Assignment – 2

Retrieve data using join with where clause

* ***Creating Table salesman***
* create table salesman

(

salesmanID int primary key,

name varchar(20) null,

city varchar(20) null,

commission decimal(3,2)

)

* ***Inserting data into salesman table***
* insert into salesman

values(5001,'James Hoog','New York',0.15)

* insert into salesman

values(5002,'Nail Knite','Paris',0.13)

* insert into salesman

values(5005,'Pit Alex','London',0.11)

* insert into salesman

values(5006,'Mc Lyon','Paris',0.14)

* insert into salesman

values(5007,'Paul Adam','Rome',0.13)

* insert into salesman

values(5003,'Lauson Hen','San Jose',0.12)



* ***Creating Table customer***
* create table customer

(

customerID int primary key,

cust\_name varchar(20) null,

city varchar(20) null,

grade int null,

salesmanID int null

)

* ***Inserting data into customer table***
* insert into customer

values(3002,'Nick Rimando','New York',100,5001)

* insert into customer

values(3007,'Brad Davis','New York',200,5001)

* insert into customer

values(3005,'Graham Zusi','California',200,5002)

* insert into customer

values(3008,'Julian Green','London',300,5002)

* insert into customer

values(3004,'Fabian Johnson','Paris',300,5006)

* insert into customer

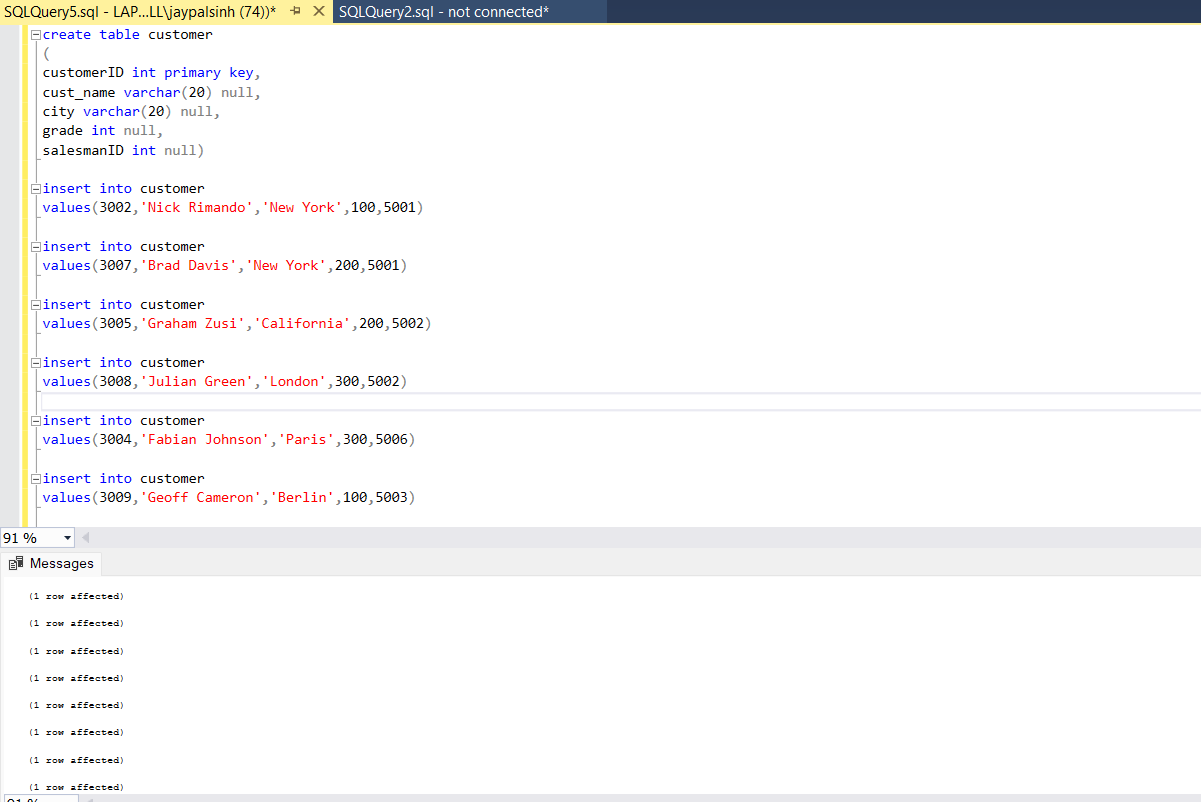
values(3009,'Geoff Cameron','Berlin',100,5003)

* insert into customer

values(3003,'Jozy Altidor','Moscow',200,5007)

* insert into customer

values(3001,'Brad Guzan','London',null,5005)



* ***Creating Table customer***
* create table orders

(

ord\_no int primary key,

purch\_amt decimal(6,2) null,

ord\_date date null,

customerID int null,

salesmanID int null

)

* ***Inserting data into customer table***
* insert into orders

values(70001,150.5,'2012-10-05',3005,5002)

