

第 7 章 数据库程序设计

数据库程序设计是程序设计的重要组成部分。自 20 世纪 60 年代以来,数据库技术得到了迅猛发展,从而拉开了信息时代的序幕。到了 21 世纪,数据处理是计算机四大应用(科学计算、过程控制、数据处理和辅助设计)的一个主要方面。本章将介绍数据库的相关概念,以及如何在 Delphi 中进行数据库应用程序的开发。

7.1 数据库系统简介

数据库系统是一个庞大的综合性系统,通常意义的数据库系统主要包括数据库、数据库管理系统、数据库用户几个部分。数据库(Database)是数据库系统的基础,用于存放所有的数据信息;数据库管理系统(Database Management System, DBMS)是数据库系统的核心,所有数据的操作与管理都必须在 DBMS 的控制下进行;数据库用户包括数据库管理员和普通用户,不同类型的用户拥有不同的权力,并可以通过多种方式,在 DBMS 中管理和使用数据库中的数据。

7.1.1 数据库的基本概念

下面介绍与数据库相关的基本概念。

1. 数据、信息与数据库

通常情况下,数据是指可以描述事物的符号记录,其形式多样,包括数字、文字、图形、音频与视频等。信息是指经过加工处理,具有一定含义,对用户决策有一定使用价值的数据库。数据库是存放在计算机中有组织的可以表现为多种形式的可共享数据集合。

2. 关系数据库

当前几乎所有的主流数据库均采用关系模式来描述和保存数据,这样的数据库叫关系数据库。它的特征是用二维表的结构来表示实体与实体集间的关系。

1) 实体

实体表示客观存在的可以相互区别的事务,实体可以是具体的对象,如一个学生、一门课程,也可以是抽象的事务,如学生选择课程。

2) 实体集

实体集是具有相同特征的同类型实体的集合,如学校全体学生的集合,公司全体员工的集合等。

3. 关系表

关系数据库中的数据关系用关系表表示。关系表是一张二维的表格,是按行和列排列的相关信息的逻辑组。表 7-1 就是一张描述学生信息的关系表。

表 7-1 学生信息表

学号	姓名	性别	年龄	院系	班级
200804121	张玲	女	20	计算机系	0802
200814502	王凡	男	19	英语系	0701
200809413	刘琴	女	18	化学系	0815
⋮	⋮	⋮	⋮	⋮	⋮

1) 字段

关系表中的一列称为一个字段,用来表示实体的属性,如表 7-1 中的字段用于表述学生的学号、姓名、性别、年龄等。

2) 记录

表中的一行称为一个记录,表示一个具体的实体信息。例如,表 7-1 中的第一行数据用于描述“张玲”的信息;第二行用于描述“王凡”的信息。

3) 索引

一个关系表可以按照某种特定顺序进行排列或保存,这种特定顺序称为关系表的索引。使用索引可以在数据库中快速检索出指定的记录,从而提高数据库的访问速度。例如,在表 7-1 中,可以将学号设置为索引字段。

4) 关键字

关键字也称为主键,是关系表中能唯一描述数据元素的字段。例如,表 7-1 中的“学号”字段可作为关键字,能唯一标识一条学生记录。学生的姓名、年龄、性别都可以相同,但学号是唯一的。

5) 关系表的操作

关系表的操作包括查询和更新两类。查询操作包括选择、投影、链接、除、交、并、差和笛卡尔积。更新操作包括插入、删除和修改。

目前,常用的数据库系统很多,如 Access、SQL Server、DB2、Sybase、Oracle 等。本章以 Windows 系统中使用较为广泛的 SQL Server 2000 数据库系统为例,介绍如何利用 Delphi 开发系统进行数据库程序设计。

7.1.2 创建数据库

创建数据库是数据库程序设计的第一步,通常情况下,创建数据库包括建立数据库和建立数据表两个过程。在 Windows 系统中,通常使用可视化的数据库管理工具,如 SQL Server 2000 来完成数据库的建立。

1. 建立数据库

建立数据库的操作步骤如下:

(1) 打开“开始”菜单,执行“所有程序”→Microsoft SQL Server→“服务管理器”命令,然后单击启动按钮启动 SQL Server 服务器。

(2) 打开“开始”菜单,执行“所有程序”→Microsoft SQL Server→“企业管理器”命令,打开企业管理器,如图 7-1 所示。



图 7-1 企业管理器

(3) 展开企业管理器左边的控制台根目录, 右击“数据库”目录, 在弹出的快捷菜单中选择“新建数据库”命令, 弹出“数据库属性”对话框。

(4) 在“名称”文本框中输入要建立的数据库的名称, 如图 7-2 所示, 单击“确定”按钮, 即可创建一个数据库。



图 7-2 “数据库属性”对话框

2. 建立数据表

建立数据库后, 还需要在数据库中建立数据表。建立数据表的操作步骤如下:

(1) 在企业管理器的控制台根目录中展开前面建立的 STUINF 数据库, 选中“表”选项, 如图 7-3 所示。

(2) 右击, 在弹出的快捷菜单中选择“新建表”命令, 打开表设计窗口, 如图 7-4 所示。

(3) 在 STUINF 数据库中建立一张名为 STUINF 的数据表(数据结构见表 7-2)。



图 7-3 选中“表”选项

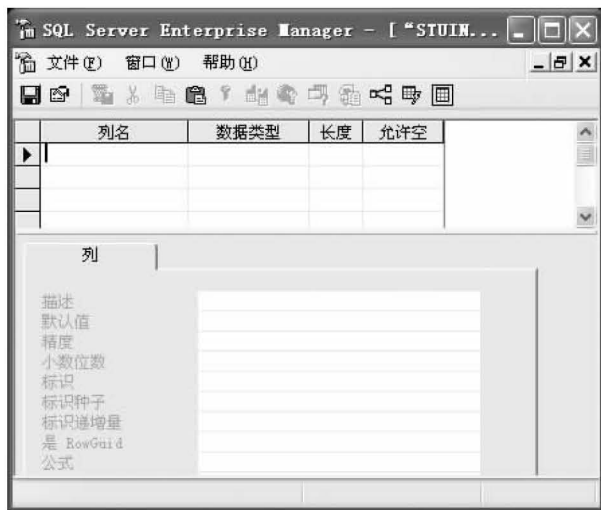


图 7-4 表设计窗口

表 7-2 STUINF 表结构

字段	数据类型	说明
STUNO	char	学号, 字符串
STUNAME	varchar	姓名, 变长字符串
STUSEX	char	性别, 字符串, 长度为 2
STUAGE	int	年龄, 整型
STUDEPT	varchar	所在院系, 变长字符串
STUCLASS	varchar	所在班级, 变长字符串

在表设计窗口中输入表 7-2 中的信息, 如图 7-5 所示, 在“允许空”列中设置该列是否允许为空(当允许为空时, 不为该列添加数据, 否则, 必须为该列添加相应数据。通常情况下,

所有数据列都不要允许为空)。右击要设置为主键的字段,如 STUNO 列,选择快捷菜单中的“设置主键”命令。所有列设置好后,单击工具栏中的“保存”按钮,即可完成数据表的创建。



