Experiment 1.2

Student Name: Shivam Kumar **UID: 23BCS14207**
Branch: CSE **Section/Group: KRG – 1B**
Semester: 5th **Date of Performance: 29/07/25**
Subject Name: ADBMS **Subject Code: 23CSP-333**

1. **Aim**: To implement and analyze SQL join operations for real-world scenarios involving employee reporting structures and financial record forecasting with fallback mechanisms.

2. **Requirements(Hardware/Software):** MySQL, PostgreSQL, Oracle, or SQL Server

3. **DBMS script and output:**

Medium-Level Problem

***Problem Title:** Organizational Hierarchy Explorer*

Problem Statement:

You are a database engineer at TalentTree Inc., tasked with creating an internal tool that maps employees to their managers. You must design a query using a self-join on a single Employee table to fetch each employee's name and department, along with their corresponding manager's name and department.

CODE:

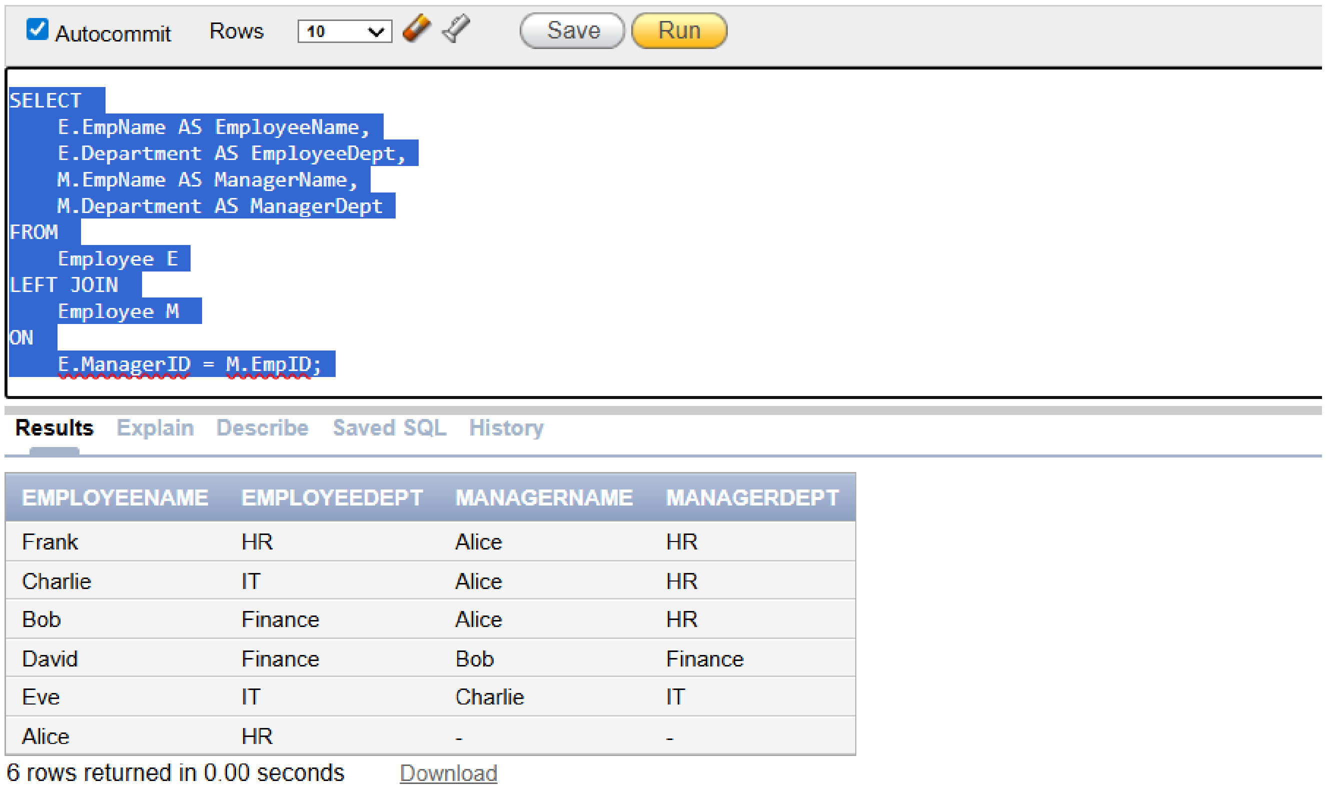
```
CREATE TABLE Employee (
    EmpID INT PRIMARY KEY,
    EmpName VARCHAR(50) NOT NULL,
    Department VARCHAR(50) NOT NULL,
    ManagerID INT NULL
);
```

