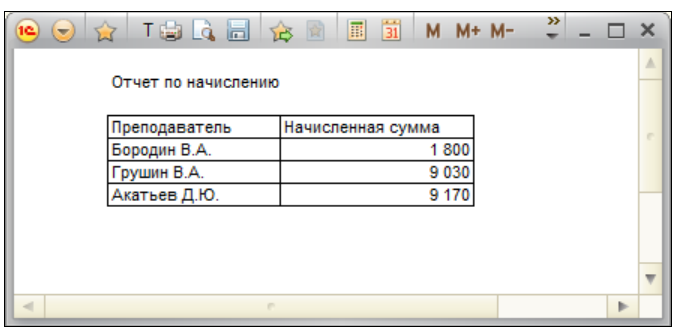


Рис. 6.28. Печатная форма отчета по начислению

Программное добавление элементов в справочник

Во всех предыдущих примерах добавление новых элементов в справочник производилось интерактивным способом. Так, в режиме **1С:Предприятие** мы открывали форму списка необходимого справочника, а затем с помощью стандартной кнопки на командной панели производили добавление элемента.

Здесь мы рассмотрим пример, в котором продемонстрируем возможность внесения элементов в справочник программным путем. Это действие может встретиться при разработке реальных конфигураций, и наша задача заключается в том, чтобы на практическом примере рассмотреть необходимую технологию.

Подготовка конфигурации

Рассматриваемая задача будет касаться учета продаж товаров. И на первом этапе создадим справочник "Товары", в котором формировать дополнительных реквизитов не потребуется. На рис. 6.29 показана заполненная форма списка справочника "Товары".

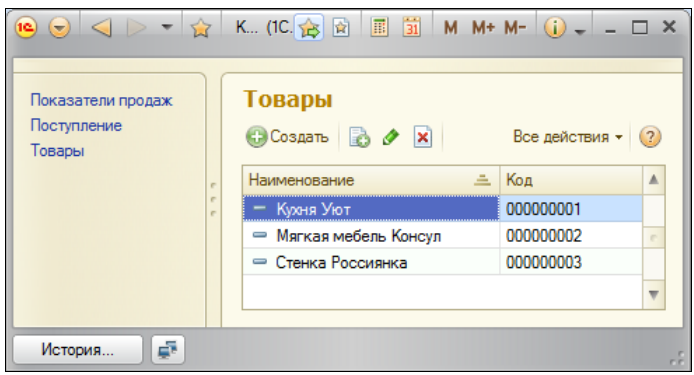


Рис. 6.29. Форма списка справочника "Товары"

Для фиксирования факта поступления товаров создадим документ "Поступление". На рис. 6.30 показана вкладка **Данные**, где необходимо поместить табличную часть **Перечень**, включающую два реквизита:

- **Товар** (тип данных — **СправочникСсылка.Товары**);
- **Количество** (тип данных— **Число**).

Для регистрации поступлений разработаем регистр накопления остатков **КоличествоПоступлений** (рис. 6.31). Сформируем измерение и ресурс регистра в соответствии с рис. 6.32.

Теперь следует перейти в режим **1С:Предприятие** и заполнить несколько документов по поступлению товаров (один из них представлен на рис. 6.33).

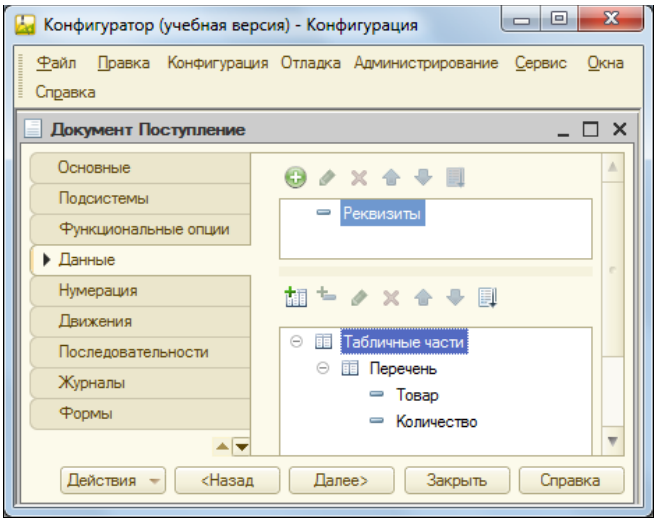


Рис. 6.30. Вкладка **Данные** окна редактирования документа "Поступление"

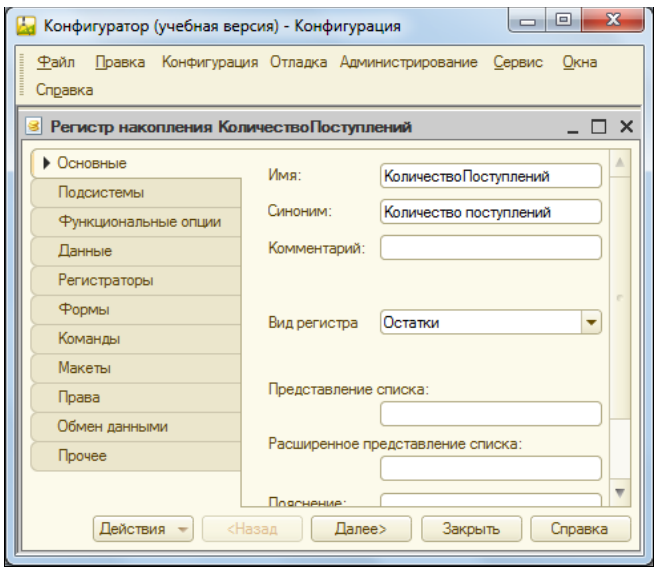


Рис. 6.31. Вкладка **Основные** окна редактирования **КоличествоПоступлений**

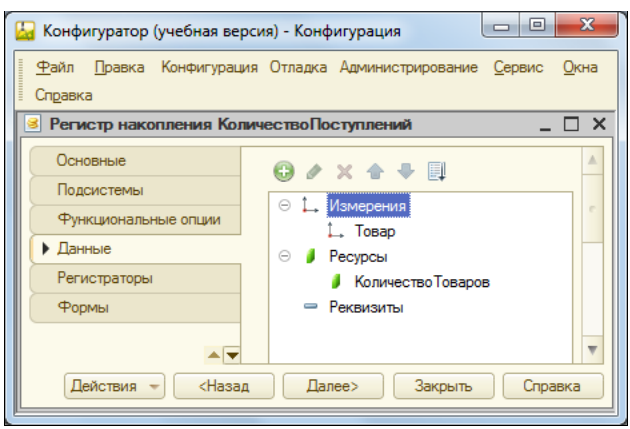


Рис. 6.32. Вкладка **Данные** окна редактирования регистра **КоличествоПоступлений**

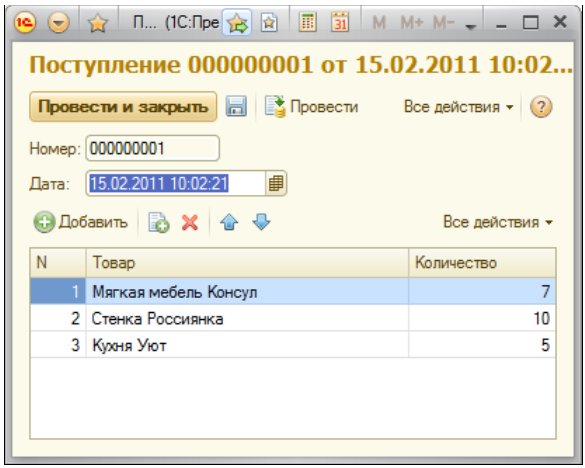


Рис. 6.33. Форма документа "Поступление" в режиме **1С:Предприятие**

В результате проведения ряда документов в регистре накопления сформируется набор записей (рис. 6.34).

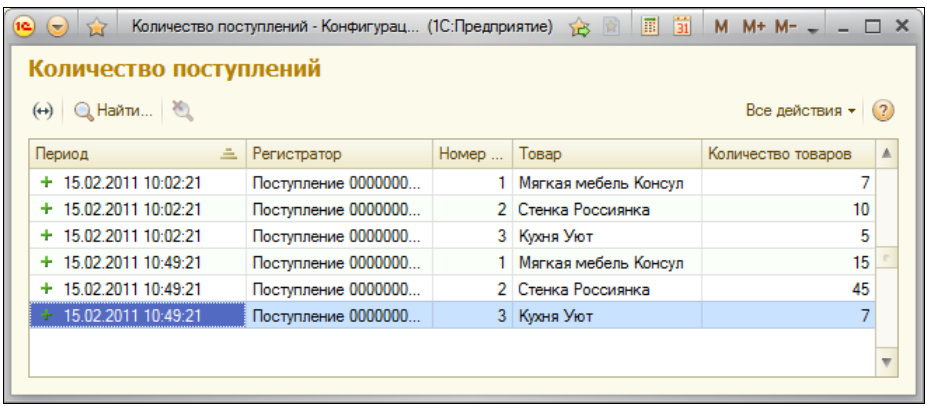


Рис. 6.34. Форма списка регистра **КоличествоПоступлений**

Разработка сервиса для просмотра

Будем считать, что наша задача заключается в формировании актуальной информации о количестве товаров на складе. При этом руководству нашей условной организации будет удобнее, если эту информацию можно просмотреть из справочника. В связи с этим добавим в конфигурацию новый объект конфигурации — справочник "ПоказателиПродаж" (рис. 6.35).

Ситуация такова, что менеджер должен обеспечить заполнение справочника "ПоказателиПродаж" данными. Внешне содержание данного справочника должно выглядеть так, как показано на рис. 6.36. После даты идет перечисление остатков по товарам на указанную дату. Для руководителя это удобно — открыл справочник и необходимая хронология остатков товаров сразу видна.

