# Experiment 2

**Student Name:** Muskan

**UID:**23BAI70172

**Branch:** BE-AIT-CSE

**Section/Group:**23AML-1(A)

**Semester:**5th

**Date of Performance:**30 july,2025

**Subject Name:**ADBMS

**Subject Code:**23CSP-333

## MEDIUM LEVEL

You are a Database Engineer at TalentTree Inc., an enterprise HR analytics platform that stores employee data, including their reporting relationships. The company maintains a centralized Employee relation that holds:
Each employee's ID, name, department, and manager ID (who is also an employee in the same table).

Your task is to generate a report that maps employees to their respective managers, showing:

The employee's name and department

Their manager's name and department (if applicable)

This will help the HR department visualize the internal reporting hierarchy.

**PROCEDURE:**

1. **Design the Employee table** with columns for employee ID, name, department, and manager ID.
2. **Insert sample employee records** into the table, ensuring some employees have managers.
3. **Write a SQL query to self-join** the Employee table to itself to map each employee to their manager.
4. Use a **LEFT JOIN** to ensure employees without a manager are still included (manager fields are shown as NULL).
5. **Display each employee's name and department** along with their manager's name and department.

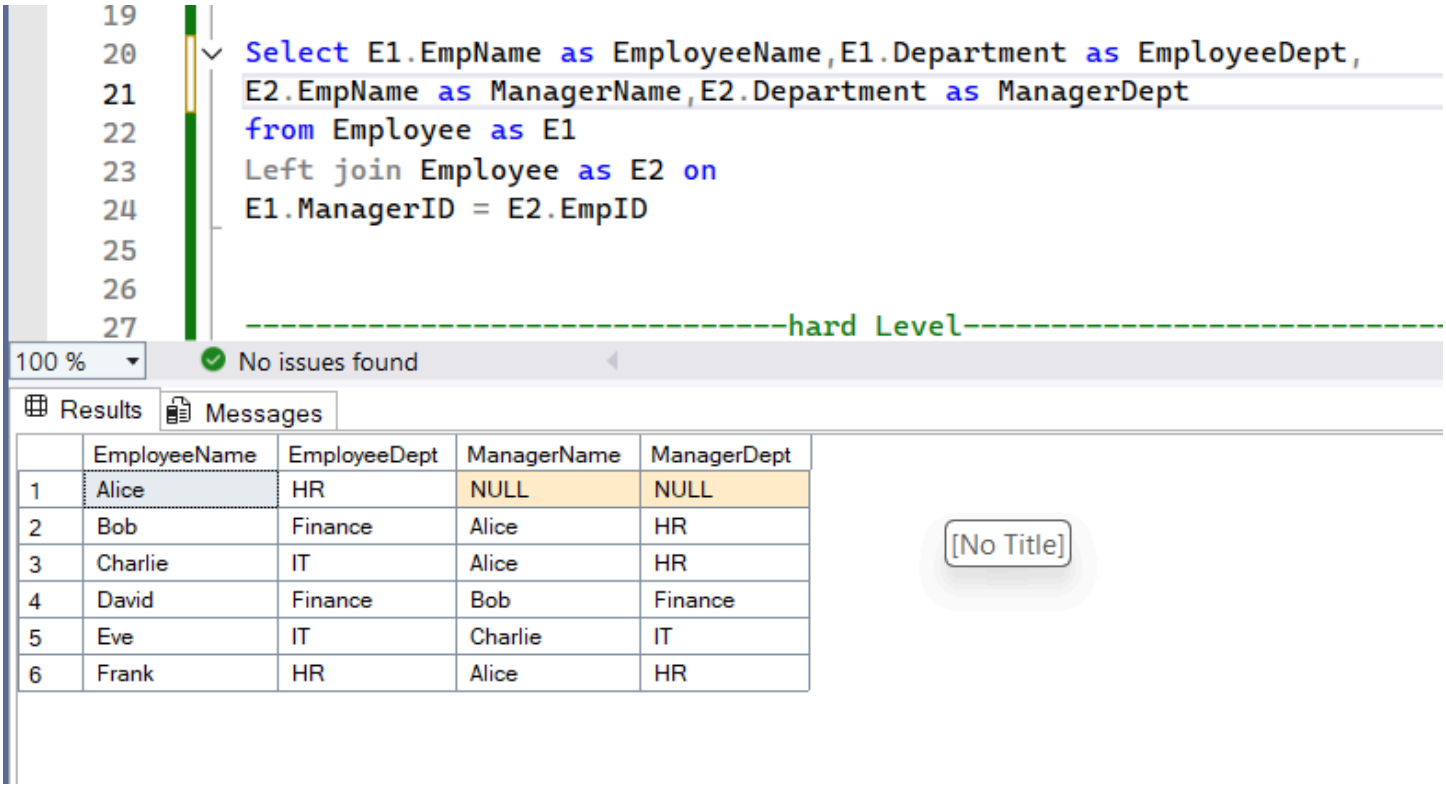
a) Creation of table and inserting values

```sql
create database exp2;
use exp2;

CREATE TABLE Employee (
    EmpID INT PRIMARY KEY,
    EmpName VARCHAR(50) NOT NULL,
    Department VARCHAR(50) NOT NULL,
    ManagerID INT NULL
);

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);
```

