**Assignment 2: Retrieve data using join with where clause**

CREATE TABLE [salesman](

salesman\_id INT PRIMARY KEY IDENTITY(1,1),

name VARCHAR(20) NOT NULL,

city VARCHAR(20) NOT NULL,

commission float NOT NULL

);

CREATE TABLE [customer](

customer\_id INT PRIMARY KEY IDENTITY(1,1),

cust\_name VARCHAR(20) NOT NULL,

city VARCHAR(10) NOT NULL,

grade INT NOT NULL,

salesman\_id INT FOREIGN KEY REFERENCES salesman(salesman\_id) NOT NULL

);

CREATE TABLE [orders](

ord\_no INT PRIMARY KEY IDENTITY(1,1),

purch\_amt FLOAT NOT NULL,

ord\_date DATE NOT NULL,

customer\_id INT FOREIGN KEY REFERENCES customer(customer\_id) NOT NULL,

salesman\_id INT FOREIGN KEY REFERENCES salesman(salesman\_id) NOT NULL

);

INSERT INTO [salesman] VALUES

('john','ahmedabad',0.12),

('divak','mumbai',0.16),

('kartik','vadodara',0.15),

('aryan','banaras',0.11),

('diya','rajkot',0.19);

INSERT INTO [customer] VALUES

('Abhi','ahmedabad',100,1),

('yash','surat',400,2),

('ketul','vadodara',200,3),

('shyam','rajkot',300,4),

('meet','banaras',500,5);

INSERT INTO [orders] VALUES

(300,'2001-02-12',1,1),

(200,'2002-02-22',2,2),

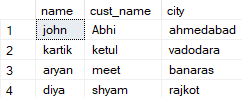
(400,'2003-04-12',3,3),

(600,'2002-08-12',4,4),

(750,'2004-02-12',5,5);

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.purch\_amt, c.cust\_name, c.city FROM orders o INNER JOIN customer c ON o.customer\_id = c.customer\_id where o.purch\_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 "customer name", c.city "customer city", s.city "salesman city", s.name "salesman name", s.commission FROM customer c INNER JOIN salesman s ON c.salesman\_id = s.salesman\_id;

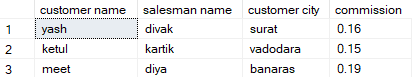


4. write a SQL query to find salespeople who received commissions of more than 12

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

commission.

SELECT c.cust\_name "customer name",s.name "salesman name", c.city "customer city", s.commission FROM customer c INNER JOIN salesman s ON c.salesman\_id = s.salesman\_id WHERE s.commission>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

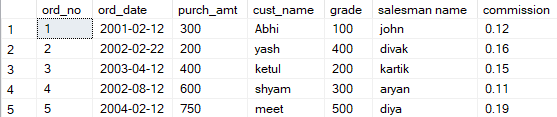
SELECT c.cust\_name "customer name", c.city "customer city", s.name "salesman name", s.city "salesman city", s.commission FROM customer c INNER JOIN salesman s ON c.salesman\_id = s.salesman\_id WHERE c.city<>s.city AND commission>0.12;



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

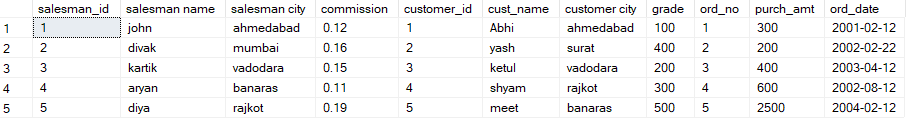
purch\_amt, Customer Name, grade, Salesman, commission

SELECT o.ord\_no, o.ord\_date, o.purch\_amt, c.cust\_name, c.grade, s.name "salesman name", s.commission FROM orders o INNER JOIN customer c ON o.customer\_id = c.customer\_id INNER JOIN salesman s ON o.salesman\_id = s.salesman\_id;



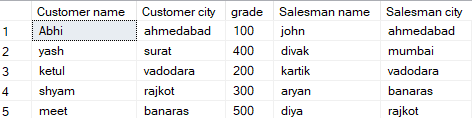
7. Write a SQL statement to join the tables salesman, customer and orders so that the same column of each table appears once and only the relational rows are returned.

