Инструкционная карта № 11

Тема: «Подключение к базе данных»

База данных - это набор файлов (таблиц), в которых находится информация. Как правило, база данных состоит из нескольких таблиц, которые размещают в одном каталоге. Таблицу можно создать, воспользовавшись входящей в состав Delphi утилитой Borland Database Desktop или организовав SQL-запрос к серверу базы данных. После того как определена структура записей базы данных, можно приступить непосредственно к созданию таблицы. Таблицы создаются при помощи входящей в состав Delphi утилиты Database Desktop.

Утилита Database Desktop позволяет выполнять все необходимые при работе с базами данных действия. Она обеспечивает создание, просмотр и модификацию таблиц баз данных различных форматов (Paradox, dBASE, Microsoft Access). Кроме того, утилита позволяет выполнять выборку информации путем создания запросов.

Для того чтобы создать новую таблицу, нужно выбором из меню Tools команды Database Desktop запустить Database Desktop. Затем в появившемся окне утилиты Database Desktop надо из меню File выбрать команду New и в появившемся списке выбрать тип создаваемого файла - Table. Затем в открывшемся диалоговом окне Create Table следует выбрать тип создаваемой таблицы (значением по умолчанию является тип Paradox 7).

В результате открывается диалоговое окно Create Paradox 7 Table, в котором можно определить структуру записей таблицы. Для каждого поля таблицы необходимо задать имя, тип и, если нужно, размер поля. Имя поля используется для доступа к данным. В качестве имени поля, которое вводится в колонку Field Name, можно использовать последовательность из букв латинского алфавита и цифр длиной не более 25 символов. Тип поля определяет тип данных, которые могут быть помещены в поле. Тип задается вводом в колонку Туре символьной константы. Типы полей и соответствующие им константы приведены в таблице 1.

|  |  |  |
| --- | --- | --- |
| Наименование | Константа | Содержимое поля |
| Alpha | A | Строка символов. Максимальная длина строки определяется характеристикой Size, значения которой находятся в диапазоне 1-255. |
| Number | N | Число из диапазона 10-307- 10308 с 15-ю значащими цифрами. |
| Money | $ | Число в денежном формате. Цифры числа делятся на группы при помощи разделителя групп разрядов. Также выводится знак денежной единицы. |
| Short | S | Целое число из диапазона -32767-32767 |
| Long Integer | I | Целое число из диапазона -2 147 483 648-2 147 483 647 |
| Date | D | Дата |
| Time | Т | Время с полуночи, выраженное в миллисекундах |
| Time stamp | @ | Время и дата |
| Memo | M | Строка символов произвольной длины. Поле типа Memo используется для хранения текстовой информации, которая не может быть сохранена в поле типа Alpha. Размер поля (1-240) определяет, сколько символов хранится в таблице. Остальные символы хранятся в файле, имя которого совпадает с именем файла таблицы, а расширение файла - mb. |
| Formatted Memo | F | Строка символов произвольной длины (как у типа Memo). Имеется возможность указать тип и размер шрифта, способ оформления и цвет символов |
| Graphic | G | Графика |
| Logical | L | Логическое значение "истина" (True) или "ЛОЖЬ" (False) |
| Auto-increment | + | Целое число. При добавлении к таблице очередной записи в поле записывается число на единицу большее, чем находится в соответствующем поле последней добавленной записи |
| Bytes | Y | Двоичные данные. Поле этого типа используется для хранения данных, которые не могут быть интерпретированы Database Desktop |
| Binary | В | Двоичные данные. Поле этого типа используется для хранения данных, которые не могут быть интерпретированы Database Desktop. Как и данные типа Memo, эти данные не находятся в файле таблицы. Поля типа Binary, как правило, содержат audio-данные |

Константа, определяющая тип поля, может быть введена с клавиатуры или путем выбора типа поля из списка, который появляется при щелчке правой кнопкой мыши в колонке Туре или при нажатии клавиши < Пробел >.

