C++ Builder 由 Excel 数据导入 Paradox 表的实现

黄秀成

摘　要：Microsoft Excel 具有可视的强大的数据处理能力，Paradox 表则具有数据存储以及易与应用程序交互的优势，两者结合使用，可以有效提高数据处理效率。结合实例，讲解由 Excel 向 Paradox 表导入数据的方法。

关键词：Paradox；C++；Excel

1　引言

众所周知，Microsoft Excel 是专业数据处理软件，通过 Excel 应用程序的窗口界面，用户可以方便地进行数据录入和处理。Paradox 主要应用于数据容量不是非常大的数据管理，可以保证有较快的数据访问速度。本文介绍在 C++ Builder 中利用 OLE，实现由 Excel 表格向 Paradox 数据表导入数据的程序设计方法。

2　Paradox 数据表

这里以一个简单的学生信息系统为例，说明数据导入功能的实现方法。Paradox 数据表的字段定义如表 1 所示。

表 1　Paradox 表字段定义

字段名称	数据类型	数据长度
学号	整型	32
姓名	字符串	8
性别	字符串	2
入学年月	字符串	10
所在班级	字符串	10
家庭住址	字符串	30
联系电话	字符串	15

下面介绍使用程序代码创建 Paradox 数据表的具体方法。在编写实现代码之前，需要做点准备工作，由于使用 TTable、TQuery 和 TRegistry 类，在头文件中添加：

```
#include <DBTables.hpp>
#include <Registry.hpp>
```

实现代码如下：

```
void CreateAccessTable()
{
    TTable *Table1=new TTable(this); //定义并初始化一
//个 TTable 类的对象

    Table1->Active=false; //设置对象为非激活状态
    Table1->DatabaseName="D:\"; //设置文件存储位置
    Table1->TableName="StudentInfo.db"; //设置文件
//名称
    Table1->TableType=ttParadox; //设定数据表类型为
//Paradox
    Table1->FieldDefs->Clear(); //清空数据表字段定义
    Table1->FieldDefs->Add("学号",ftInteger,8,false);
//用于标识不同学生
    Table1->FieldDefs->Add("姓名",ftString,8,false);
    Table1->FieldDefs->Add("性别",ftString,2,false);
Table1->FieldDefs->Add("入学年月",ftString,10,false);
Table1->FieldDefs->Add("所在班级",ftString,10,false);
Table1->FieldDefs->Add("家庭住址",ftString,30,false);
Table1->FieldDefs->Add("联系电话",ftString,15,false);
Table1->CreateTable(); //按照字段定义,创建数据表
delete Table1; //删除数据表对象
}
```

通过上述代码，成功地在 D 盘创建了一个名称为“StudentInfo.db”的 Paradox 数据表文件，下一步工作就是将 Excel 表格中的数据导入该表中。

3　导入数据

通过上面步骤，已经创建了一个 Paradox 数据表，下面将演示如何将 Excel 表格中数据导入 Paradox 数据表。前提条件是 Excel 表格中应包含 Paradox 数据表字段对应的数据。要明确表格数据格式，这里规定表格的列字段分别为学号、姓名、性别、入学年月、所在班级、家庭住址、联系电话，第一行为列标题，从第二行起则为对应的数据内容。

