第 15 章　Visual Basic 数据库编程

前面介绍了数据库编程的相关基础知识，本章将具体介绍 Visual Basic 6.0 是如何实现数据库应用系统设计的。除了前面章节介绍的 DAO、RDO 和 ADO 控件外，Visual Basic 6.0 还提供了几个数据控件，用于对数据进行显示和操作。本章将就使用 Visual Basic 6.0 提供的 ADO 控件和数据控件实现数据库应用系统进行详细讲解。

15.1　操作记录集

前面章节在介绍 ADO 控件时提到了 Connection 连接对象和 RecordSet 记录集对象。其中，Connection 对象用于实现与数据库的连接，而 RecordSet 对象用于实现对数据库中数据表的操作。配置记录集可通过 ADO 控件的 RecordSource 属性来实现。那么配置完成后，如何对记录集进行操作，以完成对数据的操作呢？这就是本节要介绍的具体内容。

15.1.1　打开记录集

如前所述，记录集返回的是一个从数据库取回的查询结果集，代表一组与数据库相关的逻辑记录集合。它所对应的数据来源可以是数据表，也可以是和 SQL 语言中查询语句（SELECT）有关的查询结果。Recordset 对象的类型有如下 3 种。

- Table 类型：记录集为表集，可以显示、添加、删除和修改，具有较好的更新性能。
- Dynaset 类型：记录集为动态集，可以显示、添加、删除和修改，并具有较大的操作灵活性。
- Snapshot 类型：记录集为快照集，只能显示，具有较好的显示速度。

除了前面提到的通过配置 RecordSource 属性来对记录集的内容进行赋值外，还有两种方法可以打开对记录集的操作：使用记录集的 Open 方法和使用 Connection 对象的 Execute 方法。

使用记录集的 Open 方法可以打开一个记录集，Open 方法的语法格式如下。

```
Recordset.Open Source, ActiveConnection, CursorType, LockType, Options
```

其中，Recordset 参数为所定义的记录集对象的实例，即记录集名称；Source 参数是一个可选项，指明了所打开的记录源信息，Source 参数可以是合法的命令、对象变量名、SQL 语句、表名、存储过程调用或保存记录集的文件名；ActiveConnection 参数也是可选项，表示合法的已打开的 Connection 对象的变量名，或者是包含 ConnectionString 参数的字符串；CursorType 即游标类型，用来确定打开记录集对象使用的指针类型。通常，Visual Basic 6.0 中支持的游标类型有如下 4 种。

- 0 – (adOpenForwardOnly)：只能在 Recordset 的记录中向前移动，但速度最快。

- 1－(adOpenKeyset)：可以在 Recordset 中任意移动，其他用户所做的记录修改可见，但其他用户添加的记录不可见，删除的记录字段值不能被使用。
- 2 － (adOpenDynamic)：可以在 Recordset 中任意移动，其他用户增、删、改的记录都可见，但速度最慢。
- 3－(adOpenStatic)：可以在 Recordset 中任意移动，其他用户增、删、改的记录都不可见。

此外，LockType 参数表示并发控制的类型，用于确定打开记录集对象使用的锁定类型。同样，在 Visual Basic 6.0 中，支持的并发控制类型有如下 4 种。

- 0－ (adLockReadOnly)：Recordset 的记录为只读。
- 1－(adLockPessimistic)：只要保持 Recordset 为打开，其他用户就无法编辑该记录集中的记录。
- 2 － (adLockOptimistic)：当使用 update 命令对 Recordset 中的记录进行更新时，将记录加锁。
- 3－ (adLockBatchOptimistic)：以批模式更新记录时加锁。

最后的 Options 参数则指定 Source 参数传递命令的类型，根据上一章节中配置 RecordSet 对象所述，该参数包含如下 4 种类型。

- 1－ (adCmdText)：SQL 语句。
- 2－ (adCmdTable)：数据表的名字。
- 4－ (adCmdStoredProc)：存储过程。
- 8－ (adCmdUnknown)：未知类型。

例如，在事件代码框中输入如下语句，表示定义了 Connection 对象和 RecordSet 对象，并指定了 Connection 对象连接的数据库是当前目录下的 Access 数据库 student。此外，该语句指定了记录集所包含的对象是一条 SQL 查询语句 select * from t_user 的返回结果，最后一条语句 rs.Open sql, cn, 1, 1 则以锁定的方式打开了该记录集，代码如下。

```
Dim cn As ADODB.Connection
Dim rs As ADODB.Recordset
Dim sql As Variant
Set cn = New ADODB.Connection
Set rs = New ADODB.Recordset
cn.Open "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=" & App.Path & "\student.mdb;Persist
Security Info=False"
sql = "select * from t_user "
rs.Open sql, cn, 1, 1
```

此外，Visual Basic 6.0 还提供了另外一种打开记录集的方法，即使用 Connection 对象的 Execute 方法来打开。Connection 对象的 Execute 方法的语法格式如下。

```
Set recordset=Connection.Execute(CommandText,RecordsAffected,Options)
```

其中的参数说明如下。

- CommandText：一个字符串，返回要执行的 SQL 命令、表名、存储过程或指定文本。
- RecordsAffected：可选项，Long 类型的值，返回操作影响的记录数。
- Options：可选项，Long 类型值，指明如何处理 CommandText 参数。

例如，使用 Connection 对象的 Execute 方法来打开上例 Access 数据库 student 中的符合

条件的 t_user 表的记录，可使用如下语句来实现。

```
Dim cn As ADODB.Connection
Set cn = New ADODB.Connection
cn.Open "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=" & App.Path & "\student.mdb;Persist Security Info=False"
Set rs=cn.Execute("select * from t_user where username='" & Text1.Text & "'")
```

上述语句的结果与使用记录集的 Open 方法功能相同，也可以选中特定 SQL 查询语句中的结果作为记录集的内容。

至此，记录集对象就被打开了，可以对该记录集中的数据进行相关的显示、添加、更新和删除等操作了，下面将一一介绍这些操作的实现。

15.1.2 添加新记录

记录集的使用不外乎就是对记录的操作，主要包括添加记录、删除记录、修改记录和查询记录，下面分别对其进行简要介绍。

在记录集中添加新的记录可以采用记录集 RecordSet 的 AddNew 方法来实现，AddNew 方法的语法格式如下。

```
Recordset.AddNew FieldList, Values
```

其中，参数说明如下。

- Recordset 为记录集对象实例。
- FieldList 为一个字段名，或者是一个字段数组。
- Values 为给要添加信息的字段赋的值，如果 FiledList 为一个字段名，那么 Values 应为一个单个的数值。假如 FiledList 为一个字段数组，那么 Values 也必须为一个个数、类型与 FieldList 相同的数组。

注意：在使用完 AddNew 方法为记录集添加新的记录后，应使用 UpDate 方法将所添加的数据存储在数据库中，否则，添加的新记录将不能被写入到数据库中。

为了使读者更好地理解记录集在数据库应用程序中的使用，下面给出一个具体的窗体实例进行演示。该实例使用 Access 数据库 student.mdb 为后台，该数据库中包含了数据表 t_user，该表的结构和数据记录如图 15-1 所示。

图 15-1 数据表记录

该实例将用户在窗体上对应 TextBox 文本框中的输入，作为一条记录添加到数据表 user 中。其实现步骤如下。

（1）新建一个“标准 EXE”工程，为该工程的窗体上添加 6 个 Label 标签控件、5 个 TextBox 文本框控件和 2 个 CommandButton 按钮控件。

（2）设置窗体和控件的属性，如表 15-1 所示。

表 15-1　控件属性设置

控件	属性	设置值
Label1	Caption	Label1.Caption="添加新记录"
Label2	Caption	Label1.Caption="学号"
Label3	Caption	Label1.Caption="姓名"
Label4	Caption	Label1.Caption="性别"
Label5	Caption	Label1.Caption="年龄"
Label6	Caption	Label1.Caption="系部"
Label1	Font	Label1.Font="楷体，三号"
Text1	Text	Text1.Text=""
Command1	Caption	Command1.Caption="添加"
Command2	Caption	Command2.Caption="取消"

这些属性设置完成后，可以在 Visual Basic 6.0 的窗体设计环境下拖动这些控件，使之看起来更为美观，如图 15-2 所示。

（2）添加 ADO 引用。由于该实例中要使用到 ADO 作为数据库的编程接口，因此需要将 ADO 对象引用添加到该工程。选择“工程”→“引用”命令，打开“引用”对话框，如图 15-3 所示，选择其中的“Microsoft ActiveX Data Objects 2.0 Library”选项后，单击“确定”按钮即可。