Одно или несколько полей можно пометить как ключевые. Ключевое поле определяет логический порядок следования записей в таблице. Например, если символьное (тип Alpha) поле Fam (Фамилия) пометить как ключевое, то при выводе таблицы записи будут упорядочены в соответствии с алфавитным порядком фамилий. Если поле Fam не помечать как ключевое, то записи будут выведены в том порядке, в котором они были введены в таблицу. Следует обратить внимание на то, что в таблице не может быть двух записей с одинаковым содержимым ключевых полей. Поэтому в рассматриваемом примере ключевыми полями должны быть поля Fam (Фамилия) и Name (Имя). Тогда в таблицу можно будет ввести информацию об однофамильцах. Однако по-прежнему нельзя будет ввести однофамильцев, у которых совпадают имена. Поэтому в качестве ключевого поля обычно выбирают поле, которое содержит уникальную информацию. Для таблицы, содержащей список людей, в качестве ключевого можно выбрать поле Pasp (Паспорт).

Для того чтобы пометить поле как ключевое, необходимо выполнить двойной щелчок в колонке Key. Следует обратить внимание на то, что ключевые поля должны быть сгруппированы в верхней части таблицы.

Если данные, для хранения которых предназначено поле, должны обязательно присутствовать в записи, то следует установить флажок Required Field. Например, очевидно, что поле Fam (Фамилия) обязательно должно быть заполнено, в то время как поле Tel (Телефон) может оставаться пустым.

Если значение, записываемое в поле, должно находиться в определенном диапазоне, то вводом значений в поля Minimum value (Минимальное значение) и Maximum value (Максимальное значение) можно задать границы диапазона.

Поле Default value позволяет задать значение по умолчанию, которое будет автоматически записываться в поле при добавлении к таблице новой записи.

Поле Picture позволяет задать шаблон, используя который можно контролировать правильность вводимой в поле информации. Шаблон представляет собой последовательность обычных и специальных символов. Специальные символы перечислены в таблице 2.

Во время ввода информации в позицию поля, которой соответствует специальный символ, будут появляться только символы, допустимые для данного символа шаблона. Например, если в позиции шаблона стоит символ #, то в соответствующую этому символу позицию можно ввести только цифру. Если в позиции шаблона стоит обычный символ, то во время ввода информации в данной позиции будет автоматически появляться указанный символ.

Например, пусть поле Tel типа А (строка символов) предназначено для хранения номера телефона, и программа, работающая с базой данных, предполагает, что номер телефона должен быть представлен в обычном виде, т.е. в виде последовательности сгруппированных, разделенных дефисами цифр. В этом случае в поле Picture следует записать шаблон: #-##-##. При вводе информации в поле Tel будут появляться только цифры (нажатия клавиш с другими символами игнорируются), причем после ввода третьей и пятой цифр в поле будут автоматически добавлены дефисы.

Таблица 2. Специальные символы, используемые при записи шаблонов

|  |  |
| --- | --- |
| Символ шаблона | Наименование |
| \* | Цифра Любая буква (прописная или строчная) Любая буква (автоматически преобразуется в прописную) Любая буква (автоматически преобразуется в строчную) Любой символ |
| & | Любой символ (если введена буква, то она автоматически преобразуется в прописную) |
| @ | Символ, следующий за символом "точка с запятой", интерпретируется как обычный символ, а не символ шаблона |
| \* | Любое количество повторяющихся, определяемых следующим за "звездочкой" символом шаблона |

Некоторые элементы данных поля могут быть необязательными, например, код города для номера телефона. Элементы шаблона, обеспечивающие ввод необязательных данных, заключают в квадратные скобки. Например, шаблон [(###)]###-##-## позволяет вводить в поле номер телефона как с заключенным в скобки кодом города, так и без кода.

Шаблоны позволяют не только контролировать правильность вводимых в поле данных путем блокирования ввода неверных символов, но и обеспечивают автоматизацию ввода данных. Это достигается путем указания в шаблоне в квадратных или фигурных скобках списка допустимых значений содержимого поля.