图 7-5 建立数据表

(4) 建立好数据表后,在数据表中添加数据,如图 7-6 所示。

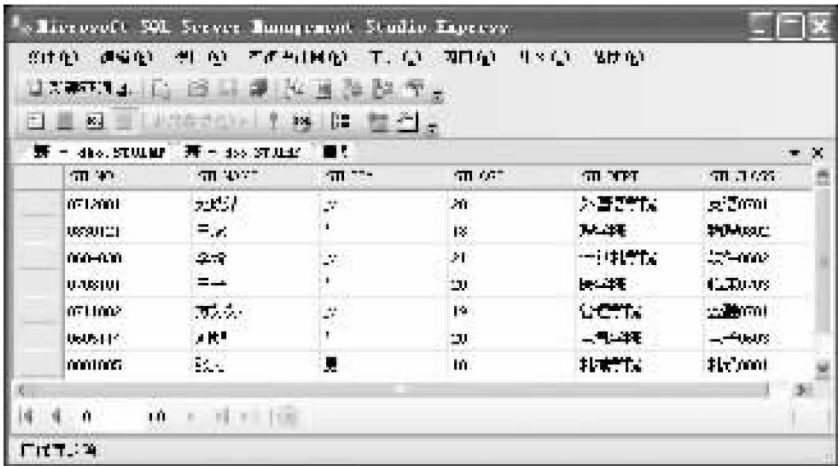


图 7-6 添加数据

7.1.3 结构化查询语言

建立数据库的目的是安全高效地进行数据操作。在数据库中,通过结构化查询语言 (Structured Query Language, SQL) 完成对数据的操作。SQL 是关系数据库的标准语言,它不仅具有强大的查询功能,还具有数据控制和数据定义等功能,它集合了数据定义语言 DDL、数据操作语言 DML 和数据控制语言 DCL 的所有功能,充分体现了关系数据库语言的优点,目前所有数据库都支持 SQL 语言,下面介绍一些简单的 SQL 语句。

1. 添加数据

INSERT 语句用于将一条记录插入指定的数据表中。

1) 完整插入

如果一张数据表中有 n 个字段,在插入一条记录时为每个字段赋值,然后将该记录插入表中,这样的插入模式称为完整插入,其语法格式如下:

```
INSERT INTO <表名> (字段 1, 字段 2, ..., 字段 n)
VALUES(取值 1, 取值 2, ..., 取值 n)
```

例如,要向 STUINF 表中完整插入一条记录时,可以使用下面的语句:

```
INSERT INTO STUINF
VALUES('200814567','王鹏','男',19,'计算机学院','软件 0812 班')
```

2) 非完整插入

如果一张数据表中有 n 个字段,在插入数据时只对其中的 m 个字段赋值,而其余 $n-m$ 个字段没有赋值,这样的插入模式称为非完整插入,其语法格式如下:

```
INSERT INTO<表名>(字段 1, 字段 2, ..., 字段 m)
VALUES(取值 1, 取值 2, ..., 取值 m)
```

例如,要以非完整插入模式在 STUINF 表中添加一条数据,可以使用下面的语句:

```
INSERT INTO STUINF(STUNO, STUNAME)
VALUES('200812421', '李婷')
```

在非完整插入记录时,没有插入数据的列均自动添加空值。

3) 注意事项

在插入数据时要注意以下两点:

- 不能插入重复的记录,即同一条记录不能插入两次。
- 在插入数据时,插入字段的属性与属性值必须一一对应。

2. 修改数据

UPDATE 语句用于修改数据表中的数据。

1) 有条件修改

在修改表中的数据时,预先设定修改的条件,只有满足条件的数据才能修改,这样的修改称为有条件修改,其语法格式如下:

```
UPDATE <表名>
SET <字段 1> = <赋值 1>, <字段 2> = <赋值 2>, .....
WHERE <条件表达式>
```

例如,可以通过下面的语句将 STUINF 表中“王鹏”的年龄修改为 20。

```
UPDATE STUINF
SET STUAGE = 20
WHERE STUNAME = "王鹏"
```

2) 无条件修改

在没有指定任何条件的情况下修改表中数据,称为无条件修改。此时,表中的所有记录都会更新,其语法格式如下:

```
UPDATE <表名>
```

SET <字段 1> = <赋值 1>, <字段 2> = <赋值 2>, ……

例如,将 STUINF 表中所有学生的性别修改为“女”,班级修改为“硬件 0810 班”,可以使用下面的语句:

```
UPDATE STUINF
SET STUSEX = "女", STUCLASS = "硬件 0810 班"
```

3. 删除数据

使用 SQL 语言中的 DELETE 语句可以删除一条记录。与 UPDATE 语句一样, DELETE 语句也分为有条件删除和无条件删除两种。

1) 有条件删除

使用有条件删除语句可以删除数据库中符合条件的记录,其格式如下:

```
DELETE FROM <表名>
WHERE <条件表达式>
```

例如,删除 STUINF 表中年龄小于 20 的学生,可以用下面的语句:

```
DELETE FROM STUINF
WHERE STUAGE < 20
```

2) 无条件删除

无条件删除是指根据条件表达式,删除数据表中所有满足条件的记录,如果不指定条件,则删除表中的所有记录。

例如,要删除 STUINF 表中年龄大于 20 的学生信息,可以使用下面的语句:

```
DELETE FROM STUINF
WHERE STUAGE > 20
```

4. 查询数据

在 SQL 语言中,最常用、功能最强大的语句是 SELECT。使用 SELECT 语句,不但可以查询需要的数据,还可以对查询到的数据进行简单处理。

1) 基本查询语句

将 SQL 语言中能实现基本查询功能的语句称为基本查询语句,这类语句仅能实现简单的查询功能,其语法格式如下:

```
SELECT <字段 1, 字段 2, …, 字段 n> FROM <表名>
WHERE <条件表达式>
```

通过基本查询语句,可以将符合条件的记录作为一个集合输出。可以在语句中指定需要显示的列,也可以用“*”表示显示所有列,例如,显示 STUINF 表中所有女生的完整记录,可以使用下面的语句:

```
SELECT * FROM STUINF
WHERE STUSEX = "女"
```

2) 扩展查询语句

为了扩展 SELECT 语句的功能,可以在 SELECT 语句中添加扩展查询选项。

(1) DISTINCT 选项。用于删除查询中重复的记录,例如,要查询 STUINF 表中的 STUCLASS 字段,如果使用语句:

```
SELECT STUCLASS FROM STUINF
```

则查询结果中会出现多条重复的记录,因为有多个学生会在同一个班级,改为下面的语句:

```
SELECT DISTINCT STUCLASS FROM STUINF
```

则一个班级只显示一条记录。

(2)GROUP BY 选项。使用 GROUP BY 选项可以将查询结果按指定字段显示,其语法格式如下:

```
SELECT <字段> FROM <表名>  
WHERE <条件表达式>  
GROUP BY <字段>
```

例如,查询 STUINF 表中年龄大于 20 的学生,并按性别显示,可以使用下面的语句:

```
SELECT * FROM STUINF  
WHERE STUAGE > 20  
GROUP BY STUSEX
```

(3)ORDER BY 选项。使用 ORDER BY 选项可以将查询结果按照指定字段中数据的规律进行排序,其语法格式如下:

```
SELECT <字段> FROM <表名>  
WHERE <条件表达式>  
ORDER BY <字段>
```

例如,将 STUINF 表中的记录按年龄大小进行排序,可以使用下面的语句:

```
SELECT * FROM STUINF  
ORDER BY STUAGE
```

3) 查询语句的统计功能

在 SQL 语言中,不仅可以使使用 SELECT 语句来查询数据,还可以通过在 SELECT 语句中添加函数对查询结果进行简单统计。

(1)COUNT 函数。可以在 SELECT 语句中添加 COUNT 函数来统计符合条件的记录条数,其语法格式如下:

```
SELECT COUNT(<字段>) FROM <表名>  
WHERE <条件表达式>
```

