第 3 章　数据库编程

本章学习目标

通过本章实例的学习，读者应该熟练掌握以下内容:

- 在 VC++中对数据库的访问
- 使用 DAO 读取 Access 数据库中的数据
- 文本文件的读写操作

3.1　程序的功能及主要知识

3.1.1　程序的功能

本章创建一个基于对话框的应用程序，程序从 MS Access 数据库中读出数据，并按要求写入一个文本文件中，程序运行界面如图 3.1 所示。

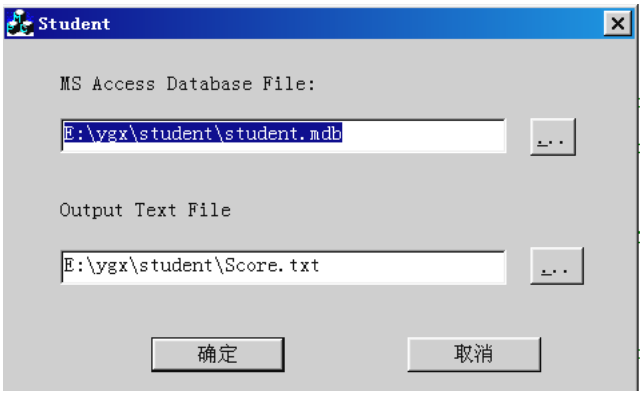

图 3.1　程序运行界面

用户可以在对话框中输入 MS Access 数据库文件名以及输出的文本文件名，也可以通过后面的按钮选择数据库文件及输出的文本文件，分别弹出图 3.2 和图 3.3 所示的通用文件对话框，选择后，单击“打开”按钮回到图 3.1 所示的界面。然后单击“确定”按钮，从数据库读取数据完毕并按要求的格式写到文件 Score.txt 中。

数据库的结构将在下面的 3.2.1 节中介绍，要求输出①成绩单文件，包含学号、姓名、年龄、成绩 1、成绩 2 和成绩 3；②通讯地址表文件，包括学号、姓名、邮政编码和通讯地址。其中通讯地址表由学生自己完成。输出的成绩单文件如图 3.4 所示。

