

文章编号: 1005—0523(2004)05—0041—04

基于 MFC ODBC 的数据转换技术的研究与实现

刘立月, 魏永丰

(华东交通大学 信息工程学院, 江西 南昌 330013)

摘要: 研究在分布式试验与测试系统中工程数据库的数据转换问题, 通过对 MFC ODBC 访问数据库机理与数据转换过程中的关键技术分析, 利用 MFC ODBC 实现了工程数据库转换。
关 键 词: 数据转换; MFC ODBC; 分布式试验与测试系统
中图分类号: TP393 **文献标识码:** A

1 引 言

在分布式试验与测试系统中, 由于系统各个节点的任务不同, 所连接的数据库管理系统不一定相同, 有的甚至是非标准的文件格式数据, 因此试验与测试系统一般希望不依赖数据库管理系统。数据管理一个重要的任务就是针对不同试验与测试任务工程数据的格式, 解决分布式异构工程数据库的数据转换问题, 实现系统的数据共享, 提高试验与测试工程数据库的利用率和交互性。

数据转换首先进行类型转换, 访问源数据库或文件系统, 将源数据库或文件的数据定义模型转换为目标数据库的数据定义模型, 然后进行数据重组, 即将源数据库或文件系统中的数据导入到目的数据库中。

2 MFC ODBC 访问数据库方法及其 VC 实现机理

MFC ODBC 封装了 ODBC 功能的类, 它通过一些类 (如 CDatabase 类、CRecordSet 类、CRecordView 类等) 提供与 ODBC 的接口, 使得用户可以不需处理

ODBC API 中的繁杂处理就可以进行数据库操作。由于它功能丰富, 操作相对简便, MFC ODBC 类在实际开发中应用最广。在工程数据库数据转换中, 采用 MFC ODBC 作为分布式试验与测试系统开发数据库接口。以下是对 MFC ODBC 与数据库接口工作原理及 VC 接口编程实现机理介绍。

2.1 ODBC 数据库访问结构

ODBC 访问数据库的结构图如图 1 所示。ODBC 是通过使用驱动程序来实现应用程序对数据库的独立性的。各个模块的功能分别是:

- 1) ODBC 应用程序通过 ODBC 函数向数据库发送 SQL 语句并处理返回结果;
- 2) ODBC 驱动程序管理器负责管理和装载特定的驱动程序;
- 3) ODBC 驱动程序负责处理 ODBC 函数调用, 提交 SQL 请求给特定的数据源, 并返回结果给应用程序。

2.2 MFC 数据库应用程序结构

MFC 数据库应用程序一般引用三个基本类: 数据源类 (CDatabase) 用来实现对数据源的一个连接; 记录集类 (CRecordset) 用来管理数据源中选定的记录集合。CRecordView 类, 是用来快速构建数据操作

界面的视图类.应用程序的视图对象与一个记录集对象绑定在一起,通过“对话框数据交换”(DDX)机制来实现视图与记录集间的信息传递.其工作原理如图 2 所示.

