SQL

```sql
-- Create Departments Table
CREATE TABLE Departments (
  DeptID INT PRIMARY KEY,
  DeptName VARCHAR(50)
);

-- Insert Sample Data into Departments
INSERT INTO Departments VALUES
(1, 'Engineering'),
(2, 'HR'),
(3, 'Marketing'),
(4, 'Finance');

-- Create Employees Table
CREATE TABLE Employees (
  EmpID INT PRIMARY KEY,
  EmpName VARCHAR(100),
  DeptID INT,
  Salary DECIMAL(10,2),
  FOREIGN KEY (DeptID) REFERENCES Departments(DeptID)
);

-- Insert Sample Data into Employees
INSERT INTO Employees VALUES
(101, 'Alice', 1, 70000),
(102, 'Bob', 1, 80000),
(103, 'Charlie', 2, 50000),
(104, 'David', NULL, 60000), -- No department assigned
(105, 'Eve', 3, 55000);
```

```sql
-- Create Projects Table
CREATE TABLE Projects (
  ProjectID INT PRIMARY KEY,
  ProjectName VARCHAR(100),
  EmpID INT,
  FOREIGN KEY (EmpID) REFERENCES Employees(EmpID)
);

-- Insert Sample Data into Projects
INSERT INTO Projects VALUES
(201, 'AI Research', 101),
(202, 'Cloud Migration', 102),
(203, 'HR Analytics', 103),
(204, 'Marketing Automation', NULL); -- No employee assigned


 CREATE TABLE Salaries (
  SalaryID INT PRIMARY KEY,
  EmpID INT,
  Salary DECIMAL(10,2),
  EffectiveDate DATE,
  FOREIGN KEY (EmpID) REFERENCES Employees(EmpID)
);

INSERT INTO Salaries VALUES
(1, 101, 65000, '2023-01-01'),
(2, 101, 70000, '2024-01-01'),
(3, 102, 75000, '2023-01-01'),
(4, 102, 80000, '2024-01-01'),
(5, 103, 48000, '2023-01-01'),
(6, 103, 50000, '2024-01-01');
```

```
-- Managers Table
CREATE TABLE Managers (
  EmpID INT PRIMARY KEY,
  ManagerID INT,
  FOREIGN KEY (EmpID) REFERENCES Employees(EmpID),
  FOREIGN KEY (ManagerID) REFERENCES Employees(EmpID)
);

INSERT INTO Managers VALUES
(101, 102), -- Alice reports to Bob
(103, 102), -- Charlie reports to Bob
(104, 101); -- David reports to Alice

-- Employee Skills Table
CREATE TABLE EmployeeSkills (
  EmpID INT,
  Skill VARCHAR(50),
  PRIMARY KEY (EmpID, Skill),
  FOREIGN KEY (EmpID) REFERENCES Employees(EmpID)
);

INSERT INTO EmployeeSkills VALUES
(101, 'Python'),
(101, 'Machine Learning'),
(102, 'Cloud Computing'),
(102, 'DevOps'),
(103, 'HR Analytics'),
(105, 'Marketing Automation');
```

QUERIES

1. Fetch Employee Details Along with Their Department Names (INNER JOIN). Fine to ignore employees who doesn't have department

```
SELECT e.EmpID, e.EmpName, e.Salary, d.DeptName
FROM Employees e
INNER JOIN Departments d ON e.DeptID = d.DeptID;
```

Explanation: Retrieves only employees who are assigned to a department.

Expected Output:

```
EmpID  | EmpName  | Salary  | DeptName
----------------------------------------
101    | Alice    | 70000   | Engineering
102    | Bob      | 80000   | Engineering
103    | Charlie  | 50000   | HR
105    | Eve      | 55000   | Marketing
```
(Excludes David, since he has no department.)

2. Fetch All Employees and Their Departments (LEFT JOIN). Include employees who doesn't belong to any department.

```
SELECT e.EmpID, e.EmpName, e.Salary, d.DeptName
FROM Employees e
LEFT JOIN Departments d ON e.DeptID = d.DeptID;
```
Explanation: Retrieves all employees, even those without a department.

Expected Output:

```
EmpID  | EmpName  | Salary  | DeptName
----------------------------------------
101    | Alice    | 70000   | Engineering
102    | Bob      | 80000   | Engineering
103    | Charlie  | 50000   | HR
104    | David    | 60000   | NULL
105    | Eve      | 55000   | Marketing
```
(David appears with NULL for DeptName.)

3. Fetch All Departments and Employees (RIGHT JOIN). Include department where there are no employees

```
SELECT e.EmpID, e.EmpName, e.Salary, d.DeptName
FROM Employees e
RIGHT JOIN Departments d ON e.DeptID = d.DeptID;
```

Explanation: Ensures all departments are displayed, even those with no employees.

Expected Output:

```
EmpID  | EmpName  | Salary  | DeptName
---------------------------------------
101    | Alice    | 70000   | Engineering
102    | Bob      | 80000   | Engineering
103    | Charlie  | 50000   | HR
105    | Eve      | 55000   | Marketing
NULL   | NULL     | NULL    | Finance
```

(Finance appears, even though no employees work there.)

