

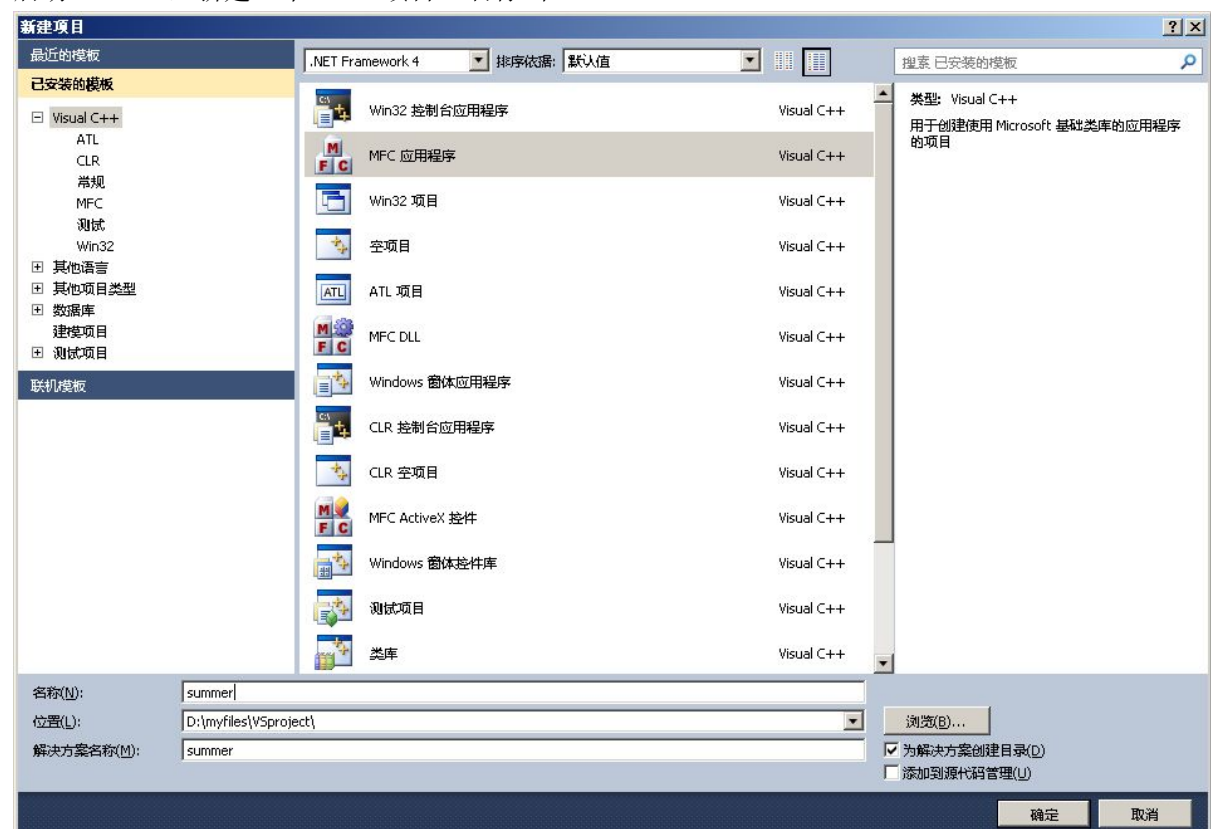
本案例通过 VS2010 环境下的 MFC ODBC 使用者连接 SQL Server 数据库
因为是通过 ODBC 数据源连接的，在此我认为关于 SQL Server 的版本是无关紧要的。
数据库准备：

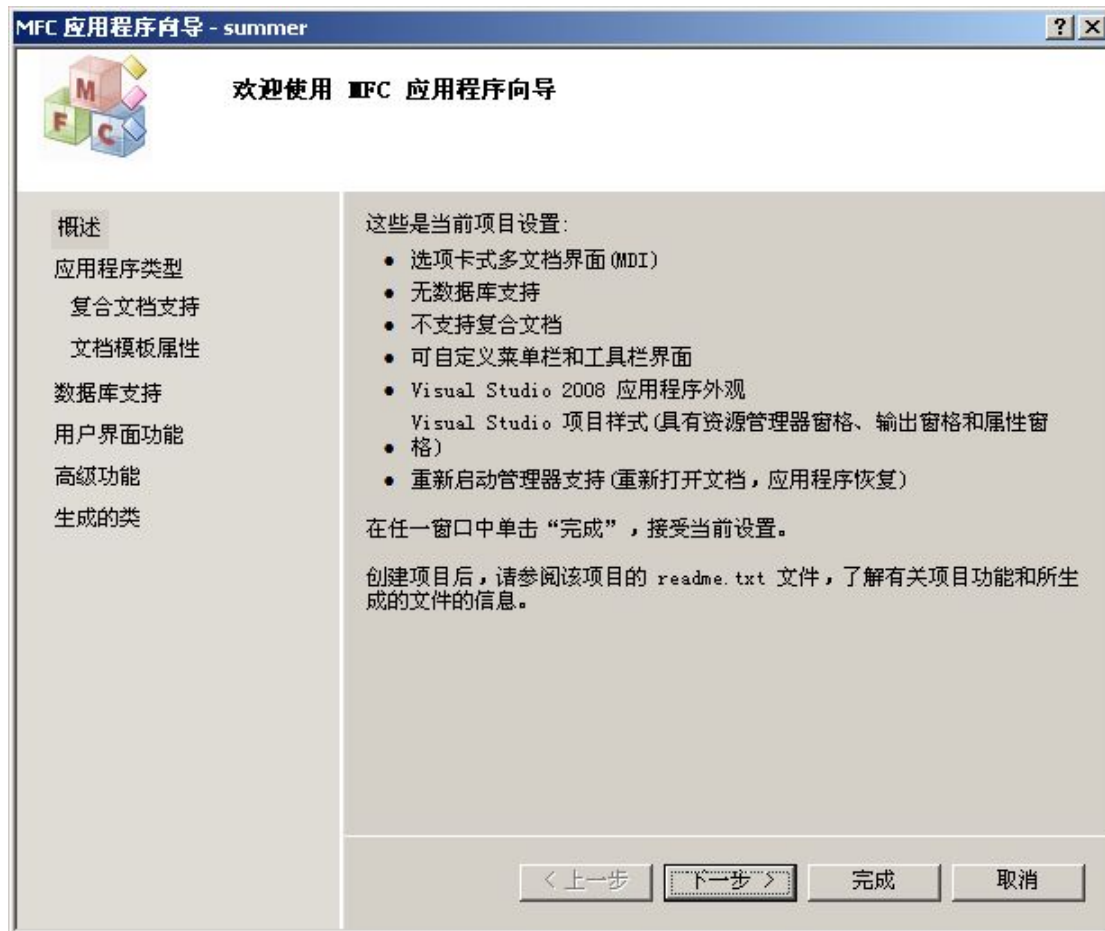
```
--先要手动创建一个数据库，名字叫studentdb
--创建好了数据库studentdb就可以执行下面的sql语句了
--这些语句执行完了以后，会在studentdb中建一个student表，然后插入四条数据

use studentdb;
create table student(
sno varchar(10),
sname varchar(10) not null,
sage int not null,
pwd varchar(10) not null);

insert into student(sno,sname,sage,pwd)
values('0001','jack',20,'7777');
insert into student(sno,sname,sage,pwd)
values('0102','tom',20,'8888');
insert into student(sno,sname,sage,pwd)
values('3472','cat',20,'9999');
insert into student(sno,sname,sage,pwd)
values('5716','dog',20,'2013');
--数据库建表工作就到这里这么顺利的完成了
```

