Глава 2

Извлечение информации с помощью запросов

При работе в режиме 1С:Предприятие первичная информация фиксируется в разнообразных информационных структурах — справочниках, регистрах и документах. И если нас интересуют данные только в такой исходной форме, то больше в принципе ничего делать и не требуется. Так, всегда можно открыть необходимый справочник, просмотреть содержимое регистра или документа из списка документов.

Однако для управленческого состава любой организации особый интерес представляет информация в обработанном виде (отбор по параметрам, итоговые и сводные данные), — например, поступление товаров за указанный временной период с группировкой по фирмам-поставщикам.

Для решения подобных задач в системе 1С:Предприятие используется специальный объект, который называется *запрос*. По существу, любой запрос представляет собой требование к системе сделать выборку необходимой информации из базы данных. Данный объект наиболее часто используется при построении отчетов.

В тексте запроса описывается, какие таблицы информационной базы используются в качестве источников информации, а также указываются поля таблиц, которые требуется обрабатывать. Таблицы, участвующие в запросе, делятся на два основных класса: реальные и виртуальные. Их отличие в том, что реальные таблицы хранятся в базе данных, а виртуальные таблицы — нет. При обращении к информации виртуальных таблиц система для выполнения запроса автоматически собирает нужную информацию из реальных таблиц базы данных. В виртуальной таблице можно использовать параметры — действительное наполнение виртуальной таблицы может определяться значениями параметров, фактические значения которых задаются в тексте запроса.

Отдельный подкласс таблиц образуют так называемые объектные таблицы. В качестве объектной таблицы всегда выступает реальная таблица базы данных. Объектные таблицы предназначены для хранения состояния объектов системы 1С:Предприятие, таких как справочники, документы, регистры и др. Каждая объектная таблица имеет один из типов объектов системы 1С:Предприятие. Например, объектам типа `Справочник.Фирмы` соответствует одна таблица, а объектам `Справочник.Товары` — другая. Каждая отдельная запись объектной таблицы хранит состояние определенного объекта соответствующего типа. В соответствие с этим у каждой объектной таблицы определено поле `Ссылка` на текущую запись.

Кроме собственно отбора данных, у запросов есть еще много полезных свойств — выполнение группировки отобранных данных, вычисление итогов и др. Запросы удобно использовать в тех ситуациях, когда требуется сформировать сложную выборку данных, сгруппированную и отсортированную необходимым образом.

В некоторой степени с запросами мы уже встречались в главе 1 книги. Фактически рассмотренные в ней примеры являются неким "анонсом" для материала данной главы. Здесь технология составления запросов будет рассмотрена, начиная с основных положений. На разнообразных примерах вы получите практические навыки по извлечению необходимых сведений из информационной базы. И в последующих главах мы будем также часто работать с запросами, а наиболее широко они будут использоваться при составлении отчетов.

Рассматриваемые далее примеры ориентированы на знакомство с основными конструкциями запросов и последующее их использование в реальных практических задачах. Как и в главе 1, мы не ограничимся только справочной информацией, а проиллюстрируем почти все синтаксические конструкции запросов на практике.

Конструкции ВЫБРАТЬ и ИЗ

В качестве первого примера рассмотрим запрос, связанный с информацией, присутствующей в табличной части справочника Фирмы. В главе 1 при разработке данного справочника мы создали табличную часть КонтактныеЛица, в которую включили фамилии сотрудников, их должности и телефоны.

Допустим, нам необходимо составить список всех лиц, работающих в фирмах-партнерах. При этом в списке напротив фамилии каждого сотрудника должен быть указан номер его телефона. Также информацию о сотруднике желательно дополнить еще и названием организации, где он работает. И, разумеется, желательно список отсортировать по фамилиям сотрудников (по алфавиту в возрастающем порядке). Такой несложный запрос выглядит следующим образом:

```
ВЫБРАТЬ Сотрудник,Телефон,
      Ссылка.Наименование КАК Организация
   ИЗ Справочник.Фирмы.КонтактныеЛица
   УПОРЯДОЧИТЬ ПО Сотрудник
```

В результате его выполнения мы получим таблицу из набора строк с тремя столбцами. Число строк в таблице соответствует числу имеющихся лиц в табличной части всех элементов справочника Фирмы.

В приведенном запросе использовано уже знакомое ключевое слово ВЫБРАТЬ, позволяющее указать список полей для данного запроса. Другое ключевое слово ИЗ позволяет определить таблицы, участвующие в рассматриваемом запросе. В данной ситуации используется табличная часть КонтактныеЛица справочника Фирмы.

Необходимо прокомментировать важный момент. Так, при осуществлении запроса к табличной части справочника обращение к обычным (вне таблицы) его реквизитам производится через поле Ссылка. В данном случае информация о фирме заключена в автоматически создаваемом в любом справочнике реквизите Наименование, который располагается вне табличной части.

В приведенном запросе мы использовали ключевое слово КАК, позволяющее определить синоним для имени Ссылка.Наименование. Мы выбрали более короткий вари-

ант — Организация. Для упорядочения данных запроса используется конструкция УПОРЯДОЧИТЬ ПО, в которой необходимо указать название поля, по которому будет выполняться сортировка.

Вообще, существует много разнообразных синтаксических конструкций запросов, с которыми мы познакомимся ниже. Важный технический момент связан с выполнением запросов в системе 1С:Предприятие 8. В этом случае, кроме конструкций языка запросов, также потребуется уже в некоторой степени знакомый нам встроенный язык программирования системы 1С:Предприятие 8.

Начнем практическую работу с того, что в режиме конфигуратора откроем конфигурацию, ранее разработанную нами в главе 1. Результаты запросов потребуется где-то отражать, и для этого в последующих примерах мы будем использовать два варианта:

- элемент управления *поле списка*, располагаемый в форме;
- табличный документ (с ним мы уже встречались в главе 1).

Для начала создадим новую обработку, при этом непосредственные технические действия в конфигураторе будут аналогичны тем, которые мы использовали в предыдущих примерах при создании объектов конфигурации. Объект конфигурации Обработка весьма похож на уже встречавшийся объект Отчет. В функциональном плане они отличаются весьма незначительно. Как правило, отчеты используются при выводе информации в печатном виде. Обработки же применяются при реализации специфических алгоритмов извлечения данных.

Ключевой параметр любого объекта конфигурации — Имя, и для него в данном случае укажем значение ИзвлечениеИнформации (рис. 2.1).

В рассматриваемой обработке нам фактически потребуется подчиненный объект конфигурации — форма, в которой мы разместим необходимые элементы управления. Также мы воспользуемся уже знакомой вкладкой Модуль, предназначенной для ввода текстов программных процедур на встроенном языке.

Таким образом, перейдем на вкладку Формы, где с помощью кнопки панели инструментов начнем работу с конструктором создания новой формы. В первом появившимся окне следует выполнить установки в соответствии с рис. 2.2. Так, мы не будем отображать командные панели, установим переключатель типа формы в положение Форма обработки и укажем, что форма будет основной.

Теперь можно сразу щелкнуть на кнопке Готово, после чего перед нами откроется окно формы (рис. 2.3). В форме уже расположены знакомые элементы управления — *кнопки*, а на вкладке Модуль сформирована заготовка для программной процедуры (рис. 2.4) обработки щелчка на кнопке Выполнить.

Для дальнейшей работы следует разместить в форме элемент управления *поле списка* (этот элемент еще не встречался в примерах главы 1). Как мы уже знаем, элементы размещаются на форме с помощью команды Вставить элемент управления из меню Форма.

В системе 1С:Предприятие 8 существует объект *список значений*. Это не сохраняемый в базе данных объект, который предназначен для формирования динамических наборов данных. Данный объект может быть заполнен значениями любого типа.

> Объект *список значений* является коллекцией значений, составляющих список.

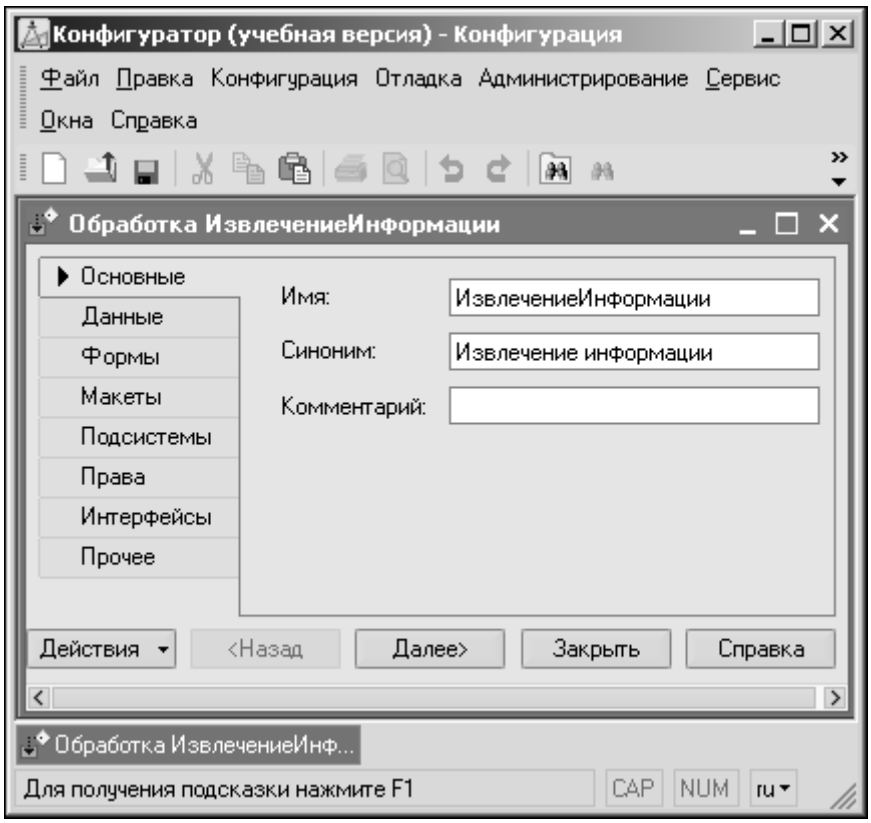

Рис. 2.1. Вкладка Основные *окна редактирования обработки* *ИзвлечениеИнформации*

Для визуального представления объекта *список значений* в системе 1С:Предприятие 8 можно использовать элемент управления *поле списка*, который предназначен для выбора одного или нескольких значений из списка. На рис. 2.5 показано размещение данного элемента управления в нашей форме.

Определимся с именами объектов, что необходимо для исключения противоречий с приводимой ниже процедурой. Что касается кнопки, то ее имя уже подобрано системой — Выполнить, а для элемента управления *поле списка* установим имя СписокОтобранныхДанных. На этом проектирование формы в конструкторе завершено и можно перейти к написанию программного кода. Разработанная форма будет базовой для ряда последующих процедур, выполняемых после щелчка на кнопке Выполнить. Базовой в том плане, что в различных примерах мы будем ее использовать. При этом будем менять текст программной процедуры, а иногда и элементы управления, расположенные на ней.

Функционирование созданной формы обработки связано с тем, что с помощью кнопки Выполнить поле списка с именем Список отобранных данных должно быть заполнено информацией о фамилиях сотрудников, организациях, где они работают, и их телефонах. Процедура, которую следует для этого разработать, приведена в листинге 2.1, а результат выполнения обработки в режиме 1С:Предприятие показан на рис. 2.6. Струк-

тура самой процедуры достаточно типична — сначала выполняется запрос к определенной таблице, а затем сведениями, полученными в результате выполнения запроса, заполняется элемент управления в форме.

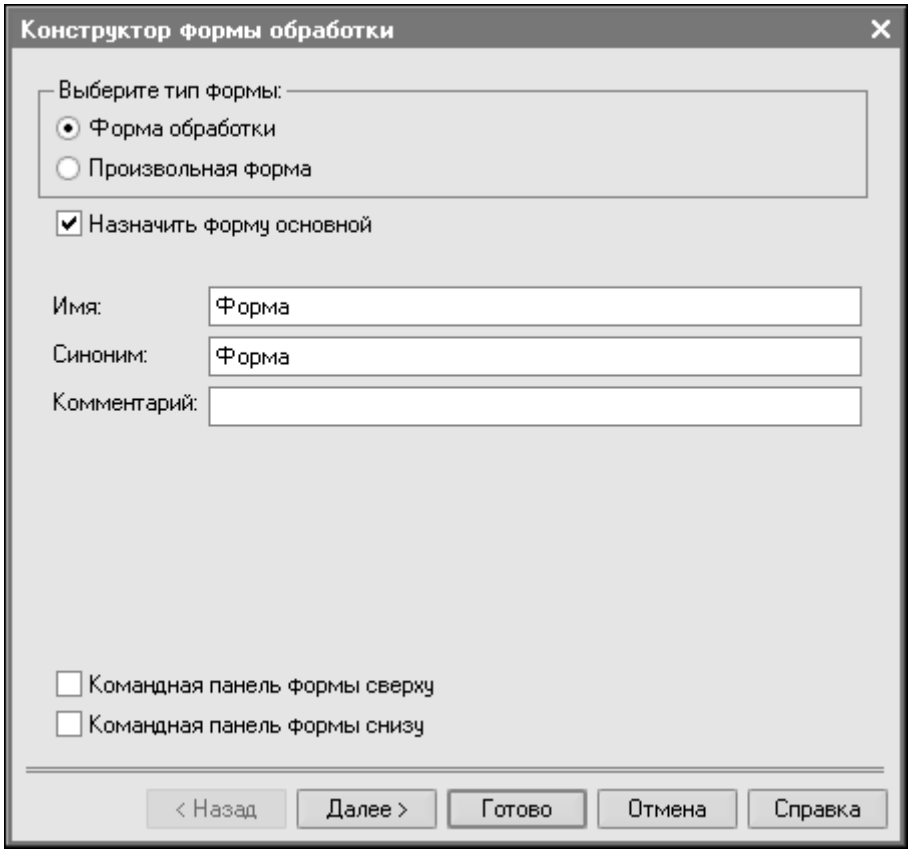

Рис. 2.2. Окно конструктора формы обработки ИзвлечениеИнформации

Листинг 2.1. Процедура обработки щелчка на кнопке Выполнить формы

```
Процедура КнопкаВыполнитьНажатие(Кнопка)
Запрос = Новый Запрос;
Запрос.Текст = "ВЫБРАТЬ Сотрудник,Телефон,
    | Ссылка.Наименование КАК Организация
    | ИЗ Справочник.Фирмы.КонтактныеЛица
    | УПОРЯДОЧИТЬ ПО Сотрудник";
Результат = Запрос.Выполнить().Выбрать();
СписокОтобранныхДанных.Очистить();
Пока Результат.Следующий() > 0 Цикл
   СписокОтобранныхДанных.Добавить(Строка(Результат.Сотрудник)+
   " ( фирма - " + Строка(Результат.Организация)+
   ")    Тел. "+Строка(Результат.Телефон));
   КонецЦикла;
КонецПроцедуры
```

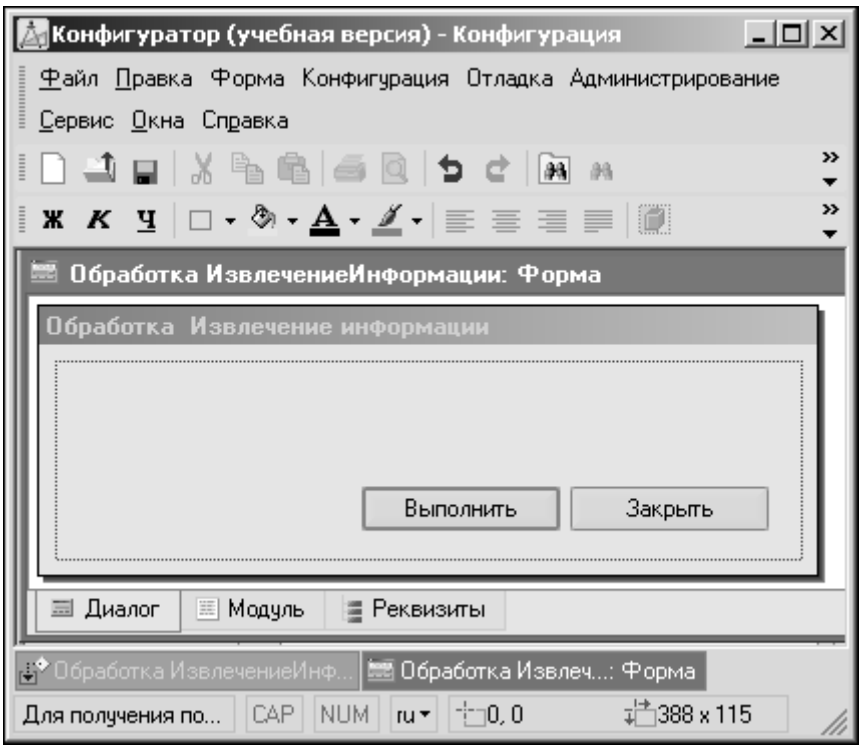

Рис. 2.3. Форма обработки ИзвлечениеИнформации *в режиме конфигуратора*

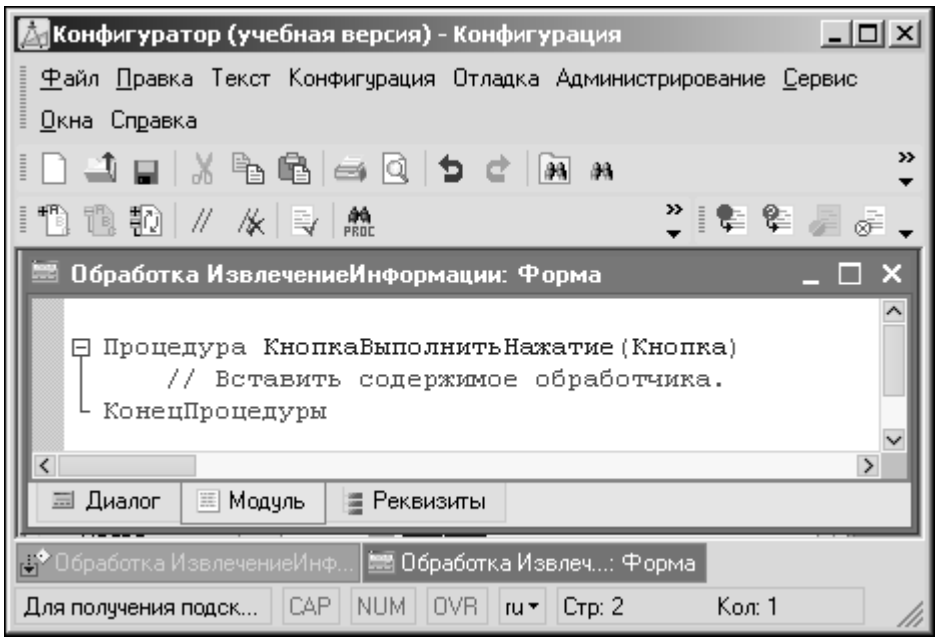

Рис. 2.4. Заготовка процедуры обработки щелчка на кнопке Выполнить

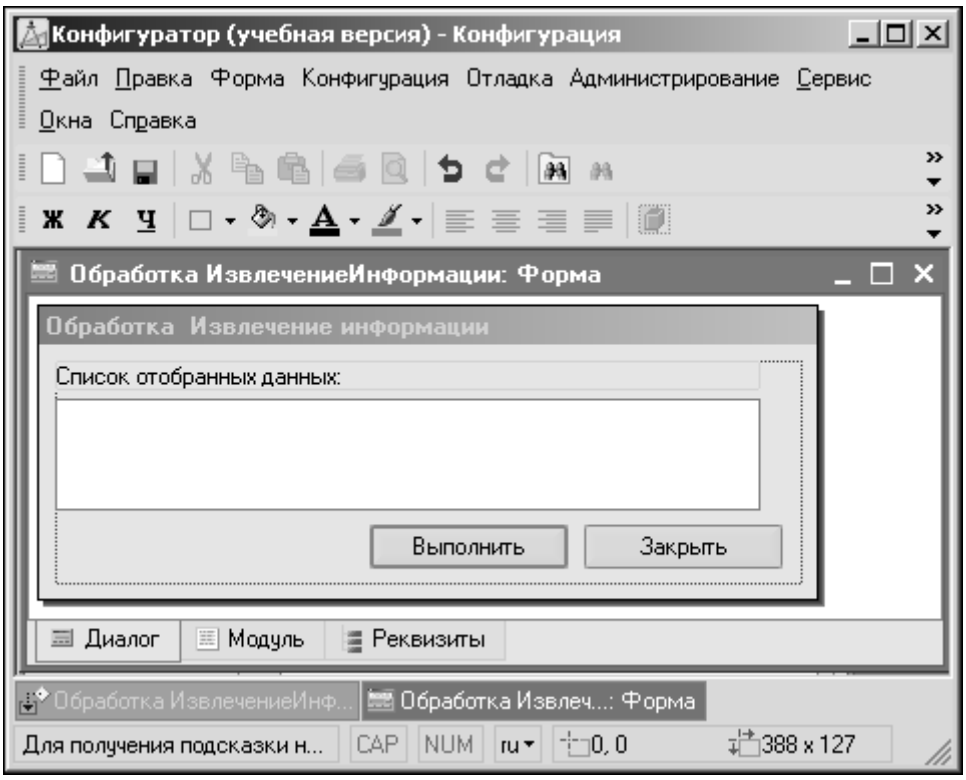

Рис. 2.5. Размещение поля списка в форме

В первой строке рассматриваемой процедуры создается новый объект типа `Запрос`, а далее с помощью его свойства `Текст` непосредственно формируется текст запроса. С этим текстом мы уже знакомились выше в данной главе. Как часто получается при составлении большинства запросов, их текст занимает более одной строки. В этом случае для продолжения строки используется символ "|". В следующей после текста запроса программной конструкции

```
Результат = Запрос.Выполнить().Выбрать()
```

сначала используется метод `Выполнить()`, что приводит к выполнению запроса. Далее другой метод `Выбрать()` позволяет произвести выборку информации из выполненного запроса. Фактически этот метод подготавливает работу другого далее используемого метода — `Следующий()`, который обеспечивает выборку очередной строки из результата выполнения запроса. Это позволяет программно последовательно перебрать строки в полученной выборке.

