Министерство образования и науки Российской Федерации
Троицкий филиал государственного образовательного учреждения
высшего профессионального образования
«Челябинский государственный университет»

Лабораторный практикум по теме:

Работа с базами данных в среде Delphi

Методические указания

Троицк 2003г

Рекомендовано учебно-методической комиссией Троицкого филиала государственного образовательного учреждения высшего профессионального образования «Челябинский Государственный Университет»

Методические указания содержат задания для выполнения лабораторных работ по теме «Работа с базами данных в среде Delphi» курса «Практикум по ЭВМ». Материал изложен в логической форме «информация-задание-указание», что способствует приобретению практических навыков в работе программиста баз данных. В конце каждой темы есть вопросы для контроля и рекомендуемая литература.

Методические указания **п**редназначены для студентов третьего курса специальности 01 02 00 «Прикладная математика и информатика».

Составитель: преподаватель кафедры математики и информатики Е.В.Глазихина

Рецензент: Н.Х.Валеева, к.п.н., зав. отделением специальности 230105 «Программное обеспечение вычислительной техники и автоматизированных систем»

Введение

При создании программ, работающих с базами данных, в системе Delphi традиционно используется механизм Borland Database Engine (BDE). Этот механизм реализован в виде набора библиотек, которые обеспечивают для программы простой и удобный доступ к базам данных независимо от их архитектуры.

При использовании механизма BDE разработчик может не задумываться над тем, как его программа будет работать с базой данных на физическом уровне: локально, в файл-серверной, либо в клиент-серверной архитектуре. При переходе к использованию СУБД разных производителей программисту не потребуется менять исходный код своей программы, достаточно внести настройки в BDE.

В поставку BDE входит два набора драйверов:

- первый набор предназначен для файл-серверных СУБД dBASE, Paradox, FoxPro, Access и данных в текстовом формате;

- второй набор ориентирован на клиент-серверные СУБД InterBase, IBM DB2, Informix, Oracle, Sybase и Microsoft SQL Server.

Реализация в системе Delphi прослойки BDE позволяет не привязывать программу к конкретной СУБД.

В лабораторном практикуме «Работа с базами данных в среде Delphi» рассмотрены основные компоненты при создании баз данных в среде Delphi, способы создания баз данных как при использовании встроенных таблиц в среду Delphi, так и своих собственных. Рассматривается создание запросов и отчетов, к каждой новой теме есть вопросы для контроля. Предусмотрены задания для самостоятельного выполнения.

Лабораторная работа №1

Разработка программ для работы с готовой базой данных

Цель работы: Научиться создавать проекты, содержащие базы данных, на основе встроенных в среду программирования Delphi баз данных и содержащихся в них таблиц.

Задачи:

- знакомство с механизмом BDE;
- знакомство с основными компонентами создания базы данных (Ttable, Tfield, TfieldDef, TdataSource);
- разработка простейшей программы с использованием таблиц Items.db и Parts.db готовой базы данных DBDEMOS.

Информация

Для работы с СУБД в Delphi существуют следующие базовые классы и компоненты:

- **Класс TTable** (Таблица)

На базовом классе TDataSet (абстрактный набор данных) основан класс TBDEDataSet, который реализует функциональность BDE при работе с наборами данных. Его наследник – класс TDBDataSet - отвечает за связь с базой данных. Именно на его основе созданы компоненты, способные работать с реляционной информацией в виде таблиц, организованных в столбцы и строки. В частности, важнейший компонент TTable является наследником класса TDBDataSet.

- **Класс Поле записи** (TField)

Данный класс описывает конкретное поле записи, которое представлено в программе виртуально: TField служит как бы оболочкой для физического поля записи, дополняя его набором свойств и методов, требуемых разработчику.

В отличие от класса TFieldDef, описывающего физическое (реально

существующее) поле, на основе класса TField создаются псевдополя: вычисляемые поля, поля соответствия и др. Тип TField реально в программе не присутствует. Вместо него используется множество его наследников, соответствующих конкретным типам полей записи (например, TDataField для поля хранящего дату, TGraphikField для поля, хранящего графическое изображение, и т.д.). Вместе с тем большинство свойств описано в этом родительском классе.

- **Класс Описания поля записи** (TFieldDef)

Класс TFieldDef используется для описания физического поля таблицы базы данных. Как только новая таблица добавляется в модуль данных, для нее формируется описание всех полей. В дальнейшем можно сформировать описание виртуальных полей, например, выбрав в контекстном меню строки Fields в окне просмотра модуля данных пункт NewField (Создать поле). После этого работать в программе с классом TFieldDef станет невозможно из-за того что, в блоке связи с источником данных произойдет автоматическая замена полей класса TFieldDef на поля класса TField.

- **Компонент Источник данных** (TdataSource)

Компонент обеспечивает связь между таблицами и другими физическими наборами данных и элементами управления на форме. Этот компонент не требует сложных настроек. Отметим только такое его свойство, как State (тип TDataSetState),которое позволяет узнать, в каком состоянии находится указанный в свойстве DataSet набор данных (например, в режиме редактирования, добавления, фильтрации, и т.д.).

Доступ к базе данных в системе Delphi выполняется с использованием набора невизуальных компонентов работы с СУБД. Как правило, эти компоненты группируются в создаваемой программе в специальном модуле данных (TDataModule). Модуль данных представляет собой хранилище объектов, которое позволяет централизованно управлять их работой.

Задание 1. Создайте новый проект и добавьте в него модуль данных командой **File ▸ New ▸ Data Module** (Файл ▸ Создать ▸ Модуль данных). Чтобы из главной формы можно было получить доступ к содержимому модуля данных, в списке доступных модулей в файле Unit1.pas нужно указать его имя (Unit2):

```
unit Unit1;

interface

uses

Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
Unit2;
```

Информация

На панели модуля данных необходимо разместить компоненты, которые обеспечат доступ к нужным таблицам базы данных. В качестве примера воспользуемся таблицами баз данных, входящих в стандартные примеры Delphi. Возьмем базу DBDEMOS, в которой имеется множество таблиц с готовой информацией.

