PL/SQL:是过程化/sql的缩写，是Oracle在sql的基础之上扩展来的一种数据库编程语言，在兼容sql的基础之上，扩充了许多新功能，是面向过程化语言和sql的结合。

PL/SQL特点：

·除了基本的sql之外，包含了控制结构和异常处理，从而具有了sql语句的简洁性和过程性。

·每个sql语句的请求处理，都会引起一次网络的传输，用户量一旦大量增加，可能会导致网络堵塞，而PLSQL是将整个语句块发送给数据库服务器，减少了网络通信的数量，提高了应用程序的运行效率。

·PLSQL语句是存储在Oracle服务器上，可以被其他的PLSQL程序或者sql调用。具有很高的重用性。

·PLSQL程序语言是一种块结构语言，PLSQL的基本代为是块。

PLSQL程序块有两种：

·匿名块：没有名称，只用一次，不能存储在数据库中

·命名块：指的是保存在数据库中的PLSQL程序块，比如我们的存储过程、函数和触发器

块的基本结构：

·匿名块的基本结果

|  |
| --- |
| declare *-- 可选*  begin  update dh02t\_person set p\_name='李四' where p\_id = 14;  *--exception 可选*  commit;  end; |

declare：声明部分，用来声明程序中的变量、类型、游标等等。

|  |
| --- |
| declare  pname varchar2(20) := '朱炳宇';  begin  update dh02t\_person set p\_name=pname where p\_id = 14;  *--exception 可选*  commit;*--提交*  end; |

begin：是PLSQL程序的主体部分，一般使用sql语句或者过程化语言来处理特定的操作。

exception：异常的处理部分，用来检查和处理异常或者错误。

块中的每一条语句都需要以分号结束，一条sql可以分成多行来写，但是最终只需要一个分号。

注释：在PLSQL中，注释分为两种，一种单行注释“--”，两外就是多行注释”/\* 需要注释的内容\*/”

范例：向表中插入一笔数据并且提交。

|  |
| --- |
| declare  pid number(18) ;  pname varchar2(20);  psex number(1);  pbir date;  begin  *--给变量赋值*  pid := 20;  pname := '姜建民';  psex := 1;  pbir := to\_date('1988-11-12','yyyy-mm-dd');    insert into dh02t\_person(p\_id,p\_name,p\_sex,p\_bir) values (pid,pname,psex,pbir);  commit;*--提交*  end; |

声明常量和变量

·声明常量：常量名 constant 类型(长度) := 值

|  |
| --- |
| declare  pid constant number(18) := 29;*--声明常量*  begin  pid := 90;*--不能再次赋值*  end; |

constant：表示的就是固定不变的，就是我们说的常量。声明之后不允许改变。

·声明变量：变量名 类型(长度) （not null） （default | := ）

|  |
| --- |
| pid number(18) not null default 21; |

声明变量和常量中一些规则：

·一行只能声明一个变量或者常量

·变量时存储值的内存区域，用来处理程序中的值，变量名称不能重复，也不能使用系统关键字。

·变量名称以字母开头（不区分大小写）,可以由字母，数字，下划线组成。

·变量名称的长度不能超过30个字符

·分支结果

在ＰＬＳＱＬ中分支分为两种：

·if then els（if then elsif then。。els）

·case（when 。。。else），是按照顺序检查表达式的值，如果一旦找到，则跳出case语句。

范例：对dh02t\_student表中的java\_score成绩字段做一个等级。50-60：优，40-50：良，30-40：及格，30以下，挂了

·使用第一种语法

|  |
| --- |
| declare  score number(2);  level\_score varchar2(3);  begin  select java\_score into score from dh02t\_student where stu\_id=1;*--将查询的值赋给变量score*  if score >=50 then  level\_score := '优';  elsif score >=40 then  level\_score := '良';  elsif score >=30 then  level\_score := '及格';  else  level\_score := '挂科';  end if;  dbms\_output.put\_line(level\_score);  end; |

·使用第二种语法：

|  |
| --- |
| declare  score number(2);  level\_score varchar2(3);  begin  select java\_score into score from dh02t\_student where stu\_id=1;*--将查询的值赋给变量score*  case  when score >=50 then  level\_score := '优';  when score >=40 then  level\_score := '良';  when score >=30 then  level\_score := '及格';  else  level\_score := '挂科';  end case;  dbms\_output.put\_line(level\_score);  end; |

·循环结果：

·简单循环

loop

语句块；

end loop；

这种循环语句没有终止，必须进行人为的控制，一般通常是加入exit或者exit的语句进行结束循环。

·for循环

for 循环的变量 in[reverse] 起始值 。。终止值 loop

语句块；

end loop；

for循环中，次数是知道（固定不变的），如果存在reverse，则表示的倒序循环

·while循环

while 条件表达式 loop

语句块；

end loop；

通过条件表达式来控制循环的执行，如果条件表达式为真，则执行循环，如果为false，则终止循环。

范例：从1+2+3+4+。。。100，输出结果

·使用简单循环

|  |
| --- |
| declare  v\_sum number(5) := 0;*--定义循环相加后的结果*  v\_int number(3) := 1;*--定义循环的变量*  begin  loop  v\_sum := v\_sum + v\_int;  v\_int := v\_int+1;  exit when v\_int>100;*--当v\_int大于100则退出循环*  end loop;  dbms\_output.put\_line('相加后结果为：' || v\_sum);  end; |