Далее в процедуре заполняется (извлеченной с помощью запроса информацией) элемент `СписокОтобранныхДанных`. Для этого данный элемент управления предварительно очищается с помощью метода `Очистить()`, а затем организуется цикл перебора строк, полученных в результате выполнения запроса. В этом цикле метод `Добавить()` позволяет пополнить список в форме информацией об очередном сотруднике.

`Строка()` — функция преобразования значений. Она осуществляет преобразование входного параметра в значение типа `строка`. Входной параметр может иметь произвольный тип данных.

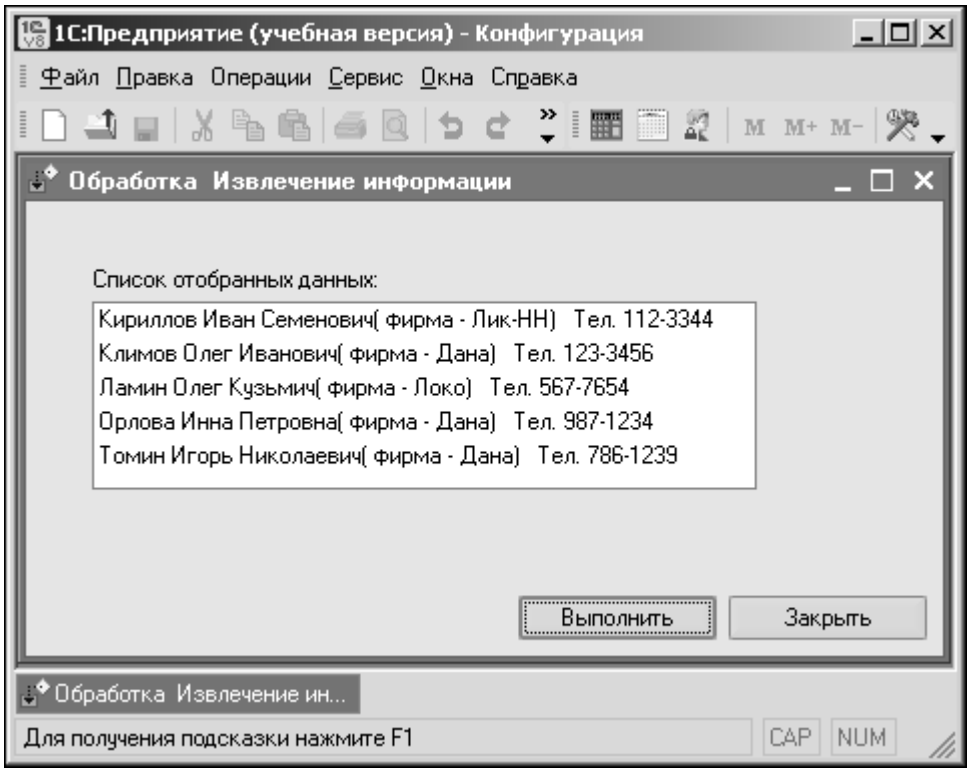

Рис. 2.6. Результат выполнения запроса по сотрудникам

Упорядочение в запросе

Изменим процедуру, представленную в листинге 2.1, с целью расположения отобранных элементов в порядке убывания номеров телефонов. Для этого она должна выглядеть в соответствии с листингом 2.2. Ключевое слово `УБЫВ` в конструкции `УПОРЯДОЧИТЬ` говорит о том, что информация должна быть отсортирована в порядке убывания значений указанного поля (в данном случае — номеров телефонов).

Листинг 2.2. Процедура вывода списка сотрудников с упорядочением по номерам телефонов

```
Процедура КнопкаВыполнитьНажатие(Кнопка)
Запрос = Новый Запрос;
Запрос.Текст = "ВЫБРАТЬ Сотрудник,Телефон,
    | Ссылка.Наименование КАК Организация
    | ИЗ Справочник.Фирмы.КонтактныеЛица
    | УПОРЯДОЧИТЬ ПО Телефон УБЫВ";
```

```
Результат = Запрос.Выполнить().Выбрать();
СписокОтобранныхДанных.Очистить();
Пока Результат.Следующий() > 0 Цикл
   СписокОтобранныхДанных.Добавить("Тел. "+Строка(Результат.Телефон) +
   "  Сотрудник -"+Строка(Результат.Сотрудник)+ " ( фирма - " +
   Строка(Результат.Организация)+")");
   КонецЦикла;
КонецПроцедуры
```

На рис. 2.7 показан результат выполнения данного запроса (в результате наибольшие значения номеров телефонов оказались вверху списка).

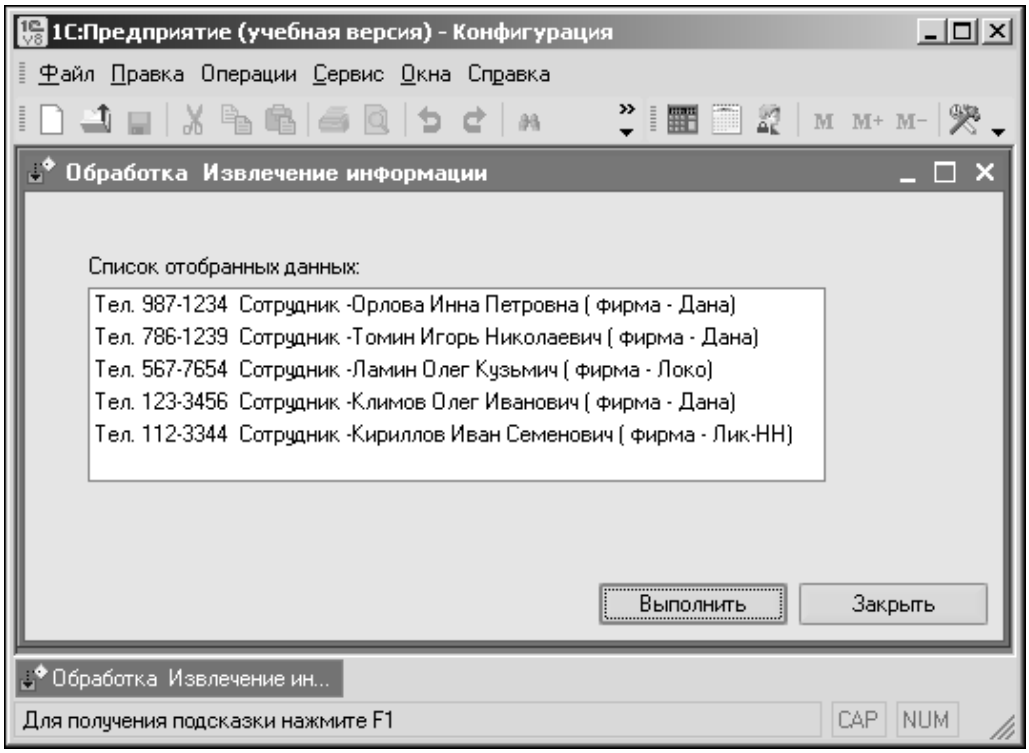

Рис. 2.7. Результат выполнения запроса с упорядочением по номерам телефонов

В случае необходимости применения упорядочения в подобном или каком-то другом запросе по возрастанию следует применить конструкцию с использованием ключевого слова `ВОЗР`. Например, для рассмотренной ситуации это может выглядеть следующим образом:

```
УПОРЯДОЧИТЬ ПО Телефон ВОЗР
```

> В предложении `УПОРЯДОЧИТЬ ПО` через запятую перечисляются условия, в соответствие 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 пример еще одного запроса на данную тему, который будет касаться регистра накопления КоличествоТоваров. Будем считать, что нам необходимо получить информацию о движении товаров по нашим филиалам в следующем формате:

- товар;
- филиал;
- регистратор;
- количество.

В регистре накопления реквизит под названием `регистратор` представляет ссылку на документ, которому подчинена рассматриваемая запись регистра. Информацию, извлеченную запросом, необходимо упорядочить сначала по филиалам, а уже внутри филиалов по количеству товаров. На рис. 2.8 показан заполненный табличный документ, который нам предстоит создать.

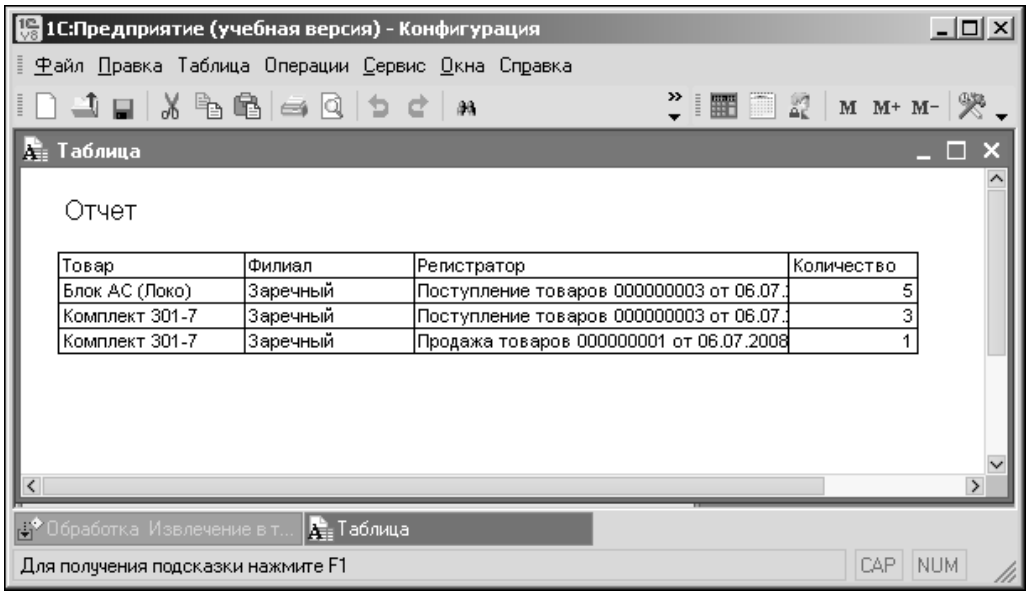

Рис. 2.8. Табличный документ с отчетом по товарам

Для создания представленного табличного документа лучше создать новую обработку, которая будет обеспечивать необходимую функциональность. Технические действия аналогичны ранее рассмотренным примерам, поэтому далее укажем только основную информацию. Так, на рис. 2.9 представлен макет табличного документа, который необходимо разработать. В приводимой далее процедуре учитывается, что имя отчета должно быть Отчет.

Необходимая процедура создания обработки, которую следует расположить на вкладке Модуль, приведена в листинге 2.3. Процедура выполняется после щелчка на кнопке, размещенной в форме обработки (аналогично примерам главы 1). Ее взаимодействие с макетом и позволяет создать необходимый табличный документ (см. рис. 2.8).

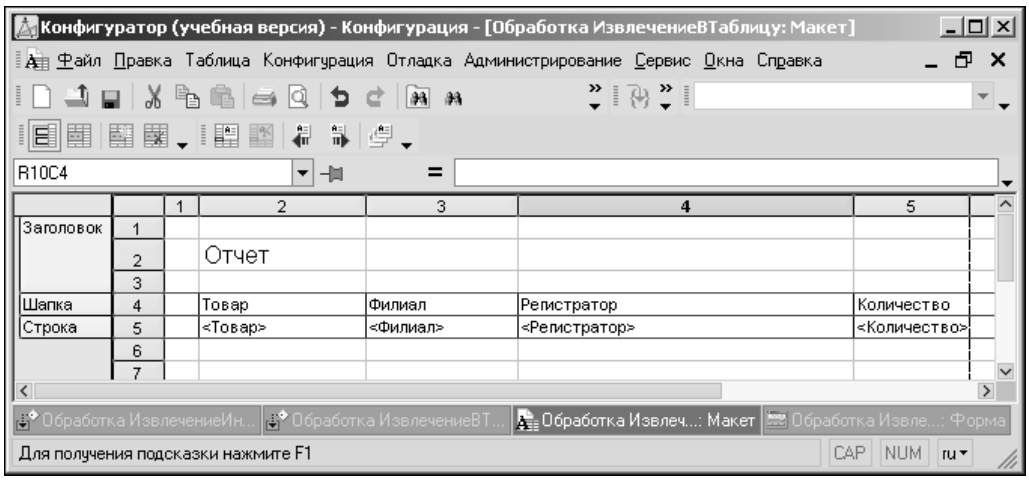

Рис. 2.9. Макет табличного документа для представления результатов обработки

Листинг 2.3. Процедура формирования табличного документа по информации из регистра накопления

```
Процедура КнопкаВыполнитьНажатие(Кнопка)
Запрос = Новый Запрос;
Запрос.Текст =
   "ВЫБРАТЬ
       | КоличествоТоваров.Товар КАК Товар,
       | КоличествоТоваров.Филиал КАК Филиал,
       | КоличествоТоваров.Регистратор КАК Регистратор,
       | КоличествоТоваров.Количество КАК Количество
   |ИЗ
       | РегистрНакопления.КоличествоТоваров КАК КоличествоТоваров
       |УПОРЯДОЧИТЬ ПО
       | Филиал,
       | Количество УБЫВ";
Результат = Запрос.Выполнить(). Выбрать();
ТабДок = Новый ТабличныйДокумент;
Макет = Обработки.ИзвлечениеВТаблицу.ПолучитьМакет("Макет");
// Заголовок
Область = Макет.ПолучитьОбласть("Заголовок");
ТабДок.Вывести(Область);
// Шапка
Область = Макет.ПолучитьОбласть("Шапка");
ТабДок.Вывести(Область);
// Перечень
Пока Результат.Следующий() Цикл
   Область = Макет.ПолучитьОбласть("Строка");
   Область.Параметры.Товар = Результат.Товар;
   Область.Параметры.Филиал = Результат.Филиал;
   Область.Параметры.Регистратор = Результат.Регистратор;
   Область.Параметры.Количество = Результат.Количество;
   ТабДок.Вывести(Область);
   КонецЦикла;
ТабДок.ОтображатьСетку = Ложь;
```

```
ТабДок.Защита = Ложь;
ТабДок.ТолькоПросмотр = Ложь;
ТабДок.ОтображатьЗаголовки = Ложь;
ТабДок.Показать();
КонецПроцедуры
```

Отбор максимальных (минимальных) значений

При большом объеме информации, получаемом в результате выполнения запроса, на практике часто требуется просмотреть лишь несколько наиб*о*льших значений или, наоборот, наименьших, например, списки самых больших поступлений товаров (по сумме либо по количеству). В этом случае в конструкцию ВЫБРАТЬ добавляется ключевое слово ПЕРВЫЕ. Если нас интересуют позиции в документах Поступление товаров (строки табличной части) с тремя самыми маленькими суммами, то для этого можно использовать процедуру, приведенную в листинге 2.4. Здесь мы опять вернемся к обработке ИзвлечениеИнформации, которая позволяет заполнить поле списка. Наша задача здесь заключается в замене предыдущего текста процедуры на новый. На рис. 2.10 отражен результат работы обработки в режиме 1С:Предприятие, где показана информация с учетом имеющихся данных в информационной базе.

Листинг 2.4. Процедура отбора трех минимальных поступлений по сумме

```
Процедура КнопкаВыполнитьНажатие(Кнопка)
Запрос = Новый Запрос;
Запрос.Текст = "ВЫБРАТЬ ПЕРВЫЕ 3 Товар,Сумма,
    | Ссылка.Фирма КАК ФирмаПоставщик
    | ИЗ Документ.ПоступлениеТоваров.ПереченьТоваров
    | УПОРЯДОЧИТЬ ПО Сумма ВОЗР";
Результат = Запрос.Выполнить().Выбрать();
СписокОтобранныхДанных.Очистить();
Пока Результат.Следующий() > 0 Цикл
   СписокОтобранныхДанных.Добавить("ТОВАР - "+Строка(Результат.Товар) +
   "   Сумма - "+Строка(Результат.Сумма)+ "  ( фирма - " +
   Строка(Результат.ФирмаПоставщик)+" )");
   КонецЦикла;
КонецПроцедуры
```

Если бы нас интересовали, наоборот, три максимальных по сумме поступления, то в тексте листинга 2.4 необходимо было бы изменить только конструкцию, связанную с упорядочением полученных данных. В этом случае текст запроса будет выглядеть следующим образом:

```
Запрос.Текст = "ВЫБРАТЬ ПЕРВЫЕ 3 Товар,Сумма,
    | Ссылка.Фирма.Наименование КАК ФирмаПоставщик
    | ИЗ Документ.ПоступлениеТоваров.ПереченьТоваров
    | КАК Перечень
    | УПОРЯДОЧИТЬ ПО Сумма УБЫВ";
```

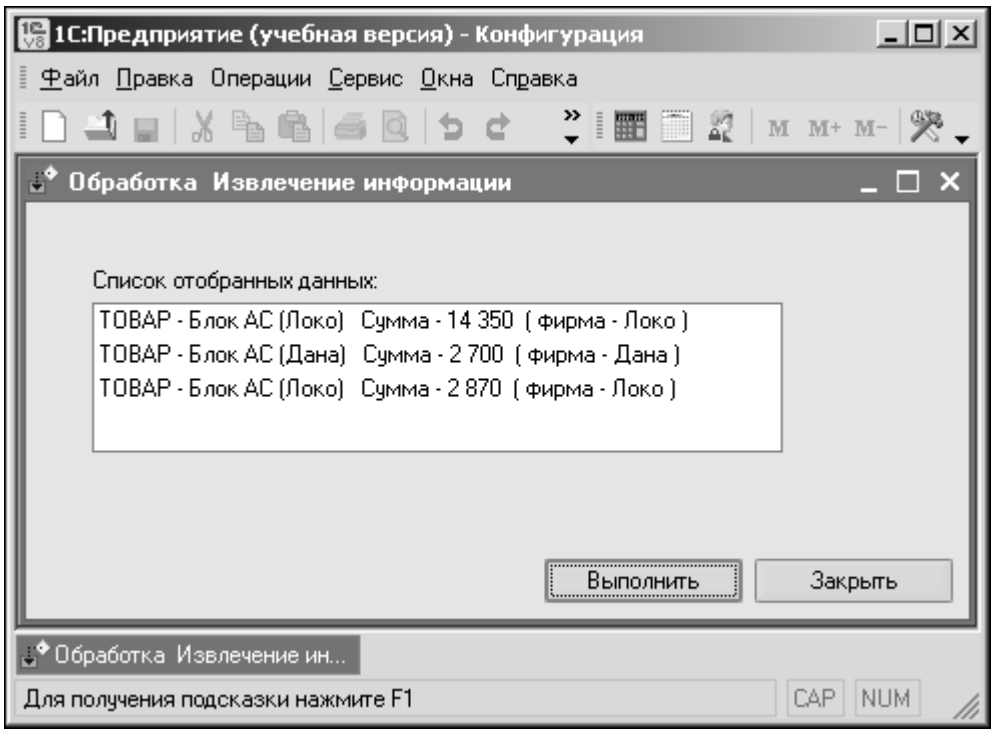

Рис. 2.10. Результат отбора поступлений с минимальными суммами

Исключение повторяющейся информации

Допустим, наша задача заключается в формировании перечня товаров, которые поступали от каждого поставщика по документам Поступление товаров, а именно: в этот список каждое конкретное название товара с указанием фирмы-поставщика должно попасть не более одного раза (даже если данный товар от указанного поставщика поступал несколько раз). Можно сказать, что в отчет должны попасть только уникальные комбинации "товар-фирма".

В языке запросов существует ключевое слово `РАЗЛИЧНЫЕ`, позволяющее отобрать только отличающиеся строки (в выборке результата выполнения запроса повторений не будет).

Возвратимся к нашему примеру и будем выбирать из документов Поступление товаров товары вместе с названиями фирм без повторяющихся комбинаций. Текст процедуры, в которой реализуется подобный отбор товаров, приводится в листинге 2.5, а на рис. 2.11 показан результат ее работы в режиме работы 1С:Предприятие. Здесь мы также вносим изменение в обработку ИзвлечениеИнформации (как и ранее, мы просто заменили предыдущую процедуру новой).