* insert into orders

values(70009,270.65,'2012-09-10',3001,5005)

* insert into orders

values(70002,65.26,'2012-10-05',3002,5001)

* insert into orders

values(70004,110.5,'2012-08-17',3009,5003)

* insert into orders

values(70007,948.5,'2012-09-10',3005,5002)

* insert into orders

values(70005,2400.6,'2012-07-27',3007,5001)

* insert into orders

values(70008,5760,'2012-09-10',3002,5001)

* insert into orders

values(70010,1983.43,'2012-10-10',3004,5006)

* insert into orders

values(70003,2480.5,'2012-10-10',3009,5003)

* insert into orders

values(70012,250.45,'2012-06-27',3008,5002)

* insert into orders

values(70011,75.29,'2012-08-17',3003,5007)

* insert into orders

values(70013,3045.6,'2012-04-25',3002,5001)

Graphical user interface, application

Description automatically generated

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

select name as Salesman, cust\_name, city

from customer

join salesman

on customer.cust\_city = salesman.city

Graphical user interface, text, application, email

Description automatically generated

**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 ord\_no, purch\_amt, cust\_name, cust\_city

from orders

join customer

on orders.customerID = customer.customerID

where purch\_amt between 500 and 2000

Graphical user interface, text, application, email

Description automatically generated

**3**. 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 Name', city, name as Salesman, commission

from salesman

join customer

on salesman.salesmanID = customer.salesmanID

Text

Description automatically generated with medium confidence

**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 cust\_name as 'Customer Name', cust\_city as 'customer city', name as Salesman, commission

from salesman

join customer

on salesman.salesmanID = customer.salesmanID

where commission > 0.12

Graphical user interface, text, application, email

Description automatically generated

**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 cust\_name as 'Customer Name', cust\_city as 'customer city', name as Salesman, city as 'salesman city', commission

from customer c

join salesman s

on c.salesmanID = s.salesmanID

where (c.cust\_city <> s.city) AND commission > 0.12

Graphical user interface, text, application, email

Description automatically generated

**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 ord\_no, ord\_date, Purch\_amt, cust\_Name as 'Customer Name', grade, name as Salesman, city as 'Salesman city', commission

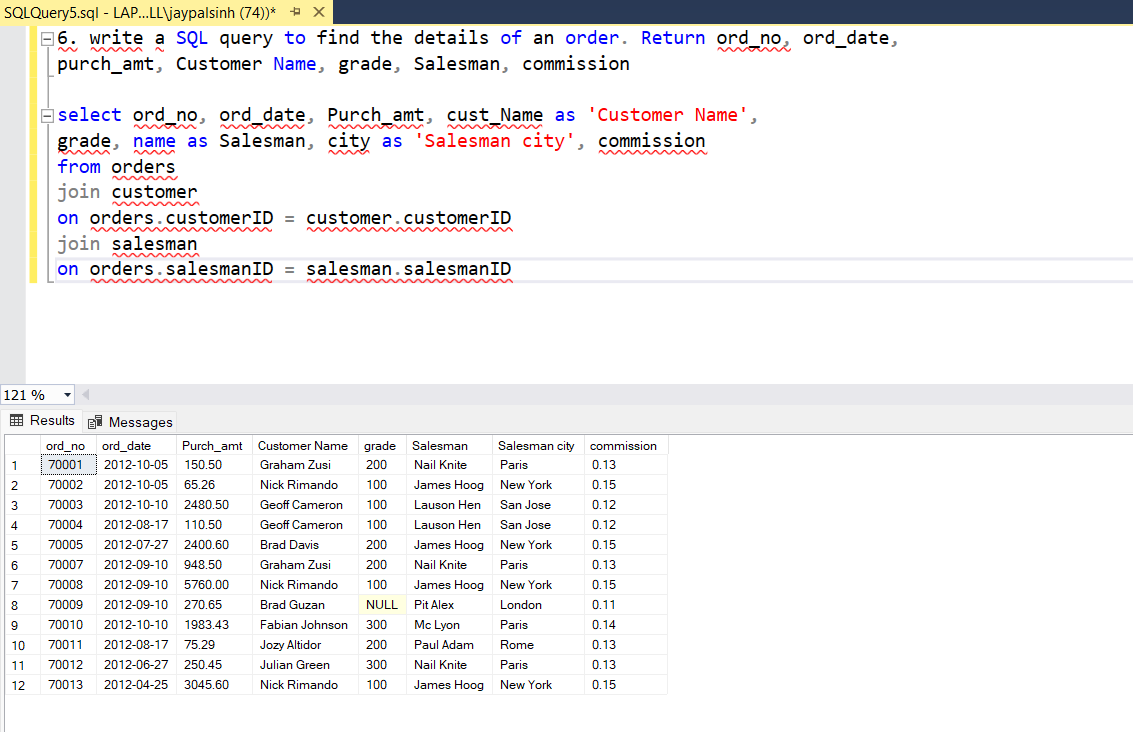
from orders

join customer

on orders.customerID = customer.customerID

join salesman

on orders.salesmanID = salesman.salesmanID



**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 cust\_name as 'customer name', cust\_city as 'customer city', grade, name as salesman, city as 'salesman city'