例如,要查找 STUINF 表中年龄小于 20 的女生人数,可以使用下面的语句:

```
SELECT COUNT(*) FROM STUINF  
WHERE STUAGE < 20 AND STUSEX = "女"
```

(2)SUM 函数与 AVG 函数。可以在 SELECT 语句中使用 SUM 和 AVG 函数分别计算查询结果中某个字段的数据之和与平均值。当然,参与计算的数据必须是数值型。

SUM 函数的语法格式如下:

```
SELECT SUM(<字段>) FROM <表名>  
WHERE <条件表达式>
```

AVG 函数的语法格式如下:

```
SELECT AVG(<字段>) FROM <表名>  
WHERE <条件表达式>
```

例如,要计算 STUINF 表中男生的平均年龄可以使用下面的语句:

```
SELECT AVG(STUAGE) FROM STUINF
```

```
WHERE STUSEX = '男'
```

(3)MAX 函数与 MIN 函数。在 SELECT 语句中,可以使用 MAX 与 MIN 函数来计算查询结果中的最大值与最小值。

MAX 函数的语法格式如下:

```
SELECT MAX(<字段>) FROM <表名>
```

MIN 函数的语法格式如下:

```
SELECT MIN(<字段>) FROM <表名>
```

例如,要查找 STUINF 表中女生的最大年龄与最小年龄,可以使用下面的语句:

```
SELECT MAX(STUAGE),MIN(STUAGE) FROM STUINF
```

```
WHERE STUSEX = "女"
```

以上介绍的是进行简单数据操作的 SQL 语句,如果想要更加深入地了解 SQL 语句的内容,可以查阅相关的书籍。

7.2 连接数据库

创建好数据库后,还需要将数据库与开发系统连接,才能进行数据库程序的开发。下面介绍连接数据库与开发系统的方法。

7.2.1 数据库连接对象

为了在应用程序中安全、高效地访问数据库,Microsoft 等软件开发公司提出了多种解决方案,其中有代表性的有 ODBC、JDBC、BDE 和 ADO 等连接模式。下面介绍其中的 AOD 连接模式。

ADO(ActiveX Data Objects,动态数据对象)连接模式的前身是 Microsoft 提出的一种通用的基于组件对象模型(Component Object Model,COM)的数据访问规则与应用程序接口函数 OLE DB。它是一种独立的数据库连接结构,可以通过 OLE DB 提供的驱动程序与任何数据库连接。ADO 连接模式则更好地封装了 OLE DB 的功能,支持多种数据库,切换数据源简单,可以更加方便快捷地访问数据库。

使用 ADO 连接模式连接数据库与开发系统的主要步骤如下:

- (1)连接至数据库,并判断其中的数据是否修改。
- (2)指定访问数据库的命令,通常是一条或多条 SQL 语句。
- (3)执行数据库访问命令。
- (4)返回执行结果,结果保存在 ADO 组件的数据缓冲区中,供应用程序调用。

7.2.2 建立 ADO 连接

建立 ADO 连接最简单的方法是创建一个通用数据连接(UDL)文件,该文件的后缀名为.udl。下面介绍具体的操作步骤。

1. 创建 UDL 文件

在 Windows 系统中不能直接创建 UDL 文件,需要先在记事本中建立一个文本文件,然后执行“文件”→“另存为”命令,打开“另存为”对话框。选择保存类型为“所有文件”,将文件的后缀名设置为 .udl,如图 7-7 所示,然后单击“保存”按钮,即可创建一个 UDL 文件。

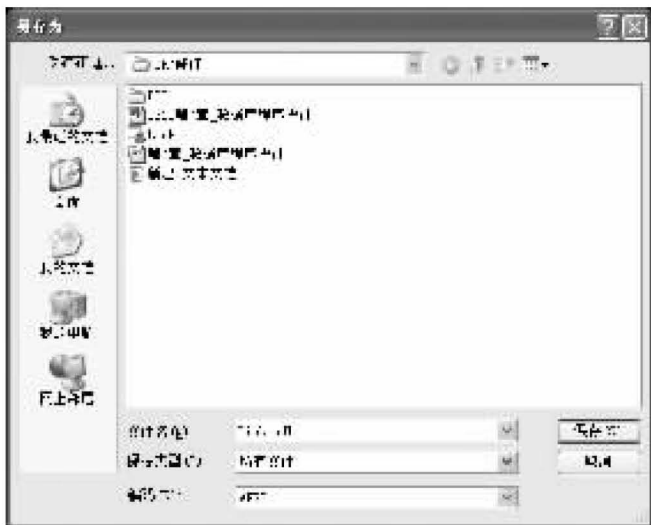


图 7-7 建立 UDL 文件

2. 选择数据库驱动

创建好 UDL 文件后,双击该文件,弹出“数据链接属性”对话框,单击“提供程序”选项卡,在“选择您希望连接的数据”列表框中选择要连接的数据。由于这里采用的是 SQL Server 数据库,所以选择 Microsoft OLE DB Provider for SQL Server 选项,如图 7-8 所示。然后单击“下一步”按钮,打开“连接”选项卡,如图 7-9 所示。



图 7-8 选择需要连接的数据



图 7-9 “连接”选项卡

3. 建立数据连接

在“连接”选项卡中执行如下操作,完成数据库与应用系统的连接。

(1)在“选择或输入服务器名称”下拉列表框中选择服务器的名称。

(2)在“输入登录服务器的信息”选项区中选择登录数据库服务器的方式,由于在安装 SQL Server 数据库时采用 Windows NT 集成安全模式(这是最常用的方式),所以选中“使用 Windows NT 集成安全设置”单选按钮,否则需要指定登录数据库的用户名和密码。

(3)选中“在服务器上选择数据库”单选按钮,然后在下边的下拉列表框中选择需要访问的数据库。

(4)单击“测试连接”按钮,如果连接成功,则弹出提示成功的对话框,如图 7-10 所示。至此,完成数据连接文件的建立。



图 7-10 测试连接成功



图 7-13 登录对话框

2. ADODataset 组件和 ADOQuery 组件

使用 ADODConnection 组件建立与数据库的连接后,如果需要获得数据库中的数据,还需要使用其他组件,如 ADODataset 和 ADOQuery 等组件。

1) ADODataset 组件


ADODataset 组件  位于组件面板的 ADO 面板中,使用该组件,可以获取数据库中的数据并将其保存在缓冲区中。

ADODataset 组件的常用属性如下:

(1) Connection。通常情况下,ADODataset 组件不直接与数据库连接,而是通过 ADODConnection 组件间接与数据库连接,方法为将 ADODataset 组件的 Connection 属性设置为要连接的 ADODConnection 组件。


(2) Active。该属性为布尔类型,用于设置是否允许使用 ADODataset 组件,只有将 Active 属性设置为 True,才能使用 ADODataset 组件。

(3) CommandText。这是 ADODataset 组件最重要的属性,其值为字符串类型,可以将 CommandText 属性设置为具体的 SQL 语句,当 ADODataset 组件执行时会自动执行该 SQL 语句,并将执行结果保存在缓冲区中供程序调用。

在设置 CommandText 属性前,必须先设置好 Connection 属性,然后选中 CommandText 属性,单击其属性值右侧的  按钮,弹出 CommandText Editor 窗口。


窗口左侧的 Tables 列表框中列出了所有可用的数据表,选中一个数据表后,在下方的 Fields 列表框中列出该数据表中的所有字段。单击 Add Table to SQL 按钮和 Add Field to SQL 按钮,可以将数据表的名称和字段名称添加到右侧的 SQL 列表框中,然后在列表框中编写 SQL 语句,如图 7-14 所示,单击 OK 按钮,即可将编写的 SQL 语句作为 CommandText 属性的值。