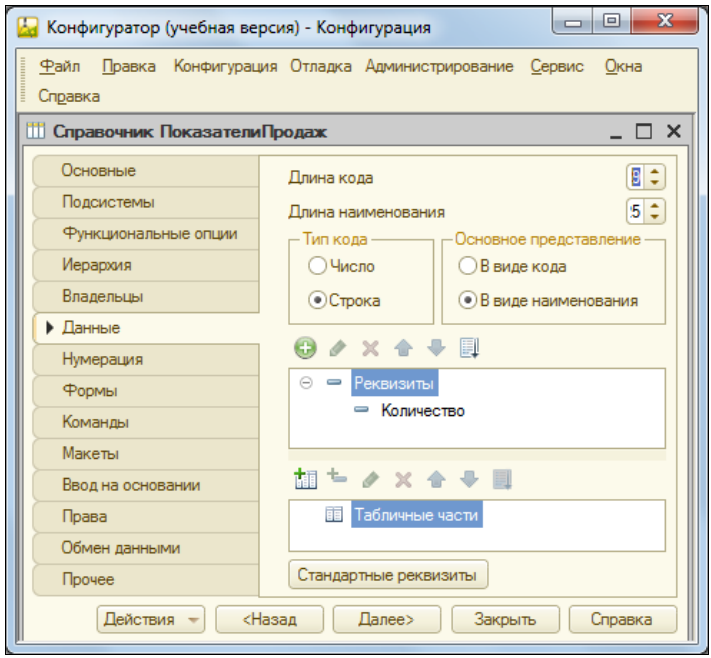


Рис. 6.35. Вкладка **Данные** окна редактирования справочника "ПоказателиПродаж"

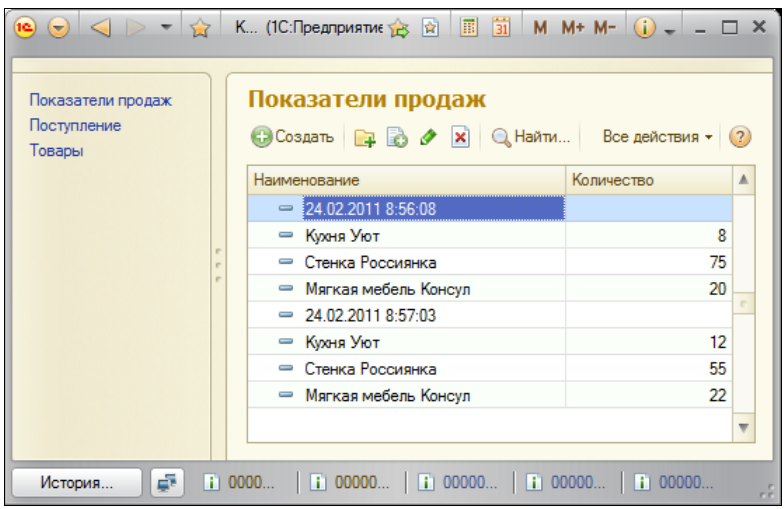


Рис. 6.36. Форма списка справочника "ПоказателиПродаж"

Реализуем теперь заполнение справочника "ПоказателиПродаж" менеджером. Занесение актуальной информации будет производиться в форме справочника "Товары" щелчком по кнопке, которую мы для этого создадим.

Таким образом, перейдем к работе с формой списка справочника "Товары". Здесь мы создадим команду **ПоказателиПродаж** (рис. 6.37). В листинге 6.11 приведена процедура обработки данной команды. Ее содержание заключается в вызове сер-

верной процедуры `СформироватьПоказатели` (листинг 6.12), в которой выполняется запрос к виртуальной таблице регистра **КоличествоПоступлений** и программно создаются элементы справочника "ПоказателиПродаж".

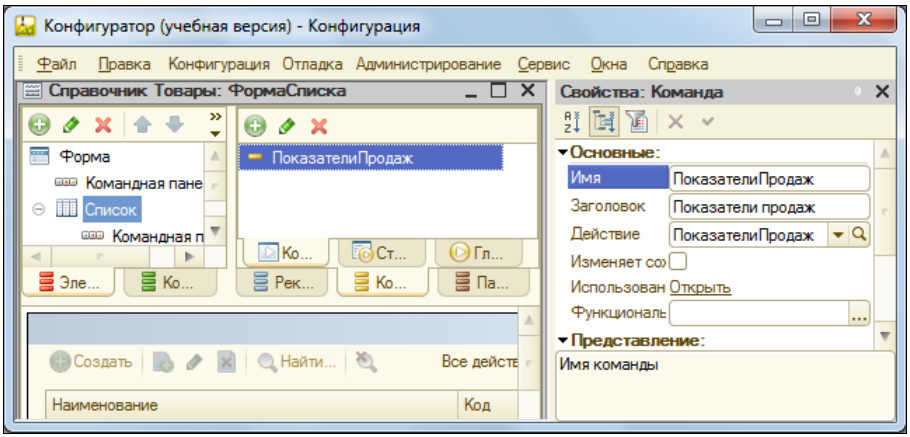


Рис. 6.37. Создание команды **ПоказателиПродаж**

Листинг 6.11. Процедура обработки команды ПоказателиПродаж

```
&НаКлиенте
Процедура ПоказателиПродаж(Команда)
СформироватьПоказатели();
КонецПроцедуры
```

Листинг 6.12. Серверная процедура СформироватьПоказатели

```
&НаСервереБезКонтекста
Процедура СформироватьПоказатели()
Запрос = Новый Запрос;
Запрос.Текст = "ВЫБРАТЬ ИнформацияОТоварах.Товар,
| ИнформацияОТоварах.КоличествоТоваровОстаток
| ИЗ РегистрНакопления.КоличествоПоступлений.Остатки
| КАК  ИнформацияОТоварах";
Результат = Запрос.Выполнить().Выбрать();
ОбъектАктТовара=Справочники.ПоказателиПродаж.СоздатьЭлемент();
ОбъектАктТовара.Наименование=Строка(ТекущаяДата());
ОбъектАктТовара.Записать();
Пока Результат.Следующий() > 0 Цикл
 ОбъектАктТовара=Справочники.ПоказателиПродаж.СоздатьЭлемент();
 ОбъектАктТовара.Наименование=Результат.Товар.Наименование;
 ОбъектАктТовара.Количество=Результат.КоличествоТоваровОстаток;
 ОбъектАктТовара.Записать();
КонецЦикла;
КонецПроцедуры
```

Теперь необходимо создать кнопку для выполнения команды **ПоказателиПродаж**, и разработка готова. Результат мы уже видели (см. рис. 6.36).

Учет бонусов и штрафов при организации оплаты

В этом разделе рассмотрим задачу учета оплаты учащимися за предоставленные образовательные услуги. Тематика будет похожа на ряд предыдущих примеров, однако исходная ситуация выглядит несколько иной. В связи с этим нам понадобятся новые информационные модели и объекты конфигурации. В силу многообразия различных схем оплаты, рассматриваемая в данном разделе информационная модель вполне может встретиться и в реальной ситуации.

Итак, будем считать, что слушателями наших курсов являются как индивидуальные клиенты, так и группы учащихся по договорам, заключенным с конкретными организациями. При зачислении слушателя (как индивидуального, так и от организации) с ним заключается договор на конкретную сумму за весь учебный цикл. Эта сумма может оплачиваться поэтапно, но при этом в процессе оплаты предусматриваются штрафные санкции (пени) либо, наоборот, (призовые) бонусы.

Поясним на примере, что представляет собой данная схема оплаты. Допустим, в договоре определены сроки поэтапного внесения денежных сумм за обучение. Например, 30 % от общей суммы договора учащийся должен внести в первом квартале года, затем следующие 20 % во втором и т. д. Однако при досрочной оплате определенной суммы она засчитывается с увеличением, скажем, на 10 %.

Так, внеся заранее 1000 рублей, учащийся получает бонус 10 % от данной суммы, и ему засчитывается 1100 рублей в качестве оплаты за будущий период. Наоборот, при несвоевременной оплате для учащегося вводятся штрафные санкции (пени). Например, если слушатель принесет в учебный центр квитанцию по оплате в 1000 рублей с запозданием, то ему могут быть назначены штрафные санкции в виде 10 % от оплаченной суммы. В результате ему в соответствии с договором зачитывается только 900 рублей. При этом разнообразные бонусы и штрафные санкции могут устанавливаться как индивидуальным слушателям, так и для организаций в целом. Также бонусы и штрафы могут устанавливаться учащимся и из-за других факторов (хорошая учеба, скидки для организаций и т. д.).

Представленная модель преследует цель продемонстрировать разнообразные технические возможности, которые могут быть востребованы и в другой практической ситуации.

В новой конфигурации мы создадим регистр накопления, который предназначен для учета внесенных слушателями курсов сумм. Этот регистр будет иметь одно измерение (для фамилии учащегося) и два ресурса (фактически внесенная сумма и эта же сумма, но с учетом бонуса). Это позволит нам извлекать из таблиц как реально заплаченную сумму, так и зачтенную сумму с учетом штрафных санкций и премий.

При учете поступления денежных средств всегда возможен и их возврат. Поэтому в рассматриваемом регистре накопления будут учитываться не только внесенные учащимися суммы, но и приниматься во внимание их возврат со стороны нашего учебного центра (например, в случае переплаты или отказа от обучения).

Для механизма внесения денежных средств и их возврата в разрабатываемой конфигурации нам потребуются соответствующие документы, при проведении которых в таблицы, созданные на основании регистра накопления, непосредственно самой информационной системой будет заноситься необходимая информация.

Важная составляющая информационной системы — это создание отчетов, необходимых руководству организации. Для получения итоговой информации по оплате мы сформируем несколько отчетов, а также попробуем обеспечить удобные сервисные возможности пользователю при работе с нашей информационной базой.

Таким образом, можно сказать, что основной план на разработку конфигурации данного раздела сформулирован, а в качестве первого практического шага в следующих разделах мы создадим несколько вспомогательных справочников. Далее мы разработаем документы для учета поступления и возврата денежных средств, а также обеспечим формирование ряда отчетов.

Справочник *Организации*

Как уже говорилось, для выполнения практических примеров нам потребуется новая конфигурация, поэтому начнем работу с создания пустой информационной базы. Первым объектом конфигурации будет справочник "Организации", в котором кроме обязательных реквизитов создадим еще один — **Бонус** (рис. 6.38). Этот дополнительный реквизит предназначен для хранения значения бонуса (в процентах) для конкретной организации. Укажем тип данных — **Число**, а точность выберем единицу, что означает десятую долю процента. Этот же параметр может использоваться и для наложения штрафных санкций — просто в этом случае его значение будет отрицательным.

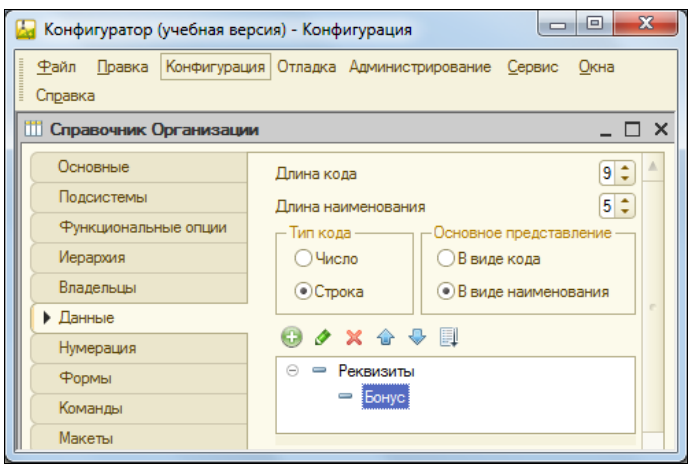


Рис. 6.38. Вкладка **Данные** окна редактирования справочника "Организации"

Забегая вперед, отметим, что после создания необходимых объектов конфигурации мы за счет программных процедур обеспечим сервисные возможности пользователю. Так, ввод информации по оплате в систему будет реализован с помощью соответствующего документа. В электронной форме этого документа после выбора названия организации значение реквизита **Бонус** будет автоматически заполняться данными на основании информации из справочника "Организации".

