· tcp-small-servers, 关闭命令：no service tcp-small-servers

· udp-small-servers, 关闭命令：no service udp-small-servers

4 加强路由器口令保护

强健的口令是保护路由器的最佳防线。口令设置应该混合大小写字母和数字，而且位数不应少于8位。

对CISCO路由器而言，可以给CONSOLE、AUX、VTY三种访问方式都设置密码。具体命令如下：

```
line console 0 (console 方式)
  login
  password 口令
line aux 0 (aux 方式)
  login
  password 口令
line vty 0 (vty 方式)
  login
  password 口令
```

上述密码都是明文形式保存在路由器的配置文件中，可以用命令对密码实现加密：

```
service password encrytion
```

5 结束语

通过以上介绍的各种方法，我们成功的将一台普通路由器配置为一台堡垒路由器，在没有增加任何投入的情况下，提高了整个园区网的安全性。但应该说明的是，堡垒路由器的实现是以牺牲整个网络的效率为代价的，可能会影响到园区网对外访问的速度。如何在安全与性能之间找到一个最佳平衡，是摆在网络管理员面前的一个重要课题。 ■

参考文献

1 Chris Brenton, Andrew Hamilton, Gary Kessler,Mastering Cisco Routers [M], 美国SYBEX Inc,1999。

2 Cisco system,cisco internetwork design [M],美国CISCO公司，1998。

3 Chris lewis, Cisco Tcp/Ip Routing Professional Reference [M]，机械工业出版社，1999。

ORACLE中的动态SQL编程

姚世军 （郑州解放军信息工程大学测绘学院0215 450052）

摘要：为了提高PL/SQL程序的性能，增加程序的灵活性，PL/SQL中增加了动态SQL。本文通过详细的例子介绍了对动态SQL编程的各种方法，并指出了在编程时应注意的问题。

关键词：PL/SQL 动态SQL

1 引言

PL/SQL是专门为ORACLE数据库设计的一种程序设计语言，它把过程化结构与ORACLE的SQL语句无缝地集成在一起，产生了强有力的结构化语言。用它可以编制存储过程、函数等数据库对象。

PL/SQL中有两类语句：一类是静态SQL语句，它在编译时语句的内容是固定的，语句不会随执行的不同而变化。静态SQL语句中只能执行数据处理语句，如INSERT、UPDATE等。另一类是动态SQL语句，程序在运行时处理和建立变化的SQL语句，在编译时并不知道语句的整个内容。

只所以需要动态SQL语句，主要有下面原因：

· 要在PL/SQL块中执行SQL的数据定义语句(CREATE等)、数据控制语句(GRANT等)和会话控制语句(ALTER SESSION等)。

· 增加程序的灵活性。如在运行时指定模式对象，或为SELECT语句生成不同的WHERE条件查询等。

· 虽然用包DBMS_SQL可以动态执行SQL语句，但DBMS_SQL缺乏对对象和集合支持，要得到更高的性能、更多的函数及集合支持时需要动态SQL语句。

动态SQL语句是由程序在运行时生成的字符串。字符串中包括有效SQL语句的文本或匿名PL/SQL块，其中可以包括用于联编参数的定位符(无需说明其类型的标识符)，它们以冒号开。例如：

```
'DELETE FROM emp WHERE sal > :my_sal AND comm < :my_comm'
```

2 用EXECUTE IMMEDIATE来实现动态SQL

除多行查询外，对于多数动态SQL语句的处理，可用PL/SQL中的语

句EXECUTE IMMEDIATE，它对动态SQL语句或匿名PL/SQL块进行语句分析，然后立即执行。其格式为：

EXECUTE IMMEDIATE dynamic_string [INTO {define_variable,... | record}]

[USING [IN | OUT | IN OUT] bind_argument,...] [RETURNING INTO bind_argument,...];

其中：dynamic_string是由SQL语句或PL/SQL块组成的字符串；define_viariable是存放选定列值的变量；record是用户定义或%ROWTYPE类型的记录，存放选定的行；bind_argument是联编参数，在IN模式时是一个表达式，它的值被传递给SQL语句或PL/SQL块；在OUT模式时是存放动态SQL语句或PL/SQL块返回的值的变量。

除了多行查询，动态串可以包括任何SQL语句（不带终止符）或任何PL/SQL块（带终止符）。如果动态串中有定位符，运行时用联编参数替换动态串中的定位符，因此每个定位符必须在USING或RETURNING INTO子句中有对应的联编参数。

对于单行查询，用INTO子句指定存放返回结果的变量或记录，INTO子句变量个数和类型必须与返回值相同。对有RETURNING子句的DML语句，用RETURNING INTO指定返回值的相应的变量。用EXECUTE IMMEDIATE实现动态SQL的例子：

```
declare
  sql_stmt varchar2(200); -- 变长字符的变量说明,最长200字节
  plsql_block varchar2(500);
  emp_id number(4) := 7566; -- 数值变量说时，缺省值为7566
  salary number(7,2);
  dept_id number(2) := 50;
  dept_name varchar2(14) := 'personnel';
  location varchar2(13) := 'dallas';
  emp_rec emp%rowtype; -- 说明记录类型变量
begin
  /* 用EXECUTE命令建立一个表 */
  execute immediate 'create table bonus (id number, amt number)';
  /* 向表中插入记录，字段值由dept_id, dept_name, location决定 */
  sql_stmt := 'insert into dept values (:1, :2, :3)';
  execute immediate sql_stmt using dept_id, dept_name, location;
  /* 从表中查询指定员工编号的记录 */
  sql_stmt := 'select * from emp where empno = :id';
  execute immediate sql_stmt into emp_rec using emp_id;
  /* 用动态语句调用命令块 */
  plsql_block := 'begin emp_pkg.raise_salary(:id, :amt); end;';
  execute immediate plsql_block using 7788, 500;
  /* 更新表中的记录 */
  sql_stmt := 'update emp set sal = 2000 where empno = :1
  returning sal into :2';
  execute immediate sql_stmt using emp_id returning into salary;
  /* 删除部分编号为dept_id的记录 */
  execute immediate 'delete from dept where deptno = :num'
  using dept_id;
  execute immediate 'alter session set sql_trace true';
end;
```

