# 数据库相关操作

数据库的操作需要用到ADO库，它包括三个基本接口：\_ConnectionPtr接口、\_CommandPtr接口、\_RecordsetPtr接口。

\_ConnectionPtr接口返回一个记录集或一个空指针。通常使用它来创建一个数据链接来执行一条不返回任何结果的SQL语句（如删除操作），使用\_ConnectionPtr来返回一个记录集不是一个好的使用方法，对于要返回记录的操作通常用\_RecordsetPtr来实现。

\_CommandPtr接口返回一个记录集。它提供了一种简单的应运来执行返回记录集的存储过程和SQL语句。在使用CommandPtr接口时，你可以利用全局的\_ConnectionPtr接口，也可以在\_CommandPtr接口里直接使用连接串。如果你只执行一次或几次数据访问操作，后者是比较好的选择，但如果你要频繁的访问数据库，并要返回很多的记录集，那么，你应该使用全局的\_CommandPtr接口创建一个数据连接，然后使用\_CommandPtr接口来执行存储过程和SQL语句，这也是本实例使用的方法。

\_RecordsetPtr是一个记录体对象，它对记录集提供了更多的控制功能，如记录锁定、游标控制等。同\_CommandPtr接口一样，你可以使用一个已创建的数据库连接，也可以用一个连接串代替连接指针赋给\_RecordsetPtr的connection成员变量，让它自己创建数据库连接。如果你要使用多个记录集，最好的方法同\_Command对象一样，使用已经创建了数据连接的全局\_ConnectionPtr接口。

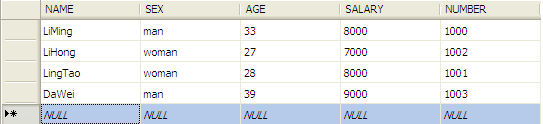
综上，可以这样操作数据库，先用\_ConnectionPtr来创建一个全局的连接，并将它作为\_RecordsetPtr和\_CommandPtr的参数。对于需要返回记录集的操作，例如查找操作，可以用\_RecordsetPte来完成，对于不需要返回记录集的操作，利用删除操作，可以使用\_ConnectonPtr和\_CommandPtr来完成。

更多操作细节可查阅《Visual C++6.0 开发指南》，这里列出操作库常用的方法，以便于为将来使用时提供参考。

## 1、准备工作

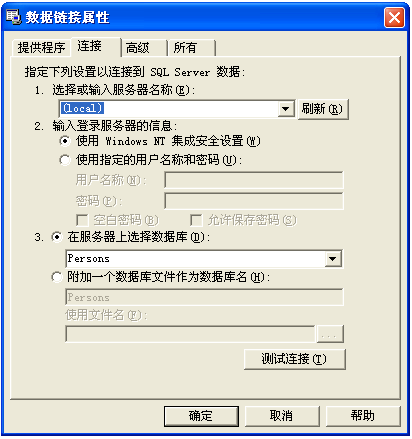
### 1.1 建立数据库

本例使用的数据库是用Microsoft SQL Server Management Studio 建立的一个建单数据库，其中有一个表，如下：



### 1.2 建立UDI文件

新建一个记事本文件，将其名称改为mydata.udi，双击进行如下设置：

### 1.3 引入相关DLL文件

在文件stdafx.h下加入以下代码：

#import "C:\Program Files\Common Files\System\ado\msado15.dll" no\_namespace rename("EOF","rsEOF")

编译，生成msado15.tli和msado15.tlh文件，将其添加到源文件中，方便查看。

### 1.4 初始化组件

在APP的InitInstance函数中加入以下初始化代码：

if (!AfxOleInit())

{

AfxMessageBox(\_T("AfxOleInit Failed!"));

return TRUE;

}

### 1.5 在对话框类同文件中定义一个全局的Connetcion智能指针对象，便于后续操作。

public:

\_ConnectionPtr m\_pConnetcion;

## 2、链接数据库

在对数据库进行操作之前，先要使用Connection对象链接数据库，先调用CreateInstance方法创建一个实例，然后对Connection对象进行有关设置，这里用到之前创建的UDL文件，再使用Open方法打开数据库。具体代码如下：

HRESULT hr;

try

{

hr=m\_pConnetcion.CreateInstance(\_\_uuidof(Connection));

if (FAILED(hr))

{

AfxMessageBox(\_T("CreateInstance Failed!"));

return;

}

m\_pConnetcion->ConnectionString=\_T("File Name=mydata.udl");

m\_pConnetcion->ConnectionTimeout=20;

hr=m\_pConnetcion->Open("","","",adConnectUnspecified);

if (FAILED(hr))

