

Документ подписан простой электронной подписью  
Информация о владельце:  
ФИО: Баламирзоев Назим Лиодинович  
Должность: И.о. ректора  
Дата подписания: 19.08.2023 22:55:06  
Уникальный программный ключ:  
2a04bb882d7edb7f479cb266eb4aaaedebeea849

Министерство науки и высшего образования РФ

ФГБОУ ВО

"Дагестанский государственный технический университет"

Кафедра УиИТСиВТ

УЧЕБНО-МЕТОДИЧЕСКИЕ УКАЗАНИЯ

К ВЫПОЛНЕНИЮ ЛАБОРАТОРНЫХ РАБОТ ПО ДИСЦИПЛИНЕ "БАЗЫ ДАННЫХ" ДЛЯ  
БАКАЛАВРОВ НАПРАВЛЕНИЯ ПОДГОТОВКИ 09.03.01 - "ИНФОРМАТИКА И  
ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА"

УДК 681.3.068.

Учебно-методические указания к лабораторной работе по дисциплине «Базы данных» для бакалавров направления полготовки 09.03.01 - "Информатика и вычислительная техника".\Махачкала, ИПЦ ДГТУ,2020.-29 с.

В учебно-методических указаниях приведены основные понятия СУБД, описано прикладное приложение MySQL-Front для работы с БД MySQL и перечислены компоненты, применяемые при работе с базами данных в среде Delphi. Рассмотрена технология разработки приложений с базой данных MySQL. Содержатся задания для самостоятельной разработки баз данных MySQL и применения их в среде Delphi.

Составители Искендерова Э.Т.- ассистент  
Тетакаев У.Р.-к.т.н., ст.преподаватель

Рецензенты: 1.Хазамова М.А., к.т.н., доцент кафедры ТиОЭ ДГТУ  
2.Кобзаренко Д.Н., зав.лабораторией информационных технологий в энергетике ИПГ ДНЦ РАН.

Печатается по постановлению Ученого совета Дагестанского государственного технического университета от \_\_\_\_\_ 2020г.

## ЛАБОРАТОРНАЯ РАБОТА №1

Цель работы: получить практические навыки работы с СУБД MySQL, научиться разрабатывать базы данных MySQL и применять их в прикладных задачах, созданных на Delphi.

### 1. СИСТЕМЫ УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ

#### 1.1. ОСНОВЫ БАЗ ДАННЫХ

Информационные системы позволяют автоматизировать сбор и обработку данных. Они являются банками данных, включающими:

- вычислительную систему;
- базу данных (БД);
- систему управления базами данных (СУБД);
- набор прикладных программ.

БД обеспечивает хранение информации и представляет собой совокупность данных, организованных по определенным правилам. БД позволяет структурировать, хранить и обрабатывать данные различного типа.

СУБД – это совокупность языковых и программных средств, предназначенных для создания, ведения и использования БД. Персональная СУБД обеспечивает возможность создания локальных СУБД. К ним относятся Paradox, dBase, FoxPro, Access. Многопользовательские СУБД позволяют создавать информационные системы, функционирующие в архитектуре клиент-сервер. К ним относятся Oracle, InterBase, Microsoft SQL Server, MySQL.

Разработку и сопровождение MySQL, самой популярной SQL-базы данных с открытым кодом, осуществляет компания MySQL AB. MySQL AB – коммерческая компания, основанная разработчиками MySQL, строящая свой бизнес, предоставляя различные сервисы для СУБД MySQL.

Прикладные программы, или приложения, служат для обработки данных, содержащихся в БД. Пользователь осуществляет управление БД и работу с ее данными именно с помощью приложения.

#### 1.2. ТАБЛИЦЫ БАЗЫ ДАННЫХ

MySQL – это система управления реляционными базами данных. В реляционной базе данных данные хранятся не все скопом, а в отдельных таблицах, благодаря чему достигается выигрыш в скорости и гибкости. Таблицы связываются между собой при помощи отношений, благодаря чему обеспечивается возможность объединять при выполнении запроса данные из нескольких таблиц. SQL как часть системы MySQL можно охарактеризовать как язык структурированных запросов плюс наиболее распространенный стандартный язык, используемый для доступа к базам данных.

MySQL поддерживает несколько типов полей, которые показаны в табл. 1.

## Типы данных полей MySQL

Тип	Название	Описания обозначения. Диапазон
1	2	3
TINYINT	Очень малое целое число	Диапазон со знаком от -128 до 127. Диапазон без знака от 0 до 255
SMALLINT	Малое целое число	Диапазон со знаком от -32768 до 32767. Диапазон без знака от 0 до 65535
MEDIUMINT	Целое число среднего размера	Диапазон со знаком от -8388608 до 8388607. Диапазон без знака от 0 до 16777215
INT INTEGER	Целое число нормального размера	Диапазон со знаком от -2147483648 до 2147483647. Диапазон без знака от 0 до 4294967295
BIGINT	Большое целое число	Диапазон со знаком от -9223372036854775808 до 9223372036854775807. Диапазон без знака от 0 до 18446744073709551615
FLOAT(точность)	Число с плавающей точкой	Атрибут точности может иметь значение $\leq 24$ для числа с плавающей точкой обычной (одинарной) точности
DOUBLE[(M,D)]	Число с плавающей точкой удвоенной точности нормального размера	Допустимые значения: от -1,7976931348623157E+308 до -2,2250738585072014E-308, 0, и от 2,2250738585072014E-308 до 1,7976931348623157E+308
REAL[(M,D)]		Данные обозначения являются синонимами для DOUBLE
DECIMAL[(M[,D])]	«Неупакованное» число с плавающей точкой	Ведет себя подобно столбцу CHAR
NUMERIC[(M[,D])]		Данные обозначения являются синонимами для DECIMAL
DATE	Дата	Поддерживается интервал от «1000-01-01» до «9999-12-31»
DATETIME	Комбинация даты и времени	Поддерживается интервал от «1000-01-01 00:00:00» до «9999-12-31 23:59:59»

1	2	3
TIMESTAMP(M)	Временная метка	Интервал от «1970-01-01 00:00:00» до некоторого значения времени в 2037 году
YEAR[(2 4)]	Год в двухзначном или четырехзначном форматах	MySQL выводит значения YEAR в формате YYYY
TIME	Время	Интервал от «-838:59:59» до «838:59:59». MySQL выводит значения TIME в формате «HH:MM:SS»
CHAR(M)	Строка фиксированной длины	Диапазон аргумента M составляет от 0 до 255 символов
VARCHAR(M)	Строка переменной длины	Концевые пробелы удаляются при сохранении значения

*Предупреждение:* следует помнить, что при выполнении вычитания между числовыми величинами, одна из которых относится к типу UNSIGNED, результат будет беззнаковым!

### 1.3. РАБОТА С ПАКЕТОМ MYSQL-FRONT

Пакет MySQL-Front предназначен для работы с таблицами базы данных MySQL. С помощью MySQL-Front можно:

- создавать и удалять базы MySQL;
- создавать и удалять таблицы;
- создавать, редактировать и удалять поля таблиц;
- добавлять, редактировать и удалять записи в таблицах;
- редактировать и выполнять SQL-запросы к таблицам.

При открытии MySQL-Front, появляется окно диалога, в котором надо создать контакт с сервером (рис 1). Прописать параметры: название контакта Description, например «localhost»; название или IP-адрес сервера Hostname – localhost; имя пользователя User – root.

После нажатия кнопки «Connect!», произойдет соединение с сервером (рис. 2). Откроется окно программы MySQL-Front. В левой части окна из древовидной структуры выберите нужную базу данных. Мы будем работать в базе данных test. Выберите базу данных test. База данных стала доступной для просмотра и редактирования в правой части окна.