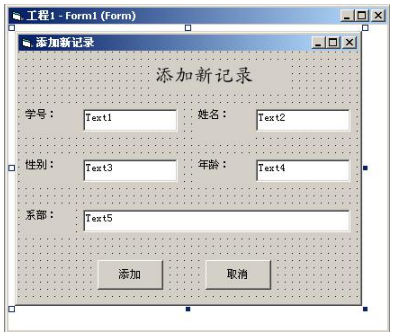

图 15-2　窗体设计

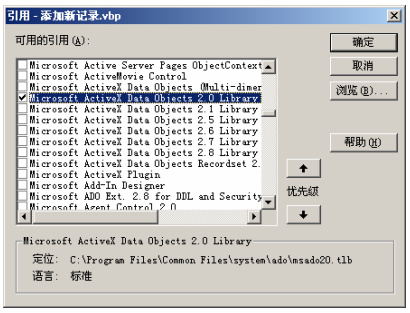

图 15-3　添加 ADO 引用

（3）添加事件代码。该实例要求用户单击“添加”按钮后，将文本框内的值写入数据表 user 中。因此，在“添加”按钮中写入如下事件代码。

```
Private Sub Command1_Click()
Dim CN As New ADODB.Connection                    ‘定义 Connection 对象
Dim rs As New ADODB.Recordset                    ‘定义 RecordSet 对象
Dim rs1 As New ADODB.Recordset
Dim sql As String
Dim sql1 As String
Dim temp As String
temp = Trim(Text1.Text)
CN.Open "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=" & App.Path & "\student.mdb;Persist
Security Info=False"
   sql = "select * from t_user where sno='" & temp & "'"
```

```
    rs.Open sql, CN, 1, 1                                    '只读打开记录集
    If rs.EOF <> True Then                                   '该学号已存在
      MsgBox "该学号已存在，请更换", 16+256, "错误"
    Else
      sql = "select * from t_user"
    rs1.Open sql, CN, 3, 2                                   '可擦写打开记录集
rs1.AddNew                                                   '添加方法
rs1!sno = temp
rs1!sname = Trim(Text2.Text)
rs1!ssex = Trim(Text3.Text)
rs1!sage = Trim(Text4.Text)
rs1!sdept = Trim(Text5.Text)
    rs1.Update                                               '写入数据库
      MsgBox "成功添加一条记录", , "提示"
    End If
End Sub
```

从上述代码可看出，其首先定义了 Connection 对象和 RecordSet 对象，接下来打开了一个 SQL 查询语句的记录集 rs，用于对用户输入的学号进行判断，查看是否在数据库中存在同样的学号。如果存在则提示错误，否则以可擦写的方式打开另一个记录集 rs1，并调用 AddNew 方法，将用户输入到文本框中的内容写入到记录集 rs1 所对应的字段中。最后以 Update 方法写入到数据库中，并给出添加成功的提示。

可以看出，记录集调用其中字段采用的是符号“!”，如 rs1!sno 表示记录集 rs1 中的字段 sno。事实上，也可以通过 rs1("sno")来调用，其效果相同。

（4）运行程序。确认 student 数据库在当前工程所在的目录下，运行该窗体，运行结果如图 15-4 所示。

在对应的文本框中分别输入要添加的学生的基本信息后，单击“添加”按钮，执行结果如图 15-5 所示。

图 15-4　运行结果

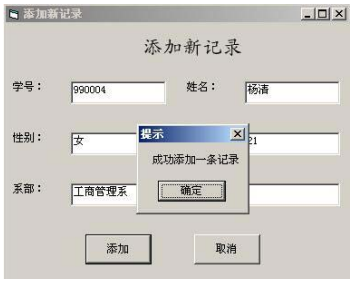

图 15-5　单击“添加”按钮

此时，可以打开 Access 数据库，查看其 t_user 数据表，看到该数据表已经增加了一条记录在最后，如图 15-6 所示。

图 15-6　添加后的数据表记录

此外，如果用户输入的学号与原有数据库中的学号重复，则显示错误提示，记录将不会写入到数据库中，如图 15-7 所示。

此外，上述代码中的数据添加较为直观，也可以使用在前面章节中提到的 SQL 语言中的插入语句，代码如下。

```
sql = "insert into student values( temp,Trim(Text2.Text),Trim(Text3.Text),Trim(Text4.Text),
Trim(Text5.Text)"
CN.Execute (sql)
```

注意：上述实例中的数据插入是很简单的，没有考虑数据的类型，因为会导致程序的健壮性较差。例如，当用户在“年龄”字段中输入的并不是数字，而是字符，那么将这个数据插入到数据表中的时候就会出现错误。因为数据表定义时，该字段的数据类型定义是数值型，而插入的数据为字符型，这就导致了数据的不一致性，将导致程序错误，异常退出，如图 15-8 所示。

图 15-7　学号重复

图 15-8　类型不匹配

因此，在具体的程序设计中，应充分注意到这些问题。解决方法为，在将数据插入到数据库前判断用户的输入是否合法。针对上述情况，增加如下语句即可。

```
If IsNumeric(Trim(Text4.Text)) = False Then                    '使用函数判断用户输入
  MsgBox "年龄字段必须为数字，请重新输入"
Else
  继续程序段
End If
```

另外，针对一些数据表中定义时指明不能为空值的字段（如主键），也需要增加判断语句，否则，插入数据库时也将出现错误，同样增加如下语句。

```
If Trim(Text1.Text))= "" Then                                  '判断用户输入是否为空
  MsgBox "学号不能为空"
Else
  继续程序段
End If
```

上述判断语句也可写在输入文本框控件 TextBox 的 LostFocus 事件中，同样可以触发。针对具体的约束条件，需要增加不同的判断语句来约束用户的输入。如果没有这些语句，程序的健壮性就较差，当用户的输入出现错误时，程序很可能会异常退出。

至此，使用 RecordSet 记录集对象进行添加新记录的操作就完成了。可以看出，添加新记录的关键语句在于 AddNew 方法和 Update 方法。在进行记录的添加操作中，还需要考虑到

数据的类型以及程序的健壮性。

15.1.3 显示记录

显示记录指通过使用 RecordSet 记录集对象提供的 Move 方法，实现对数据表中所有记录的遍历操作。例如，下面实例通过 ADO 连接数据库，使用代码控制数据在 5 个 TextBox 文本框控件上的显示。其实现步骤如下。

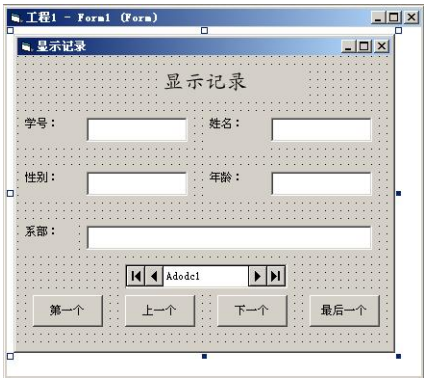

图 15-9　窗体设计

（1）窗体设计。与上述添加新记录的实例类似的，该实例新建一个“标准 EXE”工程后，设计一个如图 15-9 所示的窗体

此处窗体中添加了 Adodc 控件，其在标准控件栏中并不存在，需用户选择“工程”→“部件”命令，选择“Microsof ADO Data Control”选项添加到控件栏中，再拖动到窗体上。需要设置的属性如表 15-2 所示。

表 15-2　控件属性设置

控件	属性	设置值
Adodc1	Visibled	False
Form1	Caption	Form1.Caption="显示记录"
Command1	Caption	Command1.Caption="第一个"
Command2	Caption	Command1.Caption="上一个"
Command3	Caption	Command1.Caption="下一个"
Command4	Caption	Command1.Caption="最后一个"

