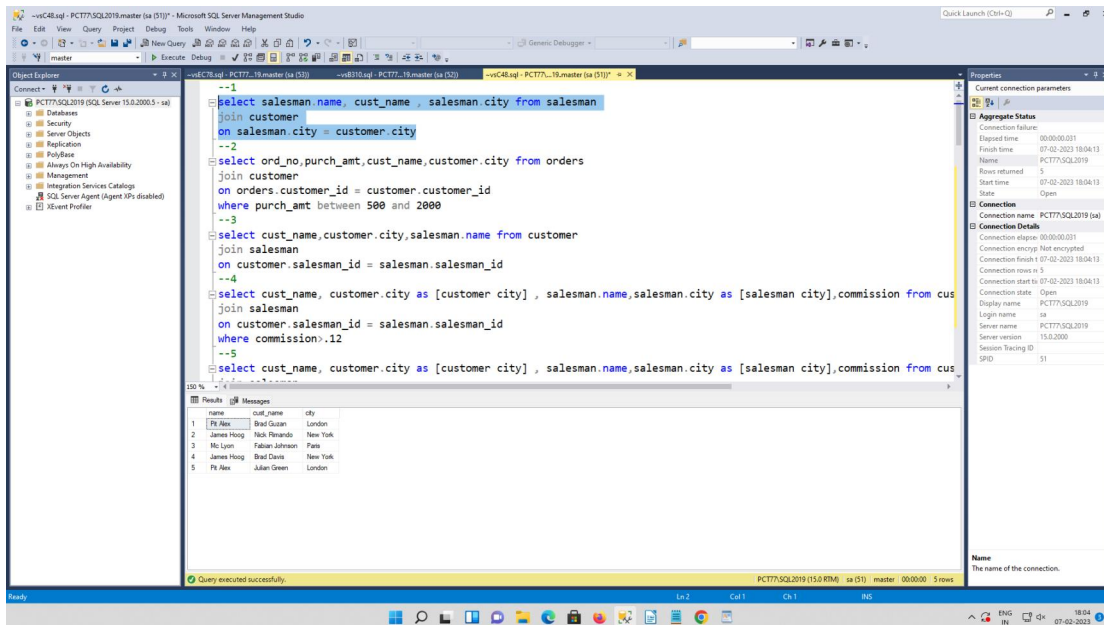


Assignment 2

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

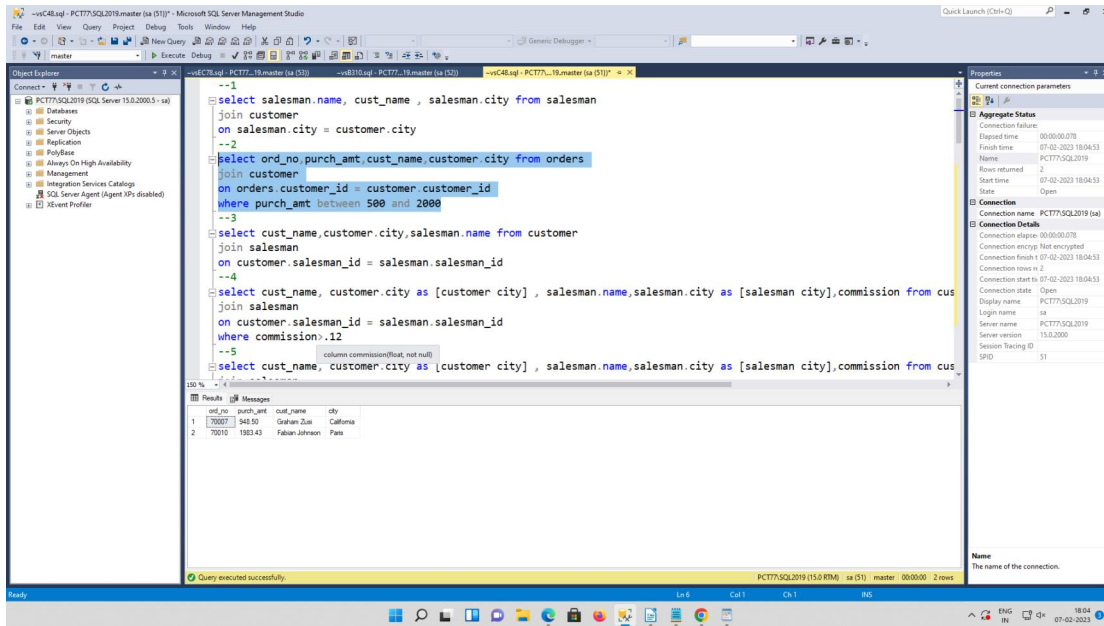


The screenshot displays the Microsoft SQL Server Enterprise Manager interface. The central pane shows a SQL query with five steps, each preceded by a comment. The query is designed to find salespersons and customers who live in the same city. The results pane at the bottom shows a table with three columns: name, cust_name, and city. The table contains five rows of data.

```
--1
select salesman.name, cust_name, salesman.city from salesman
join customer
on salesman.city = customer.city
--2
select ord_no, purch_amt, cust_name, customer.city from orders
join customer
on orders.customer_id = customer.customer_id
where purch_amt between 500 and 2000
--3
select cust_name, customer.city, salesman.name from customer
join salesman
on customer.salesman_id = salesman.salesman_id
--4
select cust_name, customer.city as [customer city], salesman.name, salesman.city as [salesman city], commission from cus
join salesman
on customer.salesman_id = salesman.salesman_id
where commission > .12
--5
select cust_name, customer.city as [customer city], salesman.name, salesman.city as [salesman city], commission from cus
```