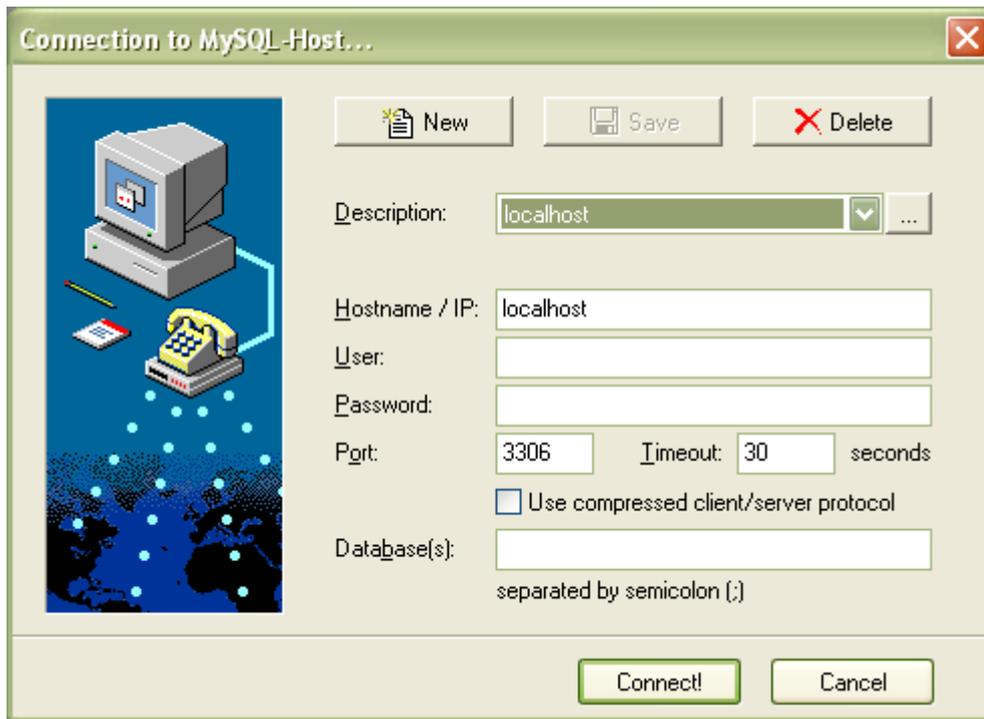


Рис. 1. Создание контакта с сервером

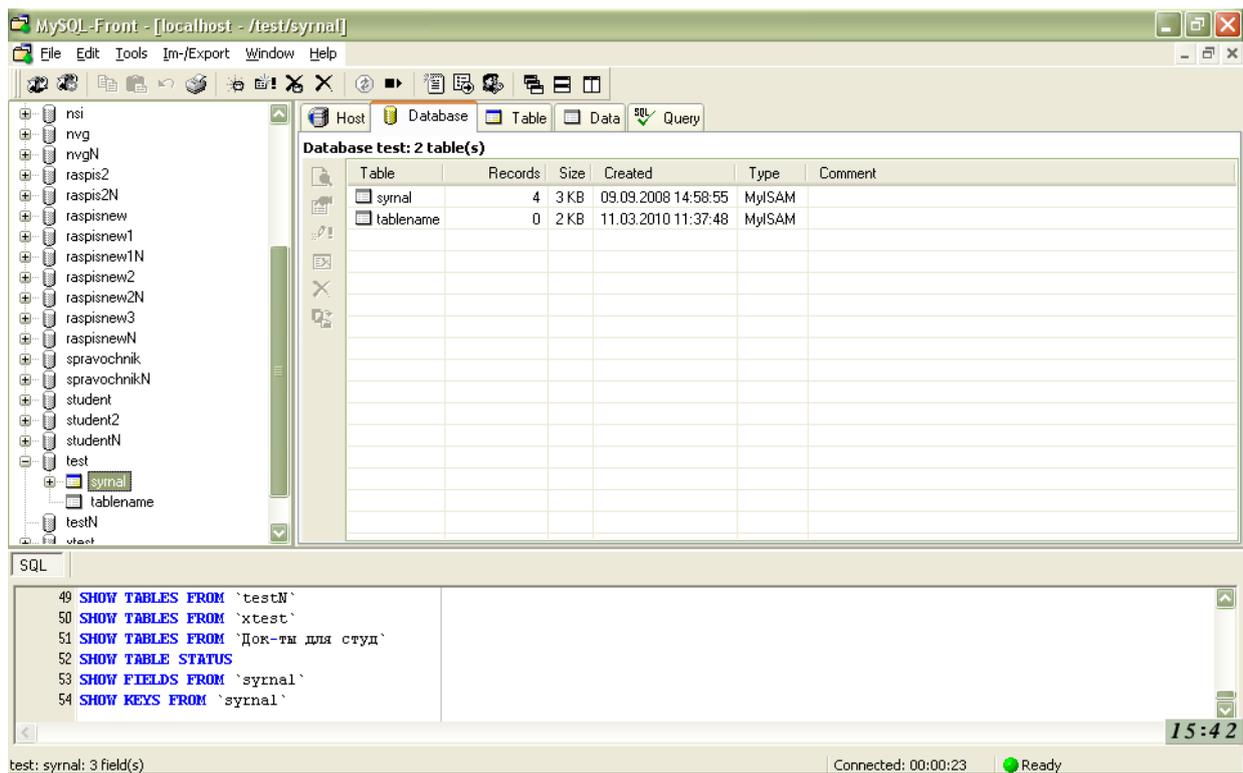


Рис. 2. Окно программы MySQL-Front

#### 1.4. СОЗДАНИЕ SQL-ЗАПРОСОВ

Редактирование и запуск SQL-запросов осуществляется в текстовом редакторе, расположенном во вкладке Query. Выполнение запроса происходит при нажатии кнопки, размещенной справа вверху от текстового редактора Execute SQL...(F9). Результат работы запроса выводится в окне, расположенном под окном текстового редактора. Запрос дублируется в окне

SQL, расположенном в нижней части. Здесь же показываются ошибки в тексте запроса.

### **1.5. СРЕДСТВА ДЛЯ РАБОТЫ С БАЗАМИ ДАННЫХ**

К средствам Delphi, предназначенным для работы с БД, относятся следующие:

- инструментальные средства (специальные программы и пакеты, обеспечивающие обслуживание БД вне разрабатываемых приложений);
- компоненты, предназначенные для создания приложений, которые осуществляют операции с БД.

### **1.6. ИНСТРУМЕНТАЛЬНЫЕ СРЕДСТВА**

Для операций с БД система Delphi предлагает такие инструментальные средства:

- Borland Database Engine (BDE) – процессор баз данных, который представляет собой набор библиотек, предназначенных для организации доступа к БД из приложений Delphi;
- BDE Administrator – утилита для настройки BDE. Позволяет настраивать различные параметры БД;
- Database Desktop – программа для создания и редактирования таблиц, SQL- и QBE-запросов;
- SQL Explorer – проводник БД, позволяющий настраивать параметры БД.

### **1.7. КОМПОНЕНТЫ**

Компоненты, связанные с БД, делятся на визуальные и невидимые:

- невидимые компоненты служат для организации доступа к данным, содержащимся в таблицах. Они представляют собой промежуточное звено между данными таблиц БД и визуальными компонентами;
- визуальные компоненты используются для создания интерфейсной части приложения. С их помощью пользователь может выполнять такие операции с таблицами, как просмотр или редактирование данных.

На странице BDE в окне Tool Palette (рис. 3) находятся невидимые компоненты, предназначенные для управления данными с использованием BDE:

- Table – набор данных, основанных на таблице БД;
- Query – набор данных, основанных на SQL-запросе;
- StoredProc – набор данных, основанных на процедуре, которая хранится на сервере;
- DataBase – соединения с БД;
- Session – текущий сеанс работы с БД;
- BatchMove – выполнение операций над группой записей;
- UpdateSQL – модификация набора данных, основанного на SQL-запросе;
- NestedTable – вложенная таблица.

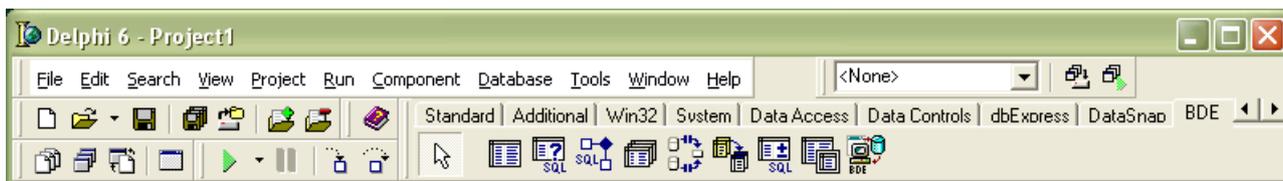


Рис. 3. Страница VDE в Delphi6

На странице Date Access в окне Tool Palette (рис. 4) находятся невидимые компоненты, с помощью которых можно организовать доступ к данным:

- Data Source – источник данных;
- ClientDataSet – клиентский набор данных;
- DataSetProvider – провайдер набор данных.



Рис. 4. Страница Data Access в Delphi6

На странице Data Controls (рис. 5) находятся визуальные компоненты, предназначенные для управления данными:

- TDBGrid – сетка (таблица);
- TDBNavigator – навигационный интерфейс;
- TDBText – надпись;
- TDBEdit – однострочный редактор;
- TDBMemo – многострочный редактор;
- TDBImage – графический образ;
- TDBListBox – простой список;
- TDBComboBox – комбинированный список;
- TDBCheckBox – независимый переключатель;
- TDBRadioGroup – группа независимых переключателей;
- TDBLookupListBox – простой список, формируемый по полю другого набора данных;
- TDBLookupComboBox – комбинированный список, формируемый по полю другого набора данных;
- TDBRichEdit – полнофункциональный текстовый редактор;
- TDBCtrlGrid – модифицированная сетка.

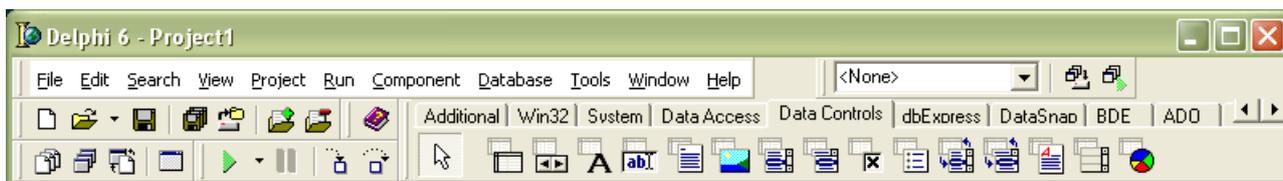


Рис. 5. Страница Data Controls в Delphi6

## 1.8. ТЕХНОЛОГИЯ СОЗДАНИЯ ПРИЛОЖЕНИЯ

В качестве примера использования возможностей Delphi для работы с БД рассмотрим технологию создания простого приложения. Основные этапы простого приложения:

- создание таблиц БД;
- создание формы приложения.

