oracle存储过程入门详解（oracle存储过程语法）

作者：有故事的人 来源：范文网 www.wtabcd.cn/fanwen/

本文原地址：https://www.wtabcd.cn/fanwen/zuowen/8c9288f961db79f351349fb1ea5c58a5.html

教学大纲：

plsql编程：hello world、程序结构、变量、流程控制、游标.存储过程：概念、无参存储、有参存储（输入、输出）.java调用存储存储过程.

1. plsql编程

1.1. 概念和目的

什么是pl/sql?

pl/sql（procedure language/sql）plsql是oracle对sql语言的过程化扩展 (类似于basic)指在sql命令语言中增加了过程处理语句（如分支、循环等），使sql语言具有过程处理能力。

1.2. 程序结构

通过plsql developer工具的test window 创建 程序模版或者通过语句在sql window编写

提示：plsql语言的大小写是不区分的

pl/sql可以分为三个部分：声明部分、可执行部分、异常处理部分。

```
-- created on 2018/3/21 by administrator declare  -- 声明变量、游标。  i integer;b
egin  -- 执行语句  --[异常处理]end;
```

其中 declare部分用来声明变量或游标（结果集类型变量），如果程序中无变量声明可以省略掉

1.3. hello world

```
begin  --打印hello world  dbms_output.put_line('hello world');end;
```

其中dbms_output 为oracle内置程序包，相当于java中的system.out,而put_line（）是调用的方法,相当于println（）方法

在sqlplus中也可以编写运行plsql程序：

sql> begin 2 3 --打印hello world 4 5 dbms_outp建行网银ut.put_
line('hello world'); 6 7 end; 8 /pl/sql 过程已成功完成。

执行结束后并未显示输出的结果，默认情况下，输出选项是关闭状态的 我们需要开启一下 **set serveroutput on**

sqlplus中执行plsql程序 需要在程序最后添加一个 / 标识程序的结束

1.4. 变量

plsql编程中常见的变量分两大类：

普通数据类型（char,varchar2, date, number, boolean,
long）特殊变量类型（引用型变量、记录型变量）

声明变量的方式为

变量名 变量类型（变量长度） 例如： v_name varchar2(20);

1.4.1. 普通变量

变量赋值的方式有两种：

直接赋值语句 := 比如: v_name := ‘ zhangsan ’语句赋值，使用select …into … 赋值：（语法 select 值 into 变量）

【示例】打印人员个人信息，包括： 姓名、薪水、地址

-- 打印人员个人信息，包括： 姓名、薪水、地址raisedeclare -- 姓名 v_name varchar
2(20) := '张三'; -- 声明变量直接赋值 --薪水 v_sal number; --地址 v_addr
varchar2(200);begin --在程序中直接赋值 v_sal := 1580; --语句赋值 select
'上海市传智播客' into v_addr from dual; --打印变量 dbms_output.put_line('姓名
：' || v_name || ',薪水：' || v_sal || ',地址：' ||v_addr);end;

1.4.2. 引用型变量

变量的类型和长度取决于表中字段的类型和长度

通过**表名.列名%type**指定变量的类型和长度，例如： v_name emp.ename%type;

【示例】查询emp表中7839号员工的个人信息，打印姓名和薪水

```
-- 查询emp表中7839号员工的个人信息，打印姓名和薪水declare  -- 姓名  v_name emp.
ename%type; -- 声明变量直接赋值  --薪水  v_sal emp.sal%type;begin  --查询表中
的姓名和薪水并赋值给变量  --注意查询的字段和赋值的变量的顺序、个数、类型要一致
select ename, sal into v_name, v_sal from emp where empno = 7839;  --打印变
量  dbms_output.put_line('姓名：' || v_name || ',薪水：' || v_sal);end;
```

引用型变量的好处：

使用普通变量定义方式，需要知道表中列的类型，而使用引用类型，不需要考虑列的类型，使用%type是非常好的编程风格，因为它使得pl/sql更加灵活，更加适应于对数据库定义的更新。

1.4.3. 记录型变量

接受表中的一整行记录，相当于java中的一个对象

语法： 变量名称 表名%rowtype， 例如： **v_emp emp%rowtype;**

【示例】

查询并打印7839号员工的姓名和薪水

```
-- 查询emp表中7839号员工的个人信息，打印姓名和薪水declare  -- 记录型变量接受一行
 v_emp emp%rowtype;begin  --记录型变量默认接受表中的一行数据，不能指定字段。
 select * into v_emp from emp where empno = 7839;  --打印变量，通过变量名
.属性的方式获取变量中的值  dbms_output.put_line('姓名：' || v_emp.ename || ',薪水：'
 || v_emp.sal);end;
```