（2）编写事件代码。由于该实例要求窗体运行后就在对应的文本框中显示出数据表 t_user 中的数据，因此应该在窗体装载时驱动事件，显示第一条记录的代码应写在 Load 事件中，代码如下。

```
Dim CN As New ADODB.Connection
Dim rs As New ADODB.Recordset
Dim sql As String
Private Sub Form_Load()
CN.Open "Provider=Microsoft.Jet.OLEDB.4.0;Data Source= " & App.Path & "\student.mdb;Persist
Security Info=False"
   sql = "select * FROM t_user "                                    '查询
   rs.Open sql, CN, 1, 1                                            '打开记录集
Text1.Text = rs("sno")                                              '数据显示在控件中
Text2.Text = rs("sname")
Text3.Text = rs("ssex")
Text4.Text = rs("sage")
Text5.Text = rs("sdept")
End Sub
```

```
Private Sub Command1_Click()                    '转向第一条记录
rs.MoveFirst
Text1.Text = rs("sno")                          '数据显示在控件中
Text2.Text = rs("sname")
Text3.Text = rs("ssex")
Text4.Text = rs("sage")
Text5.Text = rs("sdept")
End Sub
Private Sub Command2_Click()                    '转向上一条记录
rs.MovePrevious
Text1.Text = rs("sno")                          '数据显示在控件中
Text2.Text = rs("sname")
Text3.Text = rs("ssex")
Text4.Text = rs("sage")
Text5.Text = rs("sdept")
End Sub
Private Sub Command3_Click()                    '转向下一条记录
rs.MoveNext
Text1.Text = rs("sno")                          '数据显示在控件中
Text2.Text = rs("sname")
Text3.Text = rs("ssex")
Text4.Text = rs("sage")
Text5.Text = rs("sdept")
End Sub
Private Sub Command4_Click()                    '转向最后一条记录
rs.MoveLast
Text1.Text = rs("sno")                          '数据显示在控件中
Text2.Text = rs("sname")
Text3.Text = rs("ssex")
Text4.Text = rs("sage")
Text5.Text = rs("sdept")
End Sub
```

上述代码中，在 Load 事件中打开了记录集，并将数据表 t_user 中的第一条记录显示在窗体对应的文本框控件中，在各个按钮控件中使用了 MoveFirst 等方法来实现记录的移动，从而达到显示不同记录的功能。

（3）运行程序。运行该窗体，运行结果如图 15-10 所示，单击“下一个”按钮、“最后一个”按钮、“上一个”按钮和“第一个”按钮可实现显示不同记录的结果。单击“最后一个”按钮后的显示结果如图 15-11 所示。

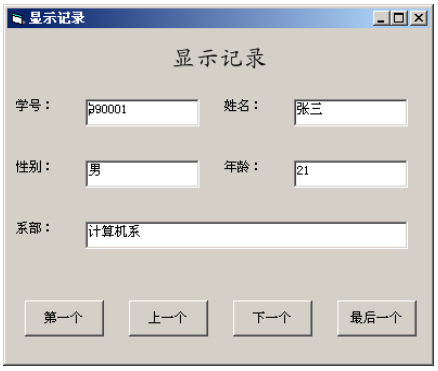

图 15-10　初始运行结果

图 15-11　单击“最后一个”按钮

上述实例中，使用了 ADO 控件 ADODC，因此没有通过引用 ADO 对象也能够完成对记录集的操作。读者可自行尝试使用 ADO 对象来完成相同的功能。

此外，从该实例可以看出，RecordSet 记录集提供的用于显示记录的 Move 方法包括如下 5 种，其中最后一个方法用于移动 *n* 条记录。

- MoveFirst 方法：移至第一条记录。
- MoveLast 方法：移至最后一条记录。
- MoveNext 方法：移至下一条记录。
- MovePrevious 方法：移至上一条记录。
- Move[n]方法向前或向后移动 *n* 条记录，*n* 为指定的数值。

15.1.4 查询记录

在 ADO 中查询记录的方法很灵活，通常有如下两种查询方法。

- 使用连接对象的 Execute 方法执行 SQL 命令，返回查询记录集。
- 使用 Command 对象的 Execute 方法，执行 CommandText 属性中设置的 SQL 命令，返回查询记录集。

第一种方法的具体语法在前面数据连接时已经介绍过了，第二种方法使用 Command 对象的 Execute 方法，其语法格式如下。

```
Command.Execute RecordsAffected, Parameters, Options
```

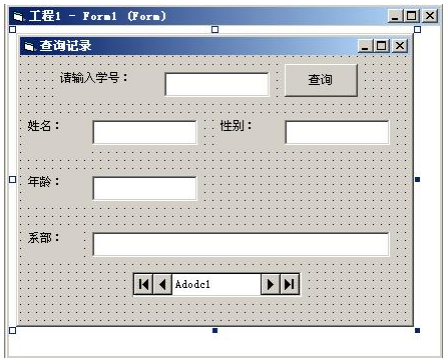

图 15-12　窗体设计

此外，在 Visual Basic 6.0 中通过 ADO 控件进行记录查询操作时，可以通过比较操作来实现。例如，在下列窗体中，用户输入需要查询的学号，如数据库中存在该学号的记录，则显示出该学号对应的信息，否则给出提示信息。其实现步骤如下。

（1）窗体设计。与上述添加新记录的实例类似，该实例新建一个“标准 EXE”工程后，设计一个如图 15-12 所示的窗体

注意：此处需将显示数据的 4 个 TextBox 文本框控件的 Enabled 属性值设置为 False，提示用户这里显示的数据是不可修改的。该实例中需要设置的属性如表 15-3 所示。

表 15-3　控件属性设置

控件	属性	设置值
Adodc1	Visibled	False
Form1	Caption	Form1.Caption="查询记录"
Command1	Caption	Command1.Caption="查询"
Text2	Enable	Text2.Enable=False
Text3	Enable	Text3.Enable=False

续表

控件	属性	设置值
Text4	Enable	Text4.Enable=False
Text5	Enable	Text5.Enable=False

（2）编写事件代码。由于该实例中也同样添加了 ADODC 控件，因此不需要再引用 ADO 对象了。设计窗体完成后，在“查询”按钮中写入如下事件代码。

```
Private Sub Command1_Click()
Dim CN As New ADODB.Connection
Dim rs As New ADODB.Recordset
Dim sql As String
Dim temp As String
temp = Trim(Text1.Text)                                    '获取用户输入并去除空格
CN.Open "Provider=Microsoft.Jet.OLEDB.4.0;Data Source= " & App.Path & "\student.mdb;Persist
Security Info=False"
   sql = "select * FROM t_user where sno='" & temp & "'"
   rs.Open sql, CN, 1, 1
   If rs.EOF <> True Then
     Text2.Text = rs("sname")
     Text3.Text = rs("ssex")
     Text4.Text = rs("sage")
     Text5.Text = rs("sdept")
   Else
     MsgBox "没有找到该学号，请确认", vbCritical, "错误"
   End If
End Sub
```

从上述代码可以看出，记录的查询是根据用户输入的学号与数据表中的学号进行比较，如发现相同的记录则显示出该记录的其他信息，否则显示没有找到的提示。

（3）运行程序。运行该窗体后，在学号对应的文本框中输入需要查询的学号 990001 的记录，其返回结果如图 15-13 所示。如果输入数据库中没有学号 990005 的记录，则返回如图 15-14 所示的结果。

图 15-13　数据查询成功

图 15-14　数据查询失败提示

从上述实例可以看出，使用 Recordset 对查询的实现是较为简单的，其变化都在于 SQL 查询语句。此外，如果将 Where 子句后的条件通配符替换掉，则可实现模糊查询。例如，要找出 student 表中所有姓“杨”的学生记录，只需在 SQL 语句中进行如下变化即可。

```
sql = "select * FROM student where sname like  杨%
```

15.1.5 更新记录数据

更新记录数据即对记录集中的数据重新赋值。操作方法为：使用 SQL 语句将要修改字段的一个数据查找出并重新赋值。例如，下面实例实现将学号为 990001 的学生系部更改为“外语系”。

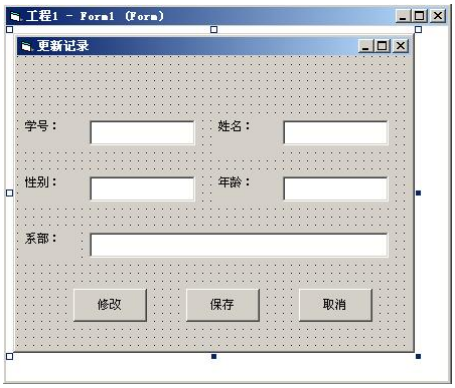

图 15-15　窗体设计

在该实例中，用户在输入对话框中输入数据，根据用户的输入显示相应信息。首先单击“修改”按钮，弹出输入对话框，用户输入学号后单击“确定”按钮，显示出对应学生的相关信息，用户可修改并确认。其实现步骤如下。

（1）窗体设计。与上述添加新记录的实例类似，该实例新建一个“标准 EXE”工程后，设计一个如图 15-15 所示的窗体。

注意：由于学号字段作为主键，是不可更改的，否则将造成数据表混乱，因此此处设置学号字段对应的文本框为不可修改。该实例中需要设置的属性如表 15-4 所示。

表 15-4　控件属性设置

控件	属性	设置值
Form1	Caption	Form1.Caption="更新记录"
Command1	Caption	Command1.Caption="修改"
Command2	Caption	Command2.Caption="保存"
Command3	Caption	Command3.Caption="取消"
Text1	Enable	Text1.Enable=False

