

## EXERCISE 9 AND 10

### PL/SQL Iterations, Subprograms and Cursors

#### LIVESQL LINKS:

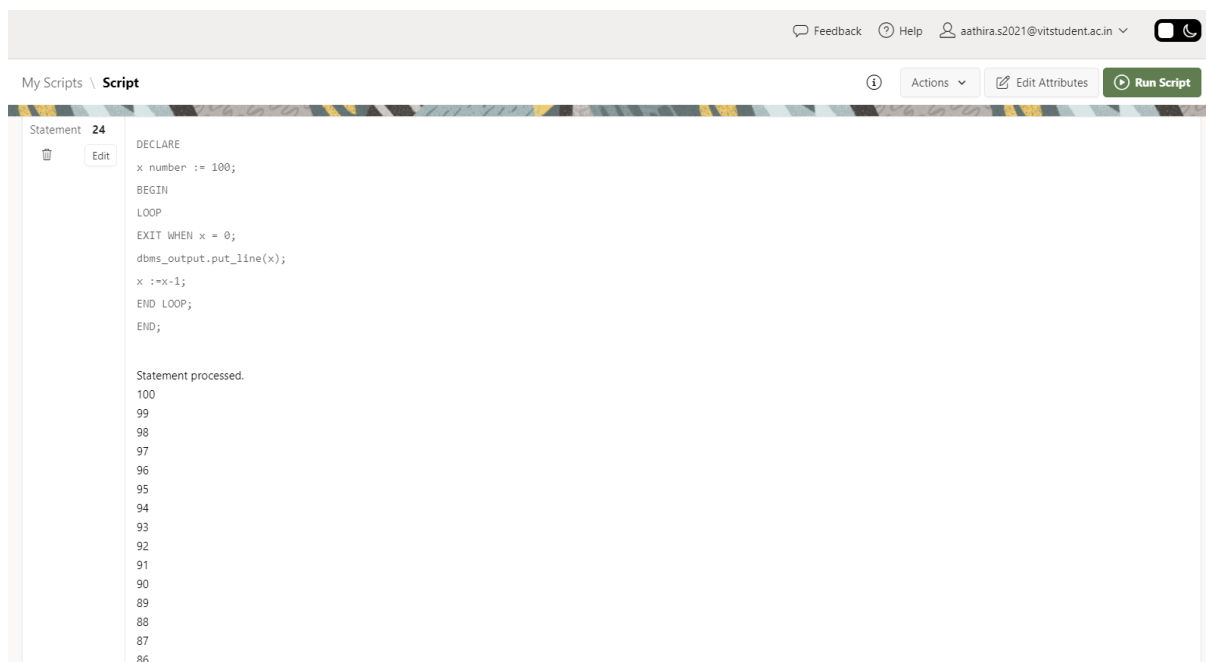
Exercise 9 & 10: <https://livesql.oracle.com/apex/livesql/s/bfv4b5ttuh2s2ajsp0zd4yuk>

#### Exercise 9

Aim: To understand the concepts of Iterations and Subprogram (Procedures and Functions)

#### Iterations (outcome: c)

1. Write a PL/SQL code to print the numbers in reverse order from 100 to 1.



The screenshot shows the Oracle LiveSQL web interface. At the top, there is a header with 'Feedback', 'Help', and a user profile 'aathira.s2021@vitstudent.ac.in'. Below the header, the page title is 'My Scripts \ Script'. On the right side of the script editor, there are buttons for 'Actions', 'Edit Attributes', and 'Run Script'. The script editor itself contains the following PL/SQL code:

```
DECLARE
x number := 100;
BEGIN
LOOP
EXIT WHEN x = 0;
dbms_output.put_line(x);
x := x-1;
END LOOP;
END;
```

Below the code, the output of the script is displayed, showing the numbers 100, 99, 98, 97, 96, 95, 94, 93, 92, 91, 90, 89, 88, 87, and 86. The first line of the output is 'Statement processed.'.

## 2. Create a PL/SQL block to find the sum of series 1+3+5+.....+n.

The screenshot shows a PL/SQL script editor interface. At the top, there's a header bar with 'Feedback', 'Help', a user profile 'aathira.s2021@vitstudent.ac.in', and a dark mode toggle. Below the header, the editor is titled 'My Scripts \ Script'. On the right side of the editor, there are buttons for 'Actions', 'Edit Attributes', and 'Run Script'. The main area contains a PL/SQL block for Statement 25. The code declares a variable 'n' as 11, 'sm' as 0, and 'curr' as 1. It then enters a 'BEGIN' block with a 'LOOP' that calculates the sum of odd numbers up to 'n'. The output shows 'Statement processed.' and the value '36'.

```
DECLARE
n number := 11;
sm number := 0;
curr number := 1;
BEGIN
LOOP
EXIT WHEN curr > n;
sm := sm + curr;
curr := curr + 2;
END LOOP;
dbms_output.put_line(sm);
END;
```

