函数和存储过程

对于PLSql程序的使用，如果每次使用都要进行编写，会非常的麻烦，这时可以采用过程和函数来命名PLSQL程序，被编译后存储到数据库中，以备后续使用。

过程和函数统称为PL/SQL子程序，他们是被命名的PL/SQL块，均存储在数据库中，并通过输入、输出参数或输入/输出参数与其调用者交换信息。过程和函数的唯一区别是函数总向调用者返回数据，而过程则不返回数据。

一、函数

1.1 函数分类

- 系统函数
  - decode，to_char...
- 自定义函数

1.2 创建

Oracle函数包含三部分：函数声明，执行部分，返回的参数

简单的基本结构可以使用工具PL\SQL创建，File->New->Program Window->Function。输入函数名，以及参数，就能自动生成。

基本语法

```
-- 函数
/*
CREATE OR REPLACE FUNCTION 函数名(参数)
RETURN 返回类型
AS
声明部分
BEGIN
  执行部分
  RETURN 返回的数据;
END;
*/
```

示例

```
-- 不带参数的
CREATE OR REPLACE FUNCTION fn_getScore
RETURN VARCHAR2  -- 不带长度
AS
scores score.score%TYPE;
BEGIN
  -- S001学生1期正式考试的成绩
  SELECT score INTO scores FROM(
         SELECT score FROM score WHERE stuno='S001' AND cid=1 ORDER BY uptime
  ) WHERE ROWNUM=1;
  RETURN scores;
END;
```

1.3 调用

```
-- 查询调用
SELECT fn_getScore() FROM dual;
-- 匿名块调用
DECLARE
  scores score.score%TYPE;
BEGIN
  scores:=fn_getScore();
  dbms_output.put_line(scores);
END;
```

1.4 参数类型

在函数中可以使用IN/OUT来标识参数类型，IN为输入参数，OUT表示输出参数，不写时默认为IN。

参数模式：IN，OUT，IN OUT。

- IN，就是从调用环境通过参数传入值，在过程中只能被读取，不能改变
- OUT，由过程赋值并传递给调用环境。不能是具有默认值的变量，也不能是常量，过程中要给OUT参数传递返回值
- IN OUT，具有IN参数和OUT 参数两者的特性，在过程中即可传入值，也可传出值。

IN

```
-- 带输入参数
CREATE OR REPLACE FUNCTION fn_getScore2(sno VARCHAR2,ids NUMBER)
RETURN VARCHAR2  -- 不带长度
AS
scores score.score%TYPE;
BEGIN
  -- S001学生1期正式考试的成绩
  SELECT score INTO scores FROM(
         SELECT score FROM score WHERE stuno=sno AND cid=ids ORDER BY uptime
  ) WHERE ROWNUM=1;
  RETURN scores;
END;

SELECT fn_getScore2('S002',3) FROM dual;
```

OUT

```
-- 带输出参数
CREATE OR REPLACE FUNCTION fn_getScore3(sno VARCHAR2,ids NUMBER,rs OUT VARCHAR2)
RETURN VARCHAR2  -- 不带长度
AS
scores score.score%TYPE;
BEGIN
  SELECT score INTO scores FROM(
         SELECT score FROM score WHERE stuno=sno AND cid=ids ORDER BY uptime
DESC
  ) WHERE ROWNUM=1;

  IF scores>=60 THEN
    rs:='通过';
  ELSE
    rs:='不通过';
  END IF;
  RETURN scores;
END;

-- 调用
DECLARE
   rs VARCHAR2(10);
   sc VARCHAR2(10);
BEGIN
  sc:=fn_getScore3(rs=>rs,ids=>1,sno=>'S002');
  dbms_output.put_line(sc||rs);
END;
```

IN OUT

当输入参数和输出参数数据类型一致时，可以只使用一个参数表示，该参数在函数执行前作为输入参数向函数传递数据，函数执行结束后作为输出参数向外传递数据信息

```
-- 输入输出参数：in out
CREATE OR REPLACE FUNCTION fn_getScore4(str IN OUT VARCHAR2,ids NUMBER)
RETURN VARCHAR2  -- 不带长度
AS
scores score.score%TYPE;
BEGIN
  SELECT score INTO scores FROM(
         SELECT score FROM score WHERE stuno=str AND cid=ids ORDER BY uptime
DESC
  ) WHERE ROWNUM=1;

  IF scores>=60 THEN
    str:='通过';
  ELSE
    str:='不通过';
  END IF;
  RETURN scores;
END;

-- 调用
DECLARE
   str VARCHAR2(10);
   rs VARCHAR2(10);
```

```
BEGIN
  str:='S001';
  dbms_output.put_line('调用前:'||str);
  rs:=fn_getScore4(str,1);
  dbms_output.put_line('调用后:'||str);
  dbms_output.put_line('调用后rs:'||rs);
END;
```