如果有一个表，有100个字段，那么你程序如果要使用这100字段话，如果你使用引用型变量一个个声明，会特别麻烦，记录型变量可以方便的解决这个问题

错误的使用：

1. 记录型变量只能存储一个完整的行数据

2. 返回的行太多了，记录型变量也接收不了

1.5. 流程控制

1.5.1. 条件分支

语法：

begin if 条件1 then 执行1 elsif 条件2 then 执行 2 else 执行3 end if; end;

注意关键字：elsif

【示例】判断emp表中记录是否超过20条，10-20之间，或者10条以下

declare --声明变量接受emp表中的记录数 v_count number;begin --查询emp表中的记录数赋值给变量 select count(1) into v_count from emp; --判断打印 if v_count > 20 then dbms_output.put_line('emp表中的记录数超过了20条为：' || v_count || '条。'); elsif v_count >= 10 then dbms_output.put_line('emp表中的记录数在10~20条之间为：' || v_count || '条。'); else dbms_output.put_line('emp表中的记录数在10条以下为：' || v_count || '条。'); end if;end;

1.5.2. 循环

在oracle中有三种循环方式，这里我们不展开,只介绍其中一种：loop循环

语法：

begin loop exit when 退出循环条件 end loop;end;

【示例】打印数字1-10

declare --声明循环变量并赋初值 v_num number := 1;begin loop exit when v_num > 10; dbms_output.put_line(v_num); --循环变量自增 v_num := v_num + 1; end loop;end;

2. 游标

2.1. 什么是游标

用于临时存储一个查询返回的多行数据（结果集,类似于java的jdbc连接返回的resultset集合），通过遍历游标，可以逐行访问处理该结果集的数据。

游标的使用方式：声明—>打开—>读取—>关闭

2.2. 语法

游标声明：

cursor 游标名[(参数列表)] is 查询语句;

游标的打开：

open 游标名;

游标的取值：

fetch 游标名 into 变量列表;

游标的关闭：

close 游标名;

2.3. 游标的属性

游标的属性返回值类型说明

其中 %notfound是在游标中找不到元素的时候返回true,通常用来判断退出循环

2.4. 创建和使用

【示例】使用游标查询emp表中所有员工的姓名和工资，并将其依次打印出来。

```
--使用游标查询emp表中所有员工的姓名和工资，并将其依次打印出来。declare --声明游标
 cursor c_emp is  select ename, sal from emp; --声明变量用来接受游标中
的元素 v_ename emp.ename%type; v_sal emp.sal%type;begin --打开游标 o
pen c_emp; --遍历游标中的值 loop  --通过fetch语句获取游标中的值并赋
值给变量  fetch c_emp  into v_ename, v_sal;  --通过%notfoun
d判断是否有值,有值打印,没有则退出循环  exit when c_emp%notfound;  db
ms_output.put_line('姓名:' || v_ename || ',薪水:' || v_sal);  end loop; --关闭
游标 close c_emp;end;
```

执行结果:

2.5. 带参数的游标

【示例】使用游标查询并打印某部门的员工的姓名和薪资，部门编号为运行时手动输入。

```
--使用游标查询并打印某部门的员工的姓名和薪资，部门编号为运行时手动输入。declare -
-声明游标传递参数 cursor c_emp(v_empno emp.empno%type) is  select ename, s
al from emp where empno = v_empno; --声明变量用来接受游标中的元素 v_en
ame emp.ename%type; v_sal emp.sal%type;begin --打开游标并传递参数 open
```

```
c_emp(10); --遍历游标中的值 loop      --通过%notfound判断是
否有值,有值打印,没有则退出循环      exit when c_emp%notfound;
     --通过fetch语句获取游标中的值并赋值给变量    fetch c_emp
 into v_ename, v_sal;    dbms_output.put_line('姓名:' || v_ename || ',薪水:' || v
_sal); end loop; --关闭游标 close c_emp;end;
```

注意:%notfound属性默认值为flase,所以在循环中要注意判断条件的位置.如果先判断在fetch会导致最后一条记录的值被打印两次(多循环一次默认);

3. 存储过程

3.1. 概念作用

之前我们编写的plsql程序可以进行表的操作,判断,循环逻辑处理的工作,但无法重复调用.

可以理解之前的代码全都编写在了main方法中,是匿名程序.
java可以通过封装对象和方法来解决复用问题

plsql是将一个个plsql的业务处理过程存储起来进行复用,这些被存储起来的plsql程序称之为存储过程

存储过程作用：

1， 在开发程序中，为了一个特定的业务功能，会向数据库进行多次连接关闭(连接和关闭是很耗费资源), 需要对数据库进行多次i/o读写，性能比较低。如果把这些业务放到plsql中，在应用程序中只需要调用plsql就可以做到连接关闭一次数据库就可以实现我们的业务,可以大大提高效率.