Statement processed.  
36

## Functions (outcome: m)

## 3. Write a function to give the number of rows in the table.

The screenshot shows a PL/SQL script editor interface. At the top, there's a header bar with 'Feedback', 'Help', a user profile 'aathira.s2021@vitstudent.ac.in', and a dark mode toggle. Below the header, the editor is titled 'My Scripts \ Script'. On the right side of the editor, there are buttons for 'Actions', 'Edit Attributes', and 'Run Script'. The main area contains two statements. Statement 26 is a PL/SQL function 'totalProjects' that returns the count of rows in a table named 'PROJECT'. Statement 27 is a SQL query 'select totalProjects from dual'. The output shows the function created and the result of the query, which is '5'.

```
CREATE OR REPLACE FUNCTION totalProjects
RETURN number IS
total number(2) := 0;
BEGIN
SELECT count(*) into total
FROM PROJECT;

RETURN total;
END;
```

Function created.

```
select totalProjects from dual
```

TOTALPROJECTS
5

Download CSV

#### 4. Write a PL/SQL to find the factorial of the given number using function.

The screenshot shows the SQL Developer interface with a script titled 'Script'. The script is as follows:

```
declare
num int:=5;
res int;

function fact(num IN int)
return int
AS
f int;

begin
f:=1;
for i in 1..num loop
f:=f*i;
end loop;
return f;
end fact;

begin

dbms_output.put_line('FINDING FACTORIAL OF:'||num);
res:=fact(num);
dbms_output.put_line('FACTORIAL IS:'||res);

end;
```

The output of the script is:

```
Statement processed.
FINDING FACTORIAL OF:5
FACTORIAL IS:120
```

### Procedure

#### 5. Write a procedure to accept an employee name and display his Department names.

The screenshot shows the SQL Developer interface with a script titled 'Script'. The script is as follows:

```
CREATE OR REPLACE PROCEDURE GETDEPT(
EMPNAME IN VARCHAR
)
IS
DEPTNAME VARCHAR(15);
BEGIN
SELECT DEPT INTO DEPTNAME
FROM EMPS
WHERE EMPNAME = EMPNAME
AND ROWNUM = 1;
DBMS_OUTPUT.PUT_LINE('DEPARTMENT NAME:' || DEPTNAME);
EXCEPTION
WHEN NO_DATA_FOUND THEN DBMS_OUTPUT.PUT_LINE('EMPLOYEE NOT FOUND.');
```

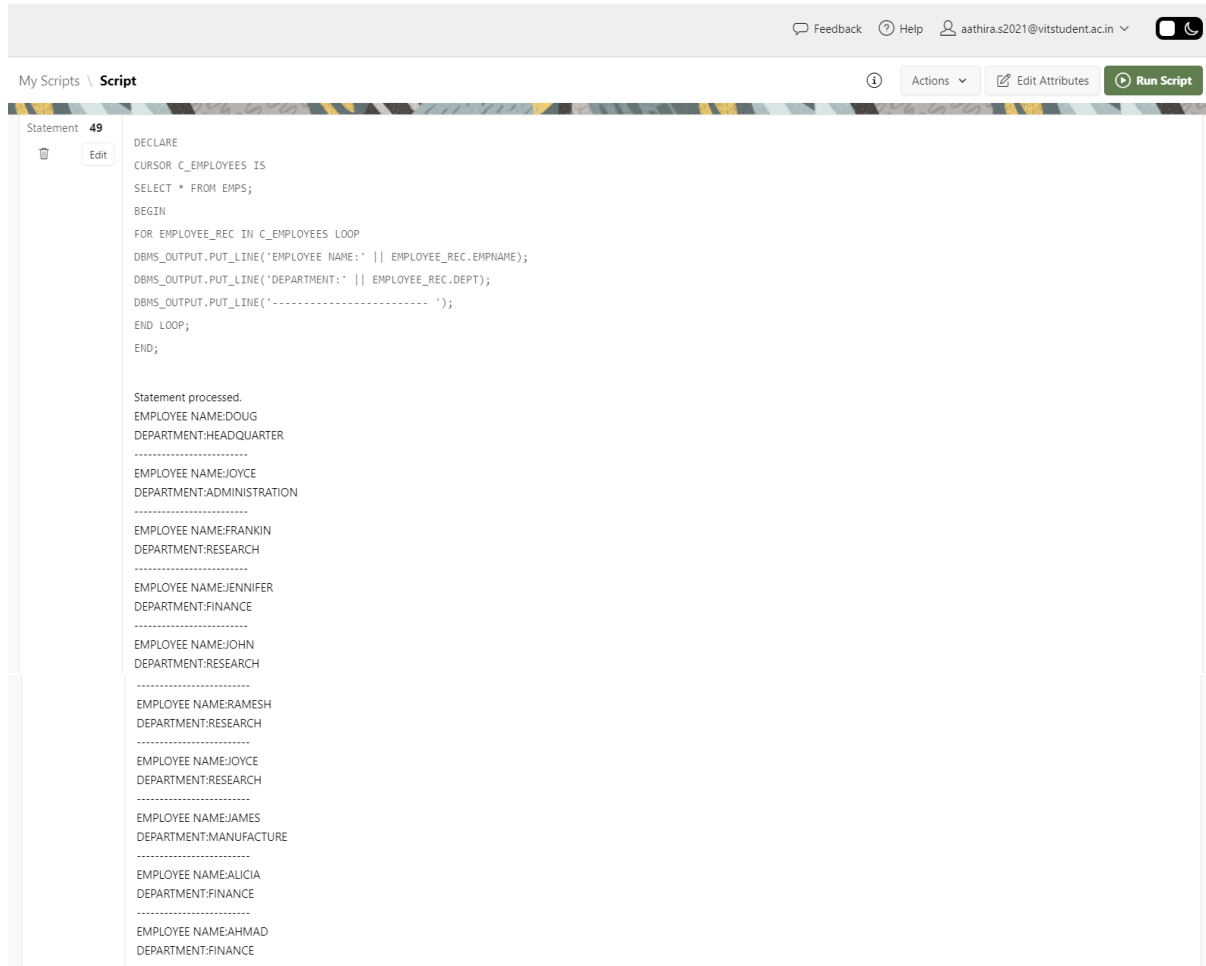
The output of the script is:

```
Statement processed.
DEPARTMENT NAME:HEADQUARTER
```

