Cursor Management: (Temporary Memory)

A work area called private SQL area is used by the oracle server to execute SQL statements and to store processed information. PL/SQL uses cursors to name the private SQL area and to access the stored information.

There are two types of Cursor

* Explicit Cursor
* Implicit Cursor

The syntax for declaring a cursor

cursor <cursor_name> is <select statement>;

Explicit Cursor:

The set of rows returned by a query can contain zero or multiple rows depending upon the query defined.

After declaring a cursor, we can use the following commands to control the cursor.

1. Open
2. Fetch
3. Close

The 'open' statement executes the query, identifies the active set and positions the cursor before the first row.

open<cursor_name>;

The 'fetch' statement retrieves the current row and advances the cursor to the next row to fetch the remaining rows.

fetch<cursor_name> into <column_name>;

After processing the last row in the active set, the cursor is disbled with the help of the 'close' command.

Close<cursor_name<;

Example illustrates cursors

```
declare
pri item.qty%type;
quantity item.qty%type;
done boolean:=true;
cursor a is select qty from item where order_id =2000;
begin
open a;
loop
fetch a into quantity;
pri:=3*quantity;
update item set actual_price = pri where order_id=2000;
exit when a%NOTFOUND;
dbms_output.put_line('one Record update');
end loop;
close a;
end;
/
```

Explicit Cursor attributes when appended to the cursor name allow us to access useful information from the retrieved row.

%not found
%found
%row count
%isopen

%Notfound:

After opening a cursor, a 'fetch' Statement is used to fetch rows from the active set, one at a time. The attribute %not found indicates whether fetch state% returns row from the active set %notfound attribute to exit a loop when 'fetch' fails to return a row.

```
declare

declare
cursor c2 is select actual_price,qty from item where qty_on_hand=2020;
total number(8);
price item.actual_price %type;
quantity item.qty%type;
begin
open c2;
loop
fetch c2 into price,quantity;
total:=price*quantity;
if  c2%FOUND then
dbms_output.put_line('The values of the total is ' || to_char(total));
exit;
else
 dbms_output.put_line('value not found');
exit;
end if;
end loop;
close c2;
end;
/
```

%Rowcount

~~~~~~~~~~

```
declare cursor c1 is select itemname from masteritem;
cnt number(10);
name masteritem.itemname%type;
begin
open c1;
loop
fetch c1 into name;
if c1%found then
cnt := c1%rowcount;
insert into tempp values(cnt,name);
else
exit;
end if;
end loop;
end;
/
```

%isopen

~~~~~~~

If the cursor is already open, then the attribute %isopen evaluates to true, else it evaluates to false.

```
declare
cursor c4 is select actual_price,qty from item where order_id=100;
begin

if c4%isopen then
dbms_output.put_line('this message is not displayed');
else
open c4;
dbms_output.put_line('Cursor opened');
end if;
close c4;
end;
/
```

IMPLICIT CURSOR

```
``````````````````````````
```

PL/SQL implicitly declares cursors for all sql data manipulation statements, including queries than return one row. For queries that return more than one row. We should use explicit cursors to access the row individually. Implicit cursor attributes can be use to access information about the most recently executed SQL statement.

```
begin
delete from masteritem where itemname='television';
if SQL%Notfound then
dbms_ouput.put_line('no records found')
else
dbms_output.put_line('record found');
end if;
end;
/
```
```