**SQL 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**

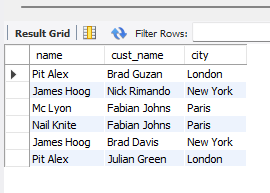
**Query:**

select saleman.name,cust\_name,customer.city from saleman,customer where saleman.city=customer.city;

OR

select saleman.name,cust\_name,customer.city from saleman inner join customer on saleman.city=customer.city;

**Output:**

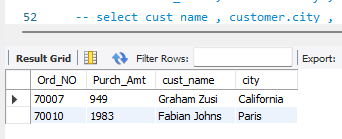


1. **write a SQL query to find those orders where the order amount exists between 500 and 2000. Return ord\_no, purch\_amt, cust\_name, city**

**Query:**

select Ord\_NO,Purch\_Amt,cust\_name,city from orders ,customer where customer.customer\_id = orders.customer\_id And Purch\_Amt between 500 and 2000;

**Output:**

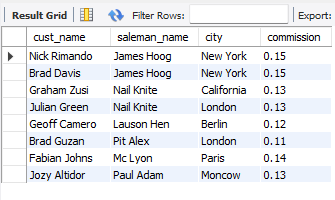


1. **write a SQL query to find the salesperson(s) and the customer(s) he represents. Return Customer Name, city, Salesman, commission**

**Query:**

select cust\_name,name as saleman\_name, customer.city,commission from saleman,customer where customer.salesman\_id = saleman.salesman\_id;

**Output:**



1. **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.**

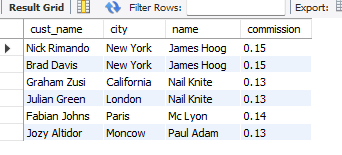
**Query:**

select cust\_name , customer.city , saleman.name, commission from saleman inner join customer on saleman.salesman\_id = customer.salesman\_id AND commission >0.12;

OR

select cust\_name , customer.city , saleman.name, commission from saleman, customer where saleman.salesman\_id = customer.salesman\_id AND commission >0.12;

**Output:**



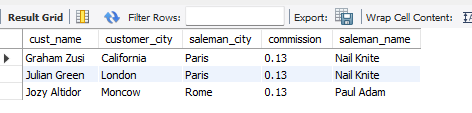
1. **write a SQL query to locate those salespeople who do not live in the same city where their customers live and have received a commission of more than 12% from the company. Return Customer Name, customer city, Salesman, salesman city, commission**

**Query:**

select cust\_name, customer.city as customer\_city,saleman.city as saleman\_city,commission ,name as saleman\_name from saleman,customer

where customer.salesman\_id = saleman.salesman\_id AND not(customer.city = saleman.city) AND commission > 0.12;

Output:



1. **write a SQL query to find the details of an order. Return ord\_no, ord\_date, purch\_amt, Customer Name, grade, Salesman, commission**

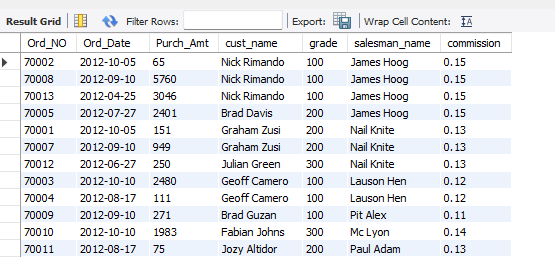
**Query:**

select Ord\_NO,Ord\_Date,Purch\_Amt,cust\_name,grade,name as salesman\_name ,commission from saleman

inner join customer on saleman.salesman\_id = customer.salesman\_id

inner join orders on customer.customer\_id = orders.customer\_id ;