### 1.8.1. Создание таблиц базы данных

Выберите базу test. Откройте окно Database. В левом окне появится список таблиц, входящих в эту базу данных. По пустому пространству окна щелкните правой клавишей мыши. Выберите строчку «Create new Table...». Появится окно диалога создания таблицы (рис. 6). Заполните необходимые поля: название таблицы, поля таблицы, типы полей.

Чтобы назначить индексы или ключи таблицы, надо по свободному месту окна правого окна щелкнуть правой клавишей мыши. Выберите строчку «Add Field or Index...». Появится окно диалога создания нового поля. Выберите вкладку Indexes (рис. 7). Щелкните по кнопке «Add Primary». Назначьте поля для ключа из списка «Available Columns». И стрелкой переместите их в колонку «Columns used». Сохраните ключ, нажав кнопку «Update Keys».

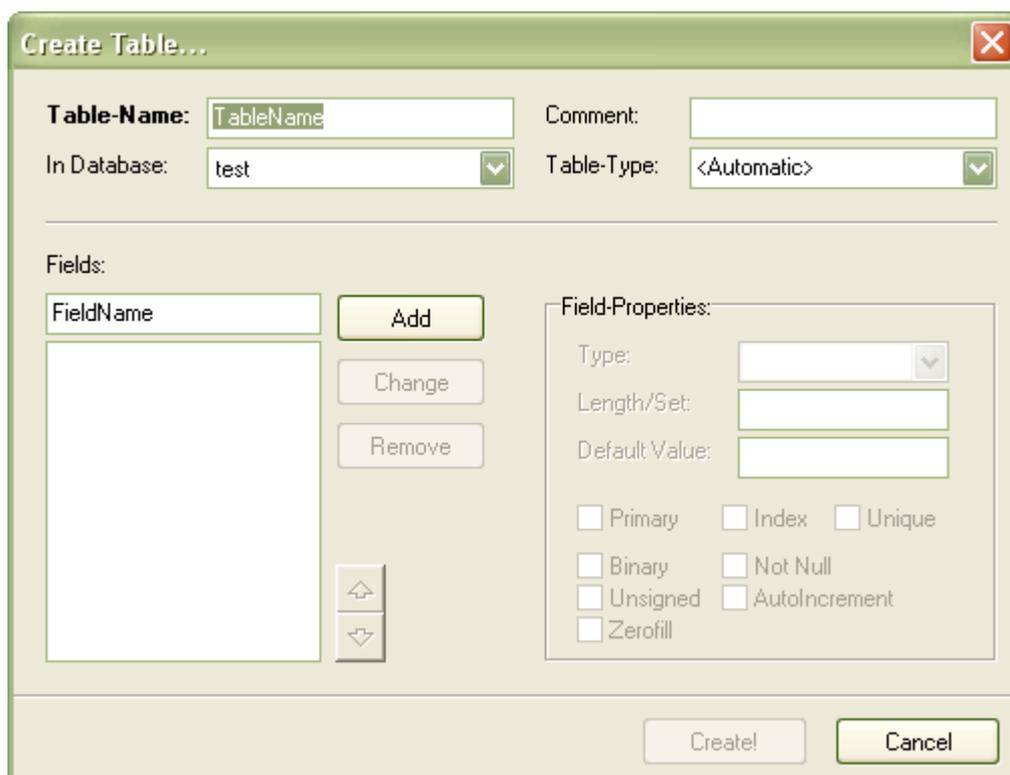


Рис. 6. Окно диалога «Create Table...»

Файлы таблиц имеют следующие расширения:

- MYD файл данных
- MYI – индексный файл (главный индекс (ключ) и вторичные индексы)
- frm – файл определения таблицы (формы) (параметры для проверки данных и целостности ссылок).

Созданные таблицы сохраняются в папке C:\xampp\mysql\data\test.

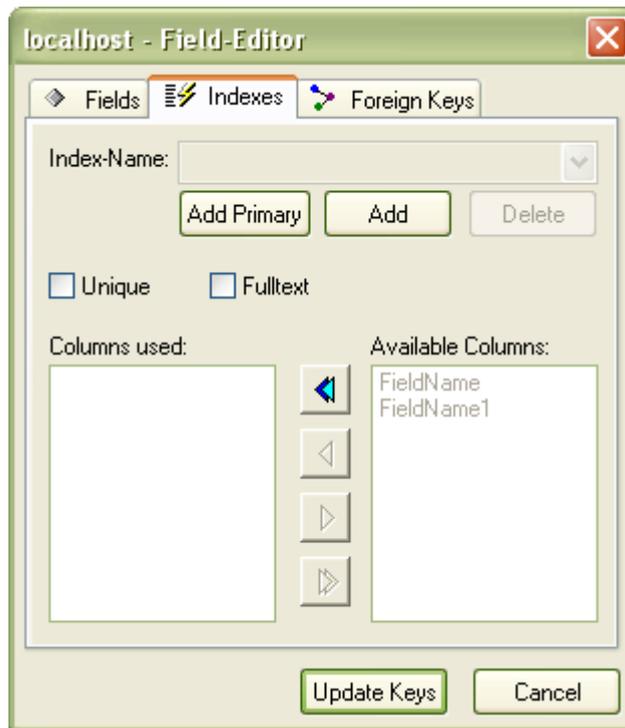


Рис. 7. Окно назначения индекса таблицы

В начале работы проверьте включение базы данных MySQL (рис. 8). Для это на рабочем столе найдите иконку «XAMPP Control Panel». Напротив раздела «MySQL» должно быть на зеленом фоне написано слово «Running». Если напротив слова «MySQL» ничего не написано, то нажмите кнопку «Start». В белом окне внизу появится текст «MySQL service started» и появится надпись на зеленом фоне «Running». Сервер запущен. Можно работать с таблицами базы данных MySQL.

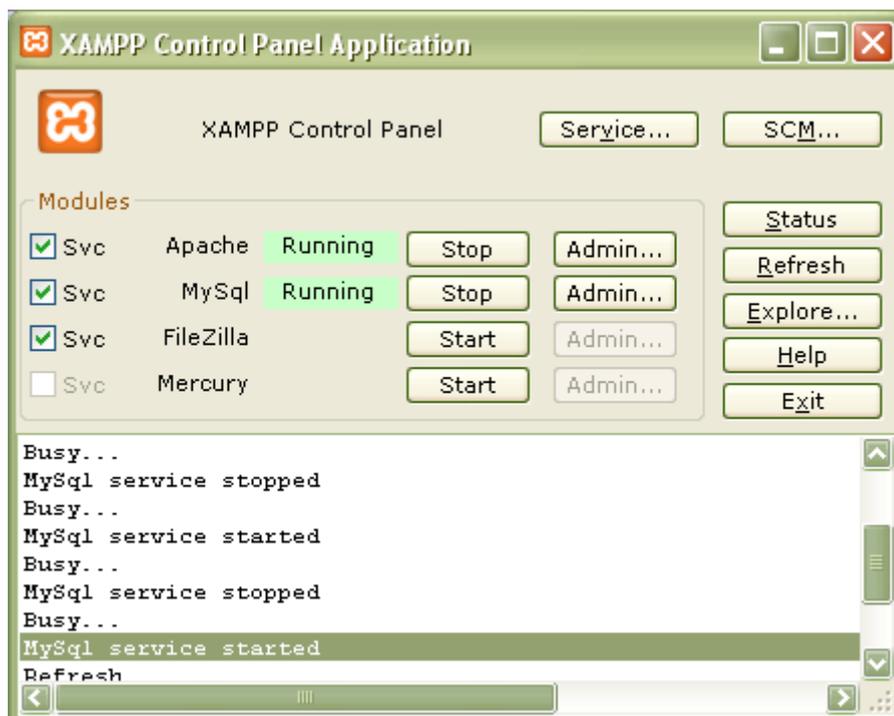


Рис. 8. Панель контроля работы сервера MySQL

### 1.8.2. Создание формы приложения

В качестве примера рассмотрим форму приложения, с помощью которого можно перемещаться по записям таблицы БД, просматривать и редактировать поля записей, вставлять в таблицу новые записи, а также удалять из таблицы записи.

Вид формы приложения представлен на рис. 9. На форме расположены следующие компоненты: Table1, DataSource1, DBGrid1 и DBNavigator1.



Рис. 9. Форма приложения для работы с БД

Компонент Table1 обеспечивает взаимодействия с БД. Для связи с требуемой таблицей необходимо установить соответствующие значения свойств DataBaseName, которое указывает путь к БД, и TableName, которое задает имя таблицы (рис. 10). После задания таблицы БД свойству Active должно быть установлено значение True.

При смене значений свойств DataBaseName или TableName нужно установить значение False свойству Active.

Имя таблицы лучше выбирать из раскрывающегося списка в поле значения свойства TableName. Если путь к БД (свойство DataBaseName) задан правильно, в этом списке отображаются все доступные файлы.

Компонент DataSource1 является промежуточным звеном между компонентом Table1, который соединен с реальной таблицей БД, и управляющими компонентами DBGrid1 и DBNavigator1, с помощью которых пользователь взаимодействует с этой таблицей (рис. 11). На компонент Table1, с которым связан компонент DataSource1, указывает свойство DataSet последнего.



