PL/SQL- (BY MURLI SIR-2019)

---

Editing by - " MD SHAMSHAD ALAM "

Batch No:- 9AM

---

---

NARESH iT(Technologies) , "Hyderabad(Ameerpet)" Start Date:- 20-10-2019

---

---

Not Theory only one PL/SQL Program:- All Program Question & Solution with(Output)

---

---

==========

==========

"Chapter Names"

==========

1. PL/SQL Introducation:->

---

----> Select..... into clause.
----> Variables Attributes(%type, %Rowtype).
----> Bind Variable.
----> Condition, control Statement.

2. CURSOR:->
---
----> Explicit cursor & life Cycle.
----> Explicit cursor & life Attributes.
----> Cursor-------- for loop.
----> Parameterized cursor.
----> Implicit cursor and Implicit Attributeds.
----> Function (or) Expression are used in Explicit cursor.
----> Update,delete,statements are used in Explicit cursor (without using where current of,for update clause).

3. EXCEPTIONS:->
---
----> Predefined exceptions.
----> User defined exceptions.
----> Unnamed exception.
----> Exception Propagations.
----> Raise-Application_error().
----> Error trapping function(Sql code, sql error).

4. SUB PROGRAMS:->
---
====>STORED PROCEDURES:=>
----> Precedure Parameter modes (int,out,intout).

----> No copy compiler hint.
----> Autonomous Transactions.
----> Authid current_user.
----> Accessible by clause(12c).

===>STORED FUNCTIONS:=>
----> DML statement are used in functions.
----> Select------- into clause used in function.
----> Corsors are used in functions.
----> when to use procedures, when to use functions.

5. TRIGGERS:->
---
----> Row level trigger.
----> Appliction of row level  trigger. (Auto-Increment Concept).
----> Trigger timing(Before/After).
----> Statement level  triggers.
----> Trigger execution order.
----> Follows Clause(11g).
----> Mutating error.
----> System  triggers.

6. PAKAGES:->
---
----> Global Variable.
----> Serially_resuable pragma.
----> Overloading procedures.
----> Forward Declaration.

7. TYPES ARE USED IN PACKAGES:->

---

----> PL/SQL record Collection.

===>COLLECTION:==>

---

----> Index by table.
----> Nested table.
----> Varray.

8. BLOCK BIND:->

---

----> Bulk Collection clause.
----> Forall statements.
----> Indices of clause(10g).
----> Sql%bulk_rowcount.
----> Bulk Execeptions.

9. REFURSOR(OR) DYNAMIC CURSOR(OR) CURSOR VARIABLE:->

---

----> Strong refursor.
----> Weak refursor.
----> Sys_refursor.
----> Passing refursor as parameter to the procedurce.

10. LOCAL SUB PROGRAM:->

---

----> Local Procedurces.
----> Local Functions.

----> Passing types as parameter to the local sub progarms.

11. UTL-FILE PACKAGE:->

---

12.SQ * LOADER:->

---

---->Control file, bad file, discard file.

13. LOBS(LARGE OBJECTS):->

---

----> Clob, blob, bfile, data types.

14. WHERE CURRENT OF FOR UPDATE CLAUSES ARE USED IN EXPLICIT CURSOR:->

---

15. AVOIDING MULATING ERROR BY USING COMPOUND TRRIGER:->

---

16. DYNAMIC SQL:->

---

17. ORACLE 11g ORACLE 12c FEATURES:->

---

==========

Date:- 20-10-2019

---

==========

* PL/SQL ==> Pl/Sql in "Procedural language" Exectension.

---

----> PL(Procedural Language) /

----> SQL( Structure Query Language)

* Version:->

---

----> Oracle 6.0 Introduces pl/sql.

----> Oracle 6.0 Pl/sql 1.0

----> Oracle 7.0 Pl/sql 2.0

----> Oracle 8.0 Pl/sql 8.0

* Block Structure:->

---

declare [optinal]

----> variable declaration, cursor,user defind exception.

begin [Mandatory]

----> DML.TCL

Statement;

---->

Select.....into clause;

----> Conditional,

control Statement;

Exception [optional]

-----> Handle

Runtime errors.

