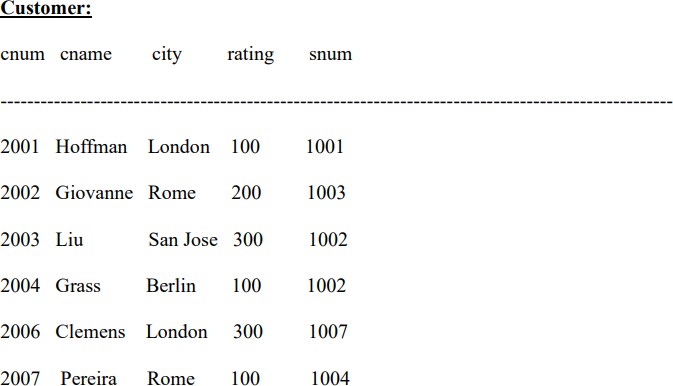
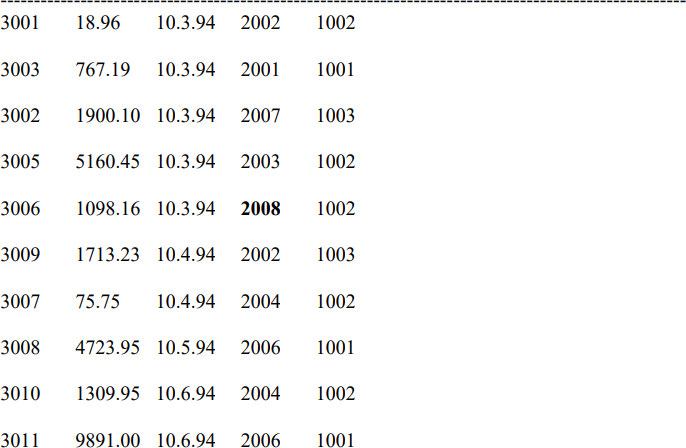
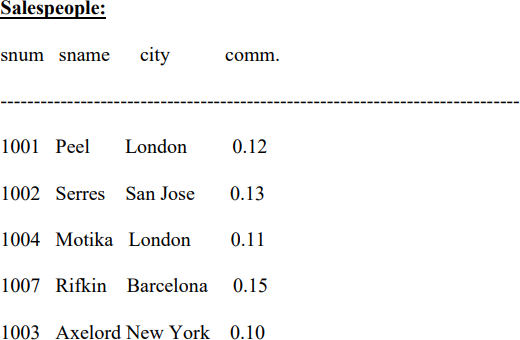
# PRACTICAL-9

**AIM**- To study and execute various JOIN commands to perform data retrieval and manipulation from Salespeople, Customer, and Order tables based on specific requirements.

## DESCRIPTION:-



**QUERY:**

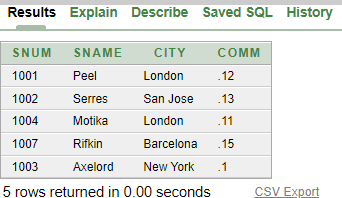
CREATE TABLE Salespeople (

snum NUMBER(4) PRIMARY KEY, sname VARCHAR2(20) NOT NULL, city VARCHAR2(15),

comm NUMBER(5,2));

INSERT INTO Salespeople VALUES (1001, 'Peel', 'London', 0.12); INSERT INTO Salespeople VALUES (1002, 'Serres', 'San Jose', 0.13); INSERT INTO Salespeople VALUES (1004, 'Motika', 'London', 0.11); INSERT INTO Salespeople VALUES (1007, 'Rifkin', 'Barcelona', 0.15); INSERT INTO Salespeople VALUES (1003, 'Axelord', 'New York', 0.10);

# Output:



## QUERY:

CREATE TABLE Customer (

cnum NUMBER(4) PRIMARY KEY, cname VARCHAR2(20) NOT NULL, city VARCHAR2(20),

rating NUMBER(3), snum NUMBER(4),

FOREIGN KEY (snum) REFERENCES Salespeople(snum));

INSERT INTO Customer VALUES (2001, 'Hoffman', 'London', 100, 1001);

INSERT INTO Customer VALUES (2002, 'Giovanne', 'Rome', 200, 1003);