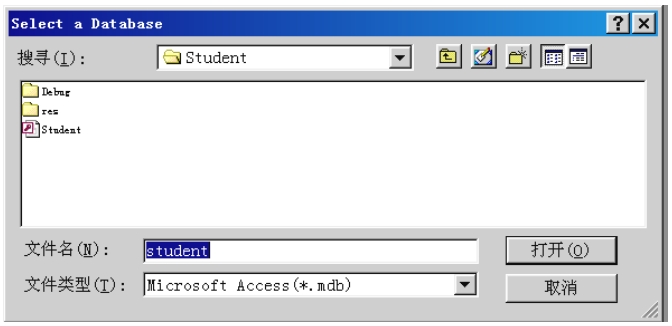

图 3.2　选择数据库文件

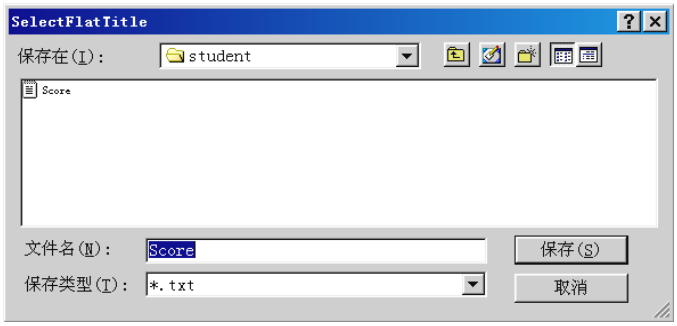

图 3.3　选择输出的文本文件

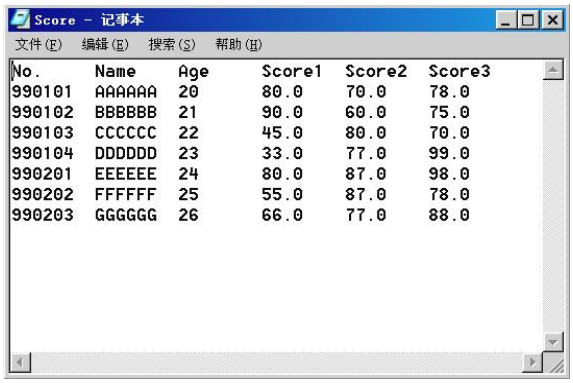

图 3.4　输出的成绩单文件

3.1.2　运用的主要知识

本程序主要运用了 DAO 相关技术，介绍了如何建立数据库连接，如何打开数据库中的表，如何使用 SQL 语句操纵数据库表中的数据等。程序中使用了 CDaoDatabase 类、CDaoRecordSet 类、文件类 CFile、通用文件对话框类 CFileDialog、字符串类 CString 以及文件查找类 CFileFind 等。

3.2　程序的实现过程及注释

3.2.1　创建 MS Access 数据库 Student.mdb

在 MS Access 97 中，创建学生数据库 Student.mdb（该数据库可以从中国水利水电出版社网站提供下载的源代码中找到），在其中添加 Student、Score 和 Address 三个表，分别存储学生基本情况、学生成绩和地址。三个表的关系通过学号字段连接，各表的结构如下：

1. Student 表

Student 表共有三个字段：SNo（学号），SName（姓名），SAge（年龄）。其中 SNo 为主关键字。字段类型如下：

SNo：文本，6

SName：文本 ，6

SAge：数字，整形

2. Score 表

Score 表共有四个字段：SNo（学号），Score1（成绩 1），Score2（成绩 2），Score3（成绩 3）。字段类型如下：

SNo：文本，6

Score1：数字，单精度

Score2：数字，单精度

Score3：数字，单精度

3. Address 表

Address 表共有三个字段：ANo（学号），Address（地址），APostNo（邮编）。字段类型如下：

ANo：文本，6

Address：文本，30

APostNo：文本，6

在三个表中输入少量数据，以备下面的程序使用。

3.2.2　创建基于对话框的程序 Student

用应用向导创建一个名为 Student 的基于对话框的应用程序。

3.2.3　从 CDaoRecordSet 类派生数据库三个表对应的类

为了使用 MFC DAO 数据库类，在 stdAfx.h 文件中输入文件包含的宏定义。

```
#include <afxdao.h>          // MFC DAO database classes
```

下面为数据库中的三个表添加的对应的类，通过这些类实现对表的操作。

选择菜单 Insert→New class…，弹出对话框如图 3.5 所示。

在 Class type 下拉列表框中选择 MFC Class，在 Base class 下拉列表框中选择 CDaoRecordset，在 Name 文本框中输入 CStudentSet。单击 OK 按钮，弹出的选择数据源对话框如图 3.6 所示。

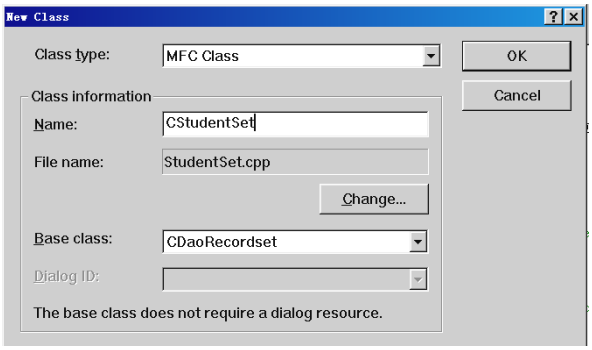

图 3.5　创建 CStudentSet 类

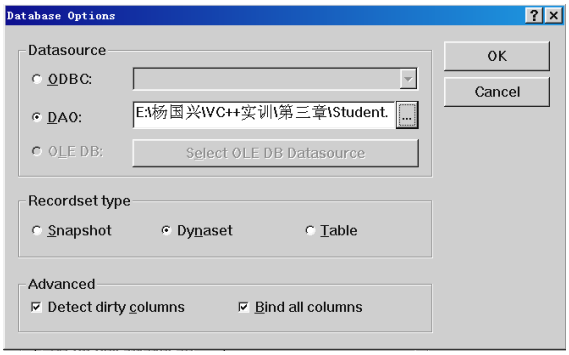

图 3.6　选择数据源

选择 DAO 单选按钮，并选择我们在上面建立的数据库（对于本例，也可以直接单击 OK 按钮，新类创建完毕）。单击 OK 按钮，弹出的选择数据库表对话框如图 3.7 所示。

