Experiment: SQL Queries Using Joins

1. AIM

To construct and execute SQL queries using various types of joins such as INNER JOIN, LEFT JOIN, RIGHT JOIN, FULL JOIN, SELF JOIN, and CROSS JOIN to retrieve data from multiple tables.

2. ALGORITHM

Step 1: Identify the tables required for performing the join operations.

Step 2: Determine the common column(s) used for joining the tables.

Step 3: Choose the appropriate type of join based on the required output: INNER, LEFT, RIGHT, FULL, SELF, or CROSS JOIN.

Step 4: Write the SQL join query using the JOIN keyword with the ON clause.

Step 5: Execute the join query in the SQL environment.

Step 6: Verify that the retrieved data matches the expected join output.

3. PROGRAM

- Example 1: INNER JOIN

```
SELECT e.employee_name, d.dept_name
FROM employees e
INNER JOIN departments d
ON e.dept_id = d.dept_id;
```

- Example 2: LEFT JOIN

```
SELECT e.employee_name, d.dept_name
FROM employees e
LEFT JOIN departments d
ON e.dept_id = d.dept_id;
```

- Example 3: RIGHT JOIN

```
SELECT e.employee_name, d.dept_name
FROM employees e
RIGHT JOIN departments d
ON e.dept_id = d.dept_id;
```

- Example 4: FULL OUTER JOIN

```
SELECT e.employee_name, d.dept_name
FROM employees e
```

```
FULL OUTER JOIN departments d
ON e.dept_id = d.dept_id;
```

- Example 5: SELF JOIN

```
SELECT e.employee_name AS Employee, m.employee_name AS Manager
FROM employees e
JOIN employees m
ON e.manager_id = m.employee_id;
```

- Example 6: CROSS JOIN

```
SELECT e.employee_name, d.dept_name
FROM employees e
CROSS JOIN departments d;
```

- Example 7: Join with Multiple Conditions

```
SELECT e.employee_name, d.dept_name
FROM employees e
JOIN departments d
ON e.dept_id = d.dept_id AND e.location = d.location;
```

- Example 8: Join Among Three Tables

```
SELECT e.employee_name, d.dept_name, l.city
FROM employees e
JOIN departments d ON e.dept_id = d.dept_id
JOIN locations l ON d.location_id = l.location_id;
```