PROJECT 2

*TEAM NAME: PACERS*

TEAM MEMBERS:

Megha Varshini M

Aluru Keerthana

Chekka V B Krishna Harshith Babu

Yogitha Prakash

Overview

Structured Query Language, abbreviated as SQL, is a domain-specific language used in programming and designed for managing data held in a relational database management system, or for stream processing in a relational data stream management system. We have performed queries , joins, sub queries and aggregate functions.

Problem Statement:

Create database, table as specified or required and find solutions for the given question/query. We need to create three tables according to the specified information.

1. **Querying SQL Databases**

* 1. Create the following table named “Employee” and insert the following tables based on above table write the following queries:

create table Employee (  
 empID int NOT NULL,

managerID int,

firstName varchar(50) NOT NULL,

lastName varchar(50) NOT NULL,

empTitle varchar(50) NOT NULL,

salary varchar(50) NOT NULL,

joiningData varchar(50) NOT NULL,

department varchar(50) NOT NULL

);

insert into Employee values (1, , ‘Monica’, ’Arora’, ‘Manager’, ‘100000’, ‘02/20/14 9:00’, ‘HR’);

insert into Employee values (2, 1, ‘Niharika’, ’Verma’, ‘Executive’, ‘80000’, ‘06/11/14 9:00’, ‘Admin’);

insert into Employee values (3, 1, 'Vishal', 'Singhal', 'Executive', '300000 ', '02/20/14 9:00', 'HR');

insert into Employee values (4, NULL, 'Amitabh', 'Singh', 'Manager', '500000 ', '02/20/14 9:00', 'Admin');

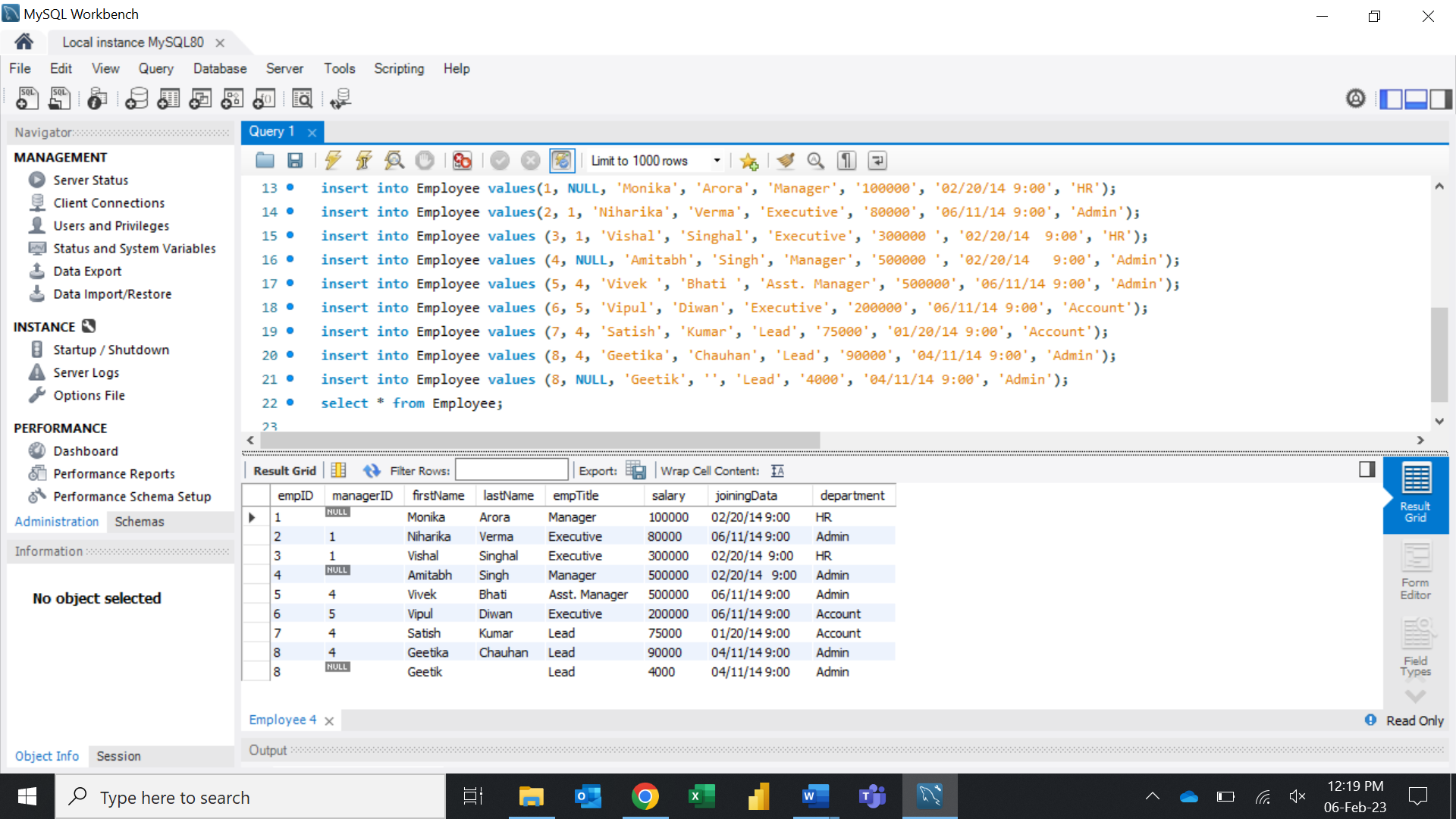
insert into Employee values (5, 4, 'Vivek ', 'Bhati ', 'Asst. Manager', '500000', '06/11/14 9:00', 'Admin');