Например, если для поля Address задать шаблон {Санкт-Петербург, Москва, Воронеж}\*@ или [Санкт-Петербург, Москва, Воронеж]\*@, то во время ввода данных в это поле название соответствующего города будет появляться сразу после ввода одной из букв: с, м или в. Отличие фигурных скобок от квадратных и, следовательно, этих шаблонов друг от друга состоит в том, что в первом шаблоне содержимое поля обязательно должно начинаться с названия одного из перечисленных в списке городов, а во втором - город может называться по-другому, однако его название придется вводить полностью.

После того как будет определена структура таблицы, таблицу следует сохранить. Для этого необходимо нажать кнопку Save As. В результате открывается окно Save Table As. В этом окне из списка Alias нужно выбрать псевдоним базы данных, частью которой является созданная таблица, а в поле Имя файла ввести имя файла, в котором нужно сохранить созданную таблицу.

Если перед тем как нажать кнопку Сохранить установить флажок Display table, то в результате нажатия кнопки Сохранить открывается диалоговое окно Table , в котором можно ввести данные в только что созданную таблицу.

Если таблица базы данных недоступна, то для того чтобы ввести данные в таблицу, таблицу нужно открыть. Для этого надо из меню File выбрать команду Open | Table, затем в появившемся диалоговом окне Open table в списке Alias выбрать псевдоним нужной базы данных и таблицу. Следует обратить внимание, что таблица будет открыта в режиме просмотра, в котором изменить содержимое таблицы нельзя. Для того чтобы в таблицу можно было вводить данные, нужно активизировать режим редактирования таблицы, для чего необходимо из меню Table выбрать команду Edit Data.

Данные в поля записи вводятся с клавиатуры обычным образом. Для перехода к следующему полю нужно нажать клавишу <Enter>. Если поле является последним полем последней записи, то в результате нажатия клавиши <Enter> в таблицу будет добавлена еще одна запись.

Если во время заполнения таблицы необходимо внести изменения в какое-то уже заполненное поле, то следует выбрать это поле, воспользовавшись клавишами перемещения курсора, нажать клавишу <F2> и внести нужные изменения.

Если при вводе данных в таблицу буквы русского алфавита отображаются неверно, то надо изменить шрифт, используемый для отображения данных. Для этого необходимо в меню Edit выбрать команду Preferences и в появившемся диалоговом окне, во вкладке General, щелкнуть на кнопке Change. В результате этих действий откроется диалоговое окно Change Font, в котором нужно выбрать русифицированный шрифт. Следует обратить внимание, что в Windows 2000 (Windows XP) используются шрифты типа Open Type, в то время как программа Database Desktop ориентирована на работу со шрифтами TrueType. Поэтому в списке шрифтов нужно выбрать русифицированный шрифт именно TrueType. После этого надо завершить работу с Database Desktop, так как внесенные в конфигурацию изменения будут действительны только после перезапуска утилиты.

Программа управления базой данных

Процесс создания программы управления базой данных рассмотрим на примере создания базы данных "Архитектурные памятники Санкт-Петербурга".

Перед тем как приступить непосредственно к разработке приложения управления базой данных, необходимо, используя утилиту Database Desktop, создать файл данных (таблицу) и добавить в нее несколько записей. В табл. 3 перечислены поля таблицы monuments (monuments - монументы, памятники). В таблицу monuments можно внести информацию о памятниках Санкт-Петербурга.

Таблица 3. Поля таблицы monuments

Поле Тип Размер Содержание

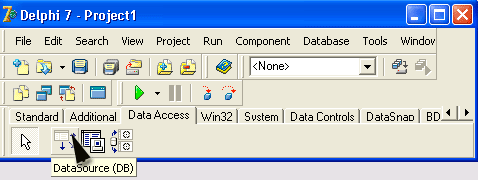
Monument A 60 Название архитектурного памятника

Architect A 40 Имя архитектора

Note A 255 Краткая историческая справка

Доступ к базе данных (таблице)

Доступ к базе данных обеспечивают компоненты Database, Table, Query и DataSource, значки которых находятся на вкладках Data Access и BDE палитры компонентов (рис. 1).