2) ADOQuery 组件

ADOQuery 组件  也位于 ADO 面板中,其功能与 ADODataset 组件类似,用于获取数据库中的数据并保存到 ADOQuery 组件的缓冲区中供程序调用。

ADOQuery 组件的常用属性如下:

(1) Connection: 与 ADODataset 组件的 Connection 属性相同。

(2) SQL: 为字符串类型,用于描述 SQL 语句。单击 SQL 属性值右侧的  按钮,可以在弹出的 SQL 语句编辑对话框中直接输入 SQL 语句。

(3) Active: 与 ADODataset 组件的 Active 属性相同。

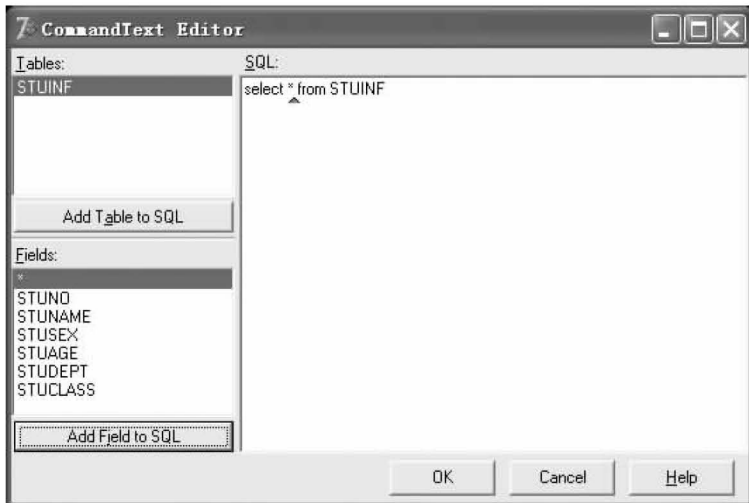


图 7-14 编写 SQL 语句

3. DataSource 组件

通常情况下,程序并不直接使用 ADODataset 组件或 ADOQuery 组件返回的数据,而是先使用 DataSource 组件进行数据过渡,该组件位于 Data Access 面板中。在程序设计过程中,通过设置 DataSource 组件的 DataSet 属性来与 ADODataset 组件或 ADOQuery 组件关联,并对这些组件中的数据进行格式化处理,将数据转换为 DataSource 组件属性值的形式供程序调用。

4. 与数据库建立连接

介绍了几个常用的数据库连接组件后,下面通过例子介绍连接数据库与 Delphi 程序的步骤。

(1) 建立一个 UDL 文件,然后测试与数据库的连接。


(2) 在 Form 窗体中添加一个 ADOConnection 组件,将其 ConnectionString 属性设置为 UDL 文件,将 Connected 属性与 KeepConnected 属性均设置为 True。

(3) 在 Form 窗体中添加一个 ADOQuery 组件,将该组件的 Connection 属性设置为上一步骤添加的 ADOConnection 组件,然后设置 SQL 属性,并将 ADOQuery 组件的 Active 属性设置为 True。

(4) 在 Form 窗体中添加一个 DataSource 组件,将其 DataSet 属性设置为上一步骤添加的 ADOQuery 组件。

至此,完成 Delphi 程序的数据库的连接。如果数据库中存在多个数据表,每个数据表要对应一个 ADOQuery 组件,且每个 ADOQuery 组件也要对应一个 DataSource 组件,以便于程序设计。

当数据库中的数据表过多时,需要在 Form 窗体中添加多个 ADOQuery 组件和 DataSource 组件,使用过多的组件将不利于界面设计和组件管理。为此,Delphi 开发系统提供了一种新的窗体类型 Data Module。执行 File→New→Data Module 命令,新建一个 Data Module 窗体。该窗体的属性很少,不需要进行特别设置,一般只需要设置 Name 属性,其他属性取默认值即可。可以像在 Form 窗体中添加组件一样将所有与数据库连接相关的组件

Selected 按钮 , 可以将选中的字段从 DBGrid 组件中移除。

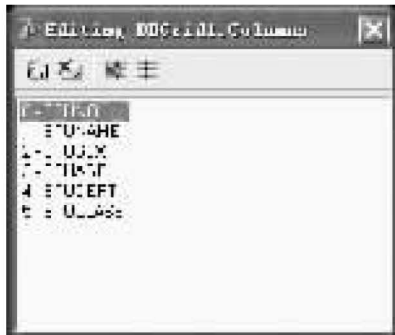


图 7-17 Columns 编辑对话框

2) 设置字段

在 Delphi 程序设计中, 可以将字段定义为 Columns 对象, 该对象的主要属性为 FieldName 和 Title。


- FieldName: 用于将 DBGrid 组件中的字段与数据表中的字段关联。
- Title: 用于设置 DBGrid 组件中字段的显示效果。该属性为集合类型, 包含的子属性有: Alignment 属性用于设置字段的显示位置; Caption 属性用于设置字段的文本内容; Width 属性用于设置字段文本的宽度。

【例 7-1】 在 Form 窗体中显示 STUINF 表中的数据。

具体操作步骤如下:

(1) 建立 UDL 文件, 测试与数据库的连接。

(2) 在 Form 窗体中添加一个 ADOConnection 组件, 一个 ADODataset 组件, 一个 DataSource 组件和一个 DBGrid 组件, 并设置这些组件的属性, 具体如下:

- 将 ADOConnection 组件的 ConnectionString 属性设置为 UDL 文件, 将 Connected 属性和 KeepConnected 属性设置为 True。
- 将 ADODataset 组件的 Connection 属性设置为 ADOConnection 组件, 将 CommandText 属性设置为“SELECT * FROM STUINF”, Active 属性设置为 True。
- 将 DataSource 组件的 DataSet 属性设置为 ADODataset 组件。
- 将 DBGrid 组件的 DataSource 属性设置为 DataSource 组件, 然后单击 Columns 属性值右侧的  按钮, 在弹出的 Columns 编辑对话框中单击 Add All Fields 按钮, 将所有字段添加到 DBGrid 组件中。

当完成上述设置后, 即可在窗体中显示 STUINF 表中的数据, 如图 7-18 所示。

(3) 设置 DBGrid 组件的 Title 属性。为了使 DBGrid 组件中的数据显示得更加规范, 还需要设置 Title 属性。选中 STUNO 字段, 将 Title 属性集中的 Caption 属性设置为“学号”, Width 属性设置为 60, Alignment 属性设置为 taCenter。采用同样的方法, 设置其他字段的属性, 最终的显示效果如图 7-19 所示。



图 7-18 在窗体中显示数据



图 7-19 设置 Title 属性后的数据显示效果

2. DBNavigator 组件

使用 DBNavigator 组件,可以在不编写代码的情况下,实现对数据库的基本操作。

1) DBNavigator 组件的属性

DBNavigator 组件位于组件面板中的 Data Controls 面板中。DBNavigator 组件的主要属性如下:

- DataSource:将该属性设置为相应的 DataSource 组件,可以选择数据源。
- VisibleButton:该属性有 10 个属性值,对应 DBNavigator 组件上的 10 个按钮,这 10 个属性值均为布尔类型,当为 True 时,显示其对应的按钮,否则不显示该按钮。

