**数据库试验指示书**

(3)

清华大学电子工程系

2024年11月

实验四 用VC++开发数据库应用程序

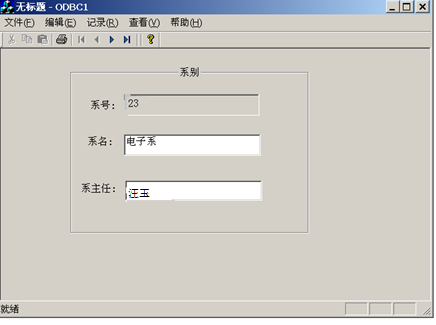
试验目的：

熟悉利用Microsoft Visual C++进行数据库ODBC应用程序的开发，掌握面向对象的程序设计以及可视化编程的一般方法。

试验内容：

用VC++自动和手动混合编程，将实验二中建立的DEPT、STU、TEACHER关系（可以任选两个），按照下面操作说明的步骤，给每个关系建立一个窗口式用户客户端界面，实现关系元组的显示、插入、删除、更新、清空的功能。

**说明:** 在TEACHER表中有一个datetime类型的数据WDATE，在STU表中有一个datetime类型的数据EDATE，由于datetime类型在VC++6.0中没有直接对应的数据类型, 可以忽略这个WDATE或EDATE，界面上不用显示。



试验步骤：

按照实验指示书3的操作说明进行。

课前准备：

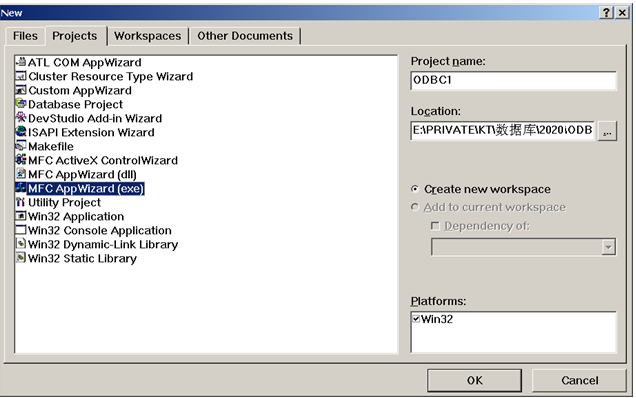
上机前认真阅读实验指示书3的操作说明，搞清其中的含义。可参阅VC++6.0或其它更高版本编译器的操作手册和有关书籍。

**操作说明：**

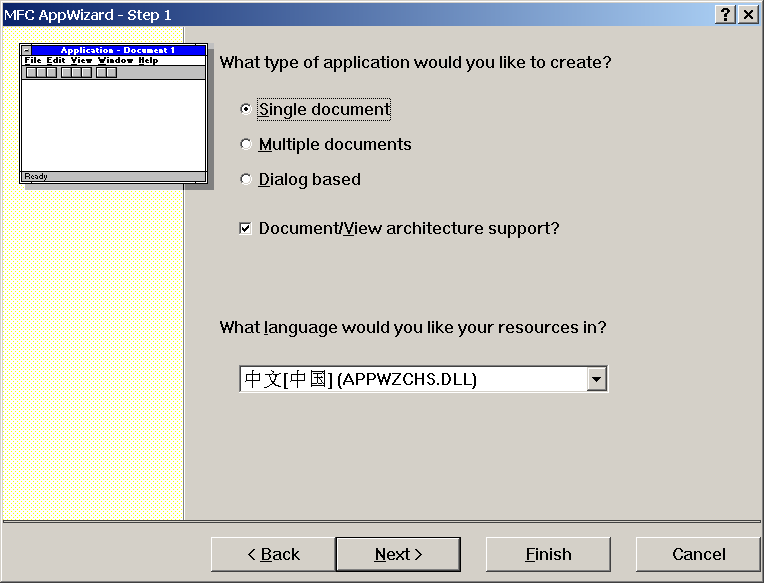
1. 首先设置好ODBC数据源。可参照前一个实验的有关说明。这里的数据源名是student。

**第一步 生成数据库应用程序框架**

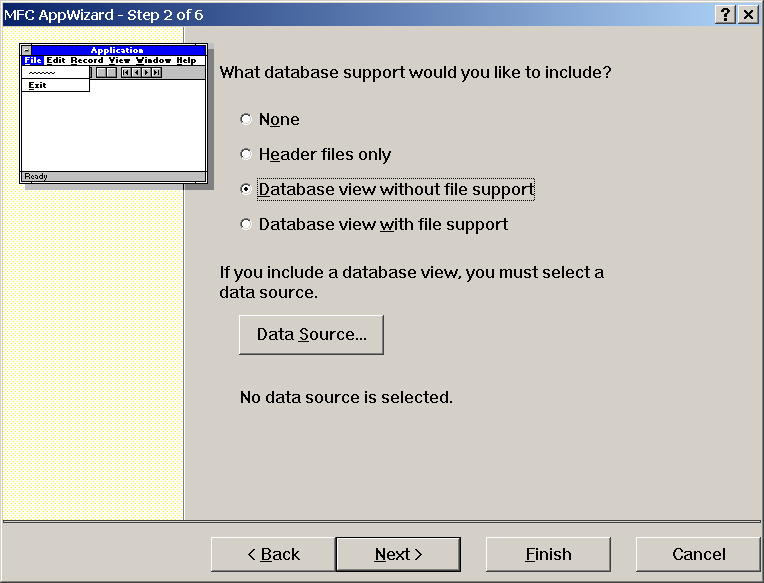
1. 在VC++6.0的菜单中，选择File下拉菜单中的New选项，会弹出New对话框。单击Projects标签，显示Projects属性项列表框，选择MFC AppWizard(exe)项，在Project Name文本框中输入“ODBC1”。



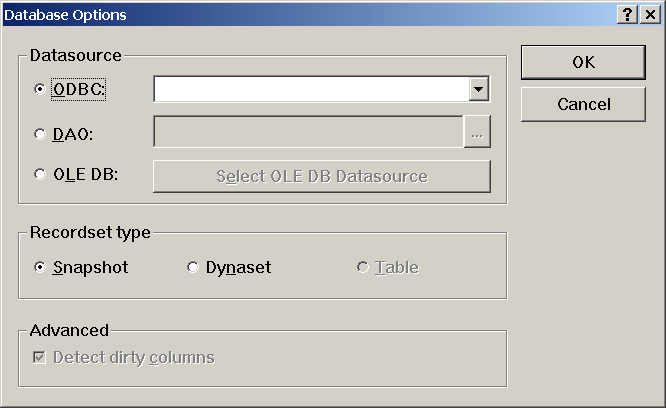
3．单击OK，进入AppWizard创建过程，显示Step1。选择其中的Single document选项，使得应用程序不能同时打开多个窗口。



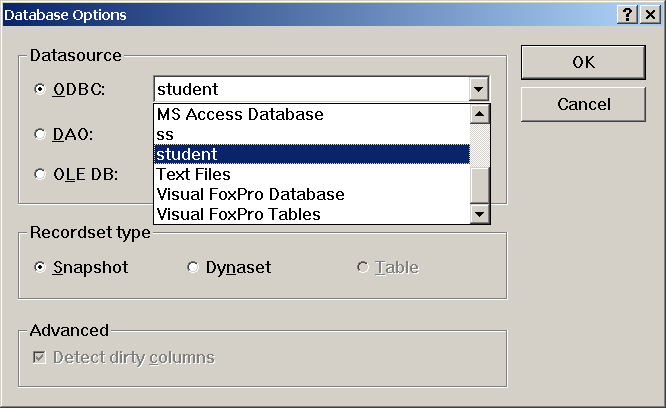
4．单击Next按钮，显示Step 2 of 6。选择其中的Database view without file support选项，使AppWizard产生查看数据库内容的类。



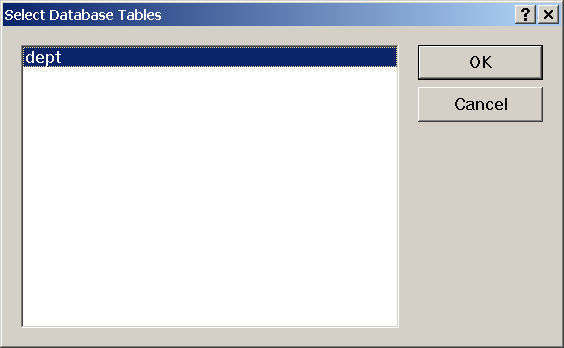
5．单击按钮Data Source，建立应用程序与数据源student之间的连接。



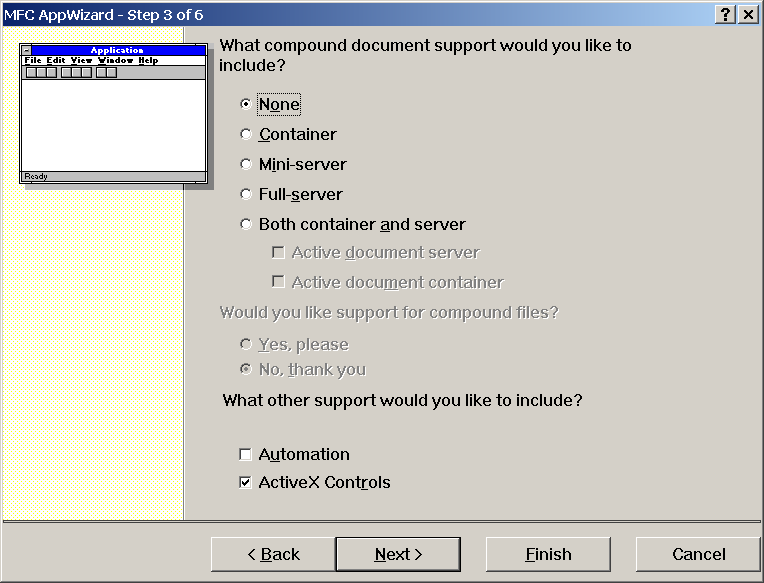
6．从下拉菜单中选择student数据源：



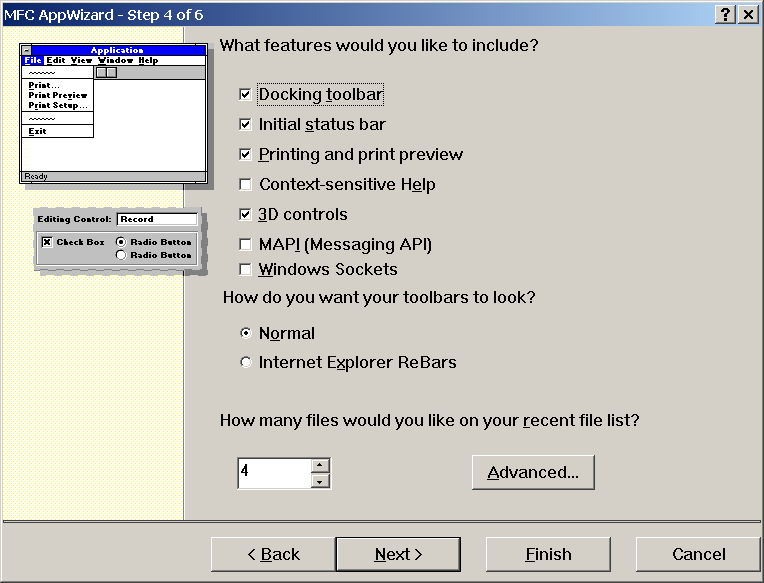
7．单击OK按钮，在打开的Select Database Tables对话框中，选择“dept”表：



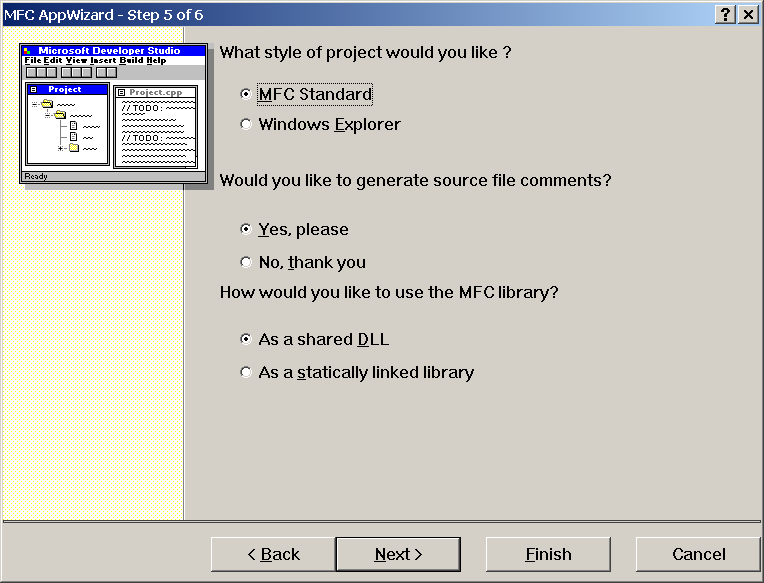
8．单击OK按钮，将再次显示Step2。此时已经完成student数据源中的dept表与应用程序之间的关联处理。单击OK，进入Step 3 of 6。