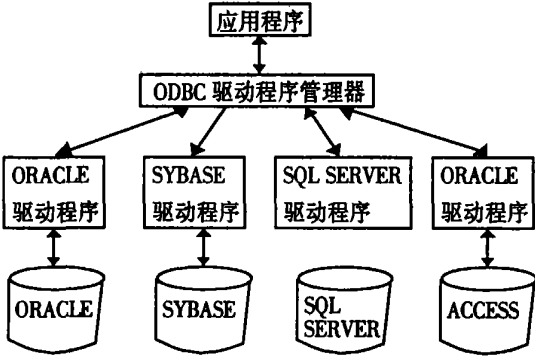


图 1 ODBC 数据库访问结构

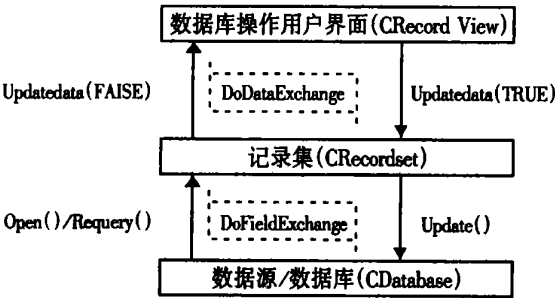


图 2 VC++ 数据库应用程序结构

3 利用 MFC ODBC 实现源数据库导入目标数据库

3.1 ODBC 源数据库转换 ACCESS 数据库的过程

在数据库转换中,有时要想实现严格的等价转换是比较困难的.转换前,要确定两种模型中所存在的各种语法和语义冲突(命名冲突、格式冲突、结构冲突).转换后,一方面将源数据库模式中所有需要共享的信息都转换到目的数据库中,另一方面这种转换又不能包含冗余的关联信息.利用 MFC ODBC 实现数据库导入目标数据库的流程图如图 3 所示.

3.2 ODBC 源数据库转换 ACCESS 数据库实例

利用 MFC ODBC 实现源数据库转换 ACCESS 数据库,以 FoxPro 数据库为例.转换前 FoxPro 库及转换后 Queries 数据库部分内容分别如图 4、图 5 所示.

4 ODBC 数据源转化成目标数据库过程中的关键技术

在 ODBC 源数据库转化 ACCESS 数据库中,关键是解决两个问题.一是如何利用 MFC ODBC 获取异构数据库结构信息;二是不同数据库数据类型匹配

问题.

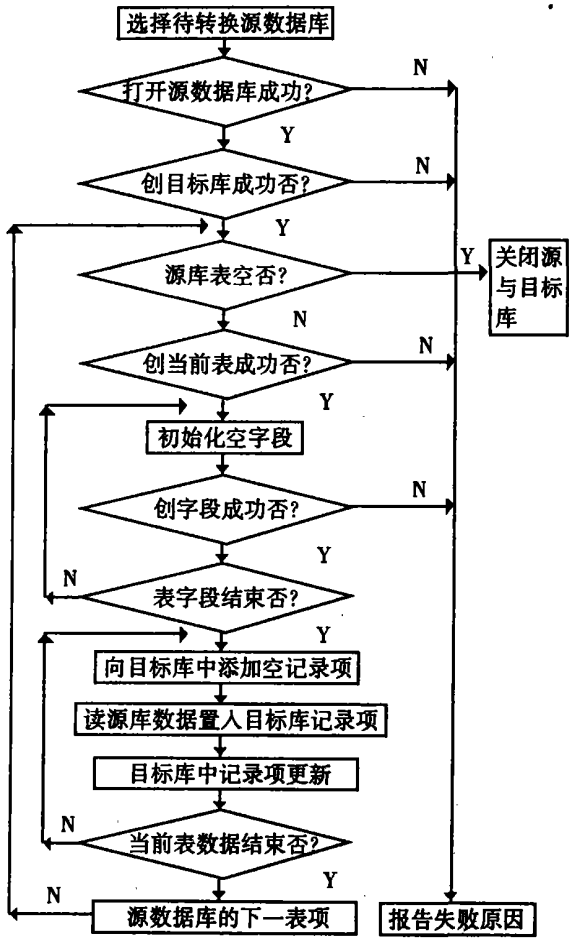


图 3 ODBC 源数据库转化 ACCESS 数据库流程图

4.1 利用 MFC ODBC 获取异构数据库结构信息

单纯利用 MFC 类获取异构数据库的结构信息仍然比较困难,因此需要将 MFC 和传统 ODBC API 编程结合起来.在实际中,可以利用 ODBC API 接口函数重载了 MFC 中 CRecordset 类的部分成员函数,创建 CColumns 和 CTable 类,以实现获取异构型数据库结构信息.其类的定义如下:

```
class CColumns : public Crecordset //用于获取字段相关信息
{
    DECLARE_DYNAMIC(CColumns)
public:
    CColumns(CDatabase * pDatabase = NULL); BOOL Open(UINT
nOpenType = forwardOnly, LPCSTR lpszSQL = NULL,
DWORD dwOptions = readOnly);
    ///{AFX_FIELD(CColumns, CRecordset)
    CString m_strQualifier; //表的资格
    CString m_strOwner; //表的拥有者
    CString m_strTableName; //字段所在的表名
```



