

Глава 15. Создание отчётности .....	373
15.1 Создание отчётности в Excel.....	374
15.2 Отчётность в Quick Reports .....	379
15.3 Печать таблиц с помощью Quick Reports .....	383
15.4 Печать связанных таблиц. ....	384
15.5 Дополнительные приборбасы.....	385



## Глава 15. Создание отчётности

С базой данных мы уже научились работать и создали теперь уже полноценный телефонный справочник. Но какая база данных без отчётности? Практически всегда возникает потребность в выгрузке данных в другую программу.

Все примеры будут работать с базами данных, потому что именно при их программировании возникает потребность в создании каких-то выходных документов. Нет смысла вести данные, которые нельзя распечатать или выгрузить в другой формат. В этой главе нам предстоит познакомиться с выгрузкой данных в Excel и научиться формировать красивые выходные документы, которые будут готовы к распечатке.

Но даже если ты пока не думаешь о создании отчётности, я всё же советую тебе ознакомиться с этой главой, потому что здесь будет достаточно много интересных вещей, о которых я пока что не говорил. Например, в прошлой главе я не мог рассмотреть все свойства и методы таблицы `ADOTable`, поэтому были рассмотрены только основные возможности. В этой главе будет ещё множество примеров по работе с данными, так что мы закрепим на практике уже знаковые методы и познакомимся с некоторыми новыми.



## 15.1 Создание отчётности в Excel

Первое, с чем мы познакомимся – отчётность в Excel. Потребность в выгрузке данных в Excel может возникнуть у каждого программиста баз данных, ведь Office установлен в нашей стране практически на каждом компьютере. А это значит, что нашу отчётность можно смело переносить между компьютерами и быть уверенным, что её смогут прочитать.

В этой части будет не так уж много визуальных манипуляций, зато кода будет предостаточно. Поэтому про мышку можно позабыть и поближе подвинуть клавиатуру.

Но в начале всё таки ткнём мышкой и добавим на форму одну кнопку, по нажатию которой будет создаваться отчётность. Я ещё добавил к программе пункт меню «Экспорт в Excel» в меню «Файл». Результат можешь увидеть на рисунке 15.1.1.

Вот теперь мышку можно убирать. Переходи в редактор кода и сразу же добавляй в раздел **uses** модуль *ComObj*. В этом модуле описаны все необходимые функции для работы с COM объектами (мы о них пока не говорили, но всё ещё впереди).

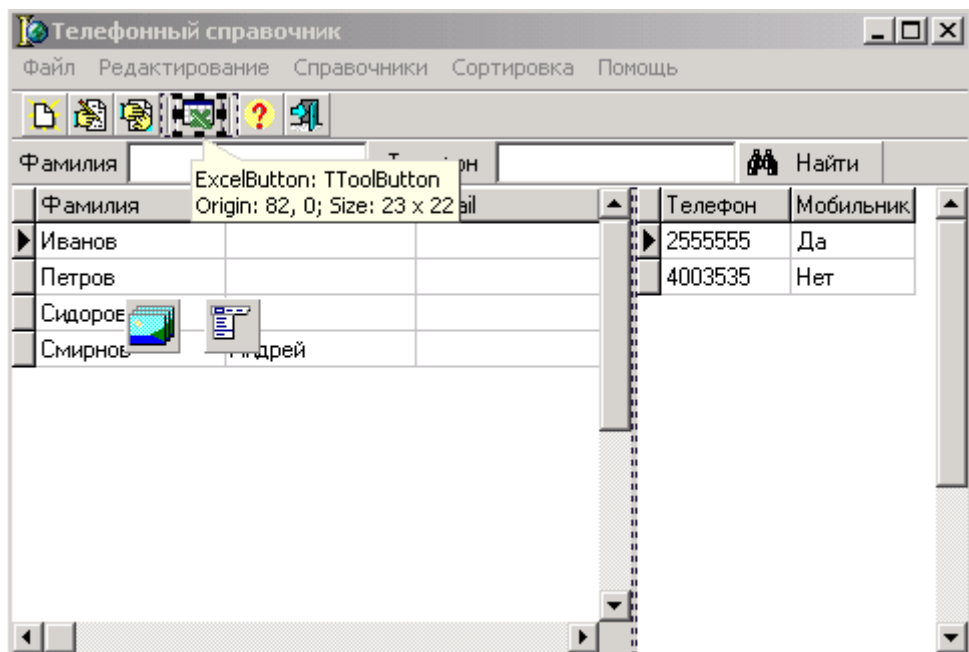


Рисунок 15.1.1 Кнопка создания отчётности в Excel

Теперь создавай обработчик события для кнопки и укажи его же в качестве обработчика для пункта меню (если ты его создал). В этом обработчике нужно написать следующий код:

```
var
  XLApp, Sheet, Column: Variant;
  index, i: Integer;
begin
  XLApp:= CreateOleObject('Excel.Application');
  XLApp.Visible:=true;
  XLApp.Workbooks.Add(-4167);
  XLApp.Workbooks[1].Worksheets[1].Name:='Отчёт';
  Column:=XLApp.Workbooks[1].Worksheets['Отчёт'].Columns;
  Column.Columns[1].ColumnWidth:=20;
  Column.Columns[2].ColumnWidth:=20;
```

```

Column.Columns[3].ColumnWidth:=20;
Column.Columns[4].ColumnWidth:=20;
Column.Columns[5].ColumnWidth:=20;

Column:=XLApp.Workbooks[1].Worksheets['Отчёт'].Rows;
Column.Rows[2].Font.Bold:=true;
Column.Rows[1].Font.Bold:=true;
Column.Rows[1].Font.Color:=clBlue;
Column.Rows[1].Font.Size:=14;

Sheet:=XLApp.Workbooks[1].Worksheets['Отчёт'];
Sheet.Cells[1,2]:='Телефонный справочник';
Sheet.Cells[2,1]:='Фамилия';
Sheet.Cells[2,2]:='Имя';
Sheet.Cells[2,3]:='e-mail';
Sheet.Cells[2,4]:='Город';
Sheet.Cells[2,5]:='Дата рождения';

index:=3;
DataModule1.BookTable.First;
for i:=0 to DataModule1.BookTable.RecordCount-1 do
begin
  Sheet.Cells[index,1]:=DataModule1.BookTable.Fields.Fields[1].AsString;
  Sheet.Cells[index,2]:=DataModule1.BookTable.Fields.Fields[2].AsString;
  Sheet.Cells[index,3]:=DataModule1.BookTable.Fields.Fields[3].AsString;
  Sheet.Cells[index,4]:=DataModule1.BookTable.Fields.Fields[5].AsString;
  Sheet.Cells[index,5]:=FormatDateTime('dddddd',
    DataModule1.BookTable.Fields.Fields[6].AsDateTime);
  Inc(index);
  DataModule1.BookTable.Next;
end;