2) 使用 DBNavigator 组件

DBNavigator 组件共有 10 个按钮,如图 7-20 所示。



图 7-20 DBNavigator 组件的按钮

各按钮的功能如下：


- First 按钮 ：将第一条记录设置为当前记录。
- Prior 按钮 ：将上一条记录设置为当前记录。
- Next 按钮 ：将下一条记录设置为当前记录。
- Last 按钮 ：将最后一条记录设置为当前记录。
- Insert 按钮 ：在当前记录前添加一条空记录。
- Delete 按钮 ：删除当前记录。
- Edit 按钮 ：允许编辑当前记录(极少使用)。
- Post 按钮 ：将修改的记录提交到数据库。
- Cancel 按钮 ：撤销修改操作。
- Refresh 按钮 ：刷新当前记录。

3. 其他数据操作组件

使用 DBGrid 组件不仅可以查看数据,还可以对数据进行增加、删除、修改等操作。但是,为了便于输入数据和保持界面的一致性,在实际应用中,一般只用 DBGrid 组件来显示数据,而将输入、修改、删除数据的操作由其他组件完成。

可以使用 Data Controls 面板中的组件,如 DBEdit、DBListBox、DBComboBox 和 DBRadioGroup 等来添加、修改和删除数据。这些组件在外观上与对应的 Edit、ListBox、ComboBox 和 RadioGroup 组件相同,但使用方法不同,主要表现在组件与数据源的关联上,下面以 DBEdit 组件为例进行介绍。

1) 设置 DBEdit 组件的属性

DBEdit 组件  的主要属性有以下两个：

- DataSource: 设置 DBEdit 组件的数据源,将该属性设置为对应的 DataSource 组件。
- DataField: 设置与 DBEdit 组件对应的数据表中的字段。

2) DBEdit 组件的使用

使用 DBEdit 组件的方法为:选中 DBEdit 组件,在其 DataField 属性值下拉列表框中选择对应的字段名称。当程序运行时,该字段的当前值即可通过 DBEdit 组件显示。

【例 7-2】 使用 Data Controls 面板中的组件对数据表进行简单操作。


以 STUINF 表中的数据为例,其中的学号、姓名和年龄用 DBEdit 组件显示;性别用 DBRadioGroup 组件显示;院系与班级用 DBComboBox 组件显示。具体操作步骤如下:

(1) 执行【例 7-1】中步骤(1)~(2)的操作,建立数据库连接并设置数据库连接组件。

(2) 在 Form 窗体中添加一个 DBNavigator 组件,将 DataSource 属性设置为 DataSource 组件。

(3) 在 Form 窗体中添加 6 个 DBEdit 组件,分别用于显示“学号”、“姓名”、“学号”、“年龄”、“院系”和“班级”文本。将这 6 个组件的 DataSource 属性均设置为 DataSource 组件,将它们的数据 Field 属性分别设置为 STUNO、STUNAME、STUNO、STUAGE、STUDEPT 和 STUCLASS 字段。

(4) 在 Form 窗体中添加一个 DBRadioGroup 组件,将 DataSource 属性设置为 DataSource 组件,将 DataField 属性设置为 STUSEX,用于显示“性别”。将 Columns 属性设

置为 2,表示一行显示两列数据。单击 Items 属性值右边的  按钮,在弹出的对话框中添加两行信息:“男”和“女”。

(5)在 Form 窗体中添加一个 DBGrid 组件,将 DataSource 属性设置为 DataSource 组件,并按照【例 7-1】中步骤(3)的方法设置 DBGrid 组件的属性。

完成设置后,程序界面如图 7-21 所示。可以在程序界面中添加、删除和修改记录,而不用另外编号代码。



图 7-21 【例 7-2】程序界面

7.3 动态数据操作

前面介绍了使用 Data Controls 面板中的组件可以在不编写代码的情况下实现基本的数据操作。但是这样的方法有两个缺点:一个是没有数据查询功能,而这个功能在数据库中是非常重要的;另一个是不能获得当前记录的值,这使得对数据的操作非常不便。为此,Delphi 系统允许直接使用 SQL 语句来动态操作数据。

7.3.1 ADOQuery 组件的方法

ADOQuery 组件是一个功能十分强大的数据操作组件,使用该组件可以在代码中插入 SQL 语句来动态操作数据;此外,ADOQuery 组件还提供了多种方法来直接操作数据,提高了程序设计的效率。下面介绍如何使用 ADOQuery 组件的方法来操作数据库。

ADOQuery 组件中用于进行数据操作的方法主要有以下几个:

- First:将与 ADOQuery 组件连接的数据表中的第一条记录设置为当前记录。
- Prior:将上一条记录设置为当前记录。
- Next:将下一条记录设置为当前记录。
- Last:将最后一条记录设置为当前记录。
- Insert:在数据表中添加一条记录。
- Delete:删除当前记录。

- Post: 将数据表中的数据提交到数据库。
- Refresh: 刷新当前数据表中的数据。

【例 7-3】 以【例 7-2】中的数据为例,使用 ADOQuery 组件实现 DBNavigator 组件的基本功能。

本例要实现的基本功能包括:添加、删除记录,将第一条记录、最后一条记录、上一条记录和下一条记录设置为当前记录以及提交记录。

操作步骤如下:

(1)在 Form 窗体中添加一个 ADOConnection 组件,一个 ADOQuery 组件和一个 DataSource 组件。

将 ADOConnection 组件的 ConnectionString 属性设置为 UDL 文件,Connected 属性和 KeepConnected 属性均设置为 True。

设置 ADOQuery 组件的 Connection 属性为 ADOConnection 组件,SQL 属性设置为“SELECT * FROM STUINF”,Active 属性设置为 True。

将 DataSource 组件的 DataSet 属性设置为 ADOQuery 组件。

(2)在 Form 窗体中添加一个 DBGrid 组件,一个 DBRadioGroup 组件和 5 个 DBEdit 组件,具体设置同【例 7-2】中对应组件的设置。

(3)在 Form 窗体中添加 7 个 Button 组件,创建界面,如图 7-22 所示。



图 7-22 【例 7-3】程序界面

(4)为“上一条”按钮的 OnClick 事件添加如下代码,将上一条记录设置为当前记录。

```
procedure TForm1.Button5Click(Sender: TObject);
begin
```

```
    ADOQuery1.Prior;
```

```
end;
```

(5)为“下一条”按钮的 OnClick 事件添加如下代码,将下一条记录设置为当前记录。

```
procedure TForm1.Button6Click(Sender: TObject);
```

```
begin
```

```
    ADOQuery1.Next;
```

```
end;
```

(6)为“首记录”按钮的 OnClick 事件添加如下代码,将第一条记录设置为当前记录。


```
procedure TForm1.Button3Click(Sender: TObject);  
begin  
    ADOQuery1.First;  
end;
```

(7)为“尾记录”按钮的 OnClick 事件添加如下代码,将最后一条记录设置为当前记录。

```
procedure TForm1.Button4Click(Sender: TObject);  
begin  
    ADOQuery1.Last;  
end;
```

(8)为“添加”按钮的 OnClick 事件添加如下代码,在当前记录前添加一条空记录。

```
procedure TForm1.Button1Click(Sender: TObject);  
begin  
    ADOQuery1.Insert;  
end;
```

(9)为“删除”按钮的 OnClick 事件添加如下代码,删除当前的记录。

```
procedure TForm1.Button2Click(Sender: TObject);  
begin  
    ADOQuery1.Delete;  
end;
```

(10)为“提交”按钮的 OnClick 事件添加如下代码,将缓冲区中的数据提交到数据库。

```
procedure TForm1.Button7Click(Sender: TObject);  
begin  
    ADOQuery1.Post;  
end;
```