图 4 转换前 FoxPro 数据库

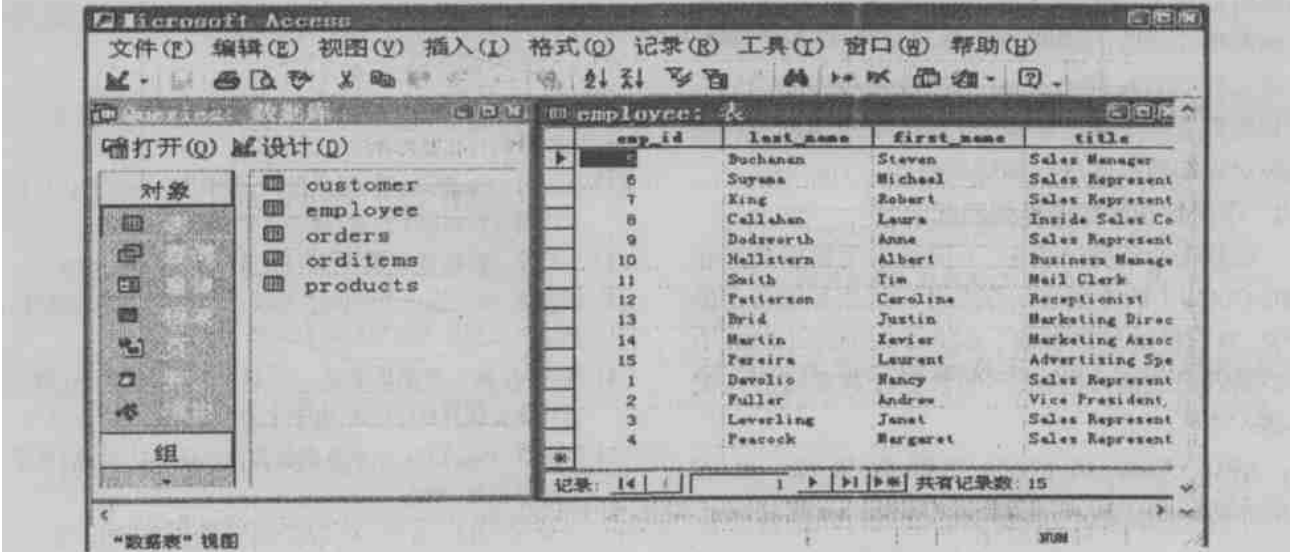


图 5 转换后 ACCESS 数据库

```
CString m_strColumnName; //字段名
int m_nDataType; //字段类型值(不同数据库类型,其值不唯一)
CString m_strTypeName; //字段类型名
long m_lPrecision; //字段精度
long m_lLength; //字段长度
int m_nScale; //数值范围
int m_nRadix; //基数
int m_nNullable; //有效性
//}}AFX_FIELD
CString m_strQualifierParam;
CString m_strOwnerParam;
CString m_strTableNameParam;
CString m_strColumnNameParam;
protected:
virtual CString GetDefaultConnect(); // default connection string
virtual CString GetDefaultSQL(); // default SQL for Recordset
virtual void DoFieldExchange(CFieldExchange * pFX); // RFX support
};
class CTables : public CRecordset //用于获取表相关信息
{
DECLARE_DYNAMIC(CTables)
public:
CTables(CDatabase * pDatabase = NULL);
BOOL Open(UINT nOpenType = forwardOnly, LPCSTR lpszSQL = NULL,
DWORD dwOptions = readOnly);
//{{AFX_FIELD(CTables, CRecordset)
CString m_strQualifier; //表的资格
CString m_strOwner; //表的拥有者
```

```
CString m_strName; //表名
CString m_strType; //表的类型
CString m_strRemarks; //注释参数
//}}AFX_FIELD
CString m_strQualifierParam;
CString m_strOwnerParam;
CString m_strNameParam;
CString m_strTypeParam;
protected:
virtual CString GetDefaultConnect(); // default connection string
virtual CString GetDefaultSQL(); // default SQL for Recordset
virtual void DoFieldExchange(CFieldExchange * pFX); // RFX support
};
```