Рис. 10. Свойства компонента Table1

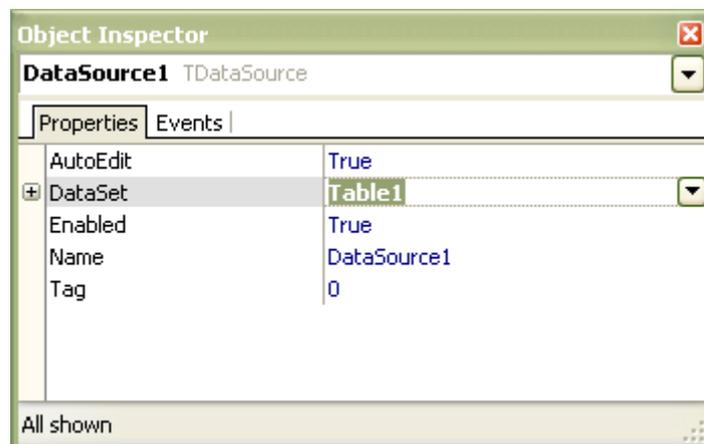


Рис. 11. Свойства компонента DataSource1

Компонент DataBase1 используется для соединения с БД (рис. 12). Свойство AliasName типа String указывает псевдоним БД. На этапе разработки приложения псевдоним выбирается из списка в Интернете объектов. Свойство DatabaseName типа String задает имя БД, используемое только в приложении для организации подключения к БД.



Рис.12. Свойства компонента Database1

Компонент DBGrid1 отображает содержимое таблицы БД в виде сетки, в которой столбцы соответствуют полям, а строки – записям. По умолчанию пользователь может просматривать и редактировать данные. Компонент DBNavigator1 предоставляет возможность перемещаться по таблице, редактировать, вставлять и удалять записи. Компоненты DBGrid1 и DBNavigator1 связываются со своим источником данных – компонентом DataSource1 через свои свойства DataSource.

При разработке приложения значения всех свойств компонентов можно задать с помощью Инспектора объектов. При этом требуемые значения выбираются из раскрывающихся списков. В табл. 2 приведены компоненты для работы с БД, а также основные свойства и их значения. В данном примере используется таблица den из базы данных test. Путь к местоположению базы данных test указывает псевдоним rrr из списка возможных Alias баз данных, прописанных на данном компьютере.

Таблица 2

Значения свойств компонентов

Компонент	Свойства	Значения
1	2	3
Table1	Active DatabaseName TableName	True (таблица den открыта) baza den

1	2	3
DataBase1	AliasName Connected DatabaseName	rrrr (алиас базы данных) True baza
DataSource1	DataSet	Table1
DBGrid1	DataSource	DataSource1
DBNavigator1	DataSource	DataSource1

### 1.9. ПРОГРАММА BDE ADMINISTRATOR

Программа BDE Administrator представляет собой администратор процессора баз данных BDE. Для вызова администратора BDE запускается файл badmin.exe. Администратор позволяет настраивать параметры БД и системные параметры (операционной системы). Параметры псевдонима: название, тип, путь. Параметры драйвера: тип, язык. Системные установки: установки по умолчанию, форматы даты, времени и числовые параметры.

Для настройки некоторого параметра в левой части окна администратора BDE выбирается нужный объект, после чего в правой части окна становится доступным список параметров этого объекта. Добавить новый объект можно, выбрав в окне администратора пункт меню Object/New (Объект/Новый).

### 1.10. РАБОТА С ПСЕВДОНИМАМИ

Псевдоним (alias) указывает на местоположение БД и представляет собой специальное имя для обозначения каталога.

Для добавления нового псевдонима перед вызовом пункта меню Object/New администратора нужно выбрать вкладку Database в левой части окна. В диалоговом окне нужно выбрать тип драйвера. Для работы с базой данных MySQL нужно выбрать пункт MySQL (рис. 13).



Рис. 13. Выбор типа драйвера

После нажатия кнопки ОК создается новый псевдоним, и его данные отображаются в окне администратора BDE (рис. 14). Новый псевдоним автоматически получает имя ODBC1 и параметры по умолчанию. Переименуем ее в название test.

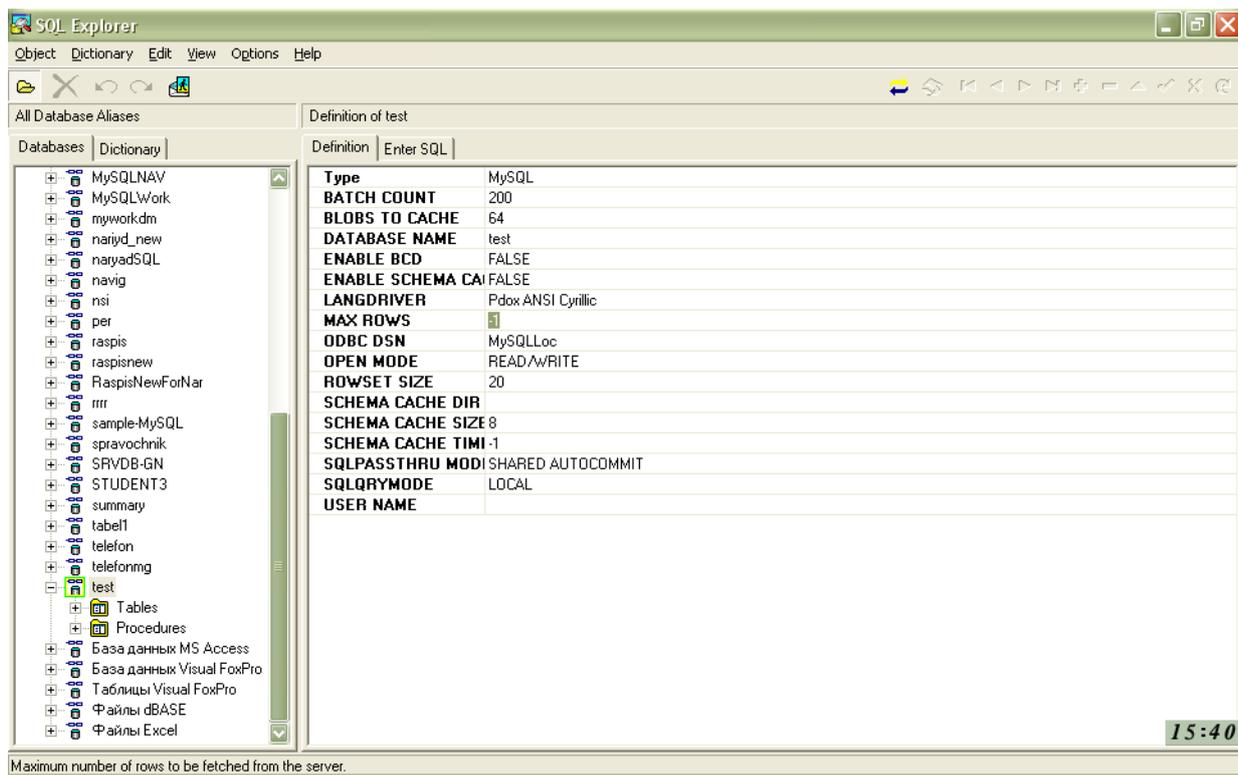


Рис. 14. Окно настройки параметров псевдонима

Псевдоним для работы с базой данных MySQL имеет следующие параметры:

- TYPE - указывает тип базы данных MySQL;
- DATABASE NAME – указывает имя удаленной базы данных;
- ODBC DNS – указывает источник данных;
- SQLQRYMODE – задает режим выполнения запросов (локальный или сетевой вариант);
- LANGDRIVER – определяет драйвер языка.

Драйвер источника базы данных MySQL устанавливается дополнительно, если нет стандартного драйвера на рабочем компьютере.

Чтобы указать источник данных MySQL, вызываем Администратор источников данных ODBC (рис. 15). Нажимаем кнопку «Добавить» и из списка драйверов (рис. 16) выбираем драйвер MySQL. В открывшемся окне (рис. 17) указываем параметры: название источника данных, сервер источника данных, название базы данных, пароль подключения.

