【济南中心】JavaEE就业班同步笔记第二阶段：Oracle-part04

小鲁哥哥 • 2017-5-26 21:35:11

【济南中心】JavaEE就业班同步笔记第二阶段：Oracle-part04

第一节

1.1 PLSQL概述

1.1.1知识概述

概述

PL/SQL也是一种程序语言，叫做过程化SQL语言（Procedural Language/SQL）。PL/SQL是Oracle数据库对SQL语句的扩展。在普通SQL语句的使用上增加了编程语言的特点，所以PL/SQL就是把数据操作和查询语句组织在PL/SQL代码的过程性单元中，通过逻辑判断、循环等操作实现复杂的功能或者计算的程序语言。

是Oracle对标准数据库语言SQL的过程化扩充，它将数据库技术和过程化程序设计语言联系起来，是一种应用开发语言，可使用循环，分支处理数据，将SQL的数据操纵功能与过程化语言数据处理功能结合起来.PL/SQL的使用，使SQL成为一种高级程序设计语言，支持高级语言的块操作，条件判断，循环语句，嵌套等，与数据库核心的数据类型集成，使SQL 的程序设计效率更高.

作用

使用PL/SQL可以编写具有很多高级功能的程序，虽然通过多个SQL语句可能也能实现同样的功能，但是相比而言，PL/SQL具有更为明显的一些优点：

1.能够使一组SQL语句的功能更具模块化程序特点；

2.采用了过程性语言控制程序的结构；

3.可以对程序中的错误进行自动处理，使程序能够在遇到错误的时候不会被中断；

4.具有较好的可移植性，可以移植到另一个Oracle数据库中；

5.集成在数据库中，调用更快；

6.减少了网络的交互，有助于提高程序性能。

语法

```
DECLARE --可选部分
--变量、常量、游标、用户定义异常的声明

BEGIN --必要部分
--SQL语句和PL/SQL语句构成的执行程序

EXCEPTION --可选部分
--程序出现异常时，捕捉异常并处理异常

END; --必须部分
```

1.2 PLSQL-变量声明与赋值

1.2.1 知识概述

声明的语法：

```
--变量名 类型（长度）;
age number;--年龄
--变量名:=变量值;
age:=20;--年龄赋值为20
```

1.3 PLSQL-select into 变量赋值

1.3.1 知识概述

语法和使用

```
--select 列名  into  变量名 from 表名 where 条件
select student_age into age from student_info where student_id='20';
```

```
declare
sname varchar2(50);
sgender varchar2(3);
begin
select student_name,student_gender into sname,sgender from student_info where student_id='1';
dbms_output.put_line(sname ||'---'||sgender);
end;
```

注意：select语句查询到的结果必须是一条记录

第二节

2.1 PLSQL-异常处理

2.1.1知识概述

2.1.1.1预定义异常:

当 PL/SQL 程序违反 Oracle 规则或超越系统限制时隐式引发。

命名的系统异常	产生原因
ACCESS_INTO_NULL	未定义对象
CASE_NOT_FOUND	CASE 中若未包含相应的 WHEN ，并且没有设置 ELSE 时
COLLECTION_IS_NULL	集合元素未初始化
CURSER_ALREADY_OPEN	游标已经打开
DUP_VAL_ON_INDEX	唯一索引对应的列上有重复的值
INVALID_CURSOR	在不合法的游标上进行操作
INVALID_NUMBER	内嵌的 SQL 语句不能将字符转换为数字
NO_DATA_FOUND	**使用 select into 未返回行**
TOO_MANY_ROWS	**执行 select into 时，结果集超过一行**
ZERO_DIVIDE	除数为 0
SUBSCRIPT_BEYOND_COUNT	元素下标超过嵌套表或 VARRAY 的最大值
SUBSCRIPT_OUTSIDE_LIMIT	使用嵌套表或 VARRAY 时，将下标指定为负数
VALUE_ERROR	赋值时，变量长度不足以容纳实际数据
LOGIN_DENIED	PL/SQL 应用程序连接到 oracle 数据库时，提供了不正确的用户名或密码
NOT_LOGGED_ON	PL/SQL 应用程序在没有连接 oralce 数据库的情况下访问数据
PROGRAM_ERROR	PL/SQL 内部问题，可能需要重装数据字典& pl./SQL 系统包
ROWTYPE_MISMATCH	宿主游标变量与 PL/SQL 游标变量的返回类型不兼容
SELF_IS_NULL	使用对象类型时，在 null 对象上调用对象方法
STORAGE_ERROR	运行 PL/SQL 时，超出内存空间
SYS_INVALID_ID	无效的 ROWID 字符串
TIMEOUT_ON_RESOURCE	Oracle 在等待资源时超时

