# Experiment 2

**Student Name: Akshay Guleria** **UID: 23BCS10517**
**Branch: CSE** **Section/Group: KRG 3-A**
**Semester: 5th** **Date of Performance:24/07/2025**
**Subject Name: ADBMS** **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:**

- To design and populate relational tables with hierarchical and temporal data.
- To perform a **self-join** on an employee table to retrieve manager-employee relationships.
- To implement a **conditional LEFT JOIN** between two tables to handle non-matching records.
- To apply the **IFNULL** function to handle missing values in joined queries.
- To practice using joins for **querying structured business-related datasets**.

## 3. DBMS script and output:

**Solution-(a)**

```
CREATE DATABASE company;
USE company;

CREATE TABLE employee (
    empid INT PRIMARY KEY,
    ename VARCHAR(50),
    department VARCHAR(50),
    managerid INT
);
INSERT INTO employee (empid, ename, 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.ename AS EmployeeName,
    e.department AS EmployeeDepartment,
    m.ename AS ManagerName,
    m.department AS ManagerDepartment
FROM
    employee e
LEFT JOIN
    employee m ON e.managerid = m.empid;
```

| EMPLOYEENAME | EMPLOYEEDEPARTMENT | MANAGERNAME | MANAGERDEPARTMENT |
|---|---|---|---|
| Frank | HR | Alice | HR |
| Charlie | IT | Alice | HR |
| Bob | Finance | Alice | HR |
| David | Finance | Bob | Finance |
| Eve | IT | Charlie | IT |
| Alice | HR | - | - |

**Solution-(b)**

```
create database company2;
use  company2;
CREATE TABLE Year_tbl (
    ID INT,
    YEAR INT,
    NPV INT
);
```

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

INSERT INTO Year_tbl (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 Queries (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
FROM
    Queries q
LEFT JOIN
    Year_tbl y ON q.ID = y.ID AND q.YEAR = y.YEAR;
```

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

## 4. Learning Outcomes:

- Understand how to model and query **hierarchical relationships** using self-joins.
- Learn to perform **LEFT JOINs** to include unmatched records from one table.
- Apply **composite join conditions** on multiple columns (e.g., ID and YEAR).
- Use **IFNULL** to handle NULL values in result sets for reporting purposes.
- Develop SQL skills for solving **real-world data retrieval scenarios** in organizations.