**Government College of Engineering, Jalgaon**
**(An Autonomous Institute of Government of Maharashtra)**

**Name :** **Semester :** V **PRN :**
**Class :** T. Y. B.Tech Computer **Academic Year :** 2024-25 **Subject :** CO307U DBMS Lab
**Course Teacher : Mr. Vinit Kakde**
**Date of Performance :** **Date of Completion :**

# Practical no. 7

**Aim:** Write SQL queries to implement joins.

SQL joins are used to query data from two or more tables, based on a relationship between certain columns in these tables.

1. INNER JOIN
2. LEFT JOIN
3. RIGHT JOIN
4. FULL JOIN

## Theory:

SQL JOIN

As the name shows, JOIN means to combine something. In case of SQL, JOIN means "to combine two or more tables".

In SQL, JOIN clause is used to combine the records from two or more tables in a database

1. INNER JOIN

In SQL, INNER JOIN selects records that have matching values in both tables as long as the condition is satisfied. It returns the combination of all rows from both the tables where the condition satisfies.

2. LEFT JOIN

The SQL left join returns all the values from left table and the matching values from the right table. If there is no matching join value, it will return NULL.

3. RIGHT JOIN

In SQL, RIGHT JOIN returns all the values from the values from the rows of right table and the matched values from the left table. If there is no matching in both tables, it will return NULL.

4. FULL JOIN

In SQL, FULL JOIN is the result of a combination of both left and right outer join. Join tables have all the records from both tables. It puts NULL on the place of matches not found.

**Queries and Output:**

1) Table1- orders

**Syntax:**

Create table table_name(column1 datatype, column2 datatype,….. column-N datatype);

**Query:**

create table orders(o_id int(5), orderno int(5),p_id int(3));

```
mysql> desc orders;
+---------+------+------+-----+---------+-------+
| Field   | Type | Null | Key | Default | Extra |
+---------+------+------+-----+---------+-------+
| o_id    | int  | YES  |     | NULL    |       |
| orderno | int  | YES  |     | NULL    |       |
| p_id    | int  | YES  |     | NULL    |       |
+---------+------+------+-----+---------+-------+
3 rows in set (0.90 sec)
```

2) Inserting values into orders

**Syntax:**

insert into orders values(&o_id,&orderno,&p_id);

**Query:**

insert into orders values(1,77895,3);
insert into orders values(2,44678,3);
insert into orders values(3,22456,1);
insert into orders values(4,24562,1);
insert into orders values(5,34764,15);

```
mysql> select * from orders;
+-------+----------+-------+
| o_id  | orderno  | p_id  |
+-------+----------+-------+
|     1 |    77895 |     3 |
|     2 |    44678 |     3 |
|     3 |    22456 |     1 |
|     4 |    24562 |     1 |
|     5 |    34764 |    15 |
+-------+----------+-------+
5 rows in set (0.00 sec)
```

3) Table-2: persons

**Syntax:**

Create table table_name(column1 datatype, column2 datatype,….. column-N datatype);

**Query:**

Create table persons(p_id int(5), lastname varchar(10),firstname varchar(15),address varchar(20), city varchar(10));

```
mysql> desc persons;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| p_id      | int         | YES  |     | NULL    |       |
| lastname  | varchar(10) | YES  |     | NULL    |       |
| firstname | varchar(15) | YES  |     | NULL    |       |
| address   | varchar(20) | YES  |     | NULL    |       |
| city      | varchar(10) | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```

4) Insert into persons tables

**Syntax:**

Insert into persons values(&p_id,'&lastname','&firstname','&address','&city');

**Query:**

insert into persons values(1,'Hansen','ola','timoteivn 10','sadnes');
insert into persons values(2,'Svendson','tove','borgn 23','sadnes');
insert into persons values(3,'Pettersen','kari','storgt 20','stavanger');

```
mysql> select * from persons;
+------+-----------+-----------+--------------+-----------+
| p_id | lastname  | firstname | address      | city      |
+------+-----------+-----------+--------------+-----------+
|    1 | Hansen    | ola       | timoteivn 10 | sadnes    |
|    2 | Svendson  | tove      | borgn 23     | sadnes    |
|    3 | Pettersen | kari      | storgt 20    | stavanger |
+------+-----------+-----------+--------------+-----------+
3 rows in set (0.00 sec)
```

5) Left Join syntax

**Syntax:**

Select column_name(s) from table_name1 left join table_name2 on table_name1.column_name=table_name2.column

**Query:**

select persons.lastname,persons.firstname,orders.orderno from persons left join orders on persons.p_id=orders.p_id order by persons.lastname;

```
+-----------+-----------+---------+
| lastname  | firstname | orderno |
+-----------+-----------+---------+
| Hansen    | ola       |   24562 |
| Hansen    | ola       |   22456 |
| Pettersen | kari      |   44678 |
| Pettersen | kari      |   77895 |
| Svendson  | tove      |    NULL |
+-----------+-----------+---------+
5 rows in set (0.04 sec)
```

6) Full outer join

**Syntax:**

Select * from table1 full join table2;

**Query:**

select * from persons full join orders;