2.1.1.2用户自定义异常

用户可以在 PL/SQL 块的声明部分定义异常，自定义的异常通过 RAISE 语句显式引发。

```
DECLARE --可选部分
--变量、常量、游标、用户定义异常的声明
test_exception exception;
BEGIN --必要部分
--SQL语句和PL/SQL语句构成的执行程序
raise test_exception;
EXCEPTION --可选部分
--程序出现异常时，捕捉异常并处理异常
  --when 异常类型 then 异常处理逻辑
  when test_exception then dbms_output.put_line('出现异常');

END;--必须部分
```

2.2 PLSQL-条件语句

2.2.1知识概述

语法：

```
--语法一
if 条件 then
   业务逻辑
end if;
--语法二
if 条件 then
   业务逻辑
else
   业务逻辑
end if;
--语法三
if 条件 then
   业务逻辑
elsif 条件 then
   业务逻辑
else
   业务逻辑
end if;
```

```
DECLARE --可选部分
--变量、常量、游标、用户定义异常的声明
--&地址符，执行时会提示输入值
sage number:=&age;
BEGIN --必要部分
--SQL语句和PL/SQL语句构成的执行程序
      if sage=1 then
         dbms_output.put_line('111');
      else
         dbms_output.put_line('222');
      end if;
END;--必须部分
```

2.3 PLSQL-循环语句

2.3.1知识概述

1、loop循环

```
declare
v_num number:=1;
begin
  loop
    dbms_output.put_line(v_num);
    v_num:=v_num+1;
    exit when v_num>100;
  end loop;
end ;
```

2、while循环

```
declare
v_num number:=1;
begin
  while v_num<=100
  loop
    dbms_output.put_line(v_num);
    v_num:=v_num+1;
  end loop;
end ;
```

3、for循环

```
begin
    for v_num in 1..100
    loop
        dbms_output.put_line(v_num);
    end loop;
end;
```

注意：for循环中变量不需要声明

第三节

3.1 PLSQL-游标概述和语法

3.1.1知识概述

3.1.1.1概述

游标是系统为用户开设的一个数据缓冲区,存放SQL语句的执行结果。我们可以把游标理解为PL/SQL中的结果集。

3.1.1.2 语法

```
declare
  cursor 游标名称 is SQL语句;
begin
  open 游标名称
  loop
     fetch 游标名称 into 变量
     exit  when  游标名称%notfound
  end loop;
  close 游标名称;
end;
```

3.2 PLSQL-游标案例

3.2.1知识概述

```
declare
  v_pricetable T_PRICETABLE%rowtype;--价格行对象
  cursor cur_pricetable is select * from T_PRICETABLE where ownertypeid=1;--定义游标
begin
  open cur_pricetable;--打开游标
  loop
    fetch cur_pricetable into v_pricetable;--提取游标到变量
    exit when cur_pricetable%notfound;--当游标到最后一行下面退出循环
    dbms_output.put_line( '价格:'
       ||v_pricetable.price ||'吨位: '||v_pricetable.minnum||'-'||v_pricetable.maxnum );
  end loop;
  close cur_pricetable;--关闭游标
end ;
```

1、声明游标

2、打开游标

3、提取游标的值赋给变量

4、退出条件

3.3 PLSQL-带参数的游标

3.3.1知识概述

```
declare
  v_pricetable T_PRICETABLE%rowtype;--价格行对象
  cursor cur_pricetable(v_ownertypeid number) is select * from T_PRICETABLE where ownertypeid=v_ownertypeid;--定义游标
begin
  open cur_pricetable(2);--打开游标
  loop
    fetch cur_pricetable into v_pricetable;--提取游标到变量
    exit when cur_pricetable%notfound;--当游标到最后一行下面退出循环
    dbms_output.put_line('价格:'||v_pricetable.price ||'吨位: '||v_pricetable.minnum||'-'||v_pricetable.maxnum );
  end loop;
  close cur_pricetable;--关闭游标
end ;
```