insert into Employee values (6, 5, 'Vipul', 'Diwan', 'Executive', '200000', '06/11/14 9:00', 'Account');

insert into Employee values (7, 4, 'Satish', 'Kumar', 'Lead', '75000', '01/20/14 9:00', 'Account');

insert into Employee values (8, 4, 'Geetika', 'Chauhan', 'Lead', '90000', '04/11/14 9:00', 'Admin');

insert into Employee values (8, NULL, 'Geetik', '', 'Lead', '4000', '04/11/14 9:00', 'Admin');

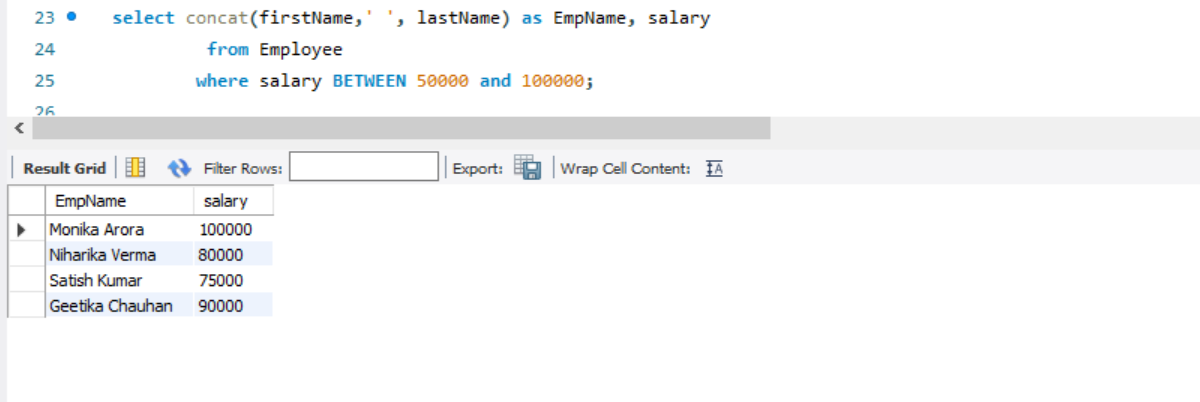


* 1. Write an SQL query to fetch EMP names between salary 50000 and 100000.

select concat(firstName, ‘ ‘, lastName) as EmpName, salary

from Employee

where salary BETWEEN 50000 and 100000;



