Name: Saran S

Date:23/11/24

```
SET SQL_SAFE_UPDATES = 0;

CREATE TABLE IF NOT EXISTS StaffDetails (
   staffId INT PRIMARY KEY,
   staffName VARCHAR(50),
   salary INT,
   city VARCHAR(50),
   joinDate DATE,
   supervisor VARCHAR(50),
   teamId INT
);

CREATE TABLE IF NOT EXISTS Teams (
   teamId INT PRIMARY KEY,
   teamName VARCHAR(50),
   totalMembers INT,
   city VARCHAR(50)
);

CREATE TABLE IF NOT EXISTS Assignments (
   assignmentId VARCHAR(20) PRIMARY KEY,
   assignmentTitle VARCHAR(50),
```

```
    teamId INT,
    FOREIGN KEY (teamId) REFERENCES Teams(teamId)
);

ALTER TABLE Assignments MODIFY assignmentId VARCHAR(30);
ALTER TABLE Assignments MODIFY assignmentTitle VARCHAR(100);

INSERT INTO Teams (teamId, teamName, totalMembers, city)
VALUES
(1001, 'HR', 0, 'New York'),
(1002, 'Finance', 0, 'Chicago'),
(1003, 'Engineering', 0, 'San Francisco'),
(1004, 'Logistics', 0, 'Dallas');

INSERT INTO Assignments (assignmentId, assignmentTitle, teamId)
VALUES
('A101', 'Web Development', 1001),
('A102', 'Data Analysis', 1002),
('A103', 'Mobile App Development', 1003),
('A104', 'System Design', 1004);

INSERT INTO StaffDetails (staffId, staffName, salary, city, joinDate, supervisor, teamId)
VALUES
(101, 'John', 80000, 'New York', '2020-03-01', 'Sophia', 1001),
(102, 'Alice', 70000, 'Chicago', '2019-07-15', 'Sophia', 1002),
(103, 'Bob', 90000, 'San Francisco', '2021-08-12', 'David', 1003),
```

(104, 'Eve', 75000, 'Dallas', '2018-11-20', 'David', 1004),

(105, 'Sophia', 95000, 'New York', '2015-02-10', NULL, NULL);

1. Write a query to display all rows and columns from the employees table.

SELECT * FROM StaffDetails;

2. Retrieve only the name and salary of all employees from the employees table.

SELECT staffName, salary FROM StaffDetails;

3. Write a query to find all employees whose salary is greater than 60,000.

SELECT * FROM StaffDetails WHERE salary > 60000;

4. List all employees who joined the company in the year 2021.

SELECT * FROM StaffDetails WHERE YEAR(joinDate) = 2021;

5. Retrieve the details of employees whose names start with the letter 'J'.

SELECT * FROM StaffDetails WHERE staffName LIKE 'J%';

1. Write a query to calculate the average salary of all employees.

SELECT AVG(salary) AS Avg_Salary FROM StaffDetails;

2. Find the total number of employees in the company.

SELECT COUNT(staffId) AS Total_Staff FROM StaffDetails;

3. Write a query to find the highest salary in the employees table.

SELECT MAX(salary) AS Max_Salary FROM StaffDetails;

4. Calculate the total salary paid by the company for all employees.

```
SELECT SUM(salary) AS Total_Salary FROM StaffDetails;
```

5. Find the count of employees in each department.

```
SELECT t.teamName, COUNT(s.staffId) AS Staff_Count
FROM StaffDetails s
JOIN Teams t ON s.teamId = t.teamId
GROUP BY t.teamName;
```

1. Write a query to retrieve employee names along with their department names (using employees and departments tables).

```
SELECT s.staffName, t.teamName
FROM StaffDetails s
JOIN Teams t ON s.teamId = t.teamId;
```

2. List all employees who have a manager (self-join on employees table).

```
SELECT s1.staffName AS Employee, s2.staffName AS Manager
FROM StaffDetails s1
JOIN StaffDetails s2 ON s1.supervisor = s2.staffName;
```