在声明游标的时候定义传入参数变量和类型

打开游标时传入实际的参数值

3.4 PLSQL-for循环取游标的值

3.4.1 知识概述

```
declare
  cursor cur_pricetable(v_ownertypeid number) is select * from T_PRICETABLE where ownertypeid=v_ownertypeid;--定义游标
begin
  for v_pricetable in cur_pricetable(3)
  loop
    dbms_output.put_line('价格:'||v_pricetable.price ||'吨位: '||v_pricetable.minnum||'-'||v_pricetable.maxnum );
  end loop;
end ;
```

第四节

4.1 存储函数的语法和应用

4.1.1知识概述

概述:

存储函数又称为自定义函数。可以接收一个或多个参数，返回一个结果。在函数中我们可以使用PL/SQL进行逻辑的处理

语法:

```
CREATE [ OR REPLACE ]  FUNCTION  函数名称
  (参数名称 参数类型, 参数名称 参数类型, ...)
RETURN 结果变量数据类型
IS
  变量声明部分;
BEGIN
  执行部分;
  RETURN 结果变量;
[EXCEPTION
  异常处理部分]
END;
```

回复帖子... 回帖

应用:

```
create function fn_getaddress(v_id number)
return varchar2
is
  v_name varchar2(30);
begin
  select name into v_name from t_address where id=v_id;
  return v_name;
end;
```

4.2 存储过程概述和语法结构

4.2.1知识概述

概述:

存储过程（Stored Procedure）是在大型数据库系统中，一组为了完成特定功能的SQL 语句

集，存储在数据库中，经过第一次编译后再次调用不需要再次编译，用户通过指定存储过程的名字并给出参数（如果该存储过程带有参数）来执行它。存储过程是数据库中的一个重要对象。

与存储函数的区别：

存储函数中有返回值，且必须返回；而存储过程没有返回值，可以通过传出参数返回多个值。存储函数可以在select语句中直接使用，而存储过程不能。过程多数是被应用程序所调用；存储函数一般都是封装一个查询结果，而存储过程一般都封装一段事务代码。

语法：

```
CREATE [ OR REPLACE ]  PROCEDURE    存储过程名称
  （参数名 类型，参数名 类型，参数名 类型）
IS|AS
  变量声明部分;
BEGIN
  逻辑部分
[EXCEPTION
  异常处理部分]
END;
```

参数类型只需指定类型，不用指定长度。

参数默认是输入参数（in），可以使输出参数（out），也可以是输入输出参数（in out）

4.3 存储过程-不带输出参数

4.3.1知识概述

```
--增加业主信息序列
create sequence seq_owners start with 11;
--增加业主信息存储过程
create or replace procedure pro_owners_add
(
  v_name varchar2,
  v_addressid number,
  v_housenumber varchar2,
  v_watermeter varchar2,
  v_type number
)
is

begin
  insert into T_OWNERS values(  seq_owners.nextval,v_name,v_addressid,v_housenumber,v_watermeter,sysdate,v_type );
  commit;
end;
```

plsql 存储过程调用

```
call pro_owners_add('赵伟',1,'999-3','132-7',1);
```

4.4 使用JDBC调用不带输出参数的存储过程

4.4.1知识概述

```
/**
 * 增加
 *
 * @param owners
 */
public static void add(Owners owners) {

    java.sql.Connection conn = null;
    java.sql.CallableStatement stmt = null;

    try {
        conn = DaoUtil.getConnection();
        //语句的写法与普通的sql语句不同
        stmt = conn.prepareCall("{call pro_owners_add(?,?,?,?,?)}");

        stmt.setString(1, owners.getName());
        stmt.setLong(2, owners.getAddressid());
        stmt.setString(3, owners.getHousenumber());
        stmt.setString(4, owners.getWatermeter());
        stmt.setLong(5, owners.getOwnertypeid());

        stmt.execute();
    } catch (SQLException e) {

        e.printStackTrace();
    } finally {
        DaoUtil.closeAll(null, stmt, conn);
    }
}
```

