-------------------------------------------------------------------------------------------------------------------------- Assignment 2 : Retrieve data using join with where clause -------------------------------------------------------------------------------------------------------------------------- Sample table1: salesman

-salesman\_id

-name -city

-commission

-- Sample table2: customer

-customer\_id

-cust\_name

-city

-grade -salesman\_id

-- Sample table3: orders

-ord\_no

-purch\_amt

-ord\_date

-customer\_id

-salesman\_id

CREATE DATABASE Company\_db;

USE Company\_db;

CREATE TABLE salesman(

Salesman\_id int IDENTITY(1,1) PRIMARY KEY,

Full\_name varchar(50) DEFAULT 'anonymous',

City varchar(50) DEFAULT 'ahmedabad',

Commission decimal(5,2) DEFAULT 20

);

SELECT \* FROM salesman;

CREATE TABLE customer(

Customer\_id int IDENTITY(1,1) PRIMARY KEY,

Cust\_name varchar(50) DEFAULT 'anonymous',

City varchar(50) DEFAULT 'ahmedabad',

Grade int,

Salesman\_id int FOREIGN KEY REFERENCES salesman(Salesman\_id)

);

SELECT \* FROM customer;

CREATE TABLE orders (

Ord\_no int IDENTITY(1,1) PRIMARY KEY,

Purch\_amt decimal(10,2),

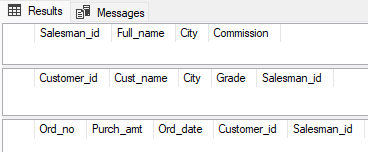
Ord\_date date,

Customer\_id int FOREIGN KEY REFERENCES customer(Customer\_id),

Salesman\_id int FOREIGN KEY REFERENCES salesman(Salesman\_id)

);

SELECT \* FROM orders;



INSERT INTO salesman ( Full\_name, Commission)

VALUES ('Person1',10),

('Person4',20),

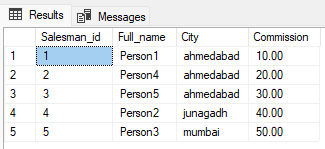
('Person5',30);

INSERT INTO salesman ( Full\_name, City,Commission)

VALUES ('Person2','junagadh',40),

('Person3','mumbai',50);

SELECT \* FROM salesman;



INSERT INTO customer ( CUST\_name, City,grade, Salesman\_id)

VALUES ('CUST1','ahmedabad',400,1),

('CUST2','mumbai',500,5),

('CUST3','ahmedabad',100,1),

('CUST4','kolkata',200,3),

('CUST5','ahmedabad',400,1),

('CUST6','junagadh',300,5),

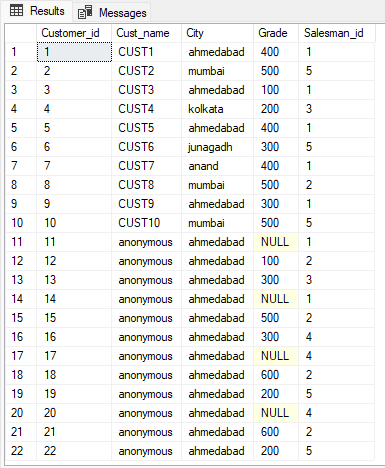
('CUST7','anand',400,1),

('CUST8','mumbai',500,2),

('CUST9','ahmedabad',300,1),

('CUST10','mumbai',500,5);

INSERT INTO customer ( grade, Salesman\_id)

VALUES (NULL,1),

(100,2),

(300,3),

(NULL,1),

(500,2),

(300,4),

(NULL,4),

(600,2),

(200,5),

(NULL,4),

(600,2),

(200,5);

SELECT \* FROM customer;

INSERT INTO orders( Purch\_amt, Ord\_date, Customer\_id, Salesman\_id)