启动 VS2010，新建一个 MFC 项目。名称叫 summer。





通过 MFC 应用程序向导进行项目设置。下一步



应用程序类型 选择 基于对话框



主框架样式 都不要了。全部取消勾选。进入下一步



高级功能就默认，进入下一步



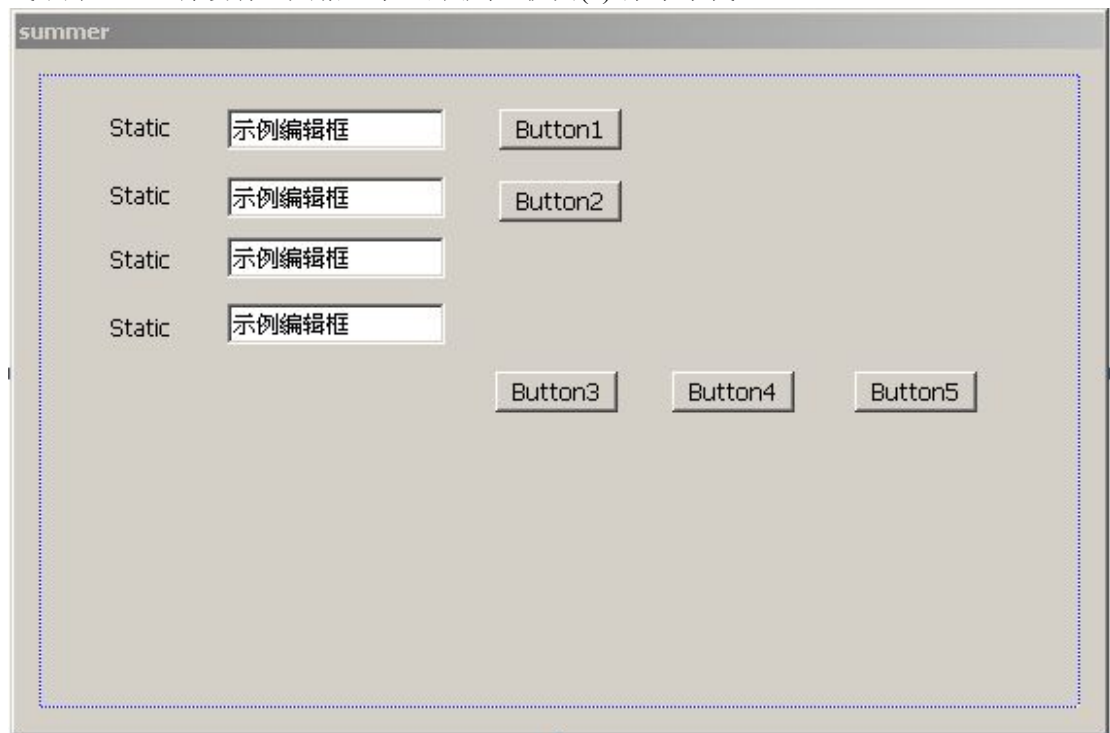
生成的类选 **CsummerDlg**

点击完成,就完成了 MFC 应用项目的创建。完成以后要等很久, VS2010 最底下显示“正在分析包含的文件……”, 第一次新建项目要等很久, 第二次也要等一段时间。

一切就绪以后, 会看到对话框面板设计。把默认建立的控件都删了。

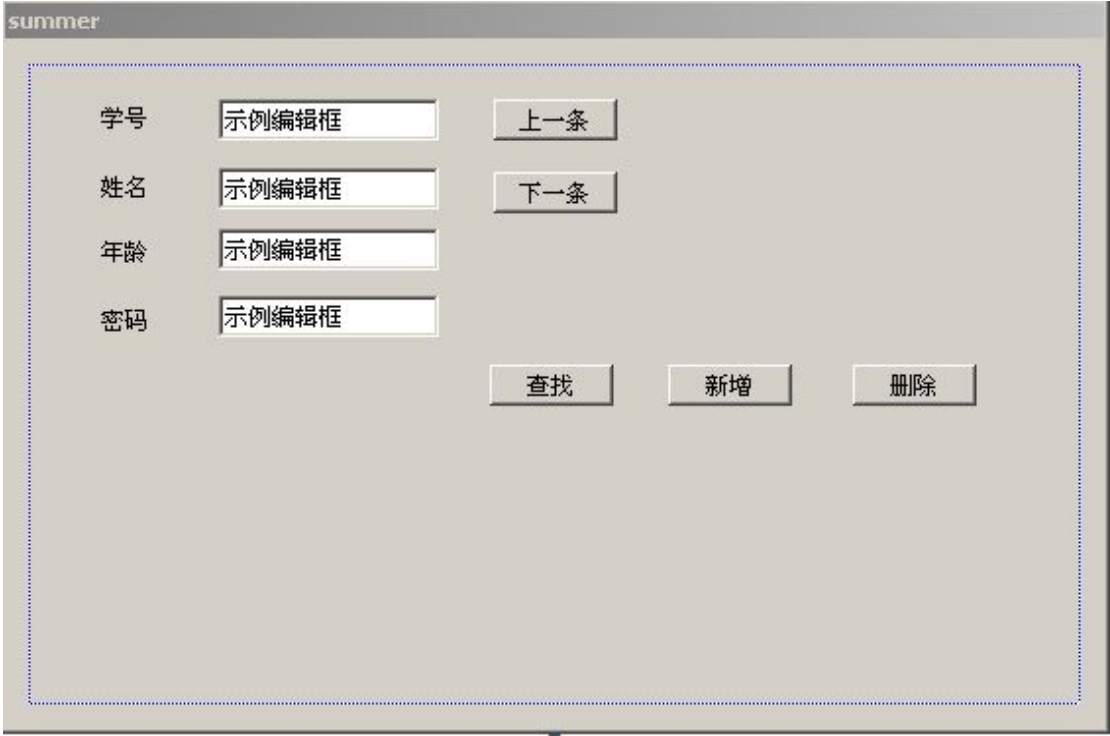
像“TODO:在此放置对话框控件。”、“确定”、“取消”按钮, 都删了。我们不要那些。