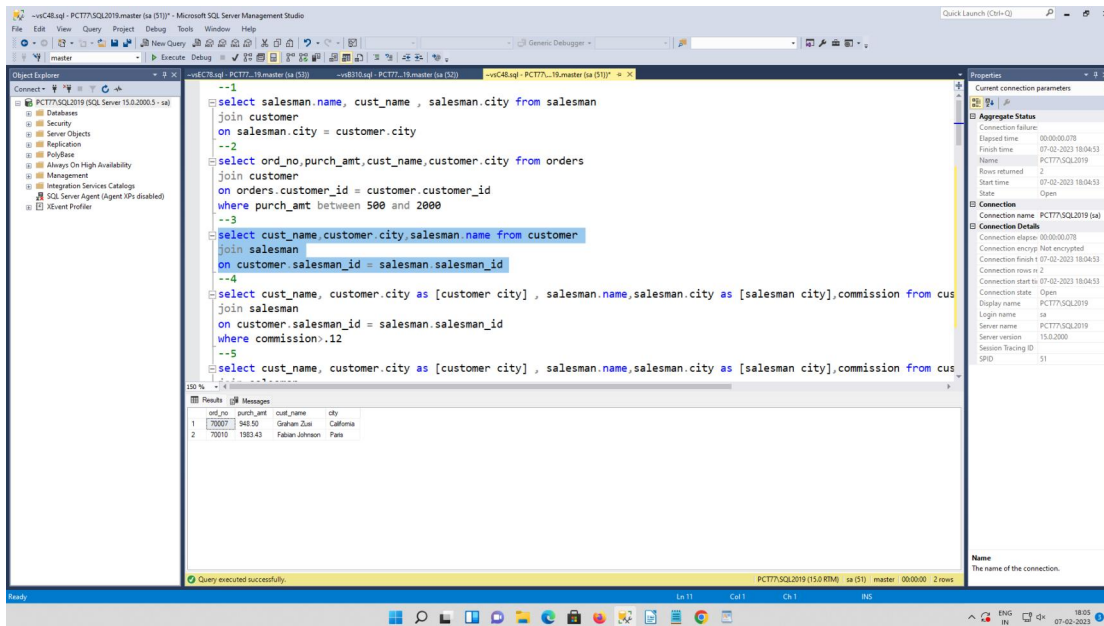
	name	cust_name	city
1	Pat Nes	Brad Gustin	London
2	James Hoog	Mark Richards	New York
3	Mc Lyon	Fabian Johnson	Paris
4	James Hoog	Brad Davis	New York
5	Pat Nes	Julian Green	London

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



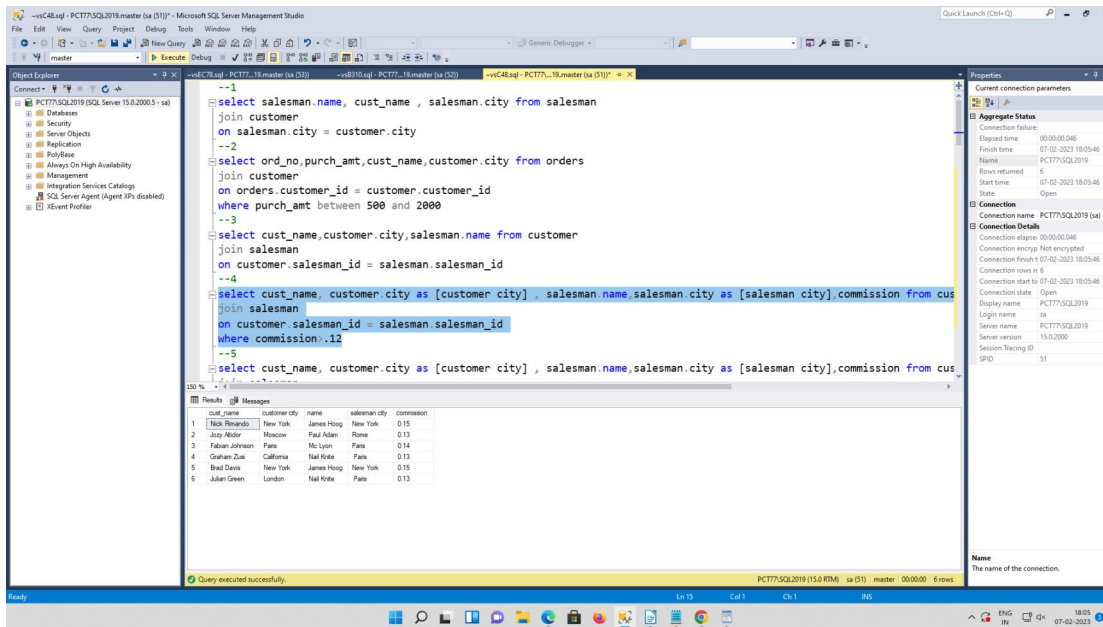
3.write a SQL query to find the salesperson(s) and the customer(s) he represents.

