



Experiment 2

Student Name:Gourav Sharma

Branch: CSE

Semester: 5th

Subject Name: ADBMS

UID: 23BCS10857

Section/Group: KRG 3-A

Date of Performance:24/07/2025

Subject Code: 23CSP-333

1. **Aim:** To demonstrate the use of self-joins and conditional joins in SQL for managing hierarchical employee relationships and performing conditional lookups using LEFT JOIN and IFNULL across two related tables.

- a. Employee-Manager Hierarchy Using Self-Join
- b. Conditional Join Between Financial Tables

2. **Objective:**

The objective of this SQL code is to develop a practical understanding of intermediate relational database concepts through two realistic scenarios. The first part demonstrates employee-management relationships using a self-join on a single table, showing how hierarchical data can be modeled and queried effectively. The second part showcases conditional data retrieval using LEFT JOIN and IFNULL to extract NPV values from time-based financial records, even when some data is missing. Together, these examples emphasize efficient data modeling, referential integrity, and robust querying practices.

3. **DBMS script and output:**

Solution-(a)

-- Medium Level Problem

```
CREATE TABLE Employee (  
    EmpID INTEGER PRIMARY KEY,  
    EmpName TEXT NOT NULL,  
    Department TEXT NOT NULL,  
    ManagerID INTEGER,  
    FOREIGN KEY (ManagerID) REFERENCES Employee(EmpID)  
);
```

```
INSERT INTO Employee (EmpID, EmpName, Department, ManagerID)  
VALUES  
(1, 'Alice', 'HR', NULL),  
(2, 'Bob', 'Finance', 1),  
(3, 'Charlie', 'IT', 1),  
(4, 'David', 'Finance', 2),  
(5, 'Eve', 'IT', 3),  
(6, 'Frank', 'HR', 1);
```

```
SELECT  
    E.EmpName AS EmployeeName,  
    E.Department AS EmployeeDept,  
    M.EmpName AS ManagerName,  
    M.Department AS ManagerDept  
FROM  
    Employee E  
JOIN  
    Employee M  
ON
```

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

E.ManagerID = M.EmpID;

| EmployeeName | EmployeeDept | ManagerName | ManagerDept |
|--------------|--------------|-------------|-------------|
| Bob | Finance | Alice | HR |
| Charlie | IT | Alice | HR |
| David | Finance | Bob | Finance |
| Eve | IT | Charlie | IT |
| Frank | HR | Alice | HR |

Solution-(b)

-- Hard Level Problem

```
CREATE TABLE TBL_YEAR (  
    ID INT,  
    YEAR INT,  
    NPV INT  
);
```

```
CREATE TABLE TBL_QUERY (  
    ID INT,  
    YEAR INT  
);
```

```
INSERT INTO TBL_YEAR (ID, YEAR, NPV)  
VALUES  
(1, 2018, 100),  
(7, 2020, 30),  
(13, 2019, 40),  
(1, 2019, 113),  
(2, 2008, 121),  
(3, 2009, 12),  
(11, 2020, 99),  
(7, 2019, 0);
```

```
INSERT INTO TBL_QUERY (ID, YEAR)  
VALUES  
(1, 2019),  
(2, 2008),  
(3, 2009),  
(7, 2018),  
(7, 2019),  
(7, 2020),  
(13, 2019);
```

```
SELECT  
    q.ID,  
    q.YEAR,  
    IFNULL(y.NPV, 0) AS NPV_Value  
FROM  
    TBL_QUERY q  
LEFT JOIN  
    TBL_YEAR y  
ON  
    q.ID = y.ID AND q.YEAR = y.YEAR;
```

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

| ID | YEAR | NPV_Value |
|----|------|-----------|
| 1 | 2019 | 113 |
| 2 | 2008 | 121 |
| 3 | 2009 | 12 |
| 7 | 2018 | 0 |
| 7 | 2019 | 0 |
| 7 | 2020 | 30 |
| 13 | 2019 | 40 |

4. Learning Outcomes (What I have Learnt):

- To demonstrate the use of **self-joins** in handling hierarchical relationships like employee–manager mappings within a single table.
- To understand how to define **foreign key constraints** to maintain referential integrity between entities.
- To use **LEFT JOINs with NULL handling (IFNULL)** for matching records across related datasets and managing missing data gracefully.
- To apply **real-world data modeling** scenarios, such as employee reporting structure and year-wise data lookups.
- To enhance proficiency in **querying and combining data** from multiple tables using conditional logic and joins.