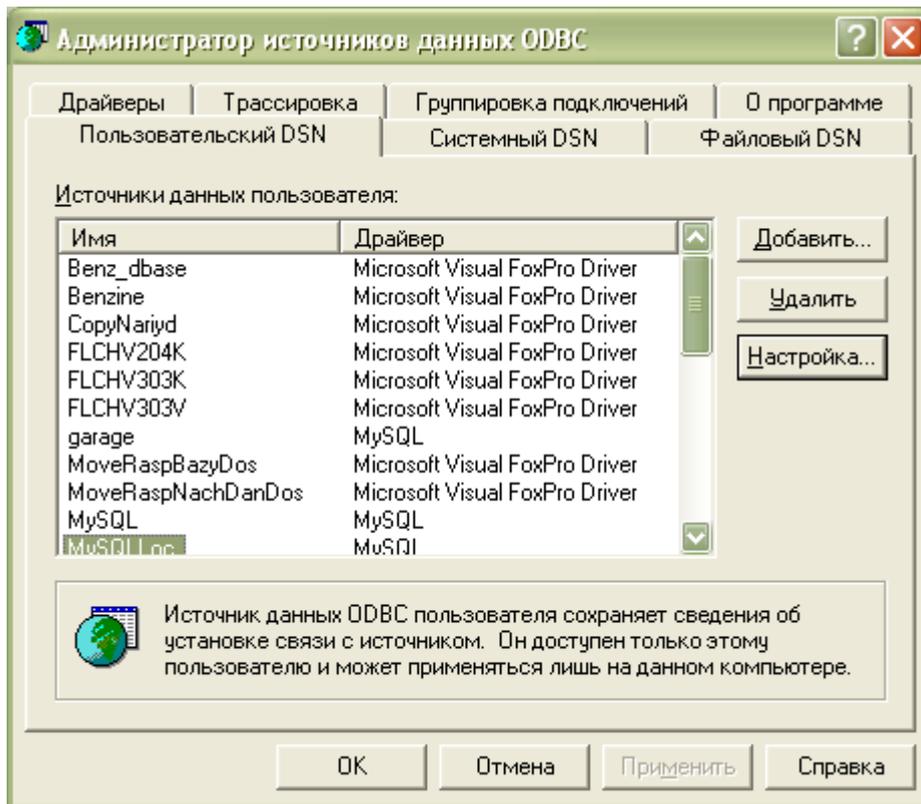


Рис. 15. Администратор источников данных ODBC

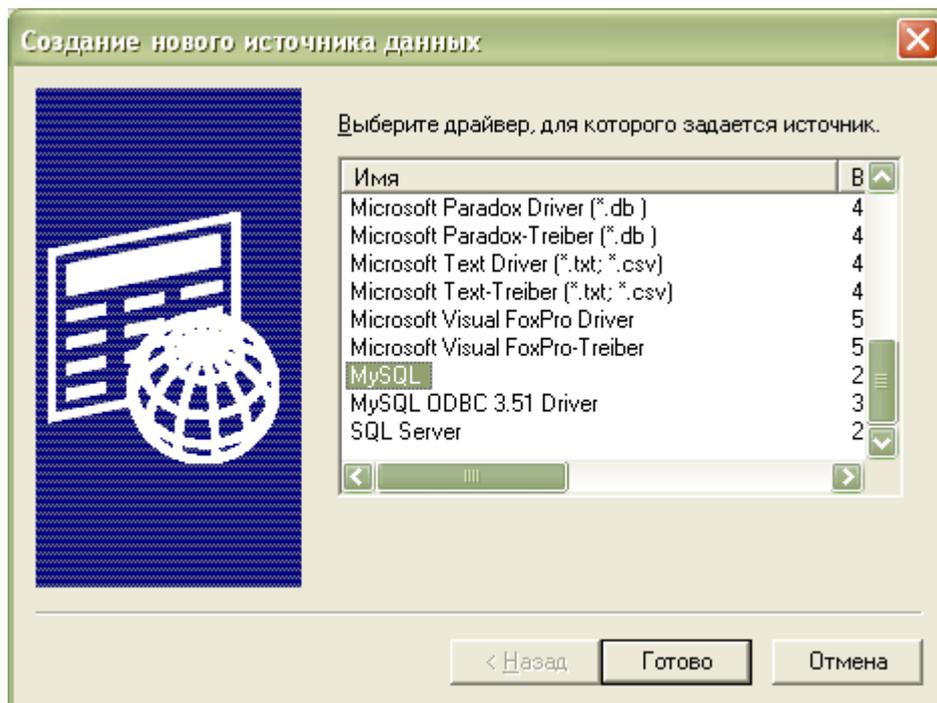


Рис. 16. Окно создания нового источника данных

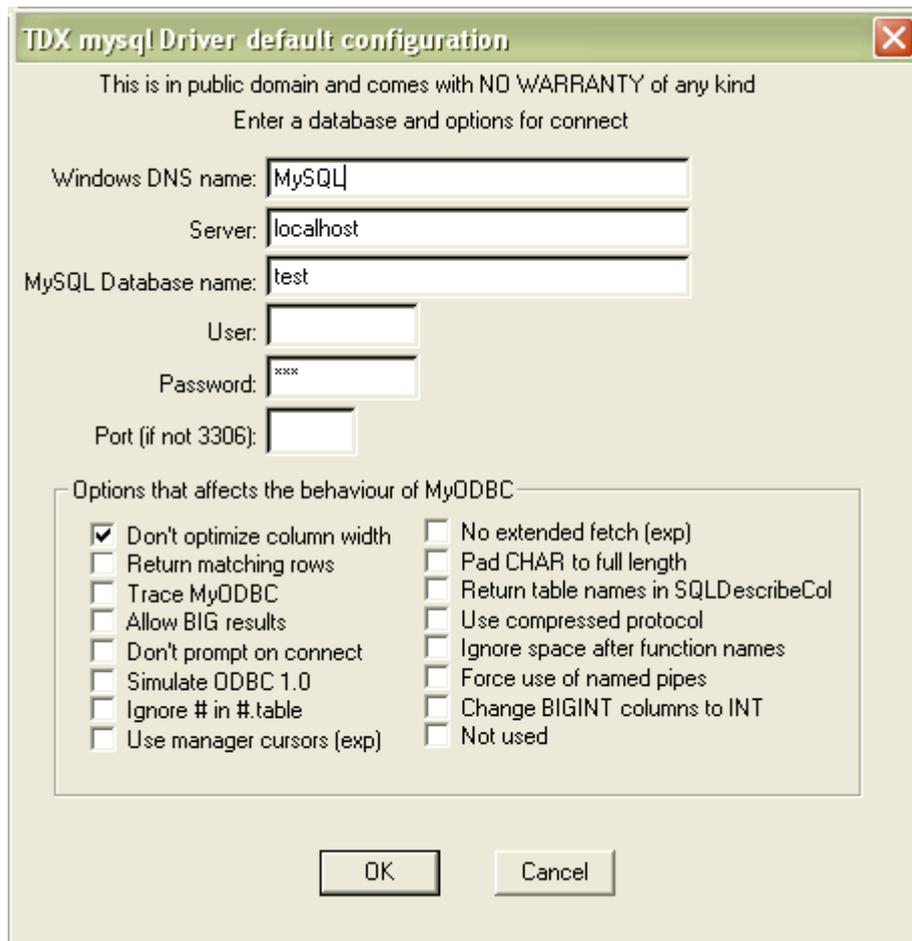


Рис. 17. Окно параметров базы данных MySQL

### 1.11. РАБОТА СО СВЯЗАННЫМИ ТАБЛИЦАМИ

Между отдельными таблицами БД может существовать связь, которая организуется через поля таблиц. Поля связи обязательно должны быть индексированными. Связь между таблицами определяет отношение подчиненности, при котором одна таблица является главной (родительской), а другая – подчиненной (дочерней или детальной). Обычно используется связь «один ко многим», когда одной записи в главной таблице может соответствовать несколько записей в подчиненной таблице.

Для организации связи между таблицами в подчиненной таблице используются свойства (в Object Inspector):

- MasterSource – источник данных главной таблицы;
- IndexName – текущий индекс;
- IndexFieldNames – поле или поля связи текущего индекса подчиненной таблицы;
- MasterFields – поле или поля индекса главной таблицы.

Для таблиц MySQL в качестве полей связи могут использоваться поля главного индекса, а для подчиненной таблицы – поля вторичного индекса.

Рассмотрим пример вывода данных из главной и подчиненной таблиц (рис. 18). Для организации связи в качестве поля связи главной таблицы берется автоинкрементное поле Kod уникального кода. По этому полю построен

главный ключ, значение которого автоматически формируется при добавлении новой записи и в пределах таблицы является уникальным. В подчиненной таблице полем связи является целочисленное поле KodPe, по которому построен вторичный индекс.

The screenshot shows a software window with a blue title bar containing the text 'Виды ремонтов выбранной ПЕ'. The window is divided into two sections. The top section, titled 'Главная таблица', contains a table with 6 columns: 'Kod', 'Transport\_predpr', 'Pe', 'Tip\_pe', and 'Data\_izgot'. The bottom section, titled 'Подчиненная таблица', contains a table with 4 columns: 'Kod', 'KodPE', 'Rem', and 'DataRem'. The second row of the main table is highlighted in blue.

Kod	Transport_predpr	Pe	Tip_pe	Data_izgot
1	западное депо	600	T-3	01.02.1980
2	западное депо	801	СПЕКТР	01.06.2000
3	западное депо	744	T-3M	01.05.1990
4	Южное депо	315	T-3	01.01.1980
5	Южное депо	500	T-3	01.01.1980
6	Южное депо	374	T-3M	01.01.1990
7	Октябрьское депо	205	628В-С	01.01.1990
8	Октябрьское депо	1	682Г	01.01.2000
9	Орджоникидзевское депо	7	628Г	01.01.2000

Kod	KodPE	Rem	DataRem
7	2	ТО-1	01.02.2000
10	2	ТО-1	01.10.2000
8	2	ТО-2	10.03.2000
9	2	Средний ремонт	25.05.2000

Рис. 18. Форма связи между таблицами «один ко многим»

В верхней части формы (рис. 18) выводится список всех подвижных единиц (ПЕ), в нижней – сведения о ремонтах выбранной ПЕ. Для наглядности в наборы данных включены все поля таблиц, которые отображаются в компонентах DBGrid. При этом названия заголовков столбцов совпадают с названиями полей.

В рассмотренном примере связь между таблицами устанавливается при выполнении приложения. Обычно таблицы связывают на этапе разработки через Инспектор объектов. При этом для установки свойства MasterFields можно использовать редактор полей связи (Field link Designer). В списке Detail Fields выбирается поле подчиненной таблицы, а в списке Master Fields – поле главной таблицы (рис. 19).

