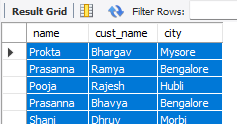
**MYSQL Assignment-2**

**1.** write a SQL query to find the salesperson and customer who reside in the same city Return Salesman, cust\_name and city.

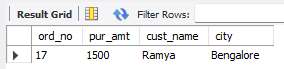
**select s.name, c.cust\_name, s.city from salesman s inner join customer c on s.city=c.city;**

****

**2.** write a SQL query to find those orders where the order amount exists between 500

and 2000. Return ord\_no, purch\_amt, cust\_name, city

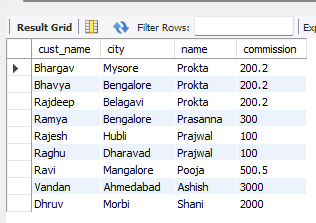
**select o.ord\_no, o.pur\_amt, c.cust\_name, c.city from orders o inner join customer c on c.customer\_id = o.customer\_id where o.pur\_amt between 500 and 2000;**

****

**3.** write a SQL query to find the salesperson(s) and the customer(s) he represents.

Return Customer Name, city, Salesman, commission

**select c.cust\_name, c.city, s.name, s.commission from salesman s inner join customer c on s.salesman\_id = c.salesman\_id;**

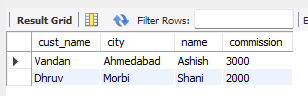


**4.** write a SQL query to find salespeople who received commissions of more than 12

percent from the company. Return Customer Name, customer city, Salesman,

commission.

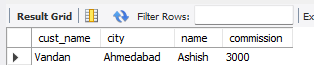
**select c.cust\_name, c.city, s.name, s.commission from salesman s inner join customer c on s.salesman\_id = c.salesman\_id inner join orders o on c.salesman\_id = o.salesman\_id where s.commission/o.pur\_amt > 0.12;**



**5.** write a SQL query to locate those salespeople who do not live in the same city where their customers live and have received a commission of more than 12% from the company. Return Customer Name, customer city, Salesman, salesman city,

commission

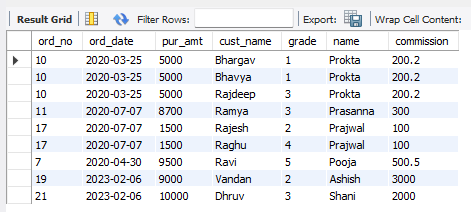
**select c.cust\_name, c.city, s.name, s.commission from salesman s inner join customer c on s.salesman\_id = c.salesman\_id inner join orders o on c.salesman\_id = o.salesman\_id where s.commission/o.pur\_amt > 0.12 and s.city != c.city;**

****

**6.** write a SQL query to find the details of an order. Return ord\_no, ord\_date,

purch\_amt, Customer Name, grade, Salesman, commission

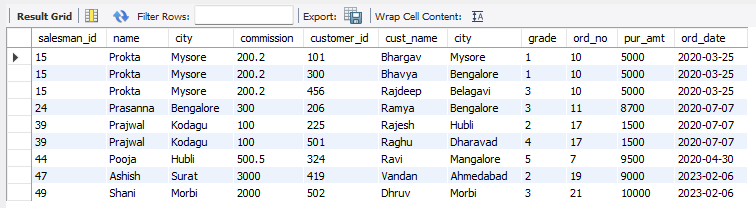
**select o.ord\_no, o.ord\_date, o.pur\_amt, c.cust\_name, c.grade, s.name, s.commission from salesman s inner join customer c on s.salesman\_id = c.salesman\_id inner join orders o on c.salesman\_id = o.salesman\_id;**

****

**7.** Write a SQL statement to join the tables salesman, customer and orders so that the

same column of each table appears once and only the relational rows are returned.

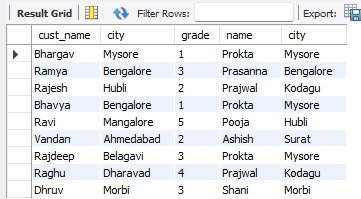
**select s.salesman\_id, s.name, s.city, s.commission, c.customer\_id, c.cust\_name, c.city, c.grade, o.ord\_no, o.pur\_amt, o.ord\_date from salesman s inner join customer c on s.salesman\_id = c.salesman\_id inner join orders o on c.salesman\_id = o.salesman\_id;**

****

**8.** write a SQL query to display the customer name, customer city, grade, salesman,

salesman city. The results should be sorted by ascending customer\_id.

**select c.cust\_name, c.city, c.grade, s.name, s.city from customer c inner join salesman s on s.salesman\_id = c.salesman\_id order by customer\_id;**

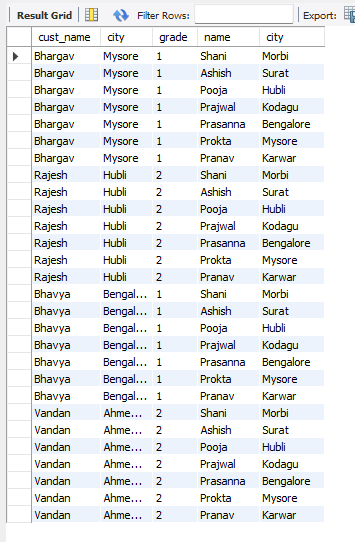


**9.** write a SQL query to find those customers with a grade less than 300. Return

cust\_name, customer city, grade, Salesman, salesmancity. The result should be

ordered by ascending customer\_id.

