SQL 游标

一般情况下，SELECT 语句查询结果是多条记录，因此需要用游标机制，将多条记录一次一条送主程序处理，从而把对集合的操作转换为对单个记录的处理。使用游标的步骤为：

a) 说明游标。用 DECLARE 语句为一条 SELECT 语句定义游标。定义游标仅仅是一条说明性语句，并不执行 SELECT 语句。

b) 打开游标。用 OPEN 语句将定义的游标打开。打开游标实际上是执行相应的 SELECT 语句，把查询结果取到缓冲区中。这时游标处于活动状态，指针指向查询结果集中的第一条记录。

c) 推进游标指针并取当前记录。用 FETCH 语句把游标指针向前推进一条记录，同时将缓冲区中的当前记录取出来送至主变量供主语言进一步处理。通过循环执行 FETCH 语句逐条取出结果集中的行进行处理。

当在 PL/SQL 块中执行 DML 和单行 select into 语句时，Oracle 会分配隐含游标。为了处理 select 语句返回的多行数据，需要使用显式游标。

显式游标专门用于处理 select 语句返回的多行数据，其包含四个属性，分别为：%ISOPEN、%FOUND、%NOTFOUND、%ROWCOUNT。当使用显式游标时，必须经历定义游标、打开游标、提取数据和关闭游标四个阶段，具体过程如下：

首先创建一个表用于实验。

```
C:\Users\JiangXue>sqlplus/nolog

SQL*Plus: Release 11.1.0.7.0 - Production on 星期二 2月 14 09:21:51 2012

Copyright (c) 1982, 2008, Oracle.  All rights reserved.

SQL> conn sys/ora505 as sysdba
已连接。
SQL> create user test identified by test
  2  default tablespace t1;

用户已创建。

SQL> grant dba to test;

授权成功。
```

在 test 用户下创建表 empinf 并为其插入一条语句。

```
SQL> conn test/test
已连接。
SQL>  create table empinf
  2   (id int primary key not null,
  3   name varchar2(20),
  4   sal number(8,2),
  5   job varchar2(10),
  6   dept varchar2(20));

表已创建。

SQL>
SQL>  insert into empinf
  2   values(1,'Scofi',7000.00,'manager','软件部');

已创建 1 行。
```

(1) 标量变量接受游标数据

```
declare cursor empinf_cursor is
    select name,job,sal from empinf where id=&number;
    v_name empinf.name%type;
    v_job empinf.job%type;
    v_sal empinf.sal%type;
begin
    open empinf_cursor;
    loop
        fetch empinf_cursor into v_name,v_job,v_sal;
        exit when empinf_cursor%notfound;
        dbms_output.put_line('姓名：'||v_name||',岗位：'||v_job||',工资：'||v_sal);
    end loop;
    close empinf_cursor;
```

首先查看 empinf 表中的记录：

```
SQL> select * from empinf;

        ID NAME                           SAL JOB        DEPT
---------- -------------------- ---------- ---------- --------------------
         1 Scofi                      7000 manager    软件部
         2 Monh                       5000 clerk      人力资源部
         3 Bech                       2000 clerk      人力资源部
```

```
SQL> conn test/test
已连接。
SQL> set serveroutput on;
SQL> set verify on;
```

```
SQL> edit
已写入 file afiedt.buf

  1  declare cursor empinf_cursor is
  2     select name,job,sal from empinf where id=&number;
  3     v_name empinf.name%type;
  4     v_job empinf.job%type;
  5     v_sal empinf.sal%type;
  6  begin
  7     open empinf_cursor;
  8     loop
  9             fetch empinf_cursor into v_name,v_job,v_sal;
 10             exit when empinf_cursor%notfound;
 11             dbms_output.put_line('姓名: '||v_name||',岗位: '||v_job||',工资
: '||v_sal);
 12     end loop;
 13     close empinf_cursor;
 14* end;
SQL> /
```