```

---

Первая строка создаёт объект *Excel* (*XLApp:= CreateOleObject('Excel.Application')*), и записывает его в переменную *XLApp*. Эта переменная типа *Variant*. *Variant* - это тип, который может принимать любые значения: строки, числа, указатели и др.

Функцию *CreateOleObject* я сейчас подробно рассматривать не буду, потому что она не относится к базам данных или отчётности, и требует отдельного разговора. Единственное, что я сейчас скажу – она позволяет наладить связь с другим приложением по технологии COM. Через эту связь можно передавать данные в чужие приложения. Для этого программа, к которой мы присоединяемся, должна иметь соответствующие возможности для получения данных из вне (как, например, Excel) и тебе должны быть известны функции с которыми можно работать. Чаще всего такие вещи документируются на сайте разработчиков. Я же не могу описать все возможности всех программ, потому что если это сделать, то «Война и Мир» покажутся детской колыбельной. Я думаю, это никому не нужно. Так что я рассмотрю только Excel, чтобы показать возможности передачи данных между приложениями.

Вторая строка (*XLApp.Visible:=true*) заставляет запустить сам *Excel*. Потом я добавляю новую рабочую книгу (*XLApp.Workbooks.Add(-4167)*). Число в скобках - это константа, которая означает создание книги и её изменять нельзя. Подробнее о всех константах ты можешь почитать в руководстве разработчика на сайте MS или в файле *excel97.pas*, а я не могу тратить на это место книги, потому что констант предостаточно.

Дальше я даю название созданной книге *XLApp.Workbooks[1].Worksheets[1].Name:='Отчёт'*. Это действие не обязательно, но я всегда это делаю, потому что меня бесит название по умолчанию "*Лист 1*". Всё должно быть понятно с первого взгляда.

Теперь у нас *Excel* запущен и создана новая книга. Можно переходить к в печатыванию данных. Но прежде чем это сделать я отформатирую колонки и строки. Для этого я получаю указатель на колонки рабочей книги (*Colum:= XLApp. Workbooks[1]. WorkSheets['Омчём']. Columns*), и записываю результат в переменную *Colum* типа *Variant*. Теперь последовательно изменяю ширину колонок (*Colum. Columns[1]. ColumnWidth := 20*). На русском эта команда будет звучать так: *Колонки.Колонка[1].ШиринаКолонки:=20*.

После этого я в ту же переменную записываю указатель на строки рабочей книги (*Colum := XLApp. Workbooks[1]. WorkSheets['Омчём']. Rows*). Для украшения строк нашего отчёта, я устанавливаю у первых двух строк жирный шрифт (*Colum. Rows[1]. Font. Bold := true*). В квадратных скобках теперь порядковый номер строки. Далее идут две строки, в которых я устанавливаю цвет первой строки в синий и размер шрифта равный 14.

Форматирование окончено, теперь можно выводить данные. Для этого я получаю указатель на лист (*Sheet:=XLApp.Workbooks[1].WorkSheets['Омчём']*). Для того, чтобы вывести данные, нужно просто присвоить значение в *Sheet.Cells[строка, колонки]*. Давай посмотрим на код вывода данных таблицы:

---

```
index:=3;
DataModule1.BookTable.First;
for i:=0 to DataModule1.BookTable.RecordCount-1 do
begin
  Sheet.Cells[index,1]:=DataModule1.BookTable.Fields.Fields[1].AsString;
  Sheet.Cells[index,2]:=DataModule1.BookTable.Fields.Fields[2].AsString;
  Sheet.Cells[index,3]:=DataModule1.BookTable.Fields.Fields[3].AsString;
  Sheet.Cells[index,4]:=DataModule1.BookTable.Fields.Fields[5].AsString;
  Sheet.Cells[index,5]:=FormatDateTime('dddddd',
    DataModule1.BookTable.Fields.Fields[6].AsDateTime);
  Inc(index);
  DataModule1.BookTable.Next;
end;
```

---

Сначала я задаю переменной *index* значение 3. Эта переменная будет отображать, в какую строку таблицы *Excel* мы сейчас должны выводить данные. Первые две строки у нас уже заняты заголовками для отчёта, поэтому данные нужно начинать выводить с третьей строки.



Обрати внимание, что строки и таблицы в *Excel* нумеруются начиная с единицы, а не с нуля, как в остальных таблицах и массивах.

---

После этого я перехожу на первую строку нашей таблицы с помощью метода *First* компонента *AdoTable*. Это необходимо, потому что пользователь может перед нажатием кнопки отчёта выделить любую строку в середине таблицы и в этом случае отчёт пойдёт от этой выделенной строки.

Теперь мы готовы запускать цикл, в котором будут перебираться все строки и информация из них будет попадать в *Excel*. Цикл я запускаю начиная с 0 и до количества строк в таблицы. Таким образом я переберу все записи.

Почему в качестве цикла я использую именно конструкцию *for..to..do*? Не знаю, просто иногда удобней так, а иногда я использую цикл *while*. С циклом *while* этот код выглядел бы так:

---

```
while DataModule1.BookTable.Eof<>true do
begin
  //Вывод данных в Excel
  Увеличение переменной index;
  Переход на следующей строку;
end;
```

---

В принципе, всё то же самое, только используется цикл *while*. Этот цикл удобнее использовать, когда нужно вывести не всю таблицу, а, например, только начиная с текущей позиции. В этом случае не надо будет перед циклом переходить на первую строку, а достаточно только запустить этот цикл, который будет выполняться, пока *DataModule1.BookTable.Eof* не станет равным *true*, т.е. не будет достигнут конец таблицы.