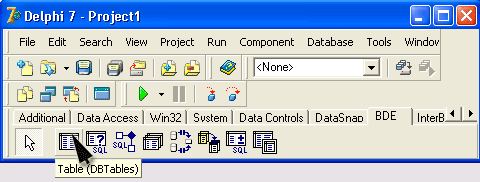


Рис. 1. Компоненты вкладок Data Access и BDE обеспечивают доступ к данным

Компонент Database представляет базу данных как единое целое, т. е. совокупность таблиц, а компонент Table - одну из таблиц базы данных. Компонент DataSource (источник данных) обеспечивает связь компонента отображения-редактирования данных (например, компонента DBGrid) и источника данных, в качестве которого может выступать таблица (компонент Tаblе) или результат выполнения SQL-запроса к таблице (компонент SQL). Компонент DataSource позволяет оперативно выбирать источник данных, использовать один и тот же компонент, например, DBGrid для отображения данных из таблицы или результата выполнения SQL-запроса к этой таблице. Механизм взаимодействия компонентов отображения-редактирования данных с данными через компонент DataSource иллюстрирует рис. 2.

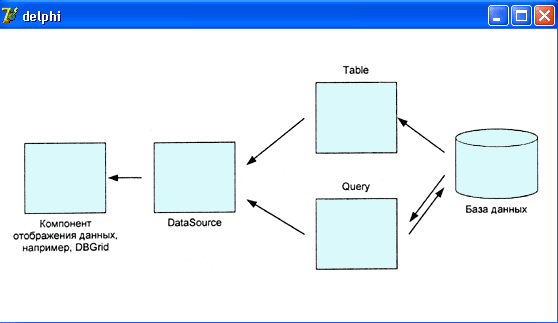


Рис. 2. Взаимодействие компонентов отображения и доступа к данным

В простейшем случае, когда база данных представляет собой одну-единственную таблицу, приложение работы с базой данных должно содержать один компонент Table и один компонент DataSource.

В табл. 5 перечислены свойства компонента Table, а в табл. 6 - свойства компонента DataSource. Свойства перечислены в том порядке, в котором следует устанавливать их значения после добавления компонентов в форму приложения.

Таблица 5. Свойства компонента Table

Name Database - Имя компонента. Используется для доступа к свойствам компонента

NameTable - Имя базы данных, частью которой является таблица (файл данных), для доступа к которой используется компонент. В качестве значения свойства следует использовать псевдоним базы данных

Name Table - Имя файла данных (таблицы данных), для доступа к которому используется компонент

Type - Тип таблицы. Таблица может быть набором данных в формате Paradox («Paradox), dBase (ttDBase), FoxPro («FoxPro) или представлять собой форматированный текстовый файл (ttASCII).

Active - Признак активизации файла данных (таблицы). В результате присваивания свойству значения True происходит открытие файла таблицы

Во время разработки формы приложения значения свойств DatabaseName и TabieName задаются путем выбора из списков. В списке DatabaseName перечислены все зарегистрированные псевдонимы, а в списке TabieName -имена файлов таблиц, которые находятся в соответствующем псевдониму каталоге.

Таблица 6. Свойства компонента DataSource

Name - Имя компонента. Используется для доступа к свойствам компонента

DataSet - Имя компонента, представляющего собой входные данные

Свойство DataSet обеспечивает связь между компонентом, представляющим собой таблицу или запрос, и компонентами, предназначенными для доступа к записям. Наличие этого свойства позволяет выбирать источник данных. Например, база данных может быть организована таким образом, что таблица, состоящая из большого числа записей, разделена на несколько подтаблиц, имеющих одинаковую структуру. В этом случае в приложении каждой подтаблице будет соответствовать свой компонент Table, а выбор конкретной подтаблицы можно осуществить установкой значения свойства DataSet.

В табл. 7 и 8 приведены значения свойств компонентов Table и DataSource для разрабатываемого приложения.

Таблица 7. Значения свойств компонента Table

Name Table1

DatabaseName Peterburg

TableName monuments . db

Active True

Таблица 8. Значения свойств компонента DataSource

Name DataSourcel

DataSet Table1

Просмотр базы данных

Пользователь может просматривать базу данных в режиме формы или в режиме таблицы. В режиме формы можно видеть только одну запись, а в режиме таблицы -- несколько записей одновременно. Довольно часто эти два режима комбинируют. Краткая информация (содержимое некоторых ключевых полей) выводится в табличной форме, а при необходимости видеть содержимое всех полей записи выполняется переключение в режим формы.