3. Find the names of employees who are working on multiple projects (using employees and projects tables).

```
SELECT s.staffName, COUNT(a.assignmentId) AS Assignment_Count
FROM StaffDetails s
JOIN Assignments a ON s.teamId = a.teamId
```

```
GROUP BY s.staffId, s.staffName
HAVING COUNT(a.assignmentId) > 1;
```

4. Write a query to display all projects and the employees assigned to them.

```
SELECT a.assignmentTitle, s.staffName
FROM StaffDetails s
JOIN Assignments a ON s.teamId = a.teamId;
```

5. Retrieve the names of employees who do not belong to any department.

```
SELECT staffName FROM StaffDetails WHERE teamId IS NULL;
```

1. Write a query to find the employees with the second-highest salary.

```
SELECT staffName, salary
FROM StaffDetails
WHERE salary = (
  SELECT MAX(salary)
  FROM StaffDetails
  WHERE salary < (
    SELECT MAX(salary)
    FROM StaffDetails
  )
);
```

2. Retrieve the names of employees whose salary is above the department average salary.

```
SELECT s.staffName, s.salary, t.teamName
FROM StaffDetails s
```

```
JOIN Teams t ON s.teamId = t.teamId
WHERE s.salary > (
  SELECT AVG(salary)
  FROM StaffDetails
  WHERE teamId = s.teamId
);
```

3. Find employees who earn more than the average salary of the entire company.

```
SELECT * FROM StaffDetails
WHERE salary > (SELECT AVG(salary) FROM StaffDetails);
```

4. Write a query to find the department with the highest number of employees.

```
SELECT teamName, totalMembers
FROM Teams
ORDER BY totalMembers DESC
LIMIT 1;
```

5. List all employees who work in a department located in 'New York'.

```
SELECT s.staffName, t.teamName
FROM StaffDetails s
JOIN Teams t ON s.teamId = t.teamId
WHERE city = 'New York';
```

1. Write a query to find employees who work in either the 'HR' or 'Finance' department.

```
SELECT s.staffName
FROM StaffDetails s
```

```
JOIN Teams t ON s.teamId = t.teamId
WHERE t.teamName IN ('HR', 'Finance');
```

2. Retrieve the names of employees who are working on both Project A and Project B.

```
SELECT s.staffName
FROM StaffDetails s
JOIN Assignments a1 ON s.teamId = a1.teamId AND a1.assignmentTitle = 'Web Development'
JOIN Assignments a2 ON s.teamId = a2.teamId AND a2.assignmentTitle = 'Data Analysis';
```

3. Find employees who are not assigned to any project.

```
SELECT s.staffName
FROM StaffDetails s
WHERE NOT EXISTS (
  SELECT 1
  FROM Assignments a
  WHERE s.teamId = a.teamId
);
```

4. Write a query to get all unique job titles across all departments.

```
SELECT DISTINCT assignmentTitle FROM Assignments;
```

5. Combine two tables (employees and former_employees) and remove duplicates.

```
SELECT * FROM StaffDetails
UNION
SELECT * FROM Former_StaffDetails;
```

1. Write a query to add a new employee to the employees table.

INSERT INTO StaffDetails VALUES (201, 'Ethan', 90000, 'Los Angeles', '2022-05-10', 'Sophia', 1001);

2. Update the salary of all employees in the 'HR' department by 10%.

UPDATE StaffDetails

SET salary = salary * 1.10

WHERE teamId = (SELECT teamId FROM Teams WHERE teamName = 'HR');

3. Delete all employees who have not worked for more than 5 years.

DELETE FROM StaffDetails

WHERE DATEDIFF(CURDATE(), joinDate) > 5 * 365;

4. Create a new table departments_backup with the same structure as the departments table.

CREATE TABLE Team_Backup AS

SELECT * FROM Teams;

5. Drop the temporary_data table from the database.

DROP TABLE IF EXISTS Team_Backup;