Только не забывай внутри цикла переходить на следующую строку. Если ты забудешь это сделать с циклом *for*, то у тебя в *Excel* попадёт столько же строк, сколько и в таблице, но все они будут одинаковыми (равны первой строке таблицы). Ну а при цикле *while* программа может зависнуть, потому что цикл будет бесконечным, ведь конец таблицы никогда не будет достигнут, если не переходить на следующие строки.

Теперь поговорим о выводе данных. Я последовательно заполняю колонки данными из таблицы, присваивая в *Sheet.Cells[index, номер колонки]*, соответствующие данные. Данные из базы данных я беру по индексу (до этого мы обращались по имени, но здесь я захотел показать тебе, как это делается по индексу). Для этого используется следующая конструкция:

```
DataModule1.BookTable.Fields.Fields[Номер поля].AsString
```

После вывода данных я увеличиваю переменную *Index*, чтобы на следующем этапе выводить данные в следующую строку и перехожу на новую строку.

В принципе, это и всё. Напоследок я дам несколько замечаний и покажу тебе универсальный способ вывод данных из сетки:

---

```
for i:=1 to Table.RecordCount do
begin
  for j:=1 to DBGrid.Columns.Count do
    Sheet.Cells[Index, j]:=DBGrid.Fields[j-1].AsString;

    Inc(Index);
    Table.Next;
  end;
```

---

Это просто общий пример не связанный с нашей программой. Здесь так же запускается цикл по всем строкам таблицы. Но вывод данных происходит по новому. Для этого запускается ещё один цикл от 1 до количества колонок в сетке *DBGrid*. А внутри этого цикла я присваиваю очередной колонке *Excel*, значение из такой же колонки сетки *DBGrid*:

```
Sheet.Cells[Номер строки, Номер колонки]:=  
DBGrid.Fields[Номер колонки в сетке].AsString;
```

Этот способ более универсален и может подойти почти всегда, когда надо вывести всё содержимое сетки DBGrid и все данные одного типа. В нашем случае в таблице есть дата, поэтому, чтобы она красиво выглядела я её форматирую с помощью *FormatDateTime*. Так что прямой перенос для этой колонки был бы не очень удобен, но остальные колонки можно было бы перенести указанным способом.

В процессе вывода данных можно изменять цвет строк *Sheet.Rows[строка].Font.Color* или колонок *Sheet.Columns[колонка].Font.Color*, простым присваиванием (как мы это делали при форматировании). Если нужно изменить цвет отдельной ячейки, то это можно сделать, присвоив новое значение в *Sheet.Cells[строка, колонка].Font.Color*.

Вот ещё некоторые параметры, которые ты можешь изменить:

*Sheet.Cells[строка, колонка].Font.Italic* - курсивный шрифт

*Sheet.Cells[строка, колонка].Font.Bold* - жирный шрифт

*Sheet.Cells[строка, колонка].Font.Underline* - подчёркнутый шрифт

*Sheet.Cells[строка, колонка].Font.Size* - размер шрифта

С помощью всего этого, ты сможешь создавать простые, но эффективные отчёты. В Delphi, в директории Lib есть файл excel97.pas, в нём ты найдёшь все доступные функции Excel. Если что-то не будет ясно, то пока отложи это занятие, чуть позже ты наберёшься навыков для понимания заголовочных файлов.

А у нашей программы остался один недостаток – она выводит данные только из основного справочника, и не учитывает телефоны. Чтобы избавиться от этого недостатка можно написать отдельный SQL запрос, который будет формировать сводную таблицу из наших двух. Этот запрос можно поместить в отдельный компонент *ADOQuery*, выполнить его и брать данные оттуда, а не из таблицы *BookTable*.

Вот пример запроса, который выбирает все строки из двух таблиц формируя отдельную базу данных.

---

```
SELECT *
FROM Справочник, Телефоны
WHERE Справочник.Key1=Телефоны.LinkKey
```

---

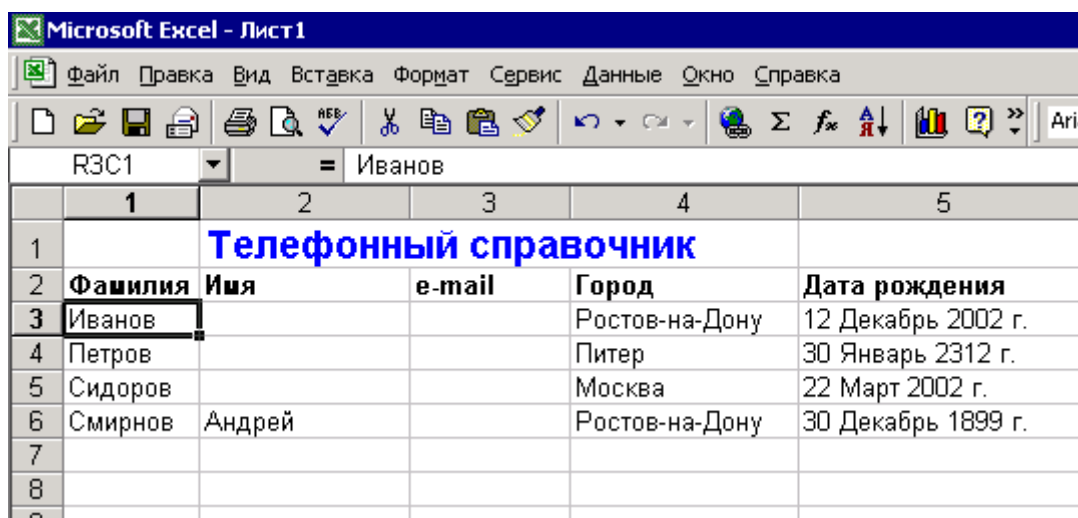



Рисунок 15.1.2 Кнопка создания отчётности в Excel

На рисунке 15.1.2 ты можешь увидеть результат работы программы. Вот так выглядит созданная мной отчётность.