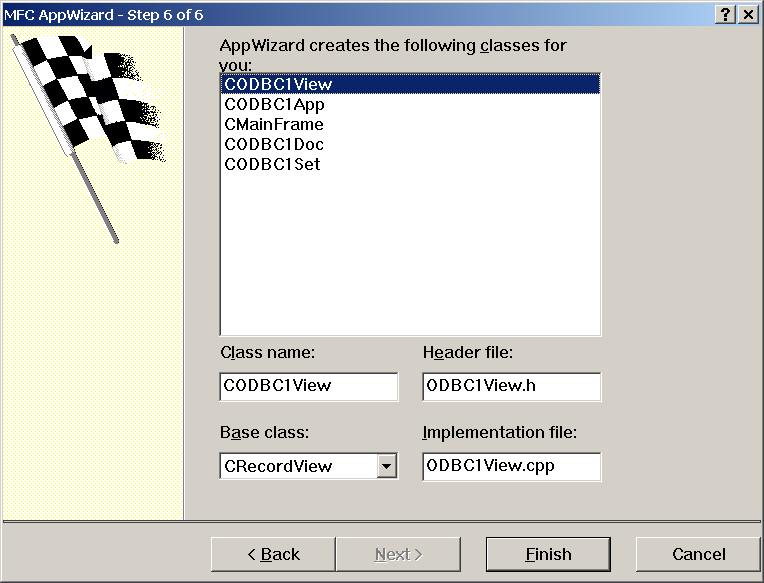
9．使用缺省选项（不做任何选择），单击Next按钮，进入Step 4 of 6。



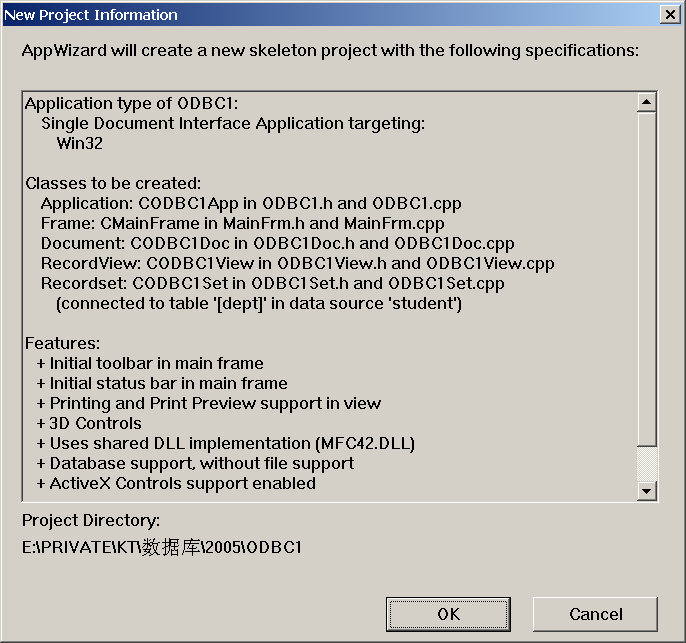
10．清除Printing and print preview复选框，单击Next按钮，进入Step 5 of 6。



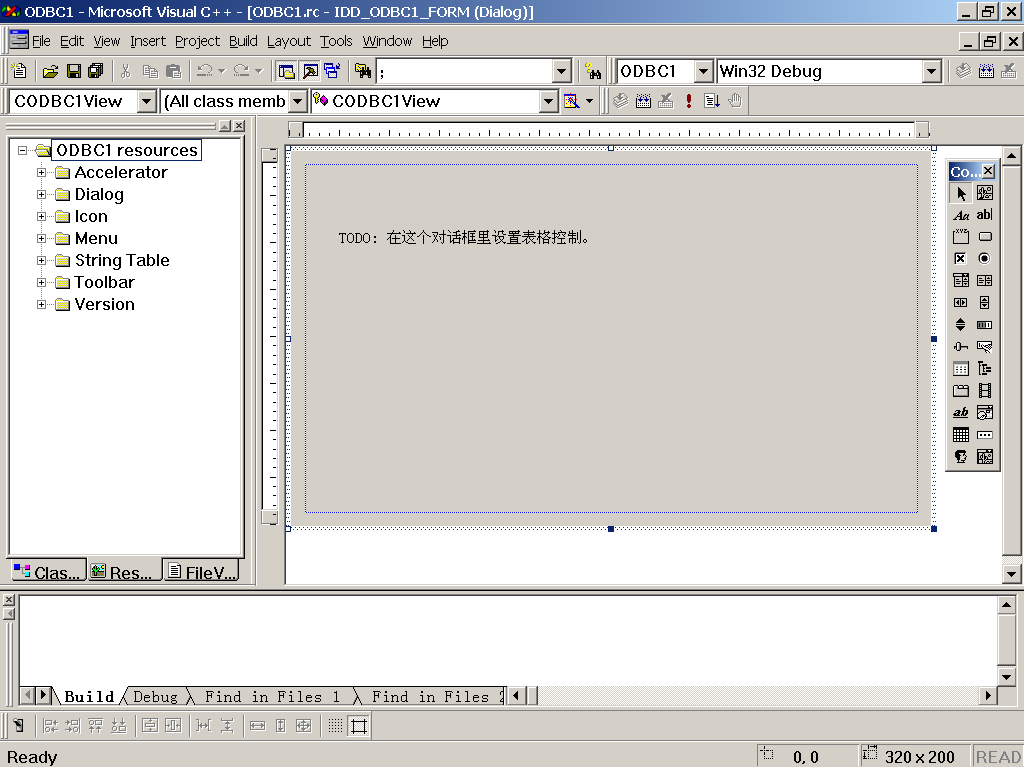
11．接受缺省选择项，单击Next按钮，进入Step 6 of 6。



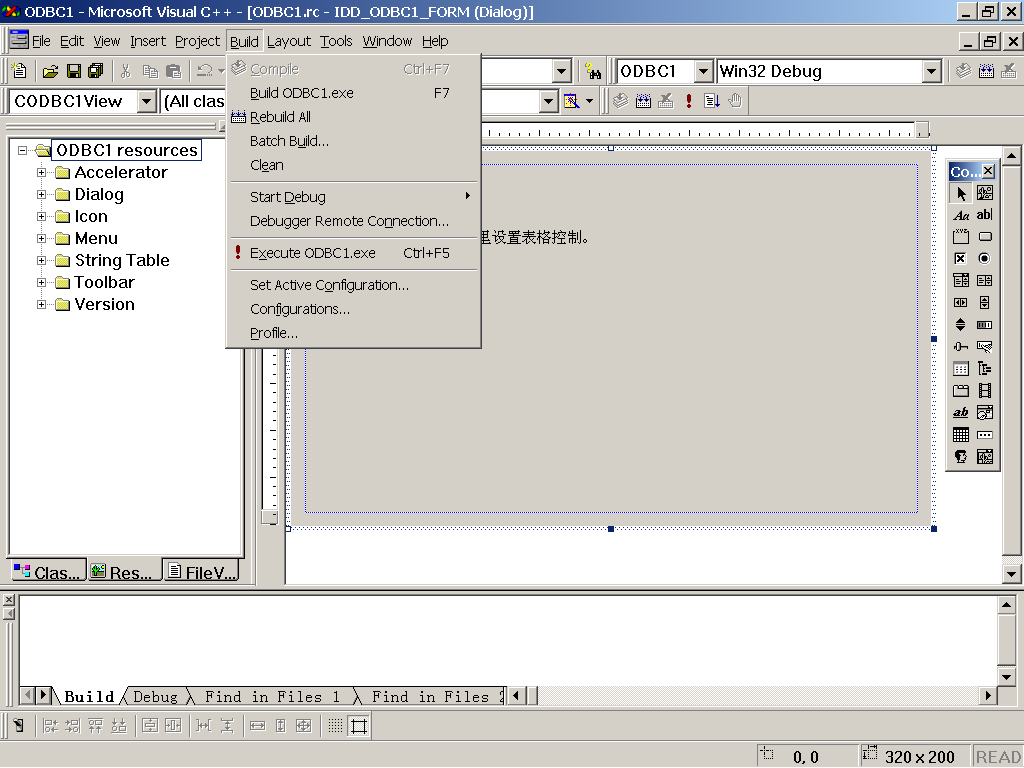
12．单击Finish按钮，显示New Project Information对话框。



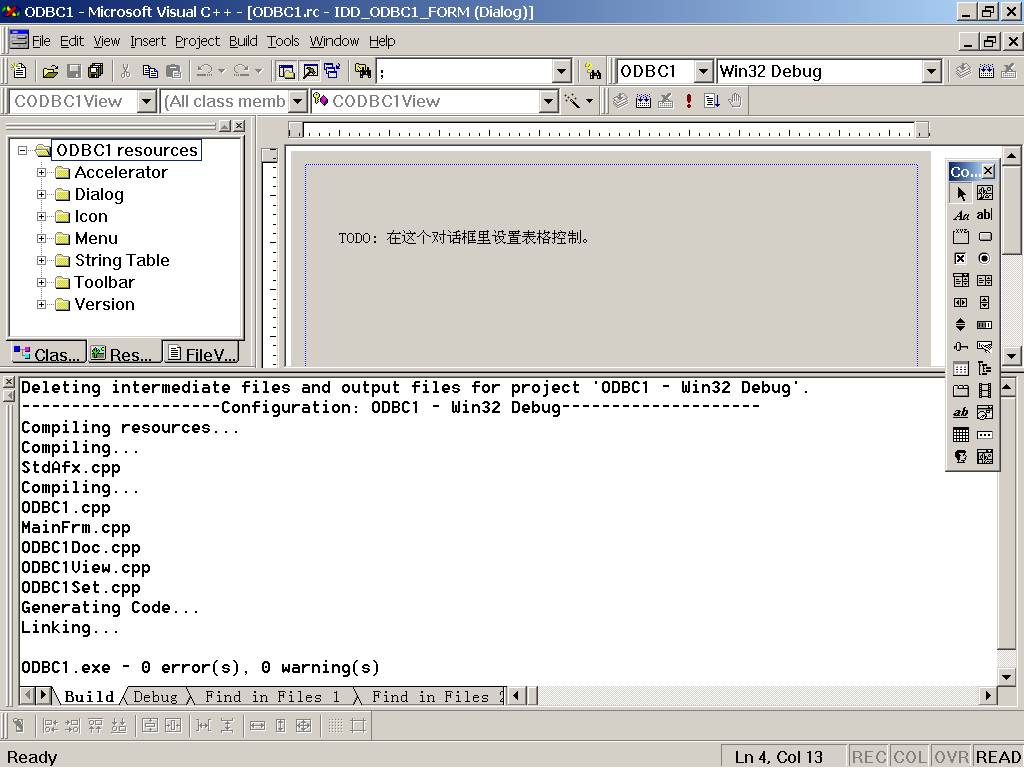
13．单击OK按钮。AppWizard开始创建基本的ODBC1应用程序。实际上只是生成了一个框架而已。



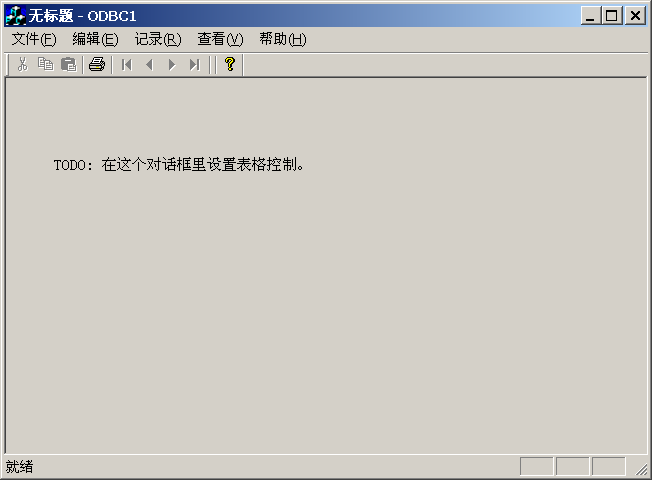
14．编译、连接（选择Build菜单中的Rebuild All全部重建）、运行（选择Build菜单中的Execute执行 ODBC1.exe）。



