SLOT: L3-L4

VIT
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

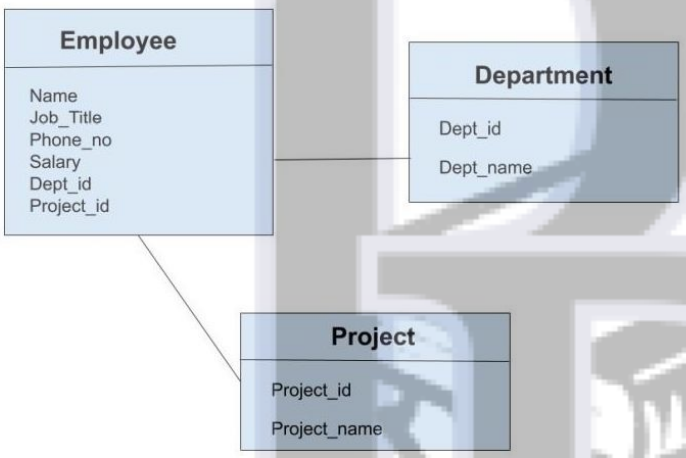
LAB: CYCLE SHEET –3 (PL/SQL) - FALL SEMESTER 2023-2024

Programme Name & Branch: B.Tech **Course Name: Database Systems LAB**

Course Code: BCSE302L

QUESTION

Consider the following Relations with Records and write PL/SQL Block for each of the Questions



Schema

```
SQL> -- Create Employee table
SQL> CREATE TABLE Employee (
  2    Name VARCHAR2(100),
  3    Job_Title VARCHAR2(100),
  4    Phone_no VARCHAR2(20),
  5    Salary NUMBER,
  6    Dept_id NUMBER,
  7    Project_Id NUMBER
  8  );

Table created.

SQL> -- Create Department table
SQL> CREATE TABLE Department (
  2    Dept_id NUMBER,
  3    Dept_name VARCHAR2(100),
  4    CONSTRAINT PK_Department PRIMARY KEY (Dept_id)
  5  );

Table created.

SQL> -- Create Project table
SQL> CREATE TABLE Project (
  2    Project_id NUMBER,
  3    Project_name VARCHAR2(100),
  4    CONSTRAINT PK_Project PRIMARY KEY (Project_id)
  5  );

Table created.

SQL>
```

```
SQL> INSERT INTO Department (Dept_id, Dept_name) VALUES (5, 'IT');

1 row created.

SQL> INSERT INTO Department (Dept_id, Dept_name) VALUES (6, 'Operations');

1 row created.

SQL> INSERT INTO Department (Dept_id, Dept_name) VALUES (7, 'Research');

1 row created.

SQL> INSERT INTO Department (Dept_id, Dept_name) VALUES (8, 'Customer Service');

1 row created.

SQL> INSERT INTO Department (Dept_id, Dept_name) VALUES (9, 'Legal');

1 row created.

SQL> INSERT INTO Department (Dept_id, Dept_name) VALUES (10, 'Production');

1 row created.

SQL> INSERT INTO Project (Project_id, Project_name) VALUES (1, 'Project A');

1 row created.

SQL> INSERT INTO Project (Project_id, Project_name) VALUES (2, 'Project B');

1 row created.

SQL> INSERT INTO Project (Project_id, Project_name) VALUES (3, 'Project C');
```

```
SQL> select * from Employee;

NAME                 JOB_TITLE            PHONE_NO     SALARY DEPT_ID PROJECT_ID
-------------------- -------------------- ------------ ------ ------- ----------
John Smith           Manager              1234567890     5000       1          1
Jane Doe             Engineer             9876543210     4000       1          1
Michael Johnson      Analyst              2345678901     3500       2          2
Emily Wilson         Developer            7890123456     4500       2          2
David Brown          Designer             5678901234     3000       3          3
Sarah Thompson       Administrator        9012345678     3800       3          3
Robert Miller        Engineer             3456789012     4200       1          1
Jennifer Davis       Analyst              8901234567     3600       2          2
Christopher Clark    Developer            6789012345     4100       2          2
Jessica Anderson     Designer             0123456789     3200       3          3

10 rows selected.

SQL> select * from Department;

DEPT_ID DEPT_NAME
------- --------------------
      1 Finance
      2 Marketing
      3 Human Resources
      4 Sales
      5 IT
      6 Operations
      7 Research
      8 Customer Service
      9 Legal
     10 Production

10 rows selected.
```

```
SQL> select * from Project;

PROJECT_ID PROJECT_NAME
---------- --------------------
         1 Project A
         2 Project B
         3 Project C
         4 Project D
         5 Project E
         6 Project F
         7 Project G
         8 Project H
         9 Project I
        10 Project J

10 rows selected.

SQL>
```