Задание 2. Поместите в модуль данных компонент TTable с панели компонентов BDE (он используется для доступа к таблице базы данных).

В Просмотрщике дерева объектов появляется дерево, описывающее логические связи между компонентами. Первоначально узлы Alias (псевдоним базы данных) и Tabel1 (объект класса TTable) выделяются знаком вопроса (объект определен не полностью и не может использоваться в программе).

Задание 3. Выделите в модуле данных компонент Tabel1. В Инспекторе объектов для свойства DatabaseName (Имя базы данных) выберите

значение DBDEMOS. После этого на левой панели модуля данных красная пометка с узла Alias будет снята.

Теперь требуется указать таблицу, связь с которой осуществляет объект Tabel1. Выберите в раскрывающемся списке свойства TabelName таблицу Items.db. Переименуйте таблицу Tabel1 в Items, чтобы в дальнейшем ориентироваться в названиях было проще. Чтобы можно было обращаться к этой таблице из программы, присвойте свойству Active значение true.

🗗**Задание 4.** Таким же образом добавьте в модуль данных таблицу Parts.db.

🗐**Информация** Таблица Items описывает заказы на детали, которые хранятся в таблице Parts. Их связь выполняется через поле PartNo таблицы Items, которое соответствует ключевому полю PartNo таблицы Parts.

🗗**Задание 5.** На основе таблицы создайте набор постоянных полей. Для этого выберите таблицу в модуле данных, выберите в контекстном меню пункт FieldsEditor (Редактор полей). В появившемся списке постоянных полей (первоначально пустом) откройте контекстное меню. Добавьте в список все поля таблицы с помощью команды Add all fields (Добавить все поля).

🗐**Информация**

Экранные элементы для работы с СУБД во многом напоминают обычные элементы управления Windows. Они отличаются тем, что предназначены для редактирования полей таблиц а не переменных программы. В принципе, их можно напрямую подключать к компонентам TTable, однако в системе Delphi реализован более гибкий подход – создан компонент промежуточного уровня TDataSource (Источник данных).

В нашем случае каждый источник данных после размещения в модуле данных связывается с конкретной таблицей с помощью свойства DataSet.

Задание 6. В модуль данных поместите два компонента TdataSource с панели компонентов Data Access (Рис. 1.1), свяжите каждый из них со своей таблицей и дайте им подходящие названия (например, SourceItems и SourceParts).

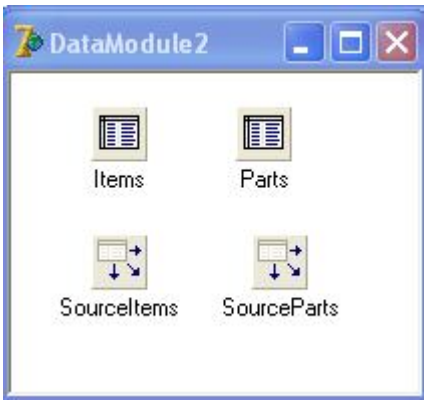


Рис. 1.1. Окно модуля данных

Информация

Чтобы получить возможность редактировать, добавлять или удалять записи таблицы, достаточно разместить на форме компонент TDBGrid с панели DataControls (Элементы управления данными).

Задание 7. Поместите на форму компонент TDBGrid. В его свойстве DataSource укажите нужный источник данных – DataModule2.SourceItems.

Задание 8. Для упрощения навигации по таблице разместите на форме под компонентом TDBGrid компонент TDBNavigator (Рис. 1.2). Значение

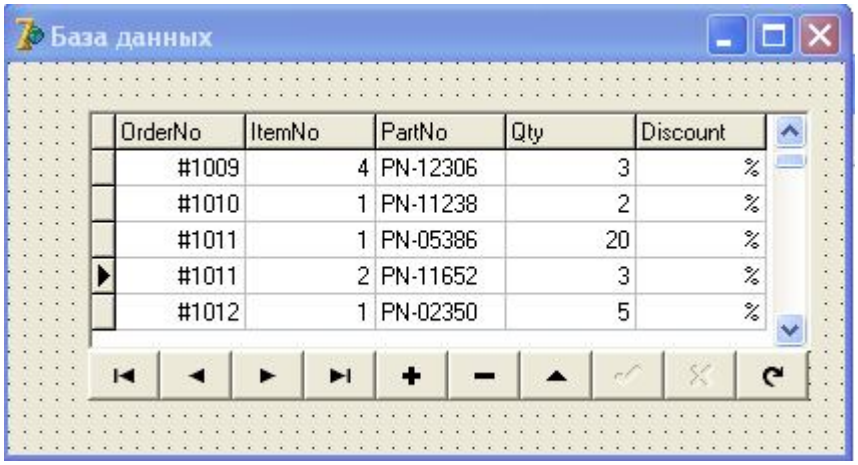


Рис. 1.2. Окно формы

свойств DataSource этих компонентов должны совпадать.

Информация

Для пользователей базы данных предпочтительнее видеть на экране комплексную информацию. Например, при воспроизведении записей таблицы Items неплохо вместо числового поля PartNo отображать текстовое название соответствующей детали из таблицы Parts. Реализовать подобные требования можно с помощью так называемых полей соответствия, которые создаются в редакторе полей (Fields Editor) для физической таблицы Items на примере связи с таблицей Parts.

Задание 9. Добавьте с помощью редактора полей в таблицу Items новое поле PartName в конце списка: выберите команду New Field (Новое поле) из контекстного меню. В появившемся диалоговом окне в текстовых полях укажите следующие значения (Рис. 1.3):

- в поле Name – имя нового поля (PartName);

- в поле Type – тип представляемого поля (String);

- на панели Field Type включите переключатель Lookup (поле соответствия).

В появившейся панели Lookup definition (Определение поля соответствия):

- в раскрывающемся списке KeyFields (Ключевые поля) указывается поле PartNo, связывающее таблицы между собой;

- в раскрывающемся списке DataSet (Связываемый набор данных) выбирается таблица Parts;

- в раскрывающемся списке Lookup Keys (Поля соответствия) указывается поле PartNo таблицы Parts, связанное с полем PartNo текущей таблицы Items;

- в раскрывающемся списке Result Field (Поле результата) указывается поле Description таблицы Parts, подставляемое вместо поля PartNo таблицы Items.