编译器逐个编译项目中的C++源程序:



运行结果是：

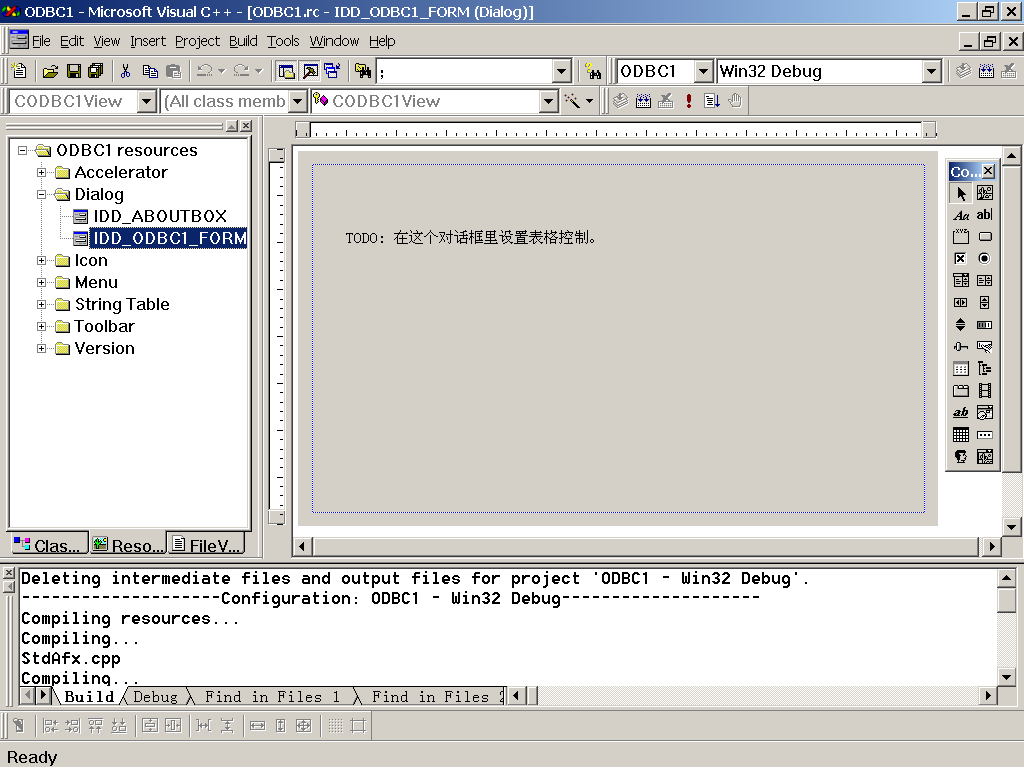


此应用程序的工具栏中有四个关于Dept表中记录的移动按钮，分别是：**首记录、前一个记录、下一个记录、尾记录**。只是此窗口中没有实际显示内容，按钮都是灰的，不能使用。因为我们还没有在视窗类中加入控制项。

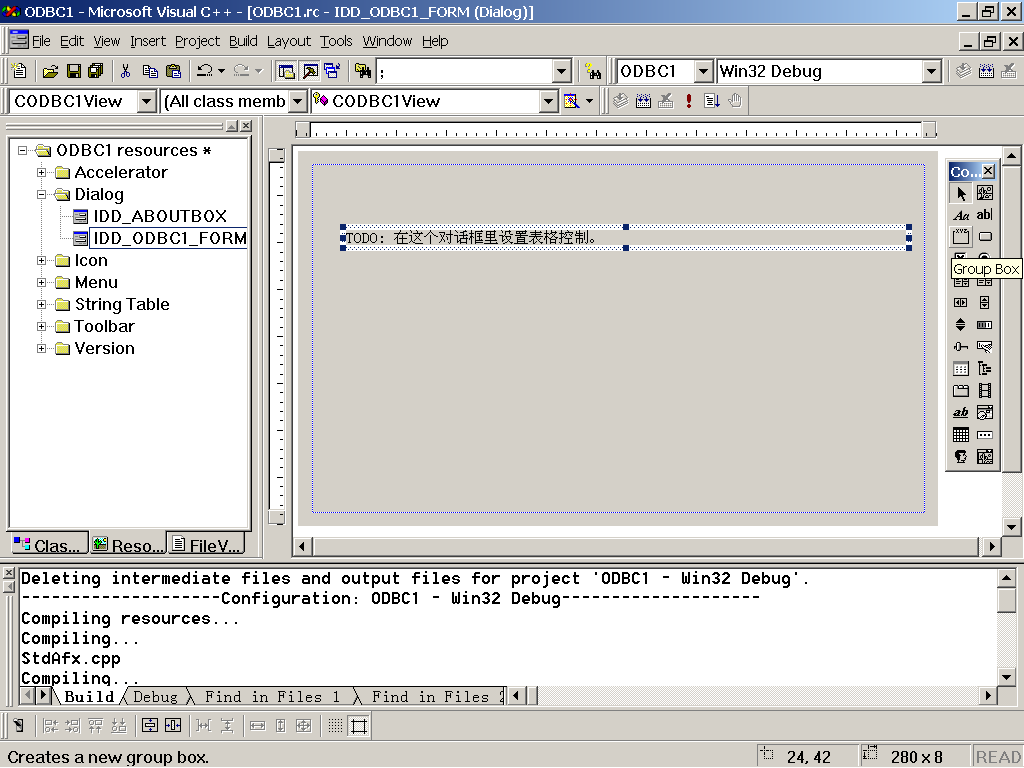
**第二步 显示数据库表中数据**

要在应用程序中直接操作数据库，则必须在视窗口中设置一些区域以显示表中各个字段。操作步骤是：

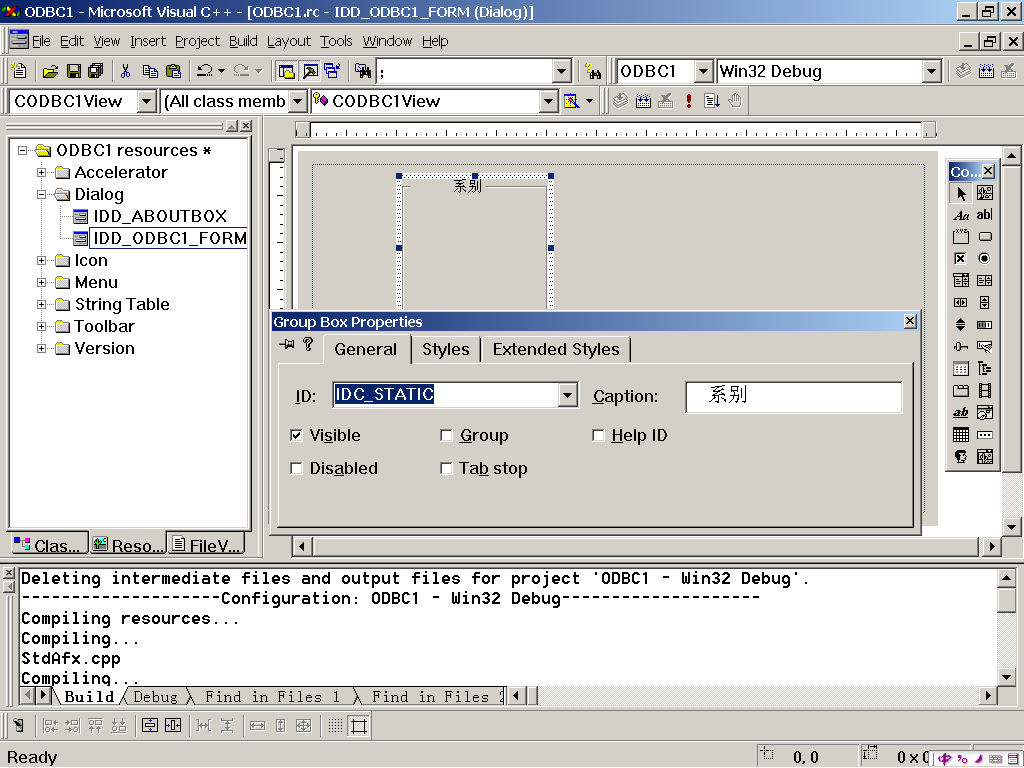
1．在项目工作区中选择ResourceView面板，显示应用程序的所有资源。

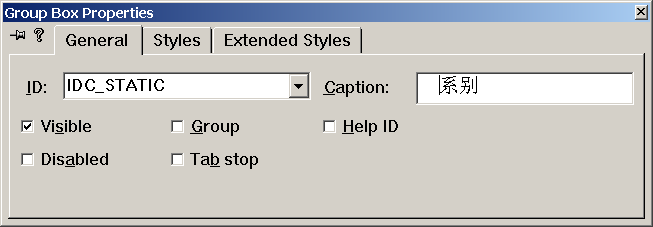


2．单击ODBC1 Resources前的“+”号，展开资源树。再展开Dialog资源文件夹，双击IDD\_ODBC1\_FORM的对话框，将在资源编辑器中打开此对话框。

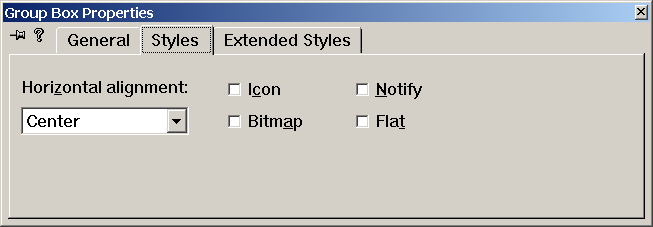


1. 单击对话框中的静态文本，按Delete键或删除此静态文本。
2. 利用右侧的Controls工具栏，增加一个Group Box静态控件，大小自己选定。选中此控件，按鼠标右键弹出下拉菜单，选中Properties属性，使得使标题Caption为“系别”。



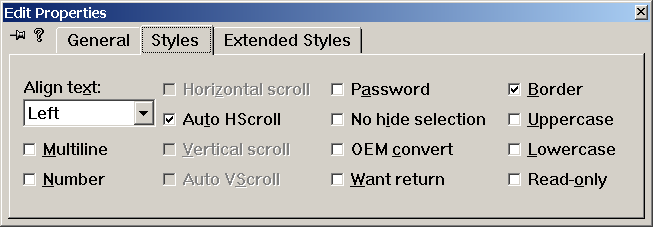


选Styles的对齐方式为：Center居中。

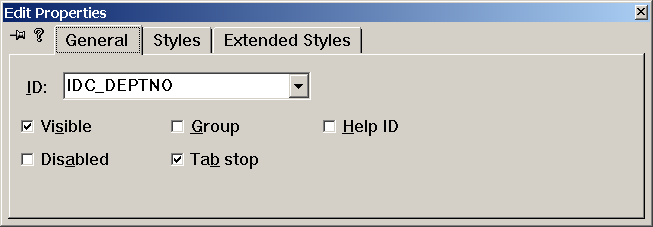


再拖出3个**静态文本控件**（Aa控件），用上面同样的方法，修改属性，使标题Caption分别为：系号、系名、系主任。

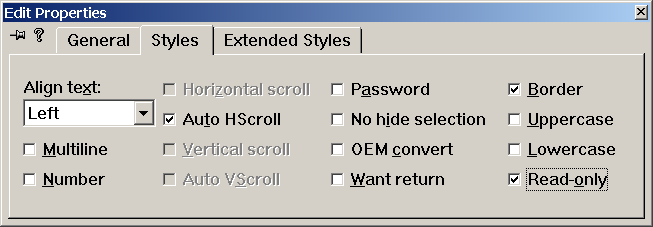
再拖出3个**文本编辑控件**（ab控件），用上面同样的方法，修改属性，弹出如下对话框：