Листинг 2.5. Процедура отбора уникальных комбинаций названия товара и названия фирмы-поставщика для поступлений товаров

```
Процедура КнопкаВыполнитьНажатие(Кнопка)
Запрос = Новый Запрос;
Запрос.Текст = "ВЫБРАТЬ РАЗЛИЧНЫЕ Товар,
    | Ссылка.Фирма.Наименование КАК ФирмаПоставщик
    | ИЗ Документ.ПоступлениеТоваров.ПереченьТоваров КАК Перечень
    | УПОРЯДОЧИТЬ ПО Товар ВОЗР";
Результат = Запрос.Выполнить().Выбрать();
СписокОтобранныхДанных.Очистить();
Пока Результат.Следующий() > 0 Цикл
   СписокОтобранныхДанных.Добавить("ТОВАР - "+Строка(Результат.Товар) +
   "  ФИРМА - "+Строка(Результат.ФирмаПоставщик));
   КонецЦикла;
КонецПроцедуры
```

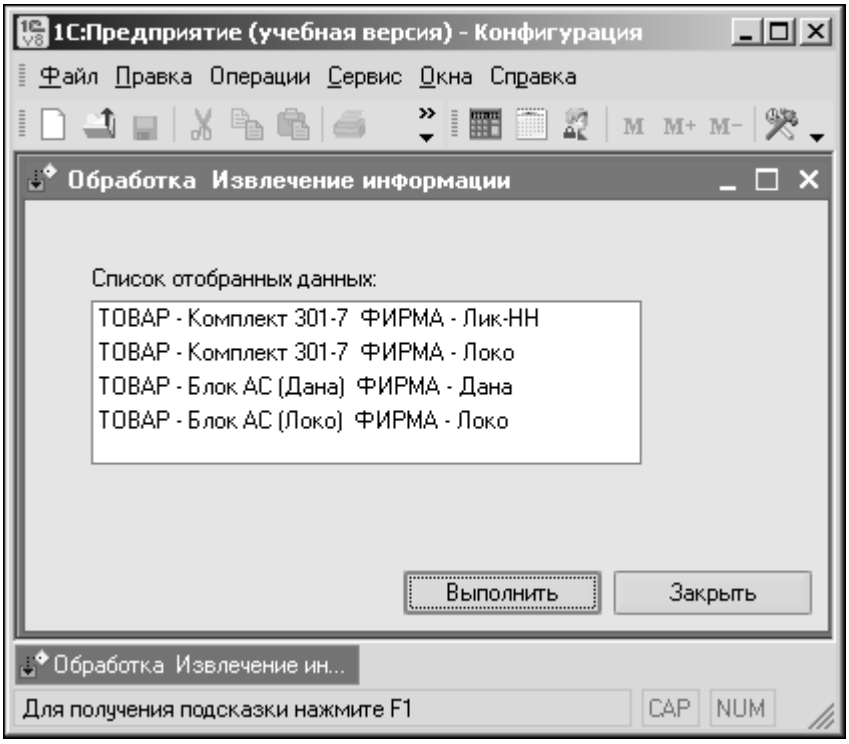

Рис. 2.11. Результат отбора товаров, которые поступали от конкретных фирм

Таким образом, в форме мы получим в списке не более одной строки с определенной комбинацией названия фирмы и названия товара.

> В начале книги уже говорилось о том, что примеры информационных баз, рассматриваемых в книге, можно скачать с сайта `http://www.vlunn.ru`. Там процедуры, относящиеся к этой главе, приводятся в различных обработках. Названия обработок соответствуют названию листингов по тексту книги.

Использование логических операторов в запросе

В предыдущих примерах нам уже встречалось ключевое слово языка запросов ГДЕ. Оно позволяет задать условие отбора данных из исходных таблиц-источников запроса. В результате в запросе оказываются отобранными только те записи, для которых выполняется заданное условие.

В конструкции ГДЕ разрешается использовать логические операторы (И, ИЛИ, НЕ), с помощью которых можно сформировать достаточно сложные логические выражения.

Применим эту возможность для получения списка поступлений товаров, которые фиксировались по каждому документу ПоступлениеТоваров в количестве, большем 5 единиц, при условии их суммарной стоимости (с учетом количества) более 5000 рублей. Реализация данного запроса, в котором происходит отбор при одновременном выполнении этих двух условий, продемонстрирована в процедуре, представленной в листинге 2.6.

Листинг 2.6. Процедура отбора поступлений товаров при выполнении условий по количеству и сумме поступления

```
Процедура КнопкаВыполнитьНажатие(Кнопка)
Запрос = Новый Запрос;
Запрос.Текст = "ВЫБРАТЬ Товар, Количество,Сумма,
    | Перечень.Ссылка.Филиал КАК Филиал
    | ИЗ Документ.ПоступлениеТоваров.ПереченьТоваров КАК Перечень
    | ГДЕ Количество > 5 И Сумма > 5000";
Результат = Запрос.Выполнить().Выбрать();
СписокОтобранныхДанных.Очистить();
Пока Результат.Следующий() > 0 Цикл
   СписокОтобранныхДанных.Добавить("ТОВАР - "+
   Строка(Результат.Товар.Наименование) +
   " ( Количество - "+Строка(Результат.Количество)+") Сумма:"+
   Строка(Результат.Сумма)+"  Филиал -"+ Строка(Результат.Филиал));
   КонецЦикла;
КонецПроцедуры
```

Результат внесенных изменений продемонстрирован на рис. 2.12. Как видите, в список отобранных данных попали только те товары, по которым выполняются указанные два условия.

Приведем теперь пример запроса, в котором комбинируются конструкции ИЛИ и И. Скажем, нас интересуют все товары, которые поступали или по цене не менее 3000 рублей в любом количестве или по цене более 2800 рублей в количестве большем двух. В листинге 2.7 приведена процедура, реализующая необходимую выборку информации при этих условиях, а на рис. 2.13 показан результат выполнения запроса в режиме 1С:Предприятие.

Листинг 2.7. Процедура отбора товаров при выполнении условий по количеству и сумме поступления

```
Процедура КнопкаВыполнитьНажатие(Кнопка)
Запрос = Новый Запрос;
Запрос.Текст = "ВЫБРАТЬ Товар,Количество,Цена,
    | Перечень.Ссылка.Дата КАК ДатаДокумента
```

```
    | ИЗ Документ.ПоступлениеТоваров.ПереченьТоваров КАК Перечень
    | ГДЕ ( Количество > 2 И Цена > 2800) ИЛИ Цена >= 3000";
Результат = Запрос.Выполнить().Выбрать();
СписокОтобранныхДанных.Очистить();
Пока Результат.Следующий() > 0 Цикл
   СписокОтобранныхДанных.Добавить("ТОВАР - "+
   Строка(Результат.Товар.Наименование) +
   " ( Количество - " + Строка(Результат.Количество)+ " Цена - " +
   Строка(Результат.Цена) +
   ") Дата документа:"+ Строка(Результат.ДатаДокумента));
   КонецЦикла;
КонецПроцедуры
```

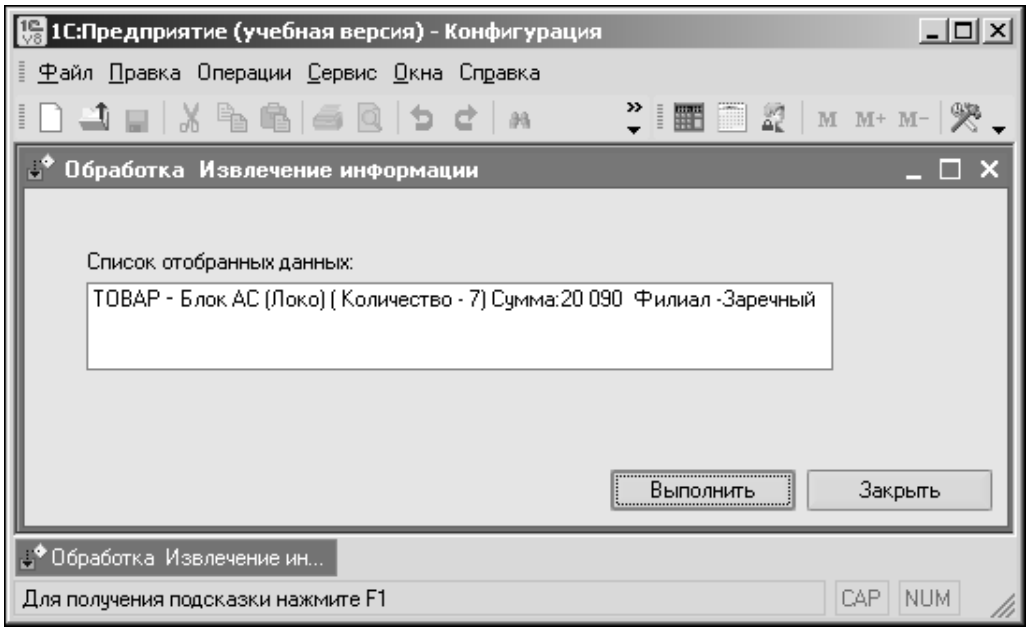

Рис. 2.12. Результат отбора поступлений товаров при одновременном выполнении двух условий

Агрегатные функции в запросах

К настоящему моменту мы на несложных примерах познакомились с технологией выборки информации из базы данных. Однако на практике часто требуется получить и разнообразную сводную информацию. В частности, по информации из документов Поступления товаров управленческому персоналу может оказаться полезно получить ответы на следующие вопросы.

- Какова сумма поступлений в разрезе товаров?
- Каково среднее количество поступлений каждого товара?
- Какова наибольшая сумма поступления по каждому товару?

Для реализации подобных запросов следует воспользоваться *агрегатными функциями* и *группировками*.

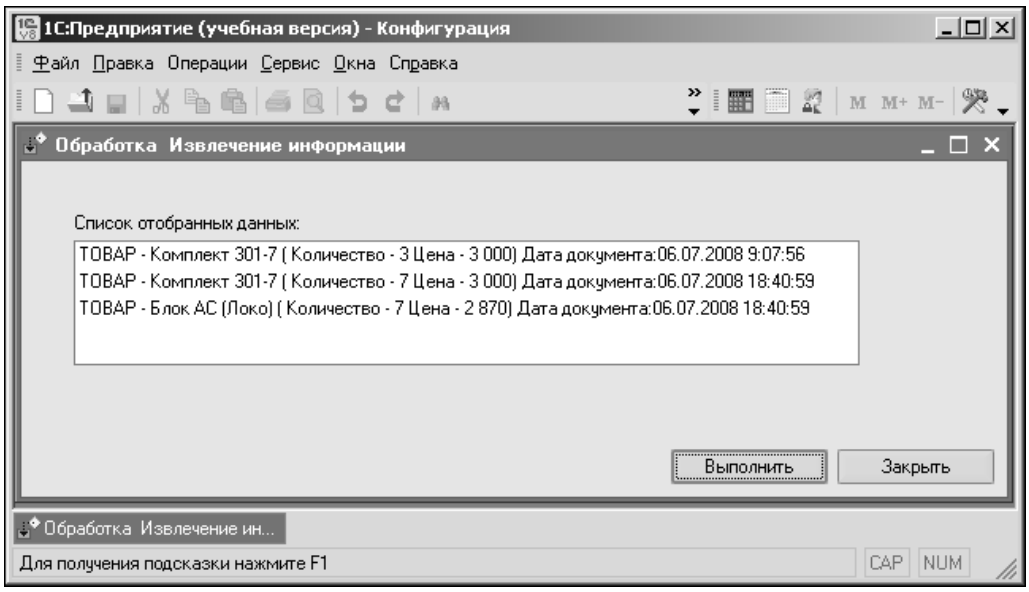

Рис. 2.13. Результат отбора поступлений товаров по комбинации двух условий

> 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. Агрегатные функции предназначены для обобщения всех значений указанного параметра. В табл. 2.1 перечислены агрегатные функции, имеющиеся в системе 1С:Предприятие 8.1.

Таблица 2.1. Агрегатные функции

Название	Комментарий
СУММА	Вычисляет сумму всех значений, содержащихся в указанном столбце. В качестве параметра функции можно указывать только поле, содержащее числовое значение
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

Что касается функции КОЛИЧЕСТВО, следует отметить три возможных способа ее применения:

- функция позволяет подсчитать количество значений указанного поля, не равных NULL;
- функция позволяет подсчитать количество *различных* значений указанного поля, не равных NULL (для этого необходимо указать ключевое слово РАЗЛИЧНЫЕ);

- функция позволяет подсчитать количество строк, полученных в результате выполнения запроса (для этого в качестве параметра функции надо указать символ “*”).

Перейдем к разработке программной процедуры, которая продемонстрирует использование агрегатных функций. При этом мы воспользуемся встречавшейся ранее в главе 1 конструкцией языка запросов — `СГРУППИРОВАТЬ ПО`. При ее использовании можно не просто выбирать записи из таблицы, а еще и сгруппировывать их определенным образом.

> Под словом *сгруппировывать* подразумевается компоновка информации по группировочным полям и вычисление агрегатных функций по каждой полученной группе.

Для выполнения запроса и отображения на экране полученной информации создадим новую обработку. Это связано с тем, что вывод информации, полученной в результате выполнения запроса, мы организуем в табличный документ. Поэтому первые действия будут связаны с изменением имеющегося объекта конфигурации — обработки ИзвлечениеВТаблицу. На вкладке Формы уже имеется простая форма (рис. 2.14) с кнопкой для формирования табличного документа. Здесь в качестве имени формы указано Форма, но у вас может быть выбрано и другое имя в соответствии с предыдущими проделанными действиями.

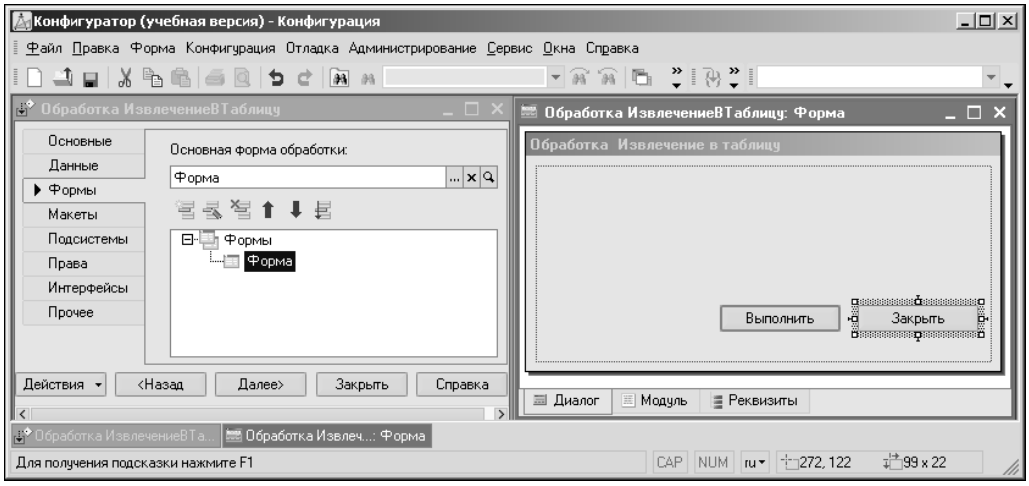

Рис. 2.14. Вкладка Формы и форма обработки ИзвлечениеВТаблицу

Следующий этап связан с формированием макета (рис. 2.15), который будет являться шаблоном табличного документа. В качестве имени макета укажем Отчет. Здесь мы видим традиционные три секции, характеризующие заголовочную и табличную части документа.

Процедура получения сводного отчета по поступившим товарам приведена в листинге 2.8. В процессе выполнения запроса производится группировка по товарам. На рис. 2.16 показано заполнение табличного документа на основе имеющихся в информационной базе сведений.

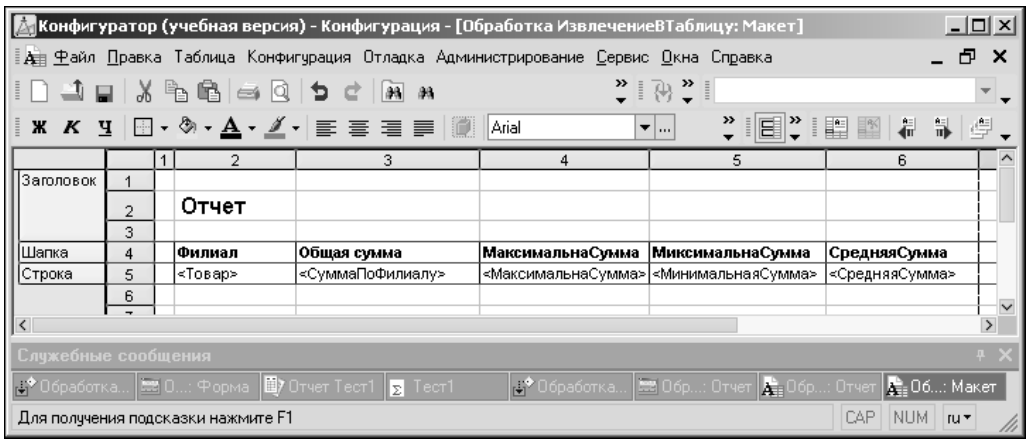

Рис. 2.15. Макет табличного документа

Листинг 2.8. Процедура формирования сводной информации

```
Процедура КнопкаВыполнитьНажатие(Кнопка)
Запрос = Новый Запрос;
Запрос.Текст = "ВЫБРАТЬ Товар,
    | СУММА(Сумма) КАК СуммаПоТовару,
    | МАКСИМУМ(Сумма) КАК МаксимальнаяСумма,
    | МИНИМУМ(Сумма) КАК МинимальнаяСумма,
    | СРЕДНЕЕ(Сумма) КАК СредняяСумма
    | ИЗ Документ.ПоступлениеТоваров.ПереченьТоваров
    | СГРУППИРОВАТЬ ПО Товар";
Результат = Запрос.Выполнить().Выбрать();
ТабДок = Новый ТабличныйДокумент;
Макет = Обработки.ИзвлечениеВТаблицу.ПолучитьМакет("Отчет");
// Заголовок
Область = Макет.ПолучитьОбласть("Заголовок");
ТабДок.Вывести(Область);
// Шапка
Область = Макет.ПолучитьОбласть("Шапка");
ТабДок.Вывести(Область);
// Перечень
Пока Результат.Следующий() Цикл
   Область = Макет.ПолучитьОбласть("Строка");
   Область.Параметры.Товар = Результат.Товар;
   Область.Параметры.ОбщаяСумма = Результат.СуммаПоТовару;
   Область.Параметры.МаксимальнаяСумма = Результат.МаксимальнаяСумма;
   Область.Параметры.МинимальнаяСумма = Результат.МинимальнаяСумма;
   Область.Параметры.СредняяСумма = Результат.СредняяСумма;
   ТабДок.Вывести(Область);
   КонецЦикла;
ТабДок.ОтображатьСетку = Ложь;
ТабДок.Защита = Ложь;
ТабДок.ТолькоПросмотр = Ложь;
ТабДок.ОтображатьЗаголовки = Ложь;
ТабДок.Показать();
КонецПроцедуры
```

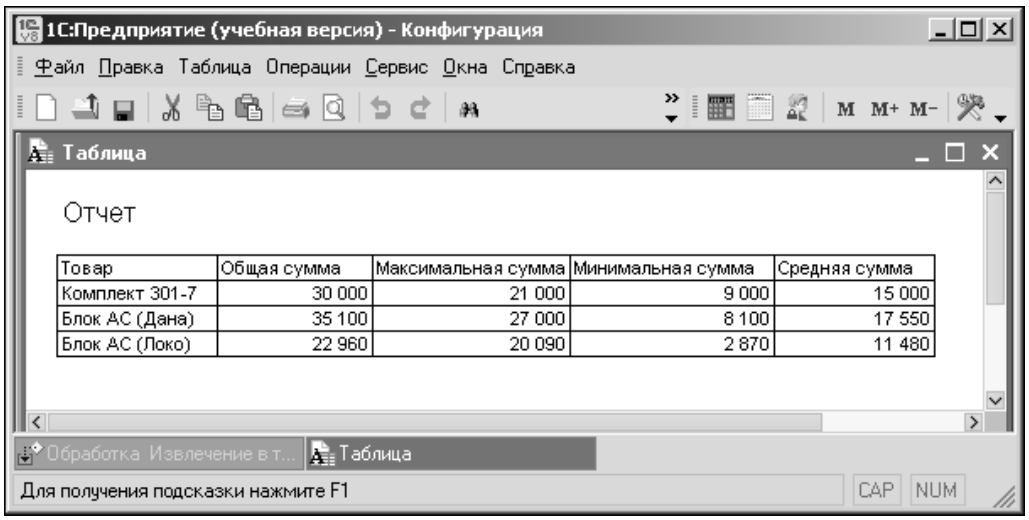

Рис. 2.16. Заполнение табличного документа сводной информацией

Для иллюстрации функции КОЛИЧЕСТВО приведем запрос, позволяющий подсчитать число всех документов Поступление товаров, а также число фирм, от которых были поступления товаров. Для реализации запроса вернемся к обработке ИзвлечениеИнформации. Здесь от нас потребуется подготовить новый вариант процедуры обработки щелчка на кнопке, расположенной в форме (листинг 2.9). Для вывода информации мы вновь воспользовались элементом *поле списка* (не будем изменять интерфейс обработки). В данном случае результатом запроса является одна строка с итоговой информацией (рис. 2.17).