Компоненты, обеспечивающие просмотр и редактирование содержимого полей базы данных, находятся на вкладке Data Controls (рис. 3).

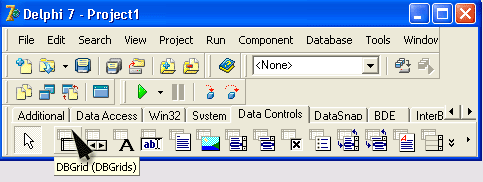


Рис. 3. Компоненты просмотра и редактирования полей базы данных

**2. Режим таблицы**

Процесс создания приложения, обеспечивающего просмотр базы данных в режиме таблицы, рассмотрим на примере программы работы с базой данных "Школа".

Пусть база данных "Школа" (псевдоним школа), представляет собой таблицу, которая находится в файле School.db. Записи таблицы school состоят из полей: Name (Имя), Fam (Фамилия), class (Класс), Adr (Адрес) и N (Личный номер). Поля Name, Fam, class и Adr являются полями символьного типа (тип А), а поле N - числовое, с автоувеличением.

Примечание

Псевдоним Школа следует создать при помощи BDE Administrator, а таблицу (файл school.db) - при помощи Database Desktop.

Сначала в форму разрабатываемого приложения нужно добавить компоненты Table и DataSource, которые обеспечивают доступ к файлу данных, и установить значения их свойств (табл. 9).

Таблица 9. Значения свойств компонентов Table1 и DataSource1

Tablel . DatabaseName - Школа school . db

Tablel . TableName - True

Tablel. Active DataSourcel . Tablel

Dataset

Для обеспечения просмотра и редактирования данных в режиме таблицы в форму приложения надо добавить компонент DBGrid, значок которого находится на вкладке Data Controls (рис. 4). Вид формы разрабатываемого приложения после добавления компонента DBGrid приведен на рис. 5

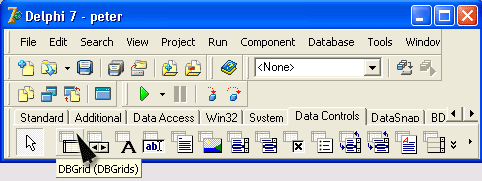


Рис. 4

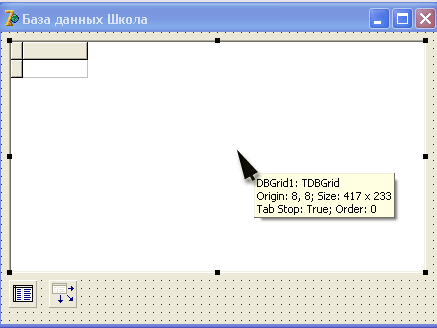


Рис. 5

Компонент DBGrid обеспечивает представление базы данных в виде таблицы. Свойства компонента DBGridl определяют вид таблицы и действия, которые могут быть выполнены над данными во время работы программы. В табл. 10 перечислены некоторые свойства компонента DBGrid.

Таблица 10 Свойства компонента DBGrid

Name - Имя компонента.

DataSource - Источник отображаемых в таблице данных.

Columns - Отображаемую в таблице информацию.

Options . dgTitles - Разрешает вывод строки заголовка столбцов.

Options . dgIndicator - Разрешает вывод колонки индикатора. Во время работы с базой данных текущая запись помечается в колонке индикатора треугольником, новая запись - звездочкой, редактируемая - специальным значком.

Options . dgColumnResize - Разрешает менять во время работы программы ширину колонок таблицы.

Options . dgColLines - Разрешает выводить линии, разделяющие колонки таблицы.

Options . dgRowLines - Разрешает выводить линии, разделяющие строки таблицы.