 На компакт диске, в директории \Примеры\Глава 15\Excel ты можешь увидеть пример этой программы.

## 15.2 Отчётность в Quick Reports

Теперь я хочу познакомить тебя с мощным средством создания отчётов, которое входит в поставку Delphi – это Quick Reports. Он не является самым быстрым, и в сети Internet можно найти множество более быстрых, как платных, так и бесплатных генераторов отчётов. Но Quick Reports очень мощный и уже установлен в Delphi и готов к работе. Именно поэтому мы будем рассматривать его, Где бы ты не сел за Delphi, этот генератор уже будет установленным и его можно использовать.

Все компоненты Quick Reports находятся на закладке *QReport* палитры компонентов. В Delphi 6 у меня уже 23 компонента, позволяющие создать умопомрачительные документы готовые к печати.

Давай сначала посмотрим на головной компонент Quick Reports – ***TQuickRep***. Этот компонент – основа любого отчёта. Он представляет собой холст листа будущего отчёта. Попробуй дважды щёлкнуть по значку этого компонента в палитре компонентов и он перенесётся на форму во всей своей красе. Тебе надо выровнять края компоненты по форме и ты увидишь готовый белый лист, на котором можно будет размещать будущее твоего документа. По краям листа ты можешь видеть синие пунктирные линии, которые показывают границы документа.

Давай посмотрим на объектный инспектор и разберёмся с полями нашего отчёта:

***Bands*** – здесь ты можешь указать, что должен иметь будущий документ, а он может содержать:

***HasColumnHeader*** – Заголовки колонок. Если твой отчёт будет содержать таблицу, то она должна иметь шапку, где будут описаны названия колонок. Вот именно эту шапку создают в этой части документа. Так что если тебе нужна будет таблица, то этому свойству нужно будет присвоить *true*.

***HasDetail*** – если в отчёте есть таблица, то вид строк делается в этом разделе.

***HasPageFooter*** – в этом разделе создаётся нижний колонтитул.

***HasPageHeader*** – здесь создаётся заголовок документа.

***HasSummary*** –содержимое этого раздела печатается один раз в конце отчёта (на последней странице).

***HasTitle*** – в этом разделе делается заголовок отчёта.

Попробуй создать новый проект и бросить на форму один компонент *QuickRep*. Теперь включи какие-нибудь разделы и посмотри на результат. На форме должны появиться области очерченные пунктирной линией и внизу этой области должно быть написано её предназначение. Так ты легко можешь отличить их между собой. Любую область можно выделить и растянуть или уменьшить (рисунок 15.2.1).

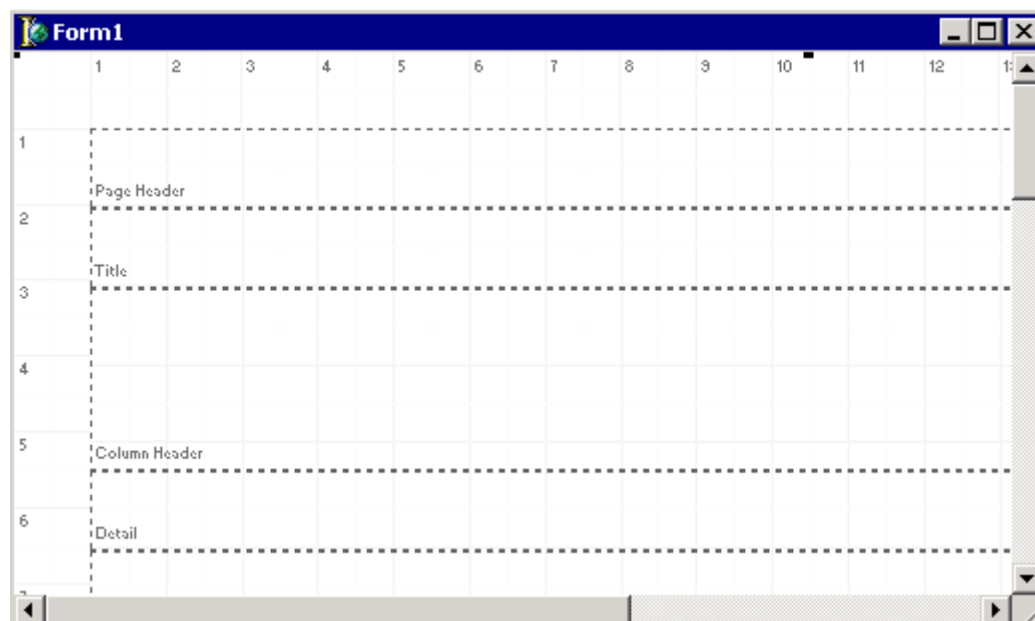


Рисунок 15.2.1 Вид компонента QuickRep с включенными областями

**DataSet** – здесь указывается набор данных (таблица) из которой отчёт будет брать данные.

**Font** – шрифт, который будет использоваться по умолчанию.

**Frame** – здесь ты указываешь параметры рамки.

**Options** – здесь тебе доступны три параметра. Если *FirstPageHeader* равно *true*, то заголовок печатается только на первой странице отчёта. Если *LastPageFooter* равен *true*, то нижний колонтитул печатается только на последней странице отчёта. Если установить свойство *Compression* в *true*, то отчёт будет сохраняться в сжатом виде.

**Page** – здесь тебе доступны все необходимые опции для контроля над бумагой отчёта. Ты можешь установить её размеры, отступы и ориентацию.

**PrinterSettings** – здесь находятся настройки принтера. С принтером мы уже работали, да и настройки практически не требуют пояснения.

**ReportTitle** – здесь находится заголовок печатаемого документа.

**ShowProgress** – если этот параметр равен *true* то во время печати тебе будет доступен индикатор хода выполнения печати.

**SnapToGrid** – нужно ли выравнивать компоненты по установленной сетке.

**Zoom** – масштаб отображения данных.

Если дважды щёлкнуть по компоненту QuickRep, то перед тобой откроется окно, в котором все эти настройки представлены в одном окне и в очень удобном виде. Они достаточно понятны и легко разобраны со всем самостоятельно (смотри рисунок 15.2.2). Я предпочитаю пользоваться именно этим окном, потому что здесь всегда можно увидеть будущий результат и если что, то все изменения можно отменить нажатием кнопки *Cancel*.

