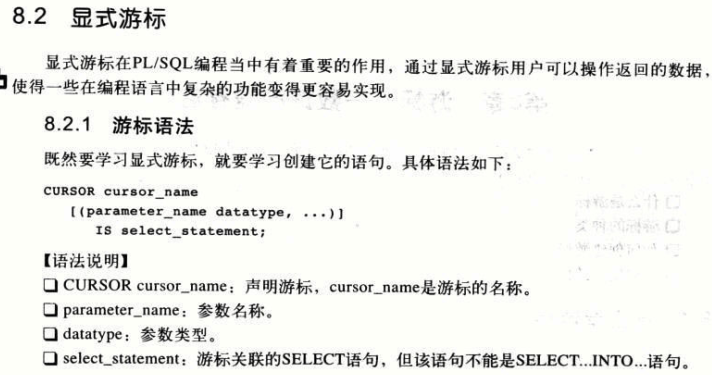
# 显式游标

## **显式游标的创建（非声明）语法**：



Cursor cursor1(import varchar2)

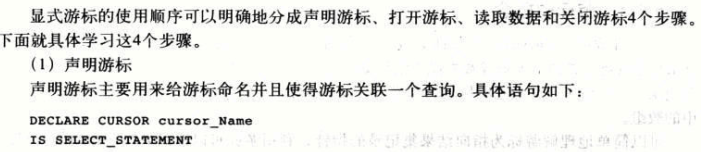
Is

Select \* from statement;

## 显式游标声明语法：

1声明游标——2打开游标——3读取数据——4关闭游标

声明游标

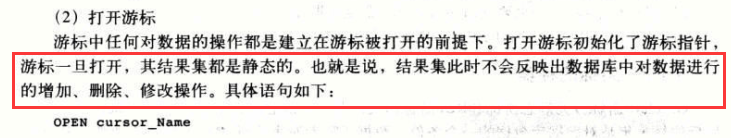


Declare cursor rzycursor

Is

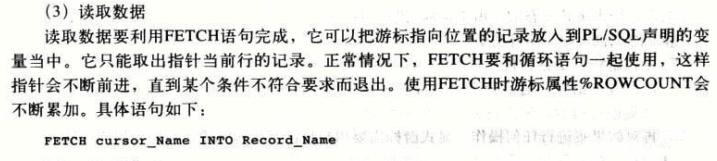
Select \* from emp; (select 语句末尾分号’;’)

打开游标



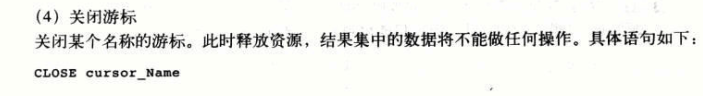
Open rzycursor; (末尾分号’;’)

读取数据



Fetch rzycursor into variable; (末尾分号’;’)

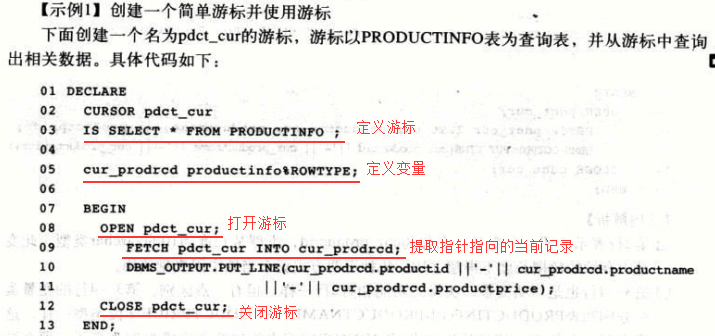
关闭数据



Close rzycursor; (末尾分号’;’)

--

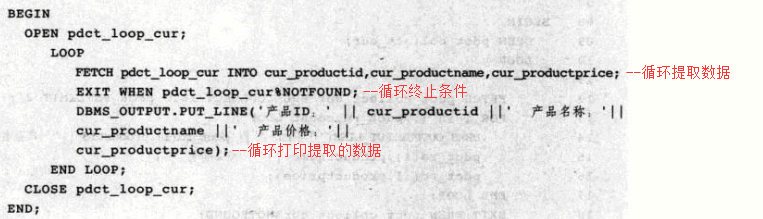
示例：



--

## CURSOR中的循环（LOOP）

（fetch cursor into variable; ——通常fetch 语句要结合loop（循环），以达到选择多条数据的效果。）



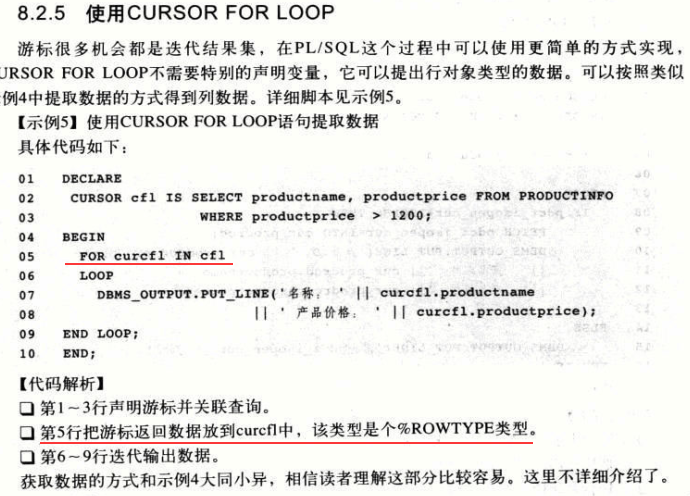
## CURSOR使用BULK COLLECT和FOR语句

Fetch...into.. (一条一条遍历数据)

