Assignment – 11 Subqueries

1) Write a query that uses a subquery to obtain all orders for the customer named Cisneros. Assume you do not know his customer number (cnum).

```
mysql> select * from orders where snum=
    -> (select snum from customers where cname='Cisneros');
```

2) Write a query that produces the names and ratings of all customers who have above-average orders.

```
mysql> select distinct cname,rating from customers,orders where
orders.cnum=customers.cnum and orders.amt>( select avg(amt) from
orders);
```

3) Write a query that selects the total amount in orders for each salesperson for whom this total is greater than the amount of the largest order in the table.

```
mysql> select sum(amt) from orders group by snum having sum(amt)>
(select max(amt) from orders);
```

Assignment – 12
Using the operators IN, ANY, and ALL

1) Write a query that selects all customers whose ratings are equal to or greater than ANY of Serres'.

```
mysql> select cname,rating from customers where rating>= any (select
rating from customers,salespeople where
salespeople.snum=customers.snum and sname='serres');
```

2) Write a query using ANY or ALL that will find all salespeople who have no customers located in their city.

```
mysql> select * from salespeople where city != all(select customers.city
from customers where salespeople.snum=customers.snum);
```

3) Write a query that selects all orders for amounts greater than any for the customers in London.

```
mysql> select * from orders where amt> any(select orders.amt from orders
join customers on orders.cnum=customers.cnum where
customers.city='london');
```

4) Write the above query using MIN or MAX.

```
mysql> SELECT * FROM orders
    -> WHERE amt > (
    ->     SELECT MIN(o.amt)
    ->     FROM orders o
    ->     JOIN customers c ON o.cnum = c.cnum
    ->     WHERE c.city = 'London')
    -> AND amt < (
    ->     SELECT MAX(o.amt) FROM orders o
    ->     JOIN customers c ON o.cnum = c.cnum
    ->     WHERE c.city = 'London');
```

Assignment – 13 Using the UNION clause.

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

```
mysql> (select cname,city,rating, "Low Rating" as status from customers
where rating <200)
    -> union
    -> (select cname,city,rating,"High Rating" as status from customers
where rating >=200) ;
```

2) Write a command that produces the name and number of each salesperson and each customer with more than one current order. Put the results in alphabetical order.

```
mysql> SELECT c.cname AS name, c.cnum AS number
    -> FROM customers c
    -> JOIN orders o ON c.cnum = o.cnum
    -> GROUP BY c.cname, c.cnum
    -> HAVING COUNT(o.onum) > 1
    -> UNION
    -> SELECT s.sname AS name, s.snum AS number
    -> FROM salespeople s
    -> JOIN orders o ON s.snum = o.snum
    -> GROUP BY s.sname, s.snum
    -> HAVING COUNT(o.onum) > 1
    -> ORDER BY name;
```

3) Form a union of three queries. Have the first select the snums of all salespeople in San Jose; the second, the cnums of all customers in San Jose; and the third the onums of all orders on October 3. Retain duplicates between the last two queries but eliminate any redundancies between either of them and the first. (Note: in the sample tables as given, there would be no such redundancy. This is besides the point.)

```
SELECT snum FROM salespeople
WHERE city = 'San Jose'
UNION
SELECT cnum
FROM customers
WHERE city = 'San Jose'
UNION ALL
SELECT onum
FROM orders
WHERE odate = '1990-10-03';
```