1. PL/SQL Block For any GivenProject-ID, Display its Project_Name

```
SET SERVEROUTPUT ON;
DECLARE
  v_project_name Project.Project_name%TYPE;
  v_project_id Project.Project_id%TYPE := 5;
BEGIN
  SELECT Project_name INTO v_project_name
  FROM Project
  WHERE Project_id = v_project_id;

  DBMS_OUTPUT.PUT_LINE('Project Name: ' || v_project_name);
EXCEPTION
  WHEN NO_DATA_FOUND THEN
    DBMS_OUTPUT.PUT_LINE('No project found for the given Project ID.');
END;
/
```

```
SQL> SET SERVEROUTPUT ON;
SQL>
SQL> DECLARE
  2    v_project_name Project.Project_name%TYPE;
  3    v_project_id Project.Project_id%TYPE := 5;
  4  BEGIN
  5    SELECT Project_name INTO v_project_name
  6    FROM Project
  7    WHERE Project_id = v_project_id;
  8
  9    DBMS_OUTPUT.PUT_LINE('Project Name: ' || v_project_name);
 10  EXCEPTION
 11    WHEN NO_DATA_FOUND THEN
 12      DBMS_OUTPUT.PUT_LINE('No project found for the given Project ID.');
 13  END;
 14  /
Project Name: Project E

PL/SQL procedure successfully completed.
```

2. PL/SQL Block to delete a record from Department where there is no employee.

```
SET SERVEROUTPUT ON;
DECLARE
  v_dept_id Department.Dept_id%TYPE := 7;
  v_employee_count NUMBER;
BEGIN
  -- Check if there are any employees associated with the department
```

```
 SELECT COUNT(*) INTO v_employee_count
 FROM Employee
 WHERE Dept_id = v_dept_id;

 -- Delete the department record if no employees are found
 IF v_employee_count = 0 THEN
   DELETE FROM Department
   WHERE Dept_id = v_dept_id;

   DBMS_OUTPUT.PUT_LINE('Department record deleted successfully.');
 ELSE
   DBMS_OUTPUT.PUT_LINE('Cannot delete department. Employees are associated with it.');
 END IF;
EXCEPTION
 WHEN NO_DATA_FOUND THEN
   DBMS_OUTPUT.PUT_LINE('No department found for the given Department ID.');
END;
/
```

```
SQL> SET SERVEROUTPUT ON;
SQL> DECLARE
  2    v_dept_id Department.Dept_id%TYPE := 7;
  3    v_employee_count NUMBER;
  4  BEGIN
  5    -- Check if there are any employees associated with the department
  6    SELECT COUNT(*) INTO v_employee_count
  7    FROM Employee
  8    WHERE Dept_id = v_dept_id;
  9
 10    -- Delete the department record if no employees are found
 11    IF v_employee_count = 0 THEN
 12      DELETE FROM Department
 13      WHERE Dept_id = v_dept_id;
 14
 15      DBMS_OUTPUT.PUT_LINE('Department record deleted successfully.');
 16    ELSE
 17      DBMS_OUTPUT.PUT_LINE('Cannot delete department. Employees are associated with it.');
 18    END IF;
 19  EXCEPTION
 20    WHEN NO_DATA_FOUND THEN
 21      DBMS_OUTPUT.PUT_LINE('No department found for the given Department ID.');
 22  END;
 23  /
Department record deleted successfully.

PL/SQL procedure successfully completed.
```

3. PL/SQL Block to Update All Null valued Phone_no with “123456789”.

```
SET SERVEROUTPUT ON;
DECLARE
BEGIN
 UPDATE Employee
 SET Phone_no = '123456789'
 WHERE Phone_no IS NULL;

 DBMS_OUTPUT.PUT_LINE('Null-valued Phone_no updated successfully.');
 DBMS_OUTPUT.PUT_LINE('Number of records updated: ' || SQL%ROWCOUNT);
EXCEPTION
 WHEN OTHERS THEN
   DBMS_OUTPUT.PUT_LINE('Error updating null-valued Phone_no: ' || SQLERRM);
END;
/
```

```
SQL> SET SERVEROUTPUT ON;
SQL> DECLARE
  2  BEGIN
  3    UPDATE Employee
  4    SET Phone_no = '123456789'
  5    WHERE Phone_no IS NULL;
  6
  7    DBMS_OUTPUT.PUT_LINE('Null-valued Phone_no updated successfully.');
  8    DBMS_OUTPUT.PUT_LINE('Number of records updated: ' || SQL%ROWCOUNT);
  9  EXCEPTION
 10    WHEN OTHERS THEN
 11      DBMS_OUTPUT.PUT_LINE('Error updating null-valued Phone_no: ' || SQLERRM);
 12  END;
 13  /
Null-valued Phone_no updated successfully.
Number of records updated: 0

PL/SQL procedure successfully completed.
```