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

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

SELECT
   E.EmpName AS EmployeeName,
   E.Department AS EmployeeDept,
   M.EmpName AS ManagerName,
   M.Department AS ManagerDept
FROM
   Employee E
LEFT JOIN
   Employee M
ON
   E.ManagerID = M.EmpID;
```

OUTPUT:



```
SELECT
    E.EmpName AS EmployeeName,
    E.Department AS EmployeeDept,
    M.EmpName AS ManagerName,
    M.Department AS ManagerDept
FROM
    Employee E
LEFT JOIN
    Employee M
ON
    E.ManagerID = M.EmpID;
```

EMPLOYEENAME	EMPLOYEEDEPT	MANAGERNAME	MANAGERDEPT
Frank	HR	Alice	HR
Charlie	IT	Alice	HR
Bob	Finance	Alice	HR
David	Finance	Bob	Finance
Eve	IT	Charlie	IT
Alice	HR	-	-

6 rows returned in 0.00 seconds

Hard-Level Problem

Problem Title: *Financial Forecast Matching with Fallback*

Problem Statement: Financial Forecast Matching with Fallback

You are a data engineer at FinSight Corp.. Your role is to retrieve NPV values for a list of financial instruments and years. If any ID-YEAR combination from the Queries_tbl is missing in the Year_tbl, you must return NPV as 0 to ensure completeness. Use LEFT JOIN along with ISNULL or COALESCE.

CODE:

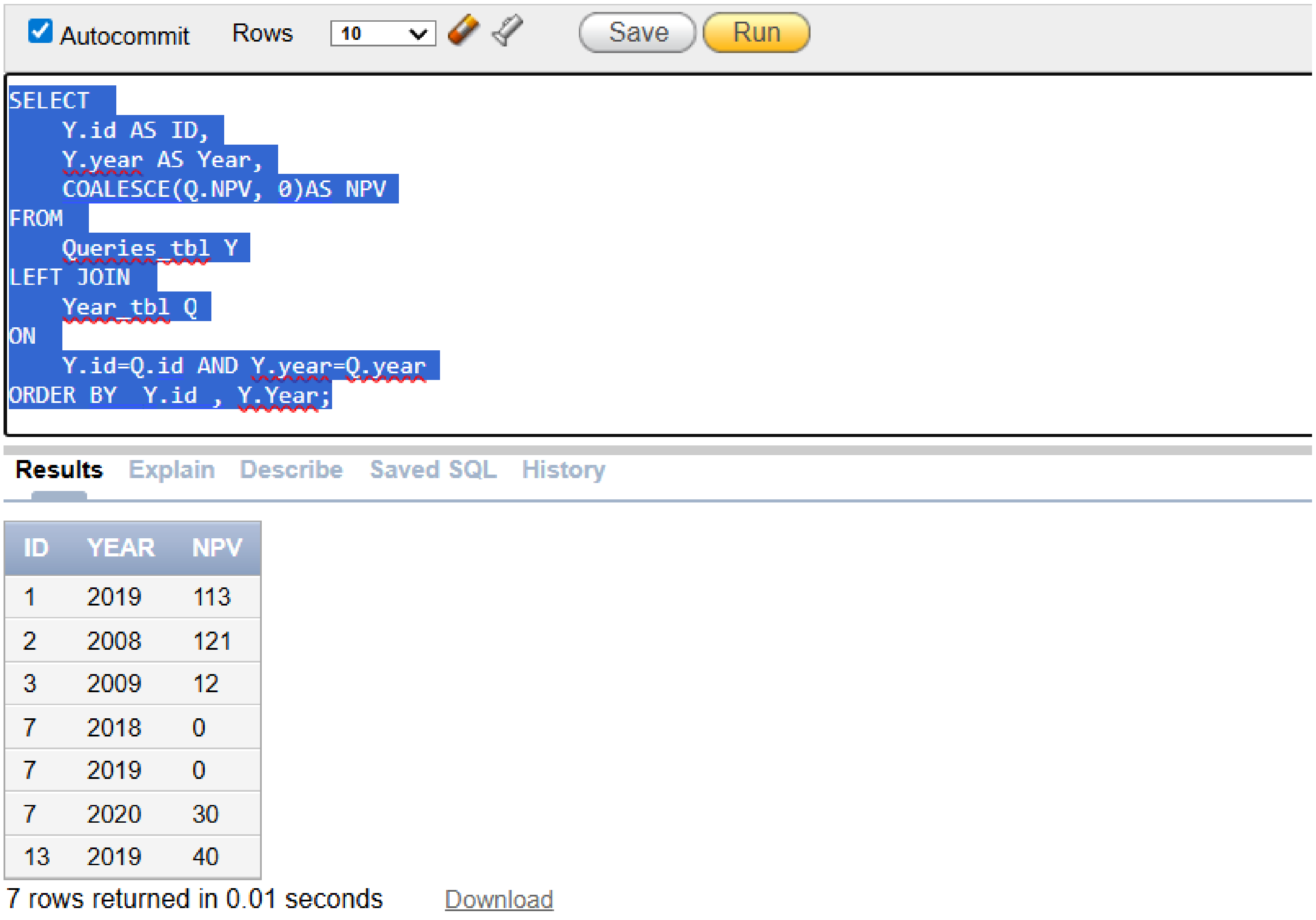
```
CREATE TABLE Year_tbl (
   id INT,
   year INT,
   NPV INT
);