{

AfxMessageBox(\_T("Open Connection Failed!"));

return;

}

}

catch (\_com\_error e)

{

\_bstr\_t bstrSource(e.Source());

\_bstr\_t bstrDescription(e.Description());

AfxMessageBox(bstrSource+bstrDescription);

return;

}

## 3、对数据库进行增加操作

### 3.1 方法一，使用\_CommandPtr接口

\_CommandPtr pCommand;

pCommand.CreateInstance(\_\_uuidof(Command));

//生成SQL语句

\_bstr\_t AddSql;

TCHAR szBuffer[512];

wsprintf(szBuffer,\_T("INSERT INTO Persons\_Table1(NAME,SEX,AGE,SALARY,NUMBER) VALUES('%s','%s',%d,%d,%d)"),

m\_strName,m\_strSex,m\_iAge,m\_iSalary,m\_iNumber);

AddSql=T2BSTR(szBuffer);

pCommand->ActiveConnection=m\_pConnetcion;

pCommand->CommandText=AddSql;

pCommand->CommandType=adCmdText;

//执行SQL语句

try

{

\_variant\_t RecordsAffected;

HRESULT hr=pCommand->Execute(&RecordsAffected,NULL,adCmdUnknown);

if (SUCCEEDED(hr))

{

if (RecordsAffected.iVal>0)

MessageBox(\_T("Add Succeed!"));

else

MessageBox(\_T("Add Failed!"));

}

else

MessageBox(\_T("Error!"));

}

catch (\_com\_error\* e)

{

MessageBox(e->ErrorMessage());

}

### 3.2 方法二，使用\_ConnectionPtr接口

HRESULT hr;

//生成SQL语句?

\_bstr\_t AddSql;

TCHAR szBuffer[512];

wsprintf(szBuffer,\_T("INSERT INTO Persons\_Table1(NAME,SEX,AGE,SALARY,NUMBER) VALUES('%s','%s',%d,%d,%d)"),

m\_strName,m\_strSex,m\_iAge,m\_iSalary,m\_iNumber);

AddSql=T2BSTR(szBuffer);

\_variant\_t RecordsAffected;

//执行SQL语句

try

{

hr=m\_pConnetcion->Execute(\_bstr\_t(AddSql),&RecordsAffected,adCmdText);

if (SUCCEEDED(hr))

{

if (RecordsAffected.iVal>0)

MessageBox(\_T("Delete Succeed!"));

else

MessageBox(\_T("Delete Failed!"));

}

else

MessageBox(\_T("Error!"));

}

catch (\_com\_error\* e)

{

MessageBox(e->ErrorMessage());

}

## 4、对数据库进行删除操作

### 4.1 方法一，使用\_CommandPtr接口：

//创建\_CommandPtr实例

\_CommandPtr pCommand;

pCommand.CreateInstance(\_\_uuidof(Command));

//生成SQL语句

\_bstr\_t DeleteSql;

TCHAR szBuffer[512];

wsprintf(szBuffer,\_T("DELETE FROM Persons\_Table1 WHERE NUMBER=%d"),m\_iNumber);

DeleteSql=T2BSTR(szBuffer);

pCommand->ActiveConnection=m\_pConnetcion;

pCommand->CommandText=DeleteSql;

pCommand->CommandType=adCmdText;

//执行SQL语句

try

{

\_variant\_t RecordsAffected;

HRESULT hr=pCommand->Execute(&RecordsAffected,NULL,adCmdUnknown);

if (SUCCEEDED(hr))

{

if (RecordsAffected.iVal>0)

MessageBox(\_T("Delete Succeed!"));

else

MessageBox(\_T("Delete Failed!"));

}

else

MessageBox(\_T("Error!"));

}

catch (\_com\_error\* e)

{

MessageBox(e->ErrorMessage());

}

### 4.2 方法二，使用\_ConnectionPtr接口

HRESULT hr;

//生成SQL语句

\_bstr\_t DeleteSql;

TCHAR szBuffer[512];

wsprintf(szBuffer,\_T("DELETE FROM Persons\_Table1 WHERE NUMBER=%d"),m\_iNumber);

DeleteSql=T2BSTR(szBuffer);

\_variant\_t RecordsAffected;

//执行SQL语句

try

{

hr=m\_pConnetcion->Execute(\_bstr\_t(DeleteSql),&RecordsAffected,adCmdText);

if (SUCCEEDED(hr))

{

if (RecordsAffected.iVal>0)

MessageBox(\_T("Delete Succeed!"));

else

MessageBox(\_T("Delete Failed!"));

