DBMS LAB

Name:-Naresh Suthar
Roll No.:-69
SRN No.:-202202174
Div.:-A(A3)

I. Consider the given database schema:

Student (studentid , studentname,instructorid,studentcity)

Instructor (instructorid,Instructorname,instructor city,specialization)

Use all types of Joins and set opoeration

1. Add primary and foreign keys

```
mysql> CREATE TABLE Student (
    ->     studentid INT PRIMARY KEY,
    ->     studentname VARCHAR(50),
    ->     instructorid INT,
    ->     studentcity VARCHAR(50),
    ->     FOREIGN KEY (instructorid) REFERENCES Instructor(instructorid)
    -> );
```

```
mysql> CREATE TABLE Student (
    ->     studentid INT PRIMARY KEY,
    ->     studentname VARCHAR(50),
    ->     instructorid INT,
    ->     studentcity VARCHAR(50),
    ->     FOREIGN KEY (instructorid) REFERENCES Instructor(instructorid)
    -> );
```

Also inserting values in a given table:-

```
mysql> INSERT INTO Instructor (instructorid, Instructorname, instructorcity, specialization)
    -> VALUES
    -> (1, 'John', 'Pune', 'Computer'),
    -> (2, 'Jane', 'Mumbai', 'Physics'),
    -> (3, 'Alice', 'Pune', 'Math'),
    -> (4, 'Bob', 'Delhi', 'Chemistry');
Query OK, 4 rows affected (0.01 sec)
Records: 4  Duplicates: 0  Warnings: 0
```

```
mysql> INSERT INTO Student (studentid, studentname, instructorid, studentcity)
    -> VALUES
    -> (1, 'Alice', 1, 'City1'),
    -> (2, 'Bob', 2, 'City2'),
    -> (3, 'Charlie', 3, 'City3'),
    -> (4, 'David', NULL, 'City4');
Query OK, 4 rows affected (0.01 sec)
Records: 4  Duplicates: 0  Warnings: 0
```

2. Find the instructor of each student.

```
mysql> SELECT s.studentid, s.studentname, i.instructorid, i.Instructorname
    -> FROM Student s
    -> JOIN Instructor i ON s.instructorid = i.instructorid;
+-----------+-------------+--------------+----------------+
| studentid | studentname | instructorid | Instructorname |
+-----------+-------------+--------------+----------------+
|         1 | Alice       |            1 | John           |
|         2 | Bob         |            2 | Jane           |
|         3 | Charlie     |            3 | Alice          |
+-----------+-------------+--------------+----------------+
3 rows in set (0.01 sec)
```

3. Find the student who is not having any instructor.

```
mysql> SELECT *
    -> FROM Student
    -> WHERE instructorid IS NULL;
+-----------+-------------+--------------+-------------+
| studentid | studentname | instructorid | studentcity |
+-----------+-------------+--------------+-------------+
|         4 | David       |         NULL | City4       |
+-----------+-------------+--------------+-------------+
1 row in set (0.00 sec)
```

4. Find the student who is not having any instructor as well as the instructor who is not having a student.

```
mysql> SELECT s.studentid, s.studentname, s.instructorid, i.instructorid, i.Instructorname
    -> FROM Student s
    -> RIGHT JOIN Instructor i ON s.instructorid = i.instructorid
    -> WHERE s.instructorid IS NULL OR i.instructorid IS NULL;
+-----------+-------------+--------------+--------------+----------------+
| studentid | studentname | instructorid | instructorid | Instructorname |
+-----------+-------------+--------------+--------------+----------------+
|      NULL | NULL        |         NULL |            4 | Bob            |
+-----------+-------------+--------------+--------------+----------------+
1 row in set (0.01 sec)
```

5. Find the students whose instructor's specialization is computer.

```
mysql> SELECT s.studentid, s.studentname, i.instructorid, i.Instructorname, i.specialization
    -> FROM Student s
    -> JOIN Instructor i ON s.instructorid = i.instructorid
    -> WHERE i.specialization = 'Computer';
+-----------+-------------+--------------+----------------+----------------+
| studentid | studentname | instructorid | Instructorname | specialization |
+-----------+-------------+--------------+----------------+----------------+
|         1 | Alice       |            1 | John           | Computer       |
+-----------+-------------+--------------+----------------+----------------+
1 row in set (0.00 sec)
```

6. Create a view containing the total number of students whose instructor belongs to "Pune".

```
mysql> CREATE VIEW PuneStudents AS
    -> SELECT COUNT(s.studentid) AS num_students
    -> FROM Student s
    -> JOIN Instructor i ON s.instructorid = i.instructorid
    -> WHERE i.instructorcity = 'Pune';
Query OK, 0 rows affected (0.02 sec)
```