Рисунок 15.2.2 Окно настроек отчёта.

Пока теории хватит, давай посмотрим, как же действует отчётность. Открой телефонный справочник, созданный в прошлой главе. Добавь на панель кнопку печати, а обработчик его события *OnClick* мы напишем чуть позже. Сейчас мы будем создавать форму отчёта.

Создай новую форму (назовём её *ReportForm*) и сохрани в модуле *ReportFormUnit*. Сразу же подключи к этому модулю, модуль данных (*DataModule*), где у нас хранятся компоненты доступа к таблицам базы данных.

Брось на форму компонент *QuickRep*. Выдели этот компонент и в объектном инспекторе включи параметры *HasTitle* и *HasDetail* свойства *Bands*.

Теперь нужно в этих секциях расположить компоненты, которые будут отображать нужную нам информацию. На закладке *QReport* палитры компонентов доступны следующие компоненты, которые можно располагать в этих разделах:

**QRLabel** – надпись. Этот компонент похож на стандартный компонент *TLabel* и просто отображает нужные данные.

**QRDBText** – данные. Этот компонент тоже похож на *TLabel*, только он предназначен для отображения значения какого либо поля из базы данных. Тип поля базы данных должен быть совместим с текстом, т.е. может быть целым числом, строкой, датой, но не может быть картинкой или бинарными данными.

**QRSysData** – системная информация. Это опять копия *TLabel* только с возможностью отображать системную информацию – дату, время, номер страницы, номер строки в таблицы, общее количество страниц и т.д.

**QRMemo** – набор строк. Этот компонент уже похож на *TMemo* и способен отображать *Memo* данные из базы данных.

**QRShape** – компонент для создания обрамлений. Он чем то похож на стандартный *TShape*.

**QRImage** – картинка. Компонент схожий с *TImage*.

Теперь примемся за оформление отчёта. Выдели область заголовка (*Title*) и увеличь её размер где-то в два раза. В правый верхний угол области (именно области *Title*, а не компонента *QuickRep*) помести один компонент *QRSysData*. Выдели его и в свойстве дата выбери значение *qrsDateTime*. Теперь этот компонент будет отображать в правом, верхнем углу дату распечатки документа. Советую всегда это делать, чтобы ты сразу видел, какая версия документа была распечатана последней.

В центре области брось компонент *QRLabel*, увеличь шрифт в свойстве *Font* и напиши в свойстве *Caption* текст «*Распечатка строки из базы Телефонов*». Слева области можешь установить картинку *QRImage*, чтобы убедиться, что работа с ней ничем не отличается от работы с компонентом *TImage*.

Оцени результат!!! Мой результат ты можешь увидеть на рисунке 15.2.3.

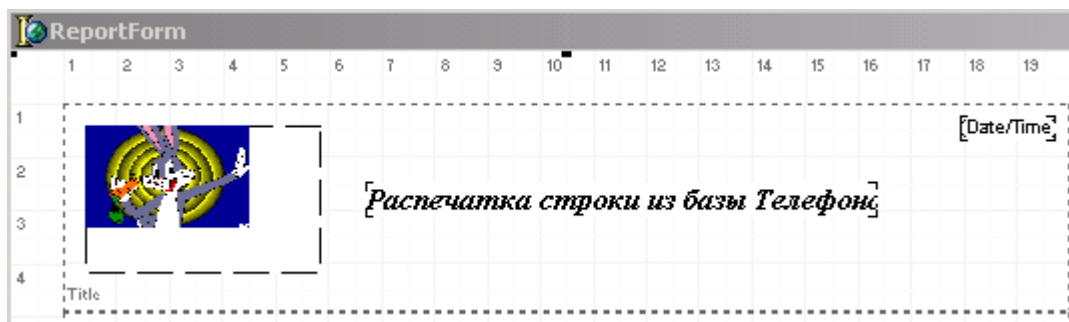


Рисунок 15.2.3 Вид заголовка нашего отчёта.

Теперь переходим к области Detail. Здесь давай выстроим с троку пять компонентов *QRLabel* и дадим им заголовки: Фамилия, Имя, e-mail, Город, Дата рождения. Под ними поставь пять компонентов *QRDBText*. У всех у них установи свойство *DataSet* в *DataModule1.BookTable*, а в свойстве *DataField* укажи соответствующие поля. У тебя должно получиться что-то похожее на рисунок 15.2.4.

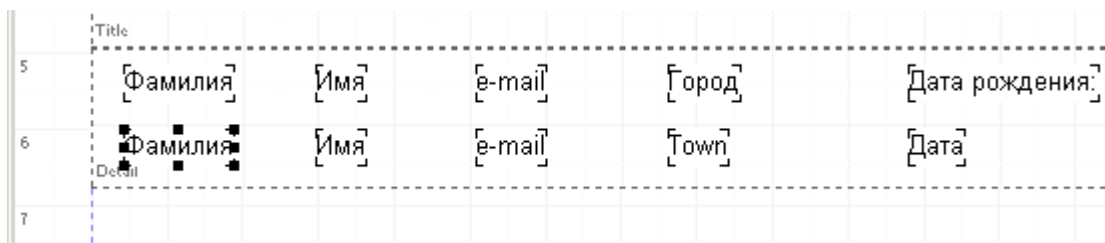


Рисунок 15.2.4 Вид блока Detail.

Теперь переходим в главный модуль и по нажатию кнопки печати пишем следующий код:

```
procedure TMainForm.PrintButtonClick(Sender: TObject);
begin
  ReportForm.QuickRep1.PreviewModal;
end;
```

В этом коде я вызываю метод *PreviewModal* компонента *QuickRep*. Этот метод модально показывает окно предварительного просмотра созданного нами документа.

Попробуй запустить программу, выделить какую-нибудь строку и нажать кнопку печати. Перед тобой откроется окно предварительного просмотра, как на рисунке 15.2.5.