Fetch...bulk collect into...(批量提取数据，在数据量大的情况下效率高于单条)

FOR 即for。。。loop循环

## CURSOR FOR LOOP



--

## CURSOR的四种属性

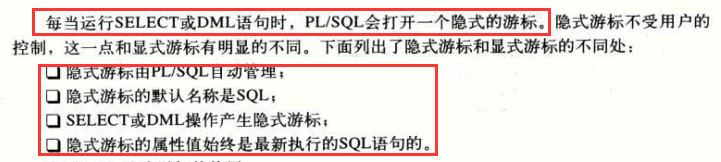
1. %ISOPEN：判断是否打开，如果打开返回true，cursor%isopen；
2. %FOUND：判断行是否有数据，如果有返回true，cursor%found；
3. %NOTFOUND：与%found相反，如果没有数据返回truecursor%notfound；
4. %ROWCOUNT：累计到当前为止使用fetch提取的数据行数；

--

## 带参数的CURSOR

--

# 隐式游标



# 游标知识巩固与练习

### --1、什么是游标？使用游标的基本步骤是什么？

/\*

当在PL/SQL块中执行查询语句（SELECT）和数据操纵语句（DML）时，

Oracle会在内存中分配一个缓冲区，缓冲区中包含了处理过程的必需信息，

包括已经处理完的行数、指向被分析行的指针和查询情况下的活动集，即查询语句返回的数据行集。

该缓冲区域称为上下文区，游标是指向该缓冲区的句柄或指针。

\*/

### --2、游标有哪几种类型？分别在什么情况下使用？

/\*

（1）显式游标：由用户定义、操作，用于处理返回多行数据的SELECT查询。

（2）隐式游标：由系统自动进行操作，用于处理DML语句和返回单行数据的SELECT查询。

\*/

### --3、用游标显示所有部门编号与名称，以及其所拥有的员工人数。

--方法一

declare

  cursor c\_emp is select \* from emp;

  cursor c\_dept is select \* from dept;

  v\_count number;

begin

  for v\_dept in c\_dept loop

    v\_count := 0;

    for v\_emp in c\_emp loop

      if v\_emp.deptno = v\_dept.deptno then

        v\_count := v\_count + 1;

      end if;

    end loop;

    dbms\_output.put\_line('编号：' || v\_dept.deptno || '是：' ||

                        v\_dept.dname || '部门，共有员工' || v\_count || '人。');

  end loop;

end;

--方法二：（PS：缺失没有员工的部门）

declare

  cursor c\_temp is (select deptno, dname, count(\*) count\_p

  from (select d.deptno, d.dname

          from dept d

          join emp e on d.deptno = e.deptno) t

 group by deptno, dname);

begin

  for v\_temp in c\_temp loop

    dbms\_output.put\_line(v\_temp.deptno || '-' || v\_temp.dname || '-' || v\_temp.count\_p);

  end loop;

end;

### --4、用游标属性%rowcount实现输出前十个员工的信息。

declare

  cursor c is (select \* from emp);

begin

  for v in c loop

    if c%rowcount<=10 then

      dbms\_output.put\_line(c%rowcount || '-' || v.empno || '-' || v.ename || '-' || v.sal);

    else exit;

    end if;

  end loop;

end;

### --5、通过使用游标来显示dept表中的部门名称，及其相应的员工列表（提示：可以使用双重循环）。

declare

  cursor c\_emp is select \* from emp;

  cursor c\_dept is select \* from dept;

begin

  for v\_dept in c\_dept loop

    dbms\_output.put\_line( v\_dept.dname || ' 部门人员列表：');

    for v\_emp in c\_emp loop

      if v\_emp.deptno = v\_dept.deptno then

        dbms\_output.put\_line('    ' || v\_emp.ename);

      end if;

    end loop;

    dbms\_output.put\_line('');

  end loop;

end;

### --6、接受一个部门号，使用For循环，从emp表中显示该部门的所有雇员的姓名，工作和薪水。

declare

  cursor c\_emp is select \* from emp where deptno = &please\_enter\_deptno;

  v\_dept dept%rowtype;

begin

  select \* into v\_dept from dept where  deptno = &please\_enter\_deptno;

  dbms\_output.put\_line(&please\_enter\_deptno || ' :: ' || v\_dept.dname);

  dbms\_output.put\_line('');

  dbms\_output.put\_line('ename :: job :: sal');

  for v\_emp in c\_emp loop

    dbms\_output.put\_line(v\_emp.ename || ' :: ' || v\_emp.job || ' :: ' || v\_emp.sal);

  end loop;

end;