7.3.2 在 ADOQuery 组件中使用 SQL 语句

前面介绍了使用 ADOQuery 组件进行处理数据的方法,使用这两种组件,可以在不编写代码的情况下对数据库中的数据进行操作,但也有不足之处,主要表现为对数据操作的功能有限,不能灵活对数据进行操作。为此,可以将 ADOQuery 组件与 SQL 语句结合起来,通过 ADOQuery 组件直接操作 SQL 语句,从而提高数据操作的灵活性。

在 ADOQuery 组件中操作 SQL 语句的操作步骤如下:

(1)关闭 ADOQuery 组件连接,使用的语句为:

```
ADOQuery.Close;
```

(2)清除 ADOQuery 组件中原有的 SQL 语句,使用的语句为:

```
ADOQuery.SQL.Clear;
```

(3)在 ADOQuery 组件中添加新的 SQL 语句,使用的语句为:

```
ADOQuery.SQL.ADD(SQLSTR);
```

(4)打开 ADOQuery 组件连接,使用的语句为:

```
ADOQuery.Open;
```

7.4 数据库程序设计实例

为了帮助读者更好地掌握如何在 Delphi 系统中进行数据库程序设计,本节将制作一个通讯录。要求通讯录具有添加、删除、修改通讯信息和按指定条件查询通讯信息的功能。

7.4.1 数据库设计

在数据库程序设计中,建立数据库与数据表是最基本也是最重要的内容之一。

1. 建立数据库

在建立数据库前应先选择要使用的数据库系统,目前,在中小型应用系统开发中,广泛使用的数据库系统是 SQL Server。这里以 SQL Server 2000 为数据库开发平台进行介绍。

创建数据库的操作步骤如下:

(1)启动 SQL Server 数据库系统,进入企业管理器,如图 7-23 所示。



图 7-23 SQL Server 数据库中的企业管理器

(2)在左侧的控制台根目录中右击“数据库”选项,在弹出的快捷菜单中选择“新建数据库”命令,弹出“数据库属性”对话框。在“名称”文本框中输入数据库的名称,这里输入 MyDataBase,然后单击“确定”按钮,完成数据库的建立。

2. 创建数据表

建立好数据库后,还需要在数据库中创建数据表。

1) 确定数据表格式

在创建数据表之前,应先确定数据表的结构,包括表中各个字段的名称、含义、数据类型以及是否可以空等。本例的数据表结构见表 7-3。

表 7-3 通讯信息表的数据结构

字段名称	含 义	数据类型	说 明
NAME	姓名	varchar	主键
SEX	性别	smallint	0:男,1:女,不能为空

续表

字段名称	含 义	数据类型	说 明
GRP	分组	varchar	不能为空
MOBILE	手机号码	varchar	可以为空
HOME	住宅电话	varchar	可以为空
BUSINESS	办公电话	varchar	可以为空
EMAIL	邮箱地址	varchar	可以为空
ADDRESS	联系地址	varchar	可以为空

2) 建立数据表

确定数据表的结构后,接下来建立数据表,具体操作步骤如下:

(1)在企业管理器的控制台根目录中展开前面创建的 MyDataBase 数据库,如图 7-24 所示。



图 7-24 展开 MyDataBase 数据库

(2)右击“表”选项,在弹出的快捷菜单中选择“新建表”命令,打开表设计窗口,建立数据表,如图 7-25 所示。

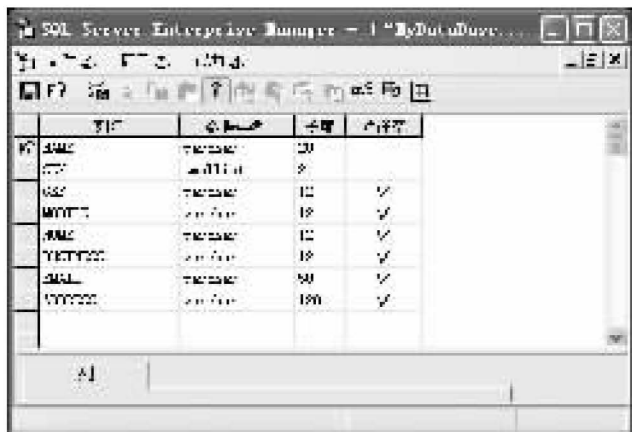


图 7-25 建立数据表

(3)右击 NAME 字段所在的行,在弹出的快捷菜单中选择“设置主键”命令,将 NAME 字段设置为主键。

(4)单击工具栏中的“保存”按钮,在弹出的“选择名称”对话框中输入数据表的名称,如图 7-26 所示。单击“确定”按钮,完成数据表的创建。



图 7-26 输入数据表的名称

3. 创建 UDL 文件

要想使用组件连接数据库,需要先建立一个用于数据库连接的 UDL 文件,其操作步骤如下:

(1)打开 Windows 中的记事本,执行“文件”→“另存为”命令,打开“另存为”对话框。将保存类型设置为“所有文件”,将文件名设置为 link.udl,单击“保存”按钮,完成 UDL 文件的创建。

(2)双击刚刚建立的 UDL 文件,打开“数据链接属性”对话框,单击“提供程序”选项卡。

(3)选择 Microsoft OLE DB Provider for SQL Server 选项后单击“下一步”按钮,打开“连接”选项卡。设置服务器名称为本机名称,登录服务器的信息为“使用 Windows NT 集成安全设置”。选中“在服务器上选择数据库”单选按钮,在下拉列表框中选择前面创建的数据库 MyDataBase,如图 7-27 所示。然后单击“测试连接”按钮,如果设置成功,则弹出提示成功的对话框。



图 7-27 设置数据链接属性

7.4.2 程序功能的实现

建立数据库并测试成功后,下面进行程序功能模块设计。本程序可以分为数据显示、数据输入、数据查询、数据修改和数据删除几个模块。

1. 数据显示模块


在 Delphi 程序设计中,使用 DBGrid 组件来显示数据表中的数据。为了方便起见,通常单独为 DBGrid 组件关联一个 ADODataSet 组件,使其独立完成数据的显示,操作步骤如下:

(1) 在 Form 窗体中添加一个 ADOConnection 组件,一个 ADODataSet 组件,一个 DataSource 组件和一个 DBGrid 组件。

(2) 选中 ADOConnection 组件,将 ConnectionString 属性设置为 UDL 文件,将 Connected 属性和 KeepConnected 属性设置为 True。

(3) 选中 ADODataSet 组件,将 Connection 属性设置为 ADOConnection 组件。在 CommandText 属性中添加 SQL 语句:SELECT * FROM COMMINF,并将 Active 属性设置为 True。

(4) 选择 DataSource 组件,将 DataSet 属性设置为 ADODataSet 组件。

(5) 选中 DBGrid 组件,将 DataSource 属性设置为 DataSource 组件。选中 Columns 属性,单击属性值右边的  按钮,在弹出的对话框中单击 Add All Fields 按钮,将所有字段添加到 DBGrid 组件中显示。

(6) 依次选中 DBGrid 组件中的 Columns 对象,将 Title 属性集中的 Caption 属性设置为相应字段的名称。此时的界面如图 7-28 所示。