SELECT s.salesman\_id, s.name "salesman name", s.city "salesman city", s.commission, c.customer\_id, c.cust\_name, c.city "customer city", c.grade, o.ord\_no, o.purch\_amt, o.ord\_date FROM salesman s INNER JOIN customer c ON s.salesman\_id = c.salesman\_id INNER JOIN orders o ON c.customer\_id = o.customer\_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 "Customer name", c.city "Customer city", c.grade, s.name "Salesman name", s.city "Salesman city" FROM customer c INNER JOIN salesman s ON c.salesman\_id = s.salesman\_id ORDER BY c.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

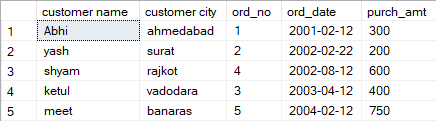
ordered by ascending customer\_id.

SELECT c.cust\_name "customer name", c.city "customer city", c.grade, s.name "salesman name", s.city "salesman city" FROM customer c INNER JOIN salesman s ON c.salesman\_id = s.salesman\_id WHERE c.grade < 300 ORDER BY customer\_id;



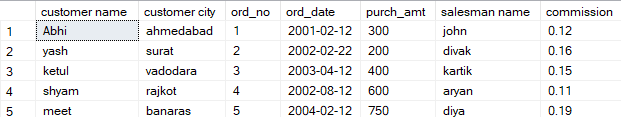
10. Write a SQL statement to make a report with customer name, city, order number, order date, and order amount in ascending order according to the order date to determine whether any of the existing customers have placed an order or not

SELECT c.cust\_name "customer name", c.city "customer city", o.ord\_no , o.ord\_date, o.purch\_amt FROM customer c LEFT JOIN orders o ON o.customer\_id = c.customer\_id ORDER BY o.ord\_date;



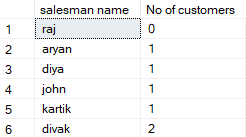
11. Write a SQL statement to generate a report with customer name, city, order number, order date, order amount, salesperson name, and commission to determine if any of the existing customers have not placed orders or if they have placed orders through their salesman or by themselves

SELECT c.cust\_name "customer name", c.city "customer city", o.ord\_no, o.ord\_date, o.purch\_amt, s.name "salesman name", s.commission FROM customer c LEFT JOIN orders o ON c.customer\_id = o.customer\_id LEFT JOIN salesman s ON c.salesman\_id = s.salesman\_id;



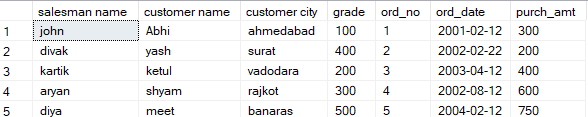
12. Write a SQL statement to generate a list in ascending order of salespersons who work either for one or more customers or have not yet joined any of the customers

SELECT s.name "salesman name", COUNT(c.customer\_id) "No of customers" FROM salesman s LEFT JOIN customer c ON s.salesman\_id = c.salesman\_id GROUP BY s.name ORDER BY COUNT(c.customer\_id);



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

SELECT s.name "salesman name", c.cust\_name "customer name", c.city "customer city", c.grade, o.ord\_no, o.ord\_date, o.purch\_amt FROM salesman s INNER JOIN customer c ON s.salesman\_id = c.salesman\_id INNER 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 s.name "salesman name", c.cust\_name "customer name" 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.purch\_amt >=2000 AND c.grade IS NOT NULL;



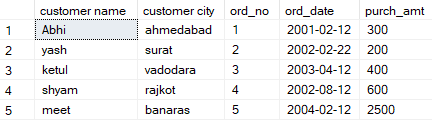
15. Write a SQL statement to generate a list of all the salesmen who either work for one or more customers or have yet to join any of them. The customer may have placed one or more orders at or above order amount 2000, and must have a grade, or he may not have placed any orders to the associated supplier.

SELECT s.name "salesman name", c.cust\_name "customer name" 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.purch\_amt >=2000 AND c.grade IS NOT NULL;



