

Date: 22/09/25
Task: Procedure Functions and loops: Program using PL/SQL procedures, functions & loops.

AIM: To implement PL/SQL procedure function and loop on number theory and business scenarios.

1. Simple PL/SQL program (Static Input)

DECLARE

message VARCHAR2(20) = 'Booking closed'.

BEGIN

dbms_output.put_line(message);

END,

Output:

Booking closed

2. Conditional statement (Dynamic input).

DECLARE

hid NUMBER(3) := 100;

BEGIN

IF (hid=10) THEN

dbms_output.put_line('value of hid is 10');

ELSE IF (hid=20) THEN

dbms_output.put_line('value of hid is 20');

ELSE IF (hid=30) THEN

dbms_output.put_line('value of hid is 30');

ELSE

dbms_output.put_line('none of the value's matching');

GND IF;
dbms - output. put_line ('Exact value of hid is (hid));
END;

Output

None of the value is matching
exact value of hid is : 100

3. Nested loops Example:

DECLARE

hid NUMBER (1);

oid NUMBER (1);

BEGIN

<< outer - loop >>

for hid IN 1...3 loop

<< inner - loops >>

for oid IN 1...3 loop

dbms - output. put_line ('hid is : || hid || and
oid is : || out ||')

GND loop inner - loop;

GND loop outer - loop;

GND;

Output:

hid is : 1 and oid is : 1

hid is : 1 and oid is : 2

hid is : 1 and oid is : 3

hid is : 2 and oid is : 1

hid is : 2 and oid is : 2

hid is : 2 and oid is : 3

hid is : 3 and oid is : 1

hid is : 3 and oid is : 2

hid is : 3 and oid is : 3

4. Procedure Example

CREATE OR REPLACE PROCEDURE booking_status
(cid IN NUMBER)

IS

BEGIN:

If C_id > 200 THEN

dbms_output.put_line ('No booking available');

ELSE

dbms_output.put_line ('Booking open');

END IF;

END;

Execution:-

BEGIN

booking_status(150);

booking_status(250);

END;

Output:

Booking Open

No Booking available

Date: 22/09/25

PL/SQL Procedure for loops.

Example 1: Using WHILE loop with cursor.

Prime check using while loop.

CREATE OR REPLACE PROCEDURE print_prime_customers

CURSOR cost_cur IS

SELECT customer_id FROM customers;

V_id NUMBER;

V_is_prime BOOLEAN;

V_i NUMBER;

BEGIN

open cost_cur;

loop

FETCH cost_cur INTO V_id;

EXIT WHEN cost_cur %NOT FOUND;

if V_id < 2 THEN

V_is_prime := FALSE;

else

V_is_prime := TRUE;

V_i := 2;

WHILE V_i <= TRUNC(SORT(V_id)) LOOP

if MOD(V_id, V_i) = 0 THEN

V_is_prime := FALSE;

END IF;

V_i := V_i + 1;

END LOOP;

END IF;

if v-is-prime THEN

DBMS-OUTPUT.PUT-LINE ('Prime customer ID: ' || v-id);

END IF;

END LOOP;

CLOSE cust-cur;

END.

The procedure checks all customer IDs in the table and prints the prime ones using a WHILE loop.

Example 2: Using for loop for first n prime numbers.

CREATE OR REPLACE PROCEDURE print-first-n-primes
(n-number) IS .

v-num NUMBER := 2;

v-count NUMBER := 0;

v-is-prime BOOLEAN;

BEGIN

WHILE v-count < n loop

v-is-prime := TRUE;

FOR i IN 2..TRUNC(SORT(v-num)) loop

if MOD(v-num) = 0 THEN

v-is-prime := FALSE;

EXIT;

END IF;

END LOOP;

if v-is-prime THEN

DBMS-OUTPUT.PUT-LINE ('Prime: ' || v-num);

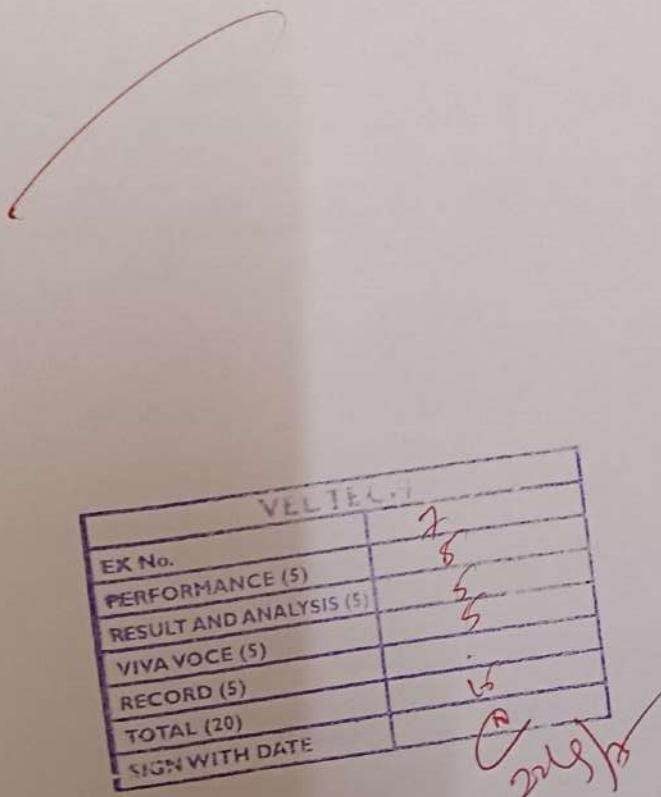
v-count := v-count + 1;

END IF;

```
V-num := V-num + 1;  
END LOOP;  
END;
```

This procedure prints the first 10 prime numbers using a for loop.

```
BEGIN  
    print-first-n-primes(10);  
END;
```



Result: Thus, the procedure function and loops program using PL/SQL procedures, functions & loops are executed successfully.