Листинг 2.9. Процедура, демонстрирующая использование функции КОЛИЧЕСТВО

```
Процедура КнопкаВыполнитьНажатие(Кнопка)
Запрос = Новый Запрос;
Запрос.Текст = "ВЫБРАТЬ КОЛИЧЕСТВО(*) КАК Всего,
    | КОЛИЧЕСТВО (РАЗЛИЧНЫЕ Поступление.Фирма) КАК ЧислоФирм
    | ИЗ Документ.ПоступлениеТоваров КАК Поступление";
Результат = Запрос.Выполнить().Выбрать();
СписокОтобранныхДанных.Очистить();
Результат.Следующий();
СписокОтобранныхДанных.Добавить("Всего документов : "+
Строка(Результат.Всего) +
" Число фирм : "+Строка(Результат.ЧислоФирм));
КонецПроцедуры
```

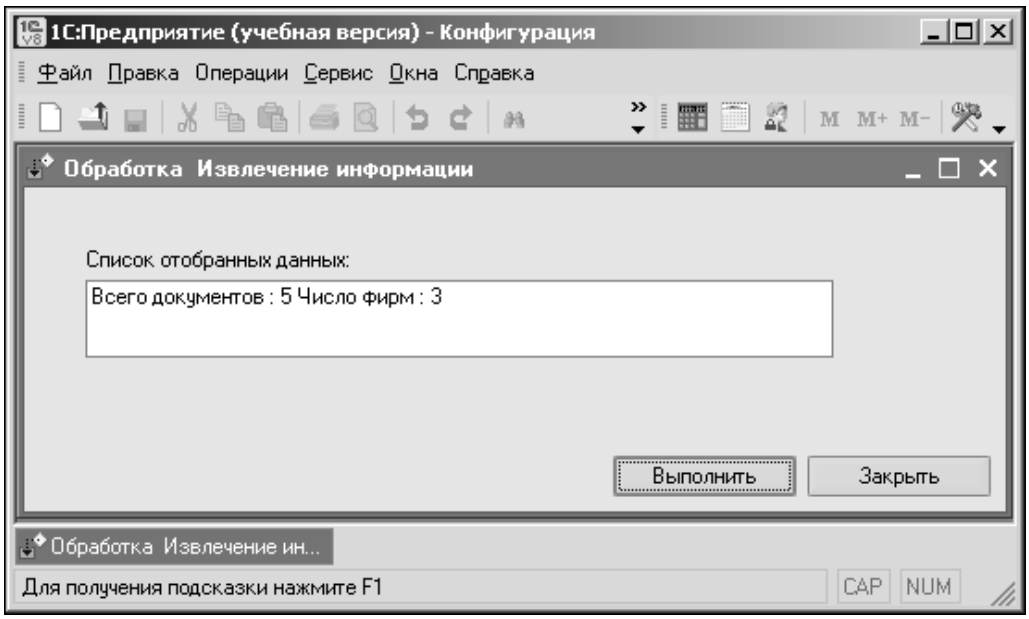

Рис. 2.17. Пример использования функции КОЛИЧЕСТВО

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 мы увидим, что параметры многочисленных условий отбора пользователь может указывать в диалоге с системой.

В главе 1 мы уже встречались с параметрическими запросами и теперь продолжим данную тему. Допустим, нам необходимо получить с помощью запроса информацию о документах по поступлениям товаров на определенный филиал не позднее указанной даты. Для получения и вывода рассматриваемой информации воспользуемся обработкой ИзвлечениеИнформации. Для начала необходимо ее дополнить двумя элементами управления (рис. 2.18). Первому дадим имя ГраницаДаты, а в качестве типа данных укажем `Дата`. Для другого элемента управления (имя — УказанныйФилиал) выберем тип данных `СправочникСсылка.Филиалы`. В результате пользователь непосредственно в форме сможет указать интересующие его в каждом конкретном случае филиал и граничную дату. На рис. 2.18 в качестве заголовка для поля `ГраницаДаты` указан более короткий вариант Дата.

Изменения в интерфейсе выполнены, осталось откорректировать процедуру обработки щелчка на кнопке. Ее новый вариант показан в листинге 2.10. В запросе мы использовали автоматически формируемые в любом документе реквизиты `Номер` и `Дата`. Результат внесения изменений в режиме 1С:Предприятие отражен на рис. 2.19.

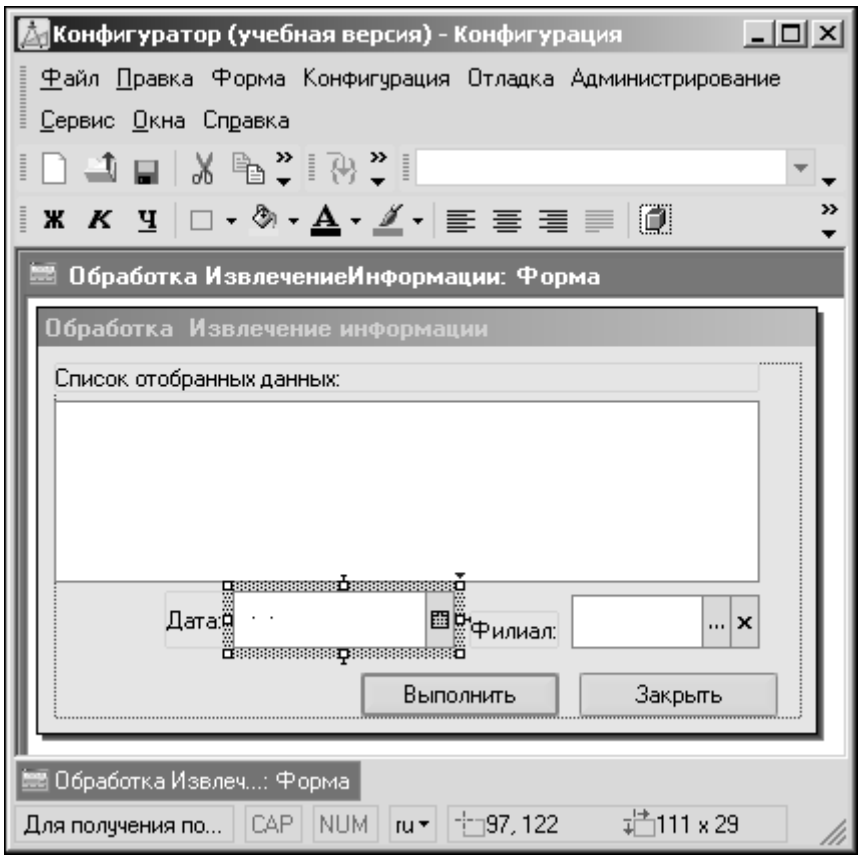

Рис. 2.18. Изменения в форме обработки ***ИзвлечениеИнформации***

Листинг 2.10. Процедура отбора товаров по филиалу и дате поступления

```
Процедура КнопкаВыполнитьНажатие(Кнопка)
Запрос = Новый Запрос;
Запрос.Текст = "ВЫБРАТЬ Дата, Номер
    | ИЗ Документ.ПоступлениеТоваров
    | ГДЕ Дата <= &ГраницаДаты И Филиал = &УказанныйФилиал";
Запрос.УстановитьПараметр("ГраницаДаты",
ЭлементыФормы.ГраницаДаты.Значение);
Запрос.УстановитьПараметр("УказанныйФилиал",
ЭлементыФормы.Филиал.Значение);
Результат = Запрос.Выполнить().Выбрать();
СписокОтобранныхДанных.Очистить();
Пока Результат.Следующий() > 0 Цикл
   СписокОтобранныхДанных.Добавить("Номер -"+Строка(Результат.Номер)+
   "   Дата - " + Строка(Результат.Дата));
   КонецЦикла;
КонецПроцедуры
```

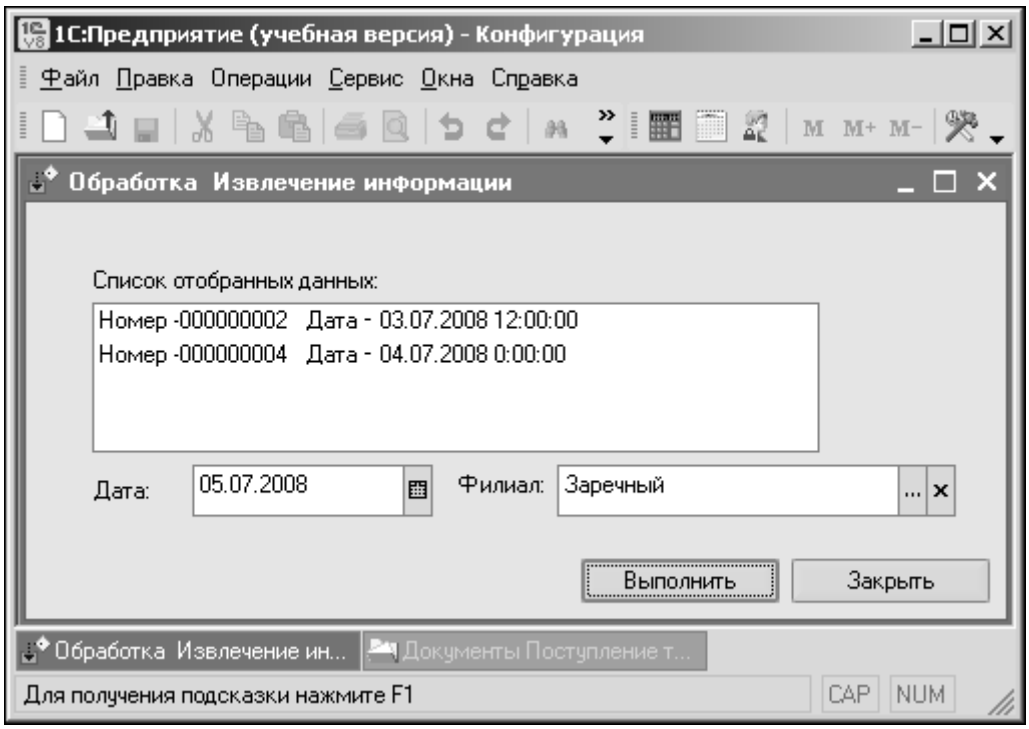

Рис. 2.19. Отбор информации по документам поступления товаров

Рассмотрим еще один пример на тему использования параметров в запросе. Допустим, что наша задача заключается в том, чтобы прочитать движения по регистру накопления КоличествоТоваров, используя ссылку на документ. Начало практических действий связано с внесением необходимых изменений в форму обработки (рис. 2.20). В форму добавлено поле ввода с именем ПолеВводаРегистратора. Тип данных для него укажем `ДокументСсылка.ПоступлениеТоваров`. Это позволит выбирать в поле ввода интересующий нас в каждом конкретном случае документ.

Следующий этап заключается в изменении процедуры обработки щелчка на кнопке — теперь она должна выглядеть так, как показано в листинге 2.11. В тексте запроса мы использовали реквизит регистра накопления `Регистратор`, который представляет ссылку на документ вызвавший движение по регистру. На рис. 2.21 показаны движения по одному из имеющихся в базе данных документов.

Листинг 2.11. Процедура отбора движений по регистру КоличествоТоваров

```
Процедура КнопкаВыполнитьНажатие(Кнопка)
Запрос = Новый Запрос;
Запрос.Текст = "ВЫБРАТЬ Товар,Филиал,Количество
    | ИЗ РегистрНакопления.КоличествоТоваров КАК КоличествоТоваров
   | ГДЕ КоличествоТоваров.Регистратор = &Регистратор";
Запрос.УстановитьПараметр("Регистратор",
ЭлементыФормы.ПолеВводаРегистратора.Значение);
Результат = Запрос.Выполнить().Выбрать();
СписокОтобранныхДанных.Очистить();
```

```
Пока Результат.Следующий() > 0 Цикл
    СписокОтобранныхДанных.Добавить("Товар -"+Строка(Результат.Товар)+
    "  Количество - " + Строка(Результат.Количество)+
    "  Филиал - " + Строка(Результат.Филиал));
    КонецЦикла;
КонецПроцедуры
```

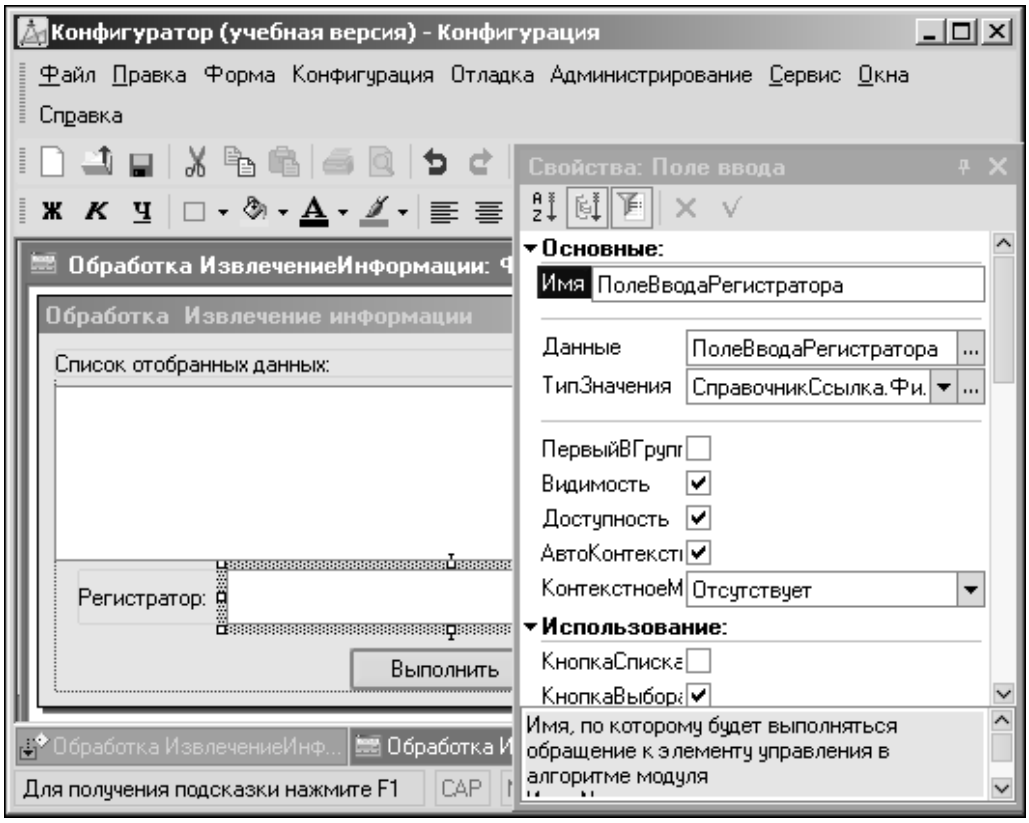

Рис. 2.20. Внесение изменений в форму обработки для отбора по регистру накопления

Рассмотрим еще один пример, связанный с документом ПоступлениеТоваров. В табличной части этого документа присутствует реквизит ЕдиницаИзмерения. Возможна ситуация, когда в каких-то документах это поле окажется незаполненным. Фактически это “брак” в документах, и наша задача его обнаружить. Если документов много, то визуальный способ будет малоэффективным. Оформим запрос в соответствии с листингом 2.12. Здесь мы выполняем отбор с условием по равенству значения в поле `ЕдиницаИзмерения` значению параметра `ПустоеПоле`. В запросе выполняется выборка неповторяющихся записей (для этого используется ключевое слово `РАЗЛИЧНЫЕ`). В этом случае при наличии в одном документе нескольких строк с не указанной единицей измерения в результат запроса документ попадет только один раз (рис. 2.22).

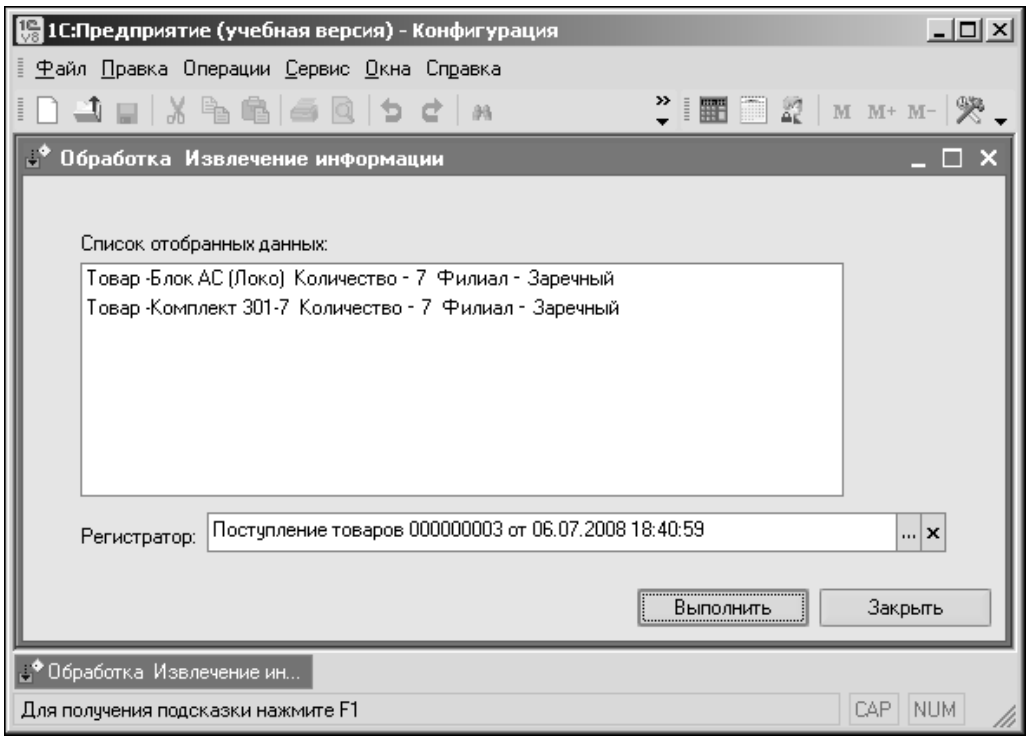

Рис. 2.21. Движения по регистру КоличествоТоваров *с учетом регистратора*

Листинг 2.12. Отбор документов с незаполненным полем единицы измерения

```
Процедура КнопкаВыполнитьНажатие(Кнопка)
Запрос = Новый Запрос;
Запрос.Текст = "ВЫБРАТЬ РАЗЛИЧНЫЕ ПереченьТоваров.Ссылка
    | КАК НезаполненныйДокумент
    | ИЗ Документ.ПоступлениеТоваров.ПереченьТоваров КАК ПереченьТоваров
    | ГДЕ ПереченьТоваров.ЕдиницаИзмерения = &ПустоеПоле";
Запрос.УстановитьПараметр("ПустоеПоле",
Справочники.ЕдиницыИзмерения.ПустаяСсылка());
Результат = Запрос.Выполнить().Выбрать();
СписокОтобранныхДанных.Очистить();
Пока Результат.Следующий() > 0 Цикл
   СписокОтобранныхДанных.Добавить("Документ-"+
      Строка(Результат.НезаполненныйДокумент));
   КонецЦикла;
КонецПроцедуры
```

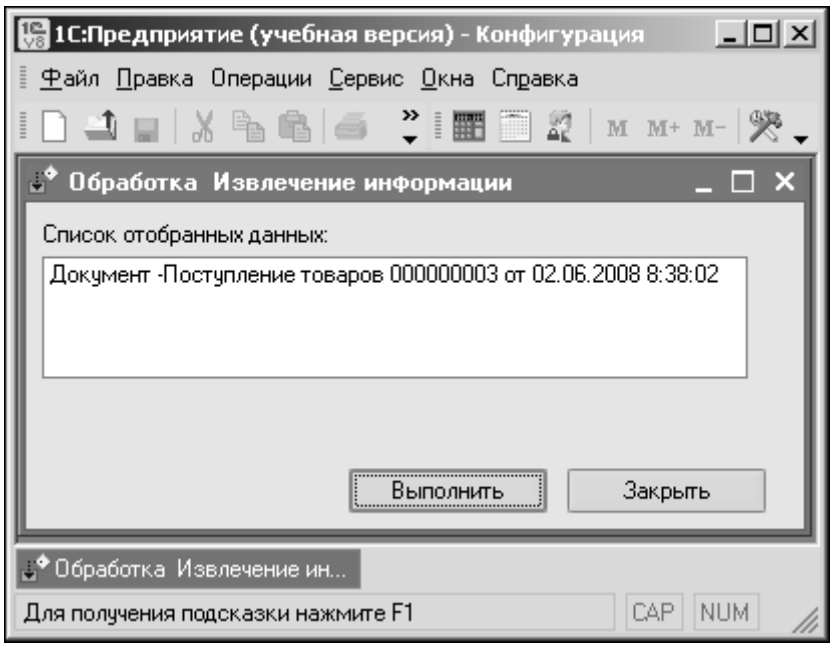

Рис. 2.22. Отбор документов с незаполненным полем единицы измерения

Ключевое слово МЕЖДУ

В языке запросов имеется вспомогательное ключевое слово МЕЖДУ, используемое для задания интервалов. Например, в запросе, приведенном в листинге 2.13, необходимо перед его выполнением установить значения параметров Дата1 и Дата2. В этом случае в результат запроса попадут только те записи регистра накопления, которые соответствуют выбранному интервалу дат. Эти значения вводятся пользователем в размещенные в форме поля ввода (рис. 2.23).

Заметим, что в конструкции рассматриваемого запроса используется параметр (измерение регистра накопления) Период, который отражает дату и время внесения информации.