End;

[Mandatory]

24/10/2019

==========

"

Variable Attributes "

==========

* Declaring a Variables:->

==========

Syntax:->

---

variablename datatype(size);

* Storing a values into variables:->

==========

Example:->

---

```
set serveroutput on;
declare
a number(10);
begin
a:=90;
dbms_output.put_line(a);
end;
/
```

Output:-

---

```
90
PL/SQL procedure successfully completed.
```

* Display Message (or) display variable :-

==========

Syntax:-

---

```
dbms_output.put_line('Message');
(or)
dbms_output.put_line('Variable');
```

Example:-

---

```
set serveroutput on;
```

```
begin
dbms_output.put_line('Welcome Md Shamshad Alam N.iT');
end;
/
```

Output:-

---

```
Welcome Md Shamshad Alam N.iT
PL/SQL procedure successfully completed.
```

* Ouput Display Message:->

==========

Syntax:->

---

```
        set sever output on;
```

Example 1:->

---

```
set serveroutput on;
declare
a number(10);
begin
a:=80;
dbms_output.put_line(a);
end;
/
```

Output:-

---

```
80
PL/SQL procedure successfully completed
```

Example 2:->

```
-----------------
set serveroutput on;
declare
a number(10):=&a;
begin
dbms_output.put_line(a);
end;
/
```

Output:-

```
-----------
Enter value for a: 50
old   2: a number(10):=&a;
new   2: a number(10):=50;
50
PL/SQL procedure successfully completed.
```

* Select.....into clause:->
  ==========

Syntax:-

```
---------
  select columnname1,columnname2..... into
variablename1,variablename2..... from table name where condition;
```

25/10/2019

========

Question 1:- Write a PL/QSL progarm for user inter Employee no than display name of the employee and salary from emp table.

==========

```
set serveroutput on;
declare
v_ename varchar2(10);
v_sal number(10);
begin
select ename,sal into
v_ename,v_sal from emp
where sno=&sno;
dbms_output.put_line(v_ename||' '||v_sal);
end;
/
```

Output:-

---

```
Enter value for sno: 7902
old   7: where sno=&sno;
new   7: where sno=7902;
FORD 3000
PL/SQL procedure successfully completed.
```

* Variable name constant data type(size): =values.

==========

Syntax:->

---

variablename datatype(size) not null:=value;

Syntax:->

---

variablename Constant datatype(size) :=value;

Example:->

---

```
set serveroutput on;
declare
a number(10) not null:=50;
b constant number(10):=8;
begin
dbms_output.put_line(a);
dbms_output.put_line(b);
end;
/
```

Output:-

---

```
50
8
PL/SQL procedure successfully completed.
```

Question 2:- Write a pl/sql program which is use to maximum sal from emp table and also display maximum salary.

```
==========================================================
========================
set serveroutput on;
declare
v_sal number (10);
begin
select max(sal) into v_sal from emp;
dbms_output.put_line(v_sal);
end;
/
```

Output:-

```
-----------
5000
```

PL/SQL procedure successfully completed.

Question 3:- In pl/sql expression we are not allow to use group function_
Example like this:-

```
==========================================================
=========
set serveroutput on;
declare
a number (10);
b number (10);
c number (10);
begin
a:=90;
b:=5;
c:=greatest(a,b);                    Notes---->[C:=Max(a,b) wrong]
```

```
dbms_output.put_line(c);
end;
/
```

Output:-

```
----------
90
PL/SQL procedure successfully completed.
```

Question 4:- Write a pl/sql progarm print of date and display.

==========

```
set serveroutput on;
declare
a date;
begin
a:=to_date('12/07/07','DD/MM/YY')+1;
dbms_output.put_line(a);
end;
/
```

Output:-

```
13-JUL-07
PL/SQL procedure successfully completed.
```

Question 5:- Write a pl/sql program lower to upper print & display.

==========

```
set serveroutput on;
declare
x varchar2(10);
begin
```

```
x:=upper('shamshad');
dbms_output.put_line(x);
end;
/
```

Output:-

```
----------
SHAMSHAD
PL/SQL procedure successfully completed.
```

26/10/2019

=========

* Variable Attributes:->

==========

(1) Column Level Attribute

(2) Row Level Attribute

1. Column Level Attribute:->

==========

Syntax:->

```
-----------
 variablename tablename columnname %type;
```

Example:->

```
-------------
set serveroutput on;
declare
v_ename emp.ename%type;
v_sal emp.sal%type;
```

```
v_hiredate emp.hiredate%type;
begin
select ename,sal,hiredate into
v_ename,v_sal,v_hiredate from emp
where sno=&sno;
dbms_output.put_line(v_ename ||' '|| v_sal ||' '|| v_hiredate);
end;
/
```