Итак, мы сформировали структуру первого справочника. Теперь перейдем в режим **1С:Предприятие**. При работе со справочником "Организации" будет вполне достаточно формы списка, которая генерируется системой по умолчанию. В качестве примера на рис. 6.39 показано заполнение данного справочника названиями нескольких организаций.

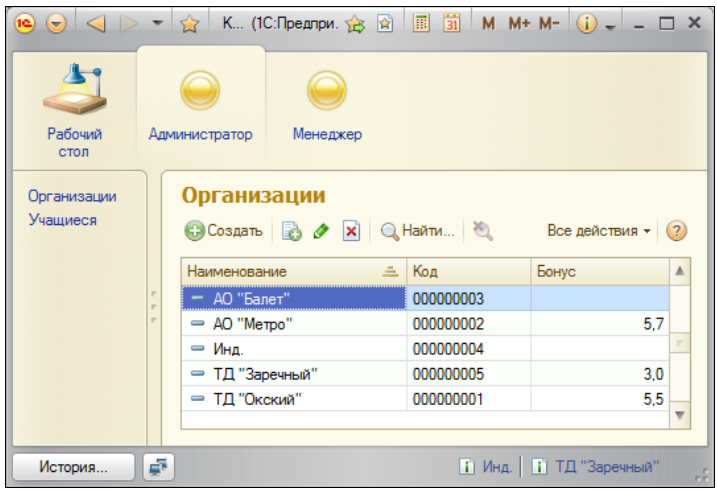


Рис. 6.39. Справочник "Организации" в режиме **1С:Предприятие**

Созданный вспомогательный справочник "Организации" нам потребуется при разработке следующего справочника "Учащиеся", в котором сформируем список учащихся нашего учебного центра.

Справочник *Учащиеся*

После внесения имени объекта на вкладке **Основные** окна редактирования объекта конфигурации перейдем к другой, более информативной вкладке — **Данные**, где необходимо разместить три дополнительных реквизита справочника (рис. 6.40). Табличная часть для элементов справочника здесь не потребуется.

Первым реквизитом будет **Организация** (тип данных — **СправочникСсылка.Организации**), а вторым — **ИндБонус** (тип данных — **Число**, с точностью до одного разряда после запятой). Второй реквизит предназначается для назначения индивидуального бонуса либо для штрафных санкций конкретному слушателю курсов. По смыслу задачи можно ввести категорию индивидуальных слушателей, включающую только лиц не входящих в группы, направляемые на обучение от

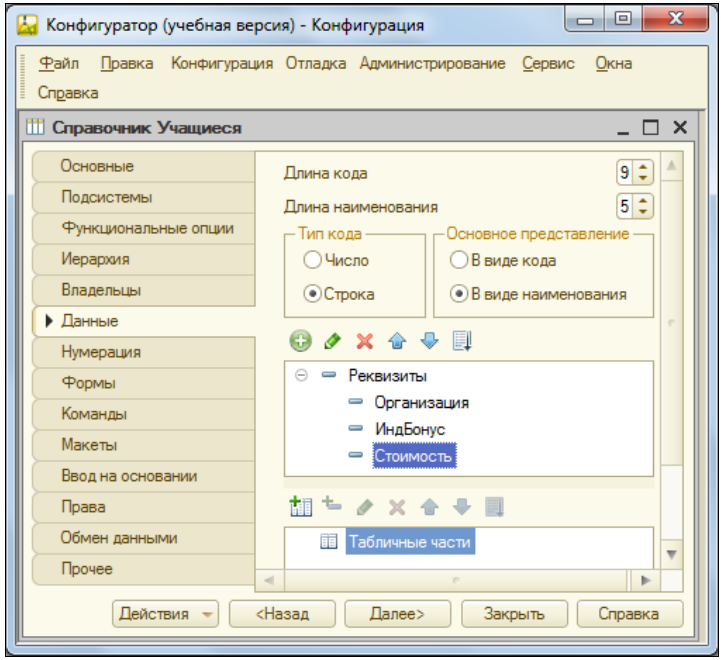


Рис. 6.40. Окно редактирования справочника "Учащиеся"

организаций. Третий реквизит **Стоимость** (тип данных — **Число**) будет хранить информацию о стоимости обучения для конкретного учащегося.

Сотрудник нашей организации при зачислении учащегося вместе с его фамилией заносит в справочник "Учащиеся" величину стоимости обучения. В дальнейшем в зависимости от регулярности оплат или других факторов можно внести значение индивидуального бонуса. А, скажем, в случае продолжительного отсутствия оплат можно внести отрицательный бонус, например минус три. Это соответствует включению пени (три процента) на оставшуюся неоплаченную сумму.

Рассмотренная информационная модель позволяет учитывать, кроме бонусов, относящихся к организации в целом, еще и индивидуальные бонусы для отдельных слушателей (но входящих в группу от конкретной фирмы). Скажем, если для О. А. Кабанова, обучающегося от АО "Метро", установлен индивидуальный бонус 1,5 %, и кроме того, для всего АО "Метро" установлен бонус 5,7 %, то внесенная за И. П. Иванова плата в 1000 рублей автоматически засчитывается как сумма в 1072 рублей.

Таким образом, структура справочника "Учащиеся" определена. Далее на очереди создание пользовательской формы списка (форма по умолчанию нас не устроит, поскольку в дальнейшем нам потребуется внести в нее изменения).

В окне редактирования объекта конфигурации на вкладке **Формы** создадим представление справочника — форму списка (рис. 6.41). Далее можно перейти в режим **1С:Предприятие** и заполнить справочник данными (пример заполнения показан на рис. 6.42).

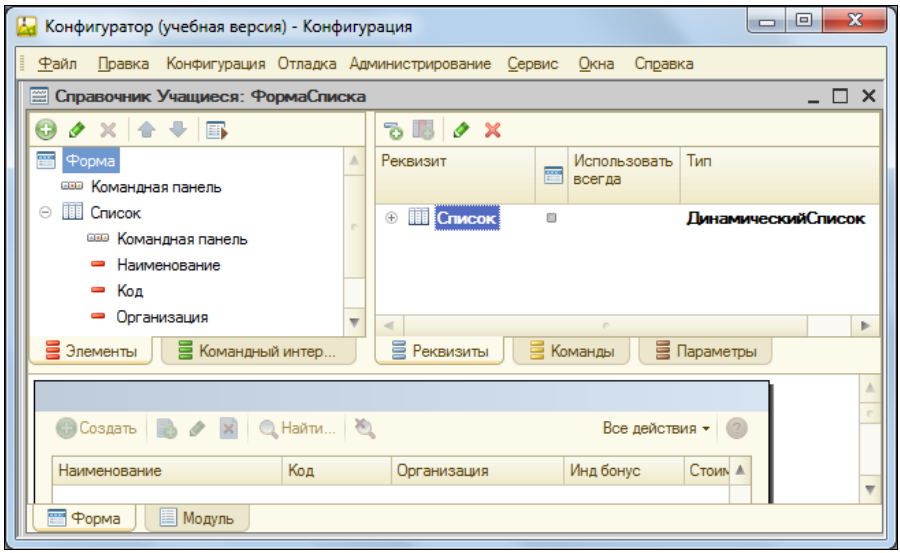


Рис. 6.41. Форма списка справочника "Учащиеся"

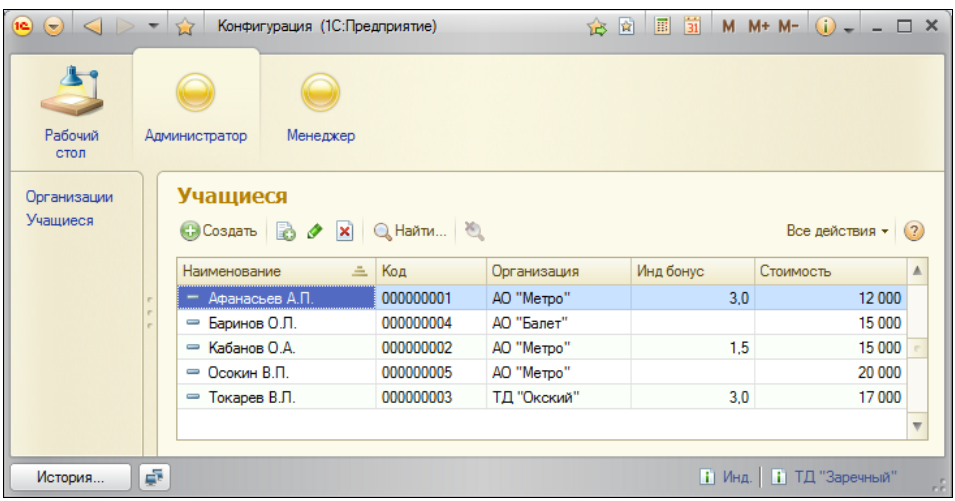


Рис. 6.42. Заполнение справочника "Учащиеся" в режиме **1С:Предприятие**

Следующая практическая задача заключается в создании регистра накопления с именем **РегистрОплаты**, в котором будет аккумулироваться информация об оплаченных суммах.

Регистр накопления по оплате

В рассматриваемой задаче для руководства учебной организации большой интерес представляет информация о том, кто и сколько оплатил за обучение. Для фиксирования оплаты создадим новый объект конфигурации — в окне **Конфигурация** добавим новый регистр накопления, который назовем **РегистрОплаты** (рис. 6.43).

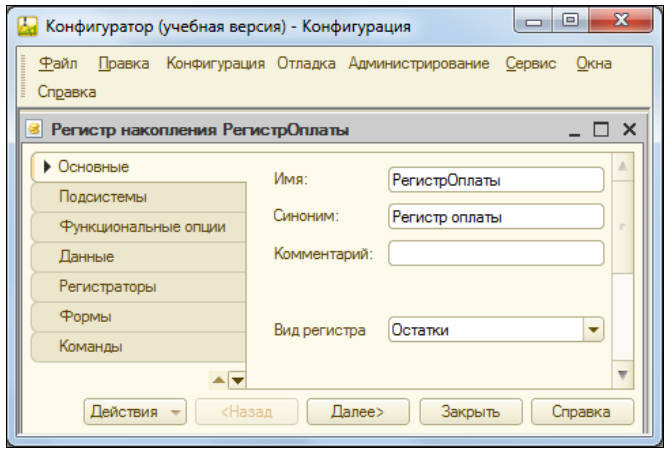


Рис. 6.43. Окно редактирования объекта **РегистрОплаты**

Важный момент, на который следует обратить внимание на этом рисунке, — поле со списком **Вид регистра**, где выберем в качестве вида регистра — **Остатки**.