Листинг 2.13. Процедура обработки щелчка на кнопке с использованием слова МЕЖДУ

```
Процедура КнопкаВыполнитьНажатие(Кнопка)
Запрос = Новый Запрос;
Запрос.Текст = "ВЫБРАТЬ Товар,
    | Филиал, Количество
    | ИЗ РегистрНакопления.КоличествоТоваров
    | ГДЕ РегистрНакопления.КоличествоТоваров.Период МЕЖДУ &Дата1 И
    | &Дата2";
Запрос.УстановитьПараметр("Дата1",
      ЭлементыФормы.Дата1.Значение);
Запрос.УстановитьПараметр("Дата2",
      ЭлементыФормы.Дата2.Значение);
Результат = Запрос.Выполнить().Выбрать();
```

```
СписокОтобранныхДанных.Очистить();
Пока Результат.Следующий() > 0 Цикл
   СписокОтобранныхДанных.Добавить(Строка(Результат.Товар)+
   "   "+Строка(Результат.Филиал)+
   "   "+Строка(Результат.Количество));
   КонецЦикла;
КонецПроцедуры
```

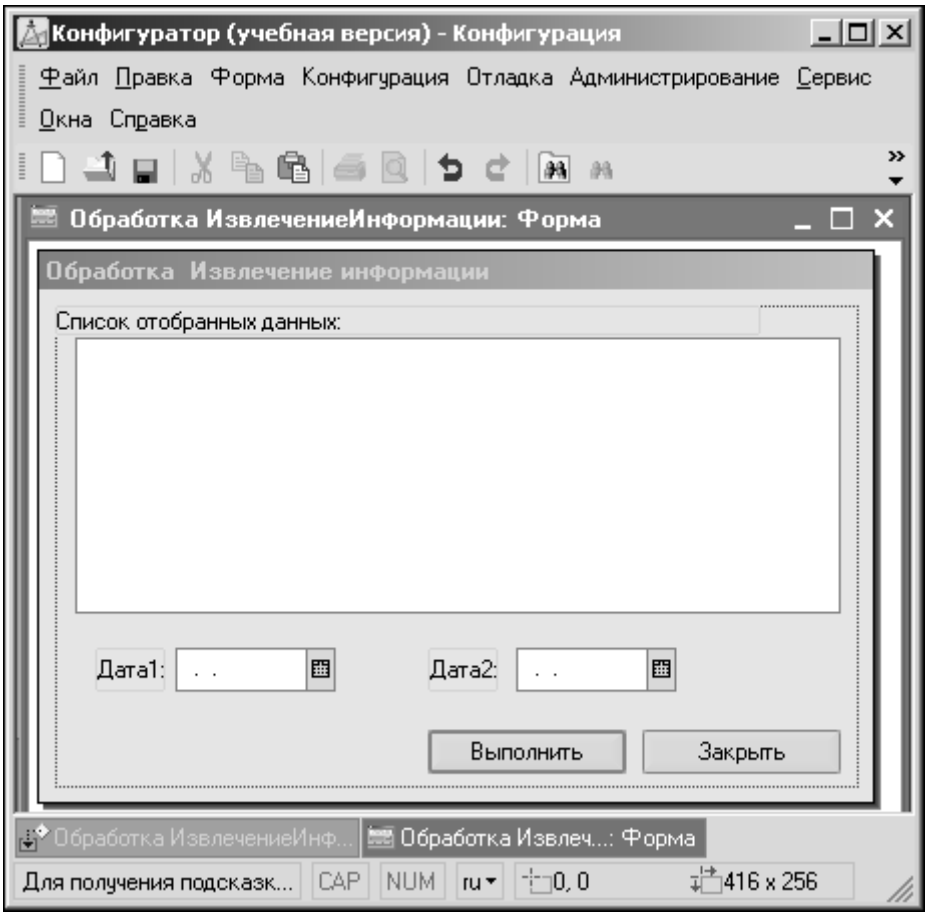

Рис. 2.23. Добавление новых элементов управления в форму

Результат выполненных изменений в режиме 1С:Предприятие продемонстрирован на рис. 2.24.

Рассмотрим теперь пример реализации подобного запроса к документам Поступление товаров (листинг 2.14). В этом случае происходит перебор документов, имеющихся в информационной базе. В запросе используется параметр `Дата`, характеризующий дату и время создания документа.

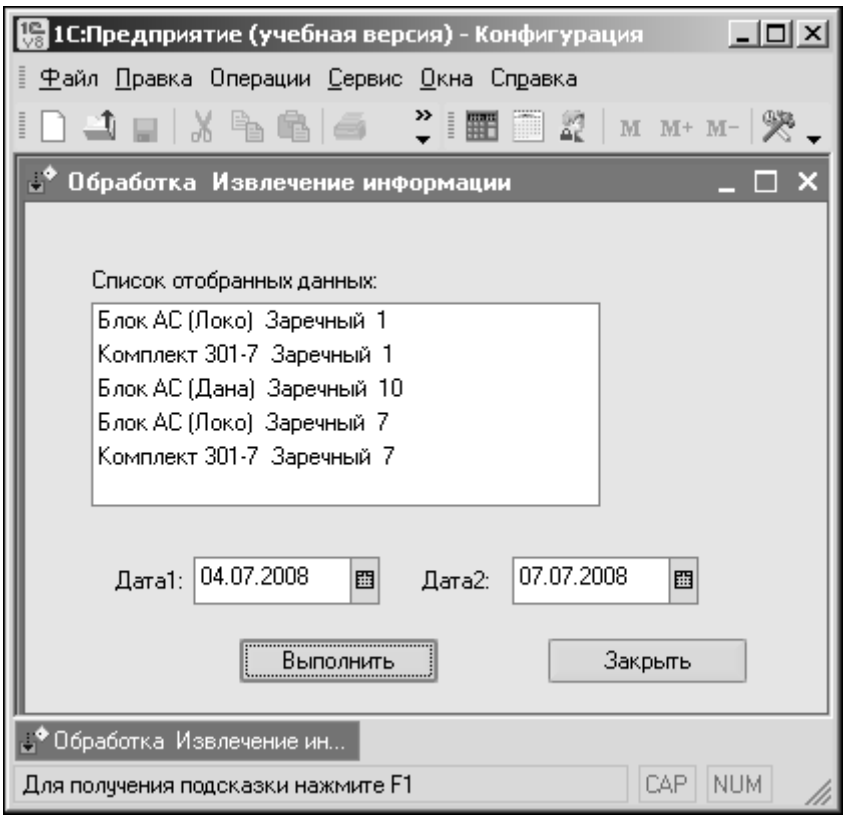

Рис. 2.24. Отбор информации из регистра накопления КоличествоТоваров

Листинг 2.14. Процедура обработки щелчка на кнопке с использованием слова МЕЖДУ и извлечением информации по документам

```
Процедура КнопкаВыполнитьНажатие(Кнопка)
Запрос = Новый Запрос;
Запрос.Текст = "ВЫБРАТЬ Товар, Количество,Сумма
    | ИЗ Документ.ПоступлениеТоваров.ПереченьТоваров
    | ГДЕ Ссылка.Дата МЕЖДУ &Дата1 И &Дата2";
Запрос.УстановитьПараметр("Дата1",
      ЭлементыФормы.Дата1.Значение);
Запрос.УстановитьПараметр("Дата2",
      ЭлементыФормы.Дата2.Значение);
Результат = Запрос.Выполнить().Выбрать();
СписокОтобранныхДанных.Очистить();
Пока Результат.Следующий() > 0 Цикл
   СписокОтобранныхДанных.Добавить("ТОВАР - "+
   Строка(Результат.Товар.Наименование) + " ( Количество - " +
   Строка(Результат.Количество)+ " Сумма - " + Стро-
ка(Результат.Сумма));
   КонецЦикла;
КонецПроцедуры
```

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.

В листинге 2.15 приведен пример процедуры с запросом, в котором используется конструкция ИМЕЮЩИЕ. Цель рассматриваемого запроса — отобрать товары, для которых итоговая сумма по поступлениям превышает 25000 руб. На рис. 2.25 показан результат выполнения отбора по имеющейся в базе информации.

Листинг 2.15. Процедура обработки щелчка на кнопке с использованием ключевого слова ИМЕЮЩИЕ

```
Процедура КнопкаВыполнитьНажатие(Кнопка)
Запрос = Новый Запрос;
Запрос.Текст = "ВЫБРАТЬ Товар, СУММА(Сумма) КАК СуммаТоваров
  | ИЗ Документ.ПоступлениеТоваров.ПереченьТоваров КАК Перечень
   | СГРУППИРОВАТЬ ПО Товар
   | ИМЕЮЩИЕ
   | СУММА(Сумма) > 25000";
Результат = Запрос.Выполнить().Выбрать();
СписокОтобранныхДанных.Очистить();
Пока Результат.Следующий() > 0 Цикл
   СписокОтобранныхДанных.Добавить(Строка(Результат.Товар)+
      "  "+Строка(Результат.СуммаТоваров));
   КонецЦикла;
КонецПроцедуры
```

Функция расчета итогов в запросах

Формирование итоговой информации является одной из наиболее востребованных функций любой информационной системы. Язык запросов системы 1С:Предприятие 8 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.

Предложение ИТОГИ позволяет определить, расчет каких итогов необходимо выполнить в запросе. При расчете итогов вычисляются значения агрегатных функций по выборкам с одинаковыми значениями полей — контрольных точек. Итоги добавляются в результат запроса как итоговые строки.

Описание итогов начинается с ключевого слова ИТОГИ. Другое ключевое слово ОБЩИЕ означает, что необходимо сформировать итоговую строку по всему результату запроса. Помимо общих итогов, можно задать расчет итогов по контрольным точкам. Для этого после обязательного ключевого слова ПО необходимо указать список контрольных точек. Каждая контрольная точка содержит выражение, вычисляемое при помощи собственного запроса. В результате необходимые значения этих выражений будут рассчитаны, а в результат основного запроса добавлены итоговые строки.

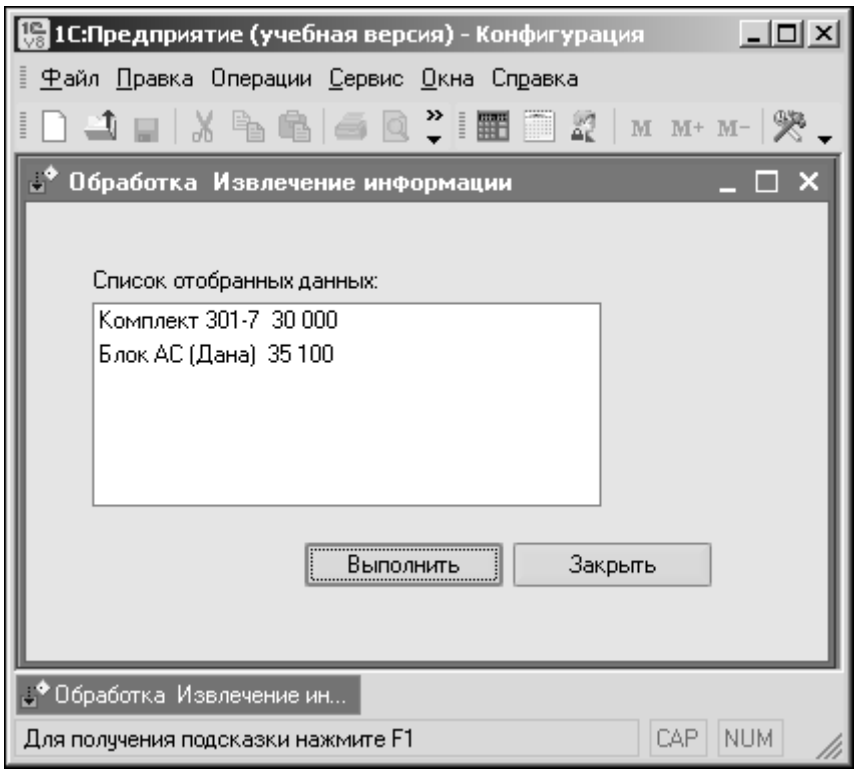

Рис. 2.25. Отбор информации по поступлениям с использованием ключевого слова ИМЕЮЩИЕ

В листинге 2.16 приведена процедура с использованием запроса, который позволяет получить итоги по поступлениям товаров на филиалы. Как видно из примера выполнения этого запроса в режиме 1С:Предприятие (рис. 2.26), в этом случае в результате появляются дополнительные строки с итоговыми суммами по филиалам.

Листинг 2.16. Процедура обработки щелчка на кнопке с формированием итогов по филиалам

```
Процедура КнопкаВыполнитьНажатие(Кнопка)
Запрос = Новый Запрос;
Запрос.Текст = "ВЫБРАТЬ Перечень.Ссылка.Филиал КАК Филиал,Товар,
    | Сумма
    | ИЗ Документ.ПоступлениеТоваров.ПереченьТоваров КАК Перечень
    | УПОРЯДОЧИТЬ ПО Филиал,Сумма
    | ИТОГИ СУММА(Сумма) ПО Филиал";
Результат = Запрос.Выполнить().Выбрать();
СписокОтобранныхДанных.Очистить();
Пока Результат.Следующий() > 0 Цикл
   СписокОтобранныхДанных.Добавить("Филиал - " + Строка(Результат.Филиал)+
   "      " + Строка(Результат.Товар)+
   "  Получено на сумму : " +Строка(Результат.Сумма)+ "  руб. ");
   КонецЦикла;
КонецПроцедуры
```

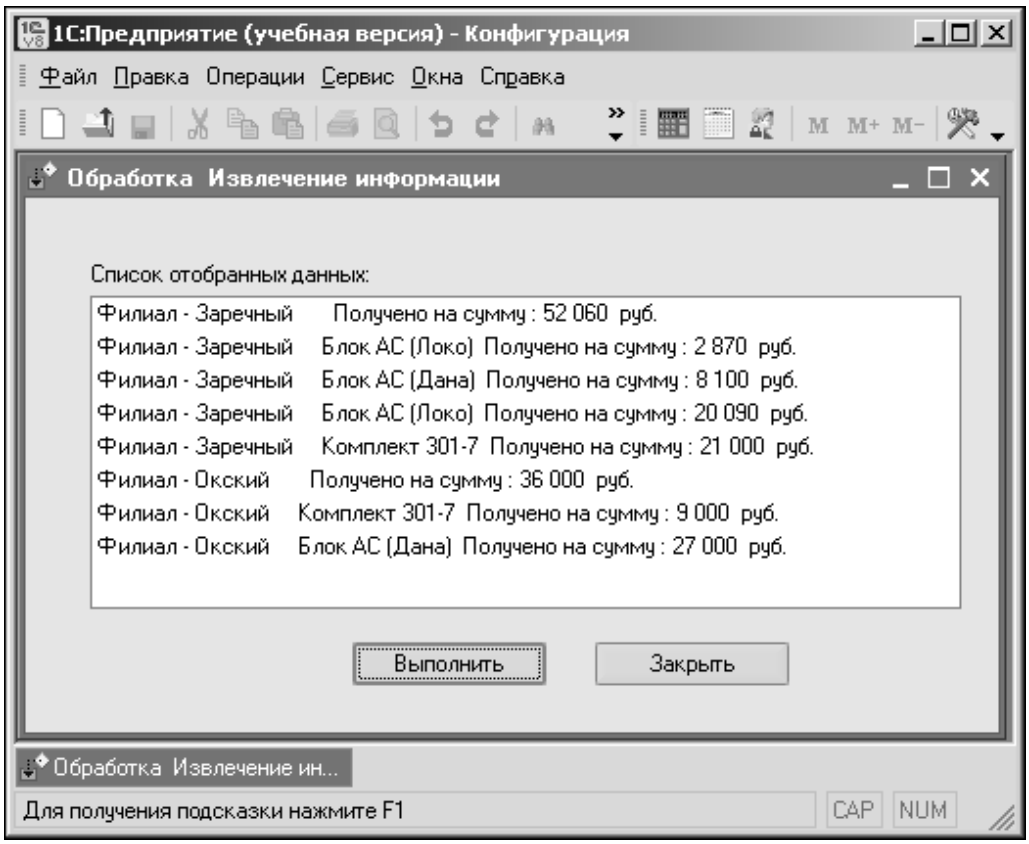

Рис. 2.26. Отбор информации с выводом итогов по филиалам

Рассмотрим теперь ситуацию, когда требуется вывести результаты данного запроса в табличный документ с соответствующим форматированием. У нас уже есть обработка ИзвлечениеВТаблицу, предназначенная для формирования табличного документа. В режиме конфигуратора перейдем к ее доработке. Во-первых, нам понадобится конкретный макет (рис. 2.27), в котором размещены четыре секции. Для имени макета выберем значение МакетСИтогами.

Необходимая процедура для формирования табличного документа показана в листинге 2.17, а результат выполнения в режиме 1С:Предприятие можно увидеть на рис. 2.28.

Листинг 2.17. Процедура обработки щелчка на кнопке с формированием итогов по филиалам

```
Процедура КнопкаВыполнитьНажатие(Кнопка)
Запрос = Новый Запрос;
Запрос.Текст = "ВЫБРАТЬ Перечень.Ссылка.Филиал КАК Филиал,Товар,
    | Сумма
    | ИЗ Документ.ПоступлениеТоваров.ПереченьТоваров КАК Перечень
    | УПОРЯДОЧИТЬ ПО Филиал,Сумма
    | ИТОГИ СУММА(Сумма) ПО Филиал";
Результат = Запрос.Выполнить().Выбрать();
```

```
ТабДок = Новый ТабличныйДокумент;
Макет = Обработки.ИзвлечениеВТаблицу.ПолучитьМакет("МакетСИтогами");
// Заголовок
Область = Макет.ПолучитьОбласть("Заголовок");
ТабДок.Вывести(Область);
// Шапка
Область = Макет.ПолучитьОбласть("Шапка");
ТабДок.Вывести(Область);
// Перечень
Пока Результат.Следующий() Цикл
   Если Результат.ТипЗаписи() =
         ТипЗаписиЗапроса.ИтогПоГруппировке Тогда
      Область = Макет.ПолучитьОбласть("Итоги");
      Область.Параметры.Филиал = Результат.Филиал;
      Область.Параметры.Сумма = Результат.Сумма;
      ТабДок.Вывести(Область);
   Иначе
      Область = Макет.ПолучитьОбласть("Строка");
      Область.Параметры.Филиал = Результат.Филиал;
      Область.Параметры.Товар = Результат.Товар;
      Область.Параметры.Сумма = Результат.Сумма;
      ТабДок.Вывести(Область);
      КонецЕсли;
   КонецЦикла;
ТабДок.ОтображатьСетку = Ложь;
ТабДок.Защита = Ложь;
ТабДок.ТолькоПросмотр = Ложь;
ТабДок.ОтображатьЗаголовки = Ложь;
ТабДок.Показать();
КонецПроцедуры
```

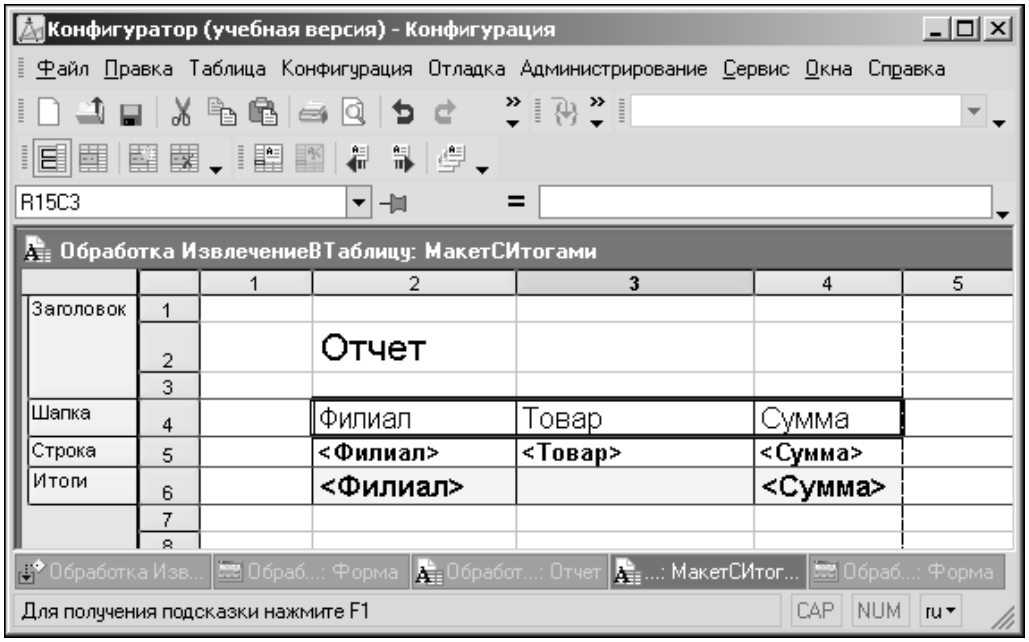

Рис. 2.27. Макет табличного документа

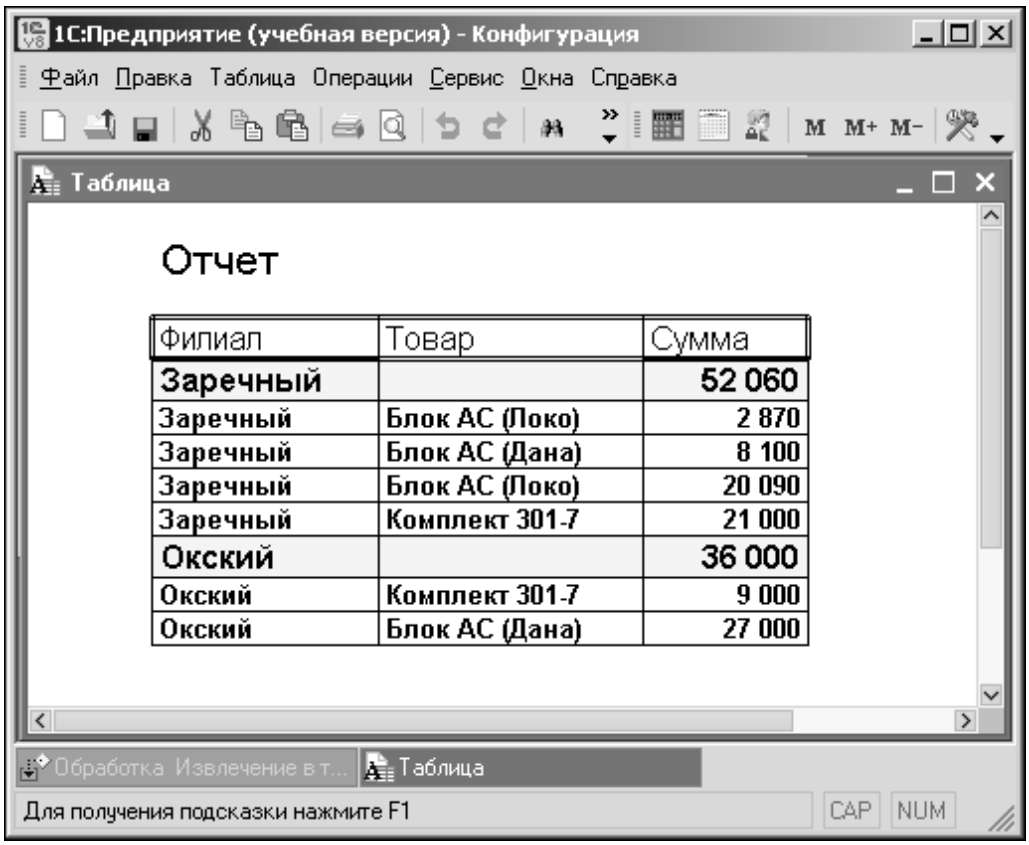

Рис. 2.28. Формирование табличного документа с итогами по филиалам

Объединение 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 упорядочение 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.