Output:->

---

```
Enter value for sno: 7902
old   8: where sno=&sno;
new   8: where sno=7902;
FORD 3000 03-DEC-81
PL/SQL procedure successfully completed.
```

2. Row Level Attributes:->

==========

Syntax:->

---

```
  variablename tablename %rowtype;
```

----> This variable also called as record type variable(%rowtype)

Example 1:->

==========

```
set serveroutput on;
declare
```

```
i emp% rowtype;
begin
select ename,sal,hiredate into
i.ename,i.sal,i.hiredate from emp
where sno=&sno;
dbms_output.put_line(i.ename ||' '|| i.sal ||' '|| i.hiredate);
end;
/
```

Output:->

---

```
Enter value for sno: 7902
old   6: where sno=&sno;
new   6: where sno=7902;
FORD 3000 03-DEC-81
PL/SQL procedure successfully completed.
```

Example 2:->

---

```
set serveroutput on;
declare
i emp% rowtype;
begin
select *into i from emp
where sno=&sno;
dbms_output.put_line(i.ename ||' '|| i.sal ||' '|| i.hiredate ||' '|| i.deptno);
end;
/
```

Output:->
---
Enter value for sno: 7902
old 5: where sno=&sno;
new 5: where sno=7902;
FORD 3000 03-DEC-81 20
PL/SQL procedure successfully completed.

* Conditional Statement :-> (if)
==========

(1) if
(2) if-else
(3) ifsif ---------->{ Not(X) [ else if ] --> It is wrong...}

1. if :->
======
Syntax:->
---
if condition then stmts;
end if;

2. if-else:->
=========
Syntax:->
---
if condition then
stmts;
else
stmts;

```
 end if;
```

3. ifsif :-> To check more than number of conditions when we are using elsif.

=======

Syntax:->

---

```
 if condition then
 stmts;
 elsif condition then
 stmts;
 elsif condition3 then
 stmts;
 ----------
 ----------
 ----------
 else
 stmts;
 end if;
  /
```

Example :->

---

```
set serveroutput on;
declare
v_deptno number(10);
begin
select deptno into v_deptno
from dept where deptno = &sno;
if v_deptno=10 then
```

```
dbms_output.put_line('ten');
elsif v_deptno = 20 then
dbms_output.put_line('twenty');
elsif v_deptno =30 then
dbms_output.put_line('thirty');
else
dbms_output.put_line('others');
end if;
end;
/
```

Ouput : ->

---

```
Enter value for sno: 40
old   5: from dept where deptno = &sno;
new   5: from dept where deptno = 40;
others
PL/SQL procedure successfully completed.

SQL> /
Enter value for sno: 90
old   5: from dept where deptno = &sno;
new   5: from dept where deptno = 90;

ERROR at line 1:
ORA-01403: no data found
ORA-06512: at line 4
```

28/10/2019

=========

Notes :-> When ever selecting into class clause try to return multiple record or try to return multiple values from singal column at
a time then oracle sever return on error ORA- 1422: Exact exact fetch returns more than requested number of rows.

Example :->

---

```
set serveroutput on;
declare
i emp%rowtype;
begin
select * into i from emp where deptno=10;
dbms_output.put_line(i.ename||''||i.sal||' '||i.deptno);
end;
/
```

Output :->

---

```
ERROR at line 1:
ORA-01422: exact fetch returns more than requested number of rows
ORA-06512: at line 4
```

* Control Statement (or) Loops :-> PL/SQL having 3 types of loops these are

==========

(1) Simple loop
(2) While loop
(3) for loop

Syntax :->
---
```
 loop
 stmts;
 end loop;
```

Example:->
---
```
set serveroutput on;
begin
loop
dbms_output.put_line('Wel Come to Shamshad ');
end loop;
end;
/
```

* To exits from infinite loop then oracle provide following  two method :->
==========
Method 1:-> Exit when true condition.
---
 * One to ten number is display :->
 ==========
```
set serveroutput on;
declare
n number(10):=1;
begin
loop
dbms_output.put_line(n);
```

```
exit when n>=10;
n:=n+1;
end loop;
end;
/
```

Output :->

```
1
2
3
4
5
6
7
8
9
10
PL/SQL procedure successfully completed.
```

Method 2:-> (Using if) :->

Syntax :->

```
if true condition then exit;
end if;
```

Example :->

```
set serveroutput on;
declare
n number(10):=1;
begin
loop
dbms_output.put_line(n);
if n>=10 then exit;
end if;
n:=n+1;
end loop;
end;
/
```