После начальных действий необходимо перейти в окне редактирования объекта конфигурации на вкладку **Данные** и создать параметры данного регистра:

- **Измерения Учащийся** (тип данных — **СправочникСсылка.Учащиеся**);
- **Ресурсы Сумма** (тип данных — **Число**);
- **Ресурсы СуммаСБонусом** (тип данных — **Число**).

На рис. 6.44 показана структура создаваемого регистра накопления (содержание вкладки **Данные** окна редактирования объекта конфигурации). Однако сразу после ее создания переходить в режим **1С:Предприятие** бессмысленно — необходимые действия в конфигураторе еще не завершены. Ведь, как мы уже знаем, создание

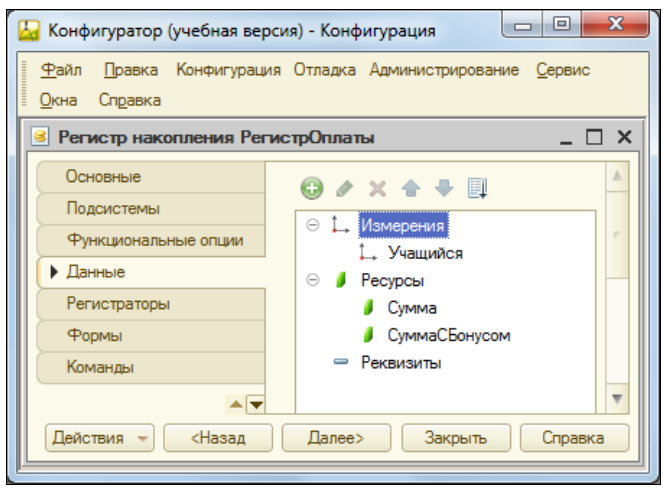


Рис. 6.44. Структура регистра накопления для учета оплаты

регистра накопления без документов, которые с ним взаимодействуют (формируют движения по регистру), не допускается.

Поэтому нужно создать документ "Оплата", в котором с помощью программной процедуры будет реализовано выполнение движений по созданному регистру **РегистрОплаты**.

Документ *Оплата*

При описании разработки документа "Оплата" подчеркнем только наиболее информативные моменты практических действий. На рис. 6.45 показана вкладка **Данные** окна редактирования объекта конфигурации. Видно, что область шапки документа содержит два реквизита:

- **Организация** (тип данных — **СправочникСсылка.Организации**);
- **Бонус** (тип данных — **Число** с точностью до одного знака, что позволяет ограничить отображение процентов десятыми долями).

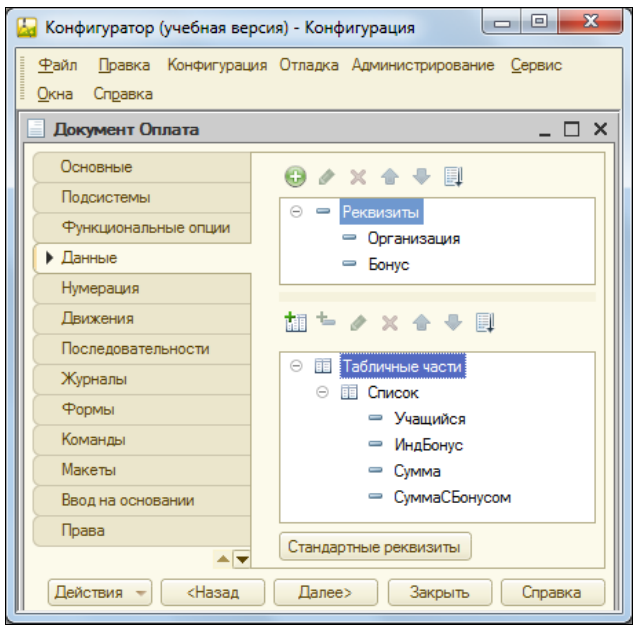


Рис. 6.45. Структура документа "Оплата"

Для табличной части документа выберем имя **Список** и внесем в нее необходимые реквизиты:

- **Учащийся** (тип данных — **СправочникСсылка.Учащиеся**);
- **ИндБонус** (тип данных — **Число** с точностью до одной десятой) для внесения бонусов индивидуальных слушателей курсов;
- **Сумма** (тип данных — **Число**) для внесения информации по оплачиваемой сумме;

- **СуммаСБонусом** (тип данных — **Число**) для внесения информации по сумме оплаты с учетом как индивидуальных, так и групповых бонусов (для слушателей от определенной организации). В дальнейшем с помощью программной процедуры значение этого реквизита будет вычисляться автоматически.

Перейдем теперь на вкладку **Движения**, где от нас требуется определить, в каких регистрах будут происходить изменения при проведении документа "Оплата". В нашей конфигурации мы располагаем лишь одним регистром накопления — **РегистрОплаты**, и только он присутствует в окне. От нас требуется сделать отметку во флажке напротив его названия.

Далее можно пойти двумя путями: либо самостоятельно написать процедуру, которая будет формировать движения при проведении подобных документов, либо воспользоваться возможностями конструктора движений. Выберем второй путь и, используя конструктор движений, автоматически получим необходимую программную процедуру в результате несложного диалога.

Конструктор движений

В режиме диалога с конструктором движений ключевым (в плане построения процедуры) является окно, представленное на рис. 6.46. Здесь требуется установить соответствие полей и выражений, что выполняется с помощью кнопки **Заполнить выражения**. После этого осталось только щелкнуть по кнопке **ОК**, что приведет к созданию конструктором программной процедуры (листинг 6.13), которая формирует движения по регистру при проведении документа.

Обратим внимание еще на одну важную деталь в окне конструктора движений регистров на рис. 6.46. Переключатель **Тип движения регистра** необходимо установить в значение **Приход**. Это соответствует добавлению ресурса данного регистра при проведении документа.

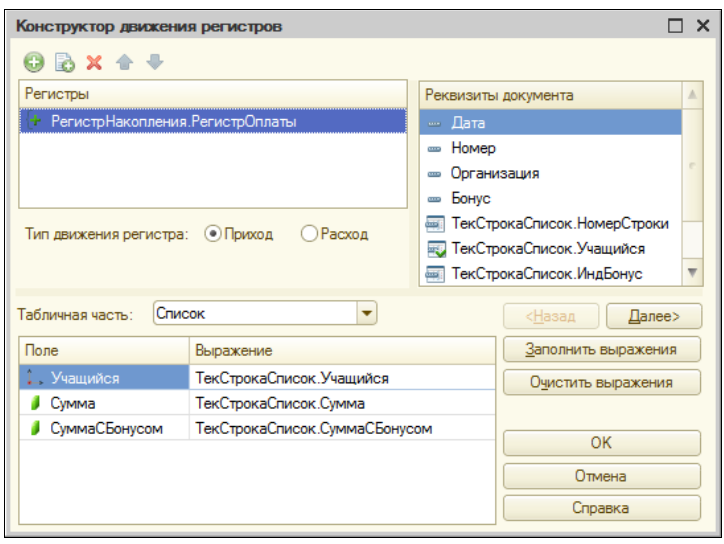


Рис. 6.46. Заполнение выражений в окне конструктора движений регистров

Листинг 6.13. Процедура, выполняемая при проведении документа Оплата

```
Процедура ОбработкаПроведения(Отказ, Режим)
//{{__КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
// Данный фрагмент построен конструктором.
  Для Каждого ТекСтрокаСписок Из Список Цикл
    // регистр РегистрОплаты Приход
    Движение = Движения.РегистрОплаты.Добавить();
    Движение.ВидДвижения = ВидДвиженияНакопления.Приход;
    Движение.Период = Дата;
    Движение.Учащийся = ТекСтрокаСписок.Учащийся;
    Движение.Сумма = ТекСтрокаСписок.Сумма;
    Движение.СуммаСБонусом = ТекСтрокаСписок.СуммаСБонусом;
 КонецЦикла;
// записываем движения регистров
Движения.РегистрОплаты.Записать();
//}}__КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
КонецПроцедуры
```

Основное содержание процедуры `ОбработкаПроведения` составляет цикл, в котором перебираются строки табличной части `Список` документа "Оплата":

```
Для Каждого ТекСтрокаСписок Из Список Цикл
```

При каждом проходе цикла с помощью метода `Добавить()` создается новая запись в регистре. В последующих строках процедуры устанавливаются значения полей регистра в соответствии с данными из очередной строки табличной части документа. Созданные в цикле записи сохраняются в базе данных с помощью метода `Записать`.

Хотя наша работа по конструированию рассматриваемого документа еще не окончена, сейчас стоит перейти в режим **1С:Предприятие** и проверить результат наших действий (протестировать созданный документ).

В настоящее время от сотрудника, работающего с документом (рис. 6.47), требуется самостоятельно заполнять все поля создаваемого документа. Так, внесение фамилии учащегося не приводит к заполнению поля **ИндБонус**, так же, как ввод названия организации не приводит к автоматическому заполнению поля **Бонус**. Эту ручную работу мы должны автоматизировать.

Планы по доработке формы документа

Рассмотрим возможности для улучшения интерфейсной составляющей документа для автоматизации работы пользователя в процессе ввода данных. Начнем с анализа последовательности действий пользователя при работе с документом. В режиме **1С:Предприятие** при создании нового документа сотрудник офиса в области шапки должен указать конкретную организацию, по которой он собирается вносить в электронную базу оплаченные суммы.

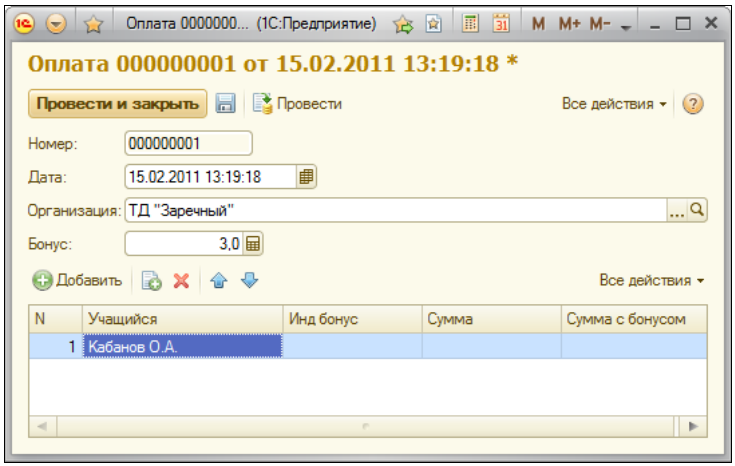


Рис. 6.47. Форма документа "Оплата" в режиме **1С:Предприятие**

Примечание

Предполагается, что с помощью одного документа осуществляется ввод данных по оплате для слушателей от определенной организации.