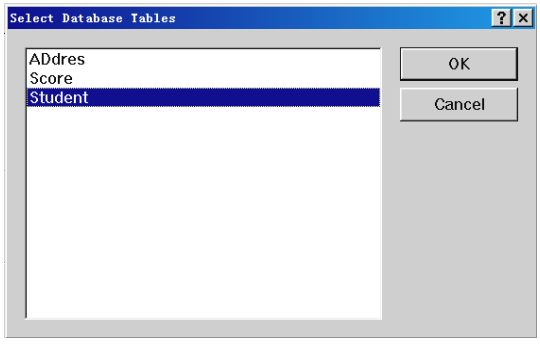

图 3.7　选择数据库表

选择 Student，单击 OK 按钮，类 CStudentSet 创建完毕。

重复上述步骤，分别创建 CScoreSet 类和 CAddressSet 类。

3.2.4　编辑对话框资源、添加变量及响应函数

1. 添加控件

编辑对话框 IDD_STUDENT_DIALOG（参照图 3.1），添加表 3.1 所示的控件。

表 3.1　控件

控件 ID	控件类型	标题
ID_STATIC	Static Text	MS Access Database File
ID_STATIC	Static Text	Output Text File
IDC_EDIT_SOURCE	Edit Box	
IDC_EDIT_TARGET	Edit Box	
IDC_BUTTON_SOURCE	Button	…
IDC_BUTTON_TARGET	Button	…
IDOK	Button	确定
IDCANCEL	Button	取消

2. 添加控件相关变量

利用类向导添加表 3.2 所示的控件相关变量。

表 3.2　控件相关变量

控件 ID	变量类型	变量名称
IDC_EDIT_SOURCE	CString	m_strSourceFile
IDC_EDIT_TARGET	CString	m_strTargetFile

3. 添加响应函数

利用类向导添加如下消息响应函数：

初始化函数：OnInitDialog()。

IDC_BUTTON_SOURCE 和 IDC_BUTTON_ TARGET 两个按钮的单击消息响应函数，OnButtonSource()和 OnButtonTarget()。

IDOK 按钮的单击消息响应函数 OnOK()。

在各函数中添加如下代码：

（1）函数 OnInitDialog()。

```
BOOL CStudentDlg::OnInitDialog()
{
    CDialog::OnInitDialog();

    // Add "About..." menu item to system menu.

    // IDM_ABOUTBOX must be in the system command range.
    ASSERT((IDM_ABOUTBOX & 0xFFF0) == IDM_ABOUTBOX);
    ASSERT(IDM_ABOUTBOX < 0xF000);

    CMenu* pSysMenu = GetSystemMenu(FALSE);
    if (pSysMenu != NULL)
    {
        CString strAboutMenu;
        strAboutMenu.LoadString(IDS_ABOUTBOX);
```

```
        if (!strAboutMenu.IsEmpty())
        {
            pSysMenu->AppendMenu(MF_SEPARATOR);
            pSysMenu->AppendMenu(MF_STRING,IDM_ABOUTBOX, strAboutMenu);
        }
    }

    // Set the icon for this dialog.  The framework does this automatically
    //  when the application's main window is not a dialog
    SetIcon(m_hIcon, TRUE);            // Set big icon
    SetIcon(m_hIcon, FALSE);           // Set small icon

    // TODO: Add extra initialization here

    m_strSourceFile = "E:\\ygx\\student\\student.mdb";
    m_strTargetFile = "E:\\ygx\\student\\Score.txt";
    UpdateData(false);

    return TRUE;  // return TRUE  unless you set the focus to a control
}
```

函数 OnInitDialog()为对话框的两个文本框（数据库文件名和输出的文本文件名）提供初始值，并显示在控件中。在做练习时文本框应该设置为自己计算机上数据库文件所在的文件夹以及要输出文本文件所在的文件夹。