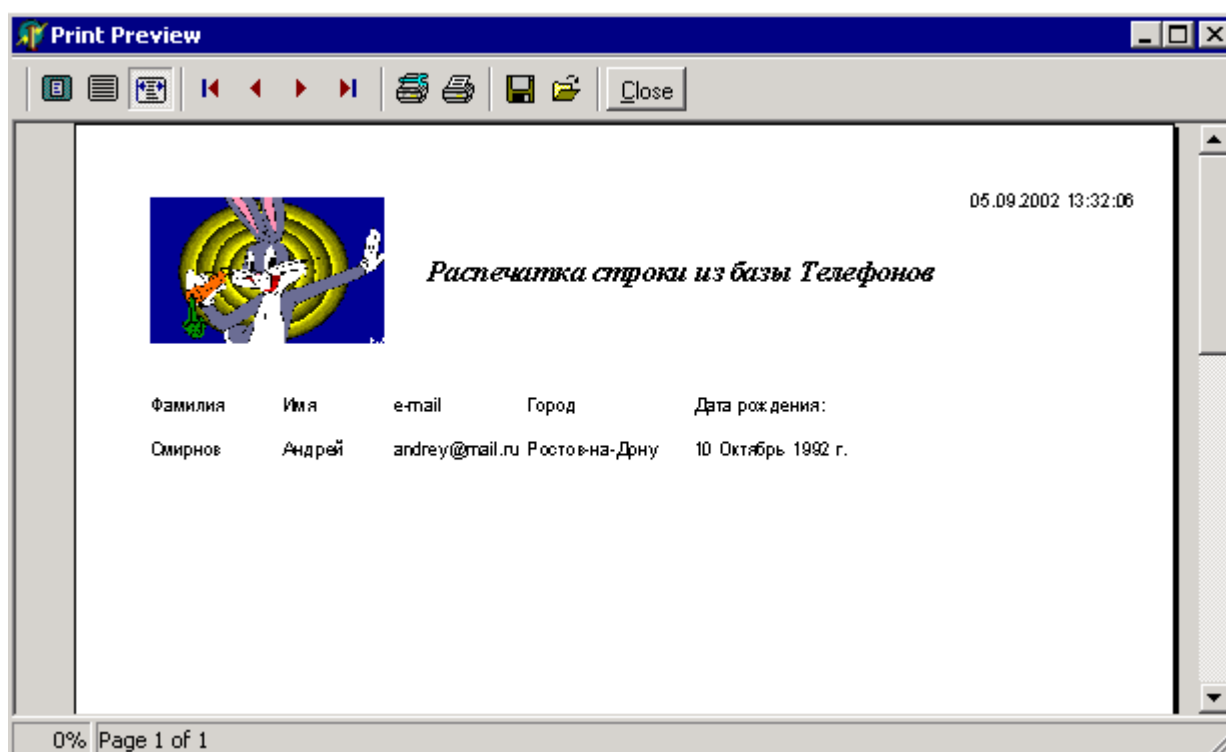



Рисунок 15.2.5 Окно предварительного просмотра.

В этом окне достаточно нажать кнопку печати, и документ будет распечатан на принтере.

 На компакт диске, в директории \Примеры\Глава 15\QuickRep ты можешь увидеть пример этой программы.

### 15.3 Печать таблиц с помощью Quick Reports

Как видишь, создание отчётности с помощью Quick Report задача не сложная. За каких-то 10 минут мы создали красивый и удобный отчёт, который легко распечатать. В нём мы подготавливали к печати одну запись из таблицы. Но что делать, если нужно распечатать все записи базы данных? Неужели нужно для каждой строки ставить отдельные компоненты? Ну конечно же нет, всё делается очень даже просто.

Открывай пример из прошлой части, сейчас мы его подкорректируем. Открой модуль *ReportForm*, где у нас находятся компоненты отчётности. Выдели *QuickRep1* и в свойстве *Bands* установи *true* у параметра *HasColumnHeader*. На форме появиться новый блок *Column Header*, который можно использовать для создания заголовков таблиц.

Теперь удерживая *Ctrl* обведи все компоненты *QRLabel* в блоке *DetailBand1*. Выбери из меню *Edit* главного меню Delphi пункт *Cut*, чтобы вырезать компоненты в буфер обмена. Теперь выдели блок *ColumnHeaderBand1* и выбери из меню *Edit* главного меню Delphi пункт *Paste*, чтобы вставить вырезанные компоненты из буфера обмена в блок *ColumnHeaderBand1*.

Снова выделяем компонент *QuickRep1* и в свойстве *DataSet* указываем таблицу *DataModule1.BookingTable*. Если сделать это, то компонент *QuickRep1* автоматически будет

перебирать все записи из этой таблицы и использовать их в компонентах, которые стоят в блоке *DetailBand1*. Как я уже говорил, этот блок предназначен для создания строк таблиц, так что теперь ты можешь увидеть это на практике.

Запусти программу и нажми кнопку печати. Перед тобой должно открыться окно, похожее на рисунок 15.3.1. Как видишь, блок заголовка и *ColumnHeaderBand1* распечатались только по одному разу, а блок *DetailBand1* был напечатан для каждой строки нашей таблицы. Блок заголовков колонок будет печататься по одному разу вверху каждой страницы, а после него будут идти строки таблиц.

