

# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING



Discover. Learn. Empower.

## Assignment -2.1

**Name:** Shubh Rai

**UID:** 23BCS12916

**Subject:** DBMS

**Semester:** 5th

**Section:** KRG - 3B

**Date of performance:** 27th July 2025

**Branch:** BE - CSE

### 1.1 Medium problem:

#### 1: AIM:

- Write an SQL query using a self join to:
- Select the employee's name, aliased as employee\_name
- Select the manager's name, aliased as manager\_name
- Select the employee's department, aliased as employee\_department
- Select the manager's department, aliased as manager\_department
- Include all employees, even if they do not have a manager
- Use a LEFT OUTER JOIN between the employee table and itself, with the correct join condition.

#### 2: CODE

```
create table employee( e_id int primary key, name
varchar(10), department varchar(20), manager_id int,
foreign key (manager_id) references employee
(e_id),
);
```

```
insert into employee(e_id, name, department, manager_id)
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 e1.name as [employee_ Name] ,e2.name as[manager_name]
,
e1.department as[employee_depart] ,
e2.department as[manager_department] from
employee as e1
left outer join
employee e2
on
e1.e_id= e2.manager_id;
```

### **3: OUTPUT:**

```
STDIN
Input for the program ( Optional )

Output:
employee_ Name manager_name employee_depart manager_department
-----
alice          BOB          HR          finance
alice          charlie       HR          IT
alice          Frank        HR          HR
BOB            david        finance     finance
charlie        eve          IT          IT
david          NULL         finance     NULL
eve            NULL         IT          NULL
Frank          NULL         HR          NULL
```

#### 4: LEARNING OUTCOME:

- Understand how to create **self-referencing foreign keys** to represent hierarchical relationships (e.g., employee–manager).
- Learn to use **table aliases** when joining a table with itself.
- Apply **LEFT OUTER JOIN** to retrieve all records from one table even if there's no matching record in the joined table.
- Retrieve **employee-manager relationships** along with their respective departments.
- Understand how to model and query **organizational structures** using SQL.