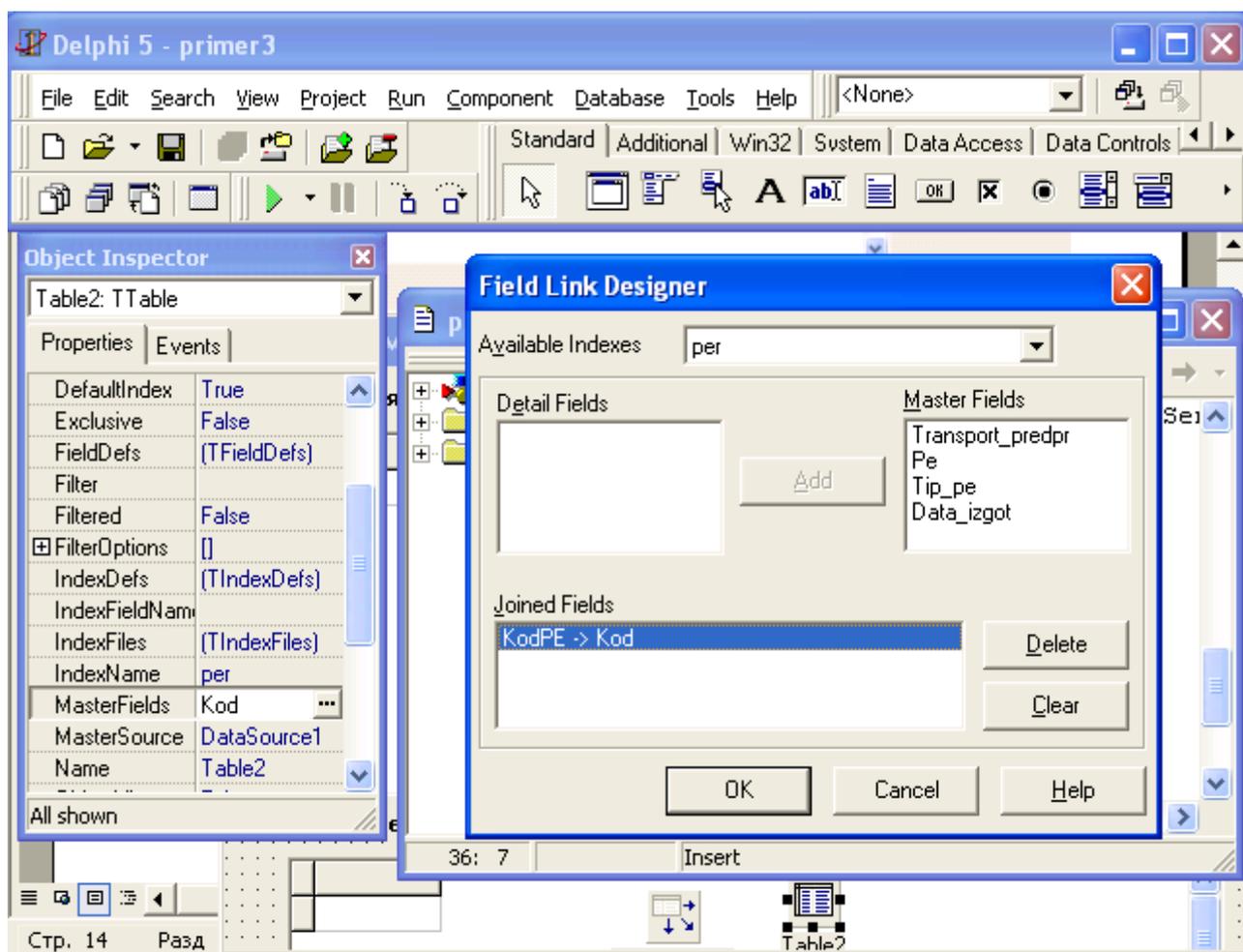


Рис. 19. Редактор полей связи

После нажатия кнопки Add выбранные поля связываются, что отображается в списке Joined Fields. Заполнение свойства MasterFields происходит после закрытия окна при нажатии кнопки OK.

В Delphi для работы с наборами данных таблиц применяются компоненты Table, Query, Decision Query и StoredProc.

Компонент Table представляет набор данных, который в некоторый момент времени может быть связан с одной таблицей БД. Расположение БД, с таблицами которой выполняются операции, указывает свойство DataBaseName типа String (рис. 20). Значением свойства является имя каталога, в котором расположены БД, или псевдоним, ссылающийся на этот каталог.

Связь между таблицей и компонентом Table устанавливается через его свойство TableName типа TFileName, которое определяет имя таблицы (и имя файла, содержащего данные). По умолчанию в набор данных Table попадают все записи связанной с ним таблицы. Для отбора данных можно использовать фильтрацию.

Для того чтобы запретить пользователю изменять записи, можно использовать свойство ReadOnly типа Boolean. По умолчанию это свойство имеет значение False. Это означает, что пользователь может редактировать записи.

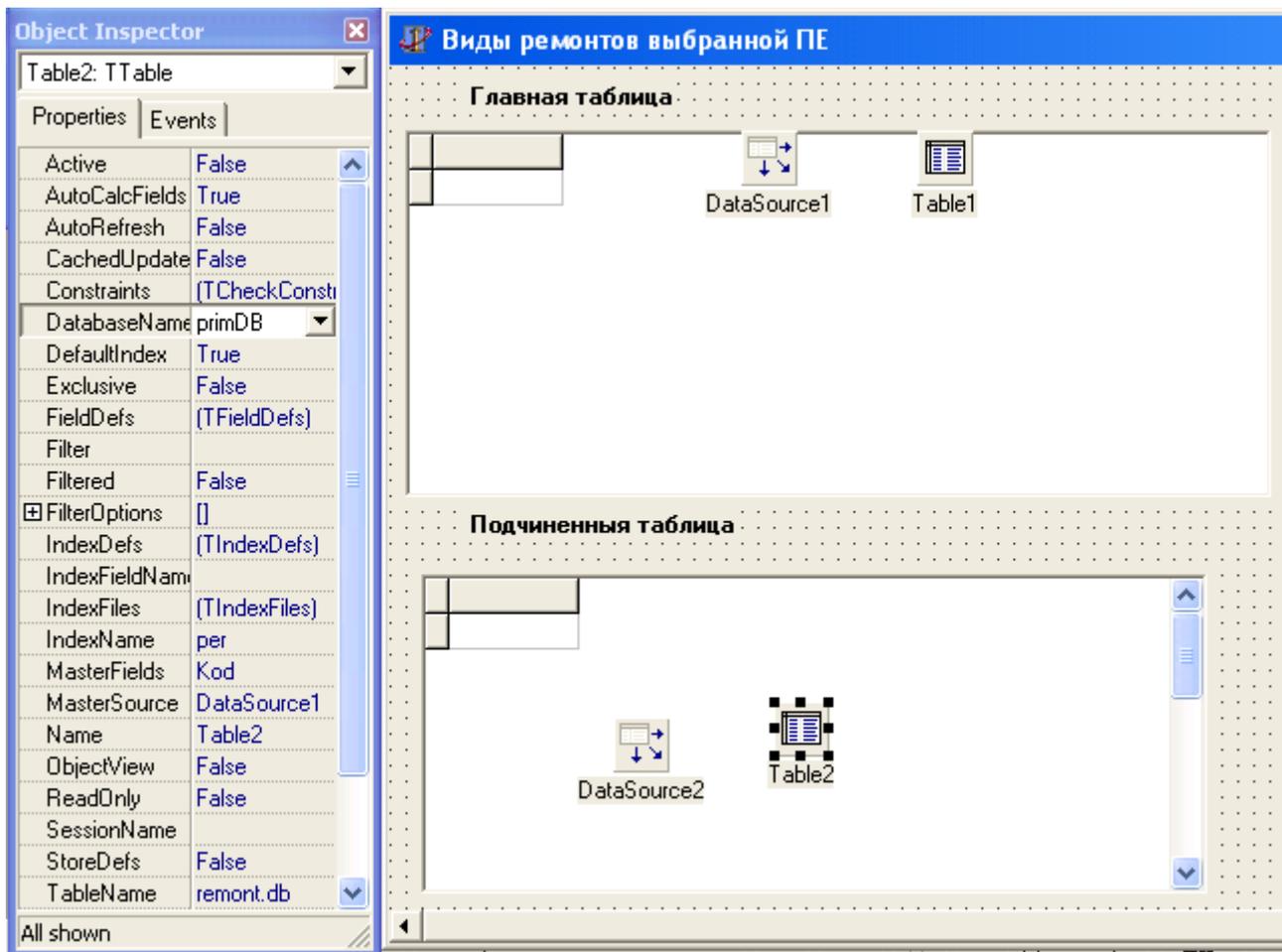


Рис. 20. Свойства компонента Table в инспекторе объектов

Установить текущий индекс можно с помощью свойства `IndexName` типа `String`. Текущий индекс выбирается из списка индексов, которые были заданы при создании таблицы. На этапе разработки приложения все возможные значения свойств `IndexName` содержат раскрывающиеся списки, доступные в Инспекторе объектов.

Наборы данных могут находиться в открытом или закрытом состоянии, на что указывает свойство `Active` типа `Boolean`. Если свойству `Active` установлено значение `True`, то набор данных открыт. Если значение свойства `Active` равно `False` (по умолчанию), то набор данных закрыт и его связь с БД разорвана.