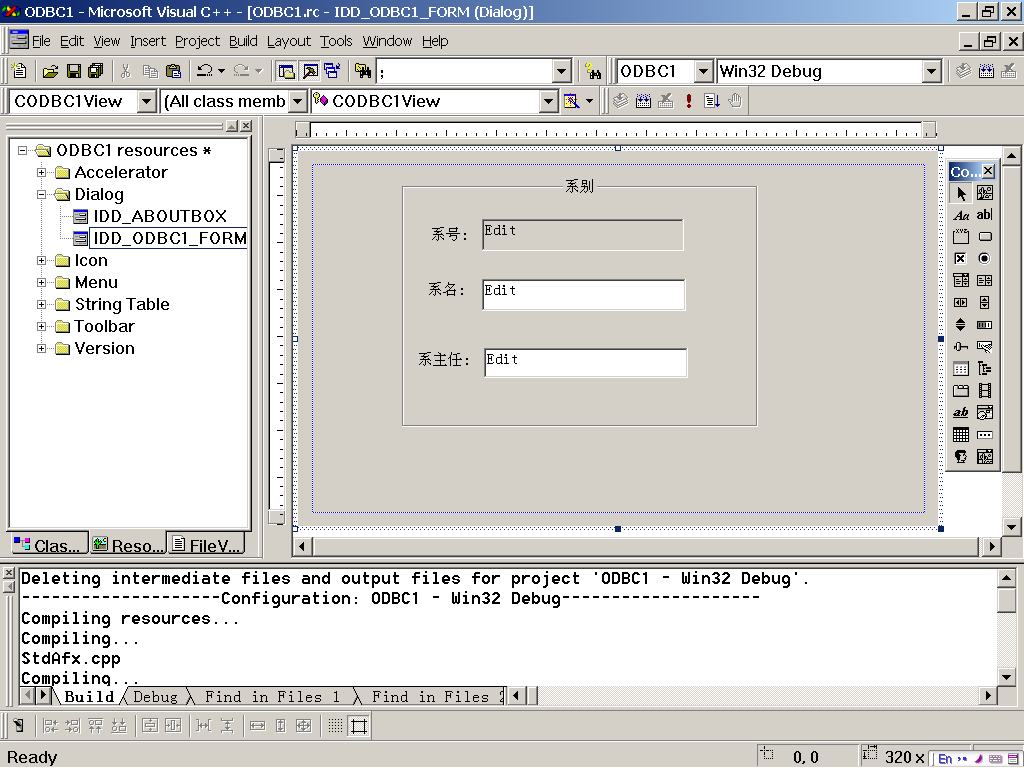
三个文本框的ID分别为：IDC\_DEPTNO、IDC\_DNAME、IDC\_HEAD。



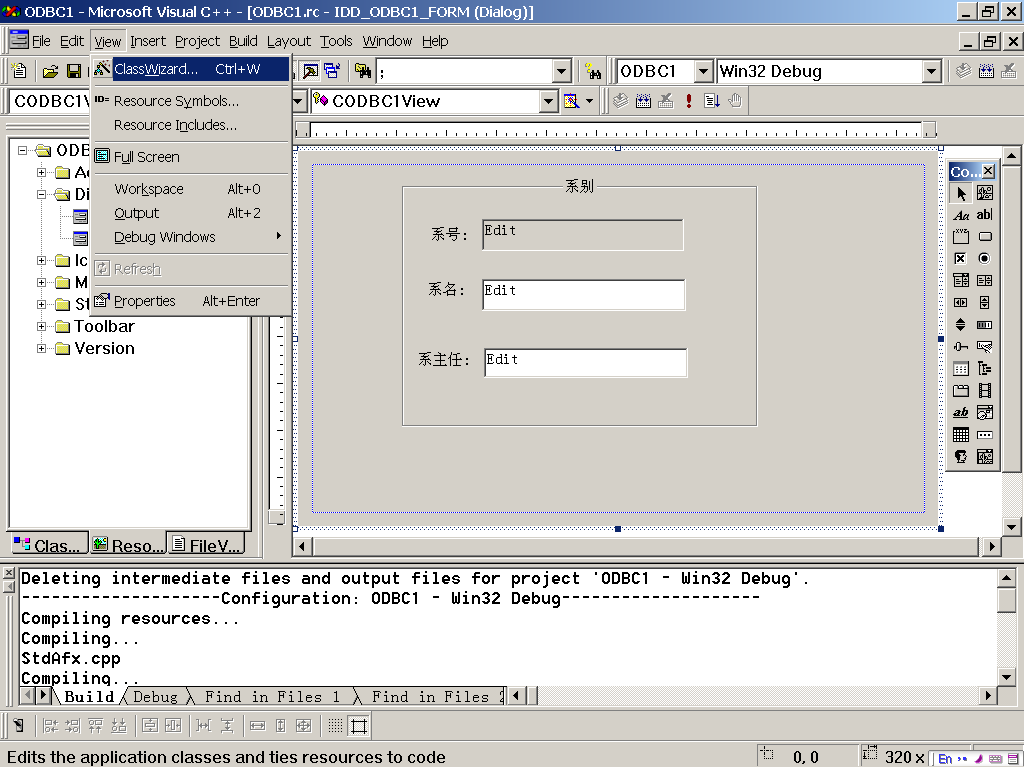
将IDC\_DEPTNO改为只读，Read-Only前打勾选中。这样保证主码DEPTNO不被修改。



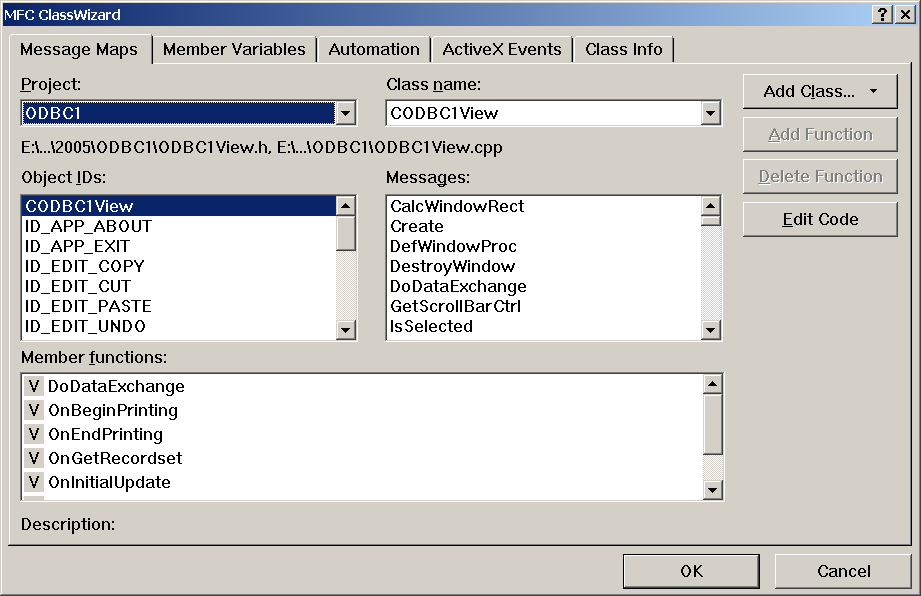
效果图为：



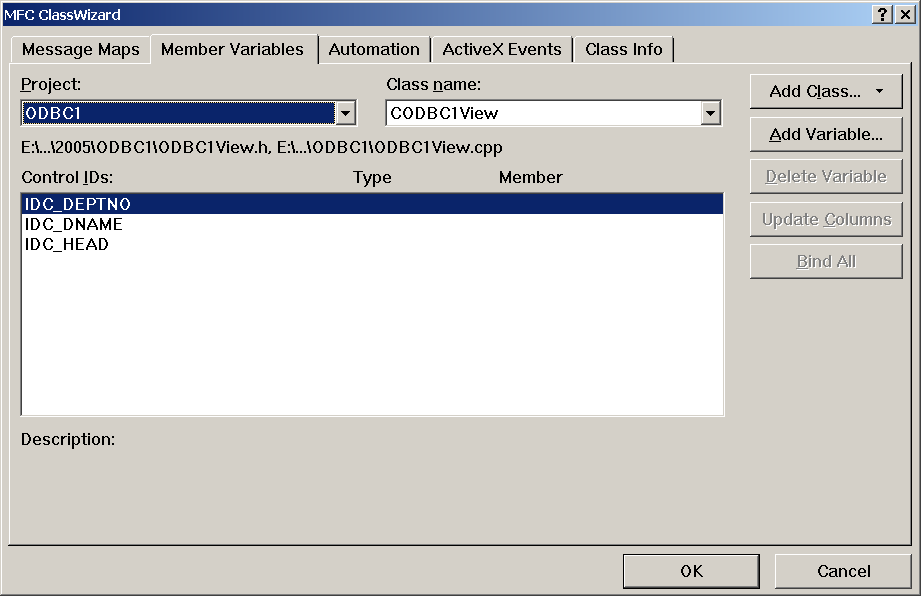
1. 利用View中的ClassWizard选项，启动ClassWizard工具：



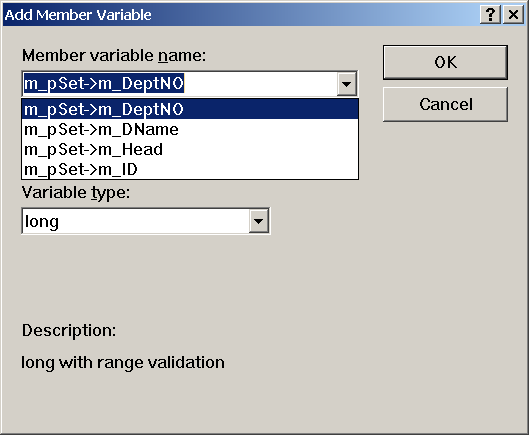
会弹出窗口：



选择Member Variables标签：

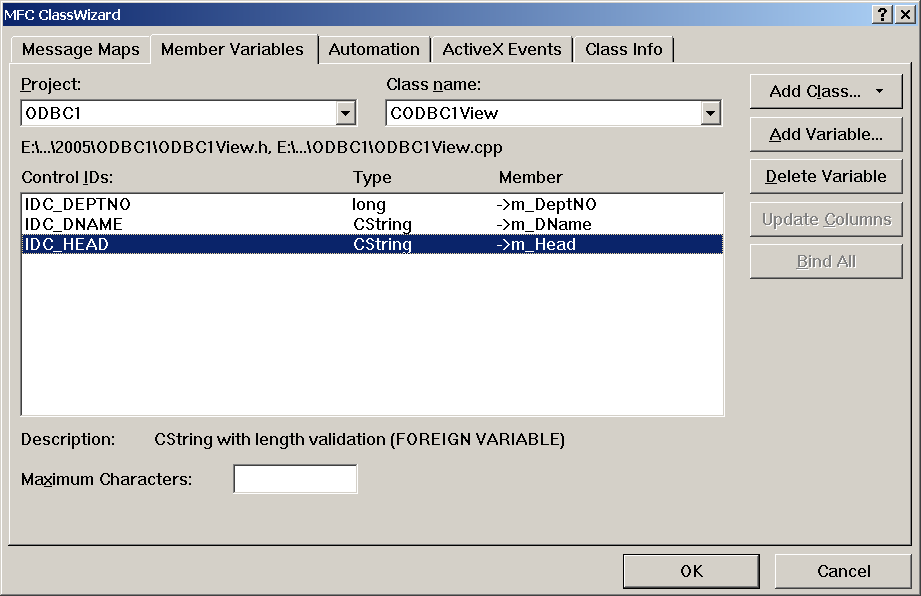


6．在Class name文本框中选择CODBC1View，在Control IDs列表框中选择IDC\_DEPTNO。单击Add Variable按钮，会显示出Add Member Variable对话框：

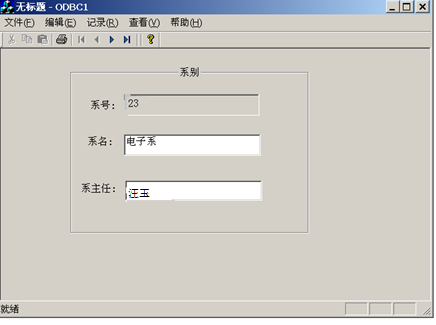


在Member variable name 下拉列表框中，选择 m\_pSet->DeptNO选项。使得m\_pSet->DeptNO与IDC\_DEPTNO控件相连。

7．同样的方法，将m\_pSet->Dname和IDC\_DNAME控件相连，将m\_pSet->Head和IDC\_HEAD控件相连。此时MFC ClassWizard属性单中的成员变量表为：