Объединение запросов начинается с обязательного ключевого слова `ОБЪЕДИНИТЬ`, после которого следует описание присоединяемого запроса. Далее может присоединяться еще один запрос и т.д.

По умолчанию при объединении запросов полностью одинаковые строки в результате запроса (когда они формируются разными запросами) заменяются одной. Если требует-

ся, чтобы были оставлены все сформированные строки, необходимо указать ключевое слово ВСЕ.

Рассмотрим ситуацию, когда требуется получить список городов, по которым имело место движение товаров. Проще говоря, необходимо составить список городов, фирмы которых отражены в документах ПоступлениеТоваров и ПродажаТоваров. Для вывода результатов используем уже знакомую обработку ИзвлечениеИнформации. Необходимая процедура, которая выполняется после щелчка на кнопке Выполнить, приведена в листинге 2.18. В результате выполнения (рис. 2.29) в поле списка окажутся города, из которых были поставки товаров либо в которые были продажи из наших филиалов.

Листинг 2.18. Процедура обработки щелчка на кнопке с использованием объединения запросов

```
Процедура КнопкаВыполнитьНажатие(Кнопка)
Запрос = Новый Запрос;
Запрос.Текст = "ВЫБРАТЬ Фирма.Город КАК Город
    | ИЗ Документ.ПоступлениеТоваров
    | ОБЪЕДИНИТЬ
    | ВЫБРАТЬ Фирма.Город
    | ИЗ Документ.ПродажаТоваров
    | УПОРЯДОЧИТЬ ПО Город";
Результат = Запрос.Выполнить().Выбрать();
СписокОтобранныхДанных.Очистить();
Пока Результат.Следующий() > 0 Цикл
   СписокОтобранныхДанных.Добавить(Строка(Результат.Город));
   КонецЦикла;
КонецПроцедуры
```

Теперь немного скорректируем задачу — укажем дополнительное условие для анализируемых документов: они должны соответствовать заданному интервалу дат. Сам интервал указывается с помощью элементов управления, размещенных в форме обработки (рис. 2.30).

Теперь осталось откорректировать две процедуры: выполняющую запрос и заполняющую табличный документ. В листинге 2.19 приведен программный код, который требуется создать для решения поставленной задачи. Здесь мы используем параметр документа Дата. Результат изменений в запросе и интерфейсе формы приведен на рис. 2.31.

Листинг 2.19. Процедура обработки щелчка на кнопке с формированием итогов

```
Процедура КнопкаВыполнитьНажатие(Кнопка)
Запрос = Новый Запрос;
Запрос.Текст = "ВЫБРАТЬ Фирма.Город КАК Город
    | ИЗ Документ.ПоступлениеТоваров
    | ГДЕ Дата МЕЖДУ &Дата1 И &Дата2
    | ОБЪЕДИНИТЬ
    | ВЫБРАТЬ Фирма.Город
    | ИЗ Документ.ПродажаТоваров
    | ГДЕ Дата МЕЖДУ &Дата1 И &Дата2
    | УПОРЯДОЧИТЬ ПО Город";
Запрос.УстановитьПараметр("Дата1",
      ЭлементыФормы.Дата1.Значение);
Запрос.УстановитьПараметр("Дата2",
      ЭлементыФормы.Дата2.Значение);
Результат = Запрос.Выполнить().Выбрать();
```

```
СписокОтобранныхДанных.Очистить();
Пока Результат.Следующий() > 0 Цикл
    СписокОтобранныхДанных.Добавить(Строка(Результат.Город));
    КонецЦикла;
КонецПроцедуры
```

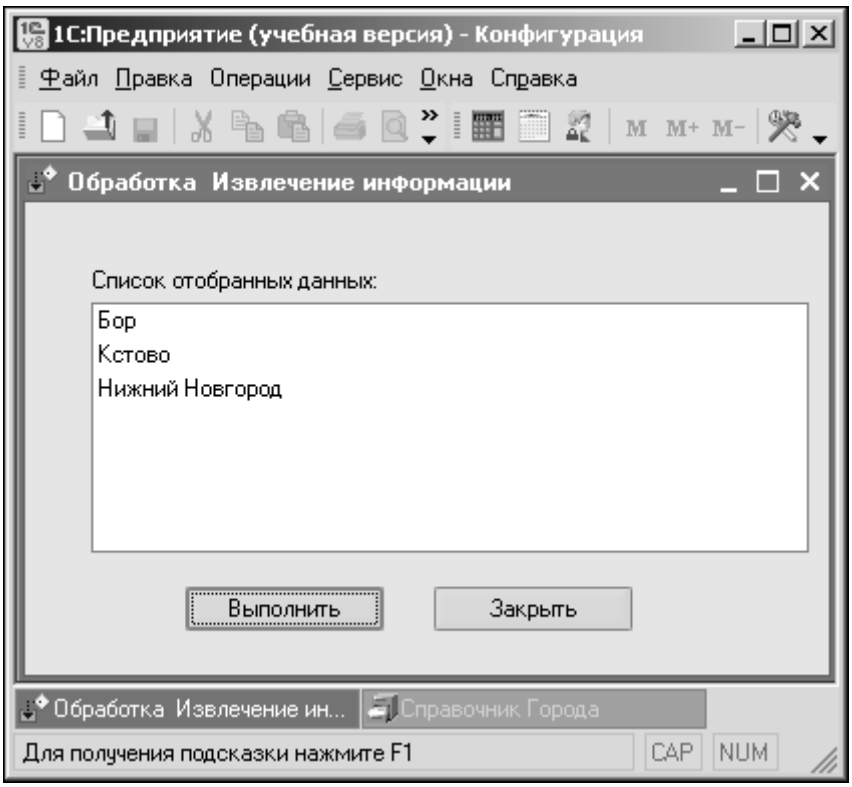

Рис. 2.29. Пример объединения запросов

Приведем еще один пример на тему объединения запросов. Допустим необходимо составить список товаров в привязке к фирме, которая выступала в роли либо поставщика, либо продавца. В листинге 2.20 приведена процедура, реализующая данный запрос. В результате в режиме 1С:Предприятие мы получим набор уникальных комбинаций: список товаров с указанием фирмы (рис. 2.32).

Листинг 2.20. Процедура формирования списка товаров с указанием фирмы

```
Процедура КнопкаВыполнитьНажатие(Кнопка)
Запрос = Новый Запрос;
Запрос.Текст = "ВЫБРАТЬ Перечень.Ссылка.Фирма КАК Фирма,Товар
    | ИЗ Документ.ПоступлениеТоваров.ПереченьТоваров КАК Перечень
    | ОБЪЕДИНИТЬ
    | ВЫБРАТЬ Перечень.Ссылка.Фирма КАК Фирма,Товар
    | ИЗ Документ.ПродажаТоваров.ПереченьТоваров КАК Перечень";
Результат = Запрос.Выполнить().Выбрать();
СписокОтобранныхДанных.Очистить();
Пока Результат.Следующий() > 0 Цикл
```

```
    СписокОтобранныхДанных.Добавить(Строка(Результат.Фирма)+ "     "+
    Строка(Результат.Товар) );
    КонецЦикла
КонецПроцедуры
```

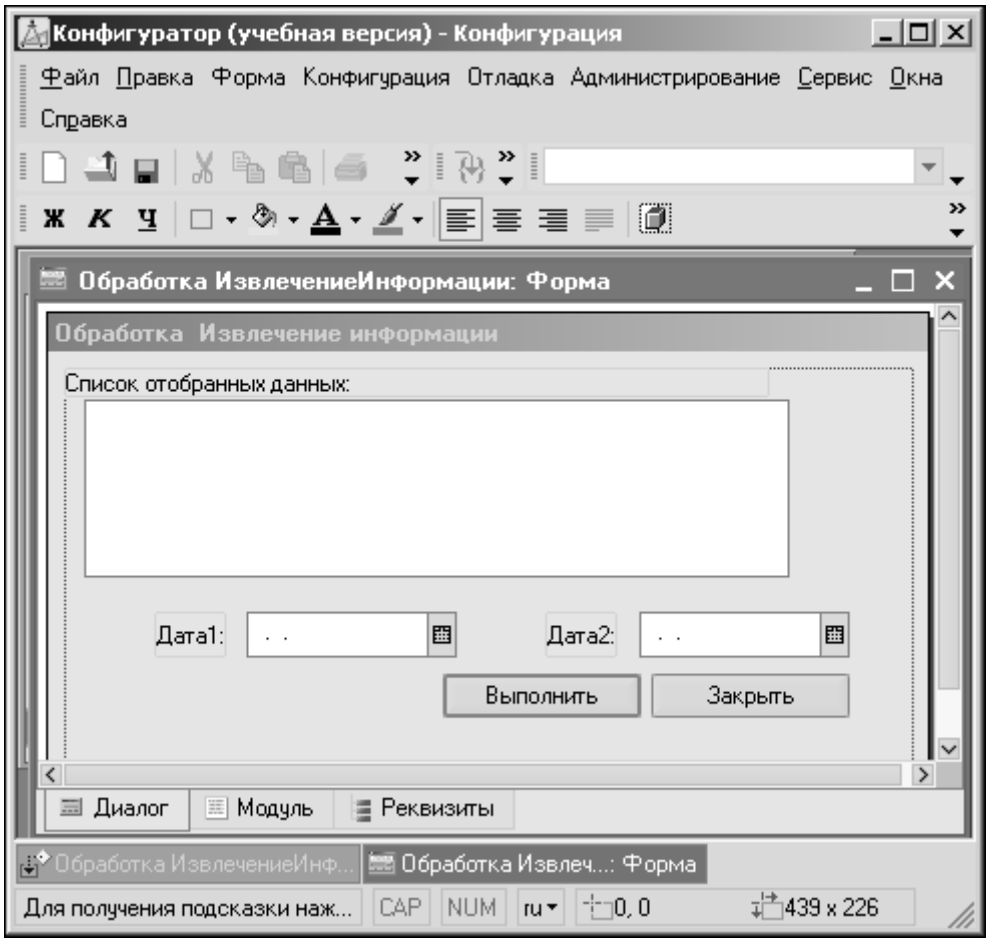

Рис. 2.30. Включение элементов управления для указания параметров запроса

Соединение таблиц при организации запроса

До настоящего времени практически во всех приводимых запросах выборка происходила из одной таблицы. В предыдущем разделе мы рассматривали соединение таблиц, однако этим ресурсы языка запросов не ограничиваются.

Важной особенностью языка запросов системы 1С:Предприятие является возможность обращения сразу к нескольким таблицам и соединения их определенным образом. Например, при работе может потребоваться выбрать названия всех товаров с отображением их количества. которые одновременно присутствовали и в документах поступления, и в документах продажи.

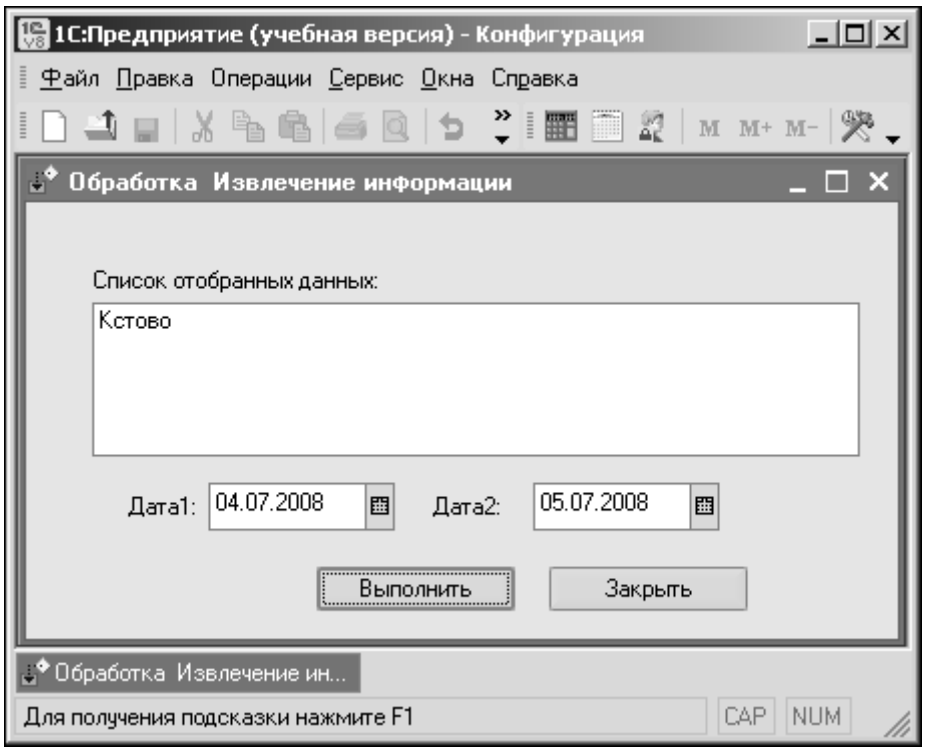

Рис. 2.31. Включение даты в качестве параметра отбора документов

Внутреннее соединение

В конструкции подобного запроса (листинг 2.21) используется ключевое слово СОЕДИНЕНИЕ, позволяющее отобрать из документов ПоступлениеТоваров и ПродажаТоваров только совпадающие товары. На рис. 2.33 приведен результат выполнения подобного запроса для имеющихся в базе данных.

Листинг 2.21. Первая процедура, демонстрирующая использование внутреннего соединения

```
Процедура КнопкаВыполнитьНажатие(Кнопка)
Запрос = Новый Запрос;
Запрос.Текст = "ВЫБРАТЬ РАЗЛИЧНЫЕ Перечень1.Товар КАК Товар1,
    | Перечень2.Товар
    | ИЗ Документ.ПоступлениеТоваров.ПереченьТоваров КАК Перечень1
    | СОЕДИНЕНИЕ
    | Документ.ПродажаТоваров.ПереченьТоваров КАК Перечень2
    | ПО Перечень1.Товар = Перечень2.Товар";
Результат = Запрос.Выполнить().Выбрать();
СписокОтобранныхДанных.Очистить();
Пока Результат.Следующий() > 0 Цикл
   СписокОтобранныхДанных.Добавить(Строка(Результат.Товар1));
   КонецЦикла;
КонецПроцедуры
```

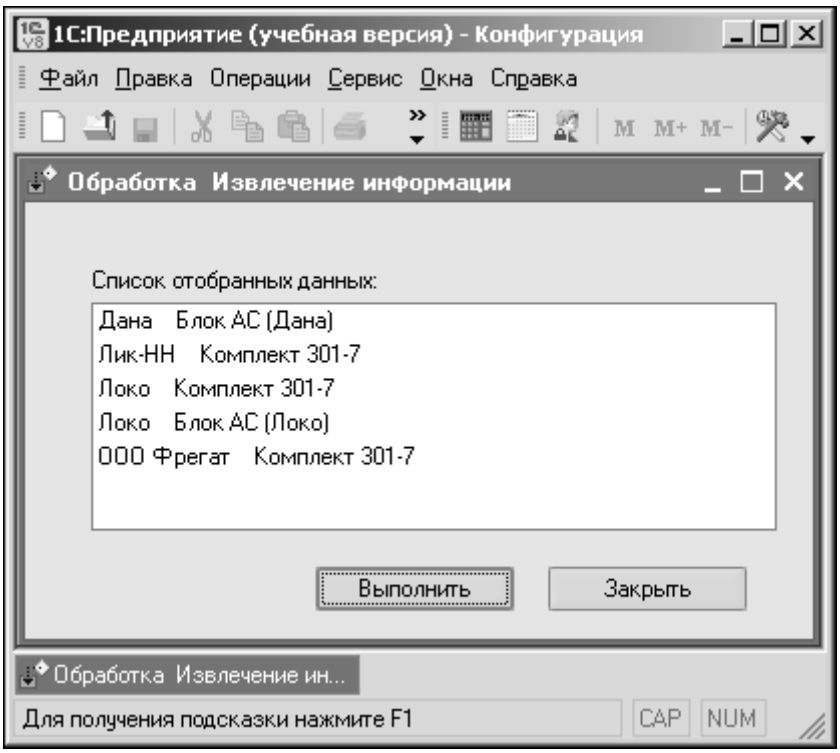

Рис. 2.32. Отбор названий товаров с указанием фирмы

Рассмотрим еще один пример на данную тему. Допустим нам необходимо отобрать филиалы, на которые поступали товары от фирм-партнеров и одновременно были перемещения с выставок. Текст процедуры приведен в листинге 2.22, а результат ее выполнения в режиме 1С:Предприятие представлен на рис. 2.34.

Листинг 2.22. Вторая процедура, демонстрирующая использование внутреннего соединения

```
Процедура КнопкаВыполнитьНажатие(Кнопка)
Запрос = Новый Запрос;
Запрос.Текст = "ВЫБРАТЬ РАЗЛИЧНЫЕ Поступление.Филиал КАК Филиал1
    | ИЗ Документ.ПоступлениеТоваров КАК Поступление
    | СОЕДИНЕНИЕ
    | Документ.ПеремещениеСВыставки КАК Перемещение
    | ПО Поступление.Филиал = Перемещение.Филиал";
Результат = Запрос.Выполнить().Выбрать();
СписокОтобранныхДанных.Очистить();
Пока Результат.Следующий() > 0 Цикл
   СписокОтобранныхДанных.Добавить(Строка(Результат.Филиал1));
   КонецЦикла;
КонецПроцедуры
```

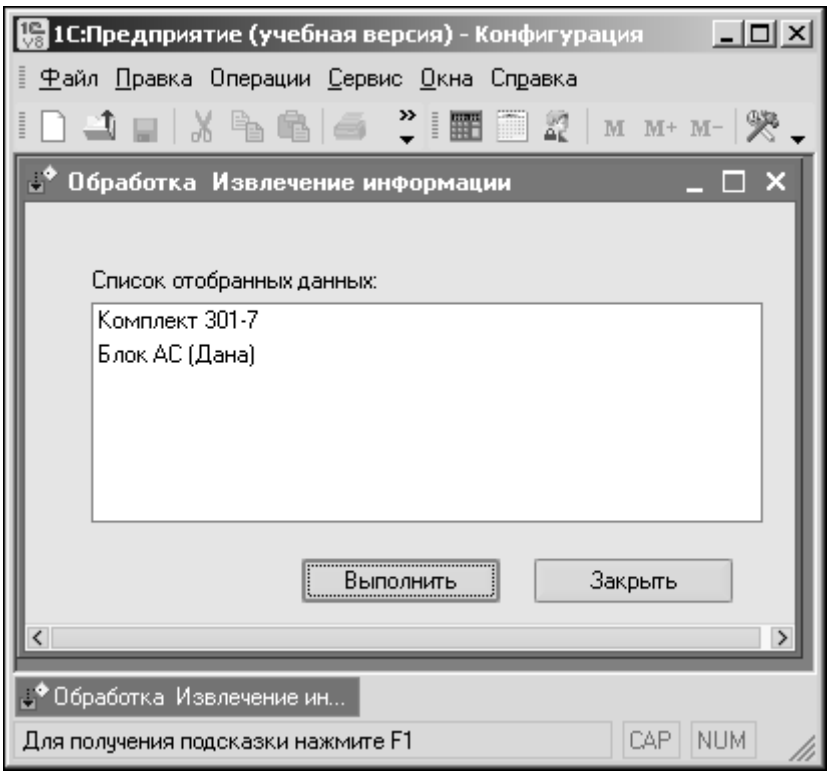

Рис. 2.33. Пример внутреннего соединения таблиц в запросе

Левое внешнее соединение

Рассмотренные примеры соединений запросов относятся к классу *внутренних*. В языке запросов системы 1С:Предприятие 8.1 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 надо включить еще и записи из первого источника (указанного слева от слова `СОЕДИНЕНИЕ`), для которых не найдено соответствующих условию записей из второго источника.

