**第08章 plsql**

**本章内容**

* PL/SQL

# 概念

pl/sql：过程语言,与结构化查询语言SQL结合而成的编程语言, 使用pl/sql 编程可以轻松地完成(分页存储、订单处理、转账存储)等复杂的查询要求

# 语法结构

declare ---声明部分

--声明变量

--声明常量

begin

--执行部分

exception --异常处理

end;

# pl/sql包含元素

* + sql语句:select into
  + 操作dml语句：insert into、update 、delete
  + dcl语句：grant、rollback、commit、revoke

# 示例

**要求：输出“helloworld”**

declare

begin

dbms\_output.put\_line('helloworld');

end;

# 常用符号

||:拼接字符串

/：执行plsql程序

&：接收键盘输入内容

设置输出参数：set serveroutput on;

输出语句： dbms\_output.put\_line(值);

# 数据类型

* + - 数字类型：number
    - 字符类型：char、varchar2、long
    - 逻辑类型：trun、false
    - 空值类型：null
    - %type:跟表中的字段的类型一致

例：v\_ename emp.ename%type;

* + - %rowtype:跟表中的行类型一致

例：v\_emp emp%rowtype;

* + - 记录类型：record

type 记录类型名称 is record

(

变量1 变量类型,

变量1 变量类型,

...............

);

v\_record 记录类型名称;

# 常量

* + - 常用名 constant 数据类型 :=值
    - 常用名 constant 数据类型 default 值

**例如：**

c\_name constant varchar2(20):='tom';

c\_name constant varchar2(20) default 'tom';

**示例：**

declare

c\_name constant varchar2(20):='tom';

begin

dbms\_output.put\_line(c\_name);

end;

# 变量

* + - 变量名称 数据类型;
    - 变量名称 数据类型 :=值;
    - 变量名称 数据类型 default 值

**例如：**

v\_name varchar2(20);

v\_name varchar2(20) :='tom';

v\_name varchar2(20) default 'tom';

**示例：**

declare

v\_name varchar2(20) :='tom';

begin

dbms\_output.put\_line(v\_name);

end;

**练习1：**

使用pl/sql输出“自我介绍”的信息,包括（姓名、性别、年龄、学校）

**练习2：**%type使用

输入部门编号，返回部门名称

**练习3：**%rowtype使用

输入部门编号，返回所有类数据

**练习4：**record使用

输出员工编号，返回员工姓名，工资，及所在部门名称

# PL/SQL中控制语句

* + if语句
  + 循环语句
  + 其它语句——goto 和null；

## if语句(三种)

* + if..then
  + if..then..else
  + if..then..elsif..then

**if..then（第一种）**

if 表达式 then

PLSQL语句块;

end if;

 **if..then..else(第二种)**

if 表达式 then

PLSQL语句块;

else

PLSQL语句块;

end if;

** if..then..elsif..then(第三种)**

if 表达式 then

PLSQL语句块;

elsif 表达式 then

PLSQL语句块;

else

PLSQL语句块;

end if;

**练习：**

从键盘输入一个金额

当金额>=0 and 金额 <100 输出“工薪阶级”

当金额>=100 and 金额<200 输出“中产阶级”

否则输出“资本家”.

**case语句：**

CASE 值

WHEN 表达式 THEN PL/SQL语句;

WHEN 表达式 THEN PL/SQL语句;

…….

[ELSE PL/SQL语句;]

END CASE;

**示例：**

*--第一种实现*

declare

v\_sex number(8):=&sex;

v\_result varchar2(20);

begin

case

when v\_sex=1 then v\_result:='男';

when v\_sex=2 then v\_result:='女';

else

v\_result:='其它';

end case;

dbms\_output.put\_line('性别：' || v\_result);

end;

*--第二种实现*

declare

v\_sex number(8):=&sex;

v\_result varchar2(20);

begin

case v\_sex

when 1 then v\_result:='男';

when 2 then v\_result:='女';

else

v\_result:='其它';

end case;

dbms\_output.put\_line('性别：' || v\_result);

end;

**练习：**

输入数字（1、2、3），分别代表：剪刀、石头、布

如果输入的数不在1-3范围，就输出“输入的数无效！”

## 循环(三种)

* + loop循环
  + while循环
  + for循环

loop(第一种)

loop

PL/SQL语句块;

exit when 条件;

end loop;

**示例：**1-10输出

declare

v\_i number := 1;

begin

loop

dbms\_output.put\_line(v\_i);

exit when v\_i=10;

v\_i:=v\_i+1;

end loop;

end;

while(第二种)

while 条件 loop

PL/SQL语句块;

end loop;

**练习：1-10输出**

for(第三种)

for i in [reverse] 开始下标..结束下标 loop

PL/SQL语句块;

end loop;

**练习：1-10输出**

## goto及null

goto：用于跳转到特定符号去执行语句(了解)

**示例：**

declare

i int := 1;

begin

loop

dbms\_output.put\_line('输出i=' || i);

if i = 5 then

goto end\_loop;

end if;

i := i + 1;

end loop;

<<end\_loop>>

dbms\_output.put\_line('循环结束');

end;

**null：**null语句不会执行任何操作，并且会直接将控制传递到下一条语句。使用null语句的主要好处是可以提高pl/sql 的可读性

**示例：**

declare

v\_number number(11) :=5;

begin

if v\_number < 10 then

dbms\_output.put\_line('此数小于10');

else

null;

end if;

end;

# 异常处理

## 异常分类

* + - 预定义异常
    - 非预定义异常
    - 自定义异常

预定义异常：预定义异常用于处理常见的oracle 错误

非预定义异常：非预定义异常用于处理预定义异常不能处理的异常

自定义异常：自定义异常用于处理与oracle 错误无关的其它情况

## 未处理异常的情况

假如输入员工编号“1234”，查询显示员工的姓名。观察结果。

declare

v\_ename emp.ename%type;

begin

select ename into v\_ename from emp where empno=&gno;

dbms\_output.put\_line('名字：'||v\_ename);

end;

## 使用异常的情况

declare

v\_ename emp.ename%type;

begin

select ename into v\_ename from emp where empno=&gno;

dbms\_output.put\_line('名字：'||v\_ename);

exception

when no\_data\_found then

dbms\_output.put\_line('编号没有！');

end;