（2）添加 ADO 引用。由于该实例中要使用到 ADO 作为数据库的编程接口，因此需要将 ADO 对象引用添加到该工程。选择“工程”→“引用”命令，打开“引用”对话框，如图 15-3 所示，选择其中的“Microsoft ActiveX Data Objects 2.0 Library”选项后，单击“确定”按钮即可。

（3）编写事件代码。根据实例的要求，单击“修改”按钮后要弹出一个输入对话框，单击“保存”按钮后要将修改后的结果写入数据库中，实现代码如下。

```
Dim sno As String                                          '定义通用变量
Dim CN As New ADODB.Connection
Dim rs As New ADODB.Recordset
Dim sql As String
Private Sub Command1_Click()                               '“修改”按钮代码
sno = InputBox("请输入要修改学生学号", "查找学生")
CN.Open "Provider=Microsoft.Jet.OLEDB.4.0;Data Source= " & App.Path & "\student.mdb;Persist
Security Info=False"
   sql = "select * FROM t_user where sno='" & Trim(sno) & "'"
   rs.Open sql, CN, 1, 1
```

```
   If rs.EOF <> True Then
   Text1.Text = rs("sno")
   Text2.Text = rs("sname")
   Text3.Text = rs("ssex")
   Text4.Text = rs("sage")
   Text5.Text = rs("sdept")
   Else
   MsgBox "没有找到该学生，请确认输入", vbCritical, "错误"
 End If
 CN.Close
End Sub
Private Sub Command2_Click()                                    '“保存”按钮代码
CN.Open "Provider=Microsoft.Jet.OLEDB.4.0;Data Source= " & App.Path & "\student.mdb;Persist
Security Info=False"
   sql = "select * FROM t_user where sno='" & Trim(sno) & "'"
   rs.Open sql, CN, 3, 2
   If rs.EOF <> True Then
     rs!sname = Trim(Text2.Text)
     rs!ssex = Trim(Text3.Text)
     If IsNumeric(Trim(Text4.Text)) = False Then
       MsgBox "年龄字段必须为数字，请重新输入"
     Else
     rs!sage = Trim(Text4.Text)
     rs!sdept = Trim(Text5.Text)
     rs.Update
     MsgBox "更新一条记录", , "提示"
     End If
   Else
     MsgBox "该数据已不存在，操作异常", vbCritical, "错误"
   End If
End Sub
Private Sub Command3_Click()                                    '“取消”按钮代码
End
End Sub
```

（4）运行程序。运行该窗体后，单击“修改”按钮，弹出如图 15-16 所示的输入对话框，在其中输入需要修改学生记录的学号 990001，显示结果如图 15-17 所示。

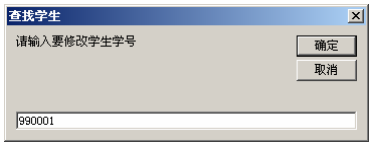

图 15-16　接收查找关键字

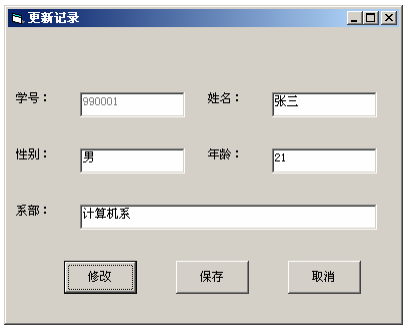

图 15-17　查找显示数据

从图 15-17 中可以看出，学号字段不能修改，其他字段都可以修改。在该窗体中，将“系部”文本框中“计算机系”改为“外语系”后，单击“保存”按钮，执行结果如图 15-18 所示。

图 15-18　更新数据

此外，用户也可以使用 SQL 语句中的 Update 语句来实现更新，读者可自行尝试。同样，修改后的程序也要其健壮性。

15.1.6　删除记录

删除操作就是将数据库中的数据删除，该操作较简单。在 ADO 中删除记录集中的数据的方法是 Delete 方法。这与 DAO 对象的方法相同，但它在 ADO 中的能力增强了，可以删掉一组记录。其语法格式如下。

```
Recordset.Delete AffectRecords
```

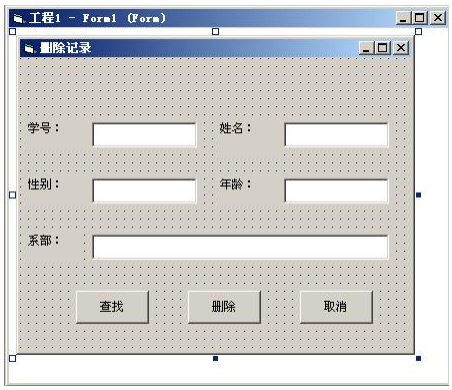

图 15-19　窗体设计

其中，AffectRecords 参数用于确定 Delete 方法所作用的方式，通常，其取值有 AdAffectCurrent 和 AdAffectGroup 两种，其功能如下。

- AdAffectCurrent：只删除当前的记录。
- AdAffectGroup：删除符合 Filter 属性设置的那些记录。为了一次能删除一组数据，还应设置 Filter 属性。

例如，下列实例删除学号为 990001 的学生记录，首先也必须先找到该记录，在窗体中显示后才能删除。其实现步骤如下。

（1）窗体设计。与上述更新记录的实例类似，该实例新建一个“标准 EXE”工程后，设计一个如图 15-19 所示的窗体。

该实例中需要设置的属性如表 15-5 所示。

表 15-5　控件属性设置

控件	属性	设置值
Form1	Caption	Form1.Caption="删除记录"
Command1	Caption	Command1.Caption="查找"

续表

控件	属性	设置值
Command2	Caption	Command2.Caption="删除"
Command3	Caption	Command3.Caption="取消"

（2）添加 ADO 引用。由于该实例中要使用到 ADO 作为数据库的编程接口，因此需要将 ADO 对象引用进入该工程。选择“工程”→“引用”命令，打开“引用”对话框，如图 15-3 所示，选择其中的“Microsoft ActiveX Data Objects 2.0 Library”选项后，单击“确定”按钮即可。

（3）编写事件代码。根据实例的要求，单击“查找”按钮后要弹出一个输入对话框，单击“删除”按钮后要将找到的记录从数据库中删除，实现代码如下。

```
Dim sno As String                                          '定义通用变量
Dim CN As New ADODB.Connection
Dim rs As New ADODB.Recordset
Dim sql As String

Private Sub Command1_Click()                               '“查找”按钮代码
sno = InputBox("请输入要修改学生学号", "查找学生")
CN.Open "Provider=Microsoft.Jet.OLEDB.4.0;Data Source= " & App.Path & "\student.mdb;Persist
Security Info=False"
   sql = "select * FROM t_user where sno='" & Trim(sno) & "'"
   rs.Open sql, CN, 1, 1
   If rs.EOF <> True Then
   Text1.Text = rs("sno")
   Text2.Text = rs("sname")
   Text3.Text = rs("ssex")
   Text4.Text = rs("sage")
   Text5.Text = rs("sdept")
   Else
   MsgBox "没有找到该学生，请确认输入", vbCritical, "错误"
  End If
  CN.Close
End Sub
Private Sub Command2_Click()                               '“删除”按钮代码
CN.Open "Provider=Microsoft.Jet.OLEDB.4.0;Data Source= " & App.Path & "\student.mdb;Persist
Security Info=False"
   sql = "select * FROM t_user where sno='" & Trim(sno) & "'"
   rs.Open sql, CN, 3, 2
   response = MsgBox("确定要删除此学生记录吗？", vbOKCancel + 32, "提示")
   If response = vbOK Then
   rs.Delete
   MsgBox "删除成功", 16 + vbApplicationModal, "提示"
   End If
End Sub
Private Sub Command3_Click()                               '“取消”按钮代码
End
End Sub
```

（4）运行程序。运行该窗体后，单击“查找”按钮，弹出如图 15-20 所示输入对话框，在其中输入需要删除学生记录的学号 990001，显示结果如图 15-21 所示。需要注意的是，删除一条记录是不可恢复的操作，因此删除前一定要经过用户的确认。

如果用户单击“确定”按钮确认删除后，该记录将从数据库中被删除，并给出返回信息，如图 15-22 所示，否则将不删除。

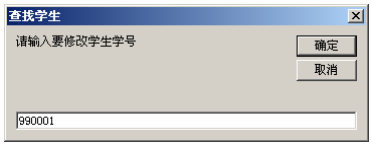

图 15-20　输入对话框

图 15-21　确认提示

图 15-22　确认删除记录

上述代码中，值得注意的就是 MsgBox 函数的使用了。此处用到了其返回值，如果返回值为 vbOK 时则确认删除，否则将不操作。

15.1.7　关闭记录集

在应用程序结束之前，应该释放分配给 ADO 对象的内存资源，Windows 系统回收这些内存资源并分配给其他的应用程序。在 Visual Basic 6.0 中，关闭记录集和数据库的方法为 Close 方法，该方法也能用在数据库对象上，将数据库关闭。