Таким образом, в результат запроса будут включены все записи из первого источника, они будут соединены с записями из второго источника при выполнении указанного условия. Строки результата запроса, для которых не найдено соответствующих условию записей из второго источника, будут содержать значение `NULL` в полях, формируемых на основании записей из этого источника.

> NULL-значения являются специальными маркерами, обозначающими не указанные (отсутствующие) значения или значения, не имеющие смысла.

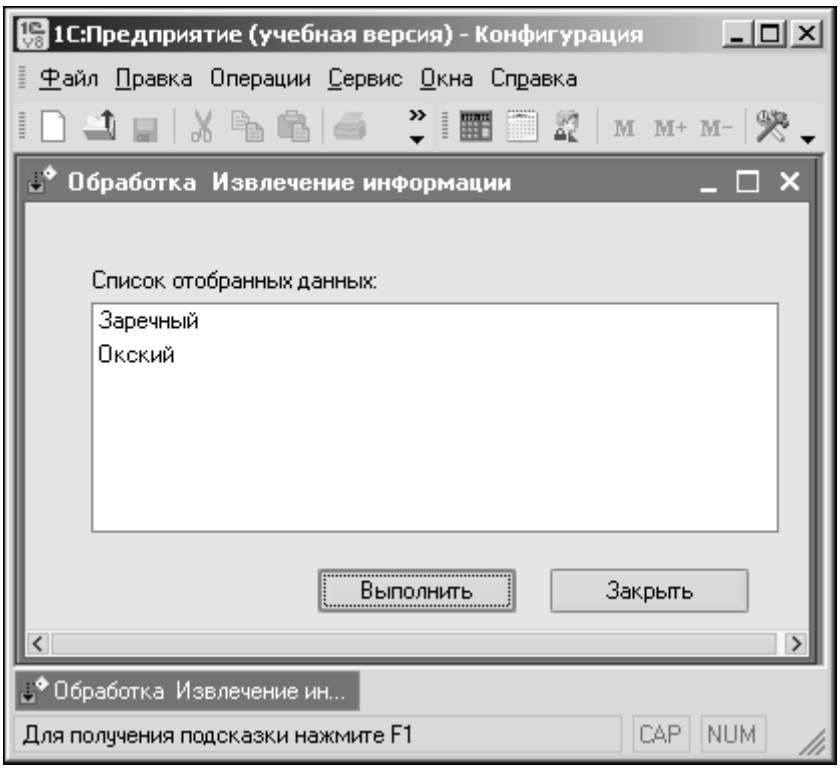

Рис. 2.34. Еще один пример внутреннего соединения таблиц в запросе

Например, необходимо выбрать фамилии всех менеджеров с указанием общей суммы выполненных ими продаж, однако при этом в результирующую таблицу необходимо включить и менеджеров, которые еще не оформляли документов по продажам. Это реализуется с помощью конструкции `ЛЕВОЕ [ВНЕШНЕЕ] СОЕДИНЕНИЕ` (листинг 2.23). Пример выполнения отбора (рис. 2.35) показывает, что в этом случае (в отличие от внутреннего соединения) в результирующем списке данных присутствуют и менеджеры, для которых еще не зарегистрировано продаж.

Листинг 2.23. Процедура, демонстрирующая левое внешнее соединение

```
Процедура КнопкаВыполнитьНажатие(Кнопка)
Запрос = Новый Запрос;
Запрос.Текст = "ВЫБРАТЬ НашиМенеджеры.Наименование КАК НашМенеджер,
    | Продажи.СуммаОборот КАК Сумма
    | ИЗ Справочник.Менеджеры КАК НашиМенеджеры
    | ЛЕВОЕ СОЕДИНЕНИЕ
    | РегистрНакопления.ПродажиПоФилиалам.Обороты КАК Продажи
    | ПО НашиМенеджеры.Наименование = Продажи.Менеджер.Наименование";
Результат = Запрос.Выполнить().Выбрать();
СписокОтобранныхДанных.Очистить();
Пока Результат.Следующий() > 0 Цикл
   СписокОтобранныхДанных.Добавить(Строка(Результат.НашМенеджер)+ "
"+
```

```
    Строка(Результат.Сумма));
    КонецЦикла
КонецПроцедуры
```

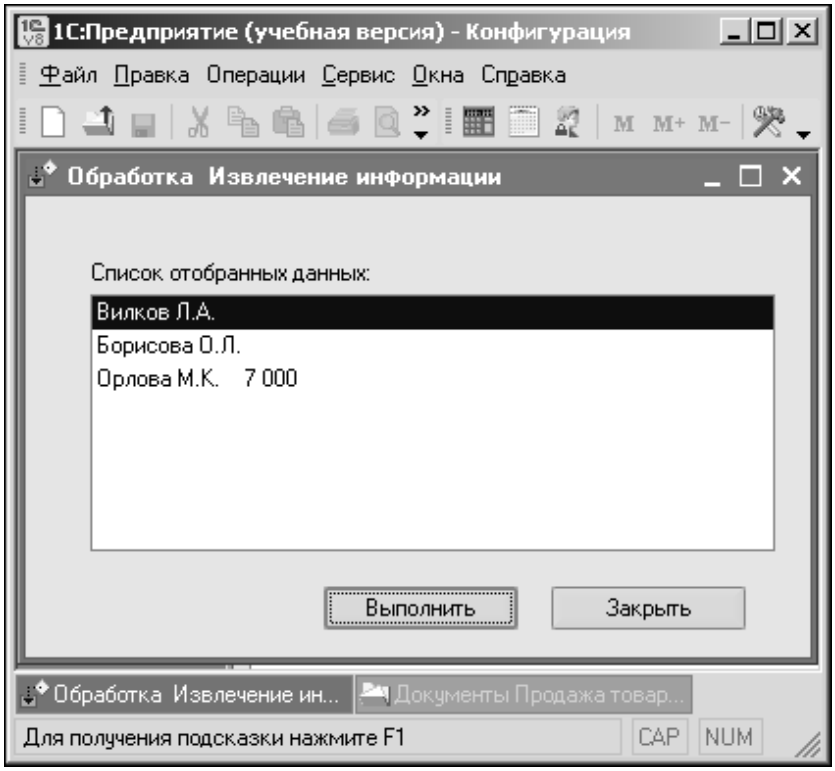

Рис. 2.35. Пример левого внешнего соединения таблиц в запросе

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 меняются местами. В качестве примера приведем формирование аналогичного с предыдущей ситуацией списка менеджеров, однако при этом используем правое соединение, а также дополнительно выведем информацию по филиалам, где были продажи. Текст программной процедуры с необходимым запросом приведен в листинге 2.24, а пример отобранной информации показан на рис. 2.36.

Листинг 2.24. Процедура, демонстрирующая правое внешнее соединение

```
Процедура КнопкаВыполнитьНажатие(Кнопка)
Запрос = Новый Запрос;
Запрос.Текст = "ВЫБРАТЬ НашиМенеджеры.Наименование КАК НашМенеджер,
    | Продажи.Филиал КАК НашФилиал,
    | Продажи.СуммаОборот КАК Сумма
    | ИЗ РегистрНакопления.ПродажиПоФилиалам.Обороты КАК Продажи
    | ПРАВОЕ СОЕДИНЕНИЕ
    | Справочник.Менеджеры КАК НашиМенеджеры
    | ПО НашиМенеджеры.Наименование = Продажи.Менеджер.Наименование";
Результат = Запрос.Выполнить().Выбрать();
СписокОтобранныхДанных.Очистить();
Пока Результат.Следующий() > 0 Цикл
   СписокОтобранныхДанных.Добавить(Строка(Результат.НашМенеджер)+"
"+
   Строка(Результат.НашФилиал)+ "    "+
   Строка(Результат.Сумма));
   КонецЦикла;
КонецПроцедуры
```

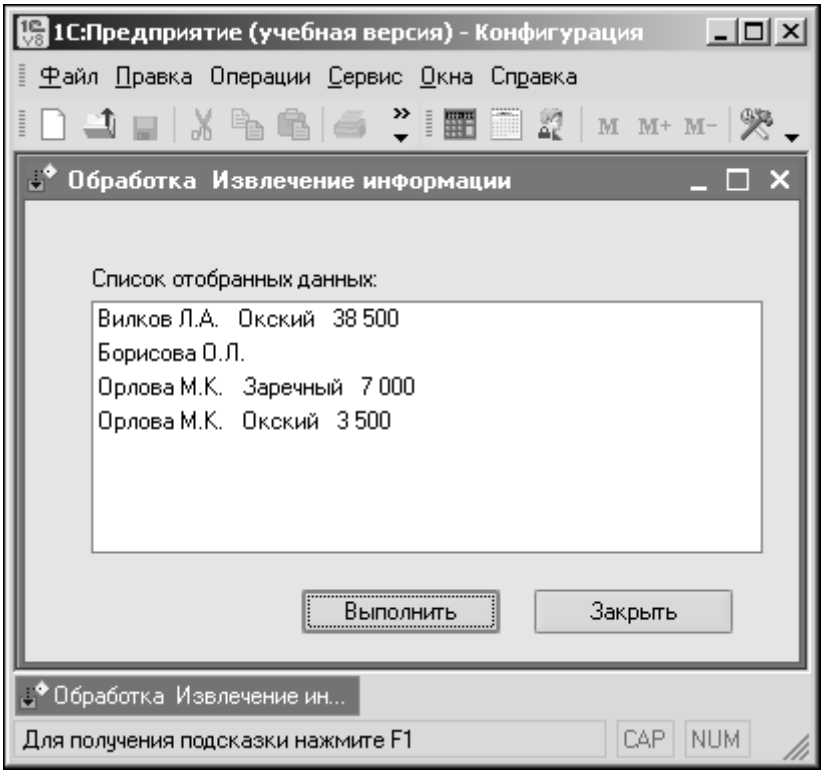

Рис. 2.36. Пример правого внешнего соединения таблиц в запросе

Рассмотрим еще одну ситуацию на данную тему. Допустим, нам необходимо составить общий список товаров с учетом их количества (сумма одинаковых товаров на всех складах). Для этого мы используем информацию из виртуальной таблицы остатков регистра накопления:

```
РегистрНакопления.КоличествоТоваров.Остатки
```

Однако в указанный список необходимо включить и товары, по которым еще не было поступлений (это новые товары, которые зарегистрированы только в справочнике Товары). Фактически подобных товаров в регистре КоличествоТоваров нет. Поэтому для извлечения информации мы воспользуемся еще одной таблицей — справочником Товары. В листинге 2.25 показан текст необходимой процедуры, в которой используется правое соединение таблиц. Результат ее выполнения в режиме 1С:Предприятие продемонстрирован на рис. 2.37. Здесь в список вошли товары, которые к нам уже поступали, а также те, которые пока не фигурировали в документах по поступлению.

Листинг 2.25. Еще одна процедура, демонстрирующая правое внешнее соединение

```
Процедура КнопкаВыполнитьНажатие(Кнопка)
Запрос = Новый Запрос;
Запрос.Текст = "ВЫБРАТЬ Товары.Ссылка КАК НазваниеТовара,
    | КоличествоТоваров.КоличествоОстаток КАК Количество
    | ИЗ РегистрНакопления.КоличествоТоваров.Остатки КАК КоличествоТоваров
    | ПРАВОЕ СОЕДИНЕНИЕ
    | Справочник.Товары КАК Товары
    | ПО Товары.Ссылка = КоличествоТоваров.Товар.Ссылка
    | УПОРЯДОЧИТЬ ПО Количество УБЫВ";
Результат = Запрос.Выполнить().Выбрать();
СписокОтобранныхДанных.Очистить();
Пока Результат.Следующий() > 0 Цикл
   СписокОтобранныхДанных.Добавить(Строка(Результат.НазваниеТовара)+
   "       "+
   Строка(Результат.Количество));
   КонецЦикла;
КонецПроцедуры
```

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 надо включить также еще и те записи из обоих источников, для которых не найдено соответствий. Таким образом, в результат запроса будут включены все записи из обоих источников и они будут соединены друг с другом при выполнении указанного условия.

Сформулируем задачу, которую нам необходимо решить. Будем считать, что необходимо отобрать фирмы, которые выступали как в качестве продавцов, так и в качестве покупателей. При этом необходимо в результат включить и те фирмы, которые выступали только в одном качестве (только в качестве продавца или только в качестве покупателя). В этом примере мы оформим результат в виде табличного документа (рис. 2.38). Таким образом, необходимые подготовительные действия будут связаны с изменениями обработки ИзвлечениеВТаблицу. Также нам необходимо создать новый макет (рис. 2.39), который будет использоваться программной процедурой заполнения табличного документа.

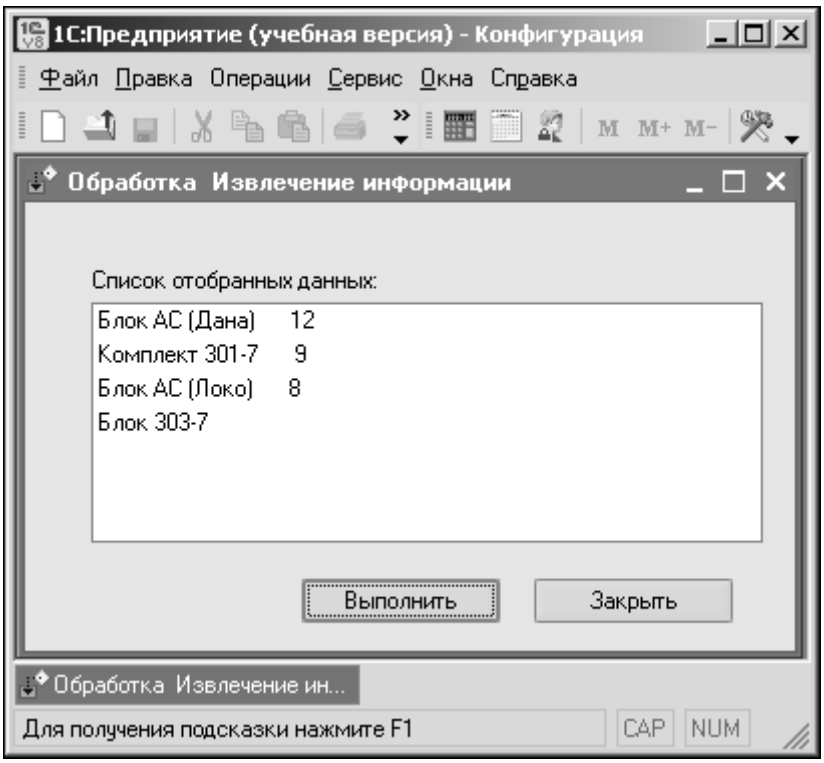

Рис. 2.37. Пример формирования списка товаров с использованием правого внешнего соединения

В листинге 2.26 представлен программный код, необходимый для выполнения запроса и заполнения табличного документа.

Листинг 2.26. Процедура, демонстрирующая полное внешнее соединение

```
Процедура КнопкаВыполнитьНажатие(Кнопка)
Запрос = Новый Запрос;
Запрос.Текст = "ВЫБРАТЬ РАЗЛИЧНЫЕ Продажа.Фирма.Наименование
    | КАК ФирмаПокупатель,
    | Поступление.Фирма.Наименование КАК ФирмаПродавец
    | ИЗ Документ.ПродажаТоваров КАК Продажа
    | ПОЛНОЕ СОЕДИНЕНИЕ
    | Документ.ПоступлениеТоваров КАК Поступление
    | ПО Продажа.Фирма.Наименование = Поступление.Фирма.Наименование";
Результат = Запрос.Выполнить().Выбрать();
ТабДок = Новый ТабличныйДокумент;
Макет = Обработки.ИзвлечениеВТаблицу.ПолучитьМакет("МакетДляПолного-
Соединения");
// Заголовок
Область = Макет.ПолучитьОбласть("Заголовок");
ТабДок.Вывести(Область);
// Шапка
Область = Макет.ПолучитьОбласть("Шапка");
ТабДок.Вывести(Область);
```

```
// Перечень
Пока Результат.Следующий() Цикл
    Область = Макет.ПолучитьОбласть("Строка");
    Область.Параметры.ФирмаПокупатель = Результат.ФирмаПокупатель;
    Область.Параметры.ФирмаПродавец = Результат.ФирмаПродавец;
   ТабДок.Вывести(Область);
    КонецЦикла;
ТабДок.ОтображатьСетку = Ложь;
ТабДок.Защита = Ложь;
ТабДок.ТолькоПросмотр = Ложь;
ТабДок.ОтображатьЗаголовки = Ложь;
ТабДок.Показать();
КонецПроцедуры
```

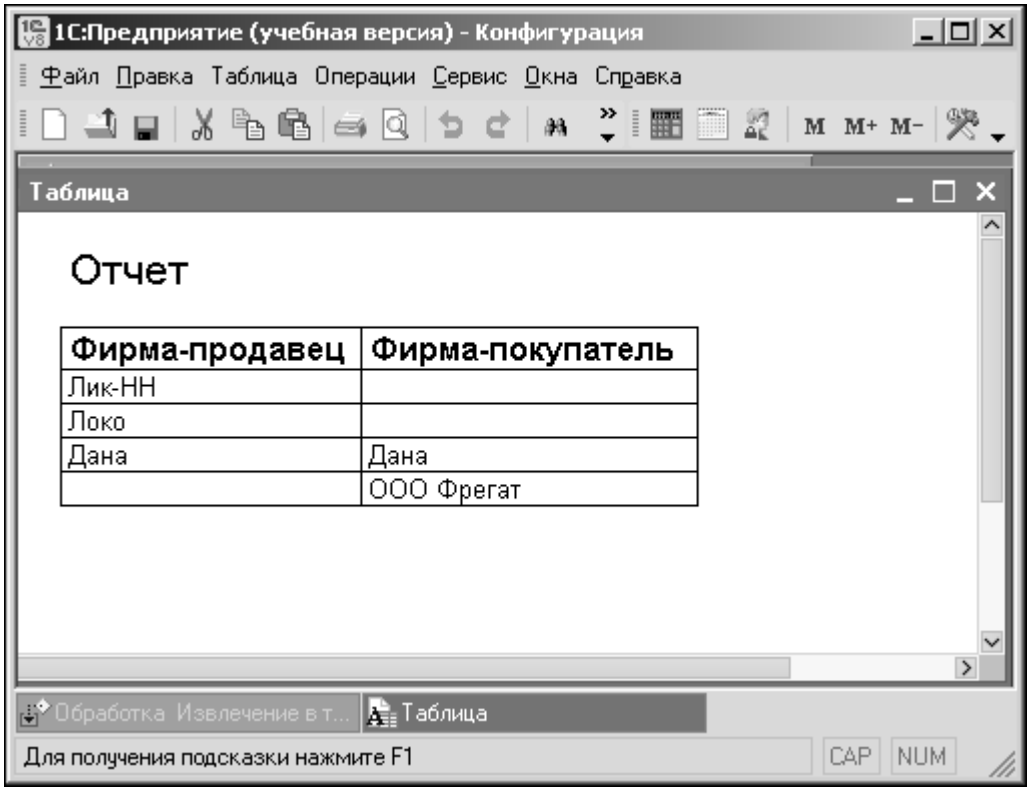

Рис. 2.38. Пример отчета с использованием полного внешнего соединения таблиц в запросе

Оператор ПОДОБНО

Часто при построении запросов мы располагаем не точной информацией о том, что требуется найти, а какой-то ее частью. Например, мы знаем, что фамилия интересующего нас человека включает начальный фрагмент — "Сидор". Ситуация достаточно распространенная: мы не помним точно фамилию человека — она может быть или *Сидоров*, или *Сидорова*, или *Сидоренко*.