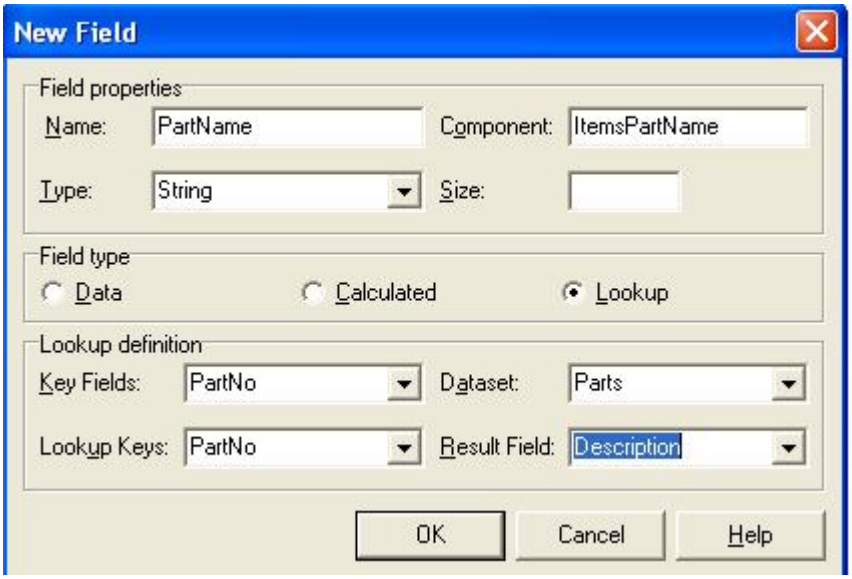


Рис. 1.3. Добавление нового поля в таблицу Items.

Задание 10. Откорректируйте компонент DBGrid1. В нем в дополнение к существующему полю PartNo появилось поле соответствия PartName, поэтому поле PartNo можно сделать невидимым путем настройки свойства Columns, явно задавая только те поля, которые нужно отобразить.

Информация

Обратите внимание на работу полей соответствия. Если выделить такое поле, то в его правой части появится кнопка со стрелкой. Щелкнув по ней, мы получим список всех значений, связанных с данным полем. Новые значения можно выбирать из этого списка. Например, таким способом можно указать другое название детали. При этом автоматически произойдет изменение

связанного ключевого поля PartNo, которое получит новое значение, соответствующее значению ключа для выбранного названия.

Нередко возникают ситуации, когда одной записи некоторой таблицы соответствует по ключевому полю сразу несколько записей другой таблицы. Например, один заказ может состоять из нескольких деталей, и их набор желательно видеть целиком в отдельном списке. Допустим, что для каждой записи таблицы Parts требуется отображать все записи таблицы Items (заказы), связанные с ней по ключевому полю.

🗗 **Задание 11.** Добавьте новый компонент для отображения содержимого набора Parts (DBGrid2). Разместите на форме рядом с таблицей компонент TDBLookupListBox (Список полей соответствия) и определите следующие свойства:

- свойство DataSource должно указывать на источник исходных данных DataModule2.SourceParts;

- свойство DataField должно указывать на поле PartNo, по которому выполняется связь с таблицей Items;

- свойство ListSource должно указывать на источник данных DataModule2.SourceItems, откуда извлекаются значения для заполнения списка;

- свойство KeyField определяет ключевое поле этого источника, привязываемое к полю, указываемому в свойстве PartNo;

- свойство ListField указывает поле, из которого будут браться значения для списка OrderNo.

Информация

В данной работе был рассмотрен визуальный способ работы с таблицами, основанный на использовании компонента TDBGrid. Этот способ не предъявляет повышенных требований к правильности ввода. В более серьезных приложениях желательно применять второй подход – программный, когда значения конкретных полей задаются с помощью элементов управления Windows. Оба этих подхода широко распространены.

Вопросы для контроля:

1. Какие вы знаете базовые классы и компоненты для работы с СУБД в среде Delphi?
2. Что представляет собой модуль данных?
3. Для чего предназначены компоненты TTable, TDataSource, TDBGrid, TDBNavigator?
4. С помощью какого свойства источник данных связывается с таблицей?
5. Что представляют собой поля соответствия?
6. Какие существуют способы работы с таблицами?

Литература

1. Фаронов В.В. Программирование баз данных в Delphi 6. Учебный курс. – СПб: Питер, 2002. – 352 с.: ил., с.142-170.
2. Базы данных: модели, разработка, реализация/ Т.С. Карпова. – СПб.: Питер, 2001. – 304 с.: ил., с. 47-65.
3. Информатика: Учебник. – 3-е перераб. изд./ Под ред. Проф. Н.В. Макаровой. – М.: Финансы и статистика, 1999. – 768 с.: ил., с. 580-886.

Лабораторная работа №2

Создание запросов

Цель работы: Научиться создавать запросы к базам данных в среде программирования Borland Delphi.

Задачи:

- знакомство с компонентом TQuery с панели BDE;
- создание запросов к базе данных, созданной в лабораторной работе №1, при помощи окна построителя запросов SQL Builder;
- применение функций при построении запросов.

Информация

Для создания запросов в Delphi используется компонент TQuery с панели BDE. Он позволяет очень гибко, визуально или программно, определить условие отбора записей из нескольких таблиц, а работать с итоговым набором собранных записей можно обычным способом, как с компонентом TTable.

Задание 1. Разместите на панели в модуле данных компоненты TQuery и DataSource. В свойстве DataSet источника данных (назовем его QuerySource) укажем значение Query1 (имя объекта-запроса), а в самом объекте Query1 зададим название базы данных (свойство DatabaseName должно получить значение DBDEMOS).

Задание 2. Выделите компонент Query1 и в его контекстном меню выберите пункт SQL Builder (Построитель запросов) (Рис. 2.1).