实现代码如下：

```
void ImportData ()
```

```
    {
      TOpenDialog *OpenDialog1=new TOpenDialog(this);
      if(OpenDialog1->Execute())
      {
        //读取注册表,判断系统是否安装 Excel
        TRegistry *Reg=new TRegistry();
        Reg->RootKey=HKEY_CLASSES_ROOT;
        if(Reg->OpenKey("\\Excel.Sheet\\CurVer",false)) //注册
//表项存在
        {
          delete Reg;
        }
        else
        {
          MessageDlg ("Excel 未安装，无法启动！",mtError,
TMsgDlgButtons() << mbOK, 0);
          delete Reg;
          return;
        }
        Variant Ex,Wb,Sh;
        try
        {
          Ex=Variant::CreateObject("Excel.Application");
        }
        catch(...)
        {
          MessageDlg("无法打开 Excel! ",mtError,TMsgDlg
Buttons() << mbOK, 0);
        }
        //启动 Excel 应用程序但不显示
        Ex.OlePropertySet("Visible",false);
        //打开指定的文件
        Ex.OlePropertyGet("WorkBooks").OleFunction("Open",
              OpenDialog1->FileName.c_str());
        Wb=Ex.OlePropertyGet("ActiveWorkBook");
        Sh=Wb.OlePropertyGet("ActiveSheet");
        //读取表格第一行前 7 个单元格内容,用于判断表格文
//件格式是否正确
        AnsiString Str[7];
        Str[0]=Sh.OlePropertyGet("Cells",1,1).OlePropertyGet("
Value");
        Str[1]=Sh.OlePropertyGet("Cells",1,2).OlePropertyGet("
Value");
        Str[2]=Sh.OlePropertyGet("Cells",1,3).OlePropertyGet("
Value");
        Str[3]=Sh.OlePropertyGet("Cells",1,4).OlePropertyGet("
Value");
        Str[4]=Sh.OlePropertyGet("Cells",1,5).OlePropertyGet("
Value");
        Str[5]=Sh.OlePropertyGet("Cells",1,6).OlePropertyGet("
Value");
        Str[6]=Sh.OlePropertyGet("Cells",1,7).OlePropertyGet("
Value");
        //删除列标题中的空格
        int pos;
        int i;
        for(i=0;i<6;i++)
        {
          pos=Str[i].Pos(" ");
          while(pos>0)
          {
            Str[i].Delete(pos,1);
            pos=Str[i].Pos(" ");
          }
        }
        //Paradox 数据表字段名称
        AnsiString TitleStr[7];
        TitleStr[0]="学号";
        TitleStr[1]="姓名";
        TitleStr[2]="性别";
        TitleStr[3]="入学年月";
        TitleStr[4]="所在班级";
        TitleStr[5]="家庭住址";
        TitleStr[6]="联系电话";
        //判断表格文件字段是否符合要求
        i=0;
        bool IsMatch=true;
        while(i<7)
        {
          if(Str[i]==TitleStr[i])
            i++;
          else
          {
            if(i==1)
              i++;
            else
            {
              IsMatch=false;
              MessageDlg (" 文件格式不匹配！",mtError,
TMsgDlgButtons() << mbOK, 0);
              break;
            }
          }
        }
        if(IsMatch) //如果文件格式匹配,则导入数据
        {
          TQuery *Query1=new TQuery(this);
          //RequestLive 属性用于确定是否希望从查询指令中
//获取可读可写的查询结果,
            //缺省设置为假,表示查询结果只读。如果要使记录
//能够编辑,需要设置该属性为真。
            //这里,要写入数据,故设置为真。
          Query1->RequestLive=true;
          Query1->DatabaseName="D:\\";
```

```
        Query1->Close();
        Query1->SQL->Clear();
    Query1->SQL->Add("select * from studentinfo.db");
        Query1->Open();
        int RowCount;
        //返回 Excel 工作表的行数
          RowCount  =Sh.OlePropertyGet  ("UsedRange").
OlePropertyGet("Rows").
                              OlePropertyGet("Count");

        AnsiString InputStr[7]; //当前导入信息
        AnsiString FilterString;//筛选条件
        int RecCount;
        int AddCount=0; //添加记录数量
        for(int j=0;j<RowCount-1;j++)
        {
          //读取学号
              InputStr[0]=Sh.OlePropertyGet("Cells",j+2,1).
OlePropertyGet("Value");

              //组织筛选条件字符串
            FilterString =Query1 ->Fields ->Fields  [0] ->
FieldName+"="+StrToInt(InputStr[0]);

            Query1->Filtered=false;
            Query1->Filter=FilterString;
            Query1->Filtered=true;

              //判断记录是否存在
            RecCount=Query1->RecordCount;
            if(RecCount==0) //不存在
            {
              Query1->Close();
              Query1->SQL->Clear();
                Query1  ->SQL  ->Add  ("select  *  from
studentinfo.db");
              Query1->Open();

                //读取姓名
              InputStr  [1]=Sh.OlePropertyGet  ("Cells",j+2,2).
OlePropertyGet("Value");
                //读取性别
              InputStr  [2]=Sh.OlePropertyGet  ("Cells",j+2,3).
OlePropertyGet("Value");
                //读取性别
              InputStr  [3]=Sh.OlePropertyGet  ("Cells",j+2,4).
OlePropertyGet("Value");
                //读取性别
              InputStr  [4]=Sh.OlePropertyGet  ("Cells",j+2,5).
OlePropertyGet("Value");
                //读取性别
              InputStr  [5]=Sh.OlePropertyGet  ("Cells",j+2,6).
OlePropertyGet("Value");
                //读取性别
              InputStr  [6]=Sh.OlePropertyGet  ("Cells",j+2,7).
OlePropertyGet("Value");

                //在数据表最后添加一个空记录
            Query1->Append();

                //将数据赋给数据记录各字段
          Query1->Fields->Fields[0]->Value=InputStr[0];
          Query1->Fields->Fields[1]->Value=InputStr[1];
          Query1->Fields->Fields[2]->Value=InputStr[2];
           Query1->Fields->Fields[3]->Value=InputStr[3];
            Query1->Fields->Fields[4]->Value=InputStr[4];
           Query1->Fields->Fields[5]->Value=InputStr[5];
           Query1->Fields->Fields[6]->Value=InputStr[6];
                //提交修改
            Query1->Post();
                //添加的记录数加 1
              AddCount++;
            }

            Query1->Filtered=false;
          }
            delete OpenDialog1;
            delete Query1;
     AnsiString Msg="共添加"+IntToStr(AddCount)+"条记录！";
      MessageDlg(Msg.c_str(),mtInformation,TMsgDlgButtons
() << mbOK, 0);
        }
        //退出 EXCEL
        Ex.OleFunction("Quit");

          //释放对象
        Ex=Unassigned;
      }
    }
```

4 结语

利用 C++ Builder，实现了 Microsoft Excel 与 Paradox 数据表的结合使用，充分发挥二者特点，达到了优势互补，有效减少了工作量，切实提高了数据处理效率。上述代码已在 Windows XP SP3 和 C++ Builder 6.0 环境下调试通过。

（收稿日期：2011-12-09）

C++ Builder由Excel数据导入Paradox表的实现

作者： 黄秀成
作者单位：
刊名： 电脑编程技巧与维护
英文刊名： Computer Programming Skills & Maintenance
年，卷(期)： 2012(5)