from customer c

join salesman s

on c.salesmanID = s.salesmanID

order by customerID

Graphical user interface, text, application, email

Description automatically generated

**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 cust\_name, cust\_city as 'customer city', grade, name as Salesman, city as 'salesman city'

from customer c

left join salesman s

on c.salesmanID = s.salesmanID

where grade < 300

order by customerID

Graphical user interface, text, application, email

Description automatically generated

**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 cust\_name as 'customer name', cust\_city, ord\_no, ord\_date, purch\_amt

from customer c

join orders o

on o.customerID = c.customerID

order by ord\_date

Graphical user interface, text, application, email

Description automatically generated

**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 cust\_name, cust\_city, ord\_no, ord\_date, purch\_amt as 'Order Amount', name as 'salesperson name', commission

from customer c

left join orders o

on c.customerID = o.customerID

left join salesman s

on c.salesmanID = s.salesmanID

Graphical user interface, text, application, email

Description automatically generated

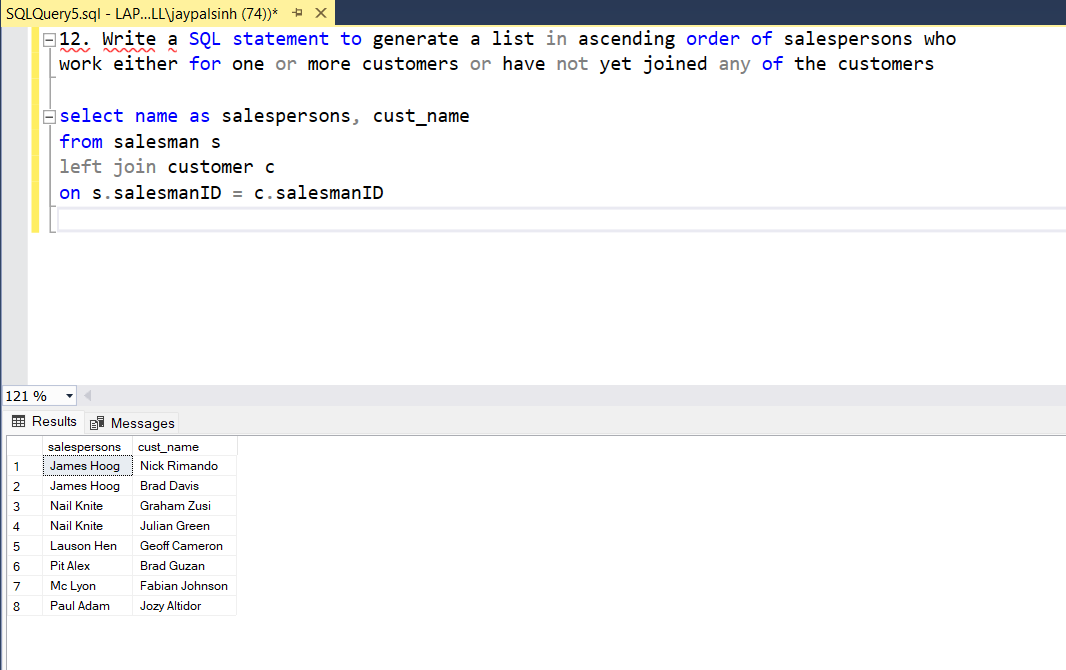
**12**. Write a SQL statement to generate a list in ascending order of salespersons who work either for one or more customers or have not yet joined any of the customers

select name as salespersons, cust\_name

from salesman s

left join customer c

on s.salesmanID = c.salesmanID



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

select name as salesperson, cust\_name as 'customer name', cust\_city , grade, ord\_no, ord\_date, purch\_amt as Amount

from customer c

right join orders o

on o.customerID = c.customerID

left join salesman s

on c.salesmanID = s.salesmanID

Graphical user interface, text, email

Description automatically generated

**14 and 15**. 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 name as salesman, cust\_name, purch\_amt as 'Order Amount', grade

from customer c

right join salesman s

on c.salesmanID = s.salesmanID

left join orders o

on c.customerID = o.customerID

where (purch\_amt >= 2000) AND (grade IS NOT NULL)

Graphical user interface, text, application, email

Description automatically generated

**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 cust\_name , cust\_city, ord\_no, ord\_date, Purch\_amt

from customer c

join orders o

on c.customerID = o.customerID

where grade is not null

Graphical user interface, text, application, email

Description automatically generated

**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

Graphical user interface, table

Description automatically generated

**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.cust\_city

Graphical user interface, text, application

Description automatically generated

**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 = c.cust\_city) AND (c.grade is not null)

Text

Description automatically generated with medium confidence

**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.cust\_city) AND (c.grade is not null)

Graphical user interface, text, application, email

Description automatically generated