b) maping employees to their respective managers

```sql
Select E1.EmpName as EmployeeName,E1.Department as EmployeeDept,
E2.EmpName as ManagerName,E2.Department as ManagerDept
from Employee as E1
Left join Employee as E2 on
E1.ManagerID = E2.EmpID
```

OUTPUT:



```sql
Select E1.EmpName as EmployeeName,E1.Department as EmployeeDept,
E2.EmpName as ManagerName,E2.Department as ManagerDept
from Employee as E1
Left join Employee as E2 on
E1.ManagerID = E2.EmpID


--------------------------------hard Level----------------------------
```

100 % No issues found

Results Messages

| | EmployeeName | EmployeeDept | ManagerName | ManagerDept |
|---|---|---|---|---|
| 1 | Alice | HR | NULL | NULL |
| 2 | Bob | Finance | Alice | HR |
| 3 | Charlie | IT | Alice | HR |
| 4 | David | Finance | Bob | Finance |
| 5 | Eve | IT | Charlie | IT |
| 6 | Frank | HR | Alice | HR |

[No Title]

# Learning Outcome:

- Understand how to represent **hierarchical relationships** (like employee-manager) in a relational database.

- Learn how to use **self joins** to map employees to their managers.
- Practice using **LEFT JOIN** to include all records, even when a relationship (manager) is missing.
- Gain skills in writing and understanding SQL queries for organizational reporting.

# MEDIUM LEVEL

You are a Data Engineer at FinSight Corp, a company that models Net Present Value (NPV) projections for investment decisions. Your system maintains two key datasets:

1.Year_tbl: Actual recorded NPV's of various financial instruments over different years:

ID: Unique Financial instrument identifier.

YEAR: Year of record

NPV: Net Present Value in that year

2.Queries_tbl: A list of instrument-year pairs for which stakeholders are requesting NPV values:

ID: Financial instrument identifier

YEAR: Year of interest.

Find the NPV of each query from the Queries table. Return the output order by ID and Year in the sorted form.

However, not all ID-YEAR combinations in the Queries table are present in the Year_tbl. If an NPV is missing for a requested combination, assume it to be 0 to maintain a consistent financial report.

**PROCEDURE:**

1. **Create the Year_tbl table** to store NPV values for each instrument and year.
2. **Insert NPV data** into Year_tbl for various instrument-year combinations.
3. **Create the Queries_tbl table** to hold those instrument-year pairs for which NPV needs to be reported.
4. **Insert queries/IDs into Queries_tbl** that require NPV lookup.
5. **Write a SQL LEFT JOIN query** to match each query pair with Year_tbl.
6. Using **ISNULL** to replace the null values to show '0' if the NPV value is missing for the given combination.
7. **Order the results by ID and year** to prepare a clear report for stakeholders.

**a) Creation of tables and inserting values:**

```
----------------------------Hard Level---------------------------------
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);
```

**b) NPV of each query from the Queries table sorted by id and year**

```
Select Q.ID as ID,Q.YEAR as year,ISNULL(Y.NPV,0)
from Queries as Q
Left join year_tbl as Y
on Q.ID = y.ID and Q.YEAR = Y.YEAR ORDER BY Q.ID,Q.year;
```

**output:**

```sql
Select Q.ID as ID,Q.YEAR as year,ISNULL(Y.NPV,0)
from Queries as Q
Left join year_tbl as Y
on Q.ID = y.ID and Q.YEAR = Y.YEAR ORDER BY Q.ID,Q.year;
```

100 % No issues found

Results Messages

| | ID | year | (No column name) |
|---|---|---|---|
| 1 | 1 | 2019 | 113 |
| 2 | 2 | 2008 | 121 |
| 3 | 3 | 2009 | 12 |
| 4 | 7 | 2018 | 0 |
| 5 | 7 | 2019 | 0 |
| 6 | 7 | 2020 | 30 |
| 7 | 13 | 2019 | 40 |

## Learning Outcome:

- Learn to handle **missing data** using SQL functions like COALESCE.
- Understand how to combine multiple tables using **LEFT JOIN** based on multiple columns.
- Gain experience with **report generation** in databases, including ordering and conditional default values.
- Practice thinking like a data engineer solving real business reporting needs