Output :->

```
------------
1
2
3
4
5
6
7
8
9
10
PL/SQL procedure successfully completed.
```

* While loop :->

==========

Syntax :->

```
while(condition)
loop
stmts;
end loop;
```

Example :->

```
set serveroutput on;
declare
n number(10):=1;
begin
while(n<=10)
loop
dbms_output.put_line(n);
n:=n+1;
end loop;
end;
/
```

Output :->

```
1
2
3
4
5
6
```

```
7
8
9
10
PL/SQL procedure successfully completed.
```

* For loop :->
  ========
  Syntax:->
  ----------
  for index variable in lowerbound..
  upper bound
  loop
  stmts;
  end loop;

Example 1 :->
---

```
set serveroutput on;
declare
n number(10):=1;
begin
for n in 1..10
loop
dbms_output.put_line(n);
end loop;
end;
/
```

Output :->

---

1
2
3
4
5
6
7
8
9
10

PL/SQL procedure successfully completed.

Example 2 :->

---

```
set serveroutput on;
declare
n number(10):=1;
begin
for n in reverse 1..10
loop
dbms_output.put_line(n);
end loop;
end;
/
```

Output :->

---

```
1
2
3
4
5
6
7
8
9
10
PL/SQL procedure successfully completed.
```

* Without declare Method 1 to 10 Print :->

==========

Example :->

```
set serveroutput on;
begin
for n in 1..10
loop
dbms_output.put_line(n);
end loop;
end;
/
```

Output :->

```
------------
1
2
3
```

```
4
5
6
7
8
9
10
PL/SQL procedure successfully completed.
```

Program :-> Wirte a PL/SQL program which is used to for 1 to 50 numbers into the following table by using for loops.

---

```
 set serveroutput on;
 create table test(sno number(10));
 Table created.

 begin for i in 1..50
 loop
 insert into test values(i);
 end loop;
 end;
   /
PL/SQL procedure successfully completed.
SQL> select * from test;
```

Output :->

---

```
 SNO
----
```

```
  1
  2
  3
  4
  5
  6
  7
  8
  9
 10
 11

 SNO
----
 12
 13
 14
 15
 16
 17
 18
 19
 20
 21
 22

 SNO
----
 23
```

```
 24
 25
 26
 27
 28
 29
 30
 31
 32
 33

 SNO
----
 34
 35
 36
 37
 38
 39
 40
 41
 42
 43
 44

 SNO
----
 45
 46
```

```
 47
 48
 49
 50

50 rows selected.
```

29/10/2019

==========

* PL/SQL Data types & Variable :->

 ========================

(1) Support all sql data types (Scalar data type) + (Boolean data type)
(2) Composite data types.
(3) Large object(loops)--> Clob,blob,bfile data types.
(4) Reference objects.
(5) Bind variable (or) Non- PL/SQL variables.

 *  Bind variable (or) Non- PL/SQL variables :->

  ================================

 Step 1 :-> Create bind variable.
 Step 2 :-> Using  bind variable.
 Step 3 :-> Display values from  bind variable.

Step 1 :-> Create bind variable :->

========================

Syntax :->

---

 variable varname datatype;

Step 2 :-> Using  bind variable :->
==========
Syntax :->
---
Step 2 :->   :bind variablename

 Step 3 :-> Display values from  bind variable :->
==========
Syntax :->
---
print variablename

Example :->
---
```
set serveroutput on;
  1  declare
  2  a number(10):=1000;
  3  begin
  4  :g:=a/2;
  5* end;
SQL> /
PL/SQL procedure successfully completed.
SQL> print g;
         G
------------
       500
============
```

**

CURSOR **

==========

----> To Process Multiple record.

----> Record by record Process.

----> All Relational database having two types static cursor these are.....
(1) Implicit cursor.
(2) Explicit cursor.

1. Explicit cursor :->
---
* Explicit cursor life cycle :->
==========
1. declare
2. open
3. fetch
4. close

1. declare :->
=========
Syntax :->
---
cursor cursorname is select * from tablename where condition;