（2）函数 OnButtonSource()。

```
void CStudentDlg::OnButtonSource()

{
    // TODO: Add your control notification handler code here
    CFileDialog dlg(true,NULL,NULL,OFN_HIDEREADONLY|OFN_OVERWRITEPROMPT);
    dlg.m_ofn.lpstrInitialDir = m_strSourceFile;
    dlg.m_ofn.lpstrTitle = "Select a Database";
    dlg.m_ofn.lpstrFilter = "Microsoft Access(*.mdb)\0*.mdb\0\0";
    int len = m_strSourceFile.GetLength() - m_strSourceFile.ReverseFind('\\');
    strcpy(dlg.m_ofn.lpstrFile,  m_strSourceFile.Right(len - 1));
    if(dlg.DoModal() == IDOK)
    {
        m_strSourceFile = dlg.GetPathName();
        UpdateData(false);
    }
}
```

单击 IDC_BUTTON_SOURCE 按钮，弹出打开文件对话框，上述代码为对话框提供了标题、打开的文件类型、默认的文件名。

通用文件对话框的成员变量 m_ofn 是 OPENFILENAME 类型的结构变量，用于确定文件对话框打开时的属性。其中成员 lpstrInitialDir 指定对话框中显示的初始路径，成员 lpstrTitle 指定对话框的标题，成员 lpstrFilter 指定对话框中显示文件的过滤条件，成员 lpstrFile 指定对

话框中显示文件的初始文件名。

（3）函数 OnButtonTarget()。

```
void CStudentDlg::OnButtonTarget()
{
    // TODO: Add your control notification handler code here
    CFileDialog dlg(false,NULL,NULL,OFN_HIDEREADONLY|OFN_OVERWRITEPROMPT);
    dlg.m_ofn.lpstrInitialDir = m_strTargetFile;
    dlg.m_ofn.lpstrTitle = "SelectFlatTitle";
    dlg.m_ofn.lpstrFilter = "*.txt\0*.txt\0\0";
    int len = m_strTargetFile.GetLength() - m_strTargetFile.ReverseFind('\\');
    strcpy(dlg.m_ofn.lpstrFile,    m_strTargetFile.Right(len - 1));
    if(dlg.DoModal() == IDOK)
    {
        m_strTargetFile = dlg.GetPathName();
        UpdateData(false);
    }
}
```

单击 IDC_BUTTON_TARGET 按钮，弹出保存文件对话框，上述代码为对话框提供了标题、保存的文件类型、默认的文件名。

（4）函数 OnOK()。

```
void CStudentDlg::OnOK()
{
    // TODO: Add extra validation here
    UpdateData(true);
    if(!CheckFile(m_strTargetFile))
        return;
    if(!ReadAndWrite(m_strSourceFile, m_strTargetFile))
        return;

    CDialog::OnOK();
}
```

单击 IDOK 按钮，首先检查目标文件名的有效性（由函数 CheckFile()完成，将在下面给出），如果是有效的文件名，则进行读写操作（由函数 ReadAndWrite()完成，将在下面给出），读写操作完成后，程序运行结束。

3.2.5　添加全局函数

在 StudentDlg.cpp 文件开始处添加文件包含，并添加三个全局函数原型。

注意：在添加文件包含时，如果文件中有对 stdAfx.h 文件的包含，一定要把其他文件包含放在 stdAfx.h 的下面。

```
#include <direct.h>
#include "StudentSet.h"
#include "ScoreSet.h"
#include "AddressSet.h"
```

```
bool CheckFile(CString strFileName);
void CreateDirectory(CString strDirectory);
bool ReadAndWrite(CString m_strSourceFile, CString m_strTargetFile);
```

其中函数 CheckFile()检查目标文件名的有效性，包括存放文件的路径是否存在或者文件是否已经存在等。函数 CreateDirectory()可以创建多级目录，函数 ReadAndWrite()从数据库中读取数据，并按要求的格式写到文本文件中。

在该文件中添加三个函数的定义。

（1）CheckFile()函数。

```
bool CheckFile(CString strFileName)
{
    int nDirectoryLen = strFileName.ReverseFind('\\');
    CString strDirectory = strFileName.Left(nDirectoryLen);
    CFileFind    find;
    int n = strFileName.ReverseFind('.');
    if(n == -1)
        strFileName += ".txt";

    if(find.FindFile(strFileName)) //如果文件已经存在，提示用户是否覆盖原有文件
    {
        int i=AfxMessageBox(strFileName + " already exists. Overwrite it?",
                            MB_YESNO|MB_ICONINFORMATION);
        if (i == IDYES)
            return true;
        else
            return false;
    }
    else
    {
        if(!find.FindFile(strDirectory + "\\*.*"))    //如果目标目录不存在，询问是否创建
        {
            int i=AfxMessageBox("The destination directory does not exist,
                                do you want to create it?",
                                MB_YESNO|MB_ICONINFORMATION);
            if(i == IDYES)
            {
                CreateDirectory(strDirectory);   // 调用该函数创建目录
                return true;
            }
            else
                return false; // Open binary file
        }
        return true;
    }
}
```