以上两个类对 CRecordset 的 Open 和 DoFieldExchange 函数进行了重载. 应用程序可以在需要时创建 CTable 或 Ccolumns 类, 并调用 Open 成员函数建立相应的表结构和字段结构记录集. 利用自定义的 CTable 和 Ccolumns 类, 应用程序能获取任何异构数据库库结构信息, 根据获得的信息可以方便地对未知数据库进行相应的操作.

4.2 不同数据库数据类型匹配问题

在数据库转换过程中, 不同数据库数据类型在 MFC ODBC 中的类型表示存在各种语法和语义上的冲突, 为了解决这些冲突, 必须深入 MFC ODBC 中的类型匹配过程, 以解决不同数据库数据类型匹配问题.

MFC ODBC 访问不同源数据库, 通过 GetFieldValue 函数获取的数据以 CDBVariant 或 CString 类形

式存储数据, 在其间涉及到 SQL data type 与 C data type 的内部转换; SetFieldValue 函数将 CDBVariant 或 CString 类数据置入目标数据库(以 ACCESS 为例)中, 其转换过程如图 6 所示.

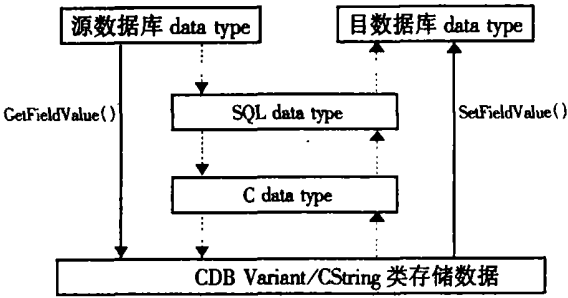


图 6 源库转换目标库中数据类型转换过程

分布式试验与测试系统中, 大都是数值之间的转换, 不同数据库之间数据类型基本能够找到匹配, 若在转换过程中, 对于无法匹配的数据类型, 可以以文本类型考虑.

参考文献:

[1] David J. Kruglinski 著, 潘爱民, 王国印译. Visual C++ 技术内幕 (第四版) 北京: 清华大学出版社, 1999.
[2] 李英军, 等译. 设计模式[M]. 机械工业出版社, 2000.
[3] 侯俊杰. 深入浅出 MFC[M]. 武汉: 华中科技大学出版社, 2001.
[4] 康诺利 (英). 数据库系统——设计、实现与管理 (第三版) (英文版)[M]. 北京: 电子工业出版社, 2002.
[5] 韩云君. Visual C++ 6.0 高级编程范例[M]. 北京: 电子工业出版社, 2002.

Research and Implementation of Data Transformation Technology
Based on MFC ODBC in DEMS

LIU Li-yue, WEI Yong-feng

(School of Information Engineering, East China Jiaotong University, Nanchang 330013, China)

Abstract: According to the research of the data transformation problem in DEMS, this paper analyzes the key technology of data transformation and how to access database by MFC ODBC. The transformation among various engineering databases has been implemented with MFC ODBC.

Key words: Data transformation; MFC ODBC; DEMS (Distributed Experiment & Measurement System)