- 关闭记录集：Rs.Close。
- 关闭数据库：Ds.Close。

例如，在前面删除记录的实例中，如果对记录的操作完成后，就可以使用如下语句将打开的记录集关闭，以便释放内存资源。

```
rs.close
```

15.2　常用数据控件

在 Visual Basic 中，提供了相当多的数据控件，用于连接数据源、显示数据，如上一章提到的 ADO 控件。本章将介绍更多的数据控件。

15.2.1　Data 控件

Data 控件是常用的一个数据控件。前面章节提到，该控件对应的数据对象模型是 DAO，主要用于连接数据源并与其他数据绑定控件一起使用。该控件可连接的数据库类型有 Access、DBF、XLS、ODBC 等。该控件是 Visual Basic 的标准控件，其在控件栏上的位置如图 15-23 所示。

Data 控件是数据控件，因此其重要的属性就是与数据连接或数据显示有关的属性，需要特别注意的是 RecordSource 属性。Data 控件的 RecordSource 属性可以是数据表名，也可以是数据表中的某些行或多个数据表中的数据组合。可以直接在 Data 控件的 RecordSource 属性栏中输入 SQL，或在代码中通过 SQL 语句将选择的记录集赋给数据控件的 RecordSource 属性。

注意：在窗体设计中设置 RecordSource 属性前，需要先设置该控件的 DatabaseName 属性，即数据库名称。否则，将出现如图 15-24 所示错误提示。

图 15-23　Data 控件

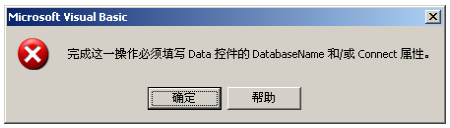

图 15-24　错误提示

通常，为了能够在程序中动态控制显示的数据，RecordSource 属性在程序中使用代码赋值用得较多。例如，下列语句获取表记录，并将其放入数据控件中。

```
Data1.RecordSource = "Select * From student "
Data1.Refresh
```

上述代码将数据表 Student 中的所有记录都放在了 Data 控件中。Data 控件的重要属性如表 15-6 所示，常用事件如表 15-7 所示，常用方法如表 15-8 所示。

表 15-6　Data 控件的重要属性

属性名称	作用
Connect	指定数据控件所要连接得数据库类型，VB 默认的是 Access 的 MDB 数据库，也可以连接 DBF、XLS、ODBC 等数据库
DatabaseName	指定具体使用的数据库文件名，包括路径名
RecordSource	指定具体可访问的数据，这些数据构成记录集对象 Recordset 对象，可以是数据库中的单个表名、一个存储查询，也可以是 SQL 查询命令
RecordsetType	确定记录集类型，有如下 3 种： 0——Table（表） 1——Dynaset（动态，默认的） 2——Snapshot（快照）
BofAction	当记录指针指向记录集的开始时，确定数据控件该采取的操作： 0——控件重定位到第一个记录 1——移过记录集开始位，定位到一个无效记录，触发数据控件对第一个记录的无效事件 Validate
EofActrion	当记录指针指向记录集的结尾时，确定数据控件该采取的操作： 0——控件重定位到最后一个记录 1——移过记录集结束位，定位到一个无效记录，触发数据控件对最后一个记录的无效事件 Validate

表 15-7　Data 控件的事件

事件名称	触发时间
Reposition	发生在一条记录成为当前记录后，只要将记录指针从一条记录移到另一条记录触发
Validate	在一条不同的记录成为当前记录前，Update 方法前或 Delete 等操作前发生该事件，检查被数据控件绑定的控件内的数据是否发生变化

表 15-8　Data 控件的常用方法

方法名称	作用	实例
Refresh	激活数据控件，使各用户对数据库的操作有效	Data1.Refresh
UpdateControls	将数据从数据库中重新读到数据控件绑定的控件内，通过起可终止用户对绑定控件内的数据修改	放弃修改按钮代码：Data1.UpdateControls
UpdateRecord	强制数据控件将绑定控件内的数据写到数据库中，不再触发 Validate 事件	确认修改按钮代码：Data1.UpdateRecord

例如，下列实例通过 Data 控件连接数据库，并通过 4 个 TextBox 控件，将数据表 t_user 中的记录全部显示在窗体上，其实现步骤如下。

（1）设计窗体。新建一个“标准 EXE”工程后，设计一个如图 15-25 所示的窗体。

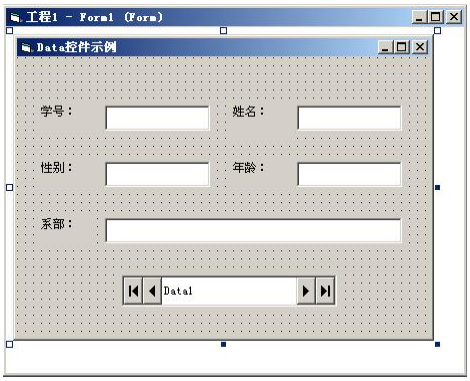

图 15-25　窗体设计

（2）设置相应控件属性，如表 15-9 所示。

表 15-9　设置控件属性

控件	属性	设置值
Data1	Connect	Data1.Connect="Access"
	DatabaseName	Data1.DatabaseName="D:\student.mdb"
	RecordSource	Data1.RecordSource="student"
Text1、Text2、Text3、Text4、Text5	DataSource	Text1.DataSource=Data1
Text1	DataField	Text1.DataFiled=Sno
Text2	DataField	Text2.DataFiled=Sname

续表

控件	属性	设置值
Text3	DataField	Text3.DataFiled=Sgentle
Text4	DataField	Text4.DataFiled=Sage
Text5	DataField	Text5.DataFiled=Sdept
Form1	Caption	Data 控件实例

需要注意的是，设置各文本框的 DataSource 属性即将其与数据库连接起来，设置其 DataFiled 属性即与表中各字段绑定起来。而 Data 控件的 Connect 属性用于设置数据库的类型为 Access 数据库，DatabaseName 属性用于设置数据库的路径，RecordSource 属性用于设置数据库中的具体表 t_user。使用 Data 控件显示数据时，绑定控件比较重要，如图 15-26 所示即将 TextBox 文本框控件绑定为显示 t_user 表中来显示学号。

（3）运行程序。通过以上实例可以看出，在该实例中，没有输入一行代码，即实现了数据在窗体上的显示，这就体现了 Visual Basic 在数据库编程上的优势。程序运行结果如图 15-27 所示。

图 15-26　绑定控件

图 15-27　Data 控件应用实例

当数据表 student 中有一条记录时，单击 Data 控件两边的按钮，可实现“第一条”、“上一条”和“下一条”、“最后一条”的功能，这样就可以全部预览到表中的数据。

15.2.2　DataGrid 控件

DataGrid 控件是 Visual Basic 6.0 中用于数据显示的常用控件之一。使用该控件显示数据，且修改相关属性便可对其中数据进行更新操作，简单直观，功能强大。

DataGrid 控件是一种类似于电子数据表的绑定控件，可以显示一系列行与列来表示 Recordset 对象的记录和字段。可以使用 DataGrid 控件来创建一个允许最终用户阅读和写入到绝大多数数据库的应用程序。DataGrid 控件可以在设计时快速配置，只需少量代码或无需代码即可。

设计过程中，设置了 DataGrid 控件的 DataSource 属性后，就会使用数据源的记录集来自

动填充该控件，以及自动设置该控件的列标头。用户还可以编辑该网格的列以及删除、重新安排、添加列标头或者调整任意一列的宽度。

在 Visual Basic6.0 中，DataGrid 控件并不在标准控件栏中，首先选择“工程”→“部件”命令，在“部件”对话框的“控件”选项卡中选中“Microsoft DataGrid Control 6.0”选项即可，如图 15-28 所示。添加完成后，DataGrid 控件在控件工具栏中的位置如图 15-29 所示。

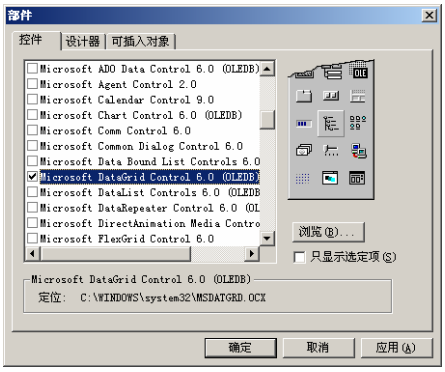

图 15-28　添加 DataGrid 控件

图 15-29　DataGrid 控件

注意：DataGrid 控件本身并不能连接数据库，其显示数据的功能需要与数据对象模型结合起来使用。

通常，该控件与 ADO 的绑定是比较常见的，而和 ADO 数据控件的绑定控件属性主要有 DataSource、DataField、DataMember、DataFormat 等。各属性功能如下。

