In **PostgreSQL**, if you have two tables:

1. employees - containing details about employees, including their manager.
2. managers - containing details about managers.

**Case 1: If Both Employees and Managers Are in the Same Table**

If the employees table has a **self-referencing foreign key** (i.e., employees have a manager\_id that refers to another employee's id), you can use **self-join**:

**Schema Example:**

sql

CopyEdit

CREATE TABLE employees (

emp\_id SERIAL PRIMARY KEY,

emp\_name VARCHAR(100),

manager\_id INT REFERENCES employees(emp\_id)

);

**Query to Get Employees Under Each Manager:**

sql

CopyEdit

SELECT e1.emp\_name AS Employee,

e2.emp\_name AS Manager

FROM employees e1

LEFT JOIN employees e2

ON e1.manager\_id = e2.emp\_id

ORDER BY e2.emp\_name; -- Sorting by Manager Name

**Case 2: If Managers Are in a Separate Table**

If you have an employees table and a separate managers table:

**Schema Example:**

sql

CopyEdit

CREATE TABLE managers (

manager\_id SERIAL PRIMARY KEY,

manager\_name VARCHAR(100)

);

CREATE TABLE employees (

emp\_id SERIAL PRIMARY KEY,

emp\_name VARCHAR(100),

manager\_id INT REFERENCES managers(manager\_id)

);

**Query to Get Employees Under Each Manager:**

sql

CopyEdit

SELECT e.emp\_name AS Employee,

m.manager\_name AS Manager

FROM employees e

JOIN managers m

ON e.manager\_id = m.manager\_id

ORDER BY m.manager\_name;

**Alternative: Grouping Employees Under Managers**

If you want to **list employees grouped by manager**, use:

sql

CopyEdit

SELECT m.manager\_name,

STRING\_AGG(e.emp\_name, ', ') AS Employees

FROM employees e

JOIN managers m

ON e.manager\_id = m.manager\_id

GROUP BY m.manager\_name;

Let me know if you need any modifications based on your table structure! 🚀