Первоначально экран построителя пуст. Название текущей базы данных (DBDEMOS) указывается в раскрывающемся списке Database (База данных) в правом верхнем углу окна. Запрос строится на основе двух таблиц: Items и Parts. Добавление таблиц к запросу выполняется выбором пунктов Items.DB и Parts.DB в раскрывающемся списке Table (Таблица).

🗗**Задание 3.** Установите связи между таблицами. Для этого найдите в таблице Items поле PartNo и протяните от него линию к таблице Parts. Флажками можно пометить поля, которые вы хотите получить в результате запроса.

🗐**Информация**

Раскрывающийся список на вкладке Joins (Соединения) содержит список связей, установленных между таблицами. Сейчас в нем выбрана только что созданная связь (Items < - > Parts), но таких связей может быть сколько угодно.

🗗**Задание 4.** Создайте условия отбора записей. Это условие задается на вкладке Criteria (Условия отбора). В столбцах Field or Value (Поле или константа) указываются сравниваемые поля или константы, в столбце Compare (Сравнить) – оператор сравнения.

🗗**Задание 5.** Укажите, что надо отобрать только детали с номером 1313 (то есть, значение поля PartNo должно быть равно 1313). Для этого щелкните на ячейке левого столбца, откройте раскрывающийся список и выберите значение Items. В правом столбце введите значение 1313, а содержимое центрального столбца (символ =, условие равенства) оставьте без изменений.

🗗**Задание 6.** В следующей строке выберите Parts.Cost. в списке доступны поля обеих таблиц, так как мы установили между ними связь. В столбце, определяющем операцию, выберите значение BETWEEN («между»). Это ключевое слово SQL говорит о том, что значение выбранного поля должно находиться в диапазоне, который указывается в двух последующих за BETWEEN полях (это будут, для примера, значения 100 и 150).

Информация

Сейчас запись соответствует запросу, если одновременно истинны все условия, заданные на вкладке Criteria – они связаны логической операцией AND (И). в раскрывающемся списке на этой вкладке такая операция обозначается пунктом All (Все). Если его изменить на ANY (Хотя бы одно), то слово AND слева от каждого условия заменится на слово OR (ИЛИ). Другое возможное значение – NONE (Ни одно) – представляет собой операцию отрицания. Если оно задано, то отбираются те записи, для которых указанное условие не истинно, а ложно. Наконец, значение NOT ALL (Не все) требует, чтобы хотя бы одно из списка условий было ложным.

Задание 7. Пометьте флажком те поля, которые необходимо показать пользователю. Например, названия деталей и номера заказов (поле OrderNo таблицы Items и поле Description таблицы Parts).

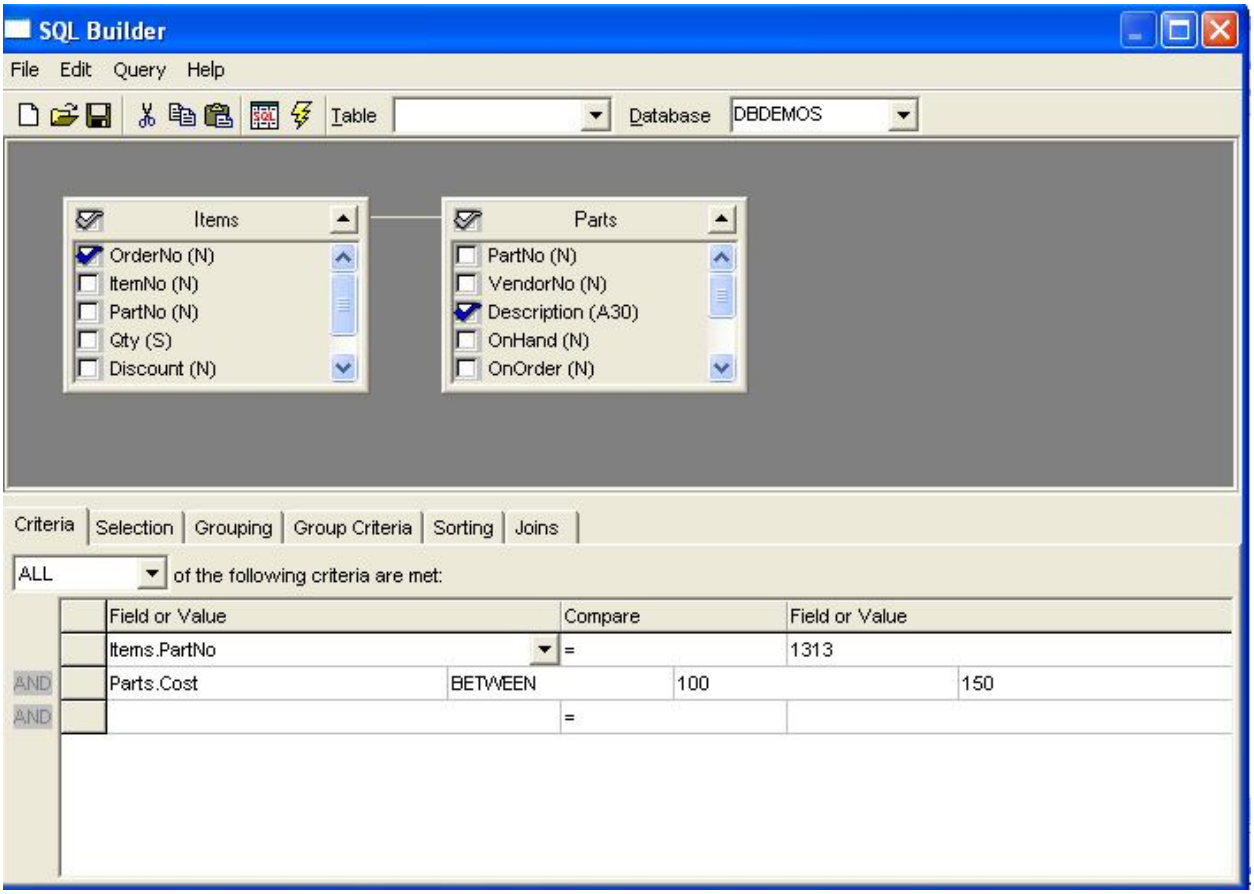
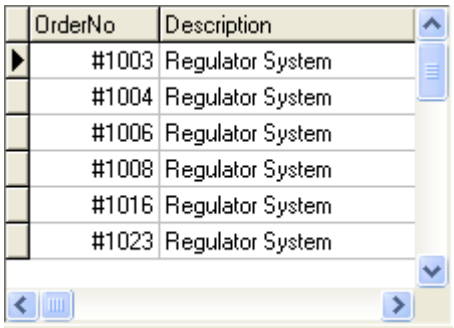


Рис. 2.1. Окно Построителя запросов.