**Output:**

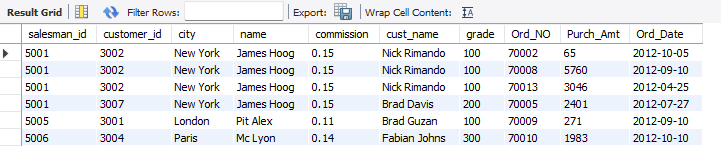


1. **Write a SQL statement to join the tables salesman, customer and orders so that the same column of each table appears once and only the relational rows are returned.**

**Query:**

select \* from saleman natural join customer natural join orders;

**Output:**



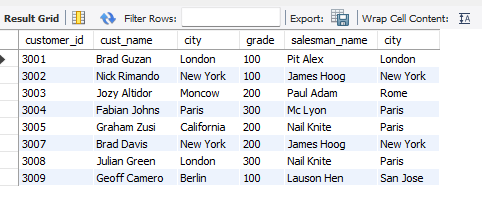
1. **write a SQL query to display the customer name, customer city, grade, salesman, salesman city. The results should be sorted by ascending customer\_id.**

**Query:**

select customer\_id,cust\_name,customer.city , grade ,name as salesman\_name,saleman.city from saleman inner join customer on saleman.salesman\_id = customer.salesman\_id

order by customer\_id asc;

**Output:**



1. **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.**

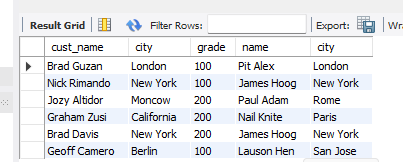
**Query:**

select cust\_name,customer.city,grade , saleman.name , saleman.city from saleman inner join customer on

customer.salesman\_id = saleman.salesman\_id And customer.grade < 300

order by customer.customer\_id asc;

**Output:**



1. **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**

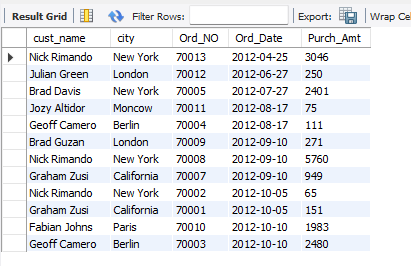
**Query:**

select cust\_name,customer.city,Ord\_NO,Ord\_Date,Purch\_Amt from customer

inner join orders on customer.customer\_id = orders.customer\_id

order by ord\_date asc;

**Output:**

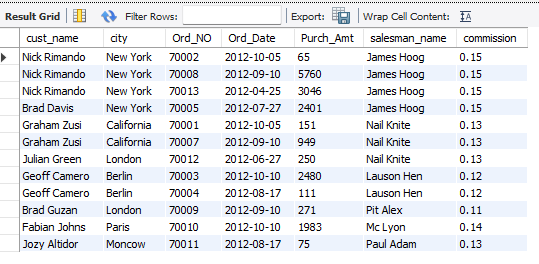


1. **Write a SQL statement to generate a report with customer name, city, order number, order date, order amount, salesperson name, and commission to determine if any of the existing customers have not placed orders or if they have placed orders through their salesman or by themselves**

**Query:**

select \* from saleman cross join customer where (saleman.city != customer.city) and customer.grade is not null;

**Output:**

****

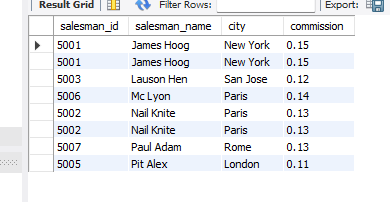
1. **Write a SQL statement to generate a list in ascending order of salespersons who work either for one or more customers or have not yet joined any of the customers**