VALUES (200.00 ,'2023-01-01',22,5),

(1200.00,'2023-01-01',2,4),

(2200.00,'2023-01-02',21,5),

(2050.00,'2023-01-02',20,3),

(2005.00,'2023-01-03',1,1),

(7200.00,'2023-01-04',3,2),

(2900.00,'2023-01-05',12,2),

(2000.00,'2023-01-05',11,5),

(200.00 ,'2023-01-05',9,3),

(100.00 ,'2023-01-06',6,3),

(2200.00,'2023-01-07',10,5),

(200.00 ,'2023-01-08',8,1),

(2030.00,'2023-01-09',5,1),

(4200.00,'2023-01-10',4,1),

(9200.00,'2023-01-10',7,4),

(1000.00,'2023-01-11',8,5),

(800.00 ,'2023-01-12',10,5),

(3000.00,'2023-01-12',2,2),

(20.00 ,'2023-01-12',12,4),

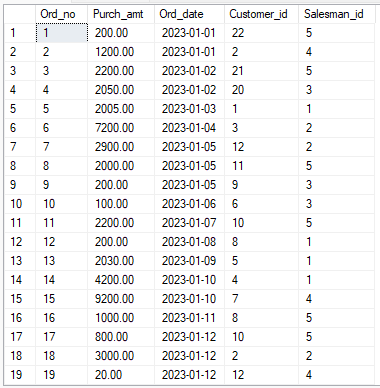
(100.00 ,'2023-01-12',3,3),

(300.00 ,'2023-01-13',13,2),

(2000.00,'2023-01-13',15,3),

(600.00 ,'2023-01-13',14,1);

SELECT \* FROM orders;

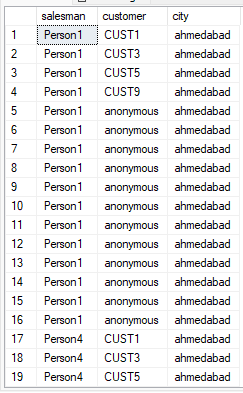


Q1 write a SQL query to find the salesperson and customer who reside in the same city. Return Salesman, cust\_name and city.

SELECT Full\_name as salesman, Cust\_name as customer, customer.City as city

FROM customer INNER JOIN salesman

ON salesman.City = customer.City;



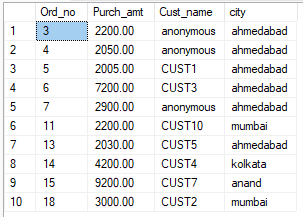
Q2 write a SQL query to find those orders where the order amount exists between 500 and 2000. Return ord\_no, purch\_amt, cust\_name, city.

SELECT Ord\_no, Purch\_amt, Cust\_name, customer.City as city

FROM orders INNER JOIN customer

ON orders.Customer\_id = customer.Customer\_id

WHERE Purch\_amt > 2000;

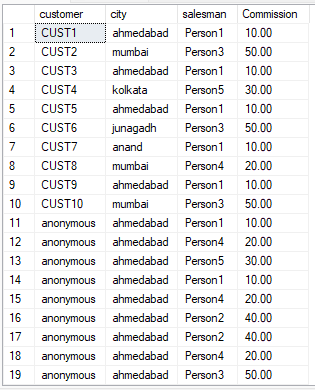


Q3 write a SQL query to find the salesperson(s) and the customer(s) he represents. Return Customer Name, city, Salesman, commission.

SELECT Cust\_name as customer, customer.City as city,Full\_name as salesman, Commission

FROM customer INNER JOIN salesman

ON salesman.Salesman\_id = customer.Salesman\_id;



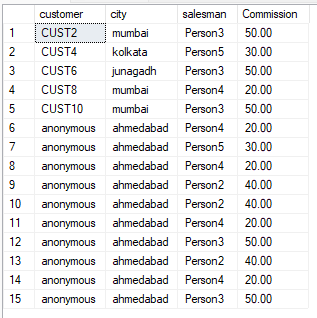
Q4 write a SQL query to find salespeople who received commissions of more than 12 percent from the company. Return Customer Name, customer city, Salesman, commission.

SELECT Cust\_name as customer, customer.City as city,Full\_name as salesman, Commission

FROM customer INNER JOIN salesman

ON salesman.Salesman\_id = customer.Salesman\_id

WHERE Commission > 12;



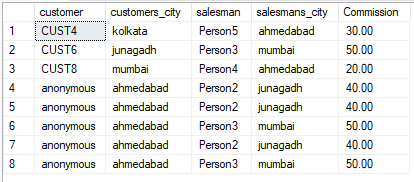
Q5 write a SQL query to locate those salespeople who do not live in the same city where their customers live and have received a commission of more than 12% from the company. Return Customer Name, customer city, Salesman, salesman city, commission.

SELECT Cust\_name as customer, customer.City as customers\_city,Full\_name as salesman, salesman.City as salesmans\_city,Commission

FROM customer INNER JOIN salesman

ON salesman.Salesman\_id = customer.Salesman\_id

WHERE salesman.City != customer.City AND Commission > 12;



Q6 write a SQL query to find the details of an order. Return ord\_no, ord\_date, purch\_amt, Customer Name, grade, Salesman, commission.

