**PL/SQL**

**REFERENCE LINK:**

**https://www.javatpoint.com/pl-sql-tutorial**

Set serveroutput on

SQL> declare

2 a integer:=10;

3 begin

4 dbms\_output.put\_line(a);

5 end;

6 /

10

PL/SQL procedure successfully completed.

SQL> declare

2 a varchar(10):='hello';

3 begin

4 dbms\_output.put\_line(a);

5 end;

6 /

hello

PL/SQL procedure successfully completed.

SQL> declare

2 a integer:=10;

3 begin

4 dbms\_output.put\_line('value of a:'||a);

5 end;

6 /

value of a:10

PL/SQL procedure successfully completed.

SQL> declare

2 pi constant number:=3.14;

3 begin

4 dbms\_output.put\_line(pi);

5 end;

6 /

3.14

PL/SQL procedure successfully completed.

SQL> declare

2 a integer:=10;

3 b integer:=20;

4 c integer;

5 begin

6 c:=a+b;

7 dbms\_output.put\_line(c);

8 end;

9 /

30

**IF-ELSE**

PL/SQL procedure successfully completed.

SQL> declare

2 a integer:=30;

3 begin

4 if(a<20) then

5 dbms\_output.put\_line('a is less than 20');

6 else

7 dbms\_output.put\_line('a is not less than 20');

8 end if;

9 end;

10 /

a is not less than 20

**CASE STATEMENT**

PL/SQL procedure successfully completed.

SQL> DECLARE

2 grade char(1) := 'A';

3 BEGIN

4 CASE grade

5 when 'A' then dbms\_output.put\_line('Excellent');

6 when 'B' then dbms\_output.put\_line('Very good');

7 when 'C' then dbms\_output.put\_line('Good');

8 when 'D' then dbms\_output.put\_line('Average');

9 when 'F' then dbms\_output.put\_line('Passed with Grace');

10 else dbms\_output.put\_line('Failed');

11 END CASE;

12 END;

13 /

Excellent

**LOOP**

PL/SQL procedure successfully completed.

SQL> DECLARE

2 i NUMBER := 1;

3 BEGIN

4 LOOP

5 EXIT WHEN i>10;

6 DBMS\_OUTPUT.PUT\_LINE(i);

7 i := i+1;

8 END LOOP;

9 END;

10 /

1

2

3

4

5

6

7

8

9

10

**WHILE LOOP**

PL/SQL procedure successfully completed.

SQL> DECLARE

2 i INTEGER := 1;

3 BEGIN

4 WHILE i <= 10 LOOP

5 DBMS\_OUTPUT.PUT\_LINE(i);

6 i := i+1;

7 END LOOP;

8 END;

9 /

1

2

3

4

5

6

7

8

9

10

**FOR LOOP**

PL/SQL procedure successfully completed.

SQL> DECLARE

2 VAR1 NUMBER;

3 BEGIN

4 VAR1:=10;

5 FOR VAR2 IN 1..10

6 LOOP

7 DBMS\_OUTPUT.PUT\_LINE (VAR1\*VAR2);

8 END LOOP;

9 END;

10 /

10

20

30

40

50

60

70

80

90

100

PL/SQL procedure successfully completed.

**PROCEDURE**

SQL> create table proc(id number(10),name varchar(10));

Table created.

SQL> create or replace procedure "INSERTUSER"

(id IN NUMBER,

name IN VARCHAR2)

is

begin

insert into proc values(id,name);

end;

/

Procedure created.

SQL> BEGIN

insertuser(101,'Rahul');

dbms\_output.put\_line('record inserted successfully');

END;

/

record inserted successfully

PL/SQL procedure successfully completed.

SQL> select \* from proc;

ID NAME

---------- ----------

101 Rahul

**FUNCTION**

SQL> create or replace function adder(n1 in number, n2 in number)

2 return number

3 is

4 n3 number(8);

5 begin

6 n3 :=n1+n2;

7 return n3;

8 end;

9 /

Function created.

SQL> DECLARE

2 n3 number(2);

3 BEGIN

4 n3 := adder(11,22);

5 dbms\_output.put\_line('Addition is: ' || n3);

6 END;

7 /

Addition is: 33

PL/SQL procedure successfully completed.

SQL> DECLARE

2 a number;

3 b number;

4 c number;

5 FUNCTION findMax(x IN number, y IN number)

6 RETURN number

7 IS

8 z number;

9 BEGIN

10 IF x > y THEN

11 z:= x;

12 ELSE

13 Z:= y;

14 END IF;

15

16 RETURN z;

17 END;

18 BEGIN

19 a:= 23;

20 b:= 45;

21

22 c := findMax(a, b);

23 dbms\_output.put\_line(' Maximum of (23,45): ' || c);

24 END;

25 /

Maximum of (23,45): 45

PL/SQL procedure successfully completed.

SQL> select \* from emp;

ID NAME AGE ADDRESS SALARY

---------- ---------- ---------- ---------- ----------

30 sai 26 mumbai 150000

40 sam 27 hyderabad 200000

50 tom 29 pune 40630

SQL> create or replace function totalemp

2 return number is

3 total number(10):=0;

4 begin

5 select count(\*) into total from emp;

6 return total;

7 end;

8 /

Function created.

SQL> declare

2 c number(20);

3 begin

4 c:=totalemp();

5 dbms\_output.put\_line('Total no of emp:'||c);

6 end;

7 /

Total no of emp:3

PL/SQL procedure successfully completed.

