Experiment No: 4

Aim:

To study joins in queries, Cartesian product, Inner Join, Natural Join , Outer join

Theory:

Cartesian Product:

The *Cartesian product* operation combines tuples from two relations, but unlike the join operation, its result contains *all* pairs of tuples from the two relations, regardless of whether their attribute values match.

Joins:

The join operation allows the combining of two relations by merging pairs of tuples, one from each relation, into a single tuple. There are a number of different ways to join relations.

Inner join /Natural Join/ Equi Join:

The INNER JOIN keyword selects all rows from both tables as long as there is a match between the columns in both tables.

SQL INNER JOIN Syntax

```
SELECT column_name(s)
FROM table1
INNER JOIN table2
ON table1.column_name=table2.column_name;
```

or:

```
SELECT column_name(s)
FROM table1
JOIN table2
ON table1.column_name=table2.column_name;
```

Outer Join:

They are of three types:

1. Left join:

The LEFT JOIN keyword returns all rows from the left table (table1), with the matching rows in the right table (table2). The result is NULL in the right side when there is no match.

SQL LEFT JOIN Syntax

```
SELECT column_name(s)
FROM table1
LEFT JOIN table2
ON table1.column_name=table2.column_name;
```

or:

```
SELECT column_name(s)
FROM table1
LEFT OUTER JOIN table2
ON table1.column_name=table2.column_name;
```

2. Right join:

The RIGHT JOIN keyword returns all rows from the right table (table2), with the matching rows in the left table (table1). The result is NULL in the left side when there is no match.

SQL RIGHT JOIN Syntax

```
SELECT column_name(s)
FROM table1
RIGHT JOIN table2
ON table1.column_name=table2.column_name;
```

or:

```
SELECT column_name(s)
FROM table1
RIGHT OUTER JOIN table2
ON table1.column_name=table2.column_name;
```

3. Full join:

The FULL OUTER JOIN keyword returns all rows from the left table (table1) and from the right table (table2).

The FULL OUTER JOIN keyword combines the result of both LEFT and RIGHT joins.

SQL FULL OUTER JOIN Syntax

```
SELECT column_name(s)
FROM table1
FULL OUTER JOIN table2
ON table1.column_name=table2.column_name;
```

MySQL does not support full join and hence we need to use union set operation over left join and right join to get equivalent output.

