

## LOOPS

### WHILE LOOP

```
19 DECLARE
20     a number(2) := 10;
21 BEGIN
22     WHILE a < 20 LOOP
23         dbms_output.put_line('value of a: ' || a);
24         a := a + 3;
25     END LOOP;
26 END;
```

```
Statement processed.
value of a: 10
value of a: 13
value of a: 16
value of a: 19
```

GO TO Statement

```
29 --GO TO
30 DECLARE
31     a number(2) := 1;
32 BEGIN
33     <<loopstart>>
34     -- while loop execution
35     WHILE a < 20 LOOP
36         dbms_output.put_line ('value of a: ' || a);
37         a := a + 3;
38         IF a = 15 THEN
39             a := a + 3;
40             GOTO loopstart;
41         END IF;
42     END LOOP;
43 END;
44 /
```

```
Statement processed.
value of a: 1
value of a: 4
value of a: 7
value of a: 10
value of a: 13
value of a: 16
value of a: 19
```

NESTED

```
45 /
46 --NESTED
47 DECLARE
48     i number(3);
49     j number(3);
50 BEGIN
51     i := 2;
52     LOOP
53         j := 2;
54         LOOP
55             exit WHEN ((mod(i, j) = 0) or (j = i));
56             j := j + 1;
57         END LOOP;
58         IF (j = i ) THEN
59             dbms_output.put_line(i || ' is prime');
60         END IF;
61         i := i + 1;
62         exit WHEN i = 20;
63     END LOOP;
64 END;
65 /
```

```
statement processed.
2 is prime
3 is prime
5 is prime
7 is prime
11 is prime
13 is prime
17 is prime
19 is prime
```

FUNCTIONS

SQL Worksheet

ClearFindActionsSaveRun

```
1  --PROCEDURES
2  CREATE OR REPLACE PROCEDURE GuneetGreetsYou (p_name IN VARCHAR2)
3  IS
4  BEGIN
5  dbms_output.put_line ('Good Morning ' || p_name);
6  END;
7  /
8  EXEC GuneetGreetsYou ('Priyanka Maam');
```

Procedure created.

Statement processed.

Good Morning Priyanka Maam

PROCEDURES

```
13 DECLARE
14     a number;
15     b number;
16     c number;
17 PROCEDURE findMin(x IN number, y IN number, z OUT number) IS
18 BEGIN
19     IF x < y THEN
20         z:= x;
21     ELSE
22         z:= y;
23     END IF;
24 END;
25 BEGIN
26     a:= 23;
27     b:= 45;
28     findMin(a, b, c);
29     dbms_output.put_line(' Minimum of (23, 45) : ' || c);
30 END;
31 /
```

Statement processed.

Minimum of (23, 45) : 23

LABELLING LOOP

```
1  DECLARE
2      i number(1);
3      j number(1);
4  BEGIN
5      << outer_loop >>
6      FOR i IN 1..3 LOOP
7          << inner_loop >>
8          FOR j IN 1..3 LOOP
9              dbms_output.put_line('i is: ' || i || ' and j is: ' || j);
10             END loop inner_loop;
11         END loop outer_loop;
12     END;
13 /
```

Statement processed.

i is: 1 and j is: 1  
i is: 1 and j is: 2  
i is: 1 and j is: 3  
i is: 2 and j is: 1  
i is: 2 and j is: 2  
i is: 2 and j is: 3  
i is: 3 and j is: 1  
i is: 3 and j is: 2  
i is: 3 and j is: 3

PROCEDURES

```

30
31 DECLARE
32     a number;
33 PROCEDURE squareNum(x IN OUT number) IS
34 BEGIN
35     x := x * x;
36 END;
37 BEGIN
38     a:= 15;
39     squareNum(a);
40     dbms_output.put_line(' Square of (15): ' || a);
41 END;
42 /
43

```

Statement processed.  
Square of (15): 225

## SALARY MID RANGE

```

DECLARE
mid_salary NUMBER;
emp_salary NUMBER;
emp_num NUMBER;

BEGIN
mid_salary:=5000;
emp_num:=2;
SELECT salary INTO emp_salary
FROM emp_details
WHERE emp_id=emp_num;

IF emp_salary> mid_salary THEN
UPDATE emp_details
SET salary=1.8*salary
WHERE emp_id=emp_num;
dbms_output.put_line('salary more than mid range');
ELSE
UPDATE emp_details
SET salary=mid_salary
WHERE emp_id=emp_num;
dbms_output.put_line('salary less than midrange');
END IF;
END;

```

Statement processed.  
salary less than midrange

## IMPLICIT CURSOR TO UPDATE SALARY

```

1 DECLARE
2
3 c_id customer.cust_id%type;
4 c_age customer.cust_age%type;
5 c_add customer.cust_add%type;
6 c_salary customer.cust_salary%type;
7 total_rows NUMBER;
8 BEGIN
9
10 UPDATE customer
11 SET cust_salary=cust_salary+5000;
12
13 IF SQL%NOTFOUND THEN
14 dbms_output.put_line('no customers updated');
15 ELSIF SQL%FOUND THEN
16 total_rows:=sql%rowcount;
17 dbms_output.put_line(total_rows||' customers updated');
18 END IF;
19 END;
20 /