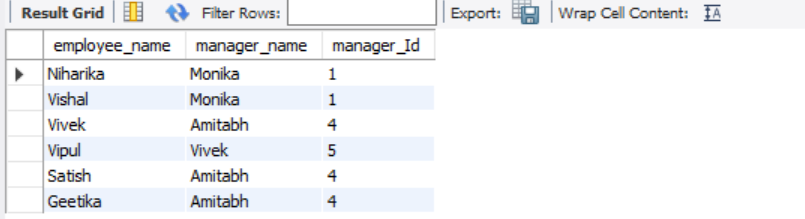
* 1. Get the Employ Name and their Manager Name.

select e1.empID, e1.firstName as EmpName, e1.managerID, e2.firstName as MgrName

from Employee e1

LEFT JOIN Employee e2

ON e1.managerID = e2.empID;



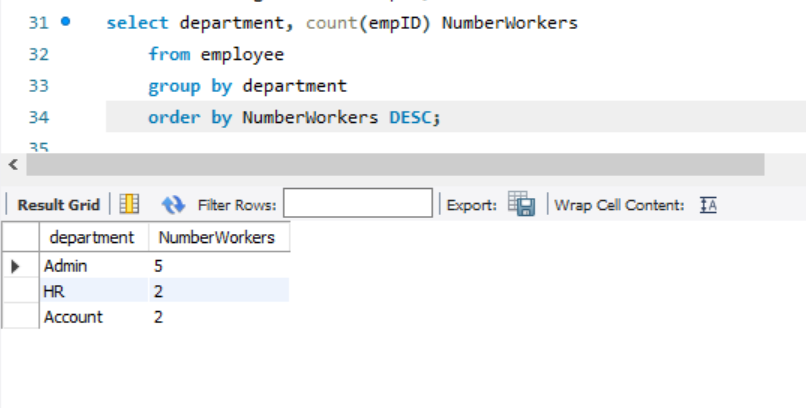
* 1. Write an SQL query to fetch the no. of EMP for each department in the descending order

select department, count(empID) NumberWorkers

from employee

group by department

order by NumberWorkers DESC;



* 1. Write an SQL query to determine the 5th highest salary without using TOP or limit

method.

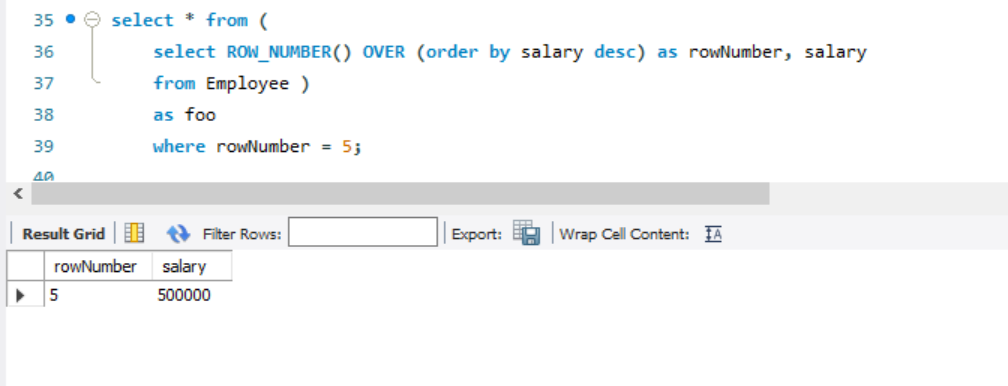
select \* from (

select ROW\_NUMBER() OVER (order by salary desc) as rowNumber, salary

from Employee )

as foo

where rowNumber = 5;



* 1. Write an SQL query to determine the nth (say n=5) highest salary from a table.