**select c.cust\_name, c.city, c.grade, s.name, s.city from customer c left join salesman s on c.salesman\_id = c.salesman\_id where c.grade < 3 order by customer\_id;**

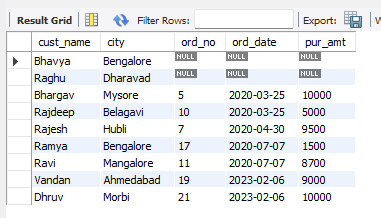


**10.** Write a SQL statement to make a report with customer name, city, order number,

order date, and order amount in ascending order according to the order date to

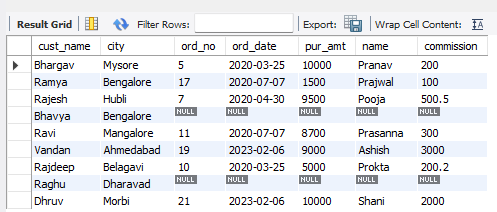
determine whether any of the existing customers have placed an order or not

**select c.cust\_name, c.city, o.ord\_no, o.ord\_date, o.pur\_amt from customer c left join orders o on c.customer\_id = o.customer\_id order by o.ord\_date asc;**



**11.** Write a SQL statement to generate a report with customer name, city, order number, order date, order amount, salesperson name, and commission to determine if any of the existing customers have not placed orders or if they have placed orders through their salesman or by themselves

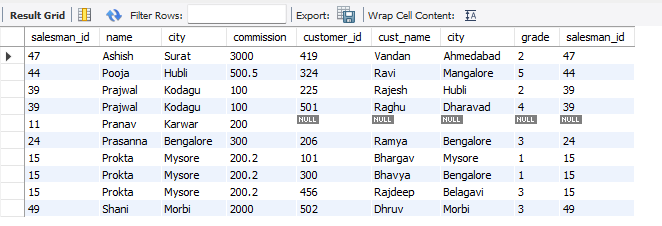
**select c.cust\_name, c.city, o.ord\_no, o.ord\_date, o.pur\_amt, s.name, s.commission from customer c left join orders o on c.customer\_id = o.customer\_id left join salesman s on o.salesman\_id = s.salesman\_id;**



**12.** Write a SQL statement to generate a list in ascending order of salespersons who

work either for one or more customers or have not yet joined any of the customers

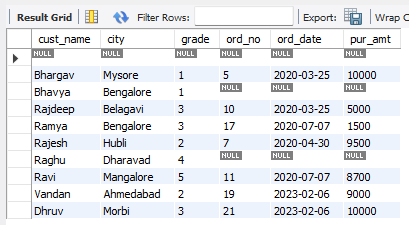
**select \* from salesman s left join customer c on s.salesman\_id = c.salesman\_id order by s.name;**



**13.** write a SQL query to list all salespersons along with customer name, city, grade,

order number, date, and amount.

**select c.cust\_name, c.city, c.grade, o.ord\_no, o.ord\_date, o.pur\_amt from salesman s left join customer c on s.salesman\_id = c.salesman\_id left join orders o on c.customer\_id = o.customer\_id;**



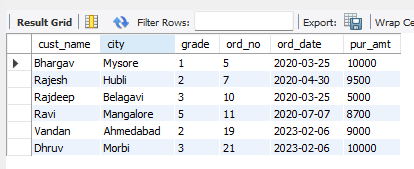
**14.** Write a SQL statement to make a list for the salesmen who either work for one or

more customers or yet to join any of the customers. The customer may have placed,

either one or more orders on or above order amount 2000 and must have a grade, or

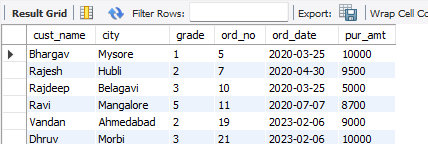
he may not have placed any order to the associated supplier.

**select c.cust\_name, c.city, c.grade, o.ord\_no, o.ord\_date, o.pur\_amt from salesman s left join customer c on s.salesman\_id = c.salesman\_id left join orders o on c.customer\_id = o.customer\_id where o.pur\_amt>=2000 and c.grade is not null;**



**15.** Write a SQL statement to generate a list of all the salesmen who either work for one or more customers or have yet to join any of them. The customer may have placed one or more orders at or above order amount 2000, and must have a grade, or he may not have placed any orders to the associated supplier.