SELECT

Ord\_no,

Ord\_date,

Purch\_amt,

Cust\_name,

Grade,

Full\_name as salesman,

Commission

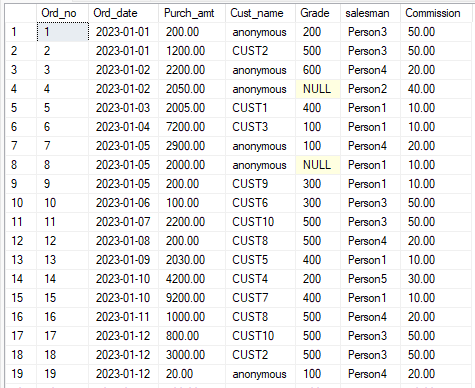
FROM customer

JOIN salesman

ON salesman.Salesman\_id = customer.Salesman\_id

JOIN orders

ON orders.Customer\_id = customer.Customer\_id;



Q7 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

salesman.Salesman\_id,

salesman.City,

customer.Customer\_id,

Ord\_no,

Purch\_amt,

Ord\_date,

Cust\_name,

Grade,

Full\_name as salesman,

Commission

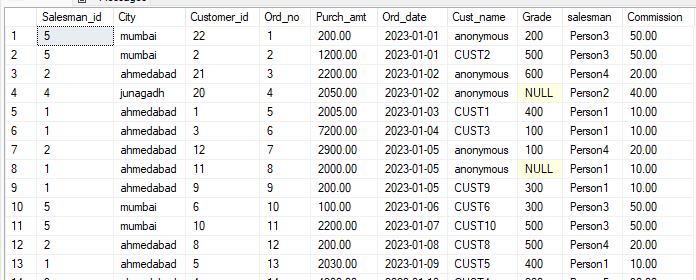
FROM customer

JOIN salesman

ON salesman.Salesman\_id = customer.Salesman\_id

JOIN orders

ON orders.Customer\_id = customer.Customer\_id;



Q8 write a SQL query to display the customer name, customer city, grade, salesman, salesman city. The results should be sorted by ascending customer\_id.

--The ORDER BY keyword sorts the records in ascending order by default.

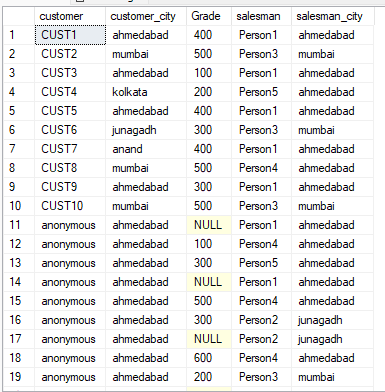
SELECT Cust\_name as customer, customer.City as customer\_city,

Grade,Full\_name as salesman, salesman.City as salesman\_city

FROM customer INNER JOIN salesman

ON salesman.Salesman\_id = customer.Salesman\_id

ORDER BY Customer\_id ASC;



Q9 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.

--The ORDER BY keyword sorts the records in ascending order by default.

SELECT Cust\_name as customer, customer.City as customer\_city,

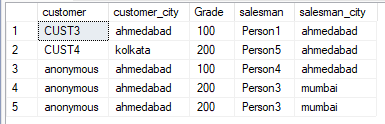
Grade,Full\_name as salesman, salesman.City as salesman\_city

FROM customer INNER JOIN salesman

ON salesman.Salesman\_id = customer.Salesman\_id

WHERE Grade < 300

ORDER BY Customer\_id ASC;



Q10 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

Cust\_name,

customer.City,

Ord\_no,

Ord\_date,

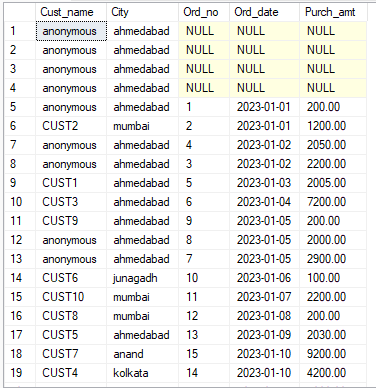
Purch\_amt

FROM customer

LEFT JOIN orders

ON orders.Customer\_id = customer.Customer\_id

ORDER BY Ord\_date ASC;