Если мы работаем с учащимися, записавшимися на курсы самостоятельно, то в поле ввода **Организация** следует выбрать значение **Инд** (см. рис. 6.39). Это значение мы условно указали, чтобы отмечать индивидуальных слушателей.

После выбора названия организации список учащихся (табличная часть документа) должен автоматически заполняться фамилиями учащихся, относящихся либо к конкретной организации, либо проходящими обучение по индивидуальной программе. Это заполнение мы реализуем с помощью программной процедуры. После этого от пользователя, работающего с формой, потребуется только внести суммы оплат напротив соответствующих фамилий.

В справочнике "Организация" вместе с названием организации присутствует еще реквизит **Бонус**. Значение бонуса после заполнения в документе поля **Организация** должно автоматически заноситься в соответствующее поле шапки документа.

Также в справочнике "Учащиеся" имеется реквизит **ИндБонус**, который определяет индивидуальный бонус либо штрафные санкции для конкретного учащегося. При заполнении табличной части документа в колонку **ИндБонус** должна автоматически заноситься информация об индивидуальном бонусе из справочника "Учащиеся".

Примечание

По замыслу реквизит **ИндБонус** ориентирован на работу с индивидуальными слушателями, однако в каких-то ситуациях его можно использовать и для учащихся от организаций (у них может быть бонус от организации, а также еще и свой индивидуальный).

После этого от сотрудника, работающего с документом, требуется внести в поле **Сумма** оплаченные суммы напротив каждого слушателя (скажем, исходя из имею-

щегося бумажного документа). В процессе этих действий после внесения очередной суммы соответствующее поле **СуммаСБонусом** должно автоматически заполняться, исходя из различных бонусов. При положительном бонусе учащемуся засчитывается в качестве оплаты бóльшая сумма, чем он реально оплатил. При этом пользователь прямо на форме документа может поменять значения полей **Бонус** и **ИндБонус**, что также должно приводить к автоматическому пересчету в поле **СуммаСБонусом**.

Все вышесказанное можно рассматривать как техническое задание для последующих действий. Наша задача будет заключаться в том, чтобы обеспечить такое функционирование формы документа с помощью программных процедур.

Программирование формы документа

Начнем реализацию поставленных задач, и так как нам потребуется программировать события, связанные с элементами на форме, то создадим пользовательскую форму документа "Оплата". Для этого в окне редактирования объекта конфигурации следует перейти на вкладку **Формы** и щелчком по пиктограмме с изображением лупы напротив раздела **Основные формы документа** начать диалог с конструктором. Выполнив несложные действия в окне конструктора, мы увидим на экране результат (рис. 6.48).

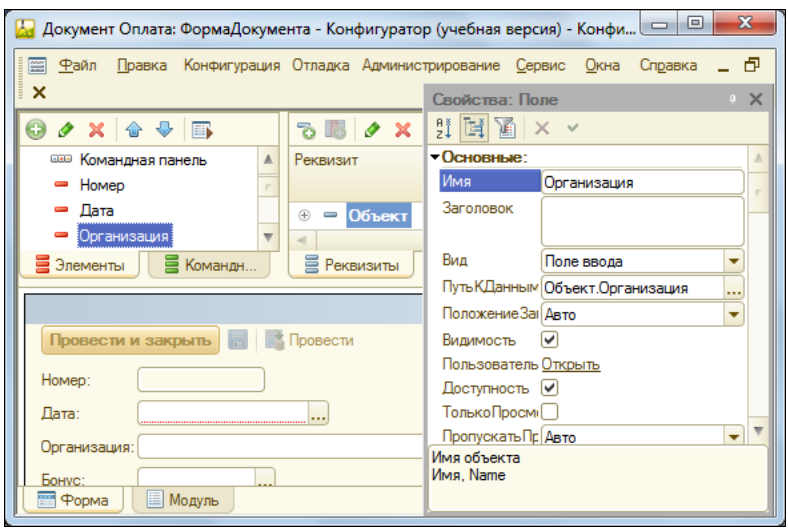


Рис. 6.48. Форма документа "Оплата"

В шапке документа расположено поле **Бонус**, которое должно автоматически заполняться при указании названия фирмы в поле **Организация**. В этом случае из справочника "Организации" извлекается значение реквизита **Бонус** и вносится в соответствующее поле в шапке формы документа.

Откроем палитру свойств поля **Организация** и обратимся к событию **ПриИзменении** (щелкнем по пиктограмме с изображением лупы). В результате на экране бу-

дет отображена сформированная заготовка процедуры обработчика этого события в модуле данной формы. Необходимо оформить обработку события ПриИзменении согласно листингу 6.14.

Кроме клиентской процедуры ОрганизацияПриИзменении требуется серверная функция ПолучитьБонус и серверная процедура ЗаполнитьТабЧасть. Функция ПолучитьБонус (листинг 6.15) в качестве входного параметра принимает ссылку на выбранную организацию в шапке документа. Отметим, что данная функция не требует передачи контекста формы на сервер.

Процедура ЗаполнитьТабЧасть (листинг 6.16) обеспечивает заполнение табличной части документа фамилиями учащихся.

Листинг 6.14. Процедура, выполняемая при заполнении поля Организация

```
&НаКлиенте
Процедура ОрганизацияПриИзменении(Элемент)
  Объект.Бонус=ПолучитьБонус(Объект.Организация);
  ЗаполнитьТабЧасть();
КонецПроцедуры
```

Листинг 6.15. Серверная функция ПолучитьБонус

```
&НаСервереБезКонтекста
Функция ПолучитьБонус(Фирма)
  Возврат Фирма.Бонус ;
КонецФункции
```

Листинг 6.16. Серверная процедура ЗаполнитьТабЧасть

```
&НаСервере
Процедура ЗаполнитьТабЧасть()
  Запрос = Новый Запрос;
  Запрос.Текст = "ВЫБРАТЬ  Ссылка,ИндБонус
  |  ИЗ  Справочник.Учащиеся
  |  ГДЕ  Организация = &НазваниеОрганизации";
  Запрос.УстановитьПараметр("НазваниеОрганизации",
          Объект.Организация);
  Результат = Запрос.Выполнить().Выбрать();
  Объект.Список.Очистить();
  Пока Результат.Следующий() > 0 Цикл
     НоваяСтрока = Объект.Список.Добавить();
     НоваяСтрока.Учащийся = Результат.Ссылка;
     НоваяСтрока.ИндБонус = Результат.ИндБонус;
 КонецЦикла;
КонецПроцедуры
```

Процедура `ЗаполнитьТабЧасть` реализует автоматическое заполнение табличной части документа и требует некоторого комментария. Необходимо сделать так, чтобы табличная часть `Список` (список учащихся) заполнялась в соответствии с информацией из справочника "Учащиеся". В данной процедуре используется запрос, в котором в качестве значения параметра выступает указанная в форме документа организация. Это так называемый параметрический запрос, с которым мы уже неоднократно встречались. В результате выполнения запроса отбираются учащиеся, относящиеся к указанной пользователем организации. Далее в цикле перебираются все элементы полученной в результате выполнения запроса выборки. Это используется для заполнения табличной части (`Список`) документа, которая предварительно очищается:

```
Объект.Список.Очистить();
```

а затем программно заполняется фамилиями учащихся из только что указанной организации. Для имеющихся данных результат работы в режиме **1С:Предприятие** приведен на рис. 6.49. Отметим, что вместе с заполнением табличной части учащимися в соответствующее поле заносится индивидуальный бонус (если он имеется).

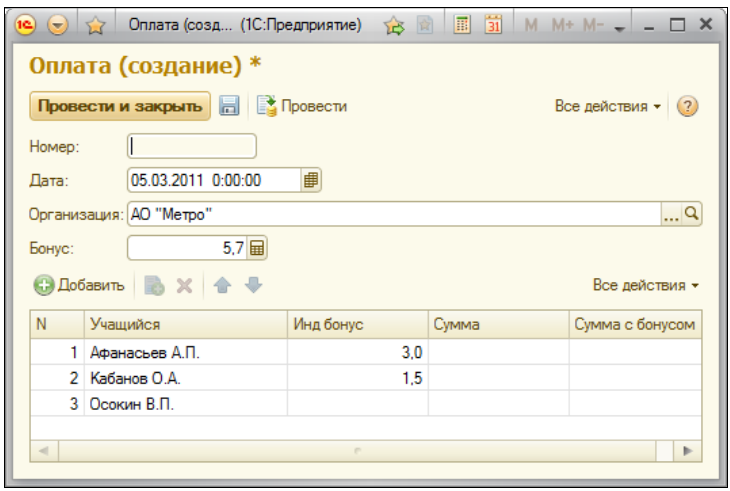


Рис. 6.49. Заполнение табличной части документа "Оплата"

Теперь от пользователя требуется ввести сумму в столбец с подписью **Сумма** напротив учащегося в соответствии с имеющимися данными. При этом после ввода этой суммы системой автоматически должно быть вычислено значение в колонке **Сумма с бонусом**. Это реализуется с помощью процедуры, представленной в листинге 6.17.

Листинг 6.17. Процедура, выполняемая при заполнении поля Сумма

```
&НаКлиенте
Процедура СписокСуммаПриИзменении(Элемент)
  ТабличнаяЧасть = Элементы.Список.ТекущиеДанные;
```

```
  БонусДляОрганизации = Объект.Бонус;
  ИндивидуальныйБонус = ТабличнаяЧасть.ИндБонус;
  ИтогБонус = БонусДляОрганизации + ИндивидуальныйБонус;
  ТабличнаяЧасть.СуммаСБонусом =
        ТабличнаяЧасть.Сумма * (1 + ИтогБонус/100);
КонецПроцедуры
```

В результате мы получим необходимое заполнение табличной части документа "Оплата" (рис. 6.50).

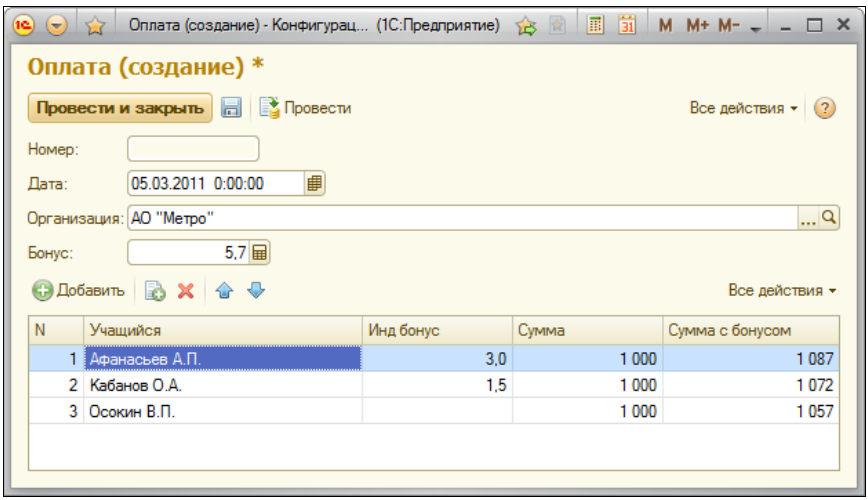


Рис. 6.50. Заполнение табличной части документа "Оплата"

При изменении бонуса для учащегося в форме документа также необходимо выполнить подобный пересчет. Для этого следует обеспечить соответствующее оформление процедуры `ПриИзменении` для поля **ИндБонус** (листинг 6.18).

