在 VC6 中利用动态 SQL 实现对 DB2 访问*

蒋龙龙[1]，何先刚[2]
(1. 重庆邮电学院，重庆 400065；2. 重庆邮电学院期刊编辑室，重庆 400065)

摘　要：提出了一种通过 CLI 层操纵数据库，使用动态 SQL 语句访问数据库的方法。利用此方法可以在 VC 中实现对 DB2 数据库的访问。此法同时也适用于编制其他数据接口。

关键词：DB2；动态 SQL；C++
中图分类号：TP311.132.3　　**文献标识码**：A

An Approach to Accessing to DB2 by Means of Dynamic SQL In VC6

JIANG Long-long[1], HE Xian-gang[2]
(1. *Institute of Optical Electronic Engineering, Chongqing University of Posts and Telecommunications, Chongqing* 400065; 2. *The Editorial Dept. of the Journal of Chongqing University of Posts and Telecommunications, Chongqing* 400065, *China*)

Abstract　In this paper the authors introduce an access to the database which is operated under CLI by using dynamic SQL statements. By means of this access, it is possible to have the access to DB2 database in VC. And this method is also suitable for programming other data interfaces.

Key words　DB2; dynamic SQL; C++

0　引　言

IBM 公司的 DB2 通用数据库是一种流行的关系数据库管理系统。由于它允许用户通过结构化查询语言来创建、修改和控制数据库，因此为有关应用提供了极大的灵活性。换言之，DB2 可以通过多种方式实现应用程序在 DB2 数据库中对数据的存取和操纵，例如：嵌入式 SQL，DB2，CLI，JAVA，或 ODBC 最终用户工具。根据具体应用的不同可以灵活地加以选择。

作为一种面向对象的结构化程序设计语言，Visual C++ 成为大型软件系统中的一种。而在 Visual C++ 6.0 中，可使用其提供的 ODBC 类库，但在一些大中型系统的运用中，该类库有很大不足，表现在对数据量处理和特殊 SQL 语句的执行等方面。因此，可以采取非常灵活的 CLI(Call Level Interface)方式来操纵数据库。

CLI 是 DB2 提供的一种编制动态 SQL 程序的方法，在 CLI 语句中没有一个真正的 SQL 语句，所有 SQL 操作均是由 C 语言编译器能识别的 C 程序调用来实现的，所以它不需要执行预编译命令，只要通过传统的 C 编译器和 CLI 库就可生成用 CLI 编制的可执行程序。CLI 与 ODBC 的第 1 层完全兼容，对 ODBC 第 2 层的大部分函数也是兼容的，所以 CLI 具有很强的可移植性，它完全可移植到别的

* 收稿日期：2000－10－12
作者简介：蒋龙龙(1973-)，男，重庆市人，重庆邮电学院光电工程学院工程师，主要研究方向为计算机及其应用；何先刚(1969-)，男，四川大竹人，工程师，在职硕士生，主要研究方向为信号与信息处理。

数据系统(如 INFORMIX,SYBASE)上运行。

1　以 CLI 方式实现动态 SQL 的设计

在 DB2 中,句柄是一个重要的概念,它表示由 CLI 所实现的幕后管理信息,在 C 中表现为一个简化的 LONG 型变量。CLI 支持 3 种句柄类型:

(1) 环境句柄:反映了应用程序的整体状态,在程序开始时要分配环境句柄,并用它定义数据库的连接,在程序结束时需释放。

(2) 连接句柄:表示应用程序对数据库的连接,一个程序可连接多个数据库,或者对同一数据库连接多次,每次连接使用一个不同的连接句柄。利用连接句柄可提交或回滚在各数据库中的事务。类似于 POWERBUILD 中的使用 CONNECT 连接事务对象。

(3) 语句句柄:反映 SQL 语句的执行状态。CLI 中没有 SQLCA(返回码),SQLDA 及结构,而语句句柄综合表达这些信息,所以程序不在需要直接管理这些数据结构,它可直接得出 SQL 语句的当前状态。

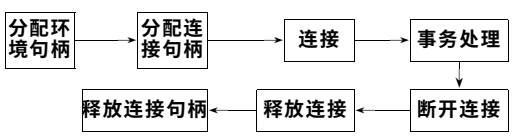

图1　CLI的处理过程

对句柄来说,一个 CLI 程序的整体结构如图 1 所示。图 1 中,CLI 的实现是一个顺序过程,在编制程序时,最关心的是 CLI 函数的使用,对 DB2 的 CLI 函数的使用可参考 ODBC 标准,对 DB2 的扩充函数可尽量不要使用,否则会限制应用程序的移植性。

2　利用 Visual C^{++}6.0 实现事务处理的举例

使用 VC^{++} 时,可使用 DB2 提供的 SQLCLIH1.H 中的函数声明,也可使用其 MFC 提供的 afxdb.h 中的 ODBC 声明(如不使用 DB2 的扩展函数)。在本文中,提供如下 2 个类:

(1) BDATA 类

BDATA 类的作用是开辟一段内存缓冲区,将 SELECT 语句查询出的数据按顺序放入内存中,在查询出多行数据时,避免使用特定的结构或链表,同时对 SELECT 语句中的字段数无限制。

(2) BSQL 类

BSQL 类中的 3 个函数分别为连接数据库,断开数据库,执行 SQL 语句。其中执行 SQL 语句使用大量的 CLI 函数,如下所示:

```
int BSql::ExecSql1Cmd(char * input)
{
    console.free();
    console.alloc(32768); //分配内存大小,如数据量较多可增加
    SQLHSTMT hstmtl; //分配语句句柄
    SQLINTEGER rowcount;
    SQLRETURN rc; //返回代码
    SQLSMALLINT ncols; //结果集中列的数目
    SQLSMALLINT colnamelen; //列名的真正长度
    SQLSMALLINT coltype; //结果列的数据类型
    SQLCHAR colname[19]; //结果列的名字
    if(henv==NULL)
    {
    return 0;
    }
    if(hdbc==NULL)
    {
    return 0;
    }
    int i=0,j=0,row=0;
      struct {SQLINTEGER ind;
              Char s[256]
              SQLSMALLINT colrow;
             } sp_code[100]; //公共的取数据绑定
    strcpy(space,input);
    if(strlen(space)<=0)
    {
    return  -1;
    }
    rc=SQLAllocStmt(hdbc, &hstmt1); //分配语句句柄
    rc = SQLExecDirect (hstmt1, (SQLCHAR * ) space, SQL_
NTS);//准备并执行 SQL 语句
    if(rc==SQL_NO_DATA_FOUND) //空集的 SQL 语句
    {
      SQLTransact(henv,hdbc,SQL_ROLLBACK);
      SQLFreeStmt(hstmtl,SQL_DROP);
    return 0;
```

```
    }
    if(rc==SQL_ERROR) // SQL 语句不合法
    {
    SQLTransact(henv,hdbc,SQL_ROLLBACK);
    SQLFreeStmt(hstmt1,SQL_DROP);
    return 0;
    }

    if(rc==SQL_SUCCESS || rc==SQL_SUCCESS_WITH_INFO){
      rc=SQLNumResultCols(hstmt1,&ncols); //返回结果集合中列的数目
      if (ncols==0)
        {

    rc=SQLRowCount(hstmt1,&rowcount);
      if(rowcount>0)//为 UPDATE,INSERT,DELETE 语句
      {
      rc=SQLEndTran(henv,hdbc,SQL_COMMIT); //提交
      if(rc! =SQL_SUCCESS)
      SQLTransact(henv,hdbc,SQL_ROLLBACK);
      SQLFreeStmt(hstmt1,SQL_DROP);
    return 1;
    }
    else
    SQLTransact(henv,hdbc,SQL_ROLLBACE);
    SQLFreeStmt(hstmt1,SQL_DROP);
    return -1;
      }
    }
    else

          for(j=1;j<(ncols +1);j++){
          rc=SQLDescribeCol(hstmt1,j,colname,18
           &colnamelen, &coltype,NULL,NULL,NULL); //对结果每一列处理
          sp_code[j].colrow=coltype;
           rc=SQLBindCol(hstmt1,j,SQL_C_CHAR,sp_code[j].s,256,&sp_code[j].ind); //数据绑定
          sp_code[j].colrow=coltype;
          }
          while((rc=SQLFetch(hstmt1))==SQL_SUCCESS)
          {
          i++
          for(row=1;row<ncols+1;row++
          {
          console.add(sp_code[row].s,25); //放入内存中
          }
          /*
          //如希望对每一字段定位，可根据 SP_CODE[ROW]的值，分别放入内存
          switch(sp_code[row].colrow)
          {
          case SQL_VARCHAR; //VARCHAR 型，INT 型等
          console.add(sp_code[row].s,25);
          break;
          }}}
        SQLFreeStmt(hstmt1,SQL_CLOSE); //释放语句句柄
    }
    SQLTransact(henv,hdbc,SQL_COMMIT);
    SQLFreeStmt(hstmt1,SQL_DROP);
    }
    return 1;

    }
```

3 结 论

将上述 2 个类加入到程序中，连接数据库后，可执行任意 SQL 语句。在实际使用中，可辅以开内存区的方式，通过将数据集中后的处理，可以使 VC++ 处理数据库达到快速高效的处理效果。其处理界面及处理效果可以与 POWERBUILD 的使用相比美。另一方面，如果加上其 DB2 的扩展函数和 API 函数，还可对数据库进行备份、恢复、创建等，实现 POWERBUILD 等软件不能实现的功能。

在我们编制的一个软件中，通过对上述各例的应用，较好地达到设计目的。此外，有关设计具有较好的可移植性，通过对有关内容稍加修改，在 AIX6000 上达到良好的运行效果。

参 考 文 献

[1] 国家智能计算机研究开发中心. DB2 关系数据库系统管理与应用开发指南[M]. 北京：电子工业出版社，1998.

[2] 侯俊杰. 深入浅出 MFC(第 2 版)[M]. 武汉：华中科技大学出版社，2000.