```

Statement processed.  
3 customers updated

## EXPLICIT CURSOR

```

1 declare
2 cursor C1 is
3 SELECT emp_id, emp_name, emp_salary
4 FROM emp_detail;
5 empId emp_detail.emp_id%type;
6 empName emp_detail.emp_name%type;
7 empSalary emp_detail.emp_salary%type;
8 begin
9 open C1;
10 fetch C1 into empId, empName, empSalary;
11 while C1%found loop
12 dbms_output.put_line('Row Number '||
13 C1%rowcount || ' is '|| empId|| ' '||
14 empName|| ' '||empSalary);
15 fetch C1 into empId, empName, empSalary;
16 end loop;
17 close C1;
18 end;

```

Statement processed.  
Row Number 1 is 1 PALAK 100000  
Row Number 2 is 2 SALONI 100000  
Row Number 3 is 3 GUNEET 100000  
Row Number 4 is 4 JASHAN 100000

## EXCEPTION HANDLING

Oracle Live SQL - SQL Worksheet

livesql.oracle.com/apex/?p=590:1:17317947860994::NO:RP::

Feedback Help gunee1517@gmail.com

SQL Worksheet

Clear Find Actions Save Run

```
11 c_addr customers.address%TYPE;
12 -- user defined exception
13 ex_invalid_id EXCEPTION;
14 BEGIN
15 IF c_id <= 0 THEN
16 RAISE ex_invalid_id;
17 ELSE
18 SELECT name, address INTO c_name, c_addr
19 FROM customers
20 WHERE id = c_id;
21 DBMS_OUTPUT.PUT_LINE ('Name: ' || c_name);
22 DBMS_OUTPUT.PUT_LINE ('Address: ' || c_addr);
23 END IF;
24
25 EXCEPTION
26 WHEN ex_invalid_id THEN
27 dbms_output.put_line('ID must be greater than zero!');
28 WHEN no_data_found THEN
29 dbms_output.put_line('No such customer!');
30 WHEN others THEN
31 dbms_output.put_line('Error!');
32 END;
33 /
34
```

Statement processed.  
No such customer!

ORACLE | Integrated Cloud

Applications & Platform Services

© 2021 Oracle Corporation · Privacy · Terms of Use  
Oracle Learning Library · Ask Tom · Dev Gym · Database Documentation · Follow on Twitter  
Live SQL 21.1.1, running Oracle Database 19c Enterprise Edition - 19.8.0.0.0  
Built with ❤️ using Oracle APEX running on Oracle Cloud Infrastructure and Oracle Kubernetes Engine

↑

## PACKAGES

Oracle Live SQL - SQL Worksheet

livesql.oracle.com/apex/?p=590:1:17317947860994::NO:RP::

Feedback Help gunee1517@gmail.com

SQL Worksheet

Clear Find Actions Save Run

```
20
21 INSERT INTO CUSTOMERS172 (ID,NAME,AGE,ADDRESS,SALARY)
22 VALUES (5, 'DEVANSH', 27, 'Bhopal', 8500.00 );
23
24 INSERT INTO CUSTOMERS172 (ID,NAME,AGE,ADDRESS,SALARY)
25 VALUES (6, 'evans', 22, 'LA', 4500.00 );
26
27
28 CREATE OR REPLACE PACKAGE BODY cust_sal1 AS
29 PROCEDURE find_sal(c_id customers172.id%TYPE) IS
30 c_sal customers172.salary%TYPE;
31 BEGIN
32 SELECT salary INTO c_sal
33 FROM customers172
34 WHERE id = c_id;
35 dbms_output.put_line('Salary: ' || c_sal);
36 END find_sal;
37 END cust_sal1;
38
39 DECLARE
40 code customers172.id%TYPE := &c_id;
41 BEGIN
42 cust_sal1.find_sal(code);
43
```

ID	NAME	AGE	ADDRESS	SALARY
1	Guneet	20	Ahmedabad	2000
2	Jashan	25	Delhi	1500
3	TANIA	23	Kota	2000

ORACLE | Integrated Cloud

Applications & Platform Services

© 2021 Oracle Corporation · Privacy · Terms of Use  
Oracle Learning Library · Ask Tom · Dev Gym · Database Documentation · Follow on Twitter  
Live SQL 21.1.1, running Oracle Database 19c Enterprise Edition - 19.8.0.0.0  
Built with ❤️ using Oracle APEX running on Oracle Cloud Infrastructure and Oracle Kubernetes Engine

↑

## TRIGGERS

SQL Worksheet

Clear Find Actions Save Run

```
1 --CREATE TABLE EMPLOYEES172(ID NUMBER(20),NAME VARCHAR(20), SALARY NUMBER(20));
2 ---INSERT INTO EMPLOYEES172 VALUES(1,'JASHAN',90000);
3 --INSERT INTO EMPLOYEES172 VALUES(2,'GUNEET',70000);
4 --INSERT INTO EMPLOYEES172 VALUES(3,'MONICA',80000);
5 --INSERT INTO EMPLOYEES172 VALUES(4,'ABHISHEK',70000);
6 --INSERT INTO EMPLOYEES172 VALUES(5,'DEVANSH',40000);
7 CREATE OR REPLACE TRIGGER display_salary_changes
8 BEFORE DELETE OR INSERT OR UPDATE ON EMPLOYEES172
9 FOR EACH ROW
10 WHEN (NEW.ID > 0)
11 DECLARE
12     sal_diff number;
13 BEGIN
14     sal_diff := :NEW.salary - :OLD.salary;
15     dbms_output.put_line('Old salary: ' || :OLD.salary);
16     dbms_output.put_line('New salary: ' || :NEW.salary);
17     dbms_output.put_line('Salary difference: ' || sal_diff);
18 END;
19 /
```

Trigger created.