**Query:**

select saleman.salesman\_id,name as salesman\_name,saleman.city,commission from saleman left join customer on customer.salesman\_id = saleman.salesman\_id

order by name**;**

**Output:**

****

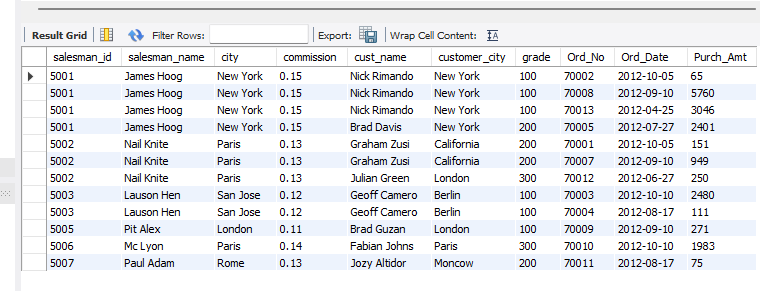
1. **write a SQL query to list all salespersons along with customer name, city, grade, order number, date, and amount.**

**Query:**

select saleman.salesman\_id,name as salesman\_name,saleman.city,commission,cust\_name ,customer.city as customer\_city,grade,Ord\_No,Ord\_Date,Purch\_Amt

from saleman left join customer on customer.salesman\_id = saleman.salesman\_id left join orders on customer.customer\_id = orders.customer\_id;

**Output:**

****

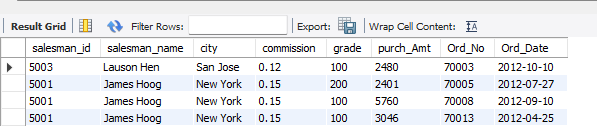
1. **Write a SQL statement to make a list for the salesmen who either work for one or more customers or yet to join any of the customers. The customer may have placed, either one or more orders on or above order amount 2000 and must have a grade, or he may not have placed any order to the associated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

**Query:**

select saleman.salesman\_id,name as salesman\_name,saleman.city,commission,grade,purch\_Amt,Ord\_No ,Ord\_Date from saleman left join customer on saleman.salesman\_id = customer.salesman\_id

left join orders on orders.customer\_id = customer.customer\_id where Purch\_Amt >=2000 and customer.grade is not null ;

**Output:**

****

1. **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.**

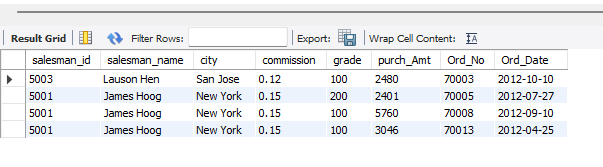
**Query:**

select saleman.salesman\_id,name as salesman\_name,saleman.city,commission,grade ,Ord\_No ,Ord\_Date from saleman

left join customer on saleman.salesman\_id = customer.salesman\_id

left join orders on orders.customer\_id = customer.customer\_id where Purch\_Amt >2000 and customer.grade is not null ;

**Output:**

****

1. **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.**

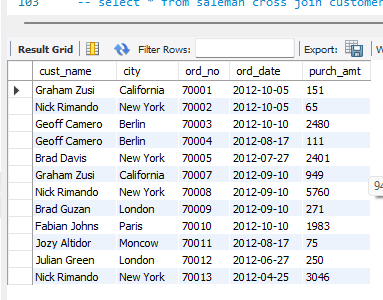
**Query:**

select customer.cust\_name, customer.city, orders.ord\_no, orders.ord\_date, orders.purch\_amt from orders left join customer on orders.customer\_id = customer.customer\_id union

select customer.cust\_name, customer.city, orders.ord\_no, orders.ord\_date, orders.purch\_amt from orders right join customer on orders.customer\_id = customer.customer\_id

where grade is not null;

