Experiment 1.2

Student Name: Rishi Jain

UID : 23BAI70569

Branch: BE-AIT-CSE

Section/Group : 23AIT_KRG_G-1

Semester: 5th

Date of Performance: 30 July 2025

Subject Name: ADBMS

Subject Code: 23CSP-333

MEDIUM - LEVEL

1. **Problem Title:** Organizational Hierarchy Explorer
2. **Problem Description:** 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:
   a. The employee's name and department
   b. Their manager's name and department (if applicable)
   c. This will help the HR department visualize the internal reporting hierarchy.
3. **SQL Commands:**
   a. Create the database and use it:

```
create database AIT_1A;
use AIT_1A;
```

   b. Create tables Employee and adding Foreign key:

```
    CREATE TABLE Employee (
        EmpID INT PRIMARY KEY,
        EmpName VARCHAR(50) NOT NULL,
        Department VARCHAR(50) NOT NULL,
        ManagerID INT NULL);
ALTER TABLE Employee ADD CONSTRAINT FK_EMPLOYEE
FOREIGN KEY (ManagerID)
references Employee(EmpID);
```

c. Insert the values in the tables:

```
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);
```

d. Selecting the Employee with their respective managers:

```
SELECT E1.EmpName [Employee Name], E2.EmpName [Manager Name],
E1.Department [Emp_dept],
E1.Department [Manager_dept]
from Employee as E1
    left outer join Employee as E2 on E1.ManagerID = E2.EmpID;
```

4. Output:

	Name	Owner	Type	Created_datetime
1	Employee	dbo	user table	2025-07-30 10:17:25.060

	Column_name	Type	Computed	Length	Prec	Scale	Nullable	TrimTrailingBlanks	FixedLenNullInSource	Collation
1	EmpID	int	no	4	10	0	no	(n/a)	(n/a)	NULL
2	EmpName	varchar	no	50			no	no	no	SQL_Latin1_General_CP1_CI_AS
3	Department	varchar	no	50			no	no	no	SQL_Latin1_General_CP1_CI_AS
4	ManagerID	int	no	4	10	0	yes	(n/a)	(n/a)	NULL

Figure 1 Employee Table Description

	Employee Name	Manager Name	Emp_dept	Manager_dept
1	Alice	NULL	HR	HR
2	Bob	Alice	Finance	Finance
3	Charlie	Alice	IT	IT
4	David	Bob	Finance	Finance
5	Eve	Charlie	IT	IT
6	Frank	Alice	HR	HR

Figure 2 Output of the Select Query

5. Learning Outcome:

a. I learnt how to link and add constraints like primary key after the table creation.
b. I learnt about different types of joints.
c. I learnt how to use LEFT OUTER JOIN to retrieve combined data from related tables.

HARD - LEVEL

1. **Problem Title:** Financial Forecast Matching with Fallback Strategy
2. **Problem Description:** 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:
   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
   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.
3. **SQL Commands:**
   a. Create the tables.

```sql
CREATE TABLE Year_tbl (
    ID INT,
    YEAR INT,
    NPV INT
);

-- Create Queries table (requested values)
CREATE TABLE Queries (
    ID INT,
    YEAR INT
);
```

   b. Insert the values.

```sql
    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);
```

c. Use a subquery to count the number of courses under each department.

```
select q.id, q.year, Isnull(y.NPV, 0) [NPV]
from Queries as q
left outer join Year_tbl as y on q.id = y.id and q.YEAR = y.YEAR
    order by q.id;
```

4. Output:

	Name	Owner	Type	Created_datetime
1	Year_tbl	dbo	user table	2025-07-30 10:28:07.473

	Column_name	Type	Computed	Length	Prec	Scale	Nullable	TrimTrailingBlanks	FixedLenNullInSource	Collation
1	ID	int	no	4	10	0	yes	(n/a)	(n/a)	NULL
2	YEAR	int	no	4	10	0	yes	(n/a)	(n/a)	NULL
3	NPV	int	no	4	10	0	yes	(n/a)	(n/a)	NULL

Figure 1 Year_tbl description

	Name	Owner	Type	Created_datetime
1	Queries	dbo	user table	2025-07-30 10:28:12.023

	Column_name	Type	Computed	Length	Prec	Scale	Nullable	TrimTrailingBlanks	FixedLenNullInSource	Collation
1	ID	int	no	4	10	0	yes	(n/a)	(n/a)	NULL
2	YEAR	int	no	4	10	0	yes	(n/a)	(n/a)	NULL

Figure 2 Queries table description

	id	year	NPV
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

Figure 3 Select Query

5. Learning Outcomes:

a. I learned how to perform left join and understand the table.
b. I learned some of the build functions of the Microsoft SQL server.
c. I learned about aliases in the SQL queries.