Return Customer Name, city, Salesman, commission

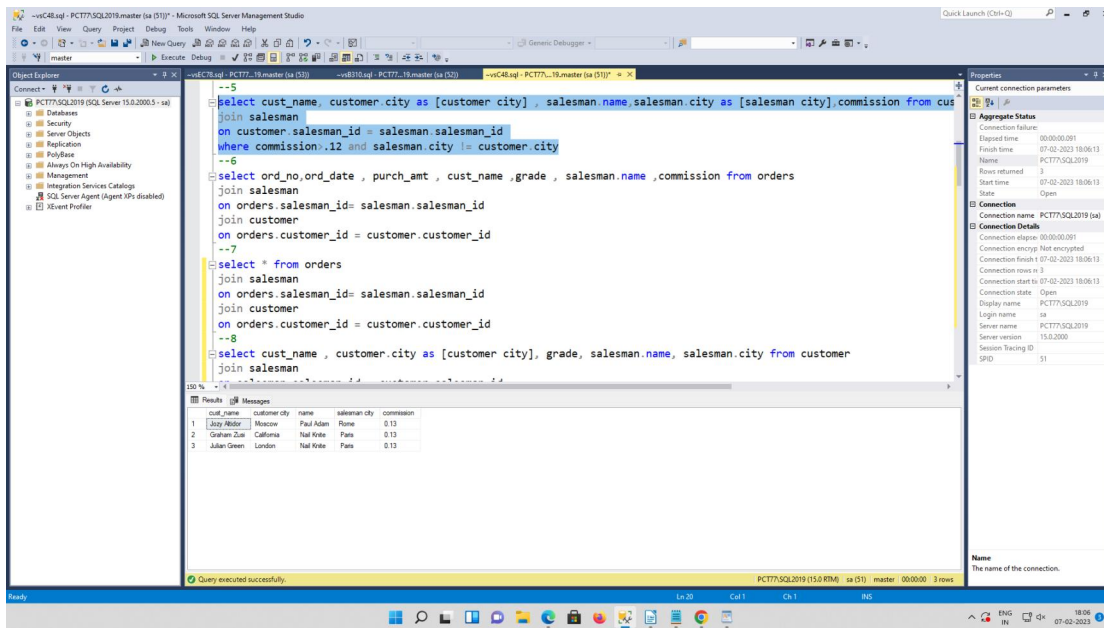


4.write a SQL query to find salespeople who received commissions of more than 12

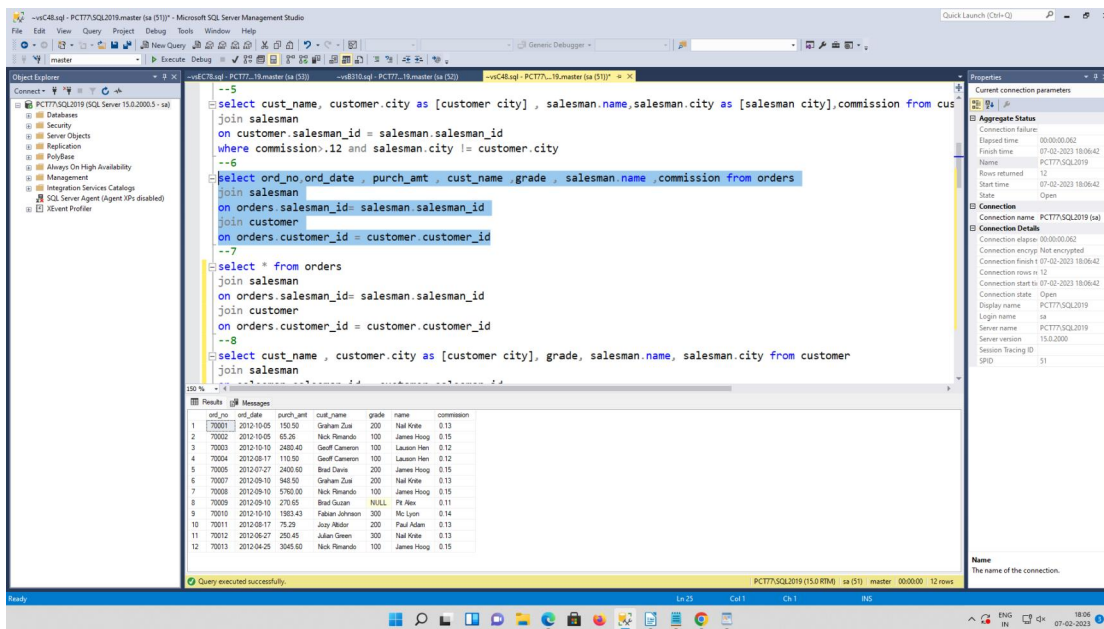
percent from the company. Return Customer Name, customer city, Salesman, commission.



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



6.write a SQL query to find the details of an order.
Return ord_no, ord_date,
purch_amt, Customer Name, grade, Salesman,
commission



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

The screenshot displays the Microsoft SQL Server Enterprise Manager interface. The central pane shows a SQL query that joins the 'customer' and 'orders' tables. The query is as follows:

```
--6
select ord_no,ord_date , purch_amt , cust_name ,grade , salesman_name ,commission from orders
join salesman
on orders.salesman_id= salesman.salesman_id
join customer
on orders.customer_id = customer.customer_id
--7
select * from orders
join salesman
on orders.salesman_id= salesman.salesman_id
join customer
on orders.customer_id = customer.customer_id
--8
select cust_name , customer.city as [customer city], grade, salesman.name, salesman.city from customer
join salesman
on salesman.salesman_id = customer.salesman_id
order by customer_id asc
--9
```

The bottom pane shows the results of the query, which are 12 rows of data. The columns are: ord_no, purch_amt, ord_date, customer_id, salesman_id, salesman_name, city, commission, customer_id, cust_name, city, grade, salesman_id.

ord_no	purch_amt	ord_date	customer_id	salesman_id	salesman_name	city	commission	customer_id	cust_name	city	grade	salesman_id	
70001	150.50	2012-10-05	3005	5002	5002	Paris	0.13	3005	Graham Zue	California	200	5002	
70002	65.26	2012-10-05	3002	5001	5001	James Hong	New York	0.15	3002	Nick Rhonda	New York	100	5001
70003	2400.40	2012-10-10	3009	5003	5003	Lauren Hen	San Jose	0.12	3009	Geoff Cameron	Berlin	100	5003
70004	110.50	2012-09-17	3009	5003	5003	Lauren Hen	San Jose	0.12	3009	Geoff Cameron	Berlin	100	5003
70005	2400.60	2012-07-27	3007	5001	5001	James Hong	New York	0.15	3007	Brad Davis	New York	200	5001
70007	940.50	2012-09-10	3005	5002	5002	Nick Rhonda	Paris	0.13	3005	Graham Zue	California	200	5002
70008	5760.00	2012-09-10	3002	5001	5001	James Hong	New York	0.15	3002	Nick Rhonda	New York	100	5001
70009	270.65	2012-09-10	3001	5005	5005	Pt New	London	0.11	3001	Brad Guzan	London	N/A	5005
70010	1983.43	2012-10-10	3004	5006	5006	Mc Lyon	Paris	0.14	3004	Fabian Johnson	Paris	300	5006
70011	75.29	2012-08-17	3003	5007	5007	Paul Adams	Rome	0.13	3003	Jay Miller	Rome	200	5007
70012	290.45	2012-06-27	3008	5002	5002	Nick Rhonda	Paris	0.13	3008	Julian Green	London	300	5002
70013	3045.60	2012-04-25	3002	5001	5001	James Hong	New York	0.15	3002	Nick Rhonda	New York	100	5001

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.