-- Additional Code

INSERT INTO orders

VALUES (200.00 , '2023-01-01',22,NULL),

(99999.99 ,'2023-01-02',20,NULL);

SELECT \* FROM orders;

Q11 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

Ord\_no,

Ord\_date,

Purch\_amt,

Cust\_name,

Grade,

Full\_name as salesman,

Commission

FROM orders

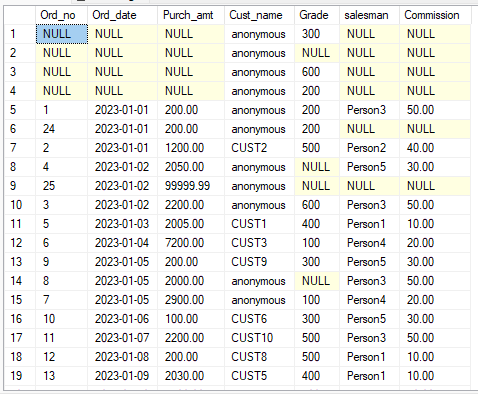
LEFT JOIN salesman

ON salesman.Salesman\_id = orders.Salesman\_id

RIGHT JOIN customer

ON orders.Customer\_id = customer.Customer\_id

ORDER BY Ord\_date ASC;



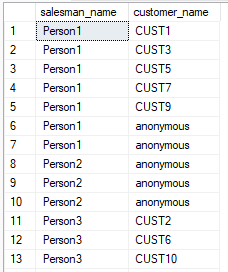
Q12 Write a SQL statement to generate a list in ascending order of salespersons who work either for one or more customers or have not yet joined any of the customers.

SELECT Full\_name as salesman\_name, Cust\_name as customer\_name

FROM salesman LEFT JOIN customer

ON salesman.Salesman\_id = customer.Salesman\_id

ORDER BY salesman.Full\_name;



Q13 write a SQL query to list all salespersons along with customer name, city, grade, order number, date, and amount.

SELECT Full\_name AS sales\_person , Cust\_name AS customer\_name,

salesman.City, Grade, Ord\_no, Ord\_date,Purch\_amt

FROM orders

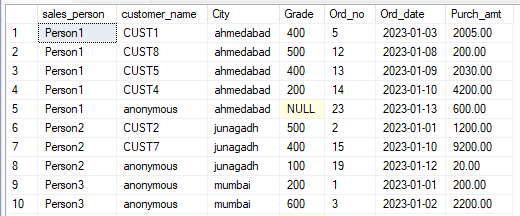
JOIN customer

ON orders.Customer\_id = customer.Customer\_id

JOIN salesman

ON salesman.Salesman\_id = orders.Salesman\_id

ORDER BY Full\_name ASC;



Q14 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 Full\_name AS sales\_person , Cust\_name AS customer\_name,

salesman.City, Grade, Ord\_no, Ord\_date,Purch\_amt

FROM salesman

JOIN customer

ON salesman.Salesman\_id=customer.Salesman\_id

JOIN orders

ON orders.Customer\_id= customer.Customer\_id

WHERE Purch\_amt < 2000

UNION

SELECT Full\_name AS sales\_person , Cust\_name AS customer\_name,

salesman.City, Grade, Ord\_no, Ord\_date,Purch\_amt

FROM salesman

JOIN customer

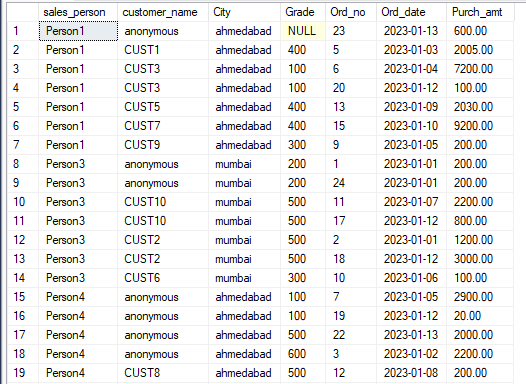
ON salesman.Salesman\_id=customer.Salesman\_id

JOIN orders

ON orders.Customer\_id= customer.Customer\_id

WHERE Purch\_amt >= 2000

AND Grade IS NOT NULL;