```
+------+----------+----------+-------------+----------+------+---------+------+
| p_id | lastname | firstname | address     | city     | o_id | orderno | p_id |
+------+----------+----------+-------------+----------+------+---------+------+
|    3 | Pettersen | kari     | storgt 20   | stavanger |    1 |   77895 |    3 |
|    2 | Svendson | tove     | borgn 23    | sadnes   |    1 |   77895 |    3 |
|    1 | Hansen   | ola      | timoteivn 10 | sadnes  |    1 |   77895 |    3 |
|    3 | Pettersen | kari     | storgt 20   | stavanger |    2 |   44678 |    3 |
|    2 | Svendson | tove     | borgn 23    | sadnes   |    2 |   44678 |    3 |
|    1 | Hansen   | ola      | timoteivn 10 | sadnes  |    2 |   44678 |    3 |
|    3 | Pettersen | kari     | storgt 20   | stavanger |    3 |   22456 |    1 |
|    2 | Svendson | tove     | borgn 23    | sadnes   |    3 |   22456 |    1 |
|    1 | Hansen   | ola      | timoteivn 10 | sadnes  |    3 |   22456 |    1 |
|    3 | Pettersen | kari     | storgt 20   | stavanger |    4 |   24562 |    1 |
|    2 | Svendson | tove     | borgn 23    | sadnes   |    4 |   24562 |    1 |
|    1 | Hansen   | ola      | timoteivn 10 | sadnes  |    4 |   24562 |    1 |
|    3 | Pettersen | kari     | storgt 20   | stavanger |    5 |   34764 |   15 |
|    2 | Svendson | tove     | borgn 23    | sadnes   |    5 |   34764 |   15 |
|    1 | Hansen   | ola      | timoteivn 10 | sadnes  |    5 |   34764 |   15 |
+------+----------+----------+-------------+----------+------+---------+------+
15 rows in set, 1 warning (0.00 sec)
```

7) Right outer join

**Syntax:**

Select column_name(s) from table_name1 left join table_name2 on table_name1.column_name=table_name2.column

**Query:**

select persons.lastname,persons.firstname,orders.orderno from persons right outer join orders on persons.p_id=orders.p_id order by persons.lastname;

```
+-----------+-----------+---------+
| lastname  | firstname | orderno |
+-----------+-----------+---------+
| NULL      | NULL      |   34764 |
| Hansen    | ola       |   22456 |
| Hansen    | ola       |   24562 |
| Pettersen | kari      |   77895 |
| Pettersen | kari      |   44678 |
+-----------+-----------+---------+
5 rows in set (0.00 sec)
```

8) Inner join

**Syntax:**

Select column_name(s) from table_name1 left join table_name2 on table_name1.column_name=table_name2.column

**Query:**

select persons.lastname,persons.firstname,orders.orderno from persons inner join orders on persons.p_id=orders.p_id order by persons.lastname;

```
+-----------+-----------+---------+
| lastname  | firstname | orderno |
+-----------+-----------+---------+
| Hansen    | ola       |   22456 |
| Hansen    | ola       |   24562 |
| Pettersen | kari      |   77895 |
| Pettersen | kari      |   44678 |
+-----------+-----------+---------+
4 rows in set (0.00 sec)
```

9) Natural join

**Syntax:**

Select * from table1 full join table2;

**Query:**

select * from persons natural join orders;

```
+------+-----------+-----------+--------------+-----------+------+---------+
| p_id | lastname  | firstname | address      | city      | o_id | orderno |
+------+-----------+-----------+--------------+-----------+------+---------+
|    3 | Pettersen | kari      | storgt 20    | stavanger |    1 |   77895 |
|    3 | Pettersen | kari      | storgt 20    | stavanger |    2 |   44678 |
|    1 | Hansen    | ola       | timoteivn 10 | sadnes    |    3 |   22456 |
|    1 | Hansen    | ola       | timoteivn 10 | sadnes    |    4 |   24562 |
+------+-----------+-----------+--------------+-----------+------+---------+
4 rows in set (0.00 sec)
```

**Conclusion:**

In the field of Database Management Systems, Join is a very important tool for retrieving data from multiple tables simultaneously.

**Question and answers**

**1. What is a join?**

A join is a SQ clause used to combine and retrieve records from two or multiple tables. SQL tables can be joined based on the relationship between the columns of those tables. Check out our **SQL joins** tutorial to know all the details about them.

**2. What are the main types of joins?**

There are six main types of joins:

- INNER JOIN
- LEFT JOIN

- RIGHT JOIN
- FULL JOIN
- SELF JOIN
- CROSS JOIN

**3. What is the difference between a LEFT JOIN and a RIGHT JOIN?**

The LEFT JOIN includes all records from the left side and matched rows from the right table, whereas the RIGHT JOIN returns all rows from the right side and unmatched rows from the left table. Essentially, both joins will throw the same result if we exchange the table order, provided that there are only two tables involved.

**4. Why are joins important in SQL management?**

SQL joins are crucial in SQL management for multiple reasons, including:

- SQL JOINS are key methods to integrate multiple tables so they are easy to read and
- They provide an efficient and accessible way to access and combine information in your database.
- Using JOINS can reduce data usage and storage on the database.

**5. What is an OUTER JOIN?**

Outer joins are joins that return matched values and unmatched values from either or both tables. LEFT JOIN, RIGHT JOIN, AND FULL JOIN are considered outer joins.

**Name & Sign of Course Teacher**

Mr. Vinit Kakde