SELECT \* FROM( SELECT Salary, DENSE\_RANK() OVER (ORDER BY salary+0,salary DESC)r FROM Employee) e WHERE e.r=5;



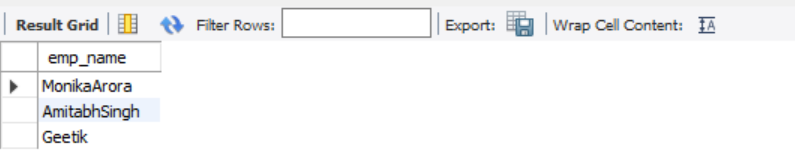
* 1. Write a SQL scripts to identify the duplicate records in the employee table in terms of EMP ID

select \* from (select \*,row\_number() over(partition by emp\_id order by emp\_id)r from employee)e where e.r <> 1;



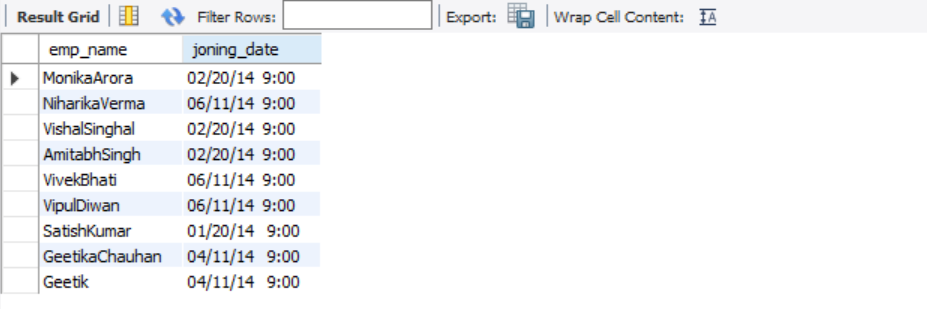
* 1. Write a SQL scripts to List the Employee whose manger is not specified.

select concat(first\_name,last\_name) as emp\_name from employee where manager\_id is NUll;



* 1. Write a SQL scripts to find the Number of Employees who joined in the year of 2014

select concat(first\_name,last\_name) as emp\_name, joning\_date from employee where JONING\_DATE like '\_\_\_\_\_/14%';



1. **Write statements Using joins and Subqueries.**

use project;

create table Customer(Cust\_id int primary key,Cust\_name Varchar(50),city varchar(20),grade int,Salesman\_id varchar(20));

insert into Customer values(3001,"Nick Rimando","New York",100,'5001');

insert into Customer values(3002,"Brad Davis","New York",200,'5001');

insert into Customer values(3003,"Graham Zusi","California",200,'5002');

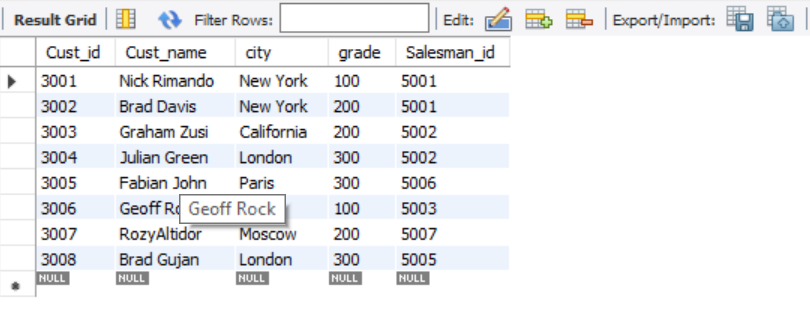
insert into Customer values(3004,"Julian Green ","London",300,'5002');

insert into Customer values(3005,"Fabian John","Paris",300,'5006');

insert into Customer values(3006,"Geoff Rock","Berlin",100,'5003');

insert into Customer values(3007,"RozyAltidor","Moscow",200,'5007');

insert into Customer values(3008,"Brad Gujan","London",300,'5005');



create table Orders(Order\_no int,Order\_date varchar(20),Order\_amt int,Cust\_Id int,Salesman\_id varchar(20),Commission varchar(20),FOREIGN KEY (Cust\_Id) REFERENCES Customer(Cust\_Id));

insert into orders values(1,'12/22/2005',160,3001,'5002','13%');

insert into orders values(9,'08/10/2005',190,3001,'5005','11%');

insert into orders values(2,'07/13/2005',500,3002,'5001','15%');

insert into orders values(4,'07/15/2005',420,3002,Null,'0');

insert into orders values(7,'12/22/2005',1000,3003,NUll,'0');

insert into orders values(5,'10/2/2006',820,3004,'5001','15%');

