### **Lab 21- JOIN in PostgreSQL**

Below is a complete lab exercise demonstrating INNER JOIN, LEFT JOIN, RIGHT JOIN, and FULL OUTER JOIN on the employees and departments tables in PostgreSQL:

Assuming we have the following table structures for employees and departments:

CREATE TABLE departments (

department\_id INTEGER PRIMARY KEY,

department\_name VARCHAR(50) NOT NULL

);

CREATE TABLE employees (

employee\_id INTEGER PRIMARY KEY,

employee\_name VARCHAR(100) NOT NULL,

department\_id INTEGER REFERENCES departments(department\_id)

);

Now, let's perform the lab exercise step by step:

**Populating the tables with sample data:**

**-- Inserting sample data into departments table**

INSERT INTO departments VALUES (1, 'HR'), (2, 'IT'), (3, 'Marketing');

**-- Inserting sample data into employees table**

INSERT INTO employees VALUES

(1, 'John Doe', 1),

(2, 'Jane Smith', 2),

(3, 'Michael Johnson', 1),

(4, 'Emily Williams', 3);

**Using INNER JOIN:**

SELECT e.employee\_id, e.employee\_name, d.department\_name

FROM employees e

INNER JOIN departments d ON e.department\_id = d.department\_id;

**Using LEFT JOIN:**

SELECT e.employee\_id, e.employee\_name, d.department\_name

FROM employees e

LEFT JOIN departments d ON e.department\_id = d.department\_id;

**Using RIGHT JOIN:**

SELECT e.employee\_id, e.employee\_name, d.department\_name

FROM employees e

RIGHT JOIN departments d ON e.department\_id = d.department\_id;

**Using FULL OUTER JOIN:**

SELECT e.employee\_id, e.employee\_name, d.department\_name

FROM employees e

FULL OUTER JOIN departments d ON e.department\_id = d.department\_id;

By executing these SQL commands, you can understand how each type of join operates on the employees and departments tables. Adjust the queries according to your specific requirements.