Листинг 6.18. Процедура, выполняемая при заполнении поля ИндБонус

```
&НаКлиенте
Процедура СписокИндБонусПриИзменении(Элемент)
ТабличнаяЧасть = Элементы.Список.ТекущиеДанные;
БонусДляОрганизации = Объект.Бонус;
ИндивидуальныйБонус = ТабличнаяЧасть.ИндБонус;
ИтогБонус = БонусДляОрганизации + ИндивидуальныйБонус;
ТабличнаяЧасть.СуммаСБонусом =
   ТабличнаяЧасть.Сумма * (1 + ИтогБонус/100);
КонецПроцедуры
```

Таким образом, пересчет будет выполняться и при изменениях в поле **ИндБонус**, и при изменениях в поле **Сумма**.

Осталось выполнить еще одну функциональную доработку формы — при изменении в поле **Бонус**, расположенном в шапке документа, необходимо пересчитать суммы (в поле **СуммаСБонусом**) для всех учащихся в табличной части документа. Для этого следует написать процедуру (листинг 6.19), выполняемую при изменениях в поле **Бонус**. Необходимые вычисления будут произведены в серверной процедуре `ПересчетТабЧасти` (листинг 6.20).

Результат пересчета табличной части документа продемонстрирован на рис. 6.51.

Листинг 6.19. Процедура, выполняемая при изменении поля Бонус

```
&НаКлиенте
Процедура БонусПриИзменении(Элемент)
  ПересчетТабЧасти();
КонецПроцедуры
```

Листинг 6.20. Серверная процедура ПересчетТабЧасти

```
 &НаСервере
 Процедура ПересчетТабЧасти()
   БонусДляОрганизации = Объект.Бонус;
   Для Каждого ТекСтрокаСписок Из  Объект.Список Цикл
      ИндивидуальныйБонус = ТекСтрокаСписок.ИндБонус;
      ИтогБонус = БонусДляОрганизации + ИндивидуальныйБонус;
      ТекСтрокаСписок.СуммаСБонусом =
      ТекСтрокаСписок.Сумма * (1 + ИтогБонус/100);
   КонецЦикла;
КонецПроцедуры
```

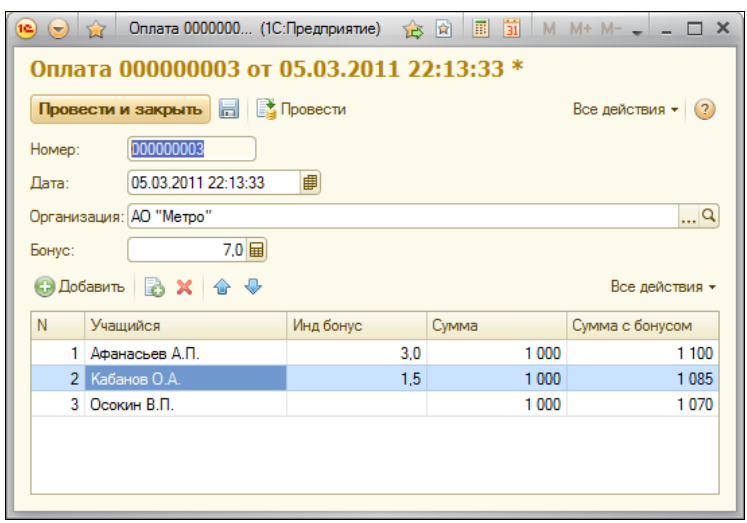


Рис. 6.51. Пересчет табличной части документа в режиме **1С:Предприятие**

Таким образом, необходимые действия по автоматизации заполнения формы и пересчету оплаченных сумм при изменении бонусов в документе "Оплата" выполнены. В настоящий момент документ обладает необходимыми сервисными функциями и выполняет движения по регистру накопления. Теперь можно перейти к разработке документа, реализующего движение денежных средств в противоположном направлении (возврат денег учащимся по какой-либо причине).

Документ *Возврат*

В случае отказа от обучения либо переплаты требуется регистрировать возврат денег учащимся. Приступим к разработке нового документа, реализующего такую регистрацию. На рис. 6.52 показана вкладка **Данные** окна редактирования объекта конфигурации. Дополнительных реквизитов в области шапки мы создавать не будем, а в табличной части **Список** разместим два реквизита:

- **Учащийся** (тип данных — **СправочникСсылка.Учащиеся**);
- **Сумма** (тип данных — **Число**).

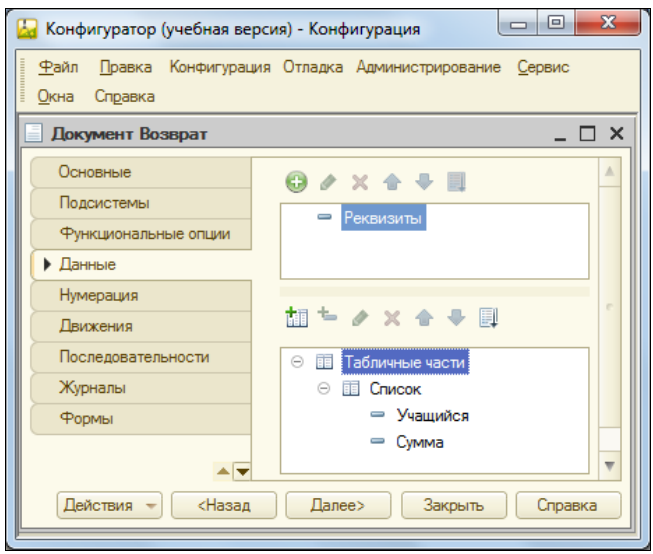


Рис. 6.52. Вкладка **Данные** документа "Возврат"

Следующая вкладка окна редактирования документа "Возврат", на которой потребуется выполнить действия, — **Движения**. В качестве регистра накопления необходимо указать **РегистрОплаты**. Далее (как мы уже поступали при создании документа "Оплата") воспользуемся конструктором движений. Основное окно конструктора движений следует заполнить в соответствии с рис. 6.53. Обратите внимание на то, что при создании документа "Возврат", в отличие от ранее разработанного документа "Оплата", необходимо сделать установку **Расход**, после чего остается щелкнуть по кнопке **ОК**. В результате мы увидим на экране текст процедуры формирования движений по регистру накопления при проведении документа (листинг 6.21).

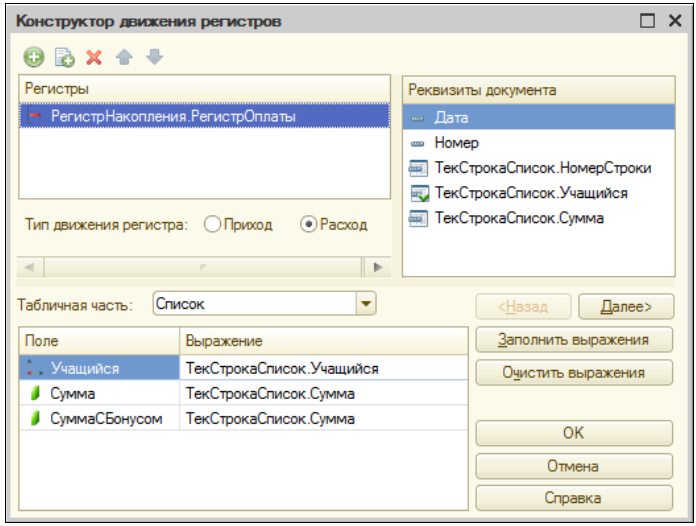


Рис. 6.53. Заполнение выражений в окне конструктора движений регистров

Листинг 6.21. Процедура, выполняемая при проведении документа Возврат

```
Процедура ОбработкаПроведения(Отказ, Режим)
//{{__КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
// Данный фрагмент построен конструктором.
  Движения.РегистрОплаты.Записывать = Истина;
  Для Каждого ТекСтрокаСписок Из Список Цикл
    // регистр РегистрОплаты Расход
    Движение = Движения.РегистрОплаты.Добавить();
    Движение.ВидДвижения = ВидДвиженияНакопления.Расход;
    Движение.Период = Дата;
    Движение.Учащийся = ТекСтрокаСписок.Учащийся;
    Движение.Сумма = ТекСтрокаСписок.Сумма;
    Движение.СуммаСБонусом = ТекСтрокаСписок.Сумма;
   КонецЦикла;
 //}}__КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
КонецПроцедуры
```

Принципиальное отличие данной процедуры от представленной в листинге 6.13 заключается в другой установке свойства `ВидДвижения`:

```
Движение.ВидДвижения = ВидДвиженияНакопления.Расход.
```

Просмотр прихода в документе *Возврат*

Рассмотрим организацию следующего функционала. В документе "Возврат" при вводе суммы возврата полезно для сотрудника учебного центра организовать удобный просмотр информации по поступлению денежных средств. На рис. 6.54 проде-

монстрирован результат, который мы собираемся получить. Так, после внесения в табличную часть документа "Возврат" очередного учащегося в окно сообщений выводится информация о произведенной им оплате (общей внесенной сумме и общей сумме с учетом бонусов).

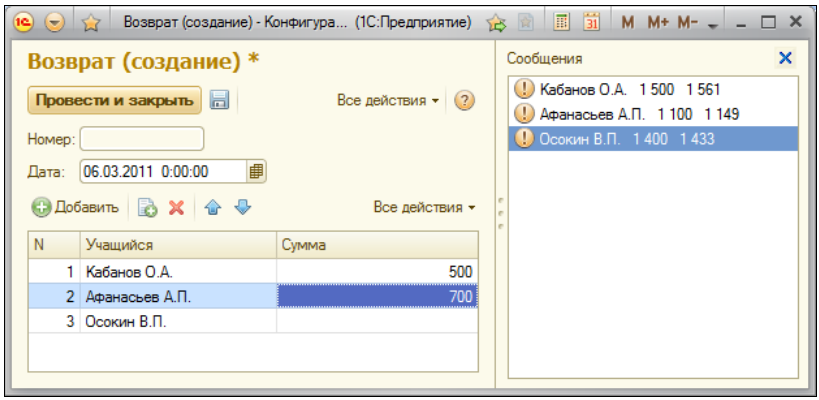


Рис. 6.54. Вывод в окно сообщений вспомогательной информации

Для реализации этого функционала обратимся к форме документа "Возврат", а именно для начала создадим пользовательскую форму данного объекта конфигурации (рис. 6.55).