函数 CheckFile()首先检查参数 strFileName 是否有扩展名，如果没有，则认为是.txt 文件，

将.txt 追加到 strFileName 之后，然后判断该文件是否存在，如果存在，可以选择覆盖或不覆盖。如果选不覆盖，函数返回 false，可重新输入或选择文件名。如果选择覆盖，函数返回 true，程序继续运行。如果该文件不存在，再判断该文件的路径是否存在，如果不存在，用户可以选择创建该目录，此时调用函数 CreateDirectory()创建目录，函数返回 true，如果不选择创建目录，函数返回 false。

（2）CreateDirectory()函数。

```
void CreateDirectory(CString strDirectory)
{
    CString strParent = strDirectory.Left(strDirectory.ReverseFind('\\'));
    CFileFind find;
    if(!find.FindFile(strParent + "\\*.*"))
        CreateDirectory(strParent);
    _mkdir(strDirectory);
}
```

函数 CreateDirectory()通过递归调用，可以创建多级目录。函数_mkdir()是 MFC 定义的全局函数，用于创建文件夹。

（3）ReadAndWrite()函数。

```
bool ReadAndWrite(CString m_strSourceFile, CString m_strTargetFile)
{
    CStdioFile TextFile;
    CDaoDatabase db;
    TRY
    {
        db.Open(m_strSourceFile);
    }
    CATCH(CDaoException, e)
    {
        AfxMessageBox("The Access database " + m_strSourceFile +
                                    " can not be opened, please check the file name and path!");
        return false;
    }
    END_CATCH

    int n = m_strTargetFile.ReverseFind('.');
    if(n == -1)
        m_strTargetFile += ".txt";

    TextFile.Open(m_strTargetFile, CFile::modeWrite | CFile::modeCreate);     // Open text file

    TextFile.WriteString("No.\tName\tAge\tScore1\tScore2\tScore3\r\n");

    CStudentSet* pStudentSet = new CStudentSet(&db);
    CScoreSet* pScoreSet = new CScoreSet(&db);
    CAddressSet* pAddressSet = new CAddressSet(&db); // For Address table
```

```
pStudentSet->m_strSort = "[SNo]";
pStudentSet->Open();
pScoreSet->Open();
pAddressSet->Open();
pStudentSet->MoveFirst();
while(!pStudentSet->IsEOF())
{
    TextFile.WriteString(pStudentSet->m_SNo);
    TextFile.WriteString("\t");
    TextFile.WriteString(pStudentSet->m_SName);
    TextFile.WriteString("\t");
    CString strTemp;
    strTemp.Format("%d", pStudentSet->m_SAge);
    TextFile.WriteString(strTemp);
    TextFile.WriteString("\t");
    CString strSNo = pStudentSet->m_SNo;
    CString strSQL;
    strSQL = "Sno = '" + strSNo +"'";
    int isFind = pScoreSet->FindFirst(strSQL);
    if(isFind !=0)
    {
        strTemp.Format("%.1f",pScoreSet->m_Score1);
        TextFile.WriteString(strTemp);
        TextFile.WriteString("\t");

        strTemp.Format("%.1f",pScoreSet->m_Score2);
        TextFile.WriteString(strTemp);
        TextFile.WriteString("\t");

        strTemp.Format("%.1f",pScoreSet->m_Score3);
        TextFile.WriteString(strTemp);
        TextFile.WriteString("\t");
    }
    else
    {
        CString strMess = "The student with number ";
        strMess += strSNo;
        strMess += " were not found int the score table! ";
                strMess +="A special value 999 was put that place. ";
        AfxMessageBox(strMess);
        TextFile.WriteString("999\t999\t999");
    }
    pStudentSet->MoveNext();
    TextFile.WriteString("\r\n");
```

```
        }
        TextFile.Close();
        pStudentSet->Close();
        pScoreSet->Close();
        pAddressSet->Close();
        db.Close();
        return true;
    }
```