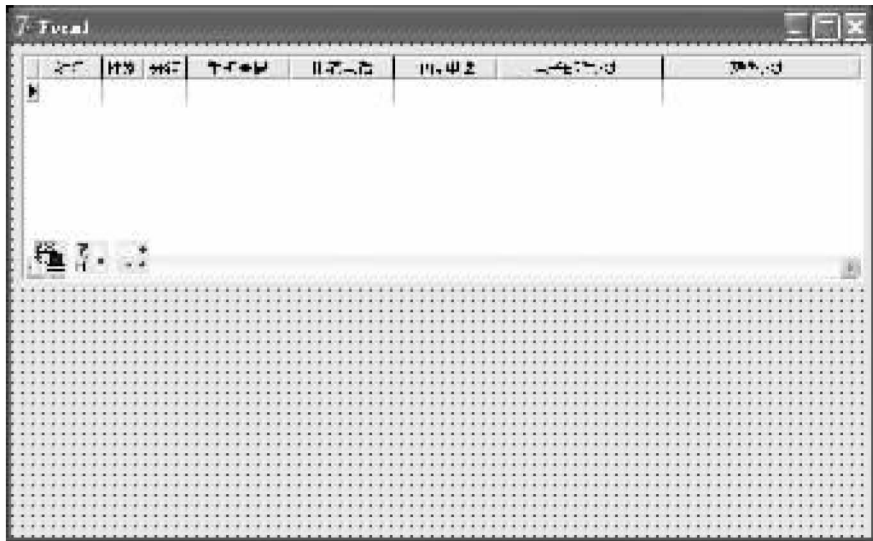


图 7-28 DBGrid 组件界面

(7) 要实现数据库的增加、修改、删除与查询数据功能,还需要添加进行数据操作的组件。添加组件后的界面如图 7-29 所示。

(8) 在 Form 窗体的 OnCreate 事件中添加如下代码,将通讯录中的联系人分为家人、朋友、同事、同学、网友和其他几类,并将朋友作为默认分类。

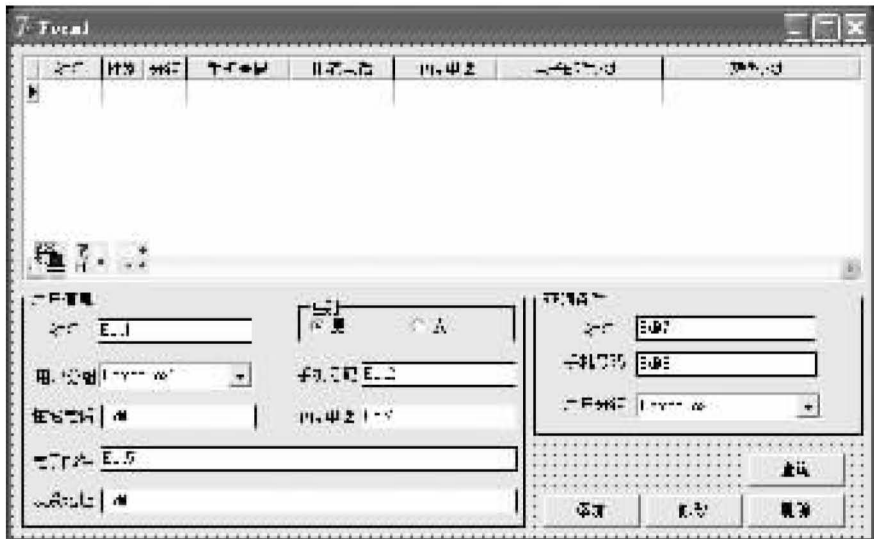


图 7-29 数据显示模块界面

```

procedure TForm1.FormCreate(Sender: TObject);
begin
    ComboBox1.Items.Add('家人');
    ComboBox1.Items.Add('朋友');
    ComboBox1.Items.Add('同事');
    ComboBox1.Items.Add('同学');
    ComboBox1.Items.Add('网友');
    ComboBox1.Items.Add('其他');
    ComboBox1.ItemIndex := 1;
    ComboBox2.Items.Add('家人');
    ComboBox2.Items.Add('朋友');
    ComboBox2.Items.Add('同事');
    ComboBox2.Items.Add('同学');
    ComboBox2.Items.Add('网友');
    ComboBox2.Items.Add('其他');
    ComboBox2.ItemIndex := 1;
end;

```

2. 数据输入模块

使用 ADOQuery 组件作为动态调用 SQL 语句的组件。在 Form 窗体中添加一个 ADOQuery 组件和一个 DataSource 组件,将 ADOQuery 组件的 Connection 属性设置为 ADOConnection 组件,将 DataSource 组件的 DataSet 属性设置为 ADOQuery 组件。

选中界面中的“添加”按钮,在其 OnClick 事件中添加如下代码,实现在数据库中添加记录的功能。

```

procedure TForm1.Button1Click(Sender: TObject);

```

```
var
    SQLSTR: string;
begin
    if Edit1.Text = '' Then
        begin
            MessageDlg('名称不能为空!', mtWarning, [mbOK], 0);
            Edit1.SetFocus;
        end
    else begin
        SQLSTR := 'insert into COMMINF values(:iname, :isex, :igroup, :imobile, :
            ihome, :ibusiness, :imail, :iaddr)';
        ADOQuery1.Close;
        ADOQuery1.SQL.Clear;
        ADOQuery1.SQL.Add(SQLSTR);
        ADOQuery1.Parameters.ParamByName('iname').Value := Edit1.Text;
        ADOQuery1.Parameters.ParamByName('isex').Value :=
            RadioGroup1.ItemIndex;
        ADOQuery1.Parameters.ParamByName('igroup').Value :=
            ComboBox1.Items[ComboBox1.ItemIndex];
        ADOQuery1.Parameters.ParamByName('imobile').Value := Edit2.Text;
        ADOQuery1.Parameters.ParamByName('ihome').Value := Edit3.Text;
        ADOQuery1.Parameters.ParamByName('ibusiness').Value := Edit4.Text;
        ADOQuery1.Parameters.ParamByName('imail').Value := Edit5.Text;
        ADOQuery1.Parameters.ParamByName('iaddr').Value := Edit6.Text;
        ADOQuery1.ExecSQL;
        DBGrid1.DataSource := DataSource1;
        ADODataset1.Close;
        ADODataset1.Open;
        ADODataset1.Last;
        Edit1.Text := '';
        Edit2.Text := '';
        Edit3.Text := '';
        Edit4.Text := '';
        Edit5.Text := '';
        Edit6.Text := '';
    end;
end;
```

3. 数据查询模块

先设置查询条件,其中,姓名的优先级最高,其次是手机号码,最后是分类。当姓名不为空时,以姓名作为查询条件;当姓名为空而手机号码不为空时,以手机号作为查询条件;只有

当姓名与手机号码均为空时,才将分类作为查找条件。

选中“查询”按钮,在其 OnClick 事件中添加如下代码,实现查询功能。

```
procedure TForm1.Button4Click(Sender: TObject);
begin
    ADOQuery1.Close;
    ADOQuery1.SQL.Clear;
    if Edit7.Text <> '' then
    begin
        ADOQuery1.SQL.Add('select * from COMMINF where NAME = :iname');
        ADOQuery1.Parameters.ParamByName('iname').Value := Edit7.Text;
    end
    else if Edit8.Text <> '' then
    begin
        ADOQuery1.SQL.Add('select * from COMMINF where MOBILE
            = :imobile');
        ADOQuery1.Parameters.ParamByName('imobile').Value := Edit8.Text;
    end
    else begin
        ADOQuery1.SQL.Add('select * from COMMINF where GRP = :igroup');
        ADOQuery1.Parameters.ParamByName('igroup').Value :=
            ComboBox2.Items[ComboBox2.ItemIndex];
    end;
    DBGrid1.DataSource := DataSource2;
    ADOQuery1.Open;
end;
```