当动态INSERT、UPDATE或DELETE语句有RETURNING子句时，输出联编参数要出现在RETUNRING INTO子句或USING子句中。

3 用OPEN-FOR来实现动态SQL

如果处理动态多行查询，必须使用OPEN-FOR等游标操作;即首先用OPEN-FOR打开多行查询的游标变量，然后用FETCH从结果集中一次取一行，所有行处理完成后，用CLOSE关闭游标变量。

OPEN-FOR语句把一个游标变量与一个多行查询关联起来、并执行查询、标识结果集、把游标指向结果集的第一行、游标%ROWCOUNT属性值置为零。动态OPEN-FOR可用USING子句，运行时，USING子句中的联编参数替换动态SELECT语句中的定位符。语法是：

OPEN {cursor_variable } FOR dynamic_string [USING bind_argument,...];

其中：cursor_variable是游标变量，dynamic_string是多行查询的动态串。

FETCH {cursor_variable } INTO {define_variable,... | record};

FETCH语句从多行查询的结果集中返回一行，把查询中的值赋给USING子句中相应的变量或列名。游标中的每个列值，在INTO子句必须有相同类型的变量对应。如果从关闭的游标中取值，PL/SQL触发预定义的异常INVALID_CURSOR。

CLOSE (游标变量 | 主机游标变量)关闭游标。

用游标处理多行动态SQL语句的例子：

```
declare
  type empcurtyp is ref cursor; -- 类型定义
```

```
    emp_cv empcurtyp; -- 变量定义
    emp_rec emp%rowtype;
    sql_stmt varchar2(200);
    my_job varchar2(15) := 'clerk';
  begin
    sql_stmt := 'select * from emp where job = :j';
    open emp_cv for sql_stmt using my_job; -- 打开游标
    /* 开始循环 */
    loop
    fetch emp_cv into emp_rec; -- 从游标中读出数据
    exit when emp_cv%notfound; -- 循环结束条件，找不到记录
    ......    - - 对读出的数据进行处理
    end loop;
    close emp_cv;  -- 关闭游标变量
  end;
```

4 块动态SQL

块联编可以提高动态SQL程序性能。因为它们按批量传递数据，所以减少PL/SQL和SQL引擎交换的数量。使用下面命令可以构造块SQL语句，然后在运行时动态执行：BULK FETCH；BULK EXECUTE IMMEDIATE；FORALL；COLLECT INTO子句；RETURNING INTO子句；%BULK_ROWCOUNT游标属性。

块联编将SQL语句中的变量与一组值联系起来。块动态联编使用下面三个语句：EXECUTE IMMEDIATE,FETCH,and FORALL.

EXECUTE IMMEDIATE dynamic_string [[BULK COLLECT] INTO define_variable....] [USING bind_argument,...] [RETURNING BULK COLLECT INTO bind_argument, ...];使用动态多行查询时要用BULK COLLECT INTO子句与联编变量关联。每列的值存放在集合中。使用返回多行的动态INSERT、UPDATE或DELETE时，用RETURNING BULK COLLECT INTO子句与输出变量关联；返回值放在一组集合中。

块FETCH语句可以从动态游标中返回内容：

FETCH dynamic_cursor BULK COLLECT INTO define_variable, ...];

批量FORALL语句让输入联编变量与动态SQL语句关联。在FORALL循环中使用EXECUTE IMMEDIATE语句，语法如下：

FORALL index IN lower bound..upper bound EXECUTE IMMEDIATE dynamic_string USING bind_argument, ... [RETURNING BULK COLLECT INTO bind_argument, ...];

其中动态串必须是INSERT、UPDATE或DELETE，不能是SELECT语句。

块动态SQL的例子：

```
DECLARE
  TYPE EmpCurTyp IS REF CURSOR; -- 类型定义
  TYPE NumList IS TABLE OF NUMBER; -- 表类型定义
  TYPE NameList IS TABLE OF VARCHAR2(15);
  emp_cv EmpCurTyp; -- 游标变量说明
  empnos NumList; -- 说明表类型变量
  enames NameList;
  sals NumList;
BEGIN
  OPEN emp_cv FOR 'SELECT empno, ename FROM emp'; --打开游标
  FETCH emp_cv BULK COLLECT INTO empnos, enames; --块读数据到表中
  CLOSE emp_cv;
  /* 用动态语句执行查询 */
  EXECUTE IMMEDIATE 'SELECT sal FROM emp' BULK COLLECT INTO sals;
  END;
```

5 结论

在存储过程或函数中使用动态SQL语句，将会大大提高程序的性能，并增加其灵活性，同时可以执行各种类型的SQL语句。但在使用动态SQL时，也应该注意以下方面：

- 不能直接用模式对象作为联编参数；
- 动态串中的定位符可以重复；
- 可以使用隐含游标的属性，如SQL%FOUND等；
- 空值NULL传递给联编参数时，必须以空值变量进行；
- 远程过程调用RPC中可以使用动态SQL；
- 避免调用死锁。

参考文献

1 瓮正科，ORACEL8.X For Windows NT实用教程，清华大学出版社。

2 ORACLE公司，PL/SQL User's Guide and Reference,Release9.0.1。