代码分析：函数 ReadAndWrite()是本程序的一个主要函数，首先打开数据库和输出的目标文件，并在输出文件的第一行输出表头。然后打开数据库中的三个表，其中在打开学生表时，按学号排序（由数据成员 m_strSort 指定）。

将学生表的记录指针移到第一条记录，从第一条记录开始循环，将学号、姓名、年龄输出到目标文件相应的位置，在成绩表中查找与该学号相同的记录，如果查到将其三门成绩输出到目标文件中，如果在成绩表中找不到相应的学号，则将一个特殊值“999”输出到对应的位置。

在表中查找指定的记录由成员函数 FindFirst()实现，其参数就是查找条件，语法格式为 SQL（结构化查询语句）的 WHERE 子句。假设在学生表中的学号是“990101”，则参数变量 strSQL 的值就是“Sno='990101'”，表示要在成绩表中查找字段 Sno 为“990101”的记录。

将三门成绩输出后，调用函数 MoveNext()，将学生表中的记录指针向后移动一条记录，进行下一次循环。循环结束后关闭所有打开的文件。

测试运行程序，打开输出的文件，查看是否符合要求。

注意：*在源程序文件夹中已包含了本程序中使用的 Access 数据库* Student.mdb。

3.3　相关知识

3.3.1　CDaoDatabase 类

CDaoDatabase 对象代表与数据库的一个连接，DAO 是数据访问对象（Data Access Object）的缩写，通过它可以操作数据库中的数据。在使用 CDaoDatabase 类时，需要包含相应的头文件。

```
#include <afxdao.h>
```

CDaoDatabase 类有很多成员函数，通过这些函数，可以对数据库进行各种操作。本程序只用到其中的两个函数：Open()和 Close()，分别用来建立数据库连接和关闭数据库连接。Open()函数的原型如下：

```
virtual void Open( LPCTSTR lpszName, BOOL bExclusive = FALSE, BOOL bReadOnly = FALSE,
                   LPCTSTR lpszConnect = _T("") );   throw( CDaoException, CMemoryException );
```

参数：

lpszName：要打开的数据库文件名。

bExclusive：是否以独占方式打开数据库，如果是，赋值为真（TRUE）；如果以共享方式打开，则赋值为假（FALSE）。

bReadOnly：是否以只读方式打开数据库，如果是，赋值为真（TRUE），否则赋值为假（FALSE）。

lpszConnect：提供 ODBC 连接的参数，如果是 Microsoft Jet database (.MDB)，该字符串

应该是空字符串。

在使用该函数时，如果要打开一个 Microsoft Jet (.MDB) 数据库，使用参数 lpszName 指定数据库名，参数 lpszConnect 为空字符串或者是提供数据库访问口令（如果数据库有口令），其格式为“;PWD=password”。

如果要打开 ODBC 数据源，参数 lpszConnect 应该是一个有效的数据源，而参数 lpszName 应该为空字符串。

Close()函数的原型如下：

```
virtual void Close( );
```

调用该函数关闭数据库。

3.3.2 CDaoRecordset 类

CDaoRecordset 对象代表数据源的一组记录，可以是一个表或一次查询。

CDaoRecordset 类有很多成员函数和数据成员，下面对其中主要的成员函数和数据成员作一下简单介绍。

1. 数据成员 m_strSort

CString 类型，指定记录的排列顺序，包含一个 SQL 语句的 ORDER BY 子句（但不含保留字 ORDER BY）。

2. 数据成员 m_strFilter

CString 类型，指定记录过滤条件，包含一个 SQL 语句的 WHERE 子句（但不含保留字 WHERE）。

3. 构造函数 CdaoRecordset()

```
CDaoRecordset( CDaoDatabase* pDatabase = NULL );
```

参数 pDatabase 包含一个 CDaoDatabase 对象的指针或空指针。

4. 函数 Open()

```
virtual void Open( int nOpenType = AFX_DAO_USE_DEFAULT_TYPE, LPCTSTR lpszSQL = NULL,
int nOptions = 0 );   throw( CDaoException, CMemoryException );
```