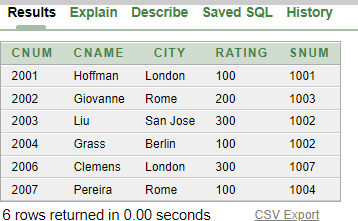
INSERT INTO Customer VALUES (2003, 'Liu', 'San Jose', 300, 1002);

INSERT INTO Customer VALUES (2004, 'Grass', 'Berlin', 100, 1002);

INSERT INTO Customer VALUES (2006, 'Clemens', 'London', 300, 1007);

INSERT INTO Customer VALUES (2007, 'Pereira', 'Rome', 100, 1004);

**Output:**



## QUERY:

CREATE TABLE "Order" (

onum NUMBER(4) PRIMARY KEY, amt NUMBER(7,2),

odate DATE,

cnum NUMBER(4), snum NUMBER(4),

FOREIGN KEY (cnum) REFERENCES Customer(cnum), FOREIGN KEY (snum) REFERENCES Salespeople(snum));

INSERT INTO "Order" VALUES (3001, 18.96, TO\_DATE('10.3.94', 'DD.MM.YY'), 2002, 1002);

INSERT INTO "Order" VALUES (3003, 767.19, TO\_DATE('10.3.94', 'DD.MM.YY'), 2001, 1001);

INSERT INTO "Order" VALUES (3002, 1900.10, TO\_DATE('10.3.94', 'DD.MM.YY'), 2007, 1003);

INSERT INTO "Order" VALUES (3005, 5160.45, TO\_DATE('10.3.94', 'DD.MM.YY'), 2003, 1002);

INSERT INTO "Order" VALUES (3006, 1098.16, TO\_DATE('10.3.94', 'DD.MM.YY'), 2008, 1002);

INSERT INTO "Order" VALUES (3009, 1713.23, TO\_DATE('10.4.94', 'DD.MM.YY'), 2002, 1003);

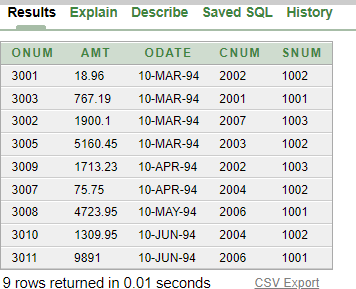
INSERT INTO "Order" VALUES (3007, 75.75, TO\_DATE('10.4.94', 'DD.MM.YY'), 2004, 1002);

INSERT INTO "Order" VALUES (3008, 4723.95, TO\_DATE('10.5.94', 'DD.MM.YY'), 2006, 1001);

INSERT INTO "Order" VALUES (3010, 1309.95, TO\_DATE('10.6.94', 'DD.MM.YY'), 2004, 1002);

INSERT INTO "Order" VALUES (3011, 9891.00, TO\_DATE('10.6.94', 'DD.MM.YY'), 2006, 1001);

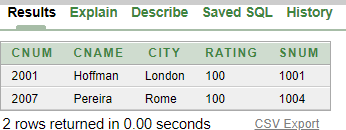
## OUTPUT:



### All customers serviced by Peel or Motika QUERY:

Select Distinct c.\* FROM Customer c JOIN Salespeople s ON c.snum = s.snum WHERE s.sname IN ('Peel', 'Motika');

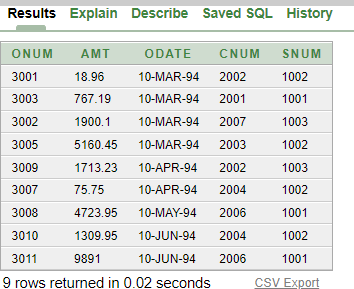
## OUTPUT:



### All orders except those with 0 or null value in the amt field QUERUY:

Select \* from "Order" where amt IS NOT NULL AND amt != 0;

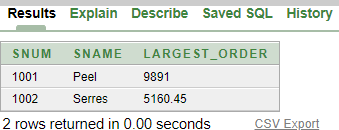
## OUTPUT:



### Largest order taken by each sales order value of more than 3000 QUERY:

Select s.snum, s.sname, MAX(o.amt) as largest\_order FROM Salespeople s JOIN "Order" o ON s.snum = o.snum WHERE o.amt > 3000 GROUP BY s.snum, s.sname;