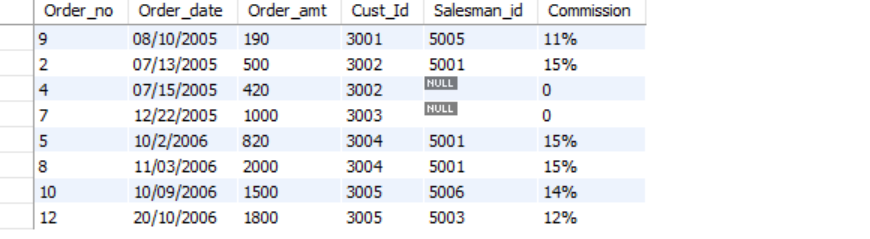
insert into orders values(8,'11/03/2006',2000,3004,'5001','15%');

insert into orders values(10,'10/09/2006',1500,3005,'5006','14%');

insert into orders values(12,'20/10/2006',1800,3005,'5003','12%');

insert into orders values(13,'20/10/2006',1800,4001,'5003','12%'); // This row wll not be inserted because of foregn key constrain.

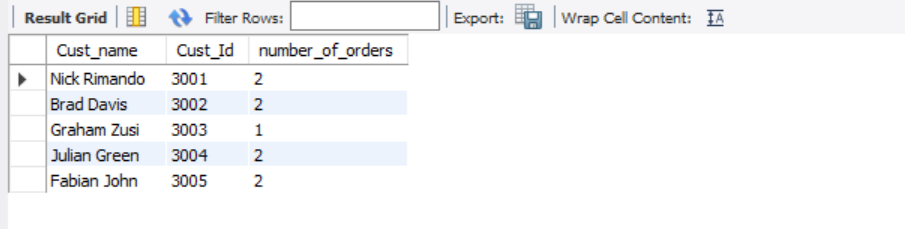
select \* from orders;



2.1 List the customer and no of order placed by customer;

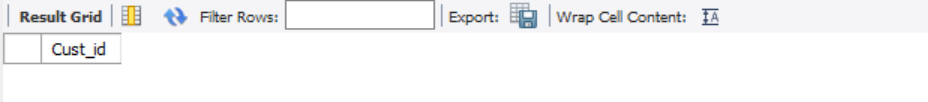
**Note**: The output to list all customer whether he placed no order or placed one or more orders

SELECT Cust\_name, c.Cust\_Id, COUNT(\*) AS number\_of\_orders FROM customer c , orders WHERE c.Cust\_Id = orders.Cust\_Id GROUP BY Cust\_Id ORDER BY c.Cust\_Id;



2.2 Find the order(s) placed by the customer who is not in the customer list.

select Cust\_id FROM orders WHERE Cust\_id NOT IN (SELECT Cust\_Id FROM customer);



2.3 Find out the customers who place the order in the month of July and August

select o.Cust\_Id,c.Cust\_name,o.order\_date from customer c join orders o on c.Cust\_Id = o.Cust\_Id having o.order\_date like '07%' or o.order\_date like '08%';

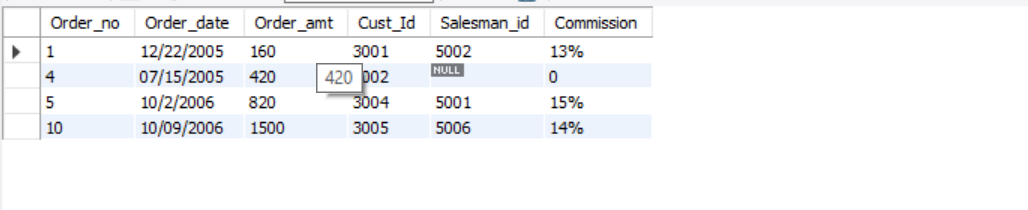


2.4 List the customer Name and his total order amount, which is less than the Avg order amount of individual orders.

select \* from orders o

where order\_amt < ( select avg(order\_amt) FROM orders b

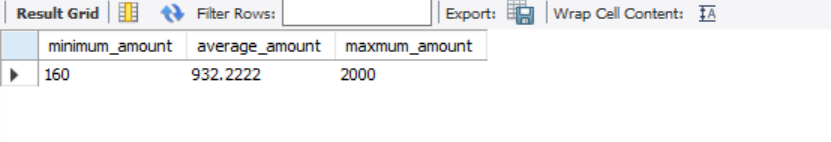
where b.cust\_id = o.cust\_id);