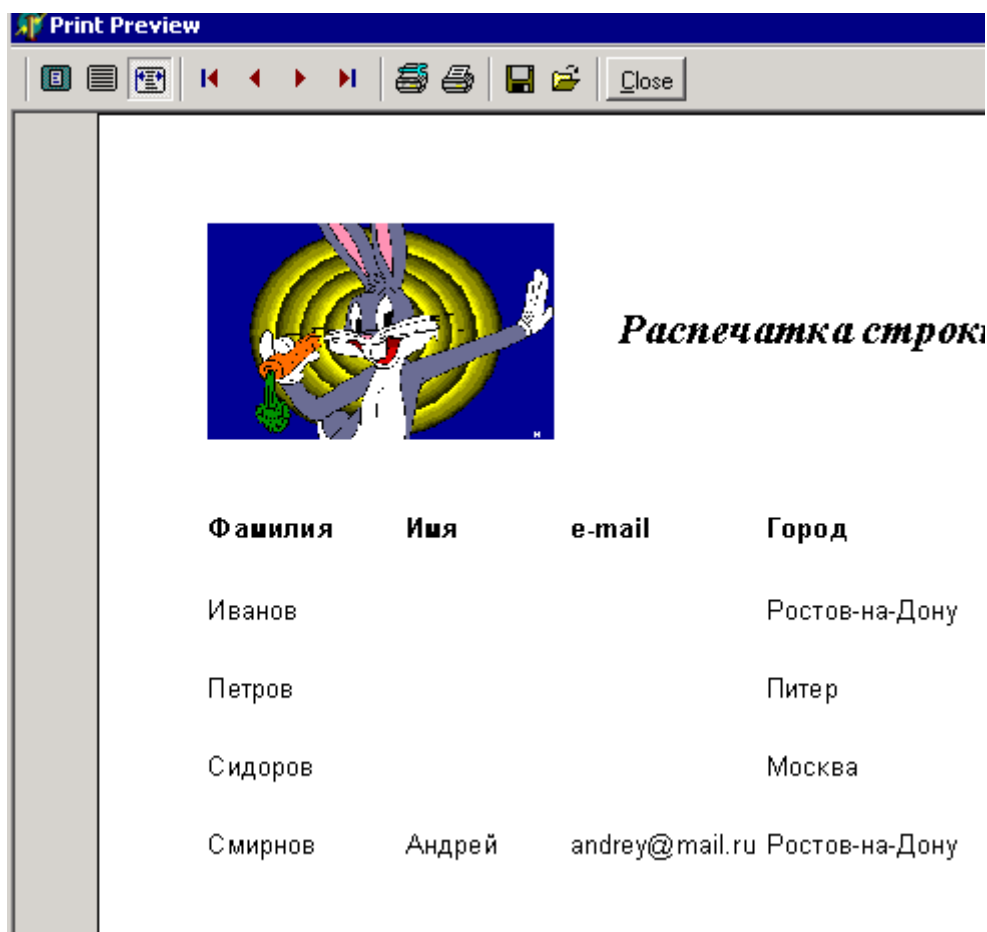



Рисунок 15.3.1 Окно предварительного просмотра.

В принципе, на этом можно было бы закончить разговор но мы же создаём таблицу, а для этого нужно нарисовать сетку. Вот тут можно заметить единственный недостаток отчётности с помощью Quick Report – каждую ячейку придётся рисовать отдельно, с помощью компонентов *QRShape*. В примере на диске я сделал рамку. Правда она получилась немного некрасивая, но у меня просто не было времени её поправлять. Если захочешь, то можешь украсить её на своё вкус, заодно и потренируешься работать с отчётами.

 На компакт диске, в директории \Примеры\Глава 15\QuickRep1 ты можешь увидеть пример этой программы.

## 15.4 Печать связанных таблиц.

У нас уже получился достаточно хороший отчёт, но теперь я хочу ещё более усложнить задачу. У нас же в телефонном справочнике используется связанная таблица и хотелось бы, чтобы в выходном документе печатались все телефоны, принадлежащие людям!!! Можно снова написать SQL запрос, который будет создавать сводную таблицу из двух и потом использовать этот запрос для отчёта, вместо таблиц, но это не выход.

Брось на нашу форму отчёта ещё один компонент – *QRSubDetail* с закладки *QReport*. Этот компонент предназначен для перебора данных относящихся к подчинённым таблицам. Установи у него следующие свойства:

*DataSet* – здесь установи *DataModule1.TelephonTable*, чтобы связать блок к с таблицей «Телефоны», которая у нас является подчинённой к основному справочнику.


*Master* – здесь нужно указать главный компонент с основными данными. Выбери в этом свойстве *QuickRep1*.

Теперь брось на этот блок компонент *QRLabel*, чтобы сделать надпись «Телефон». Справа от него брось компонент *QRDBText*. У него установи свойство *DataSet* в *DataModule1.TelephonTable*, а свойство *DataField* в «Телефон».

Фамилия	Имя	e-mail	Город
Иванов			Ростов-на-Дон
	Телефон:	2555555	
	Телефон:	4003535	
Петров			Питер
	Телефон:	2454545	
	Телефон:	902456550	

Рисунок 15.4.1 Окно предварительного просмотра.

Теперь можешь запускать программу. Нажимай кнопку печати, и если ты ввёл в свою базу данных хоть какие-нибудь телефоны, то у тебя должно получиться нечто похожее на рисунок 15.4.1.

 На компакт диске, в директории *\Примеры\Глава 15\QuickRep2* ты можешь увидеть пример этой программы.

## 15.5 Дополнительные прибомбасы.

Здесь я покажу некоторые прибомбасы, которые можно добавить в свои отчёты. Я покажу некоторые компоненты, которые не только украшают любой отчёт, но и делают его более удобным и мощным.

Первый компонент в моём обзоре – *QRExpr*. Этот компонент очень удобен для создания вычисляемых полей именно для отчёта. Вычисления будут происходить автоматически и практически не влияют на скорость работы самой программы.

У компонента есть свойство *Expression*. Если дважды щёлкнуть по нему, то перед тобой откроется окно, в котором можно создавать достаточно сложные расчёты. На рисунке 15.5.1 ты можешь увидеть это окно. Если нажать на кнопку *Database field*, то перед тобой откроется окно, в котором можно выбрать таблицу (таблица должна находиться на этой же форме) и поле, которое должно участвовать в расчёте.