Для вывода содержимого набора данных в табличном виде удобно использовать сетку, представленную компонентом `DBGrid`. Внешний вид сетки соответствует внутренней структуре таблицы БД и набора данных, при этом строке сетки соответствует запись, а столбцу – поле.

Компонент `DataSource` – источник данных. Используется как промежуточное звено между набором данных и визуальными компонентами, с помощью которых пользователь управляет этим набором данных. Для указания набора данных, с которым связан источник данных, используется свойство `DataSet` типа `TDataSet` последнего. Визуальные компоненты связаны с источником данных через свои свойства `DataSource` при проектировании в Инспекторе объектов.

## 1.12. РАЗРАБОТКА WEB-ПРИЛОЖЕНИЙ

Специальные компоненты Delphi позволяют размещать информацию из БД непосредственно на HTML-страницы.

При разработке WEB-приложений используются компоненты палитры компонентов, расположенных на странице Internet (рис. 21, 22). Эти компоненты автоматически генерируют HTML-страницы.

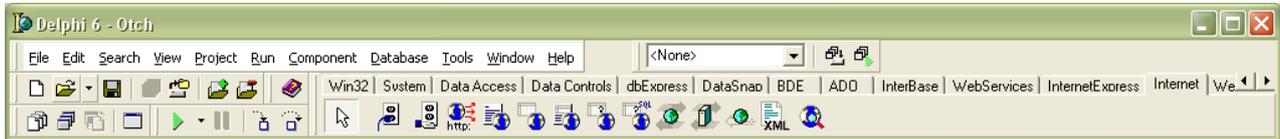


Рис. 21. Страница Internet палитры компонентов на Delphi6

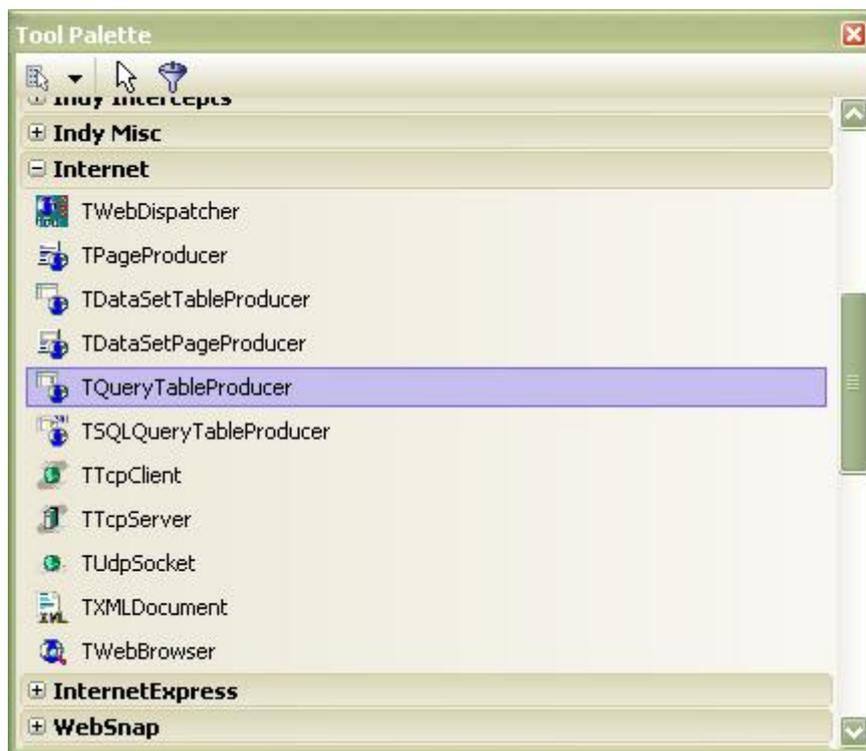


Рис. 22. Страница Internet палитры компонентов на Delphi7

Создадим новое приложение. На приложении разместим компоненты QueryTableProducer1, Query1, DataSource1, Database1, Memo1, DBGrid1 (рис. 23).

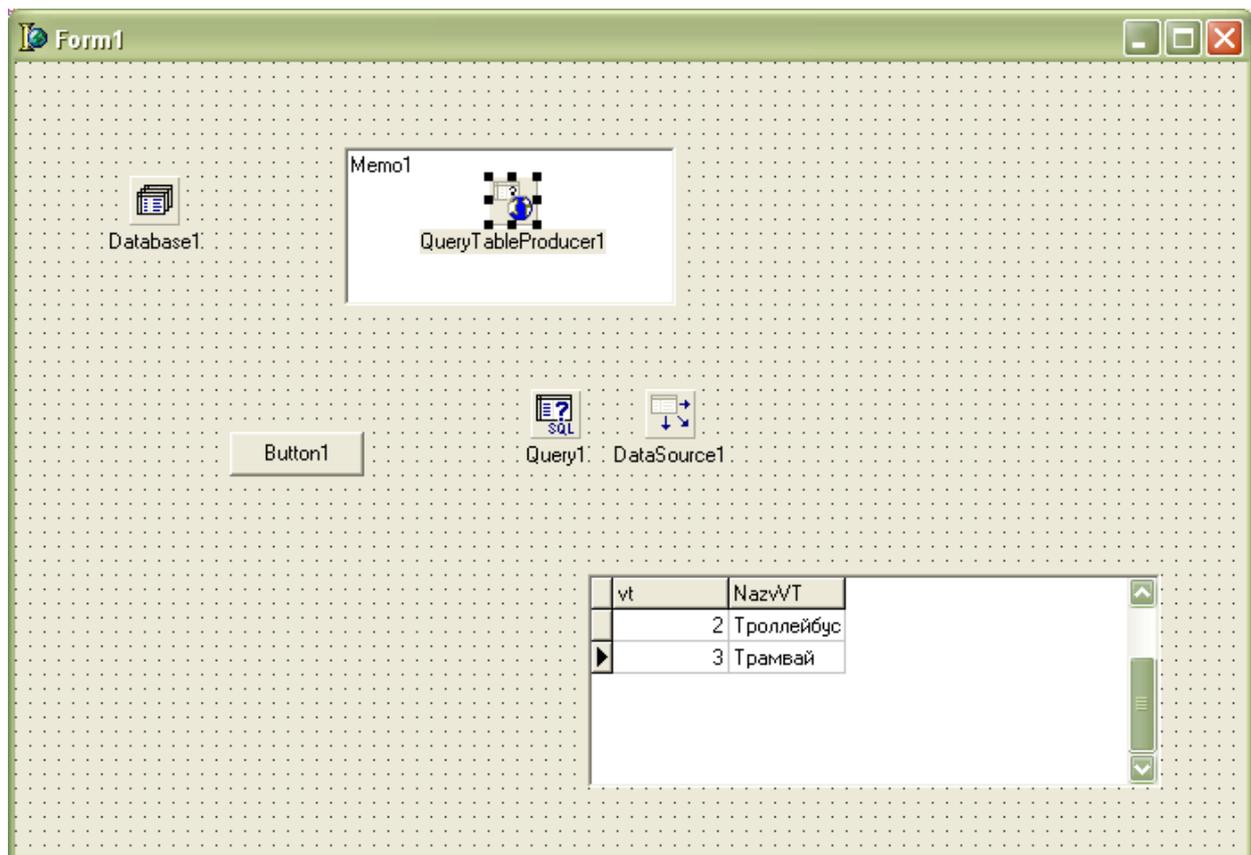


Рис. 23. Компоненты на форме приложения

## 2. ЗАДАНИЯ К ЛАБОРАТОРНОЙ РАБОТЕ

1. Установить дистрибутивы:
  - A. Запустить дистрибутив пакета XAMPP. Согласиться со всеми предложенными параметрами.
  - B. Автоматически установится имя пользователя «root». Пароль – пустое значение.
  - C. Запустить дистрибутив пакета MySQL-Front. Согласиться со всеми предложенными параметрами.
  - D. Установить драйвер источника данных MySQL. Согласиться со всеми предложенными параметрами.
2. Ознакомиться с заданием. Вариант уточнить у преподавателя.
3. Нарисовать три этапа построения ER-диаграммы для выданного задания.
4. Разработать таблицы в первой, второй и третьей нормальной форме.
5. Создать таблицы в базе данных test на сервере MySQL.
6. Создать приложение на языке программирования Delphi для третьей нормальной формы.
7. Выполнить SQL-запросы для Вашего варианта задания (п. 2÷7).
8. Разработать приложение для вывода отчета в браузер для Вашего варианта задания (п. 9).