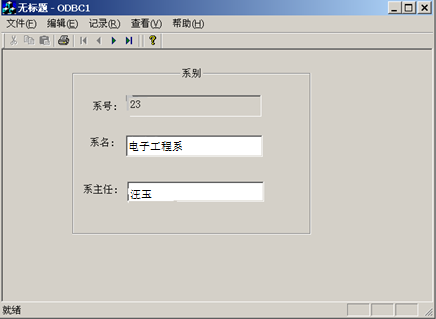
8．在ClassWizard对话框中单击OK按钮，完成以上修改。此时应用程序中增加了显示student数据库中dept表中数据的功能。ReBuild All 编译链接，运行结果是：



该应用程序的4个按钮已经可以使用：



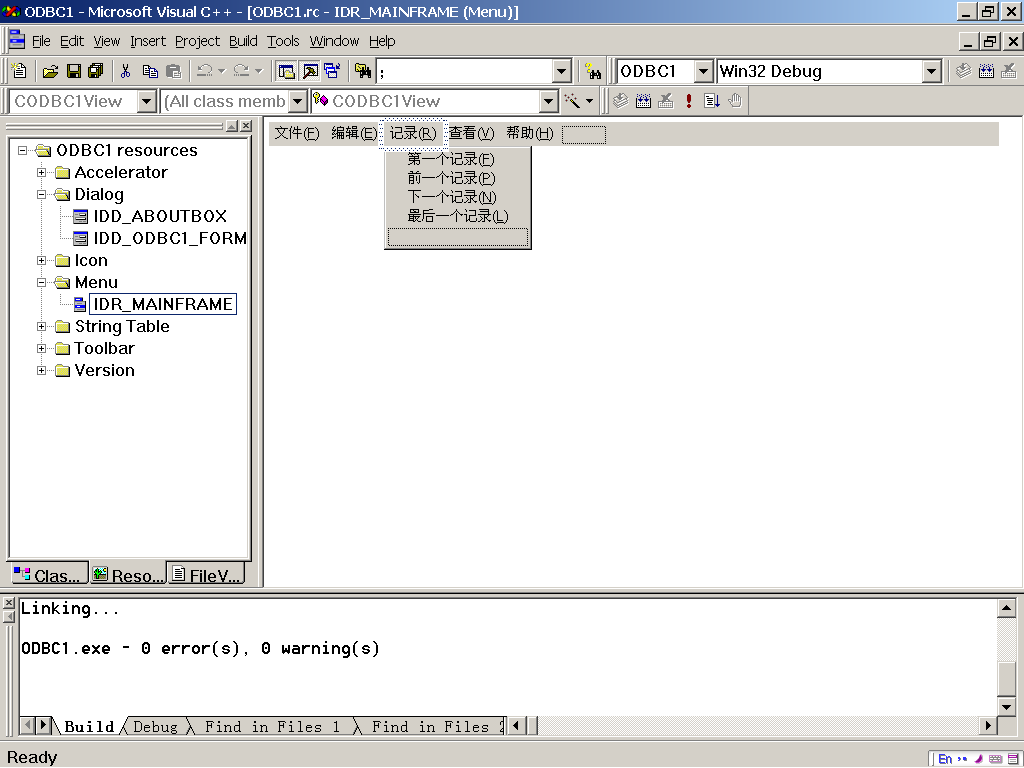
可以任意修改除了**系号**之外的任何属性的值，例如，可以把23号系的系名由“电子系”改为“电子工程系”：



**第三步 实现数据库的增加、删除、更新和清空记录功能**

借助CRecordView和CRecordset类实现。步骤是：

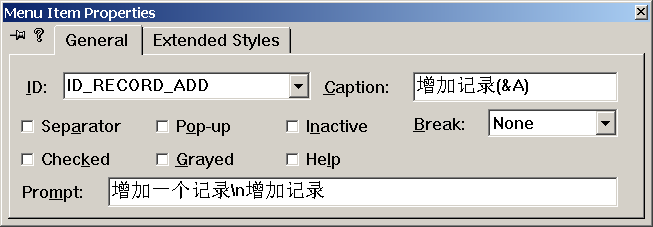
1．在项目工作区中选择ResourceView面板，展开Menu资源，双击ID为IDR\_MANFRAME的资源，显示如下：



菜单编辑器

2．在菜单编辑器中单击“记录”，打开该菜单。单击菜单底部的空白菜单项，选择View下拉菜单中的Properties属性选项。

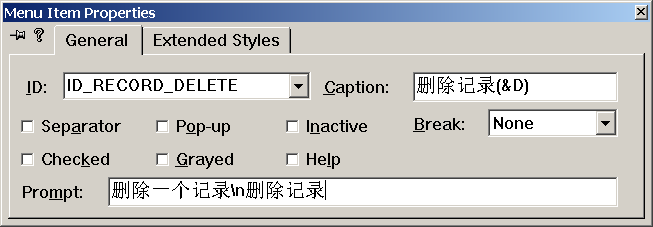
3．ID中输入：“ID\_RECORD\_ADD”, Caption中输入：“增加记录(&A)”，Prompt中输入：“增加一个记录\n增加记录”。



4．同样的方法，增加其它三个菜单命令：

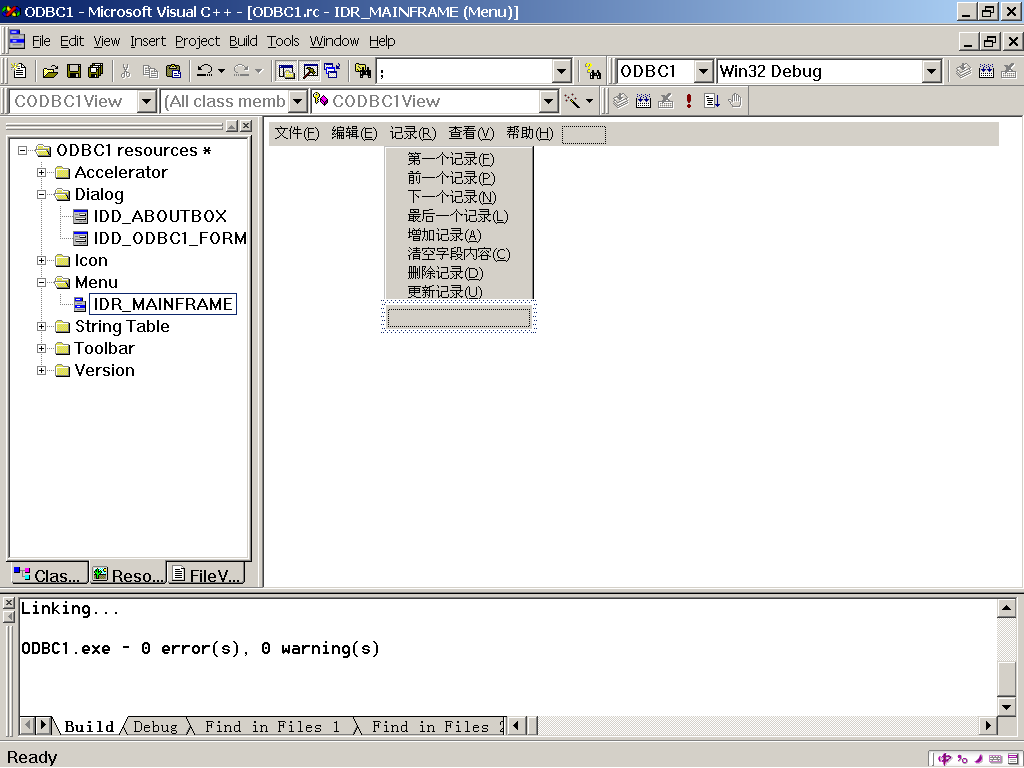
|  |  |  |  |
| --- | --- | --- | --- |
| 菜单命令 | ID | Caption | Prompt |
| 清空字段内容 | ID\_RECORD\_CLEARFIELDS | 清空字段内容(&C) | 清空字段内容\n清空字段 |
| 删除记录 | ID\_RECORD\_DELETE | 删除记录(&D) | 删除一个记录\n删除记录 |
| 更新记录 | ID\_RECORD\_UPDATE | 更新记录(&U) | 更新记录\n更新记录 |







此时“记录”菜单变为：

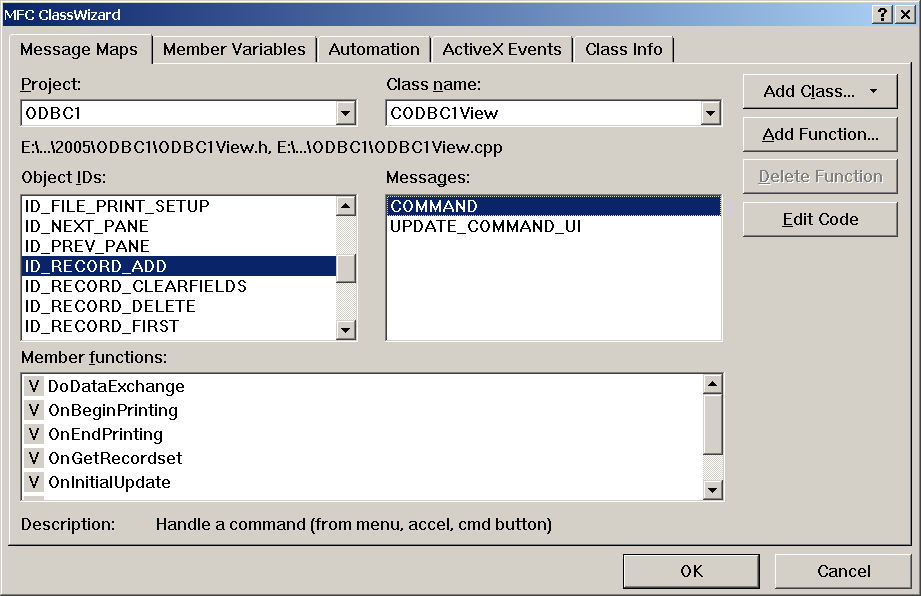


**第四步 编写命令消息的代码**

至此，已经在“记录”菜单中加入了四个命令，然后可编写获取这些命令消息的代码。由于与数据库相连的是视窗口，所以应由视窗口处理这些消息。步骤如下：

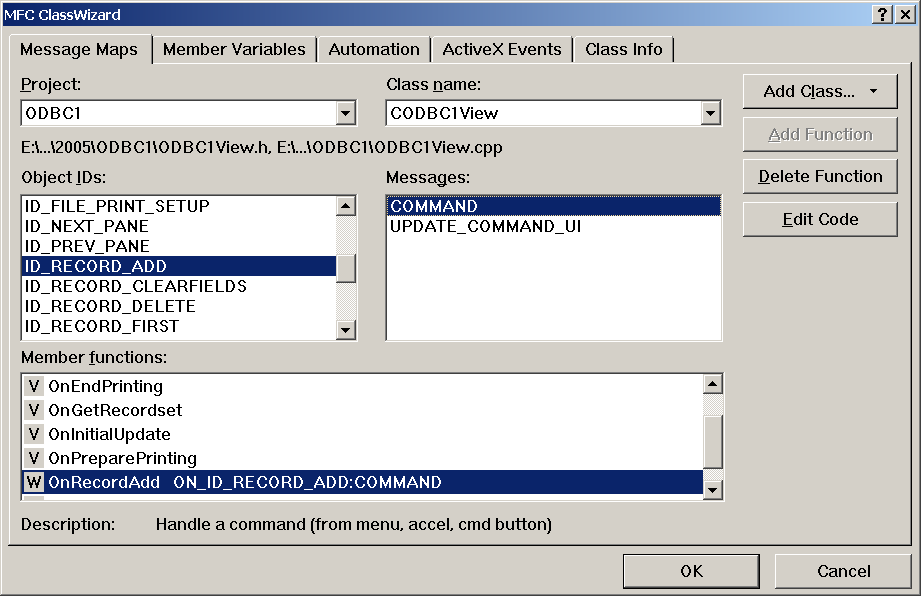
1. 启动ClassWizard，然后选择Message Maps标签，显示Message Maps属性页。
2. 将Class Name下拉列表框设置为CODBC1View，在Object IDs列表框中单击ID\_RECORD\_ADD，然后在Messages列表框中双击COMMAND，会显示出