II.Consider the following database. Execute each query given using join and subqueries.

```
CREATE TABLE departments (
        department_id INT (11) AUTO_INCREMENT PRIMARY KEY,
        department_name VARCHAR (30) NOT NULL,
        location_id INT (11) DEFAULT NULL,
        );


CREATE TABLE employees (
        employee_id INT (11) AUTO_INCREMENT PRIMARY KEY,
        first_name VARCHAR (20) DEFAULT NULL,
        last_name VARCHAR (25) NOT NULL,
        email VARCHAR (100) NOT NULL,
        phone_number VARCHAR (20) DEFAULT NULL,
        hire_date DATE NOT NULL,
        job_id INT (11) NOT NULL,
        salary DECIMAL (8, 2) NOT NULL,
        manager_id INT (11) DEFAULT NULL,
        department_id INT (11) DEFAULT NULL,
        FOREIGN KEY (department_id) REFERENCES departments (department_id)
ON DELETE CASCADE ON UPDATE CASCADE,
        FOREIGN KEY (manager_id) REFERENCES employees (employee_id)
);
```

INSERTING VALUES IN THE BOTH THE TABLES:-

```
mysql> -- Insert some sample data into departments table
mysql> INSERT INTO departments (department_name, location_id) VALUES ('Engineering', 1700);
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO departments (department_name, location_id) VALUES ('Marketing', 1800);
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO departments (department_name, location_id) VALUES ('Sales', 1700);
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO departments (department_name, location_id) VALUES ('HR', 1800);
Query OK, 1 row affected (0.01 sec)
```

```
mysql> -- Insert some sample data into employees table
mysql> INSERT INTO employees (first_name, last_name, email, phone_number, hire_date, job_id, salary, manager_id, department_id) VALUES ('Alice', 'Smith', 'alice.smith@example.com', '123-456-7890', '2022-01-01',
1, 60000.00, NULL, 1);
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO employees (first_name, last_name, email, phone_number, hire_date, job_id, salary, manager_id, department_id) VALUES ('Bob', 'Johnson', 'bob.johnson@example.com', '987-654-3210', '2022-02-01',
2, 70000.00, 1, 2);
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO employees (first_name, last_name, email, phone_number, hire_date, job_id, salary, manager_id, department_id) VALUES ('Charlie', 'Brown', 'charlie.brown@example.com', '555-555-5555', '2022-03-0
1', 1, 80000.00, 1, 1);
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO employees (first_name, last_name, email, phone_number, hire_date, job_id, salary, manager_id, department_id) VALUES ('David', 'Lee', 'david.lee@example.com', '111-222-3333', '2022-04-01', 2, 9
0000.00, 3, 2);
Query OK, 1 row affected (0.01 sec)
```

1. Find all employees who locate in the location with the id 1700.

```
mysql> SELECT * FROM employees
    -> WHERE department_id IN (SELECT department_id FROM departments WHERE location_id = 1700);
+-------------+------------+-----------+-------------------------+--------------+------------+--------+----------+------------+---------------+
| employee_id | first_name | last_name | email                   | phone_number | hire_date  | job_id | salary   | manager_id | department_id |
+-------------+------------+-----------+-------------------------+--------------+------------+--------+----------+------------+---------------+
|           1 | Alice      | Smith     | alice.smith@example.com | 123-456-7890 | 2022-01-01 |      1 | 60000.00 |       NULL |             1 |
|           3 | Charlie    | Brown     | charlie.brown@example.com | 555-555-5555 | 2022-03-01 |      1 | 80000.00 |          1 |             1 |
+-------------+------------+-----------+-------------------------+--------------+------------+--------+----------+------------+---------------+
2 rows in set (0.00 sec)
```

2. Find all employees who do not locate at the location 1700.

```
mysql> SELECT * FROM employees
    -> WHERE department_id NOT IN (SELECT department_id FROM departments WHERE location_id = 1700);
+-------------+------------+-----------+-------------------------+--------------+------------+--------+----------+------------+---------------+
| employee_id | first_name | last_name | email                   | phone_number | hire_date  | job_id | salary   | manager_id | department_id |
+-------------+------------+-----------+-------------------------+--------------+------------+--------+----------+------------+---------------+
|           2 | Bob        | Johnson   | bob.johnson@example.com | 987-654-3210 | 2022-02-01 |      2 | 70000.00 |          1 |             2 |
|           4 | David      | Lee       | david.lee@example.com   | 111-222-3333 | 2022-04-01 |      2 | 90000.00 |          3 |             2 |
+-------------+------------+-----------+-------------------------+--------------+------------+--------+----------+------------+---------------+
2 rows in set (0.00 sec)
```

3. Finds the employees who have the highest salary.

```
mysql> SELECT * FROM employees
    -> WHERE salary = (SELECT MAX(salary) FROM employees);
+-------------+------------+-----------+-----------------------+--------------+------------+--------+----------+------------+---------------+
| employee_id | first_name | last_name | email                 | phone_number | hire_date  | job_id | salary   | manager_id | department_id |
+-------------+------------+-----------+-----------------------+--------------+------------+--------+----------+------------+---------------+
|           4 | David      | Lee       | david.lee@example.com | 111-222-3333 | 2022-04-01 |      2 | 90000.00 |          3 |             2 |
+-------------+------------+-----------+-----------------------+--------------+------------+--------+----------+------------+---------------+
1 row in set (0.01 sec)
```