删光了之后, 面对光秃秃的对话框面板我们开始设计, 控件都在屏幕左边的工具箱。要是运气不好, 左边都没有工具箱显示, 那就在“视图(v)”菜单中找吧。



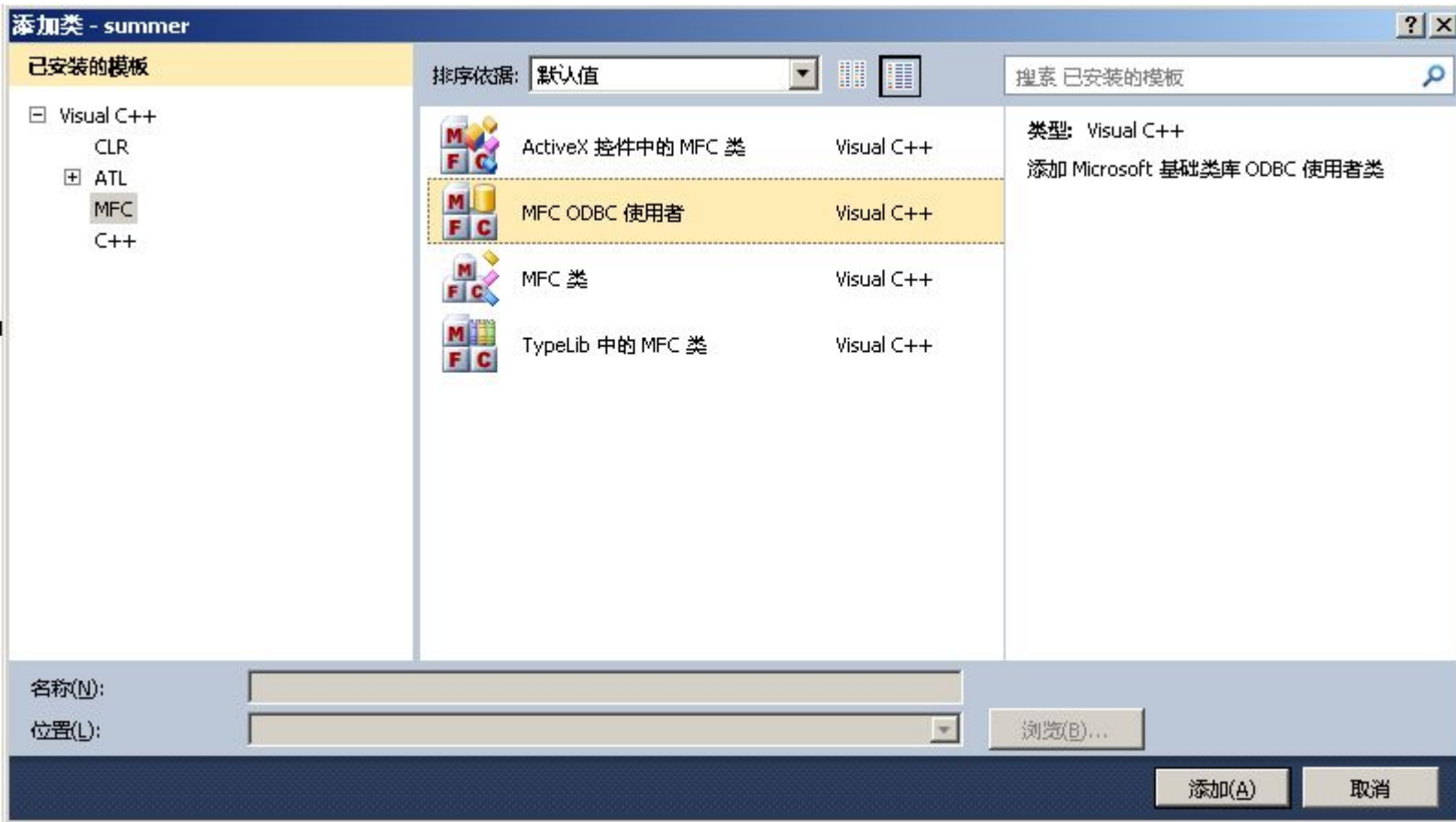
控件设计。在此用到了三种控件，分别是 Edit control、Static Text、Button。

多个控件对齐小技巧：比如选中 4 个 Static 文本控件，按下 ctrl+shift+方向键左 或 右，它们就对齐了。



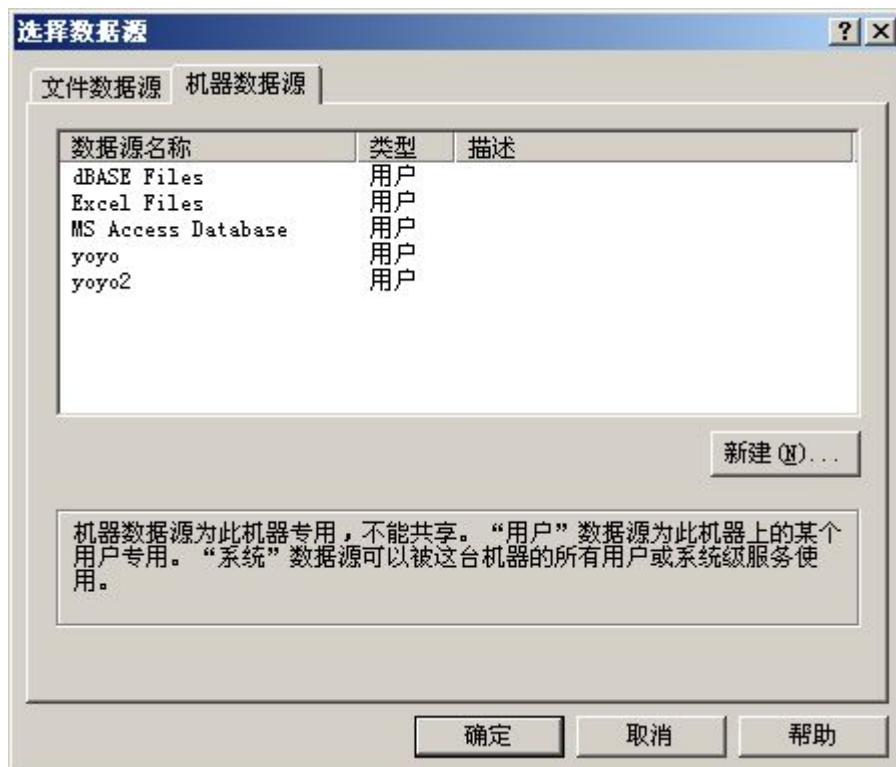
控件放好了，开始修改控件 caption。点击一个控件，能够在左边“外观”一栏修改 Caption。