Add Member Function对话框。

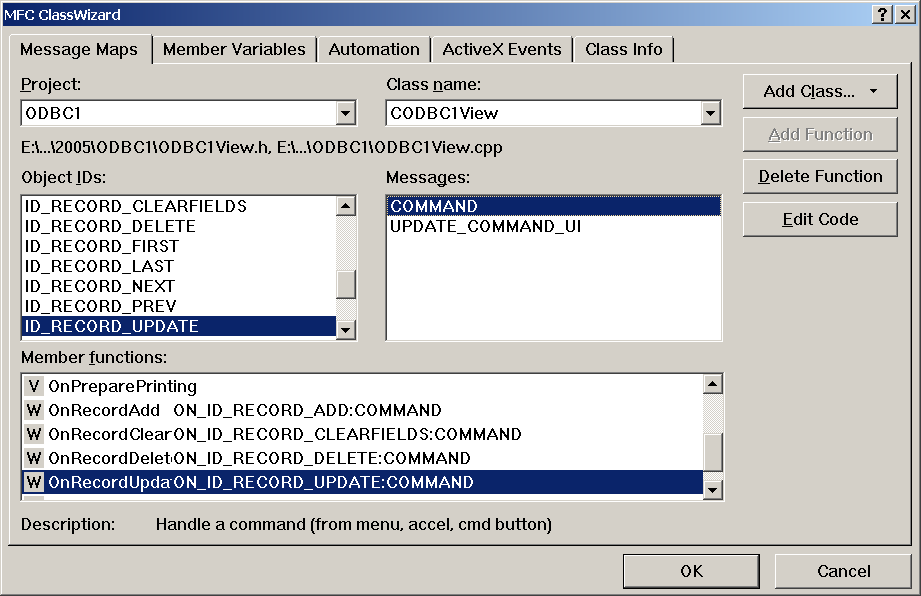




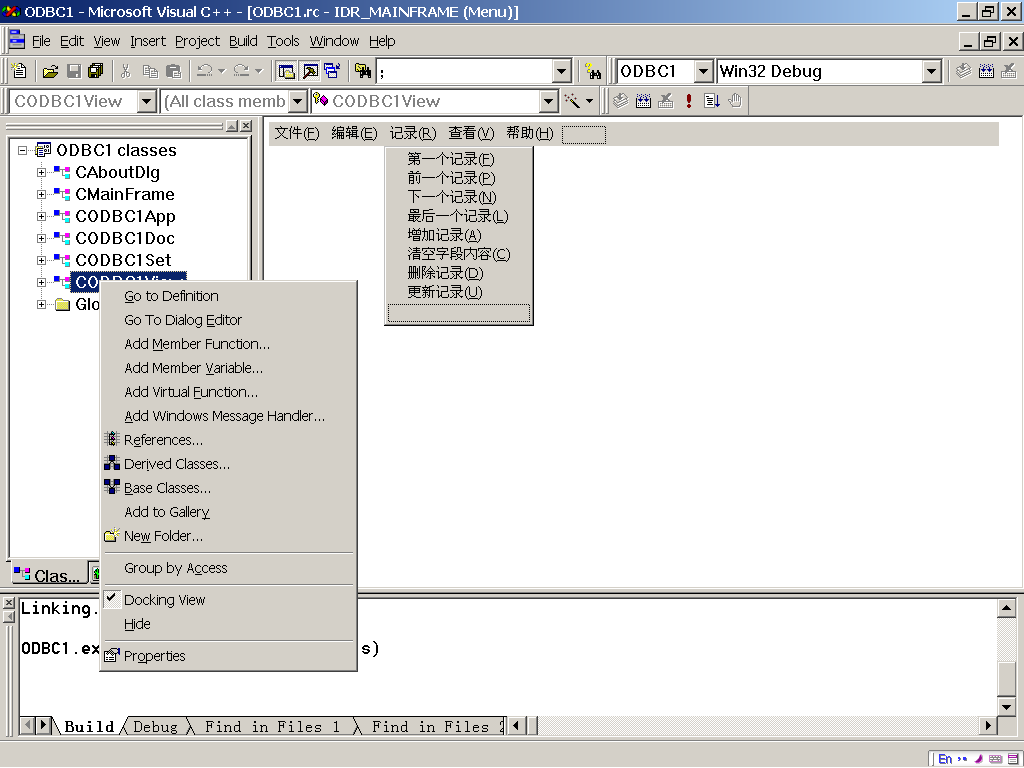
1. 单击OK按钮，接受命令处理函数的缺省名称，并关闭对话框。函数名称显示在ClassWizard对话框底部的Member functions列表框中：



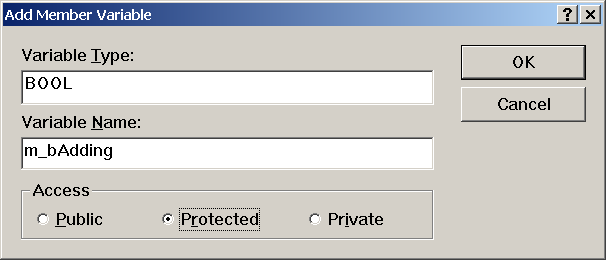
1. 同样的方法，处理ID\_RECORD\_CLEARFIELDS、ID\_RECORD\_DELETE、ID\_RECORD\_UPDATE命令。



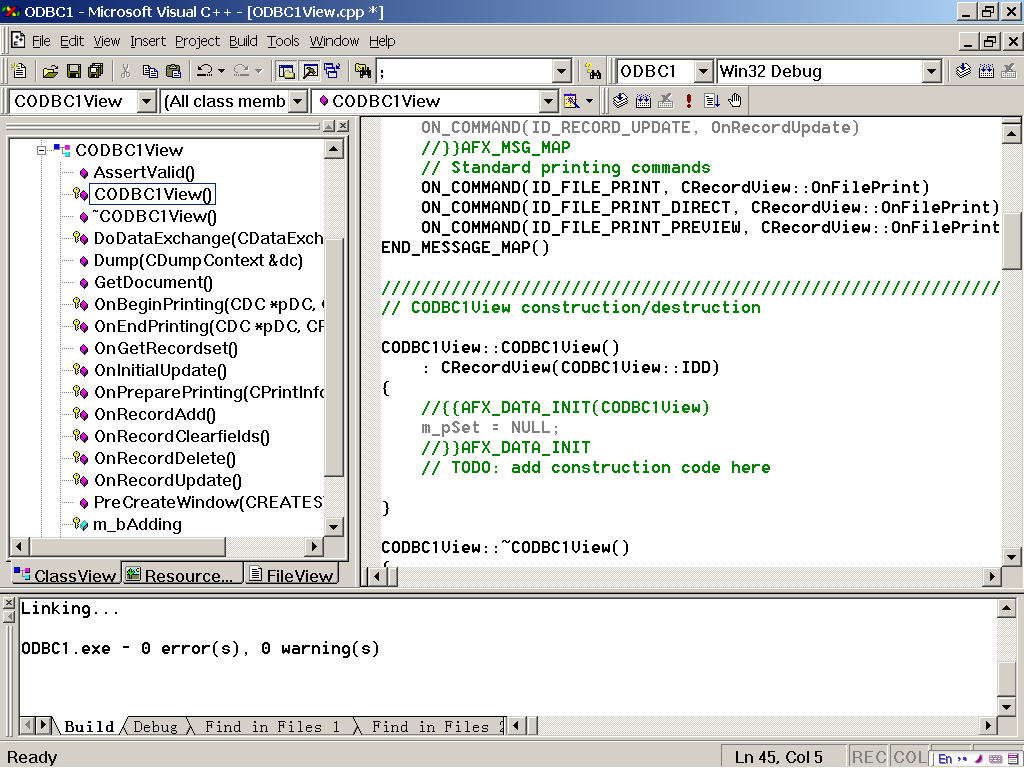
1. 单击OK按钮，关闭ClassWizard对话框。
2. 在CODBC1View类中增加一个保护成员变量m\_bAdding。做法是：打开项目工作区的ClassView面板中“CODBC1View”类名，按鼠标右键，在弹出的菜单中选择增加成员变量Add Member Variable命令，打开Add Member Variable对话框。



7．在变量类型文本框中输入“BOOL”，在变量名文本框中输入“m\_bAdding”，在存取方式中选择“Protected”方式，按下OK按钮确认。

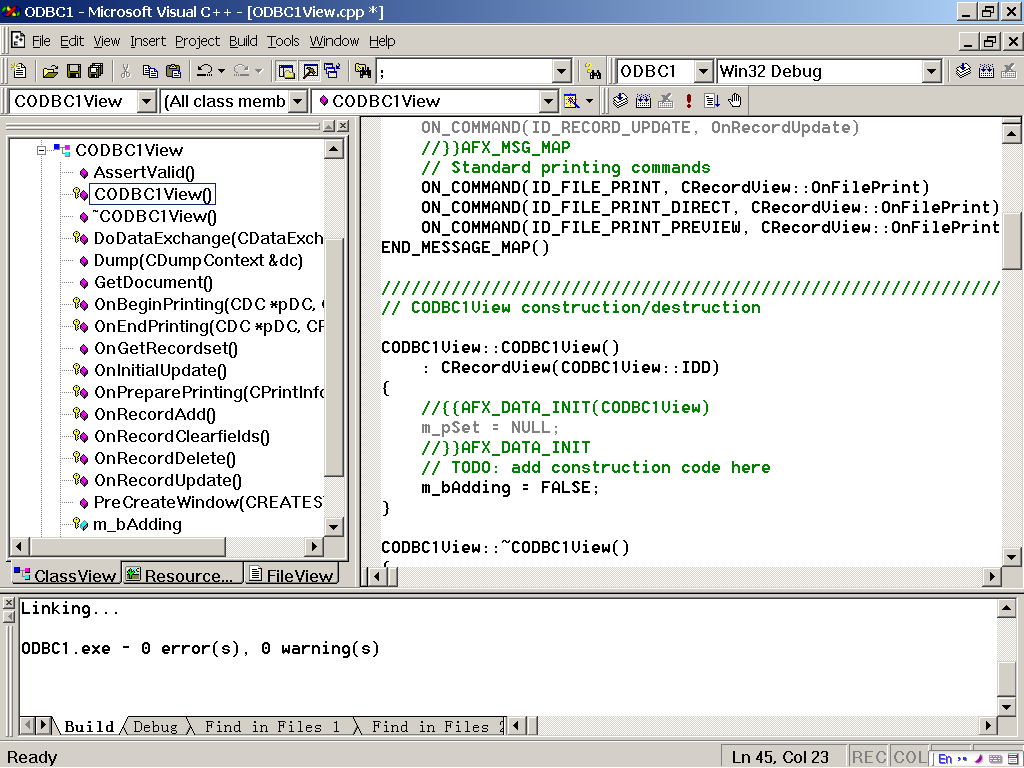


8．然后在CODBC1View构造函数中初始化变量m\_bAdding。方法是：在项目工作区的ClassView面板中展开类CODBC1View，双击构造函数CODBC1View( ), 见下图：



在函数的最后加上以下代码：

m\_bAdding = FALSE;



9．再编辑OnRecordAdd( )函数。为此双击函数OnRecordAdd( )，然后在主工作区中编辑。

编辑前函数为：

void CODBC1View::OnRecordAdd()

{

// TODO: Add your command handler code here

}

编辑后函数为：

void CODBC1View::OnRecordAdd()

{

// TODO: Add your command handler code here

m\_pSet->AddNew();

m\_bAdding =TRUE;