- DataSource 和 DataField 属性：用于连接数据表中的表和字段。
- DataMember 属性：允许处理多个数据集，可以从数据供应程序提供的几个数据成员中返回或设置一个特定的数据成员。
- DataFormat 属性：用于指定数据内容的显示格式。

数据绑定控件的主要属性如表 15-10 所示。

表 15-10　数据绑定控件属性

属性	描述
DataSource	DataList、DataCombo 所绑定数据控件的名称
DataList	由 DataSource 属性所指定的记录集中的一个字段名称。该字段将用于决定在列表中高亮显示的元素，如果作出了新的选择，则其就是当移动到一个新记录时所需更新的字段
RowSource	用于填充列表的数据控件的名称
BoundColumn	由 RowSource 属性所指定的记录集中的一个字段名称。该字段必须和将用于更新该列表的 DataField 的类型相同
ListField	由用于填充该列表的 RowSource 所指定记录集中的一个字段名称

例如，下列实例在窗体上添加了一个 DataGrid 控件和一个 ADO 控件。将 DataGrid 控件放置在窗体上后，就应为其设置属性了。DataGrid 控件最重要的一个属性就是 DataSource 属性，用于设置 DataGrid 控件的数据源，如图 15-30 和图 15-31 所示。

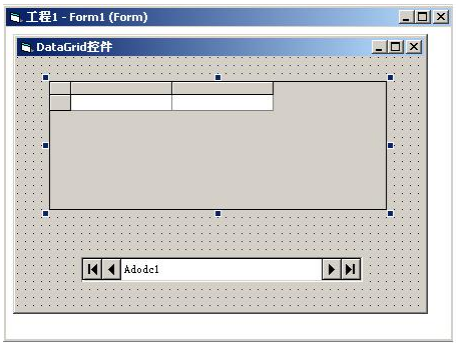

图 15-30　窗体设计

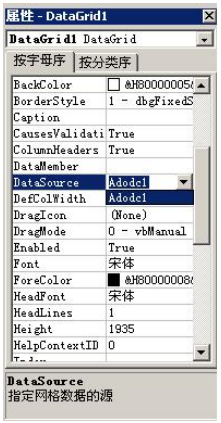

图 15-31　DataGrid 控件属性

通常，DataGrid 控件只需设置 DataSource 属性即可。至于数据库的连接操作，可以交由 Adodc 控件来完成。前面章节已介绍过使用 Adodc 控件设置数据库连接的操作，即设置其 ConnectionString 属性和 RecordSource 属性即可。例如，此处设置 RecordSource 属性为数据表 t_user。

Adodc 控件的数据连接设置完成后，右击 DataGrid 控件，在弹出的右键菜单中选择“检索字段”命令，在弹出的“检查字段”对话框中单击“是”按钮，则可刷新 DataGrid 控件显示的字段名，并以要显示数据表中的字段代替，如图 15-32 所示。

设置完成后，运行上述表单，执行结果如图 15-33 所示。

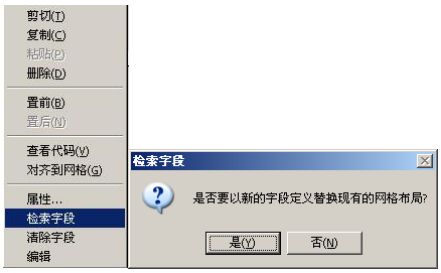

图 15-32　检索字段

图 15-33　显示表数据

此处显示的数据是 Adodc 控件中设置的数据表 t_user 的全部内容。此外，还可以设置成具体的 SQL 查询，只需修改 Adodc 控件的 RecordSource 属性即可。例如，要显示 Student 表内所有计算机系学生的记录，设置 Adodc 控件的 RecordSource 属性即可。打开其“属性页”对话框，在“命令类型”下拉列表框中选择“1－adCmdText”选项，在“命令文本”文本框中输入如下 SQL 查询语句即可，如图 15-34 所示。

```
select * from student where sdept="计算机系"
```

其执行结果如图 15-35 所示。

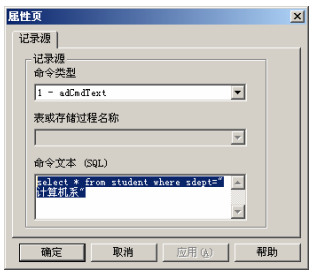

图 15-34　设置记录源

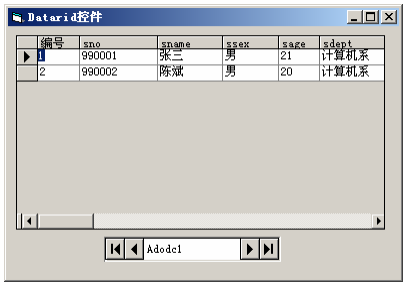

图 15-35　显示查询数据

由此可知，得益于 ADO 的优势，其显示数据是非常灵活的。在运行过程中，可以在程序中通过切换 DataSource 来察看不同的表，或者可以修改当前数据库的查询，以返回一个不同的记录集合。这将在后续章节中通过实例进行详细介绍。

图 15-36　修改数据功能设置

在上述实例中，可以发现，在 DataGrid 控件上显示的数据是可以被修改的，而且修改后的数据会直接写入数据库，这对于一些只允许用户浏览的数据表是不利的。因此，DataGrid 控件提供了几个属性，用于控制用户对其中数据的操作，如图 15-36 所示。

对 DataGrid 控件中的数据进行修改的设置属性一共有如下 3 个。

- ❑ AllowAddNew 属性：允许用户增加新的记录，默认值为 False。
- ❑ AllowDelete 属性：允许用户删除记录，默认值为 False。
- ❑ AllowUpdate 属性：允许用户修改记录，默认值为 True。

如不允许用户修改记录，只需将其 AllowUpdate 属性值设为 False 即可。此外，上述实例中显示的字段名为表的字段定义名，如“sno”、“sname”等。实际上，许多应用程序要求以别名来显示数据，如“学号”、“姓名”等，在 DataGrid 控件中也提供了相应的属性设置。在窗体设计中，右击 DataGrid 控件，在弹出的右键菜单中选择“属性”命令，如图 15-32 所示，在弹出的“属性页”对话框中选择“列”选项卡，即可设置其别名，如图 15-37 所示。

经过上述几个方面的设置，一个专业的数据显示表就可以呈现在用户眼前了。例如，如图 15-38 所示是一个绑定了 ADO、不允许用户修改数据、显示对应查询数据并以别名显示字段的数据表。

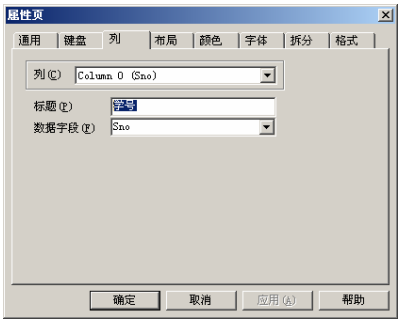

图 15-37　设置列别名

图 15-38　数据显示

15.2.3　DataList 控件

DataList 控件与标准列表框控件极为相似，但有一些重要的不同之处，这种不同使其在数据库应用程序中具有极大的适应性和实用性。例如，DataList 控件能被其绑定的数据库字段自动填充。使用 DataList 控件也需先将其添加到控件栏中。选择“工程”→“部件”命令，在“部件”对话框的“控件”选项卡中选择“Microsoft DataList Controls 6.0”选项即可，如图 15-39 所示。

此时可以注意到在控件栏上出现了两个控件：DataList 控件和 DataCombo 控件。这两个控件使用方法类似，只是 DataCombo 控件多了一个可供用户编辑的文本框。至于 DataCombo 控件，将在下一节中介绍其使用方法。

DataList 控件主要用于显示数据库中的数据，通常与 ADO 绑定使用。其与 DataGrid 控件相似，也是通过 Adodc 控件连接数据库，再使用 DataList 控件的几个数据属性将其绑定起来。例如，下列实例实现在 DataList 控件中显示所有计算机系学生的名单，供用户选择。其实现步骤如下。