INSERT INTO Year_tbl (id, year, NPV) VALUES (1, 2018, 100);
INSERT INTO Year_tbl (id, year, NPV) VALUES (7, 2020, 30);
INSERT INTO Year_tbl (id, year, NPV) VALUES (13, 2019, 40);
INSERT INTO Year_tbl (id, year, NPV) VALUES (1, 2019, 113);
INSERT INTO Year_tbl (id, year, NPV) VALUES (2, 2008, 121);
INSERT INTO Year_tbl (id, year, NPV) VALUES (3, 2009, 12);
INSERT INTO Year_tbl (id, year, NPV) VALUES (11, 2020, 99);
INSERT INTO Year_tbl (id, year, NPV) VALUES (7, 2019, 0);

CREATE TABLE Queries_tbl (
   id INT,
   year INT
);

INSERT INTO Queries_tbl (id, year) VALUES (1, 2019);
INSERT INTO Queries_tbl (id, year) VALUES (2, 2008);
INSERT INTO Queries_tbl (id, year) VALUES (3, 2009);
INSERT INTO Queries_tbl (id, year) VALUES (7, 2018);
INSERT INTO Queries_tbl (id, year) VALUES (7, 2019);
INSERT INTO Queries_tbl (id, year) VALUES (7, 2020);
INSERT INTO Queries_tbl (id, year) VALUES (13, 2019);

SELECT
   Y.id AS ID,
   Y.year AS Year,
   COALESCE(Q.NPV, 0)AS NPV
FROM
   Queries_tbl Y
LEFT JOIN
   Year_tbl Q
ON
   Y.id=Q.id AND Y.year=Q.year
ORDER BY  Y.id , Y.Year;
```



```
SELECT
    Y.id AS ID,
    Y.year AS Year,
    COALESCE(Q.NPV, 0)AS NPV
FROM
    Queries_tbl Y
LEFT JOIN
    Year_tbl Q
ON
    Y.id=Q.id AND Y.year=Q.year
ORDER BY  Y.id , Y.Year;
```

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

7 rows returned in 0.01 seconds

4. Learning Outcomes :

- Learned how to perform **`LEFT JOIN`** to merge data from two related tables.
- Applied `COALESCE()` to handle and replace **`NULL`** values in query results.
- Understood Oracle-specific syntax rules for table aliasing.
- Practiced ordering query results using `ORDER BY` clause.
- Gained experience in building queries for real-world data fallback scenarios.