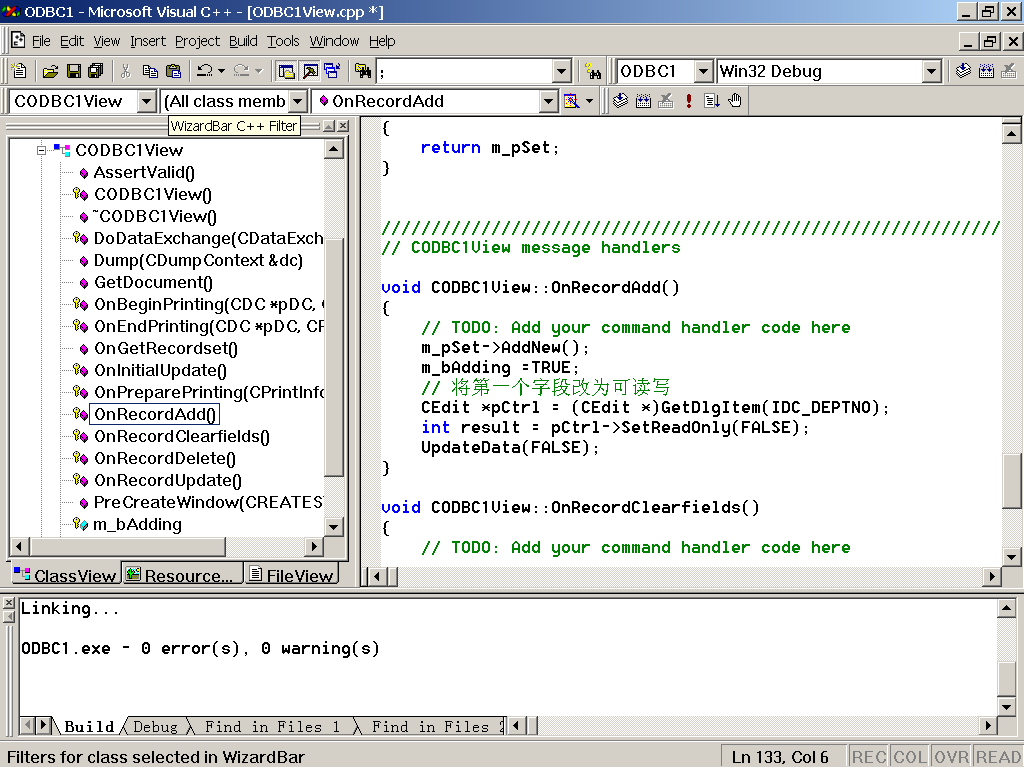
// 将第一个字段改为可读写

CEdit \*pCtrl = (CEdit \*)GetDlgItem(IDC\_DEPTNO);

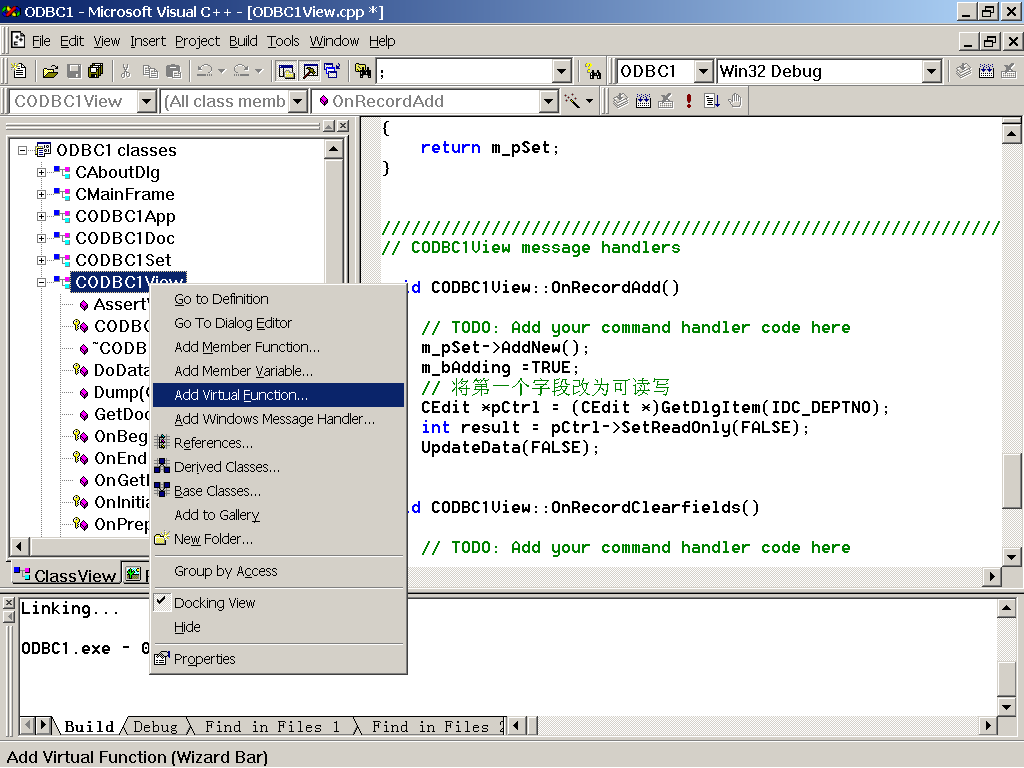
int result = pCtrl->SetReadOnly(FALSE);

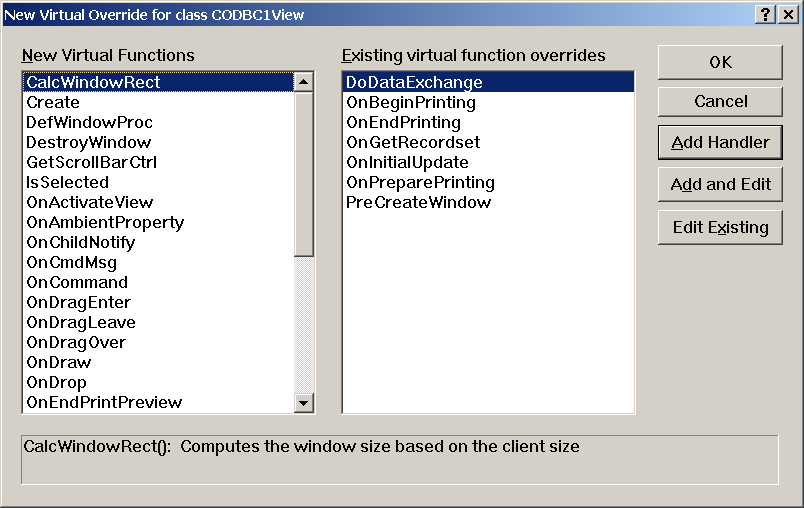
UpdateData(FALSE);

}

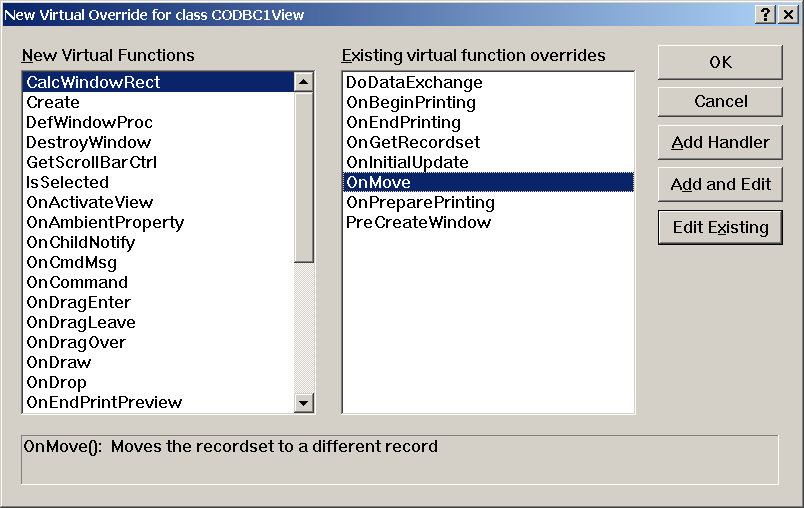


10．最后重载**虚函数OnMove**，（程序运行中，当用户移动一个数据库记录时，将调用虚函数CRecordset :: OnMove更新数据库，重载该函数主要是为了**保存新增加的记录**）。重载方法是：在项目工作区中选择ClassView面板，在CODBC1View类名上，按鼠标右键，在弹出的菜单中选择Add Virtual Function命令，会打开New Virtual Override对话框。

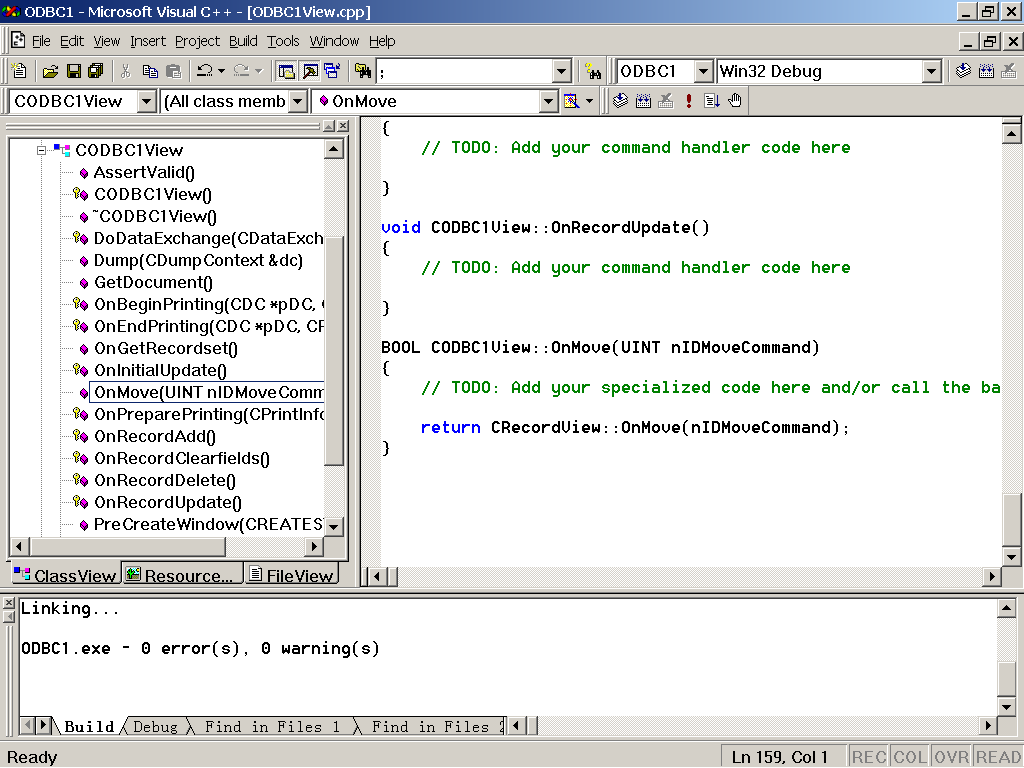




11． 向下滚动左边的New Virtual Functions列表框，选择其中的OnMove函数。单击按钮Add Handler，将该函数增加到右边的Existing virtual function overrides列表中。



12． 然后在项目工作区ClassView面板中展开“CODBC1View”类，双击其中的OnMove函数：



编辑输入下面的函数源代码：

BOOL CODBC1View::OnMove(UINT nIDMoveCommand)

{

// TODO: Add your specialized code here and/or call the base class

if (m\_bAdding) {

m\_bAdding = FALSE;

UpdateData(TRUE);

// 调用 Update

if (m\_pSet->CanUpdate())

m\_pSet->Update();

//将增加的记录移动到最后

if (!m\_pSet->IsEOF())

m\_pSet->MoveLast();

m\_pSet->Requery();

UpdateData(FALSE);

// 将第一个字段恢复为只读

CEdit \*pCtrl = (CEdit \*)GetDlgItem(IDC\_DEPTNO);

pCtrl->SetReadOnly(TRUE);

return TRUE;

}

else {

switch (nIDMoveCommand)

{

case ID\_RECORD\_PREV:

m\_pSet->MovePrev();

if (!m\_pSet->IsEOF())

break;

case ID\_RECORD\_FIRST:

m\_pSet->MoveFirst();

break;

case ID\_RECORD\_NEXT:

m\_pSet->MoveNext();

if (!m\_pSet->IsEOF())

break;

if (!m\_pSet->CanScroll()){

m\_pSet->SetFieldNull(NULL);

break;

}

case ID\_RECORD\_LAST:

m\_pSet->MoveLast();

break;

default:

ASSERT(FALSE);