🗗**Задание 8.** Сохраните запрос. Для этого щелкните на кнопке Save the current query (Сохранить текущий запрос) и укажите в диалоговом окне имя файла, в который записывается автоматически сгенерированный текст на языке SQL, соответствующий спроектированному запросу. Этот файл можно просмотреть.

🗗**Задание 9.** Отобразите содержимое запроса на экране. Для этого в редакторе полей для объекта Query1 добавьте два поля, подготовленных в запросе. Затем разместите на форме еще один компонент TDBGrid и укажите в качестве источника данных значение DataModule2.QuerySource. Добавьте в этот элемент два столбца, и таблица на форме отобразит набор записей, удовлетворяющих заданному условию (Рис. 2.2).



OrderNo	Description
#1003	Regulator System
#1004	Regulator System
#1006	Regulator System
#1008	Regulator System
#1016	Regulator System
#1023	Regulator System

Рис. 2.2. Результат выполнения запроса.

🗗**Задание 10.** Для каждой детали из таблицы Parts определите самый большой заказ из таблицы Items (по числу деталей – полю Qty). Для этого добавьте новое итоговое поле. Оно будет содержать максимальное значение поля Qty в записях таблицы Items, у которых значение поля PartNo совпадает с текущим значением таблицы Parts. Выполняется это следующим образом:

1. Сначала на вкладке Selection (Выбор) в столбце Field выберите поле Items.Qty. В столбце OutputName (Имя при выводе) его можно назвать MAX.
2. Открыв контекстное меню этой строки, выберите пункт Summary (Итог). Он определяет, что данное поле будет не простым, а

итоговым. При этом в список добавляется новый столбец Summary, содержащий поле только в текущей записи.

3. В этом столбце выберите в раскрывающемся списке нужную итоговую функцию.
4. На вкладке Grouping (Группировка) надо выбрать оба доступных поля: Items.OrderNo и Parts.Description – и переместить их на панель Grouped On (Сгруппированы). Так определяется подмножество полей, к которому будет применена итоговая функция.
5. Запустите запрос. В результирующей таблице появилось новое поле MAX, в котором для каждой записи заказа отображается максимальное число «привязанных» к ней деталей из таблицы Items.

Задание 11. Отберите только те записи, у которых значение поля Qty не менее пяти. Для этого на вкладке Group Criteria (Групповой критерий отбора) надо выполнить следующие действия:

1. В столбце Summary укажите итоговую функцию MAX.
2. В столбце Field or Value выберите поле Items.Qty.
3. В столбце Operator выберите логическую операцию >=.
4. В последнем столбце Field or Value укажите значение 5.
5. Запустите запрос на выполнение.

Вопросы для контроля:

1. Какой компонент в Delphi служит для создания запросов? На какой странице он находится?
2. Что представляет собой Построитель запросов?
3. Как в Построителе запросов можно просмотреть, какие существуют связи между таблицами?
4. Как указываются условия отбора записей?

5. Как сохранить запрос?
6. Как отобразить выполнение запроса на экране?

Литература

1. Фаронов В.В. Программирование баз данных в Delphi 6. Учебный курс. – СПб: Питер, 2002. – 352 с.: ил., с.172-183.

2. Базы данных: модели, разработка, реализация/ Т.С. Карпова. – СПб.: Питер, 2001. – 304 с.: ил., с. 66-103.

Лабораторная работа №3

Создание отчетов

Цель работы: Научиться создавать отчеты к базам данных в среде программирования Borland Delphi.

Задачи:

- знакомство с компонентом TQuickRep с панели QReport;
- создание отчета к базе данных, созданной в лабораторной работе №1;
- знакомство со вспомогательными компонентами при подготовке отчетов (TQRBand, TQRDBText, TQRLabel).

Информация

В системе Delphi отчет – это виртуальный образ бумажного листа, который в дальнейшем без изменений воспроизводится на принтере. Соответствующий компонент **TQuickRep** с панели **QReport** (Отчет) является основой такого листа (или группы листов). Этот компонент обладает множеством характеристик и позволяет детально настроиться на возможности конкретного принтера. Для отображения конкретных элементов данных служат другие компоненты панели **QReport**.

Рассмотрим пример создания простого отчета, основанного на таблице Parts.

🗗 **Задание 1.** Подготовьте отчет:

1. Разместите на форме компонент **TTable**, настройте его соответствующим образом и сделайте его активным.
2. Поместите на форму кнопку Button1 и базовый отчет **QuickRep1**. Свяжите его с таблицей **Parts**.
3. Установите на этот компонент полосу **TQRBand**, задав в ее свойстве **BandType** значение **rbDetail**, что обеспечивает возможность вывода последовательности значений полей.

Примечание. Компонент **TQRBand** предназначен для расположения и группирования в отчете конкретных элементов оформления. Содержит свойства для привлекательного оформления соответствующего раздела.

4. На полосу добавьте компонент **TQRDBText**, задав значение свойства **DataSet** равным **Partrs**, а значение свойства **DataField** равным названию поля **Description**.

Примечание. Компонент **TQRDBText** используется только для вывода значений полей наборов данных.

5. Украсьте отчет заголовком, названиями полей.

Примечание. Используйте компонент TQRLabel. Он предназначение для создания в отчете всевозможных текстовых надписей.