Code:

```
t151070031@t151070031-VirtualBox:~$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 5
Server version: 5.7.17-0ubuntu0.16.04.1 (Ubuntu)

Copyright (c) 2000, 2016, Oracle and/or its affiliates. All rights
reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input
statement.

mysql> use exp2
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| exp2               |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
5 rows in set (0.00 sec)

mysql> use exp2
Database changed
mysql> create table CUSTOMERS(ID INT,NAME VARCHAR(50),AGE
INT,ADDRESS VARCHAR(50),SALARY NUMERIC(7,2));
Query OK, 0 rows affected (0.28 sec)

mysql> INSERT INTO CUSTOMERS VALUES
(1,"Ramesh",32,"Ahmedabad",2000),(2,"Khilan",25,"Delhi",1500),(3,"Ka
ushik",23,"Kota",2000),(4,"Chaitali",25,"Mumbai",6500),(5,"Hardik",2
7,"Bhopal",8500),(6,"Komal",22,"MP",4500),(7,"Muffy",24,"Indore",100
00)
    -> ;
Query OK, 7 rows affected (0.24 sec)
Records: 7  Duplicates: 0  Warnings: 0
```

```
mysql> select * from CUSTOMERS;
+------+----------+------+-----------+----------+
| ID   | NAME     | AGE  | ADDRESS   | SALARY   |
+------+----------+------+-----------+----------+
|    1 | Ramesh   |   32 | Ahmedabad |  2000.00 |
|    2 | Khilan   |   25 | Delhi     |  1500.00 |
|    3 | Kaushik  |   23 | Kota      |  2000.00 |
|    4 | Chaitali |   25 | Mumbai    |  6500.00 |
|    5 | Hardik   |   27 | Bhopal    |  8500.00 |
|    6 | Komal    |   22 | MP        |  4500.00 |
|    7 | Muffy    |   24 | Indore    | 10000.00 |
+------+----------+------+-----------+----------+
7 rows in set (0.01 sec)

mysql> CREATE TABLE ORDERS(OID INT,DATE DATE,CUSTOMER_ID INT,AMOUNT
INT)
    -> ;
Query OK, 0 rows affected (0.31 sec)

mysql> INSERT INTO ORDERS VALUES(102,"2009-10-
08",3,3000),(100,"2009-10-08",3,1500),(101,"2009-11-
20",2,1560),(103,"2008-05-20",2,2060);
Query OK, 4 rows affected (0.16 sec)
Records: 4  Duplicates: 0  Warnings: 0

mysql> select * from ORDERS;
+------+------------+-------------+--------+
| OID  | DATE       | CUSTOMER_ID | AMOUNT |
+------+------------+-------------+--------+
|  102 | 2009-10-08 |           3 |   3000 |
|  100 | 2009-10-08 |           3 |   1500 |
|  101 | 2009-11-20 |           2 |   1560 |
|  103 | 2008-05-20 |           2 |   2060 |
+------+------------+-------------+--------+
4 rows in set (0.00 sec)

mysql> SELECT ID,NAME,DATE,AMOUNT FROM CUSTOMERS,ORDERS;
+------+----------+------------+--------+
| ID   | NAME     | DATE       | AMOUNT |
+------+----------+------------+--------+
|    1 | Ramesh   | 2009-10-08 |   3000 |
|    1 | Ramesh   | 2009-10-08 |   1500 |
|    1 | Ramesh   | 2009-11-20 |   1560 |
|    1 | Ramesh   | 2008-05-20 |   2060 |
|    2 | Khilan   | 2009-10-08 |   3000 |
|    2 | Khilan   | 2009-10-08 |   1500 |
|    2 | Khilan   | 2009-11-20 |   1560 |
|    2 | Khilan   | 2008-05-20 |   2060 |
|    3 | Kaushik  | 2009-10-08 |   3000 |
|    3 | Kaushik  | 2009-10-08 |   1500 |
|    3 | Kaushik  | 2009-11-20 |   1560 |
|    3 | Kaushik  | 2008-05-20 |   2060 |
|    4 | Chaitali | 2009-10-08 |   3000 |
|    4 | Chaitali | 2009-10-08 |   1500 |
|    4 | Chaitali | 2009-11-20 |   1560 |
|    4 | Chaitali | 2008-05-20 |   2060 |
```

```
|    5 | Hardik   | 2009-10-08 |   3000 |
|    5 | Hardik   | 2009-10-08 |   1500 |
|    5 | Hardik   | 2009-11-20 |   1560 |
|    5 | Hardik   | 2008-05-20 |   2060 |
|    6 | Komal    | 2009-10-08 |   3000 |
|    6 | Komal    | 2009-10-08 |   1500 |
|    6 | Komal    | 2009-11-20 |   1560 |
|    6 | Komal    | 2008-05-20 |   2060 |
|    7 | Muffy    | 2009-10-08 |   3000 |
|    7 | Muffy    | 2009-10-08 |   1500 |
|    7 | Muffy    | 2009-11-20 |   1560 |
|    7 | Muffy    | 2008-05-20 |   2060 |
+------+----------+------------+--------+
28 rows in set (0.01 sec)

mysql> SELECT ID,NAME,DATE,AMOUNT FROM CUSTOMERS INNER JOIN ORDERS
ON CUSTOMERS.ID=ORDERS.CUSTOMER_ID;
+------+---------+------------+--------+
| ID   | NAME    | DATE       | AMOUNT |
+------+---------+------------+--------+
|    2 | Khilan  | 2009-11-20 |   1560 |
|    2 | Khilan  | 2008-05-20 |   2060 |
|    3 | Kaushik | 2009-10-08 |   3000 |
|    3 | Kaushik | 2009-10-08 |   1500 |
+------+---------+------------+--------+
4 rows in set (0.08 sec)

mysql> UPDATE ORDERS SET CUSTOMER_ID=4 WHERE OID=103;
Query OK, 1 row affected (0.16 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> SELECT ID,NAME,DATE,AMOUNT FROM CUSTOMERS,ORDERS;
+------+----------+------------+--------+
| ID   | NAME     | DATE       | AMOUNT |