The screenshot shows the Microsoft SQL Server Enterprise Manager interface. The left pane displays the server hierarchy for 'PCT77-SQL2019 (SQL Server 15.0.2000.5 - sa)'. The central pane contains a SQL query that filters for customers with a grade less than 300 and returns their details along with the salesman's information. The right pane shows the 'Properties' window for the connection, including connection details and status.

```

--6
where commission > 12 and salesman.city != customer.city
--7
select ord_no, ord_date , purch_amt , cust_name , grade , salesman.name , commission from orders
join salesman
on orders.salesman_id = salesman.salesman_id
join customer
on orders.customer_id = customer.customer_id
--8
select * from orders
join salesman
on orders.salesman_id = salesman.salesman_id
join customer
on orders.customer_id = customer.customer_id
--9
select cust_name , customer.city as [customer city] , grade , salesman.name , salesman.city from customer
join salesman
on salesman.salesman_id = customer.salesman_id
order by customer_id asc
--10
select cust_name , customer.city as [customer city] , grade , salesman.name , salesman.city as [salesman city] from customer
join salesman
on salesman.salesman_id = customer.salesman_id
order by customer_id asc

```

cust_name	customer city	grade	name	city
Brad Davis	London	100	Paul Allen	London
Nick Reynolds	New York	100	James Hoag	New York
Julay Alder	Moscow	200	Paul Allen	Rome
Fabian Johnson	Paris	300	Mc Lyon	Paris
Graban Zue	California	200	Nal Krite	Paris
Brad Davis	New York	200	James Hoag	New York
Julay Alder	London	300	Nal Krite	Paris
Geoff Cammon	Berlin	100	Lauren Hen	San Jose

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.

The screenshot shows the Microsoft SQL Server Enterprise Manager interface. The left pane displays the server hierarchy for 'PCT77-SQL2019 (SQL Server 15.0.2000.5 - sa)'. The central pane contains a SQL query that filters for customers with a grade less than 300 and returns their details along with the salesman's information. The right pane shows the 'Properties' window for the connection, including connection details and status.

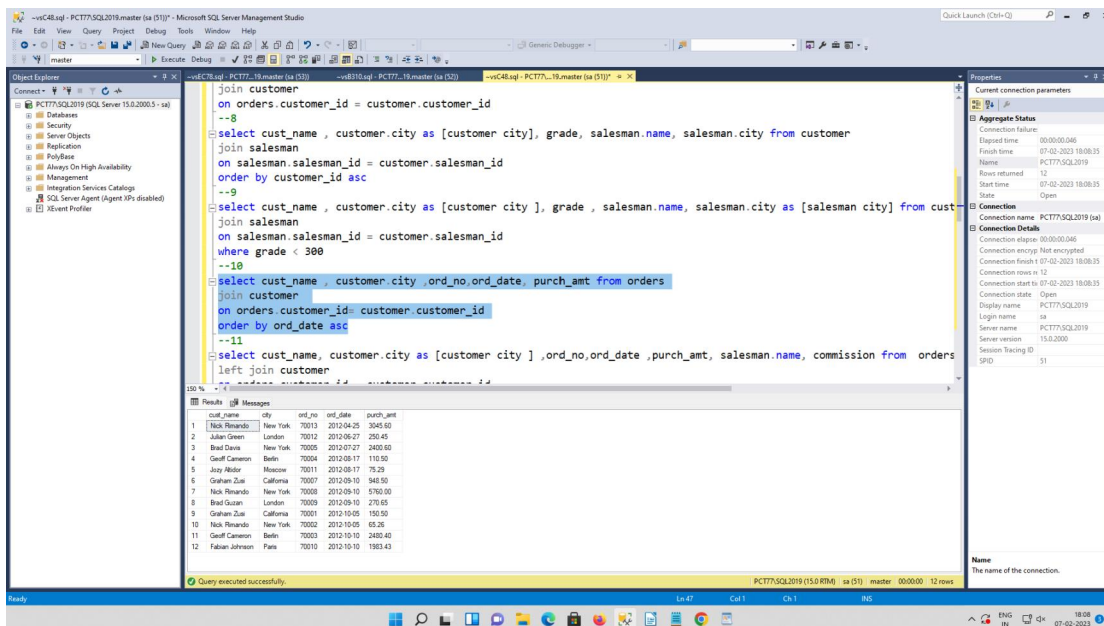
```

--8
select cust_name , customer.city as [customer city] , grade , salesman.name , salesman.city from customer
join salesman
on salesman.salesman_id = customer.salesman_id
order by customer_id asc
--9
select cust_name , customer.city as [customer city] , grade , salesman.name , salesman.city as [salesman city] from customer
join salesman
on salesman.salesman_id = customer.salesman_id
where grade < 300
--10
select cust_name , customer.city , ord_no, ord_date , purch_amt from orders
join customer
on orders.customer_id = customer.customer_id
order by ord_date asc
--11
select cust_name , customer.city as [customer city] , ord_no, ord_date , purch_amt , salesman.name , commission from orders
left join customer
on orders.customer_id = customer.customer_id
order by customer_id asc