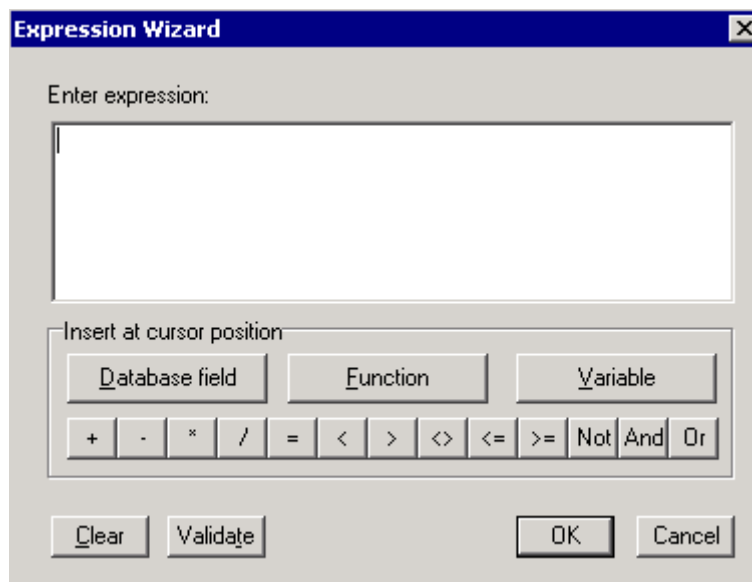


Рисунок 15.5.1 Окно предварительного просмотра.

Если нажать на кнопку *Function*, то ты увидишь громадный список доступных функций. Под кнопкой *Variable* спрятаны переменные, которые могут так же помочь в расчётах.

Всё это расписывать нет смысла, потому что здесь возможностей просто куча плюс ещё тысяча. По работе с Quick Report можно писать отдельную книгу, а моя задача показать тебе основные возможности и направить на путь истинный. Что-то я заговорил, как самый настоящий проповедник.

Теперь посмотрим на компонент *QRSysData*. Мы уже использовали его и в свойстве *Data* ставили параметр *qrsDateTime*, чтобы можно было видеть дату и время распечатки документа. Но тут есть ещё несколько полезных параметров:

- qrsDate* – дата распечатки.
- qrsDetailCount* – количество строк в таблице.
- qrsDetailNo* – номер строки в таблице.
- qrsPageNumber* – текущий номер страницы.
- qrsReportTitle* – заголовок отчёта.
- qrsTime* – время распечатки отчёта.

Теперь я хочу показать тебе, как можно ещё больше упростить создание отчёта. В Delphi есть мастер, который может облегчить создание отчётов. Для его вызова выбери из главного меню *File* пункт *New* и затем пункт *Other*. В появившемся окне перейди на закладку *Business* и выбери пункт *QuickReport Wizard* (рисунок 15.5.2).

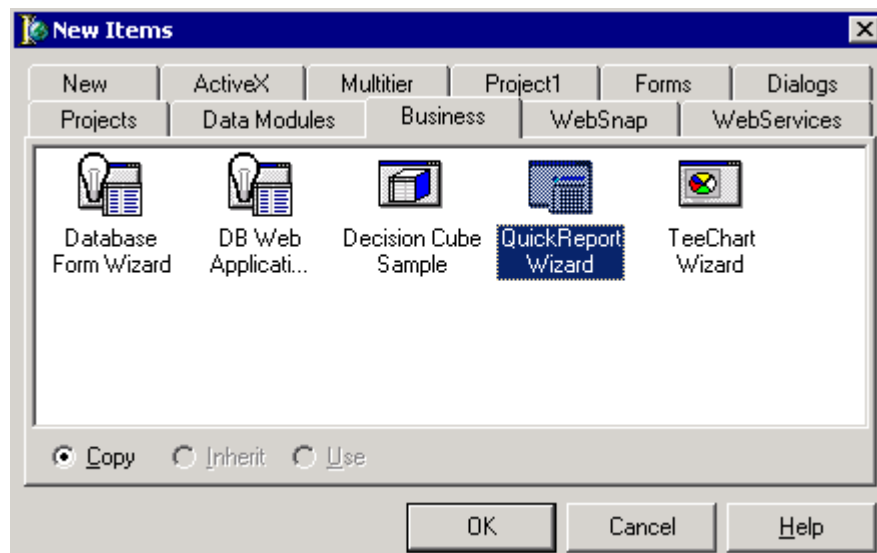


Рисунок 15.5.2 Окно предварительного просмотра.

В принципе, мастер достаточно прост, но при использовании ADO таблиц он абсолютно не подходит. Если ты будешь работать с старыми базами типа dbf или paradox, (об этом в следующей главе) то этот мастер принесёт свои плоды, но при работе с ADO, он практически бесполезен.

Пока что нам больше подходят шаблоны, которые есть в том же окне создания новой формы, но на закладке *Forms* (смотри рисунок 15.5.3). Здесь три шаблона (все они содержат имя QuickReport), содержащие всё необходимое для создания отчёта и хорошо документированные поля. Попробуй посмотреть их сам.

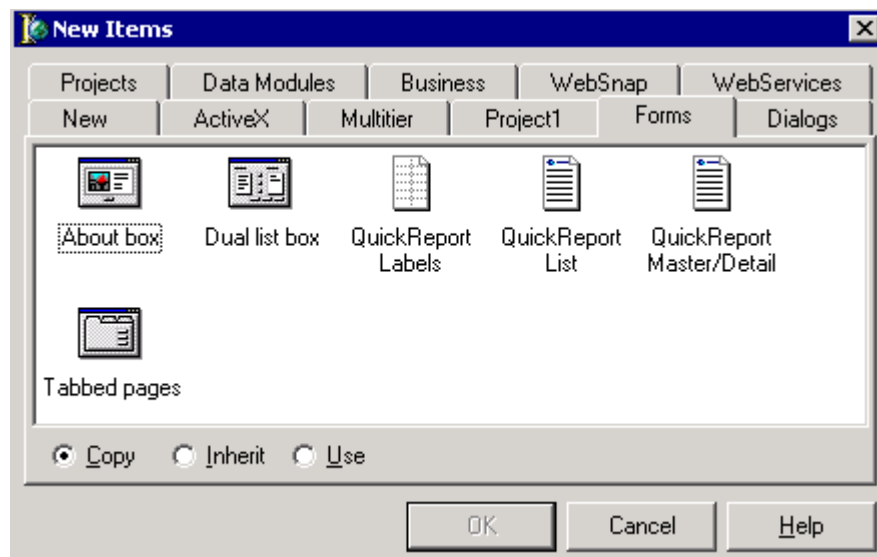


Рисунок 15.5.3 Окно шаблоном.