界面显示的设计都基本完成。接下来要新建一个 ODBC 类。右击项目名称，选择 添加→类→MFC ODBC 使用者





通过 MFC ODBC 使用者向导进行数据源的配置。在这里点击“数据源”



选择机器数据源，因为还没有我们需要的数据源，所以点击“新建”



下一步

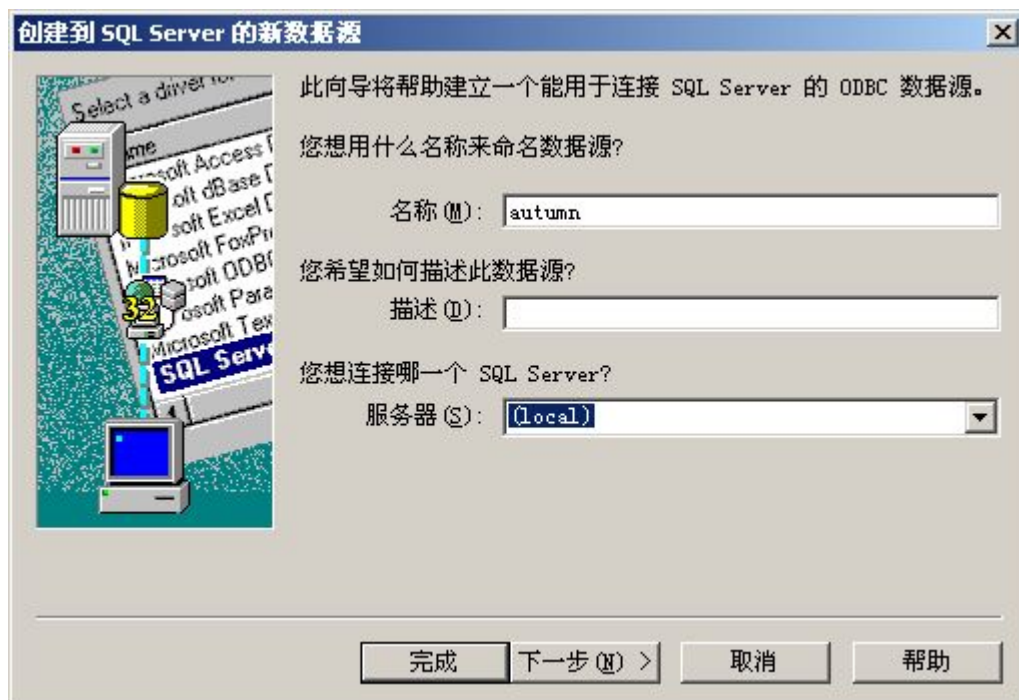


数据源驱动程序选择 SQL Server

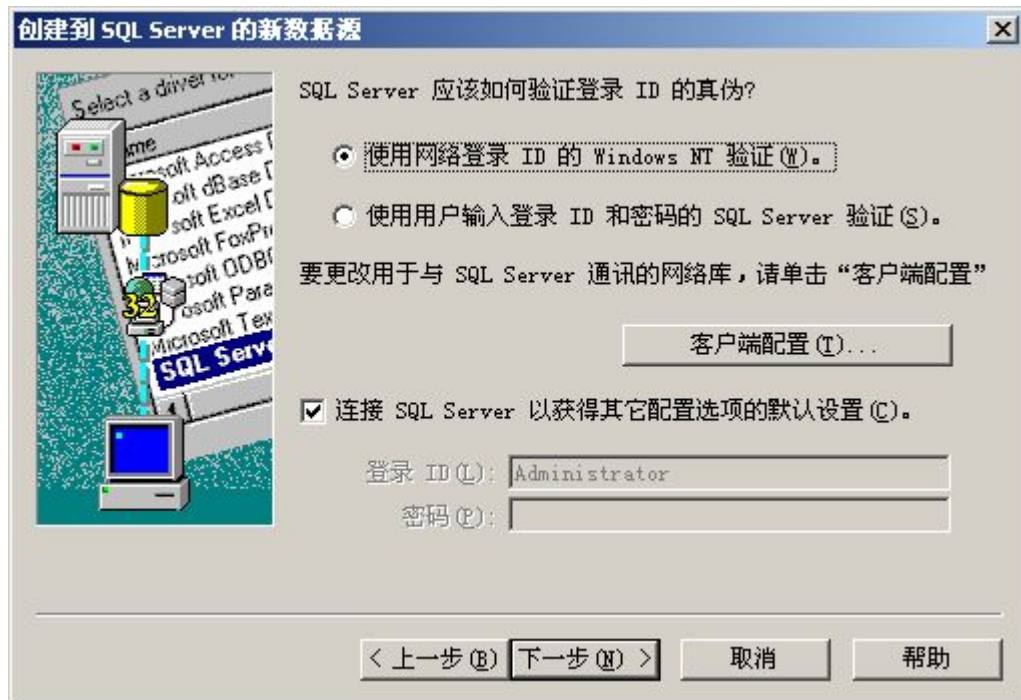
下一步



数据源新建完成。



取个名字叫 autumn，服务器下拉，选择 (local)
下一步

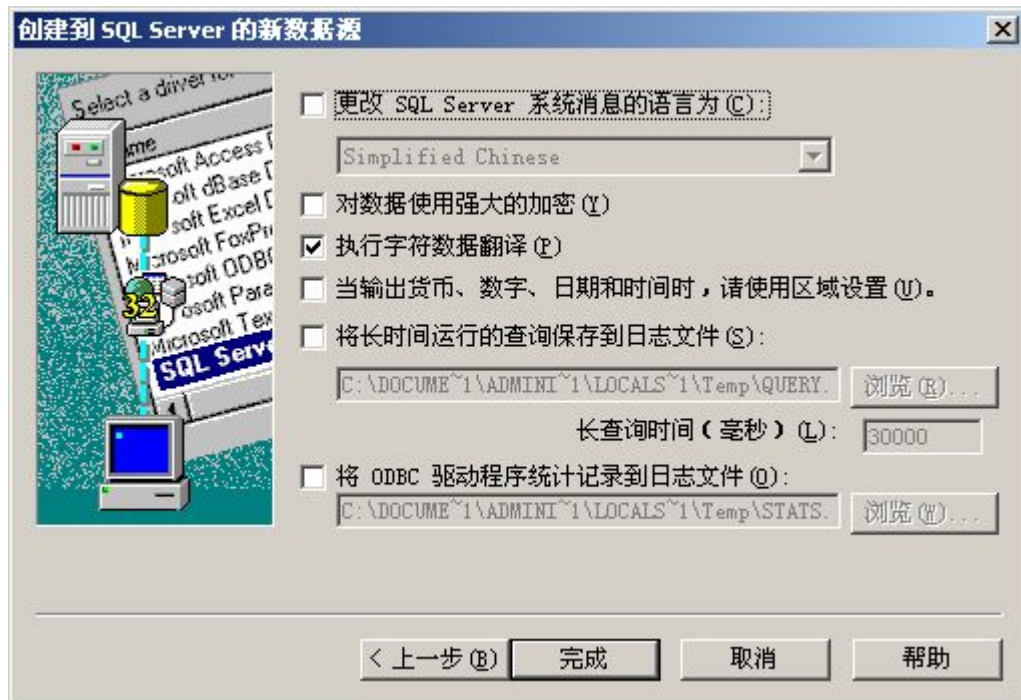


下一步



更改默认的数据库为一开始新建的 [studentdb](#)