Для того чтобы задать, какая информация будет отображена в таблице во время работы программы, нужно сначала определить источник данных для таблицы (установить значения свойства DataSource), затем - установить значения уточняющих параметров свойства Columns. Значение свойства DataSource задается обычным образом, то есть в окне Object Inspector. Чтобы установить значение свойства Columns, надо в окне Object Inspector выбрать это свойство и щелкнуть на кнопке с тремя точками. В результате открывается окно редактора колонок (рис. 6).

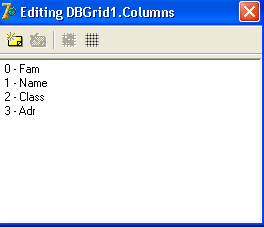


Рис 6

Для того чтобы в компонент DBGrid добавить колонку, обеспечивающую просмотр содержимого поля записи файла данных, необходимо нажать кнопку Add New, находящуюся на панели инструментов в верхней части окна (это единственная доступная после запуска редактора кнопка), выделить добавленный элемент и, используя Object Inspector, установить значения свойств этой колонки (табл. 11). Свойство columns компонента DBGrid представляет собой массив компонентов типа TCoiumn. Каждой колонке соответствует элемент массива. Устанавливая значения свойств компонентов column, программист задает вид колонок компонента DBGrid, тем самым определяет вид всей таблицы.

Таблица 11 Свойства компонента column

FieldName - Поле записи, содержимое которого выводится в колонке

Width - Ширину колонки в пикселах

Font - Шрифт, используемый для вывода текста в ячейках колонки

Color - Цвет фона колонки

Alignment - Способ выравнивания текста в ячейках колонки. Текст может быть выровнен по левому краю (taLeftJustify), по центру (taCenter) или по правому краю (taRight Justify)

Title. Caption - Заголовок колонки. Значением по умолчанию является имя поля записи

Title .Alignment - Способ выравнивания заголовка колонки. Заголовок может быть выровнен по левому краю (taLeftJustify), по центру (taCenter) или по правому краю (taRight Justify)

Title. Color - Цвет фона заголовка колонки

Title. Font - Шрифт заголовка колонки

В простейшем случае для каждой колонки достаточно установить значение свойства FieldName, которое определяет имя поля записи, содержимое которого будет отображаться в колонке, а также значение свойства Title.Caption, определяющего заголовок колонки. В табл. 12 приведены значения свойств columns компонента DBGridl.

Таблица 12. Значения свойств компонента DBGrid1

DBGrid1. Columns [0] Fam Фамилия

DBGrid1. Columns [1]Name Имя

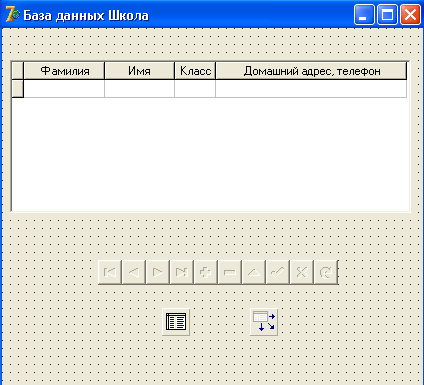
Компонент

DBGrid1. Columns [2] Class Класс

DBGrid1 . Columns [ 3 ] Adr Адрес,телефон

Последнее, что надо сделать - добавить к форме компонент DBNavigator, настроив его на работу с таблицей-источником данных (свойству DataSource Нужно Присвоить значение Table1).

Окончательный вид формы приложения приведен на рис. 7



После этого программу можно откомпилировать и запустить. Следует обратить внимание, что для того чтобы после запуска программы в окне появилась информация или, если база данных пустая, можно было вводить новую информацию, свойство Active таблицы-источника данных должно иметь значение True.

Работа с базой данных, представленной в виде таблицы, во многом похожа на работу с электронной таблицей Microsoft Excel. Используя клавиши перемещения курсора вверх и вниз, а также клавиши листания текста страницами (<Page Up> и <Page Down>), можно, перемещаясь от строки к строке, просматривать записи базы данных. Нажав клавишу <Ins>, можно добавить запись, а нажав клавишу <Del> - удалить запись. Для того чтобы внести изменения в поле записи, нужно, используя клавиши перемещения курсора влево и вправо, выбрать необходимое поле и нажать клавишу <F2>.