注意执行语句的接口为java.sql.CallableStatement,

语句的格式: {call pro_owners_add(?,?,?,?,?)}

第五节

5.1 存储过程-带传出参数

5.1.1知识概述

```
--增加业主信息存储过程
create or replace procedure pro_owners_add
(
  v_name varchar2,
  v_addressid number,
  v_housenumber varchar2,
  v_watermeter varchar2,
  v_type number,
  v_id out number
)
is
begin
  select seq_owners.nextval into v_id from dual;
  insert into T_OWNERS values( v_id,v_name,v_addressid,v_housenumber,v_watermeter,sysdate,v_type );
  commit;
end;
```

直接在参数的后面声明是out参数，不需要有返回值。

5.2 使用JDBC调用带传出参数的存储过程

5.2.1知识概述

```
/**
 * 增加
 * @param owners
 */
public static long addOut(Owners owners){
    long id=0;
    java.sql.Connection conn=null;
    java.sql.CallableStatement stmt=null;
    try {
        conn=DaoUtil.getConnection();
        stmt=conn.prepareCall("{call pro_owners_add(?,?,?,?,?,?)}");
        stmt.setString(1, owners.getName());
        stmt.setLong(2, owners.getAddressid());
        stmt.setString(3, owners.getHousenumber());
        stmt.setString(4, owners.getWatermeter());
        stmt.setLong(5, owners.getOwnertypeid());
        stmt.registerOutParameter(6, OracleTypes.NUMBER);//注册传出参数类型
        stmt.execute();
        id=stmt.getLong(6);//提取传出参数
    } catch (SQLException e) {
        e.printStackTrace();
    }finally {
        DaoUtil.closeAll(null, stmt, conn);
    }
    return id;
}
```

注意：

1、执行plsql语句的接口是java.sql.CallableStatement；

2、plsql语句的写法：{call pro_owners_add(?,?,?,?,?,?)}；

3、out参数的设置：stmt.registerOutParameter(6, OracleTypes.NUMBER);//注册输出参数

4、获取out参数的值：id=stmt.getLong(6);//提取传出参数

5.3 触发器-概述

5.3.1 知识概述

概述：

数据库触发器是一个与表相关联的、存储的PL/SQL程序。每当一个特定的数据操作语句(Insert,update,delete)在指定的表上发出时，Oracle自动地执行触发器中定义的语句序列。

应用：

数据确认

实施复杂的安全性检查

做审计，跟踪表上所做的数据操作等

数据的备份和同步

分类：

前置触发器和后置触发器

5.4 触发器-语法

5.4.1 知识概述

行级触发器：for each row，修改每一条记录都会修改

语句级触发器：可以一次修改多次记录，只触发一次。

伪纪录元素

触发语句	:old	:new
Insert	所有字段都是空(null)	将要插入的数据
Update	更新以前该行的值	更新后的值
delete	删除以前该行的值	所有字段都是空(null)

第六节

6.1 触发器-前置触发器

6.1.1 知识概述

```
create or replace trigger tri_account_updatenum1
before --前置
update of num1 --修改num1列时
on t_account --修改表t_account
for each row --行级
declare
begin
  :new.usenum:=:new.num1-:new.num0;
end;
```

6.2 触发器-后置触发器

6.2.1 知识概述

```
--创建业主名称修改日志表:用于记录业主更改前后的名称
create table t_owners_log
(
updatetime date,
ownerid number,
oldname varchar2(30),
newname varchar2(30)
);

--创建后置触发器，自动记录业主更改前后日志
create trigger tri_owners_log
after --后置
update of name
on t_owners --修改t_owners表中的name字段
for each row --行级
declare

begin
  insert into t_owners_log values(sysdate,:old.id,:old.name,:new.name);
end;
```

引用型变量：%TYPE

```
--变量名 表名.属性名%type
sname student_info.student_name;
```

示例中少写了%type

记录型变量：%ROWTYPE

```
--变量名 表名%rowtype;
student student_info%rowtype;
--使用列的值时，变量名.列名
dbms_output.put_line(student.student_name);
```

回帖

2条回帖

3371906349

可以的

沙发 • 2017-5-26 23:31:15

回帖

来自宇宙超级黑马专属苹果客户端

太炎氏太昊氏健

好详细的资料

藤椅 • 2017-5-27 06:39:00

回帖

来自宇宙超级黑马专属苹果客户端