4. 显示查询结果

查询完毕后,在 DBGrid 组件中显示符合查询条件的记录,这些记录可能多于一条。双击某条记录,可以将该记录显示在“用户信息”选项区中对应的组件上。实现该功能的方法为:选中 DBGrid 组件,在其 OnDBClick 事件中添加如下代码:

```
procedure TForm1.DBGrid1DBClick(Sender: TObject);
begin
    Edit1.Text := DBGrid1.DataSource.DataSet.FieldByName('NAME').AsString;
    RadioGroup1.ItemIndex :=
        DBGrid1.DataSource.DataSet.FieldByName('SEX').AsInteger;
    ComboBox1.Text := DBGrid1.DataSource.DataSet.FieldByName('GRP').AsString;
    Edit2.Text := DBGrid1.DataSource.DataSet.FieldByName('MOBILE').AsString;
    Edit3.Text := DBGrid1.DataSource.DataSet.FieldByName('HOME').AsString;
    Edit4.Text := DBGrid1.DataSource.DataSet.FieldByName('BUSINESS').AsString;
    Edit5.Text := DBGrid1.DataSource.DataSet.FieldByName('EMAIL').AsString;
    Edit6.Text := DBGrid1.DataSource.DataSet.FieldByName('ADDRESS').AsString;
```



```
end;
```

5. 修改当前数据

可以在“用户信息”选项区中以“姓名”为修改条件,修改除“姓名”外的所有数据,然后将修改后数据提交到数据库。实现该方法的方法为在“修改”按钮的 OnClick 事件中添加如下代码:

```
procedure TForm1.Button2Click(Sender: TObject);
begin
    ADOQuery1.Close;
    ADOQuery1.SQL.Clear;
    ADOQuery1.SQL.Add('update COMMINF set sex = :isex, grp = :
        igroup, mobile = :imobile,');
    ADOQuery1.SQL.Add(' home = :ihome, business = :ibusiness, email = :
        iemail, address = :iaddress');
    ADOQuery1.SQL.Add(' where name = :iname');
    ADOQuery1.Parameters.ParamByName('iname').Value := Edit1.Text;
    ADOQuery1.Parameters.ParamByName('isex').Value := RadioGroup1.ItemIndex;
    ADOQuery1.Parameters.ParamByName('igroup').Value := ComboBox1.Items
        [ComboBox1.ItemIndex];
    ADOQuery1.Parameters.ParamByName('imobile').Value := Edit2.Text;
    ADOQuery1.Parameters.ParamByName('ihome').Value := Edit3.Text;
    ADOQuery1.Parameters.ParamByName('ibusiness').Value := Edit4.Text;
    ADOQuery1.Parameters.ParamByName('iemail').Value := Edit5.Text;
    ADOQuery1.Parameters.ParamByName('iaddress').Value := Edit6.Text;
    ADOQuery1.ExecSQL;
    DBGrid1.DataSource := DataSource1;
    ADODataSet1.Close;
    ADODataSet1.Open;
end;
```

6. 删除指定信息

在“删除”按钮的 OnClick 事件中添加如下代码,实现删除指定联系人的功能。

```
procedure TForm1.Button3Click(Sender: TObject);
begin
    if Edit1.Text = '' then
    begin
        MessageDlg('请先选择信息!', mtError, [mbOK], 0);
        Edit1.SetFocus;
    end
    else begin
        ADOQuery1.Close;
```

```

ADOQuery1.SQL.Clear;
ADOQuery1.SQL.Add('delete from COMMINF where name = ;iname');
ADOQuery1.Parameters.ParamByName('iname').Value := Edit1.Text;
ADOQuery1.ExecSQL;
DBGrid1.DataSource := DataSource1;
ADODataset1.Close;
ADODataset1.Open;
    Edit1.Text := '';
    Edit2.Text := '';
    Edit3.Text := '';
    Edit4.Text := '';
    Edit5.Text := '';
    Edit6.Text := '';
end;
end;
至此,程序主体功能设计完毕,程序运行效果如图 7-30 所示。

```

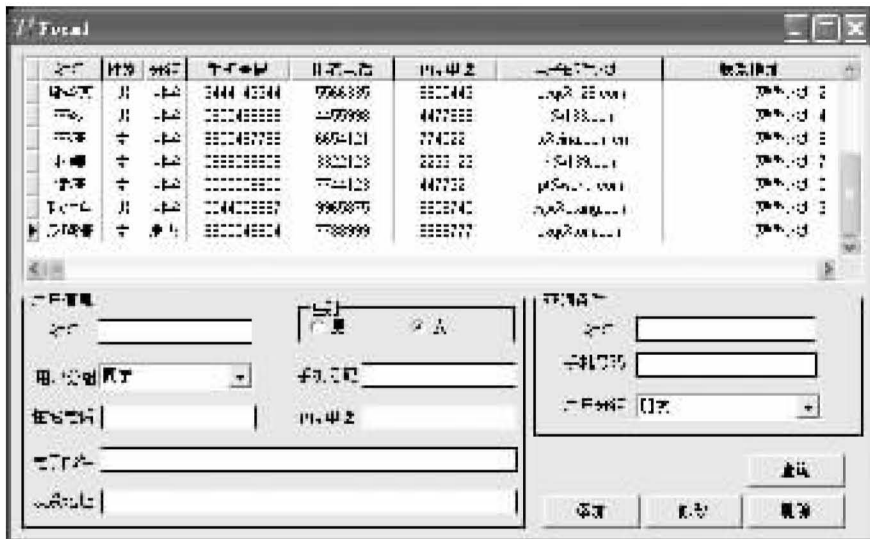


图 7-30 通讯录运行效果

本章小结

本章主要介绍了如何在 Delphi 系统中进行数据库程序设计。先使用 UDL 文件来建立 ADO 连接,然后使用 ADOConnection 组件与数据库连接,再将 ADOQuery 组件与 ADOConnection 组件进行关联,最后将 DataSource 组件与 ADOQuery 组件进行关联,完成数据库与应用程序的连接。

当数据库与应用程序建立连接后,可以使用多种数据操作组件来处理数据,如使用 DBGrid 组件以表格形式显示数据表中的数据。

习 题 7

一、选择题

- 当前的主流数据库模型是()。
 - 关系数据库
 - 网络数据库
 - 本地数据库
 - 非关系数据库
- 下列不属于数据库基本操作的是()。
 - 添加数据
 - 删除数据
 - 维护数据
 - 查询数据
- 将数据库中的数据通过 DBGrid 组件显示时,依次使用到的组件是()。
 - ADOConnection→DataSource→ADODataSet→DBGrid
 - ADOConnection→ADO DataSet→DataSource →DBGrid
 - ADO DataSet→ADOConnection→DataSource→DBGrid
 - DataSource→ADOConnection→ADODataSet→DBGrid
- 在查询语句中添加()可以删除重复的记录。
 - DISTIACT
 - AVG
 - SUM
 - ORDER
- 下列关于 DBEdit 组件与 Edit 组件的描述,说法错误的是()。
 - Edit 组件可以输入数据,DBEdit 组件不能输入数据
 - Edit 组件不能关联数据,DBEdit 组件可以关联数据
 - Edit 组件有 Text 属性,DBEdit 组件没有 Text 属性
 - 可以直接读取 Edit 组件中显示的数据,不能直接读取 DBEdit 组件中显示的数据

二、简答题

- 简述建立 UDL 文件的步骤。
- 比较有条件查询与无条件查询的区别。

三、程序设计题

建立一个名为 Test 的数据库,在数据库中建立学生信息表。学生信息表的数据结构见表 7-4。要求数据库能实现学生信息的增加、修改、删除和查询功能。

表 7-4 学生信息表

字段名称	数据类型	字段说明
学号	char	主键
姓名	varchar	不可为空
性别	char	不可为空
班级	varchar	不可为空
年龄	int	不可为空
籍贯	varchar	不可为空