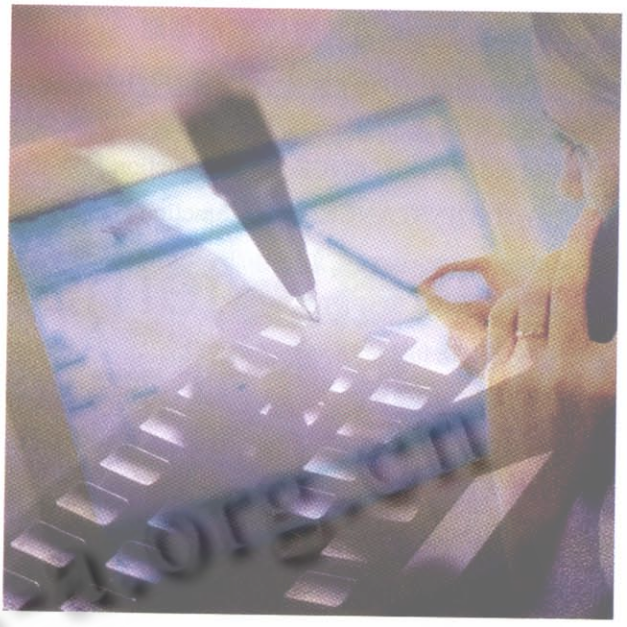
在DB2中创建用户自定义函数



华中理工大学 IBM技术中心 黄志 余祥宣

DB2是世界上最早的关系数据库之一。DB2数据库管理系统提供了UDF（User Define Function用户自定义函数）机制，允许用户创建SQL语言函数。

在目前的DB2数据库的开发中，如何创建用户自定义函数以满足应用的要求，是这类开发中常遇到的问题。本文针对AIX（UNIX）系统，对上述问题作一介绍。并给出一个简单的实例。

UDF简介

DB2最早是运行在大型机的MVS/ESA等平台上的，后来逐渐也在中型机AS/400的OS/400操作系统、RS/6000小型机的AIX操作系统平台、OS/2以及Win95、Winnt等各种平台上运行。

UDF即User Defined Function用户定义函数，允许写自己的sql的扩展。DB2的那些自建函数是一个有用的函数集，不过它们可能不能满足你所有的要求，这样就需要扩展SQL。通过扩展SQL，用户可以根据应用的需要定义自己的函数。

就实现方式而言，UDF有两种实现方式，一是Sourced方式，二是External方式。Sourced方式的实现是通过继承已有的函数的功能，因此实现较简单，只需要用SQL来创建自定义函数即可。External方式还需要通过程序设计实现。

UDF按类型划分有三类：Scalar、Column和Table。

1. Scalar：每次调用返回一个单一的值。Scalar函数可以通过Sourced方式实现，也可以通过External方式实现。

2. Column：是表的一列数据的函数。在DB2中，Column函数有时也叫聚集函数。比如DB2有的自建函数AVG()是用来计算某一列的平均值的函数。Column函数只能通过Sourced方式实现。

3. Table：返回一个表给调用它的SQL语句。表函数只能在SELECT语句的FROM子句调用。这个函数可以把SQL语言的数据处理功能用到非DB2的数据上，或者是把这些数据转移到DB2表中。比如，表函数可以把一个文件的数据转移到表中。Table函数只能通过external方式实现。

Sourced方式的UDF

Sourced方式是通过继承已有SQL函数的功能实现，其建立较简单，只需一条SQL语句即可。下面给出两个例子。

create function "+"(smallint, smallint) returns smallint Source "+"(smallint, smallint)

这个语句创建有两个smallint类型参数进行"+"运算的函数，函数的返回值是smallint类型。这个函数继承了DB2中提供的"+"函数的功能。

create function avg_us_speed(mph) returns mph source avg(integer)这条语句创建了求某个column平均值的函数avg_us_speed，其输入参数为mph(用户自定义的)类型，其输出参数为mph类型。函数继承了DB2中提供的avg函数的功能。