·使用for循环

|  |
| --- |
| declare  v\_sum number(5) :=0;  begin  for i in 1..100 loop  v\_sum := v\_sum +i;  end loop;  dbms\_output.put\_line('相加后结果为：' || v\_sum);  end; |

·使用while循环

|  |
| --- |
| declare  v\_sum number(5) := 0;*--定义循环相加后的结果*  v\_int number(3) := 1;*--定义循环的变量*  begin  while v\_int <= 100 loop  v\_sum := v\_sum + v\_int;  v\_int := v\_int+1;  end loop;  dbms\_output.put\_line('相加结果：' || v\_sum);  end; |

游标：是从数据库中查询出来的结果集放在内存中，游标就是指向该内存的指针，通过移动游标来取得该结果集中的每一笔数据，执行不同的操作。

游标的操作步骤：

·声明游标：声明游标就是定义游标的名称，并将这一游标与select语句相关联。

语法：cursor 游标名称 is select 语句

其中select以及后面语句就是查询语句

·打开游标

语法：open 游标名，游标使用之前必须打开，游标打开之后该指针指向的是结果集中的第一笔数据。

·将结果集中的的数据提取（fetch）到声明的变量中

每次fetch提取到数据之后，游标指针就移向下一笔数据：fetch 游标名 into 变量列表

into之后的变量列表用来存储游标中相应的字段，变量的个数、顺序、类型都需要和select 语句中查询的结果集相对应。

·关闭游标

语法：close 游标名，游标使用结束，都必须关闭，用来释放游标所占用的资源

范例：将查询的结果集打印

|  |
| --- |
| declare  cursor dh02tstudent is select stu\_nm ,java\_score from dh02t\_student;*--定义游标*  score number(2) := 0;  sname varchar2(10);  begin  open dh02tstudent;  fetch dh02tstudent into sname,score;  dbms\_output.put\_line(sname||'-----'||score);  fetch dh02tstudent into sname,score;  dbms\_output.put\_line(sname||'-----'||score);  fetch dh02tstudent into sname,score;  dbms\_output.put\_line(sname||'-----'||score);    close dh02tstudent;  end; |

游标的属性：在PLSQL中游标有四种属性

·第一种：%found，如果前一个fetch提取数据，如果提取到了这个值则为真，否则为假。

·第二种：%notfound, 如果前一个fetch提取数据，如果提取到了这个值则为假，否则为真。

·第三种：%isopen ，判断游标是否打开，如果打开则为真，否则为假。

·第四种：%rowcount,返回游标当前提取的行数

游标的四种属性前三种都是逻辑型，第四种是数值型，游标的属性可以用于取得游标当前的状态。

游标的循环提取：

·简单循环

·for循环

·while循环

简单循环和while都需要使用open、fetch、close关键字进行显示游标的操作，而for循环则可以对隐式游标进行处理。

·简单循环：

|  |
| --- |
| declare  cursor dh02tstudent is select \* from dh02t\_student;*--定义游标*  stu dh02t\_student%rowtype;*--定义没一行的结果列*  begin  open dh02tstudent;  loop  fetch dh02tstudent into stu ;  exit when dh02tstudent%notfound;  dbms\_output.put\_line(stu.stu\_id||'-----'||stu.stu\_nm||'-----'||stu.stu\_no||'-----'||stu.java\_score||'-----'||stu.group\_nm);  end loop;  close dh02tstudent;  end; |

·while循环

|  |
| --- |
| declare  cursor dh02tstudent is select \* from dh02t\_student;*--定义游标*  stu dh02t\_student%rowtype;*--定义没一行的结果列*  v\_count number(10);  v\_count\_num number:=0;  begin  select count(stu\_id) into v\_count from dh02t\_student;  open dh02tstudent;  while v\_count>=v\_count\_num  loop  v\_count\_num :=v\_count\_num+1;  fetch dh02tstudent into stu;  */\* fetch dh02tstudent into stu ;*  *exit when dh02tstudent%notfound;\*/*  dbms\_output.put\_line(stu.stu\_id||'-----'||stu.stu\_nm||'-----'||stu.stu\_no||'-----'||stu.java\_score||'-----'||stu.group\_nm);  end loop;  close dh02tstudent;  end; |

·for循环

|  |
| --- |
| declare  begin  for stu in(select \* from dh02t\_student) loop  dbms\_output.put\_line(stu.stu\_id||'-----'||stu.stu\_nm||'-----'||stu.stu\_no||'-----'||stu.java\_score||'-----'||stu.group\_nm);  end loop;  end; |

以上的程序没有定义游标，但是完成了游标所能完成的功能， 这种就是隐式游标。

for循环不仅可以处理隐式游标，还可以处理显示游标。

|  |
| --- |
| declare  cursor dh02tstudent is select \* from dh02t\_student;  begin  for stu in dh02tstudent loop  dbms\_output.put\_line(stu.stu\_id||'-----'||stu.stu\_nm||'-----'||stu.stu\_no||'-----'||stu.java\_score||'-----'||stu.group\_nm);  end loop;  end; |

类似于这种情况，只需要使用隐式游标即可，一般情况下，使用for循环处理游标时就不需要写显示游标。

练习：新建一张表，里面有姓名，年龄，工资字段，如果工资大于2000的则提升50%，工资大于3000的提升30%，工资大于5000的提升10%。