6. Запишите реакцию на щелчок на кнопке Button1:

```
procedure TForm1.Button1Click (Sender: TObject);
begin
    QuickRep1.Preview;
    QuickRep1.Print;
end;
```

Примечание. Сначала появится окно предварительного просмотра (из которого уже можно выполнить печать отчета), а затем выполнится его печать в принудительном порядке (см. Рис.3.1).

Вопросы для контроля:

1. Что называется отчетом?
2. На какой вкладке палитры компонентов находятся компоненты для создания отчета?
3. Какой компонент является базовым при создании отчетов? Какие его свойства вы знаете?
4. Для чего предназначена полоса отчета? Какие свойства этого компонента необходимо установить?
5. Какие еще компоненты, предназначенные для работы с отчетами вы знаете?

Литература

1. Фаронов В.В. Программирование баз данных в Delphi 6. Учебный курс. – СПб: Питер, 2002. – 352 с.: ил., с.183-188.

2. Delphi 5. Учебный курс. – М.: «Нолидж», издатель Молгачева С.В., 2001. – 608 с.: ил., с.422-430.

Лабораторная работа №4

Работа с готовой базой данных

(для самостоятельного выполнения)

Цель работы: Самостоятельное создание базы данных на основе готовых таблиц и встроенных баз данных.

Задачи:

- создание базы данных на основании навыков, полученных в работах 1-3;
- помещение необходимых компонент и описание реакций на события по заданию.

🗗**Задание.** Разработайте программу для вывода и редактирования записей готовой базы данных:

1. Создайте новый проект New Application.

2. Разместите на форме компоненты TTable и TDataSource и установите свойства компонент, используя предоставляемые средой имена для таблицы Table1 (custoly.db из базы данных DBDEMOS) и для компонента, обеспечивающего связь набора данных и компонент для его визуального отображения – DataSource1.

3. На форму внесите компоненту управления DBNavigator:

4. Создайте визуальную компоненту DBGRID для отображения всех полей таблицы из выбранной базы данных, установите необходимые свойства.

5. Добавьте на форму все компоненты, которые обеспечат вывод полей данных таблицы (DBMemo, DBText, DBEdit, DBCombobox, Label1, Label2, Label3), и впишите в предлагаемый формат необходимые свойства для установки их значения.

На Рис. 4.1.изображена полученная форма.

6. Напишите обработчик событий при открытии формы. При запуске приложения на исполнение содержимое полей данных таблиц сделайте невидимыми.

7. Для кнопки типа BitBtn создайте обработчик событий для вывода всех полей таблицы просматриваемой базы данных.

8.Запустите приложение и просмотрите таблицу стран и континентов. Получите

одну новую строку в таблице.

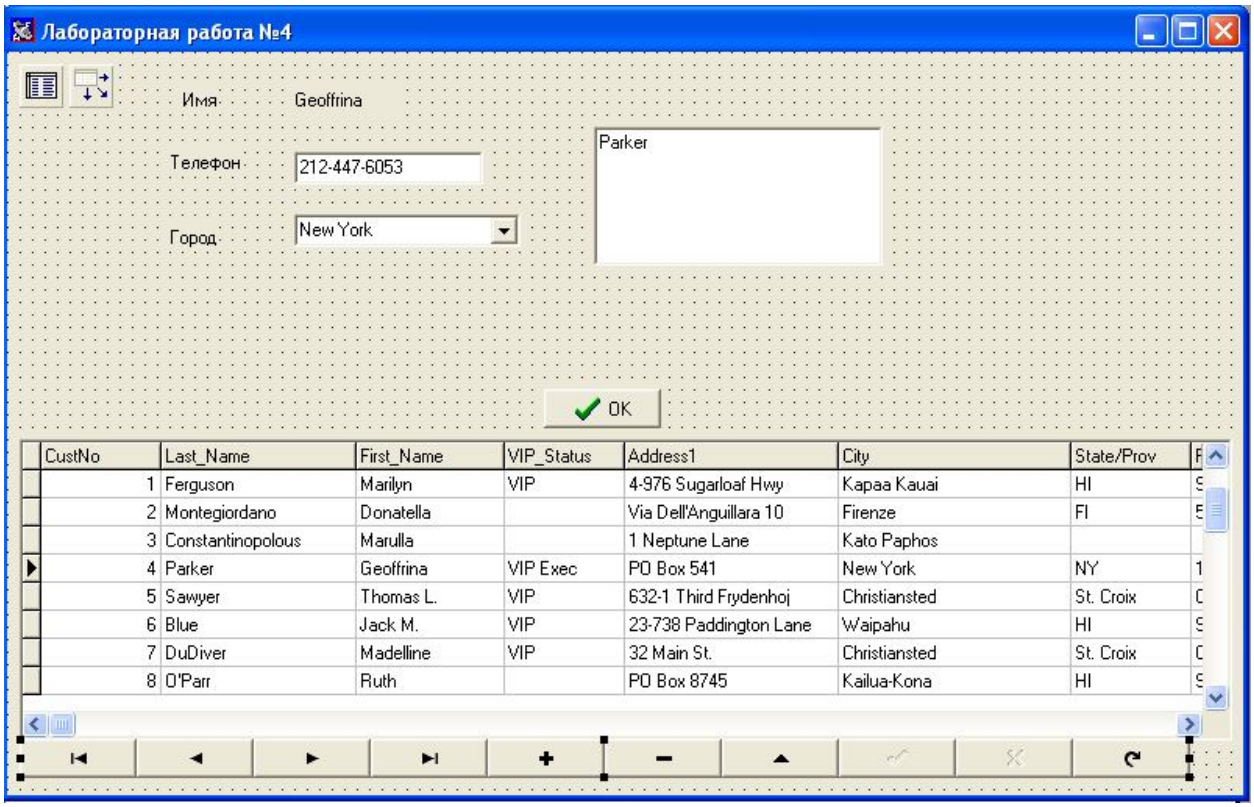


Рис. 4.1. Форма подготовленной базы данных.