Example :->

```
 --------------
  declare
  cursor c1 is select * from emp
  where job = 'CLEARK';

 2. Open :->
 =========
 Syntax :->
-------------
 open cursorname;

 3. Fetch :-> (Fetch data from cursor).
  --------------
 Syntax :->
-------------
 fetch cursorname into variablename,variablename2,.....;

30/10/2019
========
 4. Close:->
 ========
 Syntax :->
 -------------
  close cursorname;

* Program :->  Only output this program.
   =======
set serveroutput on;
declare
```

```
cursor c1 is select ename,sal from emp;
v_ename varchar2(10);
v_sal number(10);
begin
open c1;
fetch c1 into v_ename,v_sal;
dbms_output.put_line(v_ename||' '||v_sal);
fetch c1 into v_ename,v_sal;
dbms_output.put_line('My second employee name is :'||' '||v_ename||'
'||v_sal);
fetch c1 into v_ename,v_sal;
dbms_output.put_line(v_ename||' '||v_sal);
dbms_output.put_line('My'||' '||v_ename||' '||'employee salary is:'||' '||v_sal);
close c1;
end;
/
```

output

```
=======
SMITH 800
My second employee name is : ALLEN 1600
WARD 1250
My WARD employee salary is: 1250
PL/SQL procedure successfully completed.
```

2. Explicit cusor attributes :-> Ever Explicit cursor having following four attributes these are.....

==========

(1) %notfound

(2) %found

(3) %isopen
(4) %rowcount

Syntax :->
---
cursorname% attributesname;

1. %notfound :->
==========
Syntax :->
---
cursorname %notfound

Program1 :-> Write a PL/SQL Explicit cursor progarm which is use to display all employee name and than salary
========  from emp table by using %notfound Attributes..

```
set serveroutput on;
declare
cursor c1 is select ename,sal from emp;
v_ename varchar2(10);
v_sal number(10);
begin
open c1;
loop
fetch c1 into v_ename,v_sal;
exit when c1%notfound;
dbms_output.put_line(v_ename||' '||v_sal);
end loop;
```

```
close c1;
end;
/
```

Output

```
=======
SMITH 800
ALLEN 1600
WARD 1250
JONES 2975
MARTIN 1250
BLAKE 2850
CLARK 2450
SCOTT 3000
KING 5000
TURNER 1500
ADAMS 1100
JAMES 950
FORD 3000
MILLER 1300
PL/SQL procedure successfully completed.
```

Program 2 :-> Write a PL/SQL Program which is used to calculate total salary from emp table without using sum function ().

```
==========
set serveroutput on;
 declare
 cursor c1 is select sal from emp;
 v_sal number(10);
```

```
n number(10):=0;
begin
open c1;
loop
fetch c1 into v_sal;
exit when c1%notfound;
n:=n+v_sal;
end loop;
dbms_output.put_line('totol salary is:'||' '||n);
close c1;
end;
/
```

Output:
=======
totol salary is: 29025
PL/SQL procedure successfully completed.

Notes:-> [ n:=n+nvl(v-sal,0); ]
======

31/10/2019
==========
Program 3 :-> Write a PL/SQL Program which is used to display first five highest salary employee from emp table by using rowcount attributes.
==========

```
set serveroutput on;
declare
cursor c1 is select ename,sal from emp order by sal desc;
```

```
v_ename varchar2(10);
v_sal number(10);
begin
open c1;
loop
fetch c1 into v_ename,v_sal;
dbms_output.put_line(v_ename||' '||v_sal);
exit when c1%rowcount >=5;
end loop;
close c1;
end;
/
```

Output

```
=======
KING 5000
FORD 3000
SCOTT 3000
JONES 2975
BLAKE 2850
PL/SQL procedure successfully completed.
```

Program 4 :-> Write a PL/SQL Explicit cursor program which is used to display even number of record from emp table by using rowcount attributes.

```
==========
set serveroutput on;
declare
cursor c1 is select ename,sal from emp;
v_ename varchar2(10);
```

```
v_sal number(10);
begin
open c1;
loop
fetch c1 into v_ename,v_sal;
exit when c1%notfound;
if mod(c1%rowcount,2)=0 then
dbms_output.put_line(v_ename||' '||v_sal);
end if;
end loop;
close c1;
end;
/
```

Output :

```
=======
ALLEN 1600
JONES 2975
BLAKE 2850
SCOTT 3000
TURNER 1500
JAMES 950
MILLER 1300
PL/SQL procedure successfully completed.
```

For odd :-> if mod(c1%rowcount,2)=0

```
======
```

For Even :-> if mod(c1%rowcount,2)=1

======

Example :->

---

```
set serveroutput on;
 declare
 a number(10);
 b boolean;
 begin
 a:=90;
 b:=true;
 dbms_output.put_line(a);
 end;
 /
```

Output :