在DB2中还允许函数sourced在前面定义的函数。函数名的复载也是允许的。这样同一个函数名可以用多次。比如前面定义的函数avg--_us_speed函数也可以定义为avg，尽管这个名字已经存在，调用函数时，DBM（Database Manager）会启用与输入参数匹配的函数。

External 方式的 UDF 的创建

创建 External 方式的 UDF 要完成以下三个工作:

· 编写 UDF 源文件，编译连接，并把产生的可执行文件拷贝到 $DB2INSTANCE/sqllib/function 目录下。

· 在 $DB2INSTANCE/sqllib/function 目录下建立文件<prog-name>.exp,在文件中加入<prog_name>。其中 <prog_name> 是编写 UDF 程序名。

· 用 SQL 语句建立用户自定义函数。

下面给出一个创建 External Scalar UDF 的实例:

1. 编写 hztest.c 文件，运行 bldxlcudf hztest 命令。hztest.c 文件:

```
#include <wchar.h>
#include <stdio.h>
#include <sqludf.h>
#include <sqlsystm.h>
void SQL_API_FN hztest(short *input,
            short *output,
            short *input_ind,
            short *output_ind,
            SQLUDF_TRAIL_ARGS)
{
if (*input_ind==-1)
{
  /* if input is null, likewise output */
  *output_ind=-1;
  }
  else { /* input is not null. set output to *input+3
*/
          *output=*input+3;
          /*and set out null indicator to zero */
          *output_ind=0;
          }
return;
}
```

程序中，input 作数据输入指针，output 作数据输出指针，input_ind作有无数据输入的标志指针，output_ind 作有无数据输出的标志指针。当*input_ind = -1时，表示没有输入数据带入函数。*output_ind = -1，表示没有数据输出。SQLUDF_TRAIL_ARGS 是存放跟踪数据信息的数据结构。

bldxlcudf hztest 是对 hztest.c 文件进行编译、连接产生可执行文件，并把它拷贝到 $DB2INSTANCE/sqllib/function 目录下。

2. 在 $DB2INSTANCE/sqllib/function 目录下，建一个 hztest.exp 文件，在文件中写上 hztest。

3. 用 SQL 语句创建自定义函数。

以DB2的系统管理员用户(比如是db2inst1)身份建立自定义函数:

```
CREATE FUNCTION HZTEST (SMALLINT)
    RETURNS SMALLINT
    EXTERNAL NAME 'hztest!hztest'
    LANGUAGE C
    PARAMETER STYLE DB2SQL
    DETERMINISTIC
    NO SQL
    NO EXTERNAL ACTION;
```

CREATE FUNCTION中，CREATE FUNCTION HZTEST (SMALLINT) RETURN SMALLINT表示建函数 HZTEST，函数的输入参数是 SMALLINT 类型的，返回值是 SMALLINT 类型的。CREATE FUNCTION 中，EXTERNAL NAME 'hztest!hztest' 表示实现的函数库名(可执行文件)！库中函数名(文件中的函数)，这种格式是对c语言的，对不同语言的EXTERNAL NAME 格式是不同的。NO SQL 表示这个函数不能给出SQL语句，否则在运行时就会产生 (SQLSTATE 38502) 的错误。NO EXTERNAL ACTION 语句描述函数是否有外部操作，即对那些数据库管理系统不管的对象的操作。

上面创建的用户自定义函数，db2inst1 用户使用时，只要给出函数名 hztest，其他用户在使用时要给出全名 db2inst1.hztest。比如:

```
select db2inst1.hztest(no) from test
```

在建立 UDF 时，要注意到使用外部资源，可能会造成 DB2 检测不出的死锁。另外对一般文件的写操作是允许的，但写到屏幕或从键盘输入是不允许的。

总 结

本文对 DB2 中的 UDF 做了简单的介绍，举例说明了在 DB2 中建立 UDF 的方法 (包括 Sourced 方式和 External 方式)。给出建立 EXTERNAL SCALAR UDF 的例子是在 AIX (IBM 公司的 UNIX) 操作系统上，对 DB2 Version 5.0 (UDB Version 5.0) 调试通过的。■