下一步



完成

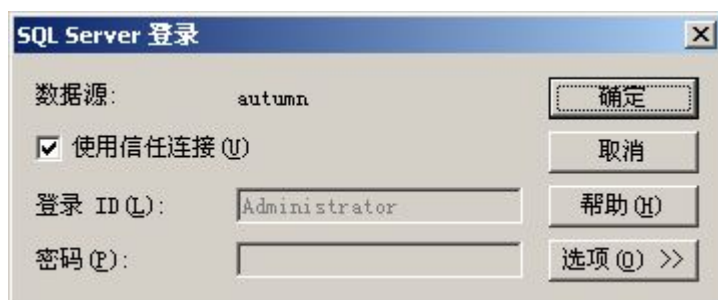


测试数据源 一般不会有问题

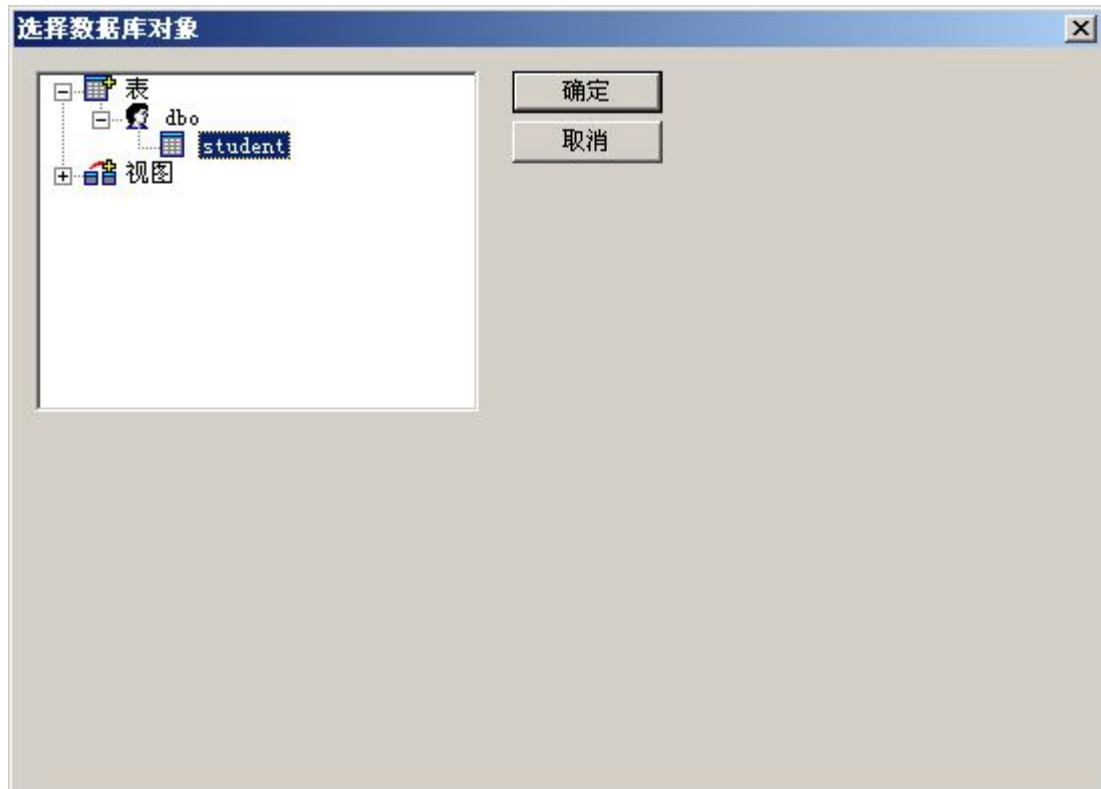
确定



这时候机器数据源下就有了我们刚创建的 autumn，选择 autumn，确定



确定

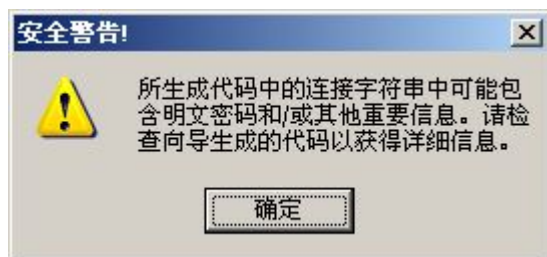


选择我们之前建好的 `student` 表。

根据我目前的经验，一张表对应一个 ODBC 使用者类。如果要用到很多表，那么就要创建很多 ODBC 使用者类。



此时生成了一个对应那张 student 表的 MFC ODBC 使用类，这个类名叫 Cstudent。
MFC ODBC 使用者创建完成。



点 击 完 成 以 后 ， 会 跳 出 一 个 安 全 警 告 。 不 理 他 。


```
#error 安全问题：连接字符串可能包含密码。
// 此连接字符串中可能包含明文密码和/或其他重要
// 信息。请在查看完此连接字符串并找到所有与安全
// 有关的问题后移除 #error。可能需要将此密码存
// 储为其他格式或使用其他的用户身份验证。
CString Cstudent::GetDefaultConnect()
{
    return _T("DSN=autumn;APP=Microsoft\u00ae Visual Studio\u00ae 2010;WSID=LILI;DATABASE=s");