5. PL/SQL Block to count the total number of Employees working for the Given Project

```
SET SERVEROUTPUT ON;
DECLARE
  v_project_id Project.Project_id%TYPE := 2;
  v_employee_count NUMBER;
BEGIN
  -- Count the number of employees working for the given project
  SELECT COUNT(*) INTO v_employee_count
  FROM Employee
  WHERE Project_Id = v_project_id;

  DBMS_OUTPUT.PUT_LINE('Total number of employees working for Project ID ' || v_project_id || ': ' || v_employee_count);
EXCEPTION
  WHEN NO_DATA_FOUND THEN
    DBMS_OUTPUT.PUT_LINE('No project found for the given Project ID.');
END;
/
```

```
SQL> SET SERVEROUTPUT ON;
SQL> DECLARE
  2    v_project_id Project.Project_id%TYPE := 2;
  3    v_employee_count NUMBER;
  4  BEGIN
  5    -- Count the number of employees working for the given project
  6    SELECT COUNT(*) INTO v_employee_count
  7    FROM Employee
  8    WHERE Project_Id = v_project_id;
  9
 10    DBMS_OUTPUT.PUT_LINE('Total number of employees working for Project ID ' || v_project_id || ': ' || v_employee_count);
 11  EXCEPTION
 12    WHEN NO_DATA_FOUND THEN
 13      DBMS_OUTPUT.PUT_LINE('No project found for the given Project ID.');
 14  END;
 15  /
Total number of employees working for Project ID 2: 4

PL/SQL procedure successfully completed.
```

6. Write a PL/SQL Procedure that will check for any updation of Project table such that the project title should not be started with "e-Commerce".

```
CREATE OR REPLACE PROCEDURE Check_Project_Update AS
BEGIN
  FOR proj IN (SELECT Project_id, Project_name FROM Project) LOOP
    IF proj.Project_name LIKE 'e-Commerce%' THEN
      RAISE_APPLICATION_ERROR(-20001, 'Project titles starting with "e-Commerce" are not allowed.');
    END IF;
  END LOOP;

  DBMS_OUTPUT.PUT_LINE('No project updates violate the condition.');
EXCEPTION
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('Error: ' || SQLERRM);
END;
/

BEGIN
  Check_Project_Update;
END;
/
```

```
SQL>
SQL> CREATE OR REPLACE PROCEDURE Check_Project_Update AS
  2  BEGIN
  3    FOR proj IN (SELECT Project_id, Project_name FROM Project) LOOP
  4      IF proj.Project_name LIKE 'e-Commerce%' THEN
  5        RAISE_APPLICATION_ERROR(-20001, 'Project titles starting with "e-Commerce" are not allowed.');
  6      END IF;
  7    END LOOP;
  8
  9    DBMS_OUTPUT.PUT_LINE('No project updates violate the condition.');
 10  EXCEPTION
 11    WHEN OTHERS THEN
 12      DBMS_OUTPUT.PUT_LINE('Error: ' || SQLERRM);
 13  END;
 14  /

Procedure created.

SQL> BEGIN
  2    Check_Project_Update;
  3  END;
  4  /
No project updates violate the condition.

PL/SQL procedure successfully completed.
```