## Exercise 10

Aim: To understand implicit and explicit cursor in PL/SQL

1. Retrieve all the rows from the table using cursors.



```
DECLARE
CURSOR C_EMPLOYEES IS
SELECT * FROM EMP;
BEGIN
FOR EMPLOYEE_REC IN C_EMPLOYEES LOOP
DBMS_OUTPUT.PUT_LINE('EMPLOYEE NAME:' || EMPLOYEE_REC.EMPNAME);
DBMS_OUTPUT.PUT_LINE('DEPARTMENT:' || EMPLOYEE_REC.DEPT);
DBMS_OUTPUT.PUT_LINE('-----');
END LOOP;
END;
```

Statement processed.

EMPLOYEE NAME:DOUG  
DEPARTMENT:HEADQUARTER  
-----

EMPLOYEE NAME:JOYCE  
DEPARTMENT:ADMINISTRATION  
-----

EMPLOYEE NAME:FRANKIN  
DEPARTMENT:RESEARCH  
-----

EMPLOYEE NAME:JENNIFER  
DEPARTMENT:FINANCE  
-----

EMPLOYEE NAME:JOHN  
DEPARTMENT:RESEARCH  
-----

EMPLOYEE NAME:RAMESH  
DEPARTMENT:RESEARCH  
-----

EMPLOYEE NAME:JOYCE  
DEPARTMENT:RESEARCH  
-----

EMPLOYEE NAME:JAMES  
DEPARTMENT:MANUFACTURE  
-----

EMPLOYEE NAME:ALICIA  
DEPARTMENT:FINANCE  
-----

EMPLOYEE NAME:AHMAD  
DEPARTMENT:FINANCE  
-----

## 2. Write a cursor program to display few records using joins.

Feedback Help aathira.s2021@vitstudent.ac.in

My Scripts \ Script

Actions Edit Attributes Run Script

Statement 57

DECLARE  
CURSOR C\_EMPLOYEES IS  
SELECT E.FNAME, E.LNAME, D.NAME  
FROM EMPLOYEE E  
JOIN DEPARTMENT D  
ON E.DEPTNUM = D.NUM;  
  
V\_FNAME EMPLOYEE.FNAME%TYPE;  
V\_LNAME EMPLOYEE.LNAME%TYPE;  
V\_DEPTNAME DEPARTMENT.NAME%TYPE;  
  
BEGIN  
OPEN C\_EMPLOYEES;  
FOR I IN 1..3 LOOP  
FETCH C\_EMPLOYEES INTO V\_FNAME, V\_LNAME, V\_DEPTNAME;  
  
DBMS\_OUTPUT.PUT\_LINE('EMPLOYEE NAME:' || V\_FNAME || ' ' || V\_LNAME);  
DBMS\_OUTPUT.PUT\_LINE('DEPTNAME NAME:' || V\_DEPTNAME);  
DBMS\_OUTPUT.PUT\_LINE('-----');  
END LOOP;  
CLOSE C\_EMPLOYEES;  
END;

Statement processed.  
EMPLOYEE NAME:DOUG GILBERT  
DEPTNAME NAME:HEADQUARTER  
-----  
EMPLOYEE NAME:JOYCE PAN  
DEPTNAME NAME:ADMINISTRATION  
-----  
EMPLOYEE NAME:FRANKIN WONG  
DEPTNAME NAME:RESEARCH  
-----