}

CString Cstudent::GetDefaultSQL()
{
    return _T("[dbo].[student]");
}

void Cstudent::DoFieldExchange(CFieldExchange* pFX)
{
    pFX->SetFieldType(CFieldExchange::outputColumn);
    // RFX_Text() 和 RFX_Int() 这类宏依赖的是
    // 产品本身的类别，而不是数据库本身的类别
}

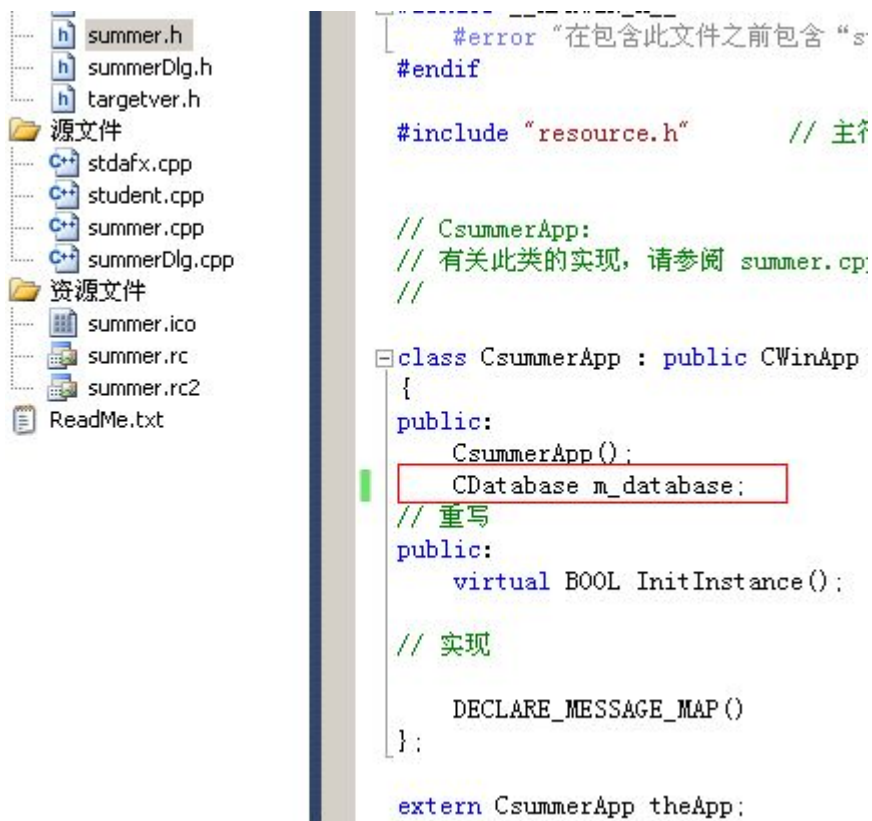
100 %
```

输出

显示输出来源(S): 生成

```
1> ClCompile:
1> stdafx.cpp
1> summerDlg.cpp
1> summer.cpp
1> student.cpp
1> d:\myfiles\vsproject\summer\summer\student.cpp (23): fatal error C1189: #error : 安全问题：连接字符串可能包含
1> 正在生成代码...
```

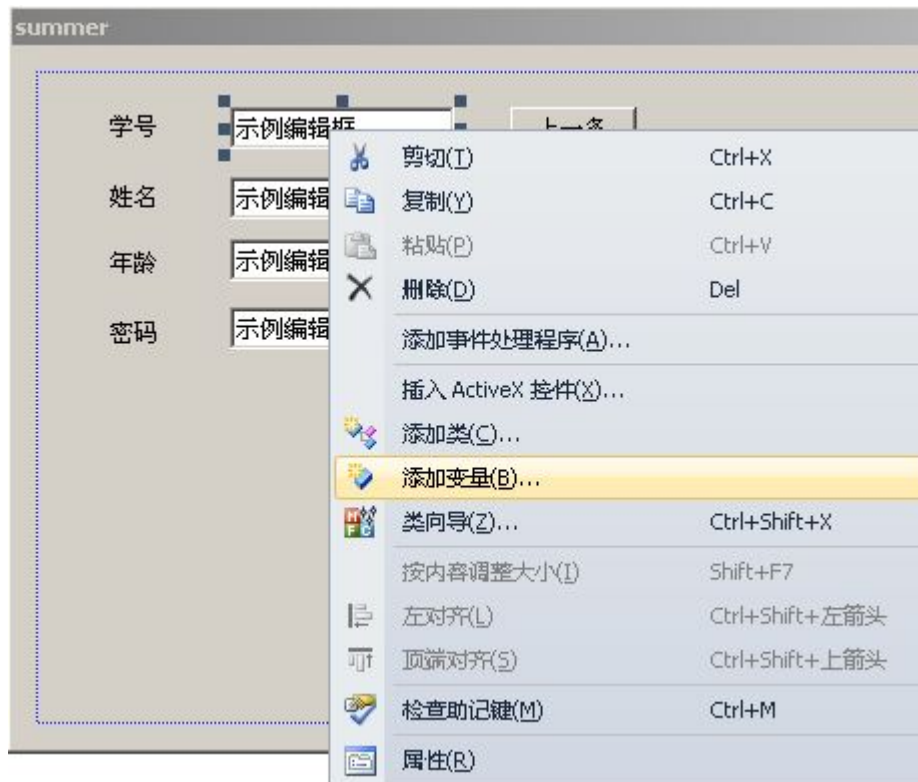
新建完了以后 ODBC 使用者类，VS2010 替我们生成了这么多代码，此时按下 F7 进行编译，会出错，双击错误，跳转到出错代码，是一个安全问题，我们把出错的这一行注释掉，再按 F7 编译就没有问题了。