3. Fetch All Departments and Employees (RIGHT JOIN). Include both department where there are no employees and employees who doesn't belong to department

```
SELECT e.EmpID, e.EmpName, e.Salary, d.DeptName
FROM Employees e
FULL outer JOIN Departments d ON e.DeptID = d.DeptID;
```

4. Fetch All Employees and Their Projects (LEFT JOIN)

```
SELECT e.EmpID, e.EmpName, p.ProjectName
FROM Employees e
LEFT JOIN Projects p ON e.EmpID = p.EmpID;
```

Explanation: Lists all employees, including those without projects.

Expected Output:

```
EmpID  | EmpName  | ProjectName
-----------------------------------
101    | Alice    | AI Research
102    | Bob      | Cloud Migration
103    | Charlie  | HR Analytics
104    | David    | NULL
105    | Eve      | NULL
```

(David and Eve have no projects.)

5. Fetch All Projects and Their Employees (RIGHT JOIN). Includ eeven if there are no employees.

```
SELECT e.EmpName, p.ProjectName
FROM Employees e
RIGHT JOIN Projects p ON e.EmpID = p.EmpID;
```

Explanation: Lists all projects, including unassigned ones.

Expected Output:

```
EmpName  | ProjectName
-------------------------
Alice    | AI Research
Bob      | Cloud Migration
Charlie  | HR Analytics
NULL     | Marketing Automation
```

(Marketing Automation has no employee.)

6. Fetch Employees Without Projects (LEFT JOIN with NULL Condition)

```
SELECT e.EmpID, e.EmpName
FROM Employees e
LEFT JOIN Projects p ON e.EmpID = p.EmpID
WHERE p.ProjectID IS NULL;
```

Expected Output:

```
EmpID  | EmpName
------------------
104    | David
105    | Eve
```

7. Fetch Employees Working in ‘Engineering’ Department (INNER JOIN with WHERE Clause)

```
SELECT e.EmpID, e.EmpName
FROM Employees e
```

```
INNER JOIN Departments d ON e.DeptID = d.DeptID
WHERE d.DeptName = 'Engineering';
```

Expected Output:

```
EmpID  | EmpName
-----------------
101    | Alice
102    | Bob
```

8. Fetch Departments Without Employees (LEFT JOIN with NULL Condition)

```
SELECT d.DeptID, d.DeptName
FROM Departments d
LEFT JOIN Employees e ON d.DeptID = e.DeptID
WHERE e.EmpID IS NULL;
```

Expected Output:

```
DeptID  | DeptName
-------------------
4       | Finance
```

9. Fetch Number of Employees Per Department (JOIN with COUNT)

```
SELECT d.DeptName, COUNT(e.EmpID) AS EmployeeCount
FROM Departments d
LEFT JOIN Employees e ON d.DeptID = e.DeptID
GROUP BY d.DeptName;
```

Expected Output:

```
DeptName     | EmployeeCount
----------------------------
Engineering  | 2
HR           | 1
Marketing    | 1
Finance      | 0
```

10. Fetch Employees and Their Department and Project Details (Multiple Joins)

```
SELECT e.EmpName, d.DeptName, p.ProjectName
FROM Employees e
LEFT JOIN Departments d ON e.DeptID = d.DeptID
LEFT JOIN Projects p ON e.EmpID = p.EmpID;
```

Expected Output:

```
EmpName  | DeptName    | ProjectName
----------------------------------------
Alice    | Engineering  | AI Research
Bob      | Engineering  | Cloud Migration
Charlie  | HR          | HR Analytics
David    | NULL        | NULL
Eve      | Marketing   | NULL
```

11. Get Employees with Their Most Recent Salary

```
SELECT e.EmpID, e.EmpName, s.Salary, s.EffectiveDate
FROM Employees e
INNER JOIN Salaries s ON e.EmpID = s.EmpID
WHERE s.EffectiveDate = (SELECT MAX(EffectiveDate) FROM Salaries WHERE EmpID =
e.EmpID);
```

or

```
SELECT e.EmpID, e.EmpName, s.Salary, max(s.EffectiveDate)
FROM Employees e
INNER JOIN Salaries s ON e.EmpID = s.EmpID
group by e.EmpID
```

Expected Output:

```
EmpID  | EmpName  | Salary  | EffectiveDate
--------------------------------------------
101    | Alice    | 70000   | 2024-01-01
102    | Bob      | 80000   | 2024-01-01
103    | Charlie  | 50000   | 2024-01-01
```

12. Find Employees with More Than One Skill

```
SELECT EmpID, COUNT(Skill) AS SkillCount
FROM EmployeeSkills
GROUP BY EmpID
HAVING COUNT(Skill) > 1;
```

Expected Output:

```
EmpID  | SkillCount
---------------------
101    | 2
102    | 2
```

13. Get Employees and Their Manager Names

```
SELECT e.EmpName AS Employee, m.EmpName AS Manager
FROM Employees e
LEFT JOIN Managers mgr ON e.EmpID = mgr.EmpID
LEFT JOIN Employees m ON mgr.ManagerID = m.EmpID;
```

Expected Output:

```
Employee  | Manager
----------------------
Alice     | Bob
Charlie   | Bob
David     | Alicehas context menu
```