参数：

nOpenType：打开的方式。

lpszSQL：包含一个 SQL 语句。

nOptions：对记录操作的方式。

有关三个参数的详细解释请参考 MSDN，一般可以使用默认值打开一组记录。

5. 函数 MoveFirst()

```
void MoveFirst( );    throw( CDaoException, CMemoryException );
```

使记录集的第一条记录成为当前记录。

6. 函数 MoveLast()

```
void MoveLast( );    throw( CDaoException, CMemoryException );
```

使记录集的最后一条记录成为当前记录。

7. 函数 MoveNext()

```
void MoveNext( );   throw( CDaoException, CMemoryException );
```

使当前的下一条记录成为当前记录。

8. 函数 MovePrev()

void MovePrev( );　throw(CDaoException, CMemoryException);

使当前的上一条记录成为当前记录。

9. 函数 FindFirst()

BOOL FindFirst(LPCTSTR *lpszFilter*);　throw(CDaoException, CMemoryException);

使第一个满足条件的记录成为当前记录。如果找到满足条件的记录返回 TRUE，否则返回 FALSE。

参数：

lpszFilter：包含一个字符串，类似 SQL 语句的 WHERE 子句，但不包含 WHERE 保留字 。

与 FindFirst()类似的函数还有 FindLast()，FindNext()，FindPrev()。

3.3.3 CStdioFile 类

CStdioFile 类是从 CFile 类继承的子类，用于处理流式文件。一个 CStdioFile 对象代表一个用函数 fopen()打开的 C 流式文件。可以用文本方式（默认）或二进制方式打开。文本方式提供对硬回车—换行符对的特殊处理。当将一个换行符（0x0A）写入一个文本方式的 CStdioFile 对象中时，字节对（0x0A，0x0D）被发送给该文件。当读一个文件时，字节对（0x0A，0x0D）被翻译为一个字节（0x0A）。由于这种特性，CStdioFile 类常被用于文本文件的处理。为了方便文本处理，CStdioFile 类还提供了两个专门读写文本文件的函数。

1. ReadString 函数

ReadString 函数的原型如下：

BOOL ReadString(CString& rString);　throw(CFileException);

返回值：返回一个布尔型值，当无数据可读时，返回 FALSE，否则返回 TRUE。

参数：

rString：是一个对 CString 对象的引用，当函数返回时，该对象包含了所读取的字符串。

此成员函数用换行符将文本文件分为若干行，然后将文本以行为单位读出到参数 rString 中。读出的数据将忽略回车—换行符。

2. WriteString 函数

WriteString 函数的原型如下：

virtual void WriteString(LPCTSTR lpsz);　throw(CFileException);

参数：

lpsz：指定一个指向存放以空字符结尾的文本字符串的缓冲区的指针。

说明：此成员函数将一个缓冲区中的数据写入文件，并且参数 lpsz 中的所有换行符都被转换成一个硬回车—换行符对写入。

需要注意的是，WriteString()函数并非是以行为单位，即只写入参数所传的字符串，不会自动加入回车—换行符。所以当需写入一行时，必须在字符串后加回车—换行符。

由于 CStdioFile 类是 CFile 类的子类，所以 CFile 类的成员函数基本都可以在 CStdioFile 类中使用。例如 CstdioFile 类的 Open()函数与 CFile 类的 Open()函数的使用方法是一样的。

3.4 本章小结

本章通过一个实例介绍了用 DAO 访问数据库的技术和方法以及文本文件的读写方法，介绍了 DAO 技术的一般知识，较系统地介绍了 CDaoDatabase 类、CDaoRecordset 类和 CStdioFile 类的数据成员和成员函数。

3.5 思考题

1．要访问 DAO 数据对象，需要在 stdAfx.h 文件中加入对哪个头文件的包含？

2．在打开数据库表时要按指定的字段排序记录，应如何处理？

3．在进行数据查询时，要用 CDaoRecordSet 类的什么函数？其参数是什么？

4．本实例程序中，CheckFile()函数的作用是什么？

5．CDaoRecordSet 类的成员函数 MoveFirst()，MoveLast()，MoveNext()，MovePrev()分别有什么功能？