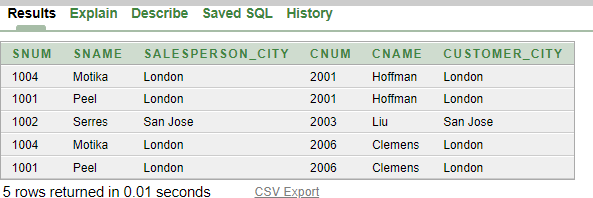
## OUTPUT:



### All combinations of salespeople and customers who belong to the same city QUERY:

Select s.snum, s.sname, s.city AS salesperson\_city, c.cnum, c.cname, c.city AS customer\_city FROM Salespeople s JOIN Customer c ON s.city = c.city;

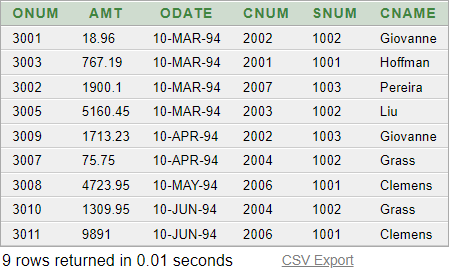
## OUTPUT:



### List each order with the name of the customer who placed the order QUERY:

Select o.\*, c.cname from "Order" o JOIN Customer c ON o.cnum = c.cnum;

## OUTPUT:

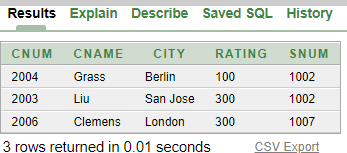


1. **Produce a listing of all the customers serviced by salespeople having a commission of more than 12%**

## QUERY:

Select distinct c.\* from Customer c JOIN Salespeople s ON c.snum = s.snum where s.comm > 0.12;

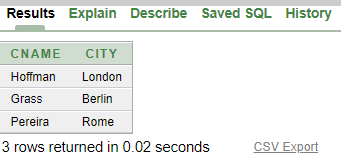
## OUTPUT:



### Produce names and cities of all customers with the same rating as Hoffman QUERY:

Select cname, city from Customer where rating = (select rating from Customer where cname = 'Hoffman');

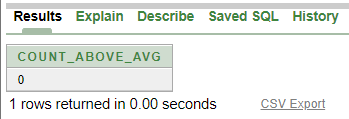
## OUTPUT:



### Count the customers with ratings above San Jose’s average QUERY:

Select COUNT(\*) as count\_above\_avg FROM Customer where rating > (SELECT AVG(rating) FROM Customer WHERE city = 'San Jose');

## OUTPUT:

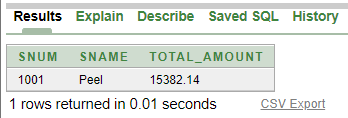


1. **Find the total amount in orders for each salesperson for whom this total is greater than the amount of the largest order in the order table**

## QUERY:

SELECT s.snum, s.sname, SUM(o.amt) as total\_amount FROM Salespeople s JOIN "Order" o ON s.snum = o.snum GROUP BY s.snum, s.sname HAVING SUM(o.amt) > (SELECT MAX(amt) FROM "Order");

## OUTPUT:

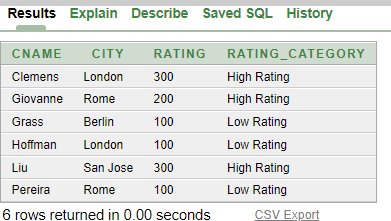


1. **Create a union of two queries that shows all customers' names, cities, and ratings. Those with a rating of 200 or greater will have the words ‘High Rating’ while others will have ‘Low Rating’.**

## QUERY:

Select cname, city, rating, 'High Rating' as rating\_category FROM Customer WHERE rating >= 200 UNION Select cname, city, rating, 'Low Rating' as rating\_category FROM Customer where rating < 200;

## OUTPUT:



**CONCLUSION:**

* + From this practical I’ve learnt that how can we use/apply concept of joins in a query form and to use different combinations of join to utilize/understand the concept.