1> creating tables dept and employee for performing JOIN OPERATIONS in mysql

#creating table dept

```
mysql>create table dept(dept_id int primary key,
dept_name varchar(30));
```

#creating employee table;

```
mysql> create table employee(emp_id int primary key,
        f_name varchar(60),l_name varchar(50),
        dept_id int,foreign key (dept_id) references dept(dept_id));
```

#inserting values

```
mysql> insert into dept values(10,"HR"),
(11,"Sales"),
(30,"IT"),
(40,"Marketing");
```

```
> insert into employee values(1,"jhon","doe",10),
(2,"jane","smith",20),(3,"mike","johnson",30),
(4,"emily","davis",40);
```

JOINS:

MySQL JOINS are used with SELECT statement. It is used to retrieve data from multiple tables. It is performed whenever you need to fetch records from two or more tables.

There are three types of MySQL joins:

- MySQL INNER JOIN (or sometimes called simple join)
- MySQL LEFT OUTER JOIN (or sometimes called LEFT JOIN)
- MySQL RIGHT OUTER JOIN (or sometimes called RIGHT JOIN)
- MySQL FULL OUTER JOIN(we cannot implement full outer join by using full outer keyword instead we use different approach).

INNER JOIN:

Inner Join clause in SQL Server creates a new table (not physical) by combining rows that have matching values in two or more tables. This join is based on a logical relationship (or a common field) between the tables and is used to retrieve data that appears in both tables.

PROB:

mysql> select * from employee

inner join dept on employee.dept_id=dept.dept_id;

```
+--------+--------+---------+---------+---------+-----------+
| emp_id | f_name | l_name  | dept_id | dept_id | dept_name |
+--------+--------+---------+---------+---------+-----------+
|      1 | jhon   | doe     |      10 |      10 | HR        |
|      2 | jane   | smith   |      20 |      20 | Sales     |
|      3 | mike   | johnson |      30 |      30 | IT        |
|      4 | emily  | davis   |      10 |      10 | HR        |
+--------+--------+---------+---------+---------+-----------+
```

LEFT OUTER JOIN:

The `LEFT OUTER JOIN` keyword returns all records from the left table (table1), and the matching records from the right table (table2). The result is NULL records from the right side, if there is no match.

mysql> select * from employee

left outer join dept on employee.dept_id=dept.dept_id;

```
+--------+--------+---------+---------+---------+-----------+
| emp_id | f_name | l_name  | dept_id | dept_id | dept_name |
+--------+--------+---------+---------+---------+-----------+
|    1 | jhon   | doe     |    10 |    10 | HR        |
|    2 | jane   | smith   |    20 |    20 | Sales     |
|    3 | mike   | johnson |    30 |    30 | IT        |
|    4 | emily  | davis   |    10 |    10 | HR        |
+--------+--------+---------+---------+---------+-----------+
```

RIGHT OUTER JOIN:

The Right Join is used to joins two or more tables and returns all rows from the right-hand table, and only those results from the other table that fulfilled the join condition. If it finds unmatched records from the left side table, it returns Null value.

mysql> select * from employee

right outer join dept on employee.dept_id=dept.dept_id;

```
+--------+--------+---------+---------+---------+-----------+
| emp_id | f_name | l_name  | dept_id | dept_id | dept_name |
+--------+--------+---------+---------+---------+-----------+
|    1 | jhon   | doe     |    10 |    10 | HR        |
|    4 | emily  | davis   |    10 |    10 | HR        |
```

```
|     2 | jane   | smith   |     20 |     20 | Sales     |
|     3 | mike   | johnson |     30 |     30 | IT        |
|  NULL | NULL   | NULL    |   NULL |     40 | Marketing |
+--------+--------+---------+---------+---------+-----------+
```

FULL OUTER JOIN:

The **FULL OUTER JOIN in MySQL** returns all rows of both tables involved in the **JOIN operation** in the result set. If there is no match for a particular row based on the specified condition, the result will include **NULL** values for columns from the table that do not have a match.

NOTE: *MySQL does not explicitly support a FULL OUTER JOIN. Instead, we can achieve it by combining a **LEFT JOIN**, a **RIGHT JOIN**, and a **UNION operator**. You can use FULL OUTER JOIN in SQL using FULL OUTER JOIN keyword.*

Mysql>select * from employee

LEFT JOIN dept on employee.dept_id=dept.dept_id **UNION**

select * from employee **RIGHT JOIN** dept on employee.dept_id=dept.dept_id;

```
+--------+--------+---------+---------+---------+-----------+
| emp_id | f_name | l_name  | dept_id | dept_id | dept_name |
+--------+--------+---------+---------+---------+-----------+
|     1 | jhon   | doe     |     10 |     10 | HR        |
|     2 | jane   | smith   |     20 |     20 | Sales     |
|     3 | mike   | johnson |     30 |     30 | IT        |
|     4 | emily  | davis   |     10 |     10 | HR        |
|  NULL | NULL   | NULL    |   NULL |     40 | Marketing |
+--------+--------+---------+---------+---------+-----------+
```

2>Identifying Duplicate row values using mysql queries;

mysql> select f_name,count(*) from empl

group by f_name

having count(*)>1;

```
+--------+----------+
| f_name | count(*) |
+--------+----------+
| John   |        2 |
+--------+----------+
```

mysql> select email,count(*) from empl

group by email

having count(*)>1;

```
+----------------------+----------+
| email                | count(*) |
+----------------------+----------+
| jhon.doe@example.com |        2 |
+----------------------+----------+
```

mysql> select f_name,l_name,count(*) from empl

group by f_name,l_name

having count(*)>1;

```
+--------+--------+----------+
| f_name | l_name | count(*) |
+--------+--------+----------+
| John   | Doe    |        2 |
+--------+--------+----------+
```

mysql> select f_name,email,count(*) from empl

group by f_name,email

having count(*)>1;

```
+--------+--------------------+----------+
| f_name | email              | count(*) |
+--------+--------------------+----------+
| John   | jhon.doe@example.com |       2 |
+--------+--------------------+----------+
```