Q15 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 Full\_name AS sales\_person , Cust\_name AS customer\_name,

salesman.City, Grade, Ord\_no, Ord\_date,Purch\_amt

FROM salesman

JOIN customer

ON salesman.Salesman\_id=customer.Salesman\_id

JOIN orders

ON orders.Customer\_id= customer.Customer\_id

WHERE Purch\_amt < 2000

UNION ALL

SELECT Full\_name AS sales\_person , Cust\_name AS customer\_name,

salesman.City, Grade, Ord\_no, Ord\_date,Purch\_amt

FROM salesman

JOIN customer

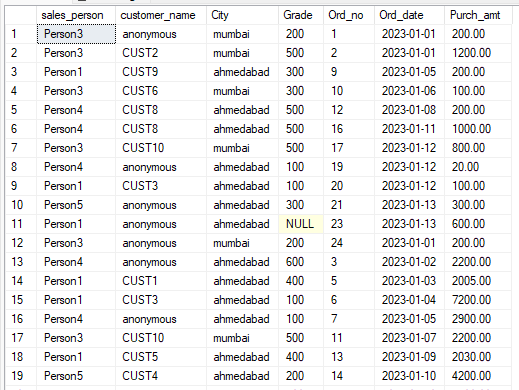
ON salesman.Salesman\_id=customer.Salesman\_id

JOIN orders

ON orders.Customer\_id= customer.Customer\_id

WHERE Purch\_amt >= 2000

AND Grade IS NOT NULL;



Q16 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 Cust\_name, City, Ord\_no, Ord\_date, Purch\_amt

FROM customer

LEFT JOIN orders

ON customer.Customer\_id = orders.Customer\_id

WHERE Grade IS NOT NULL AND Ord\_no IS NOT NULL

UNION

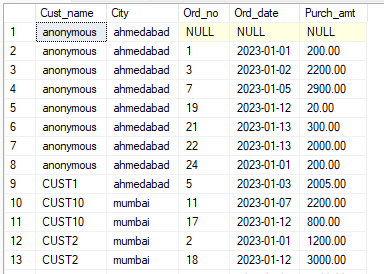
SELECT Cust\_name, City, Ord\_no, Ord\_date, Purch\_amt

FROM customer

LEFT JOIN orders

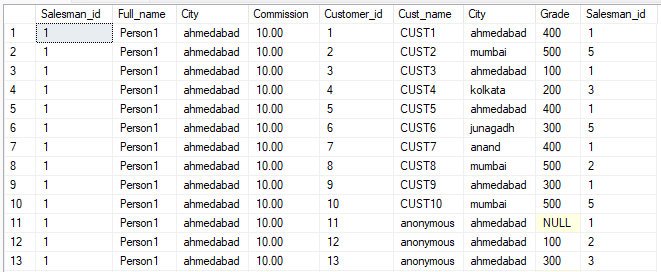
ON customer.Customer\_id = orders.Customer\_id

WHERE Grade IS NULL AND Ord\_no IS NULL;



Q17 Write a SQL query to combine each row of the salesman table with each row of the customer table.

SELECT \* FROM salesman CROSS JOIN customer;



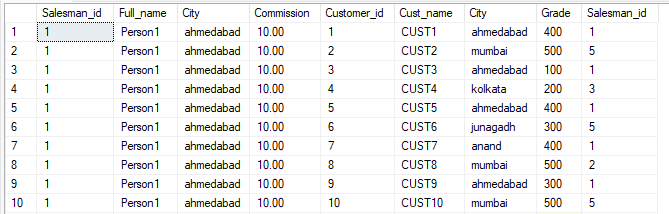
Q18 Write a SQL statement to create a Cartesian product between salesperson and customer, i.e. each salesperson will appear for all customers and vice versa for that salesperson who belongs to that city.

SELECT \*

FROM salesman

CROSS JOIN customer

WHERE salesman.City IS NOT NULL;



Q19 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.

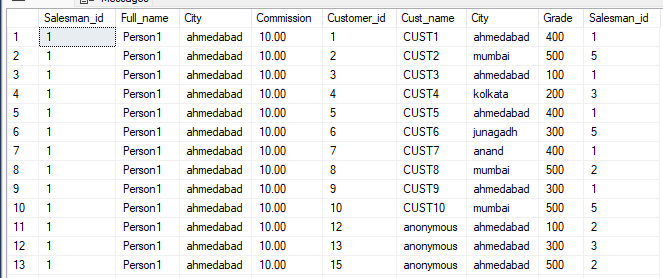
SELECT \*

FROM salesman

CROSS JOIN customer

WHERE salesman.City IS NOT NULL

AND Grade IS NOT NULL;



Q20 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

CROSS JOIN customer

WHERE salesman.City != customer.City

AND Grade IS NOT NULL;