## **ВАРИАНТЫ ЗАДАНИЙ**

### **Вариант 1**

1. Спроектировать базу данных в первой, второй и третьей нормальных формах для учета перемещения комплектующих деталей компьютеров (материнская плата, оперативная память, видеокарта, CDROM и т.п.) на предприятии: инвентарный номер компьютера, название и номер комплектующей детали, дата установки, рабочее место. Предприятие имеет несколько отделов, в каждом отделе несколько рабочих мест. Должности работников в разных отделах могут быть одинаковыми.
2. Создать запросы, в которых необходимо:
  - вывести инвентарный номер и наименование ПК, на которые за последний месяц установили один тип комплектующих деталей, например CDROM;
  - вывести список работников, на ПК которых устанавливались комплектующие детали в последний месяц.
3. С помощью запроса создать таблицу «Комплектующие детали для директора», в которой был список комплектующих деталей, установленных на ПК директора за последний месяц.
4. Для комплектующих деталей типа CDROM обновить наименование «Устройство для чтения компакт-дисков».
5. Определить, сколько рабочих мест в каждом отделе было модернизировано (установлены комплектующие детали) за последний год.
6. Вывести информацию об установках комплектующих деталей на ПК с указанием рабочего места для любого сотрудника, ФИО работника задается в режиме диалога.
7. Создать формы для ввода справочной информации (об отделах предприятия, рабочих местах, компьютерах, комплектующих деталях), а также текущей информации для накладной по установке деталей на ПК.
8. Создать многотабличную форму с помощью Мастера форм. Главная форма – накладная для установки деталей на ПК. Подчиненная форма – наличие комплектующих деталей на данном ПК.
9. С помощью компонентов библиотеки NTTPApp (страница Internet среды Delphi) создать HTML-страницу в браузере на основе таблицы «Накладная для установки КД», сгруппировав данные по инвентарному номеру ПК. Вывести комплектующие детали, дату установки КД, Отсортировать данные по номеру комплектующих деталей.

### **Вариант 2**

1. Спроектировать базу данных в первой, второй и третьей нормальных формах для ежедневного учета посещений врачей любой квалификации в поликлинике. В больничной карточке указывается: дата приема, фамилия, имя и отчество больного, возраст, адрес, ФИО врача, поставленный диагноз, назначенная процедура, номер больничного, дата открытия и закрытия больничного.

2. Создать запросы, в которых необходимо:
  - вывести список больных, посетивших педиатра за последний месяц с диагнозом ОРЗ;
  - вывести список больных и процедур, назначаемых им или нет за последний месяц;
  - вывести номера больничных листов, которые были закрыты в день их открытия.
3. С помощью запроса создать таблицу «Возможная эпидемия», содержащую список больных, которые за последнюю неделю заболели гриппом.
4. Для больных с диагнозом ОРЗ и датой открытия больничного листа недельной давности закрыть больничный лист.
5. Определить, сколько больных в возрасте от 40 до 50 лет были на больничном в течение года с одинаковыми диагнозами.
6. Вывести информацию о всех больных, которых принимал врач. ФИО врача задается пользователем в режиме диалога.
7. Создать формы для ввода справочной информации (о враче, квалификации врача, больных, диагнозах, процедурах), а также текущей информации для больничной карточки.
8. Создать многотабличную форму с помощью Мастера форм. Главная форма – больничная карточка, подчиненная – регистрация больничного листа.
9. С помощью компонентов библиотеки NTTPApp (страница Internet среды Delphi) создать HTML-страницу в браузере на основе таблицы «Больничная карточка», сгруппировав данные по ФИО врача. Вывести дату приема, ФИО больного, процедуры. Отсортировать данные по ФИО больного.

### **Вариант 3**

1. Спроектировать базу данных в первой, второй и третьей нормальных формах для диспетчера сети аптек. Должны быть указаны адрес аптеки, телефон, специализация аптеки. Информация о лекарствах: наименование лекарств, их характеристика – от каких заболеваний, доза в упаковке, доза применения, расфасовка. Информация о наличии лекарств должна содержать: дату поступления, количество упаковок лекарства, цену, название завода-изготовителя, дату продажи последней упаковки лекарства.
2. Создать запросы, в которых необходимо:
  - вывести информацию об аптеках, которые покупают лекарства в Венгрии;
  - вывести информацию о том, в каких аптеках имеется в продаже анальгин и в каких количествах;
  - вывести лекарства, на которые не устанавливается НДС.
3. С помощью запроса создать таблицу «Наиболее дорогие лекарства», содержащую список лекарств от эпилепсии, которые закупили аптеки в последний год при цене за упаковку более 1000 р.
4. Увеличить на 5 % цену продажи лекарств, поступивших в аптеку за последнюю неделю.
5. Определить сумму, которую затратила каждая аптека при покупке лекарств от гриппа за последний год.

6. Вывести информацию о лекарствах, которые покупали за последний год в зависимости от заболевания, заданного пользователем в режиме диалога.
7. Создать формы для ввода справочной информации (об аптеках, лекарствах, заболеваниях), а также текущей информации о поступлении лекарств в аптеки.
8. Создать многотабличную форму с помощью Мастера форм. Главная форма – поступление лекарств в аптеки, подчиненная форма – розничная продажа лекарств.
9. С помощью компонентов библиотеки HTTApp (страница Internet среды Delphi) создать HTML-страницу в браузере на основе таблицы «Поступление лекарств в аптеки», сгруппировав данные по номерам аптек. Вывести дату поступления лекарств, количество упаковок, цену упаковки, цену продажи. Отсортировать данные по наименованию лекарств.

#### **Вариант 4**

1. Спроектировать базу данных в первой, второй и третьей нормальных формах для учета наличия корма в зоопарке. Информация о животных следующая: название, вид, пол, количество. Рацион питания составляется в зависимости от вида и пола животного. Пусть каждое животное кормят один раз в день, животному полагается как минимум два-три вида пищи за раз. Время кормления каждого вида животного различное.
2. Создать запросы, в которых необходимо:
  - вывести список животных (вид, пол, количество), у которых в последнем месяце было усиленное питание;
  - вывести список продуктов с указанием количества и даты приема.
3. С помощью запроса создать таблицу «Самки-хищницы на витаминном питании», содержащую список животных (самок-хищников), которые в последний месяц находятся на витаминном питании.
4. Увеличить на 20 % количество выдаваемых продуктов, калорийность которых выше 500 ккал, для животных, находящихся на усиленном питании.
5. Определить, сколько килограммов каждого продукта за последний месяц было съедено каждым животным.
6. Вывести информацию о животных и выданных им продуктах в указанное в режиме диалога время.
7. Создать формы для ввода справочной информации (о видах животных, поле животных, рационе, продуктах питания), а также текущей информации о приеме пищи животными.
8. Создать многотабличную форму с помощью Мастера форм. Главная форма – прием пищи животными, подчиненная форма – информация о животных.
9. С помощью компонентов библиотеки HTTApp (страница Internet среды Delphi) создать HTML-страницу в браузере на основе таблицы «Прием пищи животными», сгруппировав данные по рациону. Вывести дату, время кормления животного, название животного, название продуктов, количество продуктов. Отсортировать данные по животному.

## Вариант 5

1. Спроектировать базу данных в первой, второй и третьей нормальных формах для учета фондов в музее. Экспонаты музея классифицируются по видам (мебель, скульптура, книги, сувениры). Каждый зал музея имеет название, номер. Каждый экспонат имеет свой инвентарный номер, известна дата поступления в музей, кто передал экспонат (организация или частное лицо), как (на благотворительной основе или музей приобрел за деньги), историческая ценность экспоната. Указать, выставлен ли экспонат в залах или находится в запасниках. Регистрируется дата последней реставрации, срок выставочной экспозиции до следующей реставрации.
2. Создать запросы, в которых необходимо:
  - вывести список и определить ценность скульптур, находящихся в запаснике, которые необходимо будет реставрировать в следующем году;
  - вывести данные о физических лицах, которые продали или передали музею экспонаты;
  - вывести список экспонатов, которые в последний год музей приобрел за деньги, причем цена приобретения соответствовала ценности экспоната.
3. С помощью запроса создать таблицу «Меценаты», содержащую список лиц (физических или юридических), которые безвозмездно передали музею экспонаты, ценность которых более 10 тыс. рублей.
4. Обновить срок следующей реставрации, заменив его на 2011 год, для мебели, которая реставрировалась в 2001 году.
5. Определить, сколько экспонатов по видам находится в запасниках и на какую сумму ценности.
6. Вывести список экспонатов ценностью выше 5 тыс. рублей, находящихся в зале, который задан пользователем в режиме диалога.
7. Создать формы для ввода справочной информации (о залах, видах экспонатов, физических и юридических лицах, которые передавали экспонаты музею, о форме приобретения экспоната), а также текущей информации о фонде музея.
8. Создать многотабличную форму с помощью Мастера форм. Главная форма – фонд музея, подчиненная форма – информация о лице, передавшем экспонат.
9. С помощью компонентов библиотеки HTTApp (страница Internet среды Delphi) создать HTML-страницу в браузере на основе таблицы «Фонд музея», сгруппировав экспонаты по залам. Вывести название экспоната, ценность, дату последней реставрации. Отсортировать данные по ценности экспоната. Определить ценность экспонатов в каждом зале и общую ценность всего фонда музея.

### Содержание отчета:

1. Цель работы
2. В отчете отразить структуру таблиц.
3. Привести скриншот разработанной формы на Delphi и отчета в браузере.
4. Результат выполнения пунктов задания отобразить в виде скриншотов.

### **Контрольные вопросы:**

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. Гофман В.Э. Delphi 7 / В.Э. Гофман, А.Д. Хомоненко, Е. Мещеряков. СПб.: БХВ-Петербург, 2010. 1120 с.
2. Архангельская А.В. Delphi 2006. Справочное пособие / А.В.Архангельская. М.: Издательство «БИНОМ», 2009. 1152 с.
3. Кузнецов М. В. MySQL 5 / М.В. Кузнецов, И.В. Симдянов. СПб.: БХВ-Петербург, 2006. 1024 с.