```

cust_name	customer city	grade	name	salesman city
Nick Reynolds	New York	100	James Hoag	New York
Julay Alder	Moscow	200	Paul Allen	Rome
Graban Zue	California	200	Nal Krite	Paris
Brad Davis	New York	200	James Hoag	New York
Geoff Cammon	Berlin	100	Lauren Hen	San Jose

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



The screenshot shows the Microsoft SQL Server Enterprise Manager interface. The central pane displays a SQL query that joins the 'customer' and 'orders' tables. The query selects customer details and order information, sorted by order date in ascending order. The results pane at the bottom shows a table with 12 rows of data.

```

--8
select cust_name , customer.city as [customer city], grade, salesman.name, salesman.city from customer
join salesman
on salesman.salesman_id = customer.salesman_id
order by customer_id asc
--9
select cust_name , customer.city as [customer city] , grade , salesman.name, salesman.city as [salesman city] from customer
join salesman
on salesman.salesman_id = customer.salesman_id
where grade < 300
--10
select cust_name , customer.city , ord_no,ord_date , purch_amt from orders
join customer
on orders.customer_id= customer.customer_id
order by ord_date asc
--11
select cust_name , customer.city as [customer city] ,ord_no,ord_date , purch_amt, salesman.name, commission from orders
left join customer
on orders.customer_id = customer.customer_id

```

	cust_name	city	ord_no	ord_date	purch_amt
1	Nick Renault	New York	70013	2012-04-25	3045.60
2	Julian Green	London	70012	2012-06-27	250.45
3	Brad Davis	New York	70005	2012-07-27	2400.60
4	Goeff Camenson	Berlin	70004	2012-08-17	110.50
5	Jayy Akbar	Moscow	70011	2012-08-17	75.29
6	Carlson Zue	California	70007	2012-09-10	948.50
7	Nick Renault	New York	70008	2012-09-10	5760.00
8	Brad Guzan	London	70009	2012-09-10	270.65
9	Carlson Zue	California	70001	2012-10-05	150.50
10	Nick Renault	New York	70002	2012-10-05	65.26
11	Goeff Camenson	Berlin	70003	2012-10-10	2480.40
12	Fabian Johnson	Paris	70010	2012-10-10	1953.43

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

The screenshot shows a SQL query in the Query Editor window of Microsoft SQL Server Enterprise Manager. The query is as follows:

```

--10
join salesman
on salesman.salesman_id = customer.salesman_id
where grade < 300
--11
select cust_name , customer.city ,ord_no,ord_date , purch_amt from orders
join customer
on orders.customer_id= customer.customer_id
order by ord_date asc
--12
select cust_name, customer.city as [customer city ] ,ord_no,ord_date , purch_amt, salesman.name, commission from orders
left join customer
on orders.customer_id = customer.customer_id
right join salesman
on orders.salesman_id = salesman.salesman_id
--13
select name , cust_name from salesman
left outer join customer
on customer.salesman_id = salesman.salesman_id
order by salesman.salesman_id
--14

```

The Results window displays the following data:

	cust_name	customer.city	ord_no	ord_date	purch_amt	name	commission
1	Nick Rimando	New York	70002	2012-10-05	65.25	James Hong	0.15
2	Brad Davis	New York	70005	2012-07-27	2400.00	James Hong	0.15
3	Nick Rimando	New York	70008	2012-09-10	5700.00	James Hong	0.15
4	Nick Rimando	New York	70013	2012-04-25	3045.00	James Hong	0.15
5	Graham Zui	California	70001	2012-10-05	100.50	Nat Klie	0.13
6	Graham Zui	California	70007	2012-05-10	940.50	Nat Klie	0.13
7	Julian Green	London	70012	2012-06-27	250.45	Nat Klie	0.13
8	Geoff Cameron	Berlin	70003	2012-10-10	2400.40	Lauson Hen	0.12
9	Geoff Cameron	Berlin	70004	2012-08-17	110.50	Lauson Hen	0.12
10	Brad Guan	London	70009	2012-05-10	270.05	PK Alex	0.11
11	Fabian Johnson	Paris	70010	2012-10-10	1803.43	Mc Lyon	0.14
12	Joey Abdo	Moscow	70011	2012-08-17	75.29	Paul Adam	0.13

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

The screenshot shows a SQL query in the Query Editor window of Microsoft SQL Server Enterprise Manager. The query is as follows:

```

--10
join salesman
on salesman.salesman_id = customer.salesman_id
where grade < 300
--11
select cust_name , customer.city ,ord_no,ord_date , purch_amt from orders
join customer
on orders.customer_id= customer.customer_id
order by ord_date asc
--12
select cust_name, customer.city as [customer city ] ,ord_no,ord_date , purch_amt, salesman.name, commission from orders
left join customer
on orders.customer_id = customer.customer_id
right join salesman
on orders.salesman_id = salesman.salesman_id
--13
select name , cust_name from salesman
left outer join customer
on customer.salesman_id = salesman.salesman_id
order by salesman.salesman_id
--14

```

The Results window displays the following data:

	name	cust_name
1	James Hong	Nick Rimando
2	James Hong	Brad Davis
3	Nat Klie	Graham Zui
4	Nat Klie	Julian Green
5	Lauson Hen	Geoff Cameron
6	PK Alex	Brad Guan
7	Mc Lyon	Fabian Johnson
8	Paul Adam	Joey Abdo