### --7、编写一个程序块，将emp表中前5人的名字，及其出的工资等级（salgrade）显示出来。

begin

  for v\_emp in (select e.ename, s.grade

  from emp e

  join salgrade s on e.sal between s.losal and s.hisal where rownum < =5) loop

    dbms\_output.put\_line(v\_emp.ename || ' :: grade ' || v\_emp.grade);

  end loop;

end;

### --8、用带参数的游标输出部门编号为10， 30的员工信息。

declare

  cursor c\_emp(v\_deptno dept.deptno%type) is select \* from emp where deptno = v\_deptno;

begin

  dbms\_output.put\_line('部门编号为：10 的员工列表');

  for v\_emp in c\_emp(10) loop

    dbms\_output.put\_line(v\_emp.ename);

  end loop;

  dbms\_output.put\_line('部门编号为：30 的员工列表');

  for v\_emp in c\_emp(30) loop

    dbms\_output.put\_line(v\_emp.ename);

  end loop;

end;

### --9、使用带参数的游标，实现接受一个部门名称，从emp表中显示该部门的所有雇员的姓名，工作和薪水。

declare

  cursor c\_emp(v\_dname dept.dname%type) is select \* from emp where deptno = (select deptno from dept where dname = v\_dname);

begin

  dbms\_output.put\_line('SALES 部门员工名单：');

  for v\_emp in c\_emp('SALES') loop

    dbms\_output.put\_line(v\_emp.ename || ' :: ' || v\_emp.job || ' :: ' || v\_emp.sal);

  end loop;

end;

### --10、用游标获取所有收入超过2000的 salesman.

declare

  cursor c\_emp is select \* from emp;

begin

  dbms\_output.put\_line('收入超过1500的SALESMAN列表：');

  --收入超过2000的salesman没有数据

  for v\_emp in c\_emp loop

    if v\_emp.job = 'SALESMAN' and v\_emp.sal > 1500 then

      dbms\_output.put\_line(v\_emp.ename || ' :: ' || v\_emp.sal);

    end if;

  end loop;

end;

### --11、编写一个PL/SQL程序块，从emp表中对名字以"A"或"S"开始的所有雇员按他们基本薪水的10%给他们加薪。

declare

  cursor c\_emp is select \* from emp2 for update;

begin

  for v\_emp in c\_emp loop

    if substr(v\_emp.ename,1,1) = 'A' or substr(v\_emp.ename,1,1) = 'S' then

      dbms\_output.put\_line('加薪员工：' || v\_emp.ename || '加薪：' || v\_emp.sal\*0.1);

      update emp2 set sal = sal\*1.1 where ename = v\_emp.ename;

    end if;

  end loop;

  commit;

end;

### --12、emp表中对所有雇员按他们基本薪水的10%给他们加薪，如果所增加后的薪水大于5000，则取消加薪。

begin

  for v\_emp in (select ename,sal,empno from emp2) loop

    if v\_emp.sal\*1.1 < 5000 then

      dbms\_output.put\_line('待加薪的员工：' || v\_emp.ename);

      update emp2 set sal = sal \* 1.1 where empno = v\_emp.empno;

    end if;

  end loop;

  commit;

end;

### --13、按照salgrade表中的标准，给员工加薪，1：5%，2：4%，3：3%，4：2%，5：1%， 并打印输出每个人，加薪前后的工资。

--方法一

declare

  cursor c\_emp is(

    select ename, sal, grade

      from emp2 e

      join salgrade s on e.sal between s.losal and s.hisal) order by ename;

begin

  for v\_emp in c\_emp loop

    update emp2 set sal = sal \* (1+((6 - v\_emp.grade)\*0.01)) where emp2.ename = v\_emp.ename;

    dbms\_output.put\_line(v\_emp.ename || '，加薪前：' || v\_emp.sal || '，等级：' ||

    v\_emp.grade || '，加薪后：' || v\_emp.sal \* (1+((6 - v\_emp.grade)\*0.01)));

  end loop;

  commit;

end;

--方法二

declare

  cursor c\_emp is(

    select ename, sal, grade

      from emp2 e

      join salgrade s on e.sal between s.losal and s.hisal) order by ename;

begin

  for v\_emp in c\_emp loop

    case v\_emp.grade

    when 1 then

      update emp2 set sal = sal \* (1+0.05) where emp2.ename = v\_emp.ename;

    when 2 then

      update emp2 set sal = sal \* (1+0.04) where emp2.ename = v\_emp.ename;

    when 3 then

      update emp2 set sal = sal \* (1+0.03) where emp2.ename = v\_emp.ename;

    when 4 then

      update emp2 set sal = sal \* (1+0.02) where emp2.ename = v\_emp.ename;

    when 5 then

      update emp2 set sal = sal \* (1+0.01) where emp2.ename = v\_emp.ename;

    else null;

    end case;

/\*

dbms\_output.put\_line(v\_emp.ename || '，加薪前：' || v\_emp.sal || '，等级：' ||

v\_emp.grade || '，加薪后：' || v\_emp.sal \* (1+((6 - v\_emp.grade)\*0.01)));

\*/

  end loop;

  commit;

end;