二、过程

2.1 创建

```
-- 过程
/*
1.参数与变量的名称不能一致
*/
/*
CREATE OR REPLACE PROCEDURE 过程名(参数1,参数2...)
AS
声明部分
BEGIN
  执行部分
  EXCEPTION
    异常部分
END;
*/
-- 无参过程:1.参数名后面的()不写
CREATE OR REPLACE PROCEDURE pro_course
AS
BEGIN
  INSERT INTO course VALUES(5,'四期');
  COMMIT;
END;
```

2.2 调用删除

```
-- 调用
-- 1.命令窗口使用
-- exec pro_stu();
-- 2.call：必须添加括号
CALL pro_stu();
-- 3.匿名块
BEGIN
  pro_stu();
END;

-- 删除
DROP PROCEDURE pro_stu;
```

2.3 IN

```
-- 带输入参数:in 可以省略
```

```
-- 1.sno student.stuno%TYPE
-- 2.参数类型不带长度
CREATE OR REPLACE PROCEDURE pro_course1(sno IN VARCHAR2)
AS
 sname student.stuname%TYPE;
BEGIN
  SELECT stuname INTO sname FROM student WHERE stuno=sno;
  dbms_output.put_line('stuname:'||sname);

  EXCEPTION
    WHEN no_data_found THEN
      dbms_output.put_line('查询的学生不存在');
    WHEN OTHERS THEN
      dbms_output.put_line('程序错误');
END;

CALL pro_course1('S001');
```

2.4 OUT

```
-- 带输出参数
CREATE OR REPLACE PROCEDURE pro_course2(sno IN VARCHAR2,names OUT VARCHAR2)
AS
BEGIN
  names:='未知';
  SELECT stuname INTO names FROM student WHERE stuno=sno;

  EXCEPTION
    WHEN no_data_found THEN
      dbms_output.put_line('查询的学生不存在');
    WHEN OTHERS THEN
      dbms_output.put_line('程序错误');
END;

DECLARE
   names VARCHAR2(10);
BEGIN
  pro_course2('S001',names);
  dbms_output.put_line('names:'||names);
END;
```

2.5 IN OUT

```
-- 输入输出参数:in out
CREATE OR REPLACE PROCEDURE pro_course3(str IN OUT VARCHAR2)
AS
BEGIN
  dbms_output.put_line('str:'||str);
  SELECT stuname INTO str FROM student WHERE stuno=str;
  dbms_output.put_line('str:'||str);

  EXCEPTION
    WHEN no_data_found THEN
      dbms_output.put_line('查询的学生不存在');
    WHEN OTHERS THEN
```

```
      dbms_output.put_line('程序错误');
END;

DECLARE
  STR VARCHAR2(10);
BEGIN
  STR:='S001';
  dbms_output.put_line('调用前str:'||str);
  pro_course3(STR);
  dbms_output.put_line('调用后str:'||str);
END;
```

2.6 输入参数赋值

```
CREATE OR REPLACE PROCEDURE PRO_COURSE4(SNAME  STUDENT.STUNAME%TYPE,
                                        CLSID  NUMBER,
                                        STATES NUMBER) AS
  STR VARCHAR2(10);
BEGIN
  SELECT STUNAME INTO STR FROM STUDENT
  WHERE STUNAME = SNAME AND CLASSID = CLSID AND STATE = STATES;
  DBMS_OUTPUT.PUT_LINE('str:' || STR);

EXCEPTION
  WHEN NO_DATA_FOUND THEN
    DBMS_OUTPUT.PUT_LINE('查询的学生不存在');
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('程序错误');
END;

-- 1.按参数顺序直接赋值;
PRO_COURSE4('赵雷',1005,1);
-- 2.按命名赋值：参数名=>值
PRO_COURSE4(CLSID=>1005,STATES=>1,SNAME=>'赵雷');
-- 3.混合使用：先按顺序,再按命名
PRO_COURSE4('赵雷',STATES=>1,CLSID=>1005);
```