Литература

1. Фаронов В.В. Программирование баз данных в Delphi 6. Учебный курс. – СПб: Питер, 2002. – 352 с.: ил., с.142-182.
2. Delphi 5. Учебный курс. – М.: «Нолидж», издатель Молгачева С.В., 2001. – 608 с.: ил., с.400-450.

Лабораторная работа №5

Создание базы данных

Цель работы: Научиться создавать собственную базу данных в среде программирования Borland Delphi.

Задачи:

- знакомство с альтернативой Database Desktop;
- создание собственного псевдонима базы данных;
- создание таблиц и связей между ними, запросов, отчетов.

Задание. Создайте базу данных «Студенты», содержащую следующие поля: номер п/п, ФИО, адрес, телефон, дата поступления.

Указание:

2. В окне Менеджера программ через меню Tools найдите альтернативу Database Desktop.
3. В появившемся окне Database Desktop выберите команду Tools ▸ Alias Manager. Окно этой команды предоставляет возможность выбора псевдонима создаваемой базы данных (Рис. 5.1).
4. Введите псевдоним STUD. Драйвер базы данных имеет уже предопределенное имя – STANDARD. Третье окно позволяет создать путь к базе данных (Рис. 5.1.).
5. Подтвердите создание псевдонима. В ответ система должна запросить подтверждение об изменении общедоступных псевдонимов. Для установок в BDE конфигурационный файл называется IDAPI.CFG. На запрос о факте изменения отвечайте OK.
6. В окне Database Desktop выберите команду File ▸ New ▸ Table. На Рис. 5.2 приведен формат создаваемой таблицы. Сохраните таблицу в псевдониме STUD под названием INF_STUD.
7. В окне Form1 расположите указанные на Рис. 5.3 компоненты. Компоненту для вывода поля таблицы Adress выберите DBRichEdit1. Установите свойства компонентов в соответствии с форматом таблицы базы данных.
8. С помощью кнопок типа BitBtn создайте обработчики событий для прочтения созданной базы данных и выхода из приложения. Заполните

таблицу базы данных конкретным содержанием.

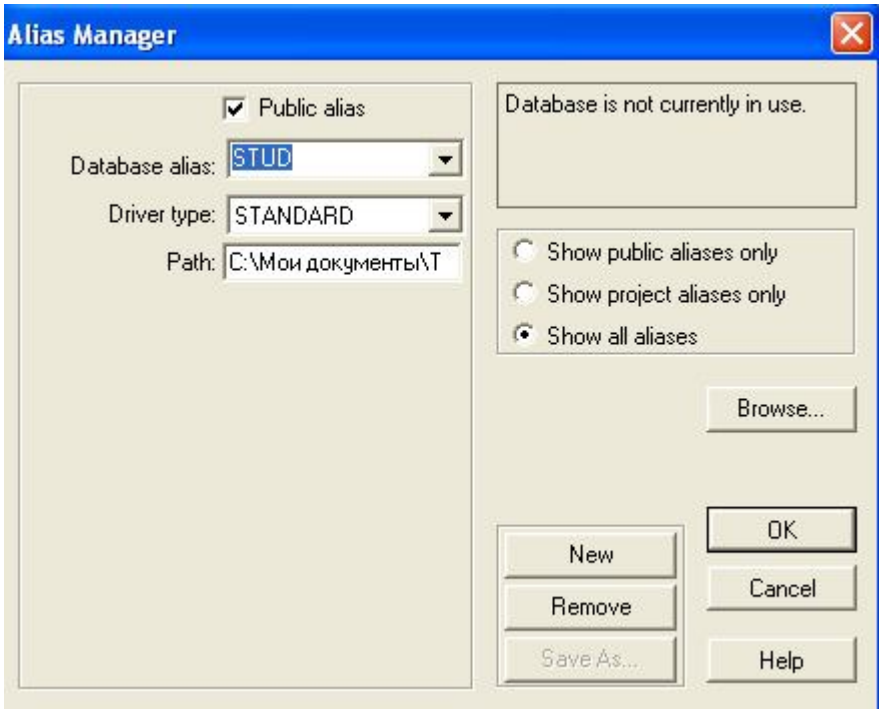


Рис. 5.1. Окно Alias Manager.

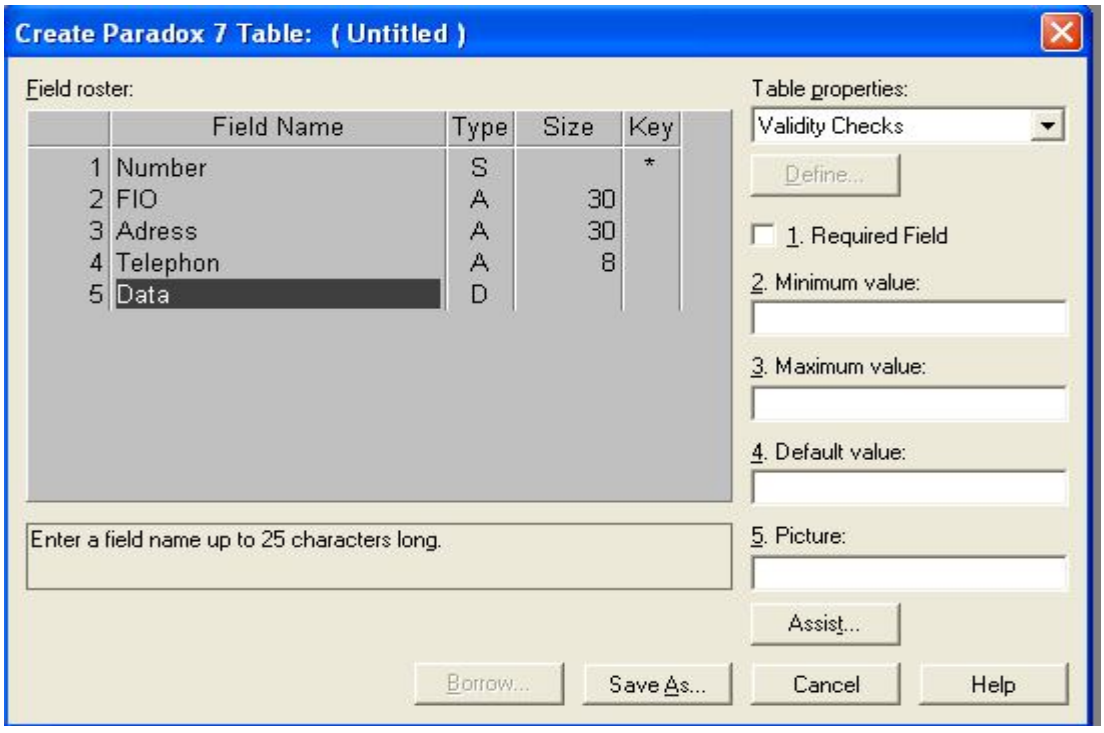


Рис. 5.2. Создание таблицы INF_STUD.