13.write a SQL query to list all salespersons along

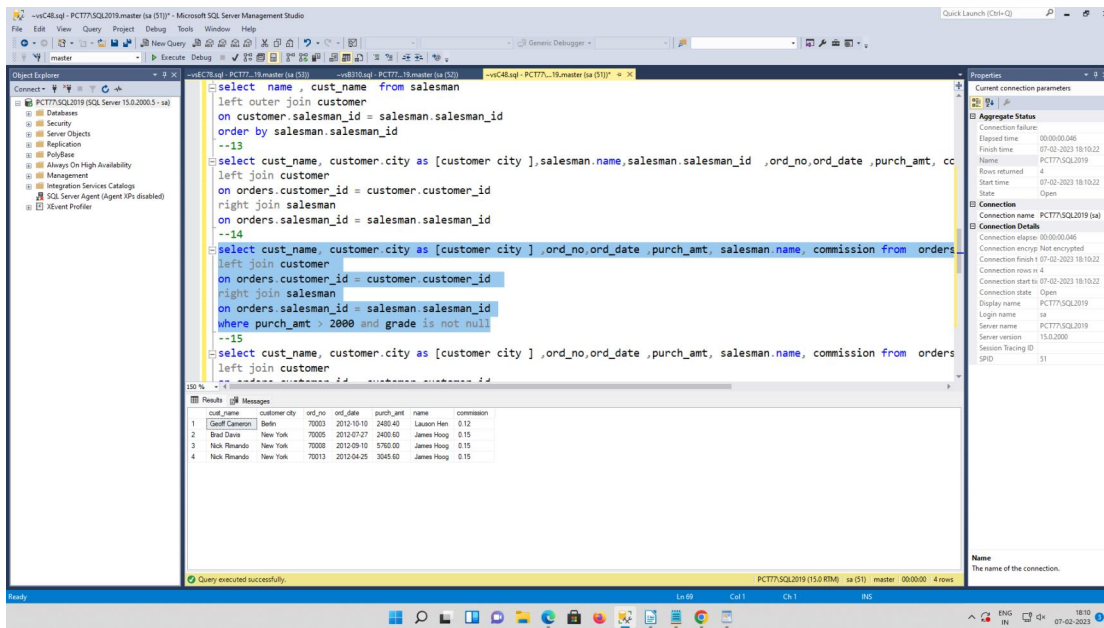
with customer name, city, grade,
order number, date, and amount

The screenshot displays the Microsoft SQL Server Enterprise Manager interface. The central pane shows a SQL query with several joins and a select statement. The results pane at the bottom shows a table with 12 rows and 8 columns. The columns are: cust_name, customer_city, name, salesman_id, ord_no, ord_date, purch_amt, and commission. The results list 12 salesmen with their respective details.

```
select cust_name, customer.city as [customer city] ,ord_no,ord_date ,purch_amt, salesman.name, commission from orders
left join customer
on orders.customer_id = customer.customer_id
right join salesman
on orders.salesman_id = salesman.salesman_id
--12
select name , cust_name from salesman
left outer join customer
on customer.salesman_id = salesman.salesman_id
order by salesman.salesman_id
--13
select cust_name, customer.city as [customer city] ,salesman.name,salesman.salesman_id ,ord_no,ord_date ,purch_amt, co
left join customer
on orders.customer_id = customer.customer_id
right join salesman
on orders.salesman_id = salesman.salesman_id
--14
select cust_name, customer.city as [customer city] ,ord_no,ord_date ,purch_amt, salesman.name, commission from orders
left join customer
on orders.customer_id = customer.customer_id
```

	cust_name	customer_city	name	salesman_id	ord_no	ord_date	purch_amt	commission
1	Nick Rendano	New York	James Hong	5001	70002	2012-10-05	65.26	0.15
2	Brad Davis	New York	James Hong	5001	70005	2012-07-27	2402.60	0.15
3	Nick Rendano	New York	James Hong	5001	70008	2012-09-10	1760.00	0.15
4	Nick Rendano	New York	James Hong	5001	70013	2012-04-25	3045.60	0.15
5	Graham Zue	California	Nail Krole	5002	70001	2012-10-05	192.50	0.13
6	Graham Zue	California	Nail Krole	5002	70007	2012-09-10	948.50	0.13
7	Julian Green	London	Nail Krole	5002	70012	2012-06-27	250.45	0.13
8	Geoff Cameron	Berlin	Lauson Hen	5003	70003	2012-10-10	2480.40	0.12
9	Geoff Cameron	Berlin	Lauson Hen	5003	70004	2012-08-17	110.50	0.12
10	Brad Guzan	London	Pt Alex	5005	70009	2012-09-10	270.65	0.11
11	Fabian Johnson	Paris	Mc Lorn	5006	70010	2012-10-10	1983.43	0.14
12	Jozey Alder	Moscow	Paul Adan	5007	70011	2012-08-17	75.29	0.13

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



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.

The screenshot displays the Microsoft SQL Server Enterprise Manager interface. The central pane shows a SQL query being executed. The query is as follows:

```
--14
left join customer
on orders.customer_id = customer.customer_id
right join salesman
on orders.salesman_id = salesman.salesman_id
where purch_amt > 2000 and grade is not null
--15
select cust_name, customer.city as [customer city ], ord_no, ord_date, purch_amt, salesman name, commission from orders
left join customer
on orders.customer_id = customer.customer_id
right join salesman
on orders.salesman_id = salesman.salesman_id
where purch_amt > 2000 and grade is not null
--16
select cust_name, customer.city, ord_no, ord_date, purch_amt from orders
right outer join customer
on customer.customer_id=orders.customer_id
--17
select * from salesman
cross join customer
--18
select * from salesman
```

The bottom pane shows the results of the query, which is a table with 4 rows and 7 columns. The columns are: cust_name, customer city, ord_no, ord_date, purch_amt, salesman name, and commission. The data is as follows:

cust_name	customer city	ord_no	ord_date	purch_amt	salesman name	commission
Paula Cameron	Belle	70003	2012-10-10	2400.40	James Hong	0.12
Brad Davis	New York	70005	2012-07-27	2400.60	James Hong	0.15
Nick Ricardo	New York	70009	2012-09-10	5760.00	James Hong	0.15
Nick Ricardo	New York	70013	2012-04-25	3045.60	James Hong	0.15