SQL> DECLARE

2 num number;

3 factorial number;

4

5 FUNCTION fact(x number)

6 RETURN number

7 IS

8 f number;

9 BEGIN

10 IF x=0 THEN

11 f := 1;

12 ELSE

13 f := x \* fact(x-1);

14 END IF;

15 RETURN f;

16 END;

17

18 BEGIN

19 num:= 6;

20 factorial := fact(num);

21 dbms\_output.put\_line(' Factorial '|| num || ' is ' || factorial);

22 END;

23 /

Factorial 6 is 720

PL/SQL procedure successfully completed.

**IMPLICIT CURSOR**

SQL> DECLARE

total\_rows number(2);

BEGIN

UPDATE emp

SET salary = salary + 5000;

IF sql%notfound THEN

dbms\_output.put\_line('no customers updated');

ELSIF sql%found THEN

total\_rows := sql%rowcount;

dbms\_output.put\_line( total\_rows || ' customers updated ');

END IF;

END;

/

3 customers updated

PL/SQL procedure successfully completed.

SQL> select \* from emp;

ID NAME AGE ADDRESS SALARY

---------- ---------- ---------- ---------- ----------

30 sai 26 mumbai 155000

40 sam 27 hyderabad 205000

50 tom 29 pune 45630

**EXPLICIT CURSOR**

SQL> DECLARE

c\_id emp.id%type;

c\_name emp.name%type;

c\_addr emp.address%type;

CURSOR c\_emp is

SELECT id, name, address FROM emp;

begin

open c\_emp;

loop

FETCH c\_emp into c\_id, c\_name, c\_addr;

EXIT WHEN c\_emp%notfound;

dbms\_output.put\_line(c\_id || ' ' || c\_name || ' ' || c\_addr);

end loop;

close c\_emp;

end;

/

30 sai mumbai

40 sam hyderabad

50 tom pune

PL/SQL procedure successfully completed.

**EXCEPTION HANDLING**

SQL> DECLARE

c\_id emp.id%type:=10;

c\_name emp.name%type;

c\_addr emp.address%type;

begin

SELECT name, address into c\_name,c\_addr FROM emp where id=c\_id;

dbms\_output.put\_line('Name:'||c\_name);

dbms\_output.put\_line('Address:'||c\_addr);

exception

when no\_data\_found then dbms\_output.put\_line('no such customer');

when others then dbms\_output.put\_line('error');

end;

/

no such customer

PL/SQL procedure successfully completed.

SQL> declare

c\_id emp.id%type:=30;

c\_name emp.name%type;

c\_addr emp.address%type;

begin

SELECT name, address into c\_name,c\_addr FROM emp where id=c\_id;

dbms\_output.put\_line('Name:'||c\_name);

dbms\_output.put\_line('Address:'||c\_addr);

exception

when no\_data\_found then dbms\_output.put\_line('no such customer');

when others then dbms\_output.put\_line('error');

end;

/

Name:sai

Address:mumbai

PL/SQL procedure successfully completed.

**USER DEFINED**

SQL> declare

c\_id emp.id%type:=&id;

c\_name emp.name%type;

c\_addr emp.address%type;

ex\_invalid\_id exception;

begin

if c\_id<=0 then

raise ex\_invalid\_id;

else

select name,address into c\_name,c\_addr from emp where id=c\_id;

dbms\_output.put\_line('name:'||c\_name);

dbms\_output.put\_line('address:'||c\_addr);

end if;

exception

when ex\_invalid\_id then dbms\_output.put\_line('id should be greater than zero');

when no\_data\_found then dbms\_output.put\_line('no such customer');

when others then dbms\_output.put\_line('error');

end;

/

Enter value for id: -6

old 2: c\_id emp.id%type:=&id;

new 2: c\_id emp.id%type:=-6;

id should be greater than zero

PL/SQL procedure successfully completed.

SQL> /

Enter value for id: 30

old 2: c\_id emp.id%type:=&id;

new 2: c\_id emp.id%type:=30;

name:sai

address:mumbai

PL/SQL procedure successfully completed.

**TRIGGERS**

SQL> CREATE OR REPLACE TRIGGER display\_salary\_changes

BEFORE DELETE OR INSERT OR UPDATE ON emp

FOR EACH ROW

WHEN (NEW.ID > 0)

DECLARE

sal\_diff number;

BEGIN

sal\_diff := :NEW.salary - :OLD.salary;

dbms\_output.put\_line('Old salary: ' || :OLD.salary);

dbms\_output.put\_line('New salary: ' || :NEW.salary);

dbms\_output.put\_line('Salary difference: ' || sal\_diff);

END;

/

Trigger created.

SQL> DECLARE

total\_rows number(2);

BEGIN

UPDATE emp

SET salary = salary + 5000;

IF sql%notfound THEN

dbms\_output.put\_line('no customers updated');

ELSIF sql%found THEN

total\_rows := sql%rowcount;

dbms\_output.put\_line( total\_rows || ' customers updated ');

END IF;

END;

/

Old salary: 155000

New salary: 160000

Salary difference: 5000

Old salary: 205000

New salary: 210000

Salary difference: 5000

Old salary: 45630

New salary: 50630

Salary difference: 5000

3 customers updated

PL/SQL procedure successfully completed.