4. Finds all employees whose salaries are greater than the average salary of all employees.

```
mysql> SELECT * FROM employees
    -> WHERE salary > (SELECT AVG(salary) FROM employees);
+-------------+------------+-----------+-------------------------+--------------+------------+--------+----------+------------+---------------+
| employee_id | first_name | last_name | email                   | phone_number | hire_date  | job_id | salary   | manager_id | department_id |
+-------------+------------+-----------+-------------------------+--------------+------------+--------+----------+------------+---------------+
|           3 | Charlie    | Brown     | charlie.brown@example.com | 555-555-5555 | 2022-03-01 |      1 | 80000.00 |          1 |             1 |
|           4 | David      | Lee       | david.lee@example.com     | 111-222-3333 | 2022-04-01 |      2 | 90000.00 |          3 |             2 |
+-------------+------------+-----------+-------------------------+--------------+------------+--------+----------+------------+---------------+
2 rows in set (0.00 sec)
```

5. Finds all departments which have at least one employee with the salary is greater than 10,000.

```
mysql> SELECT * FROM departments
    -> WHERE department_id IN (SELECT department_id FROM employees WHERE salary > 10000);
+---------------+-----------------+-------------+
| department_id | department_name | location_id |
+---------------+-----------------+-------------+
|             1 | Engineering     |        1700 |
|             2 | Marketing       |        1800 |
+---------------+-----------------+-------------+
2 rows in set (0.00 sec)
```

6. Finds all departments that do not have any employee with the salary greater than 10,000.

```
mysql> SELECT * FROM departments
    -> WHERE department_id NOT IN (SELECT department_id FROM employees WHERE salary > 10000);
+---------------+-----------------+-------------+
| department_id | department_name | location_id |
+---------------+-----------------+-------------+
|             3 | Sales           |        1700 |
|             4 | HR              |        1800 |
+---------------+-----------------+-------------+
2 rows in set (0.00 sec)
```

7. Finds all employees whose salaries are greater than the lowest salary of every department.

```
mysql> SELECT e.*
    -> FROM employees e
    -> JOIN (
    ->     SELECT department_id, MIN(salary) AS min_salary
    ->     FROM employees
    ->     GROUP BY department_id
    -> ) m ON e.department_id = m.department_id
    -> WHERE e.salary > m.min_salary;
+-------------+------------+-----------+-------------------------+--------------+------------+--------+----------+------------+---------------+
| employee_id | first_name | last_name | email                   | phone_number | hire_date  | job_id | salary   | manager_id | department_id |
+-------------+------------+-----------+-------------------------+--------------+------------+--------+----------+------------+---------------+
|           3 | Charlie    | Brown     | charlie.brown@example.com | 555-555-5555 | 2022-03-01 |      1 | 80000.00 |          1 |             1 |
|           4 | David      | Lee       | david.lee@example.com     | 111-222-3333 | 2022-04-01 |      2 | 90000.00 |          3 |             2 |
+-------------+------------+-----------+-------------------------+--------------+------------+--------+----------+------------+---------------+
2 rows in set (0.00 sec)
```

8. Finds all employees whose salaries are greater than or equal to the highest salary of every department.

```
mysql> SELECT e.*
    -> FROM employees e
    -> JOIN (
    ->     SELECT department_id, MAX(salary) AS max_salary
    ->     FROM employees
    ->     GROUP BY department_id
    -> ) m ON e.department_id = m.department_id
    -> WHERE e.salary >= m.max_salary;
+-------------+------------+-----------+-------------------------+--------------+------------+--------+----------+------------+---------------+
| employee_id | first_name | last_name | email                   | phone_number | hire_date  | job_id | salary   | manager_id | department_id |
+-------------+------------+-----------+-------------------------+--------------+------------+--------+----------+------------+---------------+
|           3 | Charlie    | Brown     | charlie.brown@example.com | 555-555-5555 | 2022-03-01 |      1 | 80000.00 |          1 |             1 |
|           4 | David      | Lee       | david.lee@example.com   | 111-222-3333 | 2022-04-01 |      2 | 90000.00 |          3 |             2 |
+-------------+------------+-----------+-------------------------+--------------+------------+--------+----------+------------+---------------+
2 rows in set (0.00 sec)
```

9. Finds the salaries of all employees, their average salary, and the difference between the salary of each employee and the average salary.

```
mysql> SELECT e.employee_id, e.salary, (e.salary - avg_salary.avg_salary) AS salary_diff
    -> FROM employees e
    -> CROSS JOIN (
    ->     SELECT AVG(salary) AS avg_salary
    ->     FROM employees
    -> ) avg_salary;
+-------------+----------+----------------+
| employee_id | salary   | salary_diff    |
+-------------+----------+----------------+
|           1 | 60000.00 | -15000.000000 |
|           2 | 70000.00 |  -5000.000000 |
|           3 | 80000.00 |   5000.000000 |
|           4 | 90000.00 |  15000.000000 |
+-------------+----------+----------------+
4 rows in set (0.00 sec)
```