7. Write a PL/SQL Function to count the Number of departments works for a Given Project-ID.

```
CREATE OR REPLACE FUNCTION Count_Departments_For_Project(
   v_project_id IN Project.Project_id%TYPE
) RETURN NUMBER IS
 v_department_count NUMBER;
BEGIN
 SELECT COUNT(DISTINCT Dept_id) INTO v_department_count
 FROM Employee
 WHERE Project_Id = v_project_id;

 RETURN v_department_count;
EXCEPTION
 WHEN NO_DATA_FOUND THEN
  RETURN 0;
END;
/

DECLARE
 v_project_id Project.Project_id%TYPE := 1;
 v_department_count NUMBER;
BEGIN
 v_department_count := Count_Departments_For_Project(v_project_id);
 DBMS_OUTPUT.PUT_LINE('Number of departments working for Project ID ' || v_project_id || ': ' || v_department_count);
END;
/
```

```
SQL> CREATE OR REPLACE FUNCTION Count_Departments_For_Project(
  2      v_project_id IN Project.Project_id%TYPE
  3  ) RETURN NUMBER IS
  4    v_department_count NUMBER;
  5  BEGIN
  6    SELECT COUNT(DISTINCT Dept_id) INTO v_department_count
  7    FROM Employee
  8    WHERE Project_Id = v_project_id;
  9
 10    RETURN v_department_count;
 11  EXCEPTION
 12    WHEN NO_DATA_FOUND THEN
 13      RETURN 0;
 14  END;
 15  /

Function created.

SQL> DECLARE
  2    v_project_id Project.Project_id%TYPE := 1;
  3    v_department_count NUMBER;
  4  BEGIN
  5    v_department_count := Count_Departments_For_Project(v_project_id);
  6    DBMS_OUTPUT.PUT_LINE('Number of departments working for Project ID ' || v_project_id || ': ' || v_department_count);
  7  END;
  8  /
Number of departments working for Project ID 1: 1

PL/SQL procedure successfully completed.
```

8. Use Cursor in PL/SQL Block to List out the Project Name and its Employee count for all projects.

```
SET SERVEROUTPUT ON;
DECLARE
 CURSOR c_projects IS
  SELECT p.Project_id, p.Project_name, COUNT(*) AS employee_count
  FROM Project p
  LEFT JOIN Employee e ON p.Project_id = e.Project_id
  GROUP BY p.Project_id, p.Project_name;
BEGIN
 FOR proj IN c_projects LOOP
  DBMS_OUTPUT.PUT_LINE('Project Name: ' || proj.Project_name || ', Employee Count: ' || proj.employee_count);
 END LOOP;
END;
/
```

```
SQL> SET SERVEROUTPUT ON;
SQL>
SQL> DECLARE
  2    CURSOR c_projects IS
  3      SELECT p.Project_id, p.Project_name, COUNT(*) AS employee_count
  4      FROM Project p
  5      LEFT JOIN Employee e ON p.Project_id = e.Project_id
  6      GROUP BY p.Project_id, p.Project_name;
  7  BEGIN
  8    FOR proj IN c_projects LOOP
  9      DBMS_OUTPUT.PUT_LINE('Project Name: ' || proj.Project_name || ', Employee Count: ' || proj.employee_count);
 10    END LOOP;
 11  END;
 12  /
Project Name: Project A, Employee Count: 3
Project Name: Project B, Employee Count: 4
Project Name: Project C, Employee Count: 3
Project Name: Project F, Employee Count: 1
Project Name: Project G, Employee Count: 1
Project Name: Project H, Employee Count: 1
Project Name: Project D, Employee Count: 1
Project Name: Project E, Employee Count: 1
Project Name: Project J, Employee Count: 1
Project Name: Project I, Employee Count: 1

PL/SQL procedure successfully completed.
```

9. Fire a Trigger during updation of Dept_ID in department table. The same dept-id should be reflected in employee table also.

```
CREATE OR REPLACE TRIGGER update_dept_id_trigger
BEFORE UPDATE ON Department
FOR EACH ROW
BEGIN
 IF :OLD.Dept_id <> :NEW.Dept_id THEN
  UPDATE Employee
  SET Dept_id = :NEW.Dept_id
  WHERE Dept_id = :OLD.Dept_id;
 END IF;
END;
/

UPDATE Department
SET Dept_ID = '13'
WHERE Dept_ID = '3';

SELECT * FROM Employee;
```

```
SQL> CREATE OR REPLACE TRIGGER update_dept_id_trigger
  2  BEFORE UPDATE ON Department
  3  FOR EACH ROW
  4  BEGIN
  5    IF :OLD.Dept_id <> :NEW.Dept_id THEN
  6      UPDATE Employee
  7      SET Dept_id = :NEW.Dept_id
  8      WHERE Dept_id = :OLD.Dept_id;
  9    END IF;
 10  END;
 11  /

Trigger created.

SQL> UPDATE Department
  2  SET Dept_ID = '13'
  3  WHERE Dept_ID = '3';

1 row updated.
```

```
SQL> SELECT * FROM Employee;

NAME                 JOB_TITLE            PHONE_NO     SALARY DEPT_ID PROJECT_ID
-------------------- -------------------- ---------- ------ ------- ----------
John Smith           Manager              1234567890     5000       1          1
Jane Doe             Engineer             9876543210     4000       1          1
Michael Johnson      Analyst              2345678901     3500       2          2
Emily Wilson         Developer            7890123456     4500       2          2
David Brown          Designer             5678901234     3000      13          3
Sarah Thompson       Administrator        9012345678     3800      13          3
Robert Miller        Engineer             3456789012     4200       1          1
Jennifer Davis       Analyst              8901234567     3600       2          2
Christopher Clark    Developer            6789012345     4100       2          2
Jessica Anderson     Designer             0123456789     3200      13          3

10 rows selected.
```

