EXPERIMENT-02

Student Name: DIVYANSH **UID: 23BCS11778**

Branch: BE-CSE **Section/Group: KRG 1(A)**

Semester: 05 **Date of Performance: 31/07/25**

Subject Name: ADBMS **Subject Code: 23CSP-333**

1. Aim:

Organizational Hierarchy Explorer

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).

2. Objective:

- To understand and apply self-join operations on a single table.
- To represent hierarchical relationships (employee-manager) within the same entity.
- To visualize organizational structure using SQL by mapping employees with their respective managers.

3. Code:

```
CREATE TABLE Employee (
    EmpID INT PRIMARY KEY,
    EmpName VARCHAR(50) NOT NULL,
    Department VARCHAR(50) NOT NULL,
    ManagerID INT NULL  -- Self-reference to EmpID
);
```

```
ALTER TABLE Employee
ADD CONSTRAINT FK_Manager FOREIGN KEY (ManagerID) REFERENCES Employee(EmpID);


-- Insert data into Employee table
INSERT INTO Employee (EmpID, EmpName, Department, ManagerID)
VALUES
(1, 'Alice', 'HR', NULL),        -- Top-level manager
(2, 'Bob', 'Finance', 1),
(3, 'Charlie', 'IT', 1),
(4, 'David', 'Finance', 2),
(5, 'Eve', 'IT', 3),
(6, 'Frank', 'HR', 1);


SELECT E1.EmpName AS [EMPLOYEEE_NAME], E2.EmpName AS [MANAGER_NAME], E1.Department AS
[EMP_DEPT], E2.Department AS [MANAGER_DEPT]
FROM EMPLOYEE AS E1
INNER JOIN
EMPLOYEE AS E2
ON
E1.ManagerID = E2.EmpID;
```

4. Output:

Results Messages

	EMPLOYEEE_NAME	MANAGER_NAME	EMP_DEPT	MANAGER_DEPT
1	Bob	Alice	Finance	HR
2	Charlie	Alice	IT	HR
3	David	Bob	Finance	Finance
4	Eve	Charlie	IT	IT
5	Frank	Alice	HR	HR

Financial Forecast Matching with Fallback Strategy (Hard)

1. Aim:

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.

2. Objective:

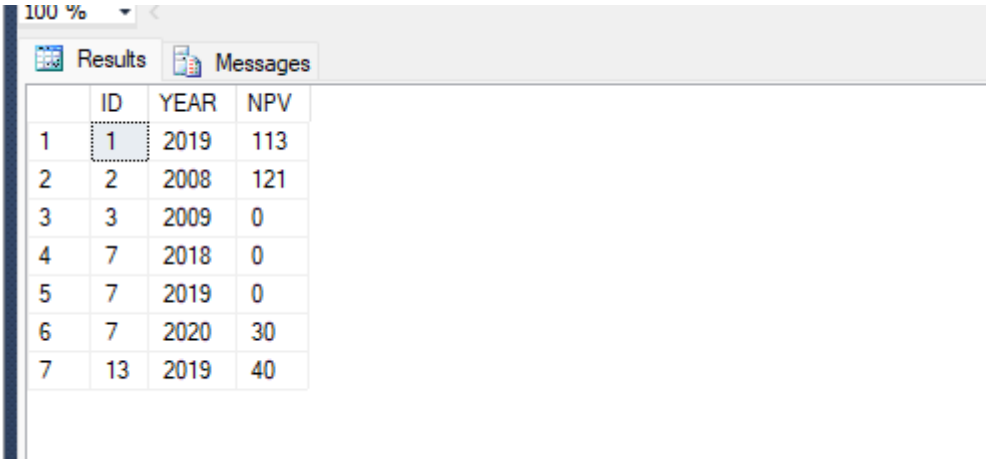
- To retrieve financial data by performing joins across multiple datasets.
- To handle missing data scenarios using fallback strategies like ISNULL() in SQL.
- To understand and apply LEFT JOIN operations for data reconciliation.
- To ensure accurate and complete reporting of Net Present Values (NPV) even when data is unavailable.
- To return results in a sorted and standardized format, facilitating better decision- making in financial forecasting.

3. Code:

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

CREATE TABLE QUERIES_TBL(
ID INT,
YEAR INT
);

INSERT INTO YEARS_TBL(ID, YEAR, NPV)
VALUES
(1,2018,100),
(7,2020,30),
(13,2019,40),
(1,2019,113),
(2,2008,121),
(3,2002,12),
(11,2020,99),
(7,2019,0);


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

SELECT Q.*,ISNULL(Y.NPV,0) AS [NPV]
FROM
YEARS_TBL AS Y
RIGHT OUTER JOIN
QUERIES_TBL AS Q
ON
```

```
Y.ID = Q.ID
AND
Y.YEAR = Q.YEAR
```

4. Output:



	ID	YEAR	NPV
1	1	2019	113
2	2	2008	121
3	3	2009	0
4	7	2018	0
5	7	2019	0
6	7	2020	30
7	13	2019	40