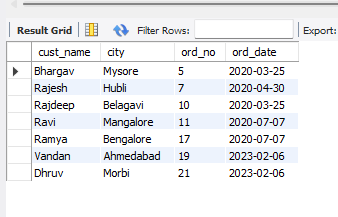
**select c.cust\_name, c.city, c.grade, o.ord\_no, o.ord\_date, o.pur\_amt from salesman s left join customer c on s.salesman\_id = c.salesman\_id left join orders o on c.customer\_id = o.customer\_id where o.pur\_amt>=2000 and c.grade is not null;**

****

**16.** Write a SQL statement to generate a report with the customer name, city, order no. order date, purchase amount for only those customers on the list who must have a grade and placed one or more orders or which order(s) have been placed by the

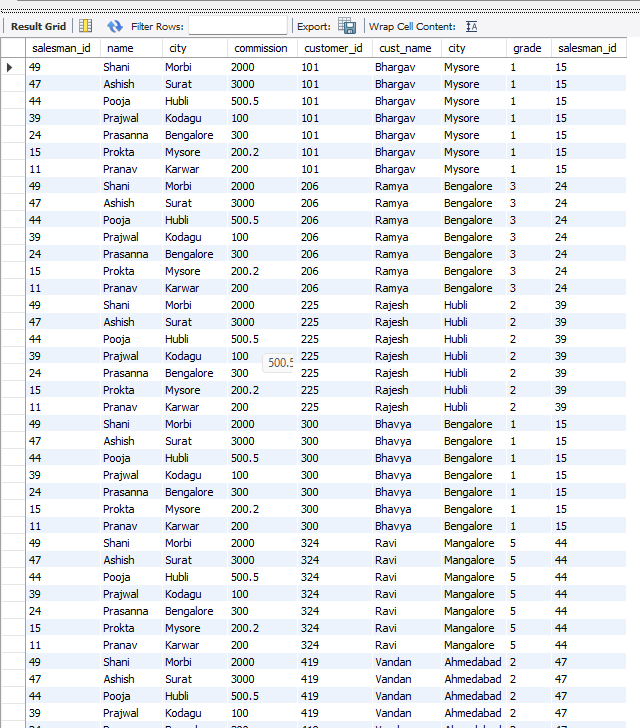
customer who neither is on the list nor has a grade.

**select c.cust\_name, c.city, o.ord\_no, o.ord\_date from customer c cross join orders o on c.customer\_id = o.customer\_id where c.grade is not null;**



**17.** Write a SQL query to combine each row of the salesman table with each row of the customer table

**select \* from salesman s cross join customer c;**

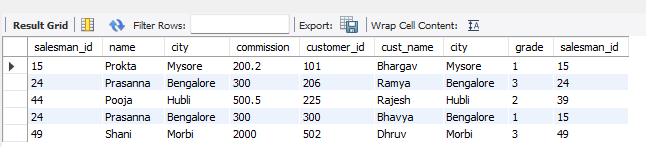
****

**18.** Write a SQL statement to create a Cartesian product between salesperson and

customer, i.e. each salesperson will appear for all customers and vice versa for that

salesperson who belongs to that city

**select \* from salesman s cross join customer c on s.city = c.city;**

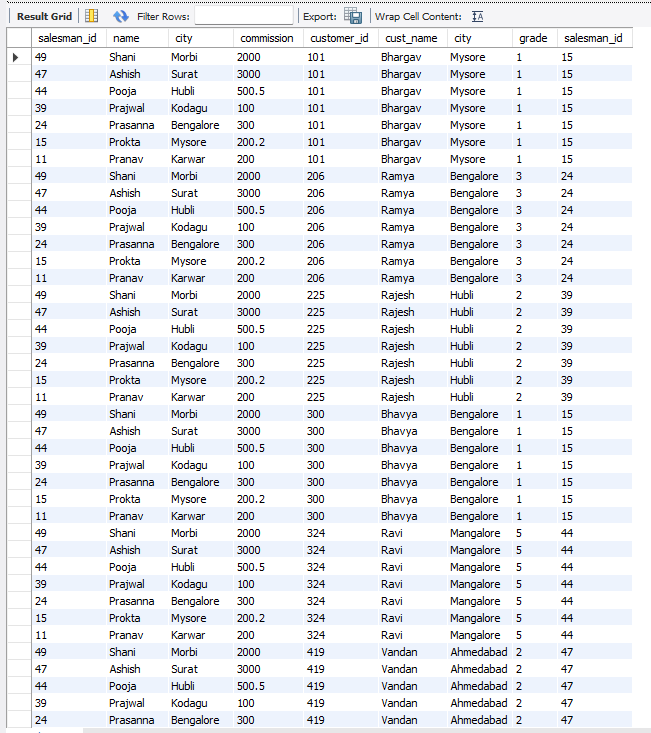
****

**19.** Write a SQL statement to create a Cartesian product between salesperson and

customer, i.e. each salesperson will appear for every customer and vice versa for

those salesmen who belong to a city and customers who require a grade

**select \* from salesman s cross join customer c on s.city is not null and c.grade is not null;**



**20.** Write a SQL statement to make a Cartesian product between salesman and

customer i.e. each salesman will appear for all customers and vice versa for those

salesmen who must belong to a city which is not the same as his customer and the

customers should have their own grade

**select \* from salesman s cross join customer c on s.city != c.city and s.city is not null;**