10. Fire a Trigger during Insertion of any record into Project Table for ensuring the existence of its employee-id in employee table. Handle Exception if employee id is not available in the employee table.

```
CREATE OR REPLACE TRIGGER check_employee_id_trigger
BEFORE INSERT ON Project
FOR EACH ROW
DECLARE
 v_employee_count NUMBER;
BEGIN
 SELECT COUNT(*) INTO v_employee_count
 FROM Employee
 WHERE Employee_ID = :NEW.Employee_ID;

 IF v_employee_count = 0 THEN
  RAISE_APPLICATION_ERROR(-20001, 'Employee ID does not exist in the Employee table.');
 END IF;
EXCEPTION
 WHEN OTHERS THEN
  RAISE_APPLICATION_ERROR(-20002, 'Error: ' || SQLERRM);
END;
/

INSERT INTO Project (Project_ID, Project_Name, Employee_ID)
VALUES (1, 'Sample Project', 999);
```

```
SQL> CREATE OR REPLACE TRIGGER check_employee_id_trigger
  2  BEFORE INSERT ON Project
  3  FOR EACH ROW
  4  DECLARE
  5    v_employee_count NUMBER;
  6  BEGIN
  7    SELECT COUNT(*) INTO v_employee_count
  8    FROM Employee
  9    WHERE Employee_ID = :NEW.Employee_ID;
 10
 11    IF v_employee_count = 0 THEN
 12      RAISE_APPLICATION_ERROR(-20001, 'Employee ID does not exist in the Employee table.');
 13    END IF;
 14  EXCEPTION
 15    WHEN OTHERS THEN
 16      RAISE_APPLICATION_ERROR(-20002, 'Error: ' || SQLERRM);
 17  END;
 18  /

Warning: Trigger created with compilation errors.

SQL> INSERT INTO Project (Project_ID, Project_Name, Employee_ID)
  2  VALUES (1, 'Sample Project', 999);
INSERT INTO Project (Project_ID, Project_Name, Employee_ID)
                                               *
ERROR at line 1:
ORA-00904: "EMPLOYEE_ID": invalid identifier
```

11. Write a PL/SQL block to list the for each department name participating in the projects along with its total money spent for salary.

```
SET SERVEROUTPUT ON;
DECLARE
 CURSOR c_project_departments IS
  SELECT DISTINCT d.Dept_name, SUM(e.Salary) AS total_salary
  FROM Department d
  JOIN Employee e ON d.Dept_id = e.Dept_id
  GROUP BY d.Dept_name;

 v_dept_name Department.Dept_name%TYPE;
 v_total_salary Employee.Salary%TYPE;
BEGIN
 FOR project_department IN c_project_departments LOOP
  v_dept_name := project_department.Dept_name;
  v_total_salary := project_department.total_salary;
  DBMS_OUTPUT.PUT_LINE('Department: ' || v_dept_name || ', Total Salary: ' || v_total_salary);
 END LOOP;
END;
/
```

```
SQL> SET SERVEROUTPUT ON;
SQL>
SQL> DECLARE
  2    CURSOR c_project_departments IS
  3      SELECT DISTINCT d.Dept_name, SUM(e.Salary) AS total_salary
  4      FROM Department d
  5      JOIN Employee e ON d.Dept_id = e.Dept_id
  6      GROUP BY d.Dept_name;
  7
  8    v_dept_name Department.Dept_name%TYPE;
  9    v_total_salary Employee.Salary%TYPE;
 10  BEGIN
 11    FOR project_department IN c_project_departments LOOP
 12      v_dept_name := project_department.Dept_name;
 13      v_total_salary := project_department.total_salary;
 14      DBMS_OUTPUT.PUT_LINE('Department: ' || v_dept_name || ', Total Salary: ' || v_total_salary);
 15    END LOOP;
 16  END;
 17  /
Department: Finance, Total Salary: 13200
Department: Marketing, Total Salary: 15700
Department: Human Resources, Total Salary: 10000

PL/SQL procedure successfully completed.
```