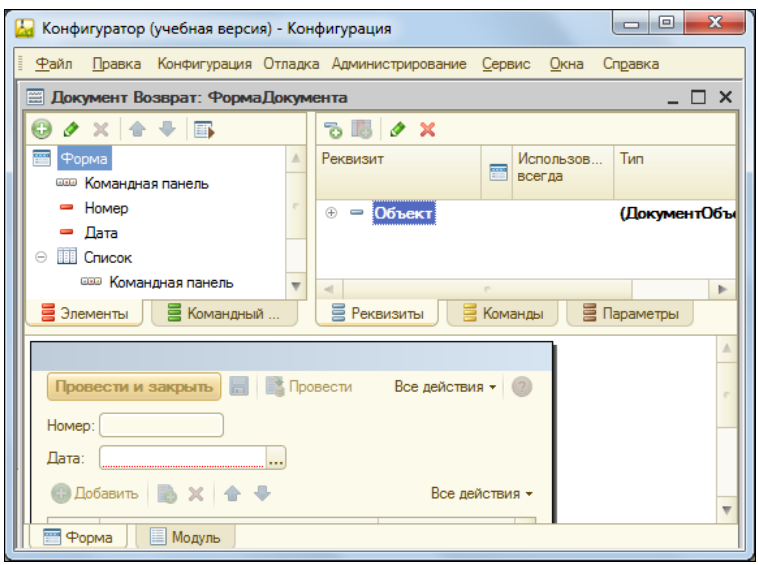


Рис. 6.55. Форма документа "Возврат"

Далее обратимся к полю **СписокУчащийся** и откроем его палитру свойств (рис. 6.56). Нам необходимо запрограммировать реакцию на событие **ПриИзменении** (для этого воспользуемся щелчком мыши по пиктограмме с изображением лупы).

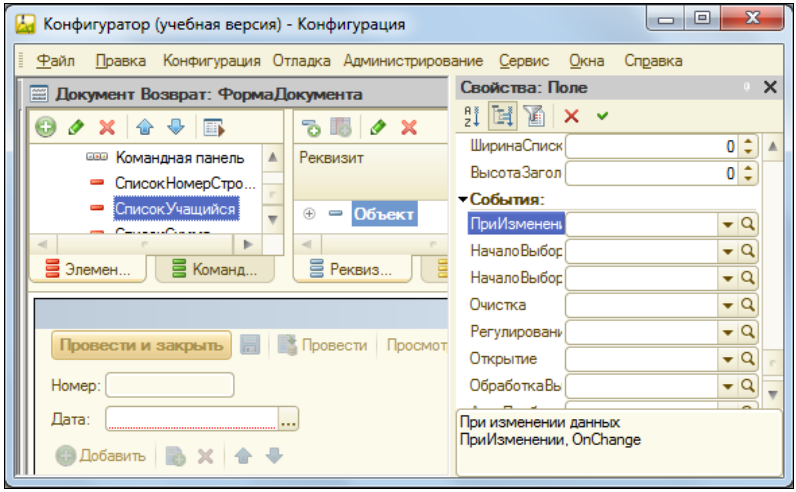


Рис. 6.56. Палитра свойств поля **СписокУчащийся**

Оформим процедуру обработки события `ПриИзменении` для поля `СписокУчащийся` в варианте, представленном в листинге 6.22.

Листинг 6.22. Процедура обработки события ПриИзменении в поле СписокУчащийся

```
&НаКлиенте
Процедура СписокУчащийсяПриИзменении(Элемент)
    ТабличнаяЧасть = Элементы.Список.ТекущиеДанные;
    ПолучитьИнформацию(ТабличнаяЧасть.Учащийся);
КонецПроцедуры
```

Кроме клиентской процедуры нам понадобится еще и серверная функция `Получить-Информацию` (листинг 6.23). В ней мы воспользуемся запросом к виртуальной таблице `РегистрОплаты.Обороты`. Из данной таблицы выбрана информация по приходу суммы и суммы с бонусом.

Таким образом, теперь можно перейти в режим **1С:Предприятие** и получить результат, который мы уже видели в начале раздела.

Листинг 6.23. Серверная функция ПолучитьИнформацию

```
&НаСервереБезКонтекста
Процедура ПолучитьИнформацию(Учащийся)
  Запрос = Новый Запрос;
  Запрос.Текст = "ВЫБРАТЬ  Учащийся,
  |  СуммаПриход, СуммаСБонусомПриход
  | ИЗ  РегистрНакопления.РегистрОплаты.Обороты
  | ГДЕ  Учащийся=&Фамилия";
   Запрос.УстановитьПараметр("Фамилия",
        Учащийся);
```

```
  Результат = Запрос.Выполнить().Выбрать();
  Сообщение=Новый СообщениеПользователю;
  Пока Результат.Следующий() > 0 Цикл
       Сообщение.Текст=Строка(Результат.Учащийся) +
          "    " + Строка(Результат.СуммаПриход) +
                "    " + Строка(Результат.СуммаСБонусомПриход);
      Сообщение.Сообщить();
   КонецЦикла;
КонецПроцедуры
```

Отчет по оплатам

Руководству любой организации требуется интегральная информация. Наша конфигурация к настоящему моменту располагает документами для регистрации фактов оплаты и возврата денежных средств, а также в ней имеется регистр накопления для фиксации перемещения денежных средств.

Рассмотрим формирование отчета по учащимся непосредственно из формы документа "Возврат". Так, в форме этого документа мы поместим кнопку, щелчком по которой будет открываться табличный документ с необходимой информацией.

Перейдем в окно редактирования документа "Возврат". На вкладке **Макеты** для начала требуется разработать макет, представленный на рис. 6.57.

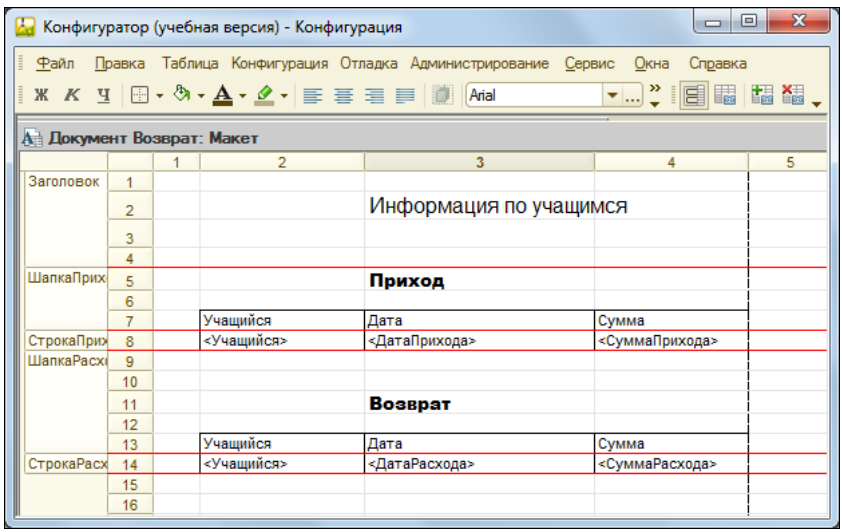


Рис. 6.57. Макет для табличного документа

После этого откроем форму документа "Возврат" в редакторе форм. Здесь требуется добавить в форму новую команду, которую назовем **ПросмотрПоУчащимся** (рис. 6.58). Процедура обработки данной команды приведена в листинге 6.24. В процессе ее выполнения требуется обращение к серверной процедуре `ПечатьВДокумент` (листинг 6.25).

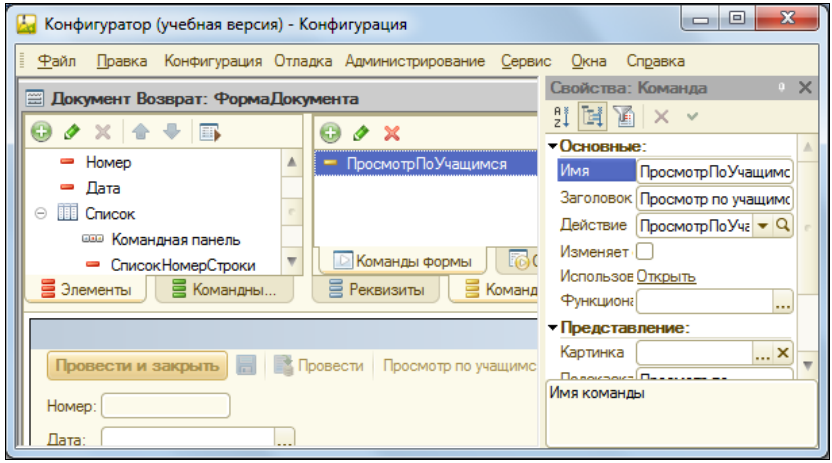


Рис. 6.58. Добавление команды **ПросмотрПоУчащимся** в форме документа

Листинг 6.24. Процедура обработки команды ПросмотрПоУчащимся

```
&НаКлиенте
Процедура ПросмотрПоУчащимся(Команда)
  ТабДок = Новый ТабличныйДокумент;
  ПечатьВДокумент(ТабДок);
  ТабДок.ОтображатьСетку = Ложь;
  ТабДок.Защита = Ложь;
  ТабДок.ТолькоПросмотр = Ложь;
  ТабДок.ОтображатьЗаголовки = Ложь;
  ТабДок.Показать();
КонецПроцедуры
```

Листинг 6.25. Серверная процедура обработки команды ПечатьВДокумент

```
&НаСервереБезКонтекста
Процедура ПечатьВДокумент(ТабДок)
  ТабДок.Очистить();
  Макет = Документы.Возврат.ПолучитьМакет("Макет");
  Заголовок= Макет.ПолучитьОбласть("Заголовок");
  ТабДок.Вывести(Заголовок);
  ШапкаПриход = Макет.ПолучитьОбласть("ШапкаПриход");
  ТабДок.Вывести(ШапкаПриход);
  Запрос = Новый Запрос;
  Запрос.Текст = "ВЫБРАТЬ  Учащийся.Наименование КАК Студент ,
  | Ссылка.Дата, СуммаСБонусом
  | ИЗ  Документ.Оплата.Список
  | УПОРЯДОЧИТЬ ПО Студент";
  Результат = Запрос.Выполнить().Выбрать();
  Пока Результат.Следующий() > 0 Цикл
```

```
    Область = Макет.ПолучитьОбласть("СтрокаПриход");
    Область.Параметры.Учащийся = Результат.Студент;
    Область.Параметры.ДатаПрихода = Результат.Дата;
    Область.Параметры.СуммаПрихода = Результат.СуммаСБонусом;
    ТабДок.Вывести(Область);
  КонецЦикла;
  ШапкаРасход = Макет.ПолучитьОбласть("ШапкаРасход");
  ТабДок.Вывести(ШапкаРасход);
  Запрос = Новый Запрос;
  Запрос.Текст = "ВЫБРАТЬ  Учащийся.Наименование КАК Студент ,
  | Ссылка.Дата, Сумма
  | ИЗ  Документ.Возврат.Список
  | УПОРЯДОЧИТЬ ПО Студент";
  Результат = Запрос.Выполнить().Выбрать();
  Пока Результат.Следующий() > 0 Цикл
    Область = Макет.ПолучитьОбласть("СтрокаРасход");
    Область.Параметры.Учащийся = Результат.Студент;
    Область.Параметры.ДатаРасхода = Результат.Дата;
    Область.Параметры.СуммаРасхода = Результат.Сумма;
    ТабДок.Вывести(Область);
  КонецЦикла;
КонецПроцедуры
```

Далее осталось поместить в форме документа кнопку для выполнения команды **ПросмотрПоУчащимся** и перейти в режим **1С:Предприятие** для проверки работоспособности разработки (рис. 6.59).