打开 summer.h, 在其中声明一个 CDatabase 类型的成员变量 m_database



打开 summerDlg.cpp , 新包含 student.h 这个头文件, 这样一来就直接使用 Cstudent 这个 ODBC 使用者类了。



在对话框设计面板，右击第一个 Edit Control 控件,选择“添加变量”

如果找不到控件面板设计界面，就在 视图→资源视图→Dialog 找到之。

项目代码的管理则是在解决方案管理器中（默认在左边显示），左下方可以切换。

添加成员变量向导 - summer

欢迎使用添加成员变量向导

访问(A): public

变量类型(V): CString

变量名(N): c_sno

控件 ID(I): IDC_EDIT1

控件类型(Y): EDIT

类别(T): Value

最大字符数(X):

最小值(M):

最大值(M):

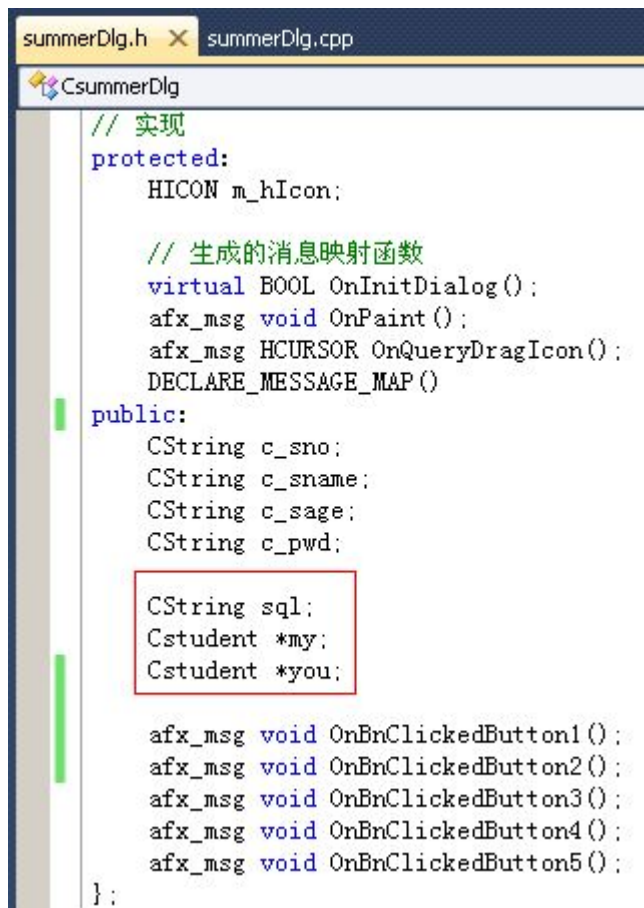
.h 文件(F):

.cpp 文件(C):

注释(M) (// 不需要 表示法):

完成 取消

类别选择 Value 型，变量名填 c_sno。其实按照规范，这里的变量名应当是 m_sno，但是为了不与 ODBC 使用者里面的 m_sno 混淆，我还是写 c_sno。表示是编辑控件(IDC_EDIT1)的成员变量。类似的，剩下三个编辑框控件（Edit Control），都通过这样的方式新建成员变量，变量类别都是 Value。变量名分别是 c_sname, c_sage, c_pwd。



```
summerDlg.h X summerDlg.cpp
CsummerDlg
// 实现
protected:
    HICON m_hIcon;

    // 生成的消息映射函数
    virtual BOOL OnInitDialog();
    afx_msg void OnPaint();
    afx_msg HCURSOR OnQueryDragIcon();
    DECLARE_MESSAGE_MAP()
public:
    CString c_sno;
    CString c_sname;
    CString c_sage;
    CString c_pwd;

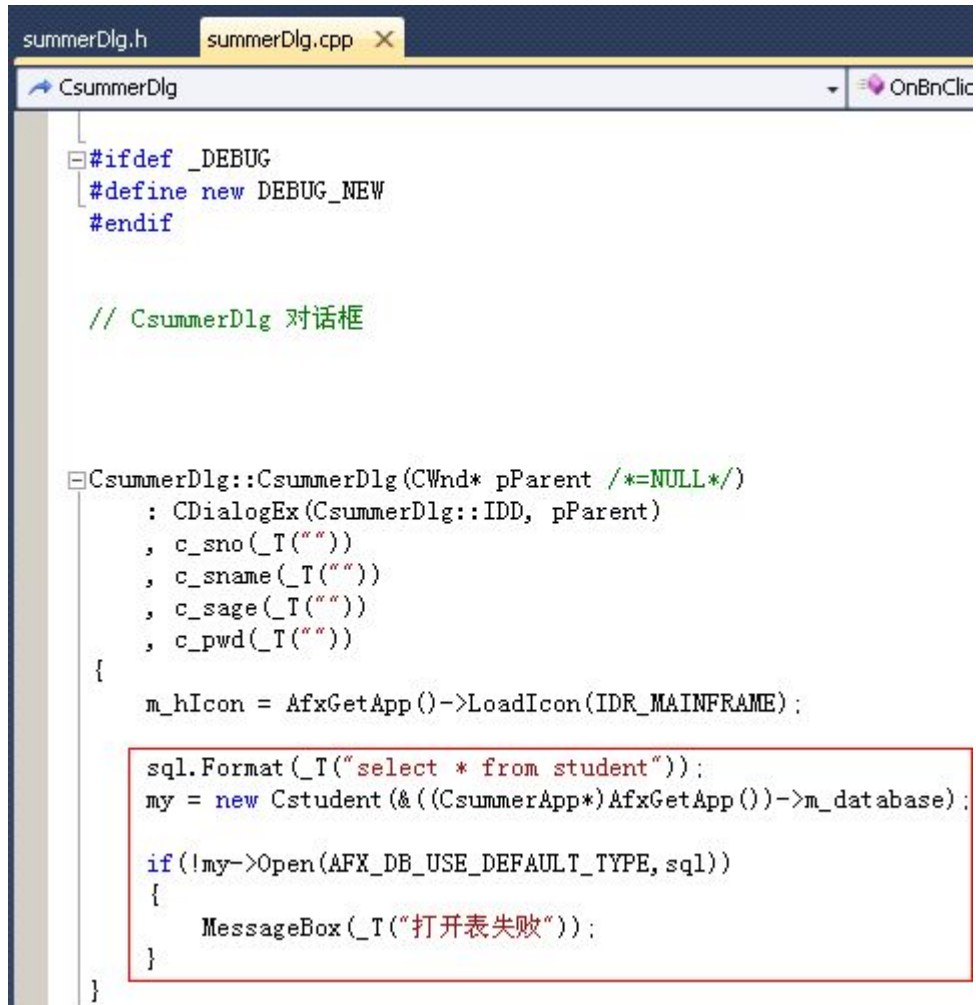
    CString sql;
    Cstudent *my;
    Cstudent *you;

    afx_msg void OnBnClickedButton1();
    afx_msg void OnBnClickedButton2();
    afx_msg void OnBnClickedButton3();
    afx_msg void OnBnClickedButton4();
    afx_msg void OnBnClickedButton5();
};
```