9. Создайте обработчик событий для поиска студентов по введенной дате (поступивших в один день).

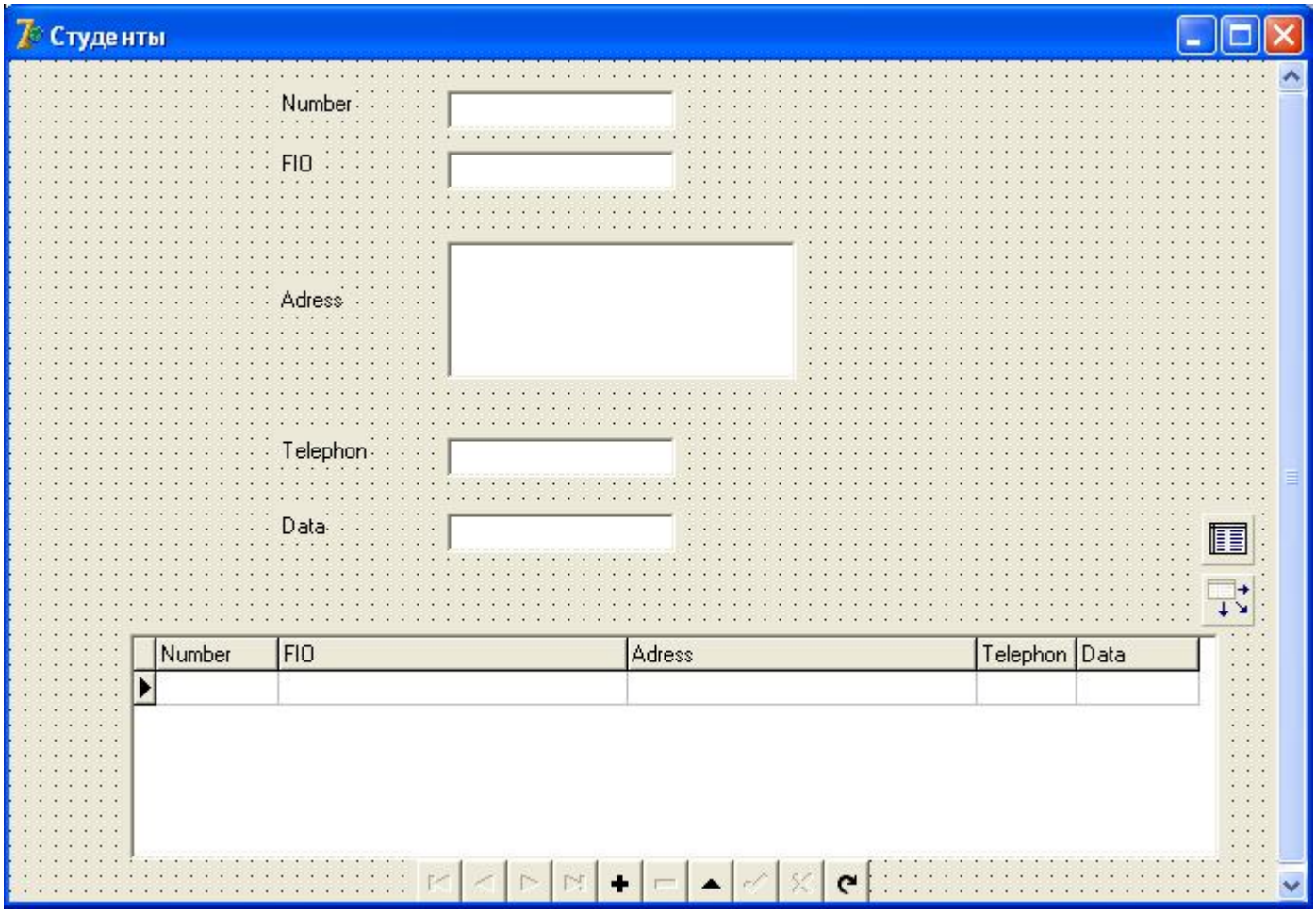


Рис. 5.3. Окно формы.

Литература

1. Базы данных: модели, разработка, реализация/ Т.С. Карпова. – СПб.: Питер, 2001. – 304 с.: ил., с. 104-120.

2. Коннолли Томас, Бегг Каролин, Страчан Анна. Базы данных: проектирование, реализация и сопровождение. Теория и практика, 2-е издание.: Пер. с англ.: Уч. пособие. – М.: Издательский дом «Вильямс», 2000. – 1120 с.: ил., с. 344-393.

3. Гэри Хансен, Джеймс Хансен. Базы данных: разработка и управление: Пер. с англ. – М.: ЗАО «Издательство БИНОМ», 2000. – 704 с.: ИЛ., С. 119-189.

Содержание

Лабораторный практикум по теме

“Работа с базами данных в среде Delphi”

Составитель: Елена Васильевна Глазихина

Редактор

Подписано в печать 2003

Формат 60Х 841/16 Бумага типографская №2

Печать офсетная. Усл. печ. л…Уч.-изд. л.

Тираж 50 экз. Заказ…

Троицкий филиал ГОУВПО “ЧелГУ”

457100 Челябинская обл., г. Троицк, ул. Разина,9

Типография ИП Пименов Д.С.

Свидетельство № А6922 от 10.07.1998г.