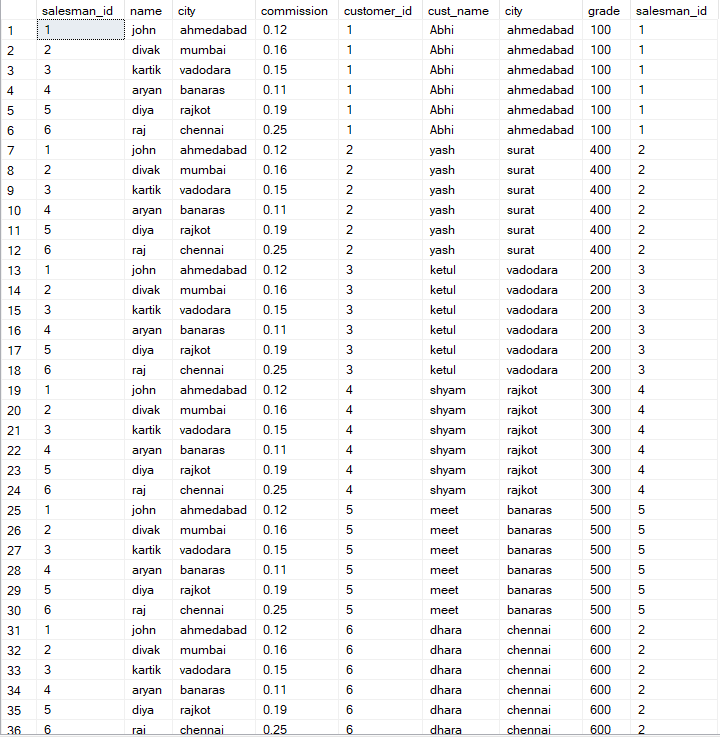
16. Write a SQL statement to generate a report with the customer name, city, order no. order date, purchase amount for only those customers on the list who must have a grade and placed one or more orders or which order(s) have been placed by the customer who neither is on the list nor has a grade.

SELECT c.cust\_name "customer name", c.city "customer city", o.ord\_no, o.ord\_date, o.purch\_amt FROM customer c FULL OUTER JOIN orders o ON c.customer\_id = o.customer\_id WHERE (c.grade IS NOT NULL AND o.ord\_no IS NOT NULL) OR (c.grade IS NULL AND o.ord\_no IS NULL);



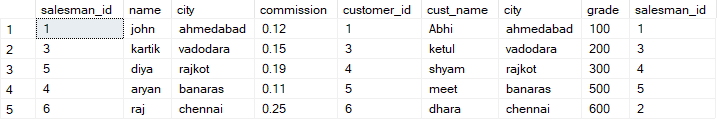
17. Write a SQL query to combine each row of the salesman table with each row of the customer table

SELECT \* FROM salesman CROSS JOIN customer;



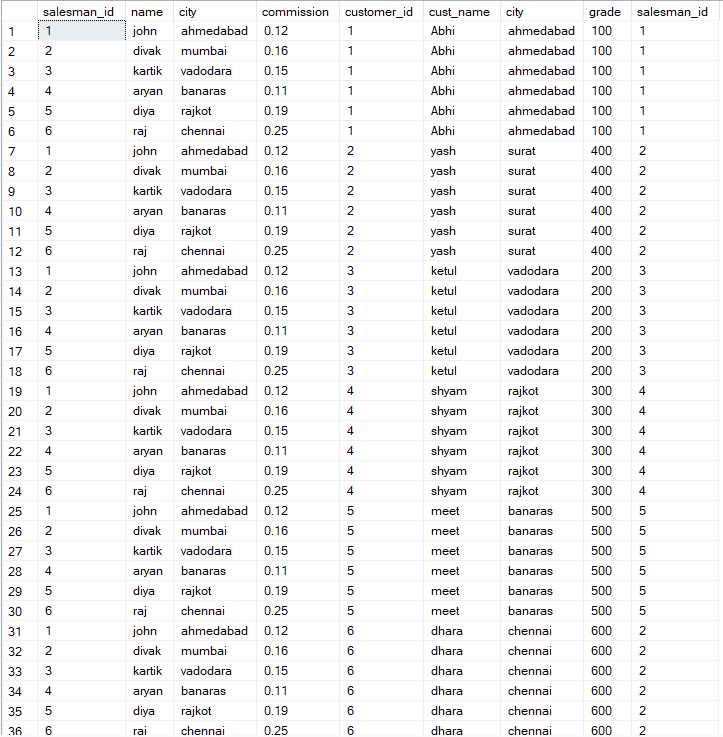
18. Write a SQL statement to create a Cartesian product between salesperson and customer, i.e. each salesperson will appear for all customers and vice versa for that salesperson who belongs to that city

SELECT \* FROM salesman s CROSS JOIN customer c WHERE 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 WHERE 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 WHERE s.city <> c.city AND c.grade IS NOT NULL;