三、包

```
/*
1.包规范中只做定义,
2.包体中做具体的实现,必须实现包规范中定义的所有的函数和过程;
3.包规范中定义的是公有内容,可以在外部进行调用;
4.包规范中定义的常量和变量,在包体中可以直接使用;
5.包体中可以有私有的变量和常量，函数，过程,在外部不能进行调用,在包体内可以;
6.包体的名称与包规范对应,一个包规范对应一个包体;
*/

-- 包规范
CREATE OR REPLACE PACKAGE pck_package1
AS
  pai CONSTANT NUMBER(5,4):=3.1415;
```

```
  r NUMBER(10);
  h NUMBER(10);

  -- 函数,求面积
  FUNCTION getMinji RETURN NUMBER;

  -- 过程,求圆柱体积
  PROCEDURE getTiji;
END pck_package1;

-- 包体
CREATE OR REPLACE PACKAGE BODY pck_package1
AS
  tiji NUMBER(10);

  -- 求球形的体积
  FUNCTION getTiji2 RETURN NUMBER
  AS
  BEGIN
    RETURN 4/3*pai*r*r*r;
  END;

  -- 求圆的面积
  FUNCTION getMinji RETURN NUMBER
  AS
  BEGIN
    RETURN pai*r*r;
  END;

  -- 求圆柱的体积
  PROCEDURE getTiji
  AS
  BEGIN
    tiji:=getTiji2(); -- 调用函数
    dbms_output.put_line('圆柱体积：'||pai*r*r*h);
    dbms_output.put_line('球形体积：'||tiji);
  END;
END pck_package1;
```

3.1 调用

```
-- 1.调用
SELECT pck_package1.getMinji FROM dual;

CALL pck_package1.getTiji();

-- 2.包名.对象
BEGIN
  -- 调用变量，常量
  dbms_output.put_line('圆周率：'||pck_package1.pai);
  pck_package1.r:=10;
  dbms_output.put_line('圆的半径为：'||pck_package1.r);

  -- 调用函数
  dbms_output.put_line('圆的面积：'||pck_package1.getMinji());
```

```
  -- 调用过程
  pck_package1.h:=10;
  pck_package1.getTiji();
END;
```

3.2 删除

```
-- 删除：删除包规范,包体删除;删除包体,包规范还在
DROP PACKAGE pck_package1;
DROP PACKAGE BODY pck_package;
```

四、视图

```
-- 视图:虚表,没有实际数据
-- 试图可以做增删改,在源表中可以找到唯一的记录做匹配
CREATE VIEW v_stu
AS
SELECT student.*,1 ones,2 twos FROM student;

SELECT * FROM v_stu;
INSERT INTO v_stu(stuno,stuname,phone,classid,state,IDCARD)
VALUES('S999','张鑫',123456,1003,0,'411326199703226l8X');

UPDATE v_stu SET phone=1111111 WHERE stuno='S999';

DELETE v_stu WHERE stuno='S999';

CREATE VIEW v_score
AS
SELECT stuno,cid,MAX(score) score FROM score GROUP BY stuno,cid ORDER BY
stuno,cid;


SELECT * FROM v_score;
INSERT INTO v_score VALUES('S011',1,100);
```

五、总结

FUNCTION

- 函数用于计算和返回一个结果值，把经常需要进行的计算写成函数,函数的调用是表达式的一部分。
- 只在创建时进行编译，以后每次执行函数都不需要重新编译。
- 函数必须有一个返回值，而过程没有做强制的规定。
- RETURN在声明部分需要定义一个返回参数的类型，而在函数体中必须有一个RETURN语句。如果函数结束时还没有遇到返回语句，就会发生错误。

PROCEDURE

- 只在创建时进行编译，以后每次执行存储过程都不需再重新编译。

- 当对数据库进行复杂操作时，可将此复杂操作用存储过程封装起来与数据库提供的事务处理结合一起使用。

PACKAGE

- PACKAGE可以将存储过程分门别类，且可以在包内定义公共变量及类型，供包内存过使用

区别

- 标识符不同。函数的标识符为FUNCTION，过程为：PROCEDURE。
- 有无返回值不同。函数中有且必须有一个返回值，而过程中没有返回值。
- 类型说明不同。因为函数需要返回值所以需要给函数设置类型说明，但过程中没有。
- 调用方式不同。函数的调用可以出现在表达式中，但过程必须单独调用
- PACKAGE则与FUNCTIONS和PROCEDURES可以说完全没有关系，他只是函数与存过的一个集合容器