打开 summerDlg.h 这个头文件。可以看到之前添加的控件成员变量都在这里出现了。需要指出的是：如果是直接手动在这里声明变量，那么效果比起上一步操作中通过添加控件的成员变量的效果是不一样的。我的理解就是控件的成员变量是和控件绑定的，是属于控件的，而手动声明的变量仅仅是类的成员变量，并没有和控件一一对应起来。

这里我们还需要手动声明三个成员变量，见图片红框中，至于为什么要声明他们，最好的解释就是因为后面要用。不声明就没法用啊。

至于最下面的那些 `afx_msg void OnBnClickedButton1();` 也许这时候你的 summerDlg.h 里面还没出现，如果你有点经验的话应该知道这些函数也是自动生成的——当你双击一个设计好的按钮。



```
summerDlg.h    summerDlg.cpp X
CsummerDlg
OnBnClick

#ifdef _DEBUG
#define new DEBUG_NEW
#endif

// CsummerDlg 对话框

CsummerDlg::CsummerDlg(CWnd* pParent /*=NULL*/)
: CDialogEx(CsummerDlg::IDD, pParent)
, c_sno(_T(""))
, c_sname(_T(""))
, c_sage(_T(""))
, c_pwd(_T(""))
{
    m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);

    sql.Format(_T("select * from student"));
    my = new Cstudent(&((CsummerApp*)AfxGetApp())->m_database);

    if(!my->Open(AFX_DB_USE_DEFAULT_TYPE, sql))
    {
        MessageBox(_T("打开表失败"));
    }
}
```

打开 summerDlg.cpp,找到图片中那个方法的实现部分。添加红框中的代码。

这里一共添加了三段代码。第一段是个格式化，第二段和第三段是固定写法。我目前也解释不清楚。其中 sql 和 my 都是在 summerDlg.h 这个头文件中声明过的变量，而 m_database 是 summer.h 中手动声明的变量，那个 CsummerApp 也能在 summer.h 中找到。

```

void CsummerDlg::OnBnClickedButton1()
{
    // “上一条” 按钮
    my->MovePrev();
    if (my->IsBOF() != 1) //如果没有到最前面
    {
        c_sno = my->m_sno;
        c_sname = my->m_sname;
        c_sage.Format(_T("%d"), my->m_sage);
        c_pwd = my->m_pwd;
        UpdateData(FALSE);
    }
    else
    {
        AfxMessageBox(_T("已经到最前了"));
        my->MoveNext();
    }
}

```

双击“上一条”按钮开始写代码。

```

void CsummerDlg::OnBnClickedButton2()
{
    // “下一条” 按钮
    my->MoveNext();
    if (my->IsEOF() != 1) //如果没有到最后
    {
        c_sno = my->m_sno;
        c_sname = my->m_sname;
        c_sage.Format(_T("%d"), my->m_sage);
        c_pwd = my->m_pwd;
        UpdateData(FALSE);
    }
    else
    {
        AfxMessageBox(_T("已经到最后了"));
        my->MovePrev();
    }
}

```

双击“下一条”按钮，写代码。

“上一条”和“下一条”按钮代码完成以后，按 F5 编译运行，就能够看到效果了。


```

void CsummerDlg::OnBnClickedButton3()
{
    // “查找” 按钮
    GetDlgItemTextW(IDC_EDIT1, c_sno);
    CString sql2;
    you = new Cstudent(&((CsummerApp*)AfxGetApp())->m_database);
    sql2.Format(_T("select * from student where sno = '%s' "), c_sno);

    if(!you->Open(AFX_DB_USE_DEFAULT_TYPE, sql2))
    {
        MessageBox(_T("打开表失败"));
    }
    if((you->m_sno = c_sno) && you->GetRecordCount() != 0)
    {
        c_pwd = you->m_pwd;
        c_sage.Format(_T("%d"), you->m_sage);
        c_sname = you->m_sname;
        SetDlgItemTextW(IDC_EDIT2, c_sname);
        SetDlgItemTextW(IDC_EDIT3, c_sage);
        SetDlgItemTextW(IDC_EDIT4, c_pwd);
    }
    else
    {
        MessageBox(_T("查无此人"));
    }
}

```

“查找”按钮的代码。这里的查找功能实现了通过学号查找其他信息。

```

void CsummerDlg::OnBnClickedButton4()
{
    // “新增” 按钮
    UpdateData(TRUE);
    my->AddNew(); //如果写成 my->Edit() 就是修改操作
    my->m_sname = c_sname;
    my->m_sno = c_sno;
    my->m_sage = _ttoi(c_sage); //把CString转为int型
    my->m_pwd = c_pwd;

    my->Update();
    my->Requery();
    MessageBox(_T("添加成功"));
}

```

“添加”按钮的代码。如果要做成修改功能，把 AddNew()改成 Edit()即可。

```

void CsummerDlg::OnBnClickedButton5()
{
    // “删除” 按钮
    my->Delete();
    my->Requery();
    MessageBox(_T("删除成功"));

    /*删除完了以后把记录集指针移到前一个*/
    my->MovePrev();
    c_sno = my->m_sno;
    c_sname = my->m_sname;
    c_sage.Format(_T("%d"), my->m_sage);
    c_pwd = my->m_pwd;
    UpdateData(FALSE);
}

```

“删除”按钮对应的代码。

按下 F5 编译运行。即可查看所有功能效果。



在此解释一下，UpdateData(TURE)的功能就是获取编辑控件上输入的数据，这些数据在UpdateData(TURE)之后都被赋值给对应的控件成员变量。相应的，UpdateData（FALSE）就是把这些变量在编辑控件中显示。

至此，通过 VS2010 的 MFC ODBC 使用者连接 SQL Server，实现增、删、改、查的操作案例也告一段落了。

作者：[二里荔枝](#)