（1）设计窗体。新建一个“标准 EXE”工程后，设计一个如图 15-40 所示的窗体。

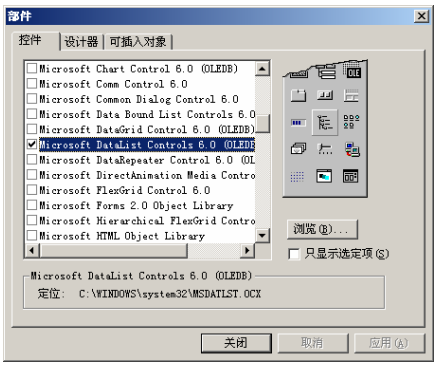

图 15-39　添加 DataList 控件

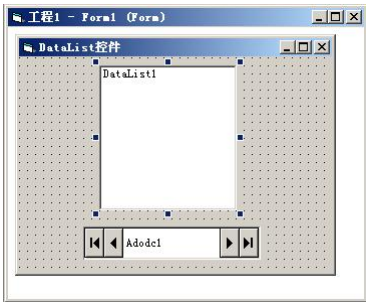

图 15-40　设计窗体

该窗体中只有 ADO 及 DataList 控件。其中，Adodc 控件用于连接数据库，DataList 控件用于显示用户要求的输出数据。

（2）设置属性绑定数据。这里的绑定数据指的就是设置相关控件的属性来实现数据绑定。其中，Adodc 控件的设置不变，如表 15-11 所示。

表 15-11　Adodc 属性设置

控件	属性	设置值
Adodc1	Visibled	False
Adodc1	ConnectionString	Provider=Microsoft.Jet.OLEDB.4.0;Data Source=D:\student.mdb;Persist Security Info=False
Adodc1	RecordSource	Select * from t_user where sdept="计算机系"

DataList 控件的主要属性如下。

- DataSource：用于连接数据源，一般是 Adodc 控件的名称。
- DataField：用于指定显示字段，如“sname”姓名字段。
- RowSource：用于设定列表数据源，一般是 Adodc 控件的名称。
- ListField：用于指定显示字段，如“sname”姓名字段。

在该实例中，可参照上述设置来绑定数据。该实例要求显示出所有计算机系学生的姓名，而 Adodc 控件中取得了计算机系学生的所有记录，因此可在 DataSource 属性和 RowSource 属性中指定 Adodc1 控件名。在 DataField 属性和 ListField 属性中指定其字段 sname 即可，如图 15-41 所示。

（3）运行程序。绑定数据完成后，运行该实例，运行结果如图 15-42 所示。

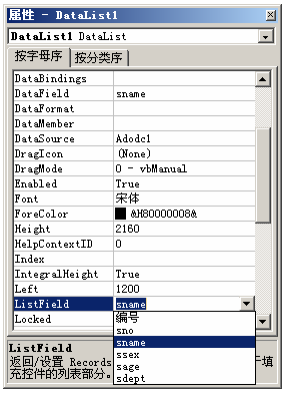

图 15-41　绑定数据

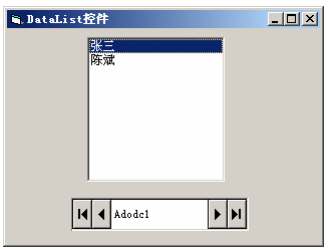

图 15-42　DataList 控件应用实例

15.2.4　DataCombo 控件

DataCombo 控件与 DataList 控件类似，这里也通过一个实例来介绍其使用。例如，使用 DataCombo 控件显示所有计算机系学生名单。窗体设计和数据绑定与 DataList 控件基本相同，需要注意绑定属性也是 DataSource、DataField、RowField 和 ListField 等。运行结果如图 15-43 所示。

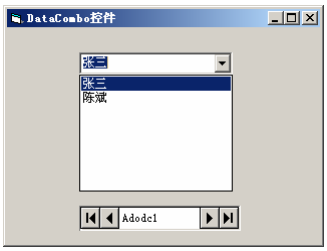

图 15-43　DataCombo 控件应用实例

此外，还需要注意该控件的 Style 属性，DataCombo 控件有 3 种显示形式，分别对应 Style 属性的 3 个值，分别如下。

- 0 – dbcDropdownCombo：对应下拉组合框，其显示形式如图 15-43 所示。在该形式下，文本框是可编辑的。
- 1 – dbcSimpleCombo：对应简单组合框，其显示形式如图 15-44 所示。
- 2 – dbcDropdownList：对应下拉列表框，其显示形式如图 15-45 所示。在该形式下，文本框是不可编辑的。

图 15-44　DataCombo 的简单组合框

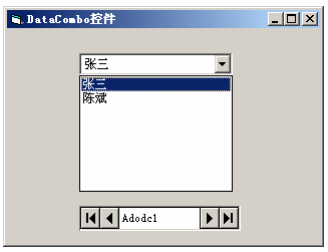

图 15-45　DataCombo 的下拉列表框

除了上述几个数据控件外，还有 DataReport 控件，即报表设计器。这是具体程序设计中经常用到的，将在下一章进行具体介绍。

15.3　数据库管理器

数据库管理器是 Visual Basic 6.0 提供的便于用户进行数据库应用系统开发的工具。该工具可为没有安装数据库的客户端提供数据库支持。

15.3.1　建立数据库

在 Visual Basic 6.0 中，系统提供了一个用于支持数据库的工具，当本机没有安装相应数据库时，使用该工具可建立数据库，并提供访问。下面以建立 Access 数据库为例，详细介绍使用该工具建立数据库的步骤。

（1）选择“外接程序”→“可视化数据库管理器”命令，打开数据库管理器，如图 15-46 所示。

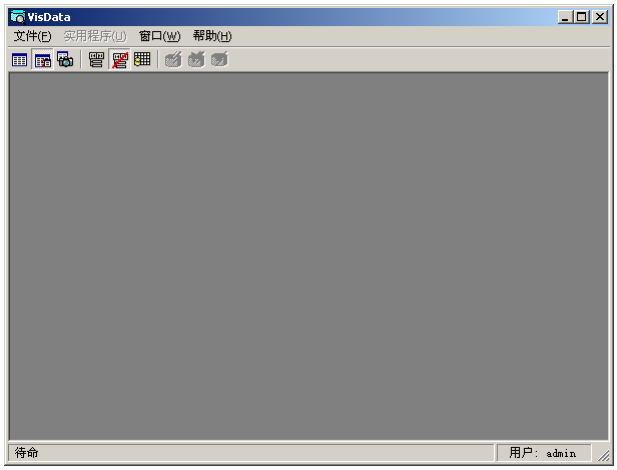

图 15-46　数据库管理器

（2）创建数据库。选择“文件”→“新建”菜单项，出现数据库类型选择菜单，如图

15-47 所示。该数据库管理器可创建多个类型的数据库。此处选择数据库类型菜单中的“Microsoft Access”，在版本菜单中选择要创建的数据库版本。

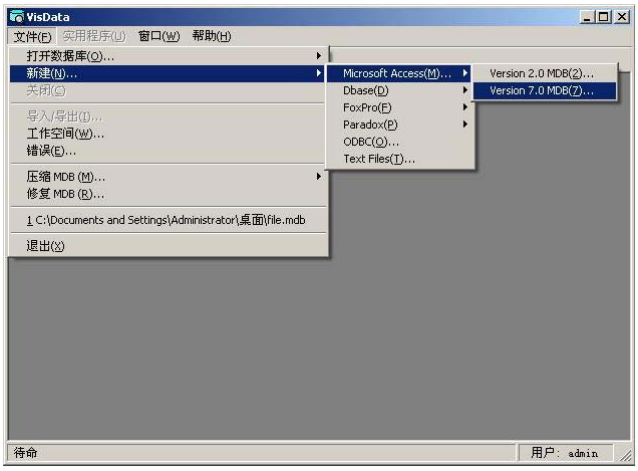

图 15-47　选择数据库类型

选择要创建的数据库类型及版本后，弹出新建数据库对话框。在此对话框中，输入要创建的数据库名 student.mdb。此时，在可视化数据库管理器窗口中出现“数据库窗口”窗口和“SQL 语句”窗口，如图 15-48 所示。

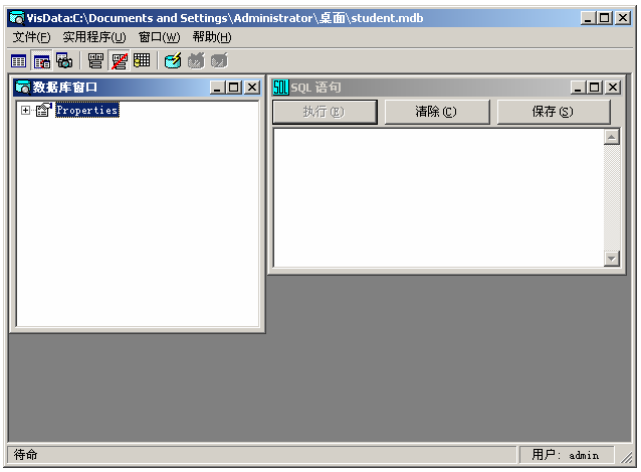

图 15-48　创建数据库

（3）创建数据表。“数据库窗口”以树型结构显示数据库中的所有对象，在窗口空白处右击，在弹出的右键菜单中选择“新建表”或“刷新列表“命令即可新建一个表，如图 15-49 所示。