**Output:**

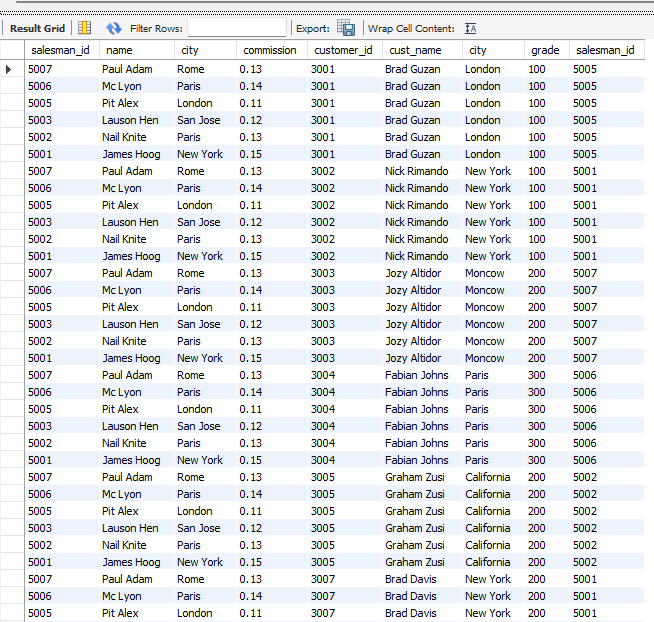
****

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

**Query:**

select \* from saleman cross join customer;

**Output:**

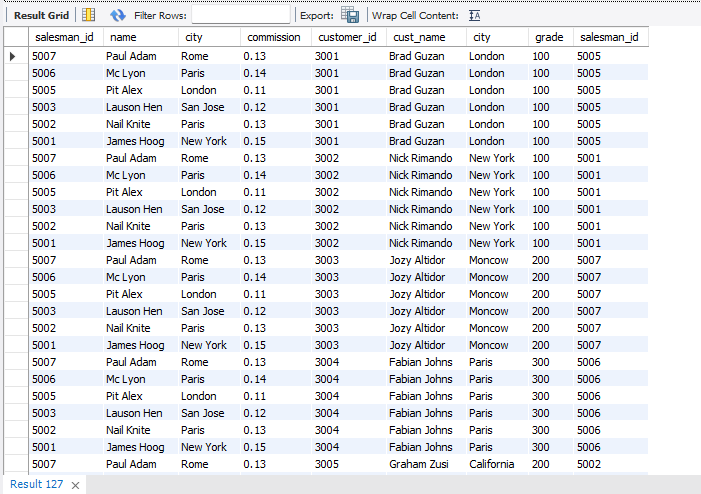


1. **Write a SQL statement to create a Cartesian product between salesperson and customer, i.e. each salesperson will appear for all customers and vice versa for that salesperson who belongs to that city**

**Query:**

select \* from saleman cross join customer where saleman.city is not null ;

**Output:**

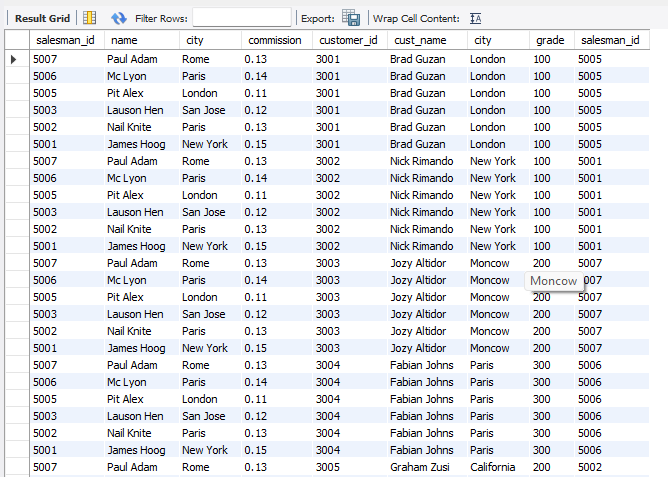


1. **Write a SQL statement to create a Cartesian product between salesperson and customer, i.e. each salesperson will appear for every customer and vice versa for those salesmen who belong to a city and customers who require a grade**

**Query:**

select \* from saleman cross join customer where saleman.city is not null and customer.grade is not null ;

**Output:**



1. **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**

**Query:**

select \* from saleman cross join customer where (saleman.city != customer.city) and customer.grade is not null;

**Output:**