}

else

MessageBox(\_T("Error!"));

}

catch (\_com\_error\* e)

{

MessageBox(e->ErrorMessage());

}

## 5、对数据库进行查找操作

由于可能返回大量的结果，所以使用\_RecordsetPtr接口。

HRESULT hr;

//创建实例

\_RecordsetPtr pRecordset;

hr=pRecordset.CreateInstance(\_\_uuidof(Recordset));

if (FAILED(hr))

{

AfxMessageBox(\_T("Create Instance of Recordset failed!"));

return;

}

//生成SQP语句

CString strSqr;

\_variant\_t var;

CString strValue;

strSqr.Format(\_T("%d"),m\_iNumber);

strSqr=\_T("SELECT \* FROM Persons\_Table1 WHERE NUMBER=")+strSqr;

//执行查找操作

try

{

hr=pRecordset->Open(

\_variant\_t(strSqr),

m\_pConnetcion.GetInterfacePtr(),

adOpenDynamic,

adLockOptimistic,

adCmdText

);

if (FAILED(hr))

{

AfxMessageBox(\_T("Recordset Open Failed!"));

return;

}

//获取查找到的结果

if (!pRecordset->rsEOF)

{

var=pRecordset->GetCollect(\_T("NAME"));

if(var.vt!=VT\_NULL)

m\_strName=(LPCSTR)\_bstr\_t(var);

var=pRecordset->GetCollect(\_T("SEX"));

if(var.vt!=VT\_NULL)

m\_strSex=(LPCSTR)\_bstr\_t(var);

var=pRecordset->GetCollect(\_T("AGE"));

if(var.vt!=VT\_NULL)

m\_iAge=var.iVal;

var=pRecordset->GetCollect(\_T("SALARY"));

if(var.vt!=VT\_NULL)

m\_iSalary=var.iVal;

UpdateData(FALSE);

}

else

{

MessageBox(\_T("Not Found!"));

}

}

catch (\_com\_error\* e)

{

AfxMessageBox(e->ErrorMessage());

return;

}

//关闭记录集

pRecordset->Close();

pRecordset=NULL;

在这里只查询一条的记录，如果查结果包含多条记录，可使用如下语句遍历所查到的记录：

while(!pRecordset->rsEOF)

{

获取结果

pRecordset->MoveNext();

}

## 6、对数据库进行修改操作

先查询到要修改的记录，然后使用\_Recourdset的PutCollect方便修改记录值，记录使用Update方法更新记录，再关闭记录集。

HRESULT hr;

\_RecordsetPtr pRecordset;

hr=pRecordset.CreateInstance(\_\_uuidof(Recordset));

if (FAILED(hr))

{

AfxMessageBox(\_T("Create Instance of Recordset failed!"));

return;

}

CString strSqr;

\_variant\_t var;

CString strValue;

strSqr.Format(\_T("%d"),m\_iNumber);

strSqr=\_T("SELECT \* FROM Persons\_Table1 WHERE NUMBER=")+strSqr;

try

{

hr=pRecordset->Open(

\_variant\_t(strSqr),

m\_pConnetcion.GetInterfacePtr(),

adOpenDynamic,

adLockOptimistic,

adCmdText

);

if (FAILED(hr))

{

AfxMessageBox(\_T("Recordset Open Failed!"));

return;

}

if (!pRecordset->rsEOF)

{

pRecordset->PutCollect(\_T("NAME"),\_variant\_t(m\_strName));

pRecordset->PutCollect(\_T("SEX"),\_variant\_t(m\_strSex));

pRecordset->PutCollect(\_T("AGE"),\_variant\_t(m\_iAge));

pRecordset->PutCollect(\_T("SALARY"),\_variant\_t(m\_iSalary));

}

else

{

MessageBox(\_T("Not Found!"));

}

}

catch (\_com\_error\* e)

{

AfxMessageBox(e->ErrorMessage());

return;

}

pRecordset->Update();

pRecordset->Close();

pRecordset=NULL;

## 7、总结

对数据库的操作，首先需要建立与数据库的连接，这里使用全局的\_ConnectionPtr接口完成，对数据库的操作有的返回记录集，如查找操作和修改（由于修改之前必须先需要查找到待修改的记录，所以把修改操作也归为有返回记录集的操作）有的不返回记录集，如插入和删除操作，对于有返回记录集的操作，可使用\_RecordsetPtr接口，而无返回记录集的操作可使用\_CommandPtr和\_ConnectionPtr接口，不同的操作代码相似，差别主要在SQL语句上（例如插入和删除的操作）。