}

UpdateData(FALSE); // 显示移动操作的结果

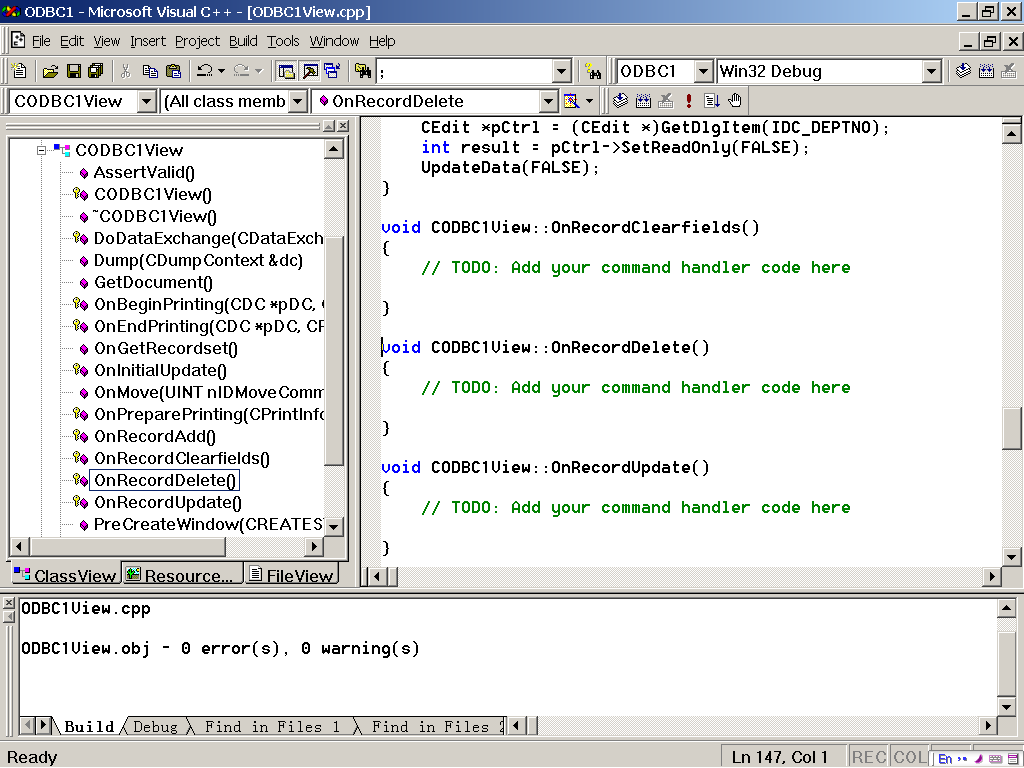
return TRUE;

}

return CRecordView::OnMove(nIDMoveCommand);

}

13．在项目工作区的ClassView面板中，展开“CODBC1View”类，双击其中的OnRecordDelete( )函数：



编辑输入下面的OnRecordDelete函数源代码：

void CODBC1View::OnRecordDelete()

{

// TODO: Add your command handler code here

try {

m\_pSet->Delete();

}

catch (CDBException \*e) {

AfxMessageBox(e->m\_strError);

e->Delete();

m\_pSet->MoveFirst();

UpdateData(FALSE);

return;

}

if (m\_pSet->IsEOF())

m\_pSet->MoveLast();

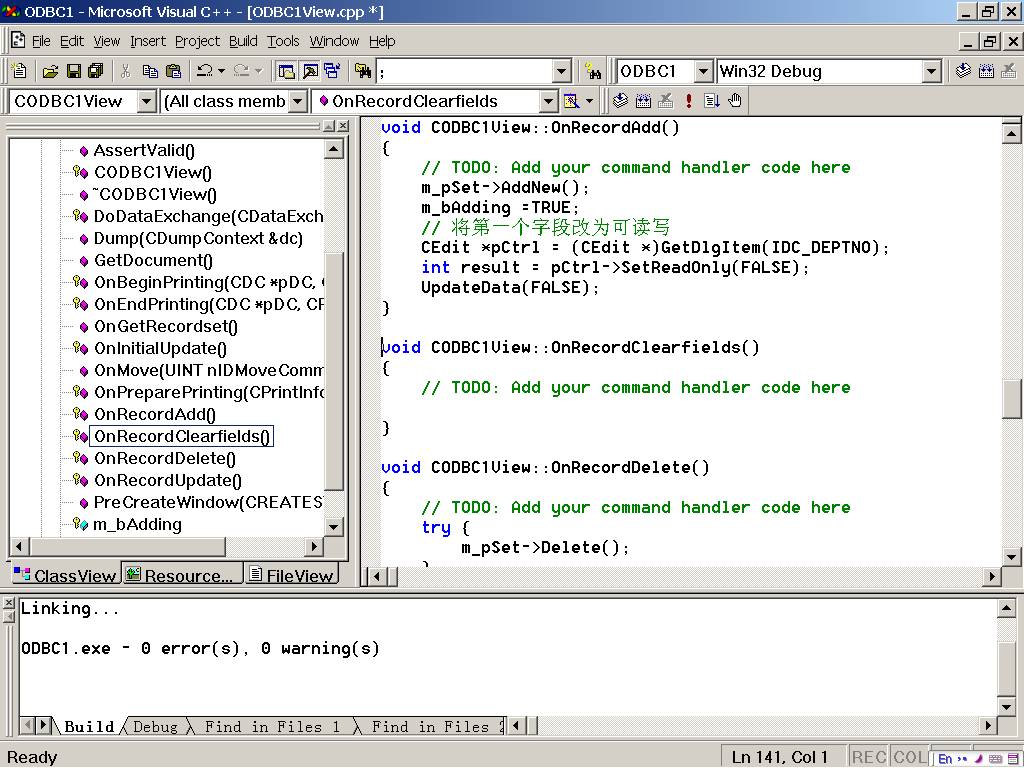
if (m\_pSet->IsBOF())

m\_pSet->SetFieldNull(NULL);

UpdateData(FALSE);

}

14．再在项目工作区的ClassView面板中，展开“CODBC1View”类，双击其中的OnRecordClearfields( )函数：



编辑输入下面的OnRecordClearfields( )函数源代码：

void CODBC1View::OnRecordClearfields()

{

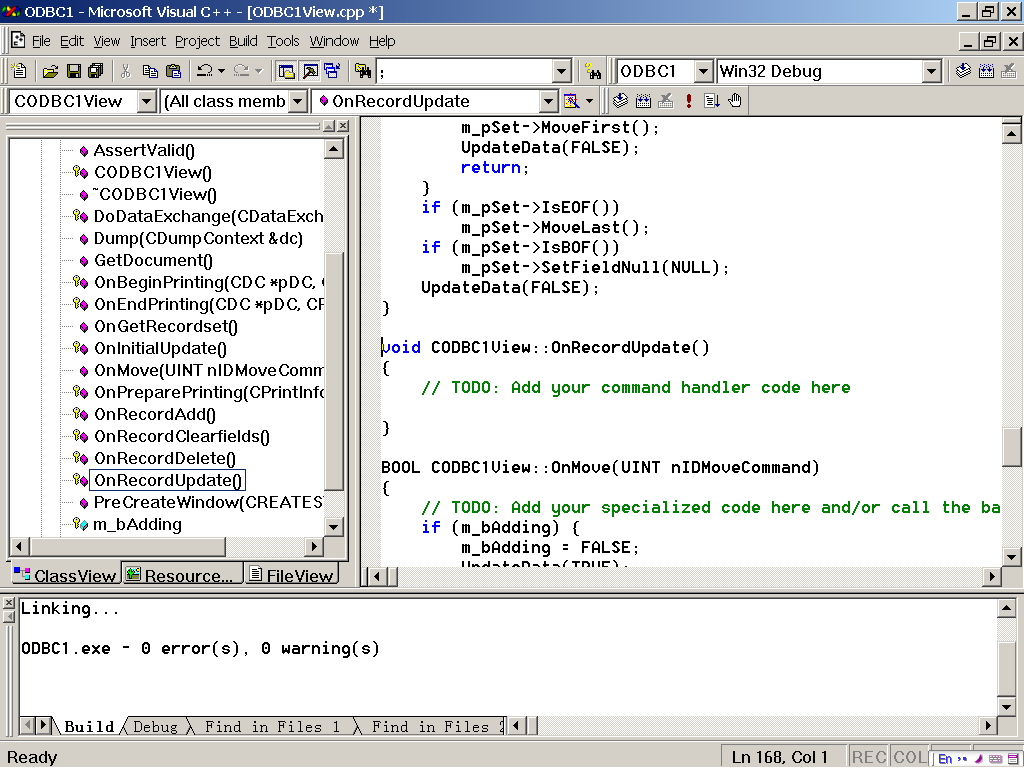
// TODO: Add your command handler code here

m\_pSet->SetFieldNull(NULL);

UpdateData(FALSE);

}

15.再在项目工作区的ClassView面板中，展开“CODBC1View”类，双击其中的OnRecordUpdate ( )函数：



编辑输入下面的OnRecordUpdate ( )函数源代码：

void CODBC1View::OnRecordUpdate()

{

// TODO: Add your command handler code here

m\_pSet->Edit();

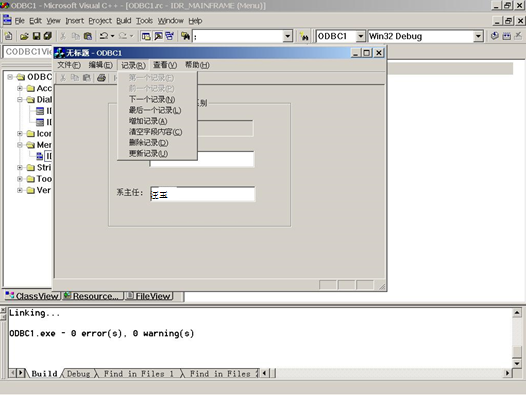
UpdateData(TRUE);

if (m\_pSet->CanUpdate() )

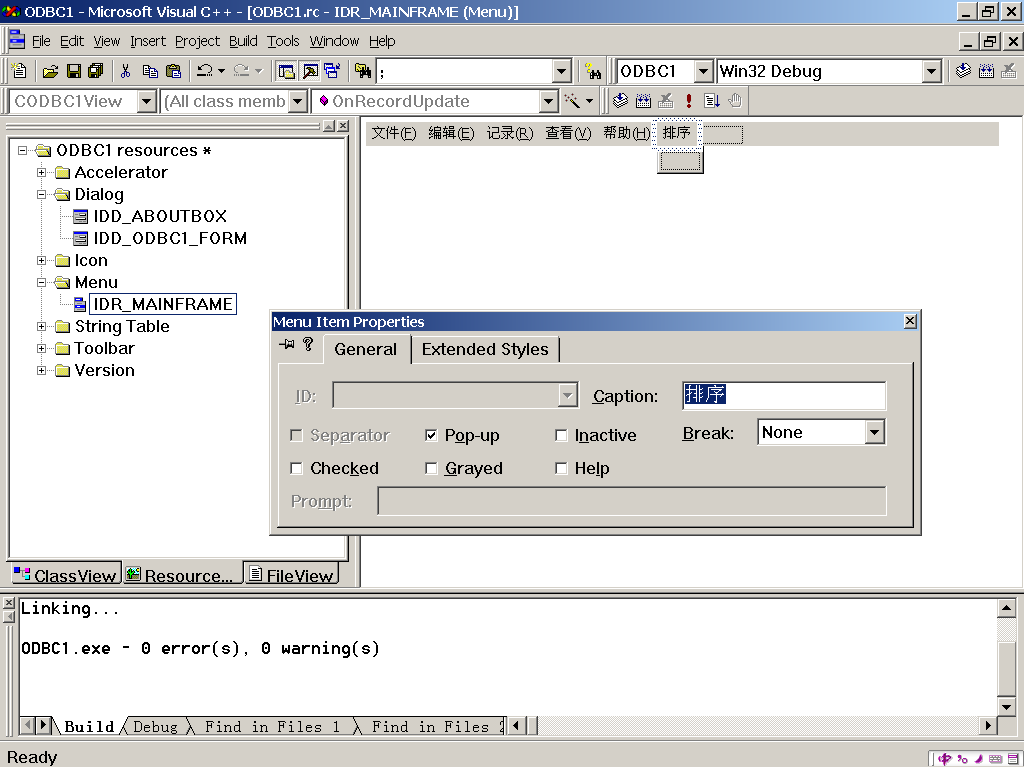
m\_pSet->Update();

}

至此，在应用程序中实现了**增加记录、删除记录、清空字段内容**和**更新记录**等功能，可以用 “Rebuild All ”重新编译链接、运行应用程序Odbc1，并可以自己进行各种测试，验证程序的正确性和效果。



除此之外，还可以在菜单栏中增加一个新菜单，例如，增加“排序”菜单，方法是在ResourceView中，展开Menu，双击 “IDR\_MAINFRAME”, 在菜单栏的最右边空白虚线方块内双击鼠标，就可以输入新菜单名“排序”：



再按照前面在菜单中增加菜单选项的步骤，例如，增加“系号”，“系名”，“系主任”及其处理函数。这个不作要求，有兴趣的同学可以自己尝试。