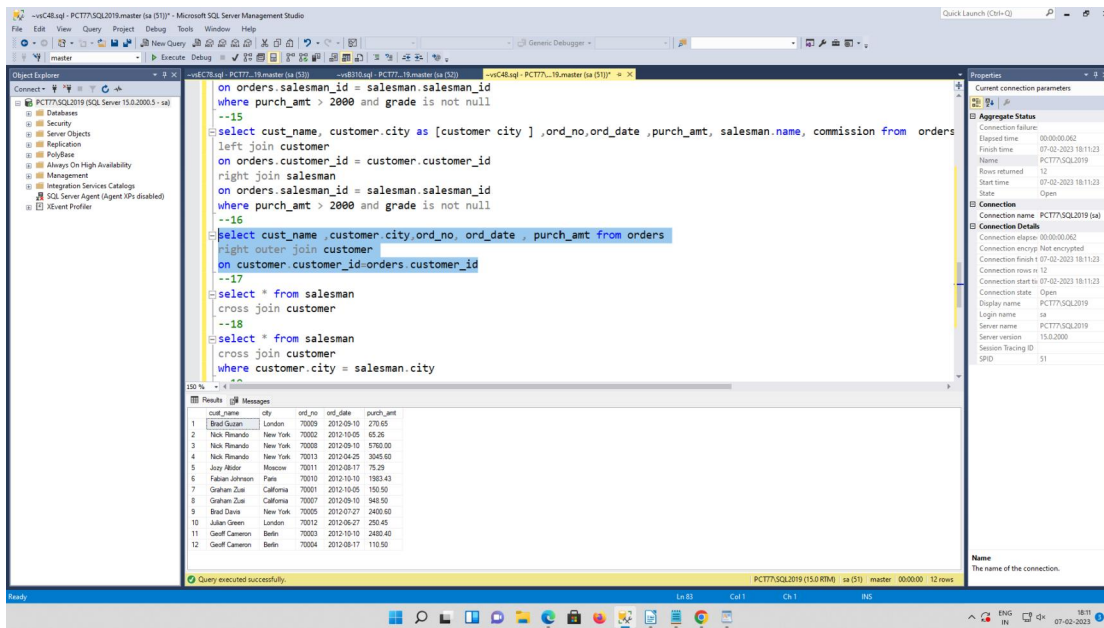
The right pane shows the connection parameters for the current connection, including the connection name, connection string, and connection details.

16. Write a SQL statement to generate a report with the customer name, city, order no.

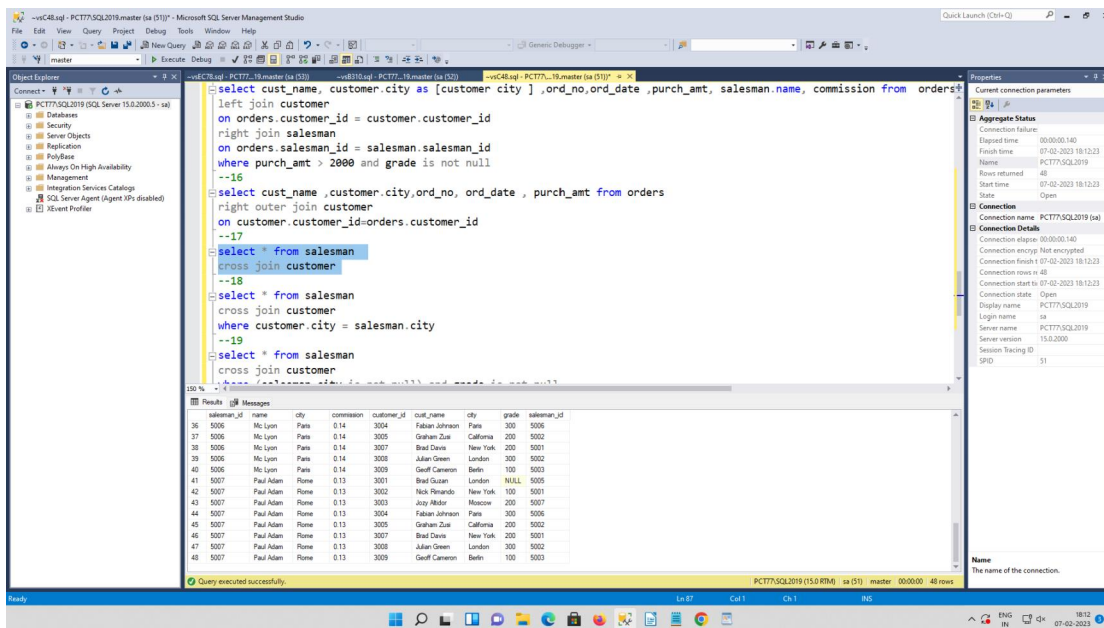
order date, purchase amount for only those customers on the list who must have a

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



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



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

The screenshot displays the Microsoft SQL Server Enterprise Manager interface. The central pane shows a SQL query being executed on the 'master' database. The query is as follows:

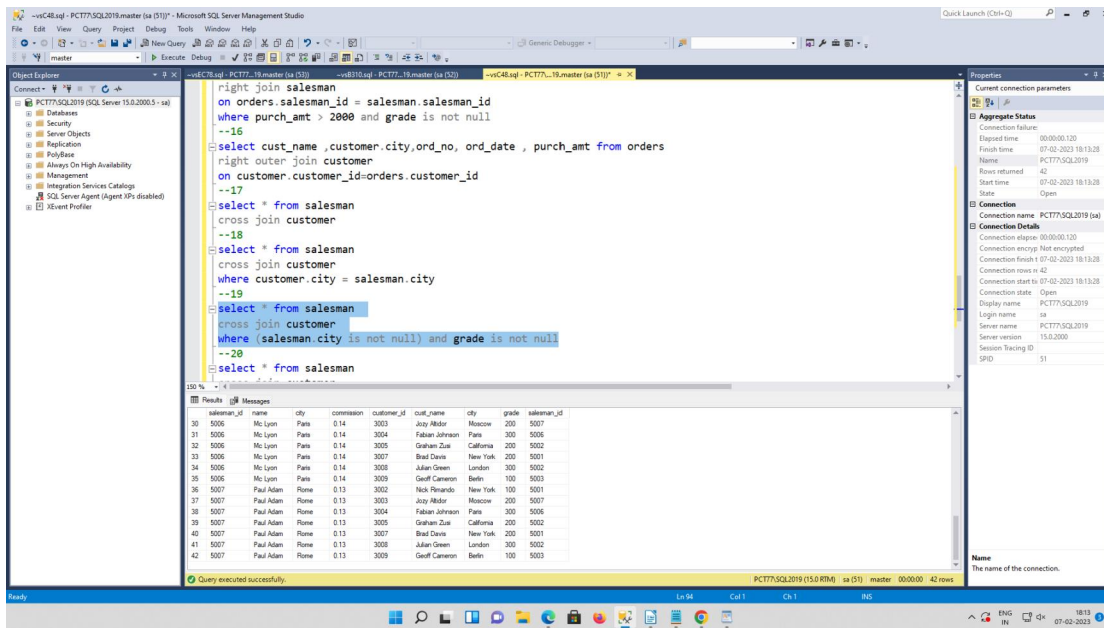
```
--15
right join salesman
on orders.salesman_id = salesman.salesman_id
where purch_amt > 2000 and grade is not null
--16
select cust_name ,customer.city,ord_no, ord_date , purch_amt from orders
right outer join customer
on customer.customer_id=orders.customer_id
--17
select * from salesman
cross join customer
--18
select * from salesman
cross join customer
where customer.city = salesman.city
--19
select * from salesman
cross join customer
where (salesman.city is not null) and grade is not null
--20
select * from salesman
```

The bottom pane shows the results of the query, which is a Cartesian product of salesmen and customers. The results are as follows:

salesman_id	name	city	commission	customer_id	cust_name	city	grade	salesman_id
5005	Pt. Alex	London	0.11	3001	Bad Guan	London	NULL	5005
5001	James Hoog	New York	0.15	3002	Nick Reynolds	New York	100	5001
5005	Mc Lynn	Pitts	0.14	3004	Falken Johnson	Pitts	200	5005
5001	James Hoog	New York	0.15	3007	Bad Davis	New York	200	5001
5005	Pt. Alex	London	0.11	3008	Julian Green	London	300	5002

The right pane shows the 'Properties' window for the connection, displaying details such as 'Connection name', 'Connection details', and 'Connection state'.

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



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

Microsoft SQL Server Enterprise Edition (64-bit) - Microsoft SQL Server Management Studio

File Edit View Query Project Debug Tools Window Help

Object Explorer

- Server
- Security
- Server Objects
- Replication
- PolyBase
- Always On-High Availability
- Management
- Integration Services Catalogs
- SQL Server Agent (Agent XPs disabled)
- XEvent Profiler

Query Editor

```
--18 cross join customer
--19
select * from salesman
cross join customer
where customer.city = salesman.city
--19
select * from salesman
cross join customer
where (salesman.city is not null) and grade is not null
--20
select * from salesman
cross join customer
where customer.city != salesman.city and grade is not null
```

Results

salesman_id	name	city	commission	customer_id	cust_name	city	grade	salesman_id	
26	5006	Mc Lyon	Paris	0.14	3002	Nick Reynolds	New York	100	5001
27	5006	Mc Lyon	Paris	0.14	3003	Joy Abitor	Moscow	200	5007
28	5006	Mc Lyon	Paris	0.14	3005	Graham Zusi	California	200	5002
29	5006	Mc Lyon	Paris	0.14	3007	Brad Davis	New York	200	5001
30	5006	Mc Lyon	Paris	0.14	3008	Julian Green	London	300	5002
31	5006	Mc Lyon	Paris	0.14	3009	Geoff Cameron	Berlin	100	5003
32	5007	Paul Adam	Rome	0.13	3002	Nick Reynolds	New York	100	5001
33	5007	Paul Adam	Rome	0.13	3003	Joy Abitor	Moscow	200	5007
34	5007	Paul Adam	Rome	0.13	3004	Fabian Johnson	Paris	300	5006
35	5007	Paul Adam	Rome	0.13	3005	Graham Zusi	California	200	5002
36	5007	Paul Adam	Rome	0.13	3007	Brad Davis	New York	200	5001
37	5007	Paul Adam	Rome	0.13	3008	Julian Green	London	300	5002
38	5007	Paul Adam	Rome	0.13	3009	Geoff Cameron	Berlin	100	5003

Messages

Query executed successfully.

Properties

Current connection parameters

Aggregate Status

Connection failure:

Elapsed time: 00:00:00.109

Finish time: 07-02-2023 18:13:55

Name: PCT77-SQL2019

Rows returned: 38

Start time: 07-02-2023 18:13:54

State: Open

Connection

Connection name: PCT77-SQL2019 (sa)

Connection Details

Connection elapsed: 00:00:00.109

Connection encrypted: Not encrypted

Connection finish: 07-02-2023 18:13:55

Connection rows: 38

Connection start: 07-02-2023 18:13:54

Connection state: Open

Display name: PCT77-SQL2019

Login name: sa

Server name: PCT77-SQL2019

Server version: 15.0.2000

Session tracing ID: 31

Name: The name of the connection.

Ready

SQL Server 15.0 (RTM) - sa (51) - master - 00:00:00 - 38 rows

1813 07-02-2023