注意：循环结束的条件是通过判断游标的 notfound 属性来实现的，当游标中值被取完后退出循环。

(2) 记录变量接收游标数据

采用 PL/SQL 记录变量接收游标数据时，以行来获取数据，因此不需要为每列都设定标量。具体如下：

```
declare
      cursor empinf_cursor_record is
      select name, sal from empinf where id>1;
      empinf_record empinf_cursor_record%rowtype;
begin
      open empinf_cursor_record;
      loop
            fetch empinf_cursor_record into empinf_record;
            exit when empinf_cursor_record%notfound;
            dbms_output.put_line('姓名：'||empinf_record.name||',工资：'||
empinf_record.sal);
      end loop;
      close empinf_cursor_record;
end;
/
```

```
SQL> edit
已写入 file afiedt.buf

  1  declare
  2     cursor empinf_cursor_record is
  3     select name, sal from empinf where id>1;
  4     empinf_record empinf_cursor_record%rowtype;
  5  begin
  6     open empinf_cursor_record;
  7     loop
  8             fetch empinf_cursor_record into empinf_record;
  9             exit when empinf_cursor_record%notfound;
 10             dbms_output.put_line('姓名: '||empinf_record.name||',工资: '||
 11  empinf_record.sal);
 12     end loop;
 13     close empinf_cursor_record;
 14* end;
SQL> /
姓名: Monh,工资: 5000
姓名: Bech,工资: 2000

PL/SQL 过程已成功完成。
```

上述过程完成了用记录 empinf_cursor 接收显式游标 empinf_cursor_record。可以看到，采用记录接收游标数据比标量接收游标数据方便多了。

(3) 更新游标行

```
declare
    cursor empinf_cursor is
    select name, sal,id from empinf for update;
begin
    for i in empinf_cursor
    loop
        if i.id =1 then
        dbms_output.put_line('姓名：'||i.name||',原工资：'||i.sal);
        update empinf set sal=sal*1.5 where current of empinf_cursor;
        end if;
    end loop;
end;
```

```
SQL> declare
  2     cursor empinf_cursor is
  3     select name, sal,id from empinf for update;
  4  begin
  5     for i in empinf_cursor
  6     loop
  7             if i.id =1 then
  8             dbms_output.put_line('姓名: '||i.name||',原工资: '||i.sal);
  9             update empinf set sal=sal*1.5 where current of empinf_cursor;
 10             end if;
 11     end loop;
 12  end;
 13  /
```

```
PL/SQL 过程已成功完成。

SQL> select * from empinf
  2  ;

        ID NAME                          SAL JOB        DEPT
---------- -------------------- ---------- ---------- --------------------
         1 Scofi                     10500 manager    软件部
         2 Monh                       5000 clerk      人力资源部
         3 Bech                       2000 clerk      人力资源部
```

上述通过游标修改 id 为 1 的员工 Scofi 的工资，使其变为原来的 1.5 倍，并输出原来的工资。

注：使用 For..Loop 的方式读取 cursor，open、fetch、close 都是隐式打开的。所以，大家不用担心忘记关闭游标而造成性能上的问题。使用 For..Loop 读取游标后，存储记录的变量不需要定义，而且，可以自动匹配类型。

(4) 批量提取

Oracle 9i 以后可以使用 fetch…bulk collect 提取所有数据，具体如下所示：

```
declare
    cursor empinf_cursor is
    select * from empinf where lower(job)=lower('&job');
    type empinf_table_type is table of empinf%rowtype;
    empinf_table empinf_table_type;
begin
    open empinf_cursor;
    fetch empinf_cursor bulk collect into empinf_table;
    close empinf_cursor;
    for i in 1..empinf_table.count loop
    dbms_output.put_line('姓名:'||empinf_table(i).name||', 工资:'||empinf_table(i).sal);
    end loop;
end;
```

```
SQL> edit
已写入 file afiedt.buf

  1  declare
  2     cursor empinf_cursor is
  3     select * from empinf where lower(job)=lower('&job');
  4     type empinf_table_type is table of empinf%rowtype;
  5     empinf_table empinf_table_type;
  6  begin
  7     open empinf_cursor;
  8     fetch empinf_cursor bulk collect into empinf_table;
  9     close empinf_cursor;
 10     for i in 1..empinf_table.count loop
 11     dbms_output.put_line('姓名:'||empinf_table(i).name||',工资:'||empinf_tab
le(i).sal);
 12     end loop;
 13* end;
 14  /
输入 job 的值:  clerk
原值    3:      select * from empinf where lower(job)=lower('&job');
新值    3:      select * from empinf where lower(job)=lower('clerk');
姓名:Monh,工资:5000
姓名:Bech,工资:2000

PL/SQL 过程已成功完成。
```

可以看到，当输入 clerk 时，将其所对应的记录显示出来，实现了批量提取所有数据的目的。如果输入的 job 值 empinf 表中不存在，则不会显示结果，如下：

```
SQL> /
输入 job 的值:  engineer
原值    3:      select * from empinf where lower(job)=lower('&job');
新值    3:      select * from empinf where lower(job)=lower('engineer');

PL/SQL 过程已成功完成。
```