+------+----------+------------+--------+
|    1 | Ramesh   | 2009-10-08 |   3000 |
|    1 | Ramesh   | 2009-10-08 |   1500 |
|    1 | Ramesh   | 2009-11-20 |   1560 |
|    1 | Ramesh   | 2008-05-20 |   2060 |
|    2 | Khilan   | 2009-10-08 |   3000 |
|    2 | Khilan   | 2009-10-08 |   1500 |
|    2 | Khilan   | 2009-11-20 |   1560 |
|    2 | Khilan   | 2008-05-20 |   2060 |
|    3 | Kaushik  | 2009-10-08 |   3000 |
|    3 | Kaushik  | 2009-10-08 |   1500 |
|    3 | Kaushik  | 2009-11-20 |   1560 |
|    3 | Kaushik  | 2008-05-20 |   2060 |
|    4 | Chaitali | 2009-10-08 |   3000 |
|    4 | Chaitali | 2009-10-08 |   1500 |
|    4 | Chaitali | 2009-11-20 |   1560 |
|    4 | Chaitali | 2008-05-20 |   2060 |
|    5 | Hardik   | 2009-10-08 |   3000 |
|    5 | Hardik   | 2009-10-08 |   1500 |
|    5 | Hardik   | 2009-11-20 |   1560 |
|    5 | Hardik   | 2008-05-20 |   2060 |
|    6 | Komal    | 2009-10-08 |   3000 |
```

```
|    6 | Komal    | 2009-10-08 |   1500 |
|    6 | Komal    | 2009-11-20 |   1560 |
|    6 | Komal    | 2008-05-20 |   2060 |
|    7 | Muffy    | 2009-10-08 |   3000 |
|    7 | Muffy    | 2009-10-08 |   1500 |
|    7 | Muffy    | 2009-11-20 |   1560 |
|    7 | Muffy    | 2008-05-20 |   2060 |
+------+----------+------------+--------+
28 rows in set (0.01 sec)

mysql> SELECT ID,NAME,DATE,AMOUNT FROM CUSTOMERS INNER JOIN ORDERS
ON CUSTOMERS.ID=ORDERS.CUSTOMER_ID;
+------+----------+------------+--------+
| ID   | NAME     | DATE       | AMOUNT |
+------+----------+------------+--------+
|    2 | Khilan   | 2009-11-20 |   1560 |
|    3 | Kaushik  | 2009-10-08 |   3000 |
|    3 | Kaushik  | 2009-10-08 |   1500 |
|    4 | Chaitali | 2008-05-20 |   2060 |
+------+----------+------------+--------+
4 rows in set (0.00 sec)

mysql> SELECT ID,NAME,DATE,AMOUNT FROM CUSTOMERS LEFT JOIN ORDERS ON
CUSTOMERS.ID=ORDERS.CUSTOMER_ID;
+------+----------+------------+--------+
| ID   | NAME     | DATE       | AMOUNT |
+------+----------+------------+--------+
|    3 | Kaushik  | 2009-10-08 |   3000 |
|    3 | Kaushik  | 2009-10-08 |   1500 |
|    2 | Khilan   | 2009-11-20 |   1560 |
|    4 | Chaitali | 2008-05-20 |   2060 |
|    1 | Ramesh   | NULL       |   NULL |
|    5 | Hardik   | NULL       |   NULL |
|    6 | Komal    | NULL       |   NULL |
|    7 | Muffy    | NULL       |   NULL |
+------+----------+------------+--------+
8 rows in set (0.00 sec)

mysql> SELECT ID,NAME,DATE,AMOUNT FROM CUSTOMERS RIGHT JOIN ORDERS
ON CUSTOMERS.ID=ORDERS.CUSTOMER_ID;
+------+----------+------------+--------+
| ID   | NAME     | DATE       | AMOUNT |
+------+----------+------------+--------+
|    2 | Khilan   | 2009-11-20 |   1560 |
|    3 | Kaushik  | 2009-10-08 |   3000 |
|    3 | Kaushik  | 2009-10-08 |   1500 |
|    4 | Chaitali | 2008-05-20 |   2060 |
+------+----------+------------+--------+
4 rows in set (0.00 sec)



mysql> SELECT ID,NAME,DATE,AMOUNT FROM CUSTOMERS LEFT JOIN ORDERS ON
CUSTOMERS.ID=ORDERS.CUSTOMER_ID UNION  SELECT ID,NAME,DATE,AMOUNT
FROM CUSTOMERS RIGHT JOIN ORDERS ON CUSTOMERS.ID=ORDERS.CUSTOMER_ID;
```

```
+------+----------+------------+--------+
| ID   | NAME     | DATE       | AMOUNT |
+------+----------+------------+--------+
|    3 | Kaushik  | 2009-10-08 |   3000 |
|    3 | Kaushik  | 2009-10-08 |   1500 |
|    2 | Khilan   | 2009-11-20 |   1560 |
|    4 | Chaitali | 2008-05-20 |   2060 |
|    1 | Ramesh   | NULL       |   NULL |
|    5 | Hardik   | NULL       |   NULL |
|    6 | Komal    | NULL       |   NULL |
|    7 | Muffy    | NULL       |   NULL |
+------+----------+------------+--------+
8 rows in set (0.01 sec)

mysql>  SELECT ID,NAME,DATE,AMOUNT FROM CUSTOMERS RIGHT JOIN ORDERS
ON CUSTOMERS.ID=ORDERS.CUSTOMER_ID UNION  SELECT ID,NAME,DATE,AMOUNT
FROM CUSTOMERS LEFT JOIN ORDERS ON CUSTOMERS.ID=ORDERS.CUSTOMER_ID;
+------+----------+------------+--------+
| ID   | NAME     | DATE       | AMOUNT |
+------+----------+------------+--------+
|    2 | Khilan   | 2009-11-20 |   1560 |
|    3 | Kaushik  | 2009-10-08 |   3000 |
|    3 | Kaushik  | 2009-10-08 |   1500 |
|    4 | Chaitali | 2008-05-20 |   2060 |
|    1 | Ramesh   | NULL       |   NULL |
|    5 | Hardik   | NULL       |   NULL |
|    6 | Komal    | NULL       |   NULL |
|    7 | Muffy    | NULL       |   NULL |
+------+----------+------------+--------+
8 rows in set (0.01 sec)

mysql> EXIT
Bye
```

Conclusion:

Thus, the various join operations were studied and executed to get desired output. Errors which creeped up during execution were analysed and solution to them were found .