3. Aggregate functions:

* 1. Use the inbuilt functions and find the minimum, maximum and average amount from the orders table

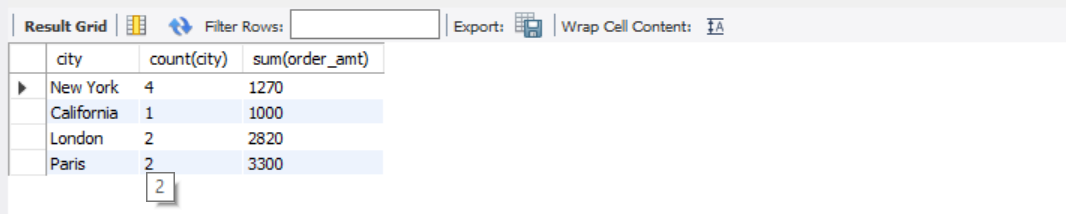
select min(order\_amt) as minimum\_amount,avg(order\_amt) as average\_amount,max(order\_amt) as maxmum\_amount from orders;



* 1. Find the “number of orders” and total Order Value(amount) from each customer city

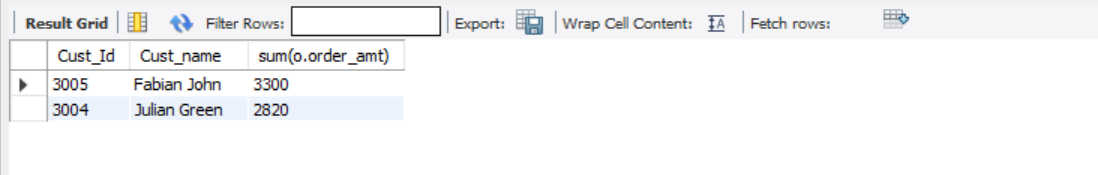
Expected Output

|  |  |  |
| --- | --- | --- |
| City | No of Orders | Order Value |
| New York | 4 | 1270 |
| California | 1 | 1000 |
| London | 2 |  |

select distinct city,count(city),sum(order\_amt) from customer c inner join orders o on c.Cust\_Id = o.Cust\_Id group by c.city;

* 1. Take the top 2 customers based on their total order amount

select distinct o.Cust\_Id,c.Cust\_name,sum(o.order\_amt) from orders o join customer c on o.Cust\_Id = c.Cust\_Id group by o.Cust\_Id order by (sum(o.order\_amt)) desc limit 2;



3.4 Take the top most order no based on highest Order Value(amount) from each customer city

Expected Output

|  |  |  |
| --- | --- | --- |
| City | Order\_No | Max Order amount |
| New York | 2 | 500 |
| California | 7 | 1000 |
| London | 8 |  |

select distinct f.city,max(f.Order\_amt) as max\_price from (select c.city,o.Order\_no ,o.Order\_amt from customer c inner join orders o on o.Cust\_Id = c.Cust\_Id) as f group by(f.city);



3.5 Total Order Amount for Year 2005 Vs 2006 and the different in terms order amount by Customer City

|  |  |  |  |
| --- | --- | --- | --- |
|  | Year2005 | Year206 | Diff |
| New York | 5000 | 6000 | -1000 |
| California | 6000 | 3000 | 3000 |
| London | 3000 | 1000 | 2000 |

select c.city,sum(o.Order\_amt) as 2005\_amount from customer c inner join orders o on o.Cust\_Id = c.Cust\_Id where o.Order\_date like '\_\_\_\_\_\_2005%' or o.Order\_date like '\_\_\_\_\_\_2006%' group by c.city;