2， oracle官方给的建议：能够让数据库操作的不要放在程序中。在数据库中实现基本上不会出现错误，在程序中操作可能会存在错误.(如果在数据库中操作数据,可以有一定的日志恢复等功能.)

3.2. 语法

create or replace procedure 过程名称[(参数列表)] isbeginend 文科和理科[过程名称];

根据参数的类型，我们将其分为3类讲解：

l 不带参数的

l 带输入参数的

l 带输入输出参数(返回值)的。

3.3. 无参存储

3.3.1. 创建存储

通过plsql developer或者语句创建存储过程:

【示例】 通过调用存储过程打印hello world

创建存储过程:

--通过调用存储过程打印hello worldcreate or replace procedure p_hello isbegin dbms_output.put_line('hello world');end p_hello;

通过工具查看创建好的存储过程:

3.3.2. 调用存储过程

1.通过plsql程序调用：

begin --直接输入调用存储过程的名称 p_hello;end p_hello;

2.在sqlplus中通过exec命令调用:

提示:sqlplus中显示结果的前提是需要 set serveroutput on

注意：

第一个问题：is和as是可以互用的，用哪个都没关系的
第二个问题：过程中没有declare关键字，declare用在语句块中

3.4. 带输入参数的存储过程

【示例】查询并打印某个员工（如7839号员工）的姓名和薪水 – 存储过程：要求，调用的时候传入员工编号，自动控制台打印。

--查询并打印某个员工（如7839号员工）的姓名和薪水--要求，调用的时候传入员工编号，自动控制台打印。create or replace procedure p_querynameandsal(i_empno in emp.empno%type) is --声家用电器说明书明变量接受查询结果 v_ename emp.ename%type; v_sal emp.sal%type;begin --根据用户传递的员工号查询姓名和薪水 select ename, sal into v_ename, v_sal from emp where empno = i_empno; --打印结果 dbms_output.put_line('姓名:' || v_ename || ',薪水:' || v_sal);end p_querynameandsal;

命令调用：

sql> exec p_querynameandsal(7839);姓名:king,薪水:5000pl/sql 过程已成功完成。

plsql程序调用：

begin p_querynameandsal(7839);end;

执行结果:

3.5. 带输出参数的存储过程

【示例】输入员工号查询某个员工（7839号员工）信息，要求，将薪水作为返回值输出，给调用的程序使用。

--输入员工号查询某个员工（7839号员工）信息，要求，将薪水作为返回值输出，给调用的程序使用。create or replace procedure p_querysal_out(i_empno in emp.empno%type,o_sal o
ut emp.sal%type) isbegin select sal into o_sal from emp where empno = i_e
mpno;end p_querysal_out;

plsql程序调用：

declare --声明一个变量接受存储过程的输出参数 v_sal emp.sal%type;begin p_quer
ysal_out(7839, v_sal); --注意参数的顺序 dbms_output.put_line(v_sal);end;

注意:调用的时候,参数要与定义的参数的顺序和类型一致江西学籍管理系统入口.

3.7. java程序调用存储过程

需求:如果一条语句无法实现结果集,
比如需要多表查询,或者需要复杂逻辑查询,我们可以选择调用存储查询出你的结果.

3.7.1. 分析jdk api

通过connection对象的preparecall方法可以调用存储过程

得出结论: 通过connection对象调用preparecall方法传递一个转义sql语句调用存储过程, 输入参数直接调用set方法传递.输出参数需要注册后,执行存储过程,通过get方法获取.参数列表的下标是从1开始

3.7.2. 实现代码

准备环境：

l 导入oracle的jar包【示例】通过员工号查询员工的姓名和薪资package cn.itcast.oracle.jdbc;i
mport oracle.jdbc.oracletypes;import java.sql.callablestatement;import java.sql.connection;import ja
va.sql.drivermanager;public class proceduretest { public static void main(string[] a
rgs) throws exception { //1.加载驱动 class.forname("oracle.jd
bc.driver.oracledriver"); //2.获得连接对象 //2.1 设置连接
字符串 string url ="jdbc:oracle:thin:@localhost:1521:xe" ; s
tring name = "scott"; string password = "tiger"; con
nection conn = drivermanager.getconnection(url, name, password); //3.
获取语句对象 string sql = "{call p_querysal_out(?,?)}";//转义语法,{call
存储过程(参数列表)} callablestatement call = conn.preparecall(sql);
 //4.设置输入参数 call.setint(1,7839); //5.注册输出
参数 call.registeroutparameter(2, oracletypes.double); //6.执行
存储过程 call.execute(); //7.获取输出参数
double sal = call.getdouble(2); system.out.println("薪水:"+sal);
 //8.释放资源 call.close(); conn.close(); }}