=======

```
  90
```

2. %rowcount :->

==========

Syntax :->

---

```
cursorname %newcount
```

Example :->

---

```
set serveroutput on;
declare
```

```
cursor c1 is select ename,sal from emp;
v_ename varchar2(10);
v_sal number(10);
begin
open c1;
fetch c1 into v_ename,v_sal;
dbms_output.put_line(v_ename||' '||v_sal);
fetch c1 into v_ename,v_sal;
dbms_output.put_line(v_ename||' '||v_sal);
dbms_output.put_line('Number of records
fetched from the cursor memory area is:' ||' '||c1%rowcount);
close c1;
end;
/
```

Output:

```
======
SMITH 800
ALLEN 1600
Number of records
fetched from the cursor memory area is: 2
PL/SQL procedure successfully completed.
```

Notes :-> By using cursor we can also transport data from one oracle into another table.

=====

Program 4 :-> Write a PL/SQL Explicit cursor program which is used to transfor employees who getting more

========= then 3000 sal from emp table into another table.

```
set serveroutput on;
declare
cursor c1 is select ename,sal from emp where sal>2000;
v_ename varchar2(10);
v_sal number(10);
n number(10);
begin
open c1;
loop
fetch c1 into v_ename,v_sal;
exit when c1%notfound;
n:=c1%rowcount;
insert(n,v_name,v_sal);
end loop;
close c1;
end;
/
```

Output :->

---

01/11/2019

==========

Program 4 :-> Write a PL/SQL Program which is used to display 5th records emp table by using row count attribute.

=========.

```
set serveroutput on;
```

```
declare
cursor c1 is select * from emp;
i emp%rowtype;
begin
open c1;
loop
fetch c1 into i;
exit when c1%notfound;
if c1%rowcount=5 then
dbms_output.put_line(i.ename||' '||i.sal||' '||i.deptno);
end if;
end loop;
end;
/
```

Output :->

---

```
MARTIN 1250 30
PL/SQL procedure successfully completed.
```

3. %found :->

  =========

Syntax :->

---

```
cursorname %found
```

Example :->

---

```
set serveroutput on;
```

```
declare
cursor c1 is select * from emp where ename='&name';
i emp%rowtype;
begin
open c1;
fetch c1 into i;
if c1%found then
dbms_output.put_line('u r employee exist'||' '||i.ename||' '||i.sal);
elsif c1%notfound then
dbms_output.put_line(' u r employee does not exist');
end if;
close c1;
end;
/
```

Output :->

```
----------
Enter value for name: SMITH
old   2: cursor c1 is select * from emp where ename='&name';
new   2: cursor c1 is select * from emp where ename='SMITH';
u r employee exist SMITH 800
PL/SQL procedure successfully completed.

SQL> /
Enter value for name: abc
old   2: cursor c1 is select * from emp where ename='&name';
new   2: cursor c1 is select * from emp where ename='abc';
u r employee does not exist
PL/SQL procedure successfully completed.
```

4. %isopen :->
==========
Syntax :->
---
cursorname %isopen

Example :->
---
```
set serveroutput on;
declare
cursor c1 is select * from emp;
i emp%rowtype;
begin
open c1;
if c1%isopen then
dbms_output.put_line(' Cursor is already opened');
else
dbms_output.put_line('cursor is not  opened');
end if;
end;
/
```

Output :->
---
Cursor is already opened
PL/SQL procedure successfully completed.

* Eliminatinng explicit cursor life cycle (or) Cursor..... for loop :->

==========

Syntax :->

---

```
for indexvarname in cursorname
loop
stmts;
end loop;
```

----> this cursor for loop use been executebale session is PL/SQL block.

Notes :-> In cursor for loop index variable internally behaves like a records type variable with in a (%rowtype).

---

Shortcut Method :->

---

Example :->

---

```
set serveroutput on;
declare
cursor c1 is select * from emp;
begin
for i in c1
loop
dbms_output.put_line(i.ename||' '||i.sal);
end loop;
end;
/
```

Output :->
---
SMITH 800
ALLEN 1600
WARD 1250
JONES 2975
MARTIN 1250
BLAKE 2850
CLARK 2450
SCOTT 3000
KING 5000
TURNER 1500
ADAMS 1100
JAMES 950
FORD 3000
MILLER 1300
PL/SQL procedure successfully completed.

Notes :-> we can also eliminate declare rection of the cursor by using cursor for loop, in this case we must be
---------- specify cursor select statement in place of cursor name with in cursor for loop.

Syntax :->
---
for indexname in (select stmt)
loop
stmts;
end loop;

Example :->
---