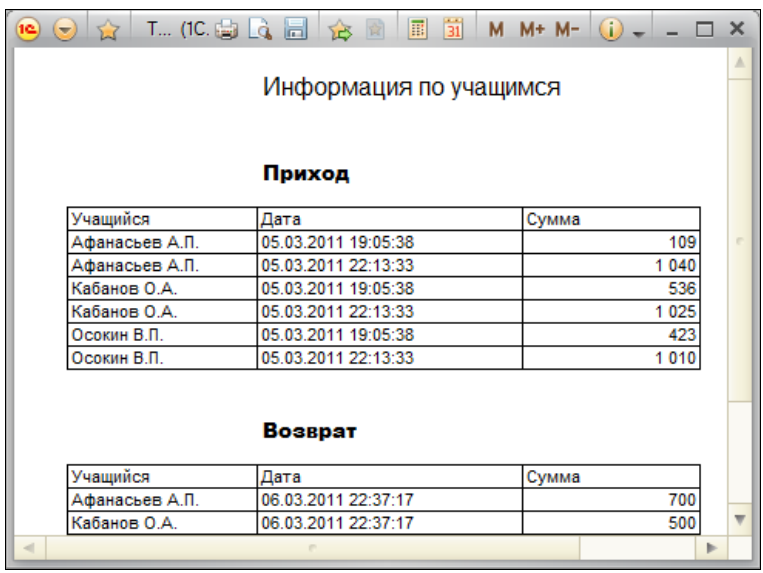


Рис. 6.59. Табличный документ с информацией по приходу и расходу средств

Итоги

Программирование событий, связанных с формами и элементами управления, расположенными на них, является одним из основных применений встроенного языка платформы 1С:Предприятие 8.2. В этой главе мы рассмотрели обработку событий, связанных с изменениями данных в формах документов и справочников. При этом практически во всех рассмотренных примерах для извлечения информации нам требовался язык запросов.

В целом после рассмотрения всех глав у читателя должно сформироваться полное представление о технологии разработки прикладных решений на платформе 1С:Предприятие 8.2.

ПРИЛОЖЕНИЕ

Описание прилагаемого компакт-диска

На прилагаемом к книге компакт-диске вы найдете информационные базы, разработанные в книге, а также файлы приложения Microsoft Word, которые упоминаются в *главе 3*.

Названия папок соответствуют названиям глав книги. Кроме того, отдельно приведены информационные базы в архивированном виде.

Для установки информационных баз необходимо:

1. Скопировать папку, содержащую интересующую вас информационную базу с компакт-диска на жесткий диск.
2. В окне запуска программы 1С:Предприятие 8.2 нажать кнопку **Добавить**.
3. В открывшемся окне выбрать вариант **Добавление в список существующей информационной базы**.
4. Подобрать любое название для информационной базы.
5. Указать папку, куда вы ранее скопировали информационную базу.

Литература

1. Белоусов П. С., Островерх А. В. 1С:Предприятие от 8.0 до 8.1. — М.: 1С-Паблишинг, 2008. — 286 с.: ил.
2. Габец А. П., Гончаров Д. И. 1С:Предприятие 8.0. Простые примеры разработки. — М.: 1С-Паблишинг, 2005. — 420 с.: ил.
3. Радченко М. Г. 1С:Предприятие 8.0. Практическое пособие разработчика. Примеры и типовые приемы. — М.: 1С-Паблишинг, 2006. — 656 с.: ил.
4. Радченко М. Г. 1С:Предприятие 8.1. Практическое пособие разработчика. Примеры и типовые приемы. — М.: 1С-Паблишинг; СПб.: Питер, 2007. — 512 с.: ил.
5. Профессиональная разработка в системе 1С:Предприятие 8 / Под ред. М. Г. Радченко. — М.: 1С-Паблишинг; СПб.: Питер, 2006. — 808 с.: ил.
6. Радченко М. Г., Хрусталева Е. Ю. 1С:Предприятие 8.2. Практическое пособие разработчика. Примеры и типовые приемы. — М.: 1С-Паблишинг, 2009. — 874 с.: ил.
7. Анжеронок В. А. и др. Разработка управляемого интерфейса. — М.: 1С-Паблишинг, 2010. — 723 с.: ил.
8. Хрусталева Е. Ю. Разработка сложных отчетов в 1С:Предприятии 8. Система компоновки данных. — М.: 1С-Паблишинг, 2008. — 513 с.: ил.
9. Кашаев С. М. Программирование в Microsoft Excel на примерах. — СПб.: БХВ-Петербург, 2007. — 320 с.: ил. + Видеокурс (на CD-ROM).
10. Кашаев С. М. 1С:Предприятие 8. Учимся программировать на примерах. — СПб.: БХВ-Петербург, 2008. — 336 с.: ил. + CD-ROM.
11. Кашаев С. М. Офисные решения с использованием Microsoft Excel 2007 и VBA. — СПб.: Питер, 2008. — 317 с.: ил. + CD-ROM.
12. Кашаев С. М. 1С:Предприятие 8.1. Учимся программировать на примерах. — СПб.: БХВ-Петербург, 2009. — 400 с.: ил. + CD-ROM.

Предметный указатель

А

ActiveX 147

М

Microsoft Word 149

О

OLE 151

А

Агрегатная функция 181

В

Виртуальная таблица 105

Г

Гиперссылка оповещения 81

Д

Движения 97, 277
Дерево объектов конфигурации 21
Директива
◊ НаКлиенте 57
◊ НаСервере 57
◊ НаСервереБезКонтекста 71
Документ
◊ реквизит 91
◊ проведение 90
◊ шапка 63, 91

З

Запрет незаполненных значений 135
Запрос 59, 158
◊ NULL 207
◊ ВЫБРАТЬ 159
◊ ГДЕ 177
◊ ИЕРАРХИЯ 174
◊ ИМЕЮЩИЕ 191
◊ ИТОГИ 191, 193
◊ КАК 168
◊ КОЛИЧЕСТВО 182
◊ ЛЕВОЕ [ВНЕШНЕЕ] СОЕДИНЕНИЕ 206
◊ МАКСИМУМ 182
◊ МЕЖДУ 187
◊ МИНИМУМ 182
◊ ОБЪЕДИНИТЬ 196
◊ ПЕРВЫЕ 171
◊ ПОЛНОЕ [ВНЕШНЕЕ] СОЕДИНЕНИЕ 209
◊ ПРАВОЕ [ВНЕШНЕЕ] СОЕДИНЕНИЕ 208
◊ РАЗЛИЧНЫЕ 175
◊ СГРУППИРОВАТЬ ПО 184
◊ СОЕДИНЕНИЕ 204
◊ СРЕДНЕЕ 182
◊ СУММА 181
◊ УПОРЯДОЧИТЬ ПО 168

И

Иерархия 32
◊ групп и элементов 32
◊ элементов 32

Имя 25
Индексирование 135
Информационная база 18
Информационная панель 81
Истина 60
История 82

К

Код 28, 34
Команда 161
Комментарий 57
Конструктор
◊ движений 277, 294
◊ запроса 217
◊ макета 164
◊ настроек 223, 224, 229
◊ печати 51
◊ схемы компоновки данных 216
◊ форм 40, 117
Конфигуратор 20

Л

Лента 147
Логические операторы 178

М

Массив 151
Метаданные 15
Метод
◊ Open 152
◊ Вставить 151
◊ Выбрать 59, 162
◊ Вывести 59
◊ Выполнить 162
◊ Добавить 73, 100
◊ Записать 295
◊ Очистить 72
◊ Показать 60
◊ Получить 124, 270
◊ ПолучитьМакет 58
◊ ПолучитьПоследнее 124, 270
◊ Следующий 163
◊ СрезПоследних 274
◊ ТипЗаписи 192
◊ УстановитьПараметр 118, 186
Модуль 56

Н

Наименование 28, 34
Независимый режим записи 114, 129
Нумератор 91

О

Область системных команд 83
Обороты 276
Объединение результатов запросов 194
Объект
◊ СОМОбъект 152
◊ УправляемаяФорма 123
Объект конфигурации
◊ документ 22
◊ идентификатор 24
◊ перечисление 23
◊ подчиненный 26
◊ основной 23
◊ справочник 22
Окно редактирования объекта конфигурации 23, 24, 77
Отладка 29

П

Палитра свойств 25
Панель
◊ действий 80
◊ навигации 30, 80
◊ разделов приложения 79
Период 121, 189
Периодичность 134, 264
Подсистема 76, 80
Подсистемы подчиненные 79
Подчинение регистратору 114, 129
Приход 99
Процедура ОбработкаПроведения 100

Р

Рабочая область окна приложения 30
Рабочий стол 79
Расход 99
Регистр накопления
◊ движение 91
◊ измерение 92
◊ обороты 96, 105
◊ остатки 96
◊ приращение ресурсов 102

◊ ресурс 92
◊ тип движения 104
Регистр сведений 111, 129
◊ измерение 113
◊ непериодический 114
◊ периодический 121
◊ реквизит 113
◊ ресурс 113
Регистратор 102
Редактор
◊ макета 164
◊ форм 42
Режим 1С:Предприятие 15, 18, 29
Реквизит 34
Родитель 33

С

Свойство
◊ Защита 60
◊ ОтображатьЗаголовки 60
◊ ТолькоПросмотр 60
◊ ОтображатьСетку 59
◊ ЭтоГруппа 209
Система компоновки данных 213
Событие ПриИзменении 68, 269, 297
Справочник 27
◊ иерархический 32
◊ печатная форма 47
◊ реквизит 28
◊ табличная часть 39
◊ элемент 28
Схема компоновки данных 213

Т

Табличная часть 35, 65, 91
Табличный документ 164
Тип движения регистра 99
Толстый клиент 86
Тонкий клиент 86

Ф

Форма 31, 32, 261
◊ списка 30, 37
◊ элемента 31, 37
Функция Строка 163

Я

Язык запросов 16