选择“新建表”命令后，即出现如图 15-50 所示的“表结构”对话框。在该对话框中，可输入新表的名称，并添加字段，也可从表中删除字段，还可以添加或删除作为索引的字段。

（4）生成表。在“表结构”对话框中单击“添加字段”按钮可对表输入字段，输入字段的同时可对其字段类型、大小、验证规则等进行设置，如图 15-51 所示。

图 15-49　新建表

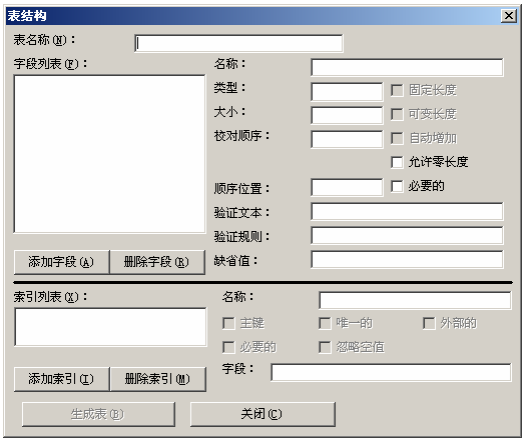

图 15-50　“表结构”对话框

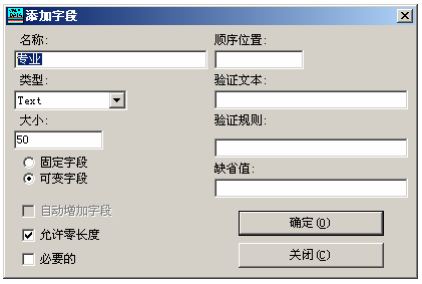

图 15-51　添加字段

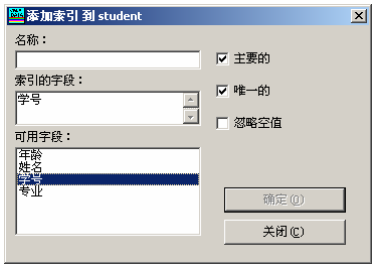

图 15-52　添加索引

录入字段后，在 “表结构”对话框中为其添加索引，如图 15-52 所示。完成上述步骤后，单击“生成表”按钮，即建立了该表。

建立一张表后，可以再建立另一张表。如果要在已经存在的数据库文件内增加一张新表，只需要在数据库管理器选择“文件”→“打开数据库”菜单项，其余操作过程与建立数据表的操作相同。创建好的表在数据库管理器中的显示如图 15-53 所示。

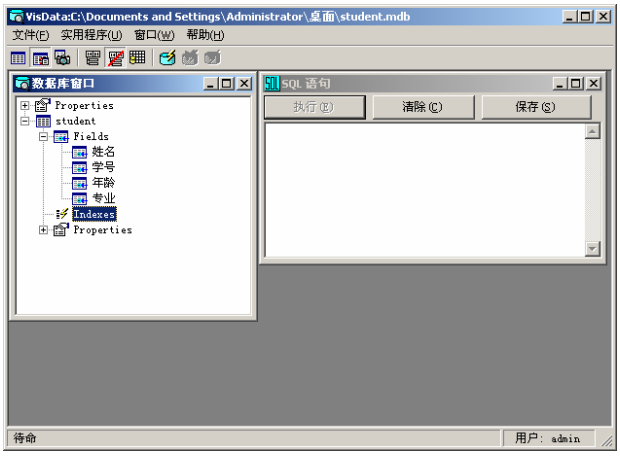

图 15-53　建立数据表

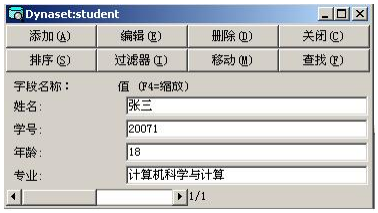

图 15-54　添加记录

（5）添加记录。表建立好后，双击出现在数据库窗口中的“student”表名，打开表格输入对话框，选择对应命令进行添加、编辑、增删记录等操作。例如，单击“添加”按钮后，打开添加对话框输入记录结果，如图 15-54 所示。

至此，一个包含数据记录的 Access 数据库就已经创建完成了，用户可像访问由 Microsoft Access 程序创建的数据库一样访问该数据库。

15.3.2　外接程序管理器

除了数据库管理器外，Visual Basic 6.0 还提供了外接程序管理器，用于在 Visual Basic 6.0 中使用其他的应用程序，如图 15-55 所示。

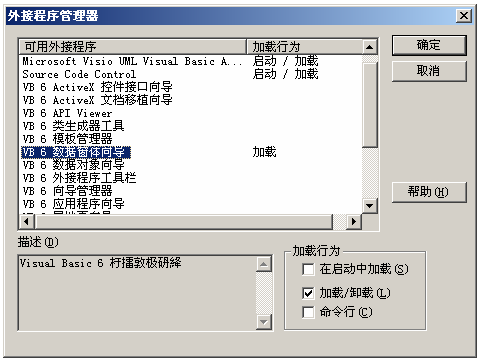

图 15-55　外接程序管理器

该管理器提供了许多加载向导，读者可自行测试其使用效果。

15.4　应用实例

至此，使用 Visual Basic 6.0 进行数据库编程就大体介绍完了。下面通过一个具体的实例来回顾一下数据库编程的完整过程。

例如，下面实例设计一个用户登录程序，用户登录信息存储在数据库 File.mdb 中的 user 表中，要求使用 ADO 连接数据库，其实现步骤如下。

（1）设计登录界面，如图 15-56 所示。

图 15-56　用户登录界面设计

（2）连接数据库。由于用户登录信息存储在数据表中，因此当用户输入登录信息并单击“确定”按钮后，需要将用户输入的信息与数据库中信息对比，如相同则通过认证，否则给出错误提示。此处通过 ADO 来与数据库建立连接，代码如下。

```
Dim cn As ADODB.Connection
Set cn = New ADODB.Connection
cn.Open "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=" & App.Path & "\file.mdb;Persist Security
Info=False"
```

（3）打开记录集对象。连接数据库后，即可打开记录集以供使用了，实现代码如下。

```
Dim rs As ADODB.Recordset
Dim sql As Variant
Set rs = New ADODB.Recordset
sql = "select * from user where username='" & Text1.Text & "'"
 rs.Open sql, cn, 1, 1                                    'cn 为上一步骤中定义的 ADO 的 Connection 对象
```

（4）使用记录集。此处只需从数据表中将数据取出并与用户的输入进行比较即可，无需对数据库进行写入操作，因此其主要使用的是查询方法。其实现代码如下。

```
If rs.EOF = True Then                                     '没有找到用户输入的该用户名
   MsgBox "该用户名不存在", vbOKCancel + 48, "提示"
 Else
   If rs("userpassword") = Trim(Text2.Text) Then          '找到该用户名，且密码对应正确
     main.Show                                            '显示主程序界面
     Unload Me                                            '登录框关闭
   Else
     MsgBox "密码输入错误", vbOKCancel + 48, "错误"       '密码输入错误
     Text2.Text = ""                                      '清空密码框，准备接收下次输入
     Text2.SetFocus                                       '密码框获得焦点
   End If
 End If
```

（5）断开连接。当使用数据库完成后，使用如下语句断开连接。

```
cn.Close
rs.Close
```

至此，该实例完成。读者可将其代码组合起来，测试运行来查看其效果。事实上，这也是大多数应用程序使用的一个验证登录的实例程序，读者可仔细理解。

15.5　小结

本章重点介绍了 RecordSet 记录集对象，包括其对象定义、数据操作等，着重讲解了该对象对数据的查询、添加、更新和删除操作，对于每一种操作，都有详细的实例代码。此外，本章主要对数据控件进行了介绍，包括 Data 控件、DataGrid 控件、DataList 控件和 DataCombo 控件。这些控件都需要与 ADO 绑定数据，该章以实例的形式讲解了其在具体编程环境中的使用。总的来说，该章是使用 Visual Basic 进行数据库编程的入门章节，读者需仔细理解。