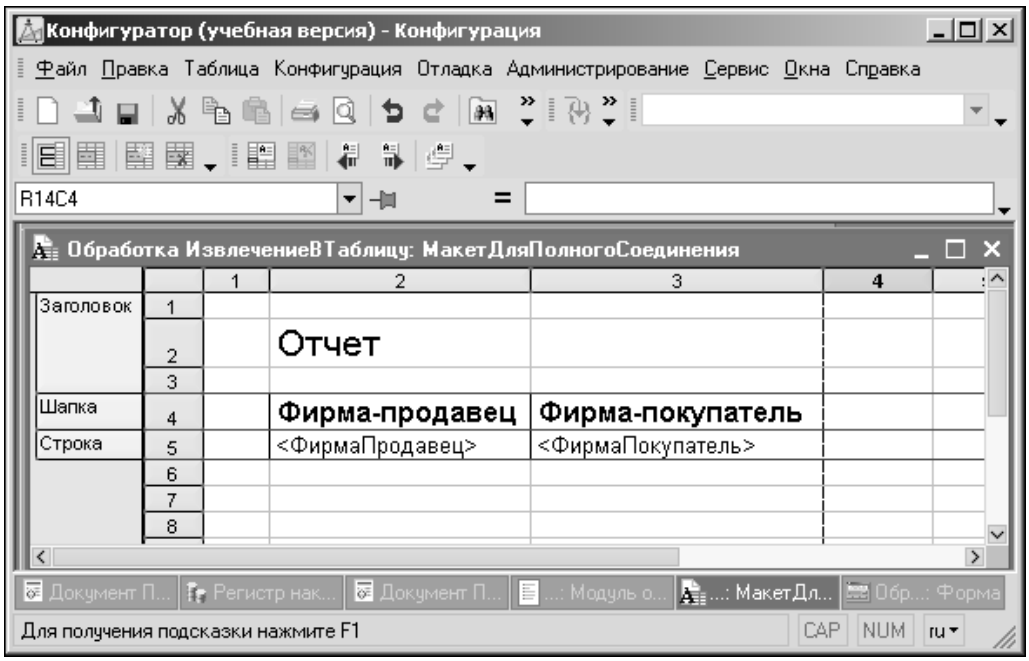

Рис. 2.39. Макет табличного документа для отбора фирм

Для иллюстрации оператора поиска информации по фрагменту текста необходимо в среде 1С:Предприятие создать интерфейс, аналогичный показанному на рис. 2.40. Практические шаги в данном случае заключаются во внесении требуемых изменений в форму обработки ИзвлечениеВТаблицу. Так, в форме мы разместили поле ввода (имя Должность, тип данных — `строка`). Теперь пользователь может внести любую часть названия должности, если даже он его знает не полностью. Далее, после щелчка на кнопке Сформировать, формируется табличный документ, представляющий существующие в справочнике Фирмы фамилии сотрудников с указанием их телефонов, которые удовлетворяют условиям поиска.

Текст процедуры обработки щелчка на кнопке Выполнить приведен в листинге 2.27. Данная процедура выполняет запрос к табличной части справочника Фирмы. Источником запроса является справочник, а отбор выполняется по значению реквизита `Должность`. Отбираются только те сотрудники, в должности которых присутствует значение параметра `Фрагмент`. Условие отбора обеспечивается за счет следующей конструкции языка запросов:

```
ГДЕ Должность ПОДОБНО &Фрагмент
```

Здесь оператор `ПОДОБНО` позволяет сравнить значение выражения, указанного слева от него, со строкой шаблона, указанной справа. Значение выражения должно быть типа `строка`.

Перед выполнением процедуры в режиме 1С:Предприятие необходимо разработать макет табличного документа (рис. 2.41).

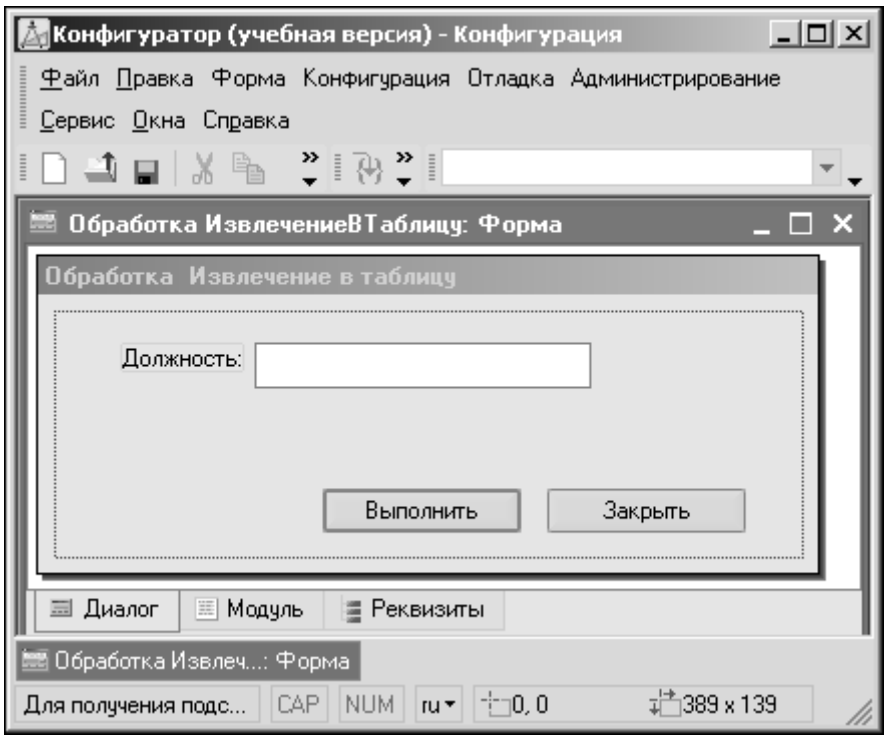

Рис. 2.40. Изменения в форме для организации поиска сотрудника по должности

Листинг 2.27. Процедура, демонстрирующая использование оператора ПОДОБНО

```
Процедура КнопкаВыполнитьНажатие (Кнопка)
Запрос = Новый Запрос;
Запрос.Текст = "ВЫБРАТЬ КонтактныеЛица.Ссылка КАК Фирма,
    | Сотрудник,Должность,Телефон
    | ИЗ Справочник.Фирмы.КонтактныеЛица КАК КонтактныеЛица
    | ГДЕ Должность ПОДОБНО &Фрагмент";
Запрос.УстановитьПараметр("Фрагмент",
      "%"+ЭлементыФормы.Должность.Значение+"%");
Результат = Запрос.Выполнить().Выбрать();
ТабДок = Новый ТабличныйДокумент;
Макет = Обработки.ИзвлечениеВТаблицу.ПолучитьМакет("МакетДляПоиска");
// Заголовок
Область = Макет.ПолучитьОбласть("Заголовок");
ТабДок.Вывести(Область);
// Шапка
Область = Макет.ПолучитьОбласть("Шапка");
ТабДок.Вывести(Область);
// Перечень
Пока Результат.Следующий() Цикл
   Область = Макет.ПолучитьОбласть("Строка");
   Область.Параметры.Фирма = Результат.Фирма;
   Область.Параметры.Сотрудник = Результат.Сотрудник;
   Область.Параметры.Должность = Результат.Должность;
   Область.Параметры.Телефон = Результат.Телефон;
```

```
    ТабДок.Вывести(Область);
    КонецЦикла;
ТабДок.ОтображатьСетку = Ложь;
ТабДок.Защита = Ложь;
ТабДок.ТолькоПросмотр = Ложь;
ТабДок.ОтображатьЗаголовки = Ложь;
ТабДок.Показать();
КонецПроцедуры
```

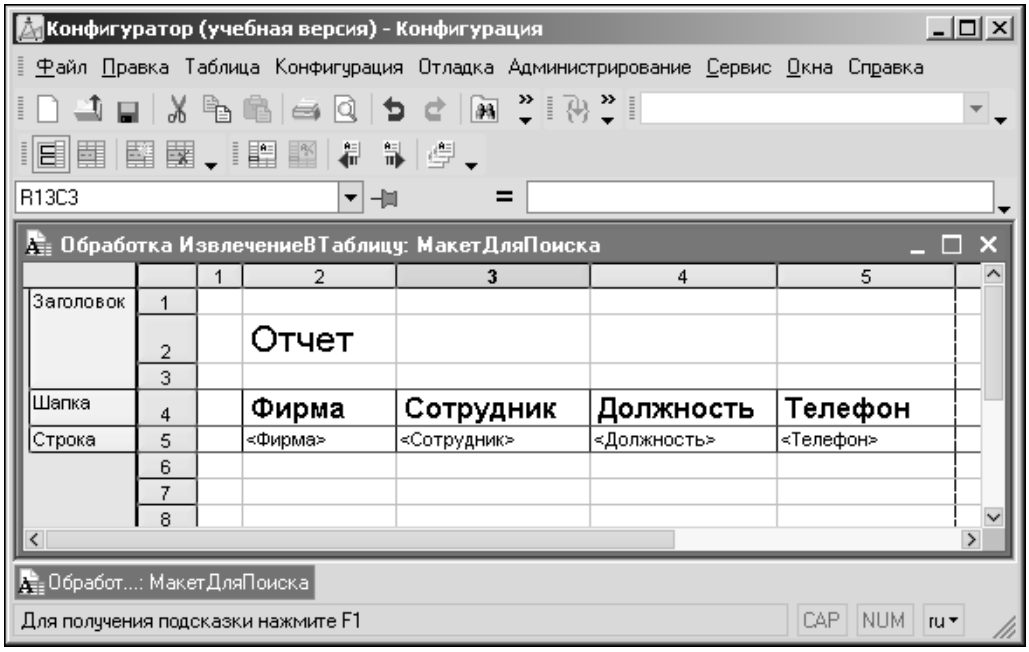

Рис. 2.41. Макет для табличного документа

В приведенной процедуре новой является следующая конструкция:

```
Запрос.УстановитьПараметр("Фрагмент",
      "%"+ЭлементыФормы.Должность.Значение+"%")
```

Здесь учитывается, что строка шаблона допускает использование ряда служебных символов (табл. 2.2). И значение параметра `Фрагмент` устанавливается уже знакомым нам способом с использованием символа “`%`”, обозначающего последовательность, содержащую любое количество символов.

Таблица 2.2. Служебные символы, применяемые в операторе ПОДОБНО

Символ	Комментарий
`%`	Обозначает последовательность, содержащую любое количество символов
_ (подчеркивание)	Обозначает один произвольный символ
`[...]` (один или несколько символов в квадратных скобках)	Любой одиночный символ из перечисленных внутри квадратных скобок. В перечислении могут встречаться диапазоны, например, `a-z`, что означает произвольный символ, входящий в диапазон, включая концы диапазона
`[^...]`	Любой одиночный символ, кроме тех, которые перечислены следом за символом отрицания

На рис. 2.42 приведен результат поиска в режиме 1С:Предприятие — мы получили необходимый табличный документ с информацией по сотрудникам.

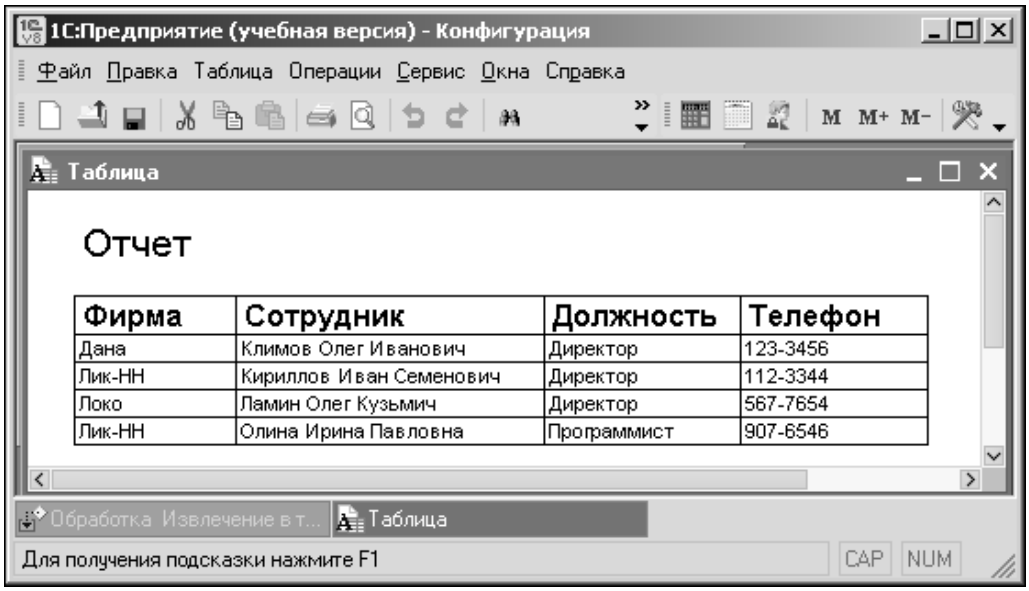

Рис. 2.42. Формирование информации по строке поиска

Функция ПРЕДСТАВЛЕНИЕ

Эта функция предназначена для получения строкового представления значения произвольного типа. В качестве параметра функции `ПРЕДСТАВЛЕНИЕ` может выступать выражение любого типа. Например, можно привести к строковому типу информацию о датах документов. В качестве примера рассмотрим вывод дат документов ПоступлениеТоваров и ПеремещениеСВыставки в поле списка в форме обработки ИзвлечениеИнформации.

Текст необходимой процедуры обработки щелчка на кнопке Выполнить приведен в листинге 2.28. В рассматриваемой процедуре осуществляется объединение двух запросов, которые выполняют:

- извлечение дат документов ПоступлениеТоваров;
- извлечение дат документов ПеремещениеСВыставки.

Листинг 2.28. Процедура, демонстрирующая использование функции ПРЕДСТАВЛЕНИЕ

```
Процедура КнопкаВыполнитьНажатие(Кнопка)
Запрос = Новый Запрос;
Запрос.Текст = "ВЫБРАТЬ ПРЕДСТАВЛЕНИЕ(Поступление.Дата) КАК Дата1
    | ИЗ Документ.ПоступлениеТоваров КАК Поступление
    | ОБЪЕДИНИТЬ
    | ВЫБРАТЬ ПРЕДСТАВЛЕНИЕ(Перемещение.Дата)
    | ИЗ Документ.ПеремещениеСВыставки КАК Перемещение";
```

```
Результат = Запрос.Выполнить().Выбрать();
СписокОтобранныхДанных.Очистить();
Пока Результат.Следующий() > 0 Цикл
   СписокОтобранныхДанных.Добавить(Результат.Дата1);
   КонецЦикла;
КонецПроцедуры
```

Результат выполнения процедуры отбора дат документов показан на рис. 2.43.

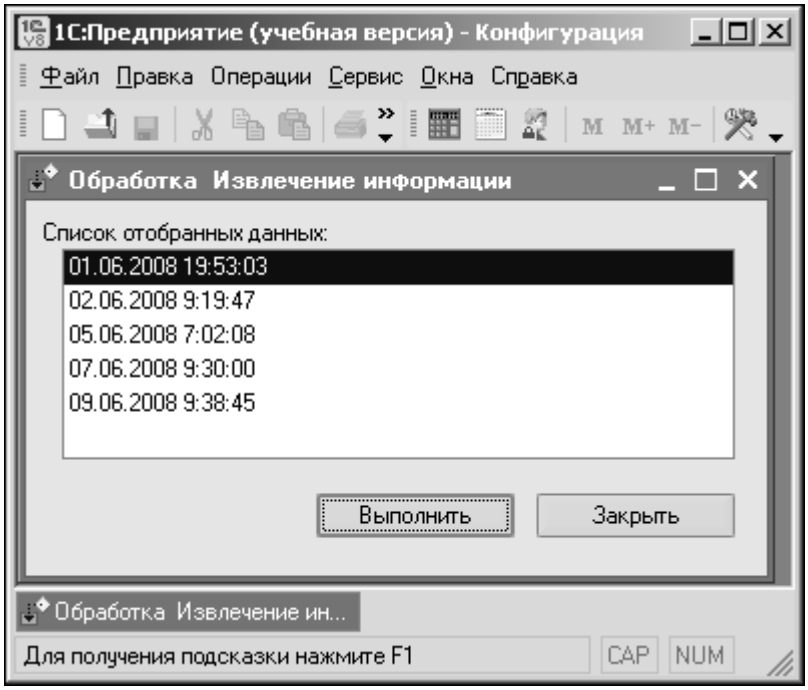

Рис. 2.43. Информация о датах документов ***ПоступлениеТоваров*** *и* ***ПеремещениеСВыставки***

Функция РАЗНОСТЬДАТ

Часто в практических ситуациях требуется вычисление разности между двумя датами — например, между датой окончания обучения и датой зачисления на обучение. Функция `РАЗНОСТЬДАТ` как раз и предназначена для получения разности между датами. У нее три параметра: первые два имеют тип `Дата`, а третий определяет, в каких единицах измерения необходимо получить результат. Здесь возможны следующие варианты: `Секунда`, `Минута`, `Час`, `День`, `Месяц`, `Квартал`, `Год`.

Допустим, нам необходимо получить информацию по максимальной разности дат поступления товаров на филиал и продажей с филиала. Для определенности эту разность отобразим в днях. Лучшее представление о задаче дает результат ее выполнения в режиме 1С:Предприятие, показанный на рис. 2.44.

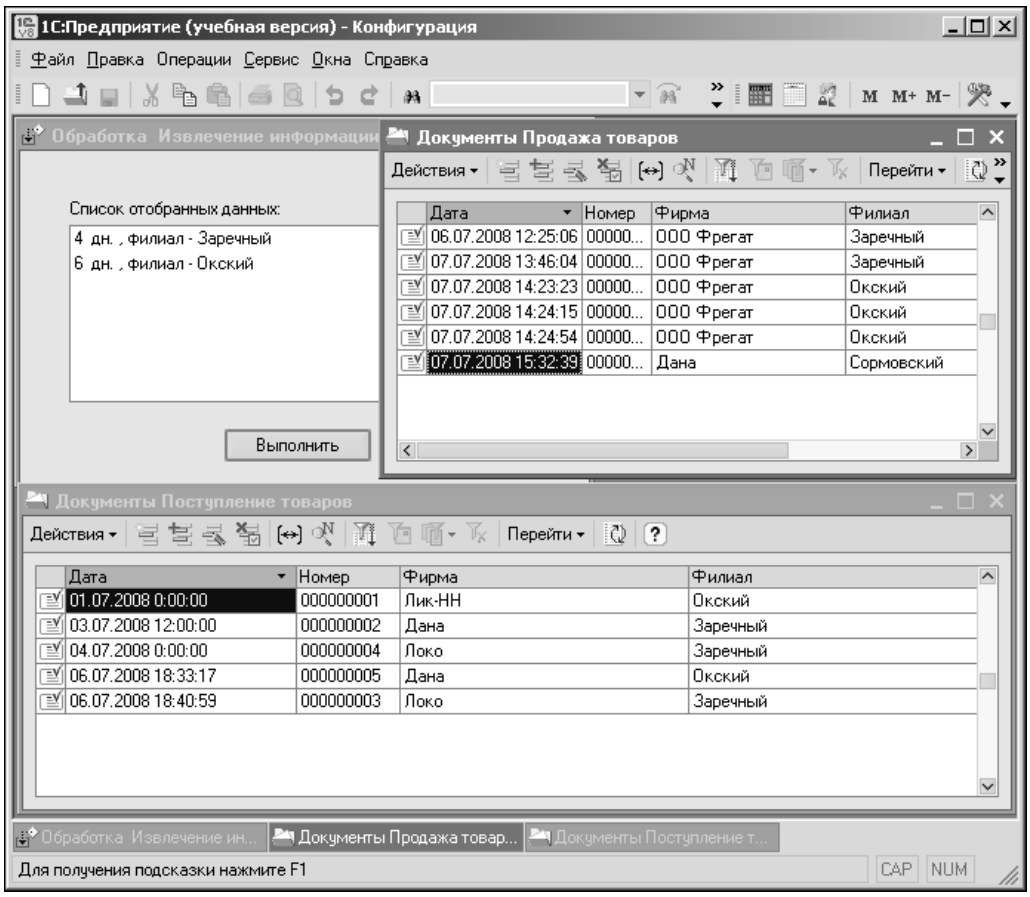

Рис. 2.44. Отчет по максимальной разности дат документов поступления и продажи

В листинге 2.29 приведена необходимая процедура, использующая соединение таблиц в конструкции запроса.

Листинг 2.29. Процедура, демонстрирующая использование функции РАЗНОСТЬДАТ

```
Процедура КнопкаВыполнитьНажатие(Кнопка)
Запрос = Новый Запрос;
Запрос.Текст = "ВЫБРАТЬ Поступление.Филиал КАК НашФилиал,
    | МАКСИМУМ(РазностьДат(Поступление.Дата,Продажа.Дата, День))
       КАК Разность
    | ИЗ Документ.ПоступлениеТоваров КАК Поступление
    | СОЕДИНЕНИЕ
    | Документ.ПродажаТоваров КАК Продажа
    | ПО Поступление.Филиал = Продажа.Филиал
    | СГРУППИРОВАТЬ ПО Поступление.Филиал";
Результат = Запрос.Выполнить().Выбрать();
СписокОтобранныхДанных.Очистить();
Пока Результат.Следующий() > 0 Цикл
   СписокОтобранныхДанных.Добавить(Строка(Результат.Разность) +
```

```
    "  дн. , филиал - "+
    Строка(Результат.НашФилиал));
    КонецЦикла;
КонецПроцедуры
```

Итоги

В этой главе мы на примерах познакомились с языком запросов. Разнообразные примеры продемонстрировали этот важный инструментальный механизм, который разработчики используют для построения необходимых выборок из базы данных. Взяв за исходный материал созданную ранее информационную базу, мы фактически дополнили разработанную конфигурацию разнообразными средствами просмотра информации в удобном для пользователя виде.

Большинство примеров приведено в полном виде, включающем как текст запроса, так и саму процедуру, позволяющую данный запрос выполнить. Такое представление должно быть удобно для читателей. Конечно, конструкции языка запросов весьма разнообразны, и всю информацию на данную тему мы, естественно, в этой главе не охватили. Однако теперь читатели уже смогут самостоятельно отправиться по лабиринтам программных ресурсов системы 1С:Предприятие, используя встроенную справочную документацию, а также ряд очень хороших изданий [1–5].

С языком запросов мы встретимся и в последующих главах, где много места будет уделено построению отчетов. Кроме того, мы узнаем о наличии в системе 1С:Предприятие 8.1 специальных средств автоматизации, предназначенных для формирования отчетов. Но в целом уже сейчас читатели смогут самостоятельно строить запросы для решения собственных задач получения выборки информации из их таблиц баз данных.