**Name:** Roohma Kadri

=============================================================

**ASSIGNMENT 02**

**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

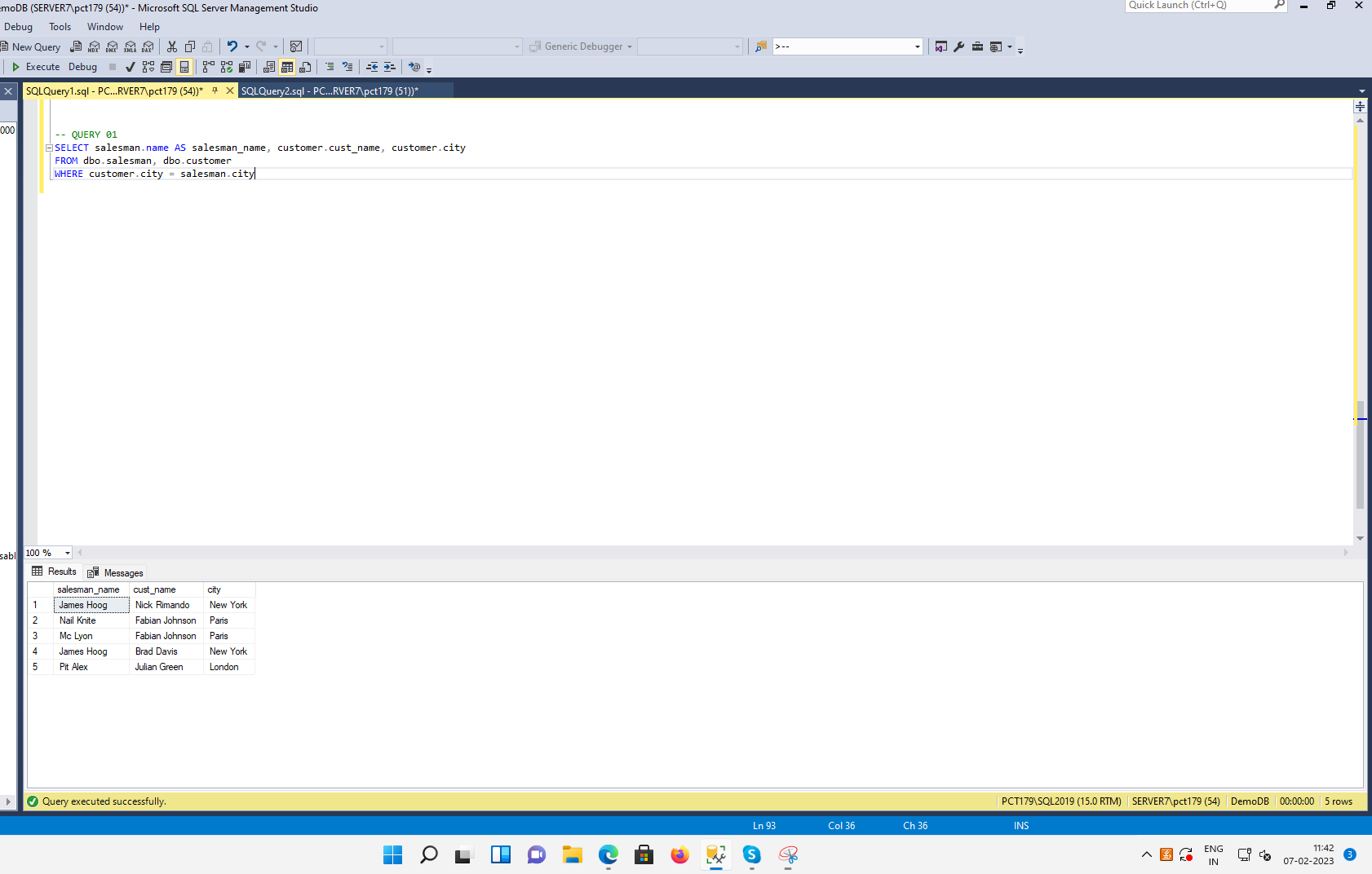
=============================================================

**QUERY 01:** Write a SQL query to find the salesperson and customer who reside in the same city. Return Salesman, cust\_name and city

SELECT salesman.name AS salesman\_name, customer.cust\_name, customer.city

FROM dbo.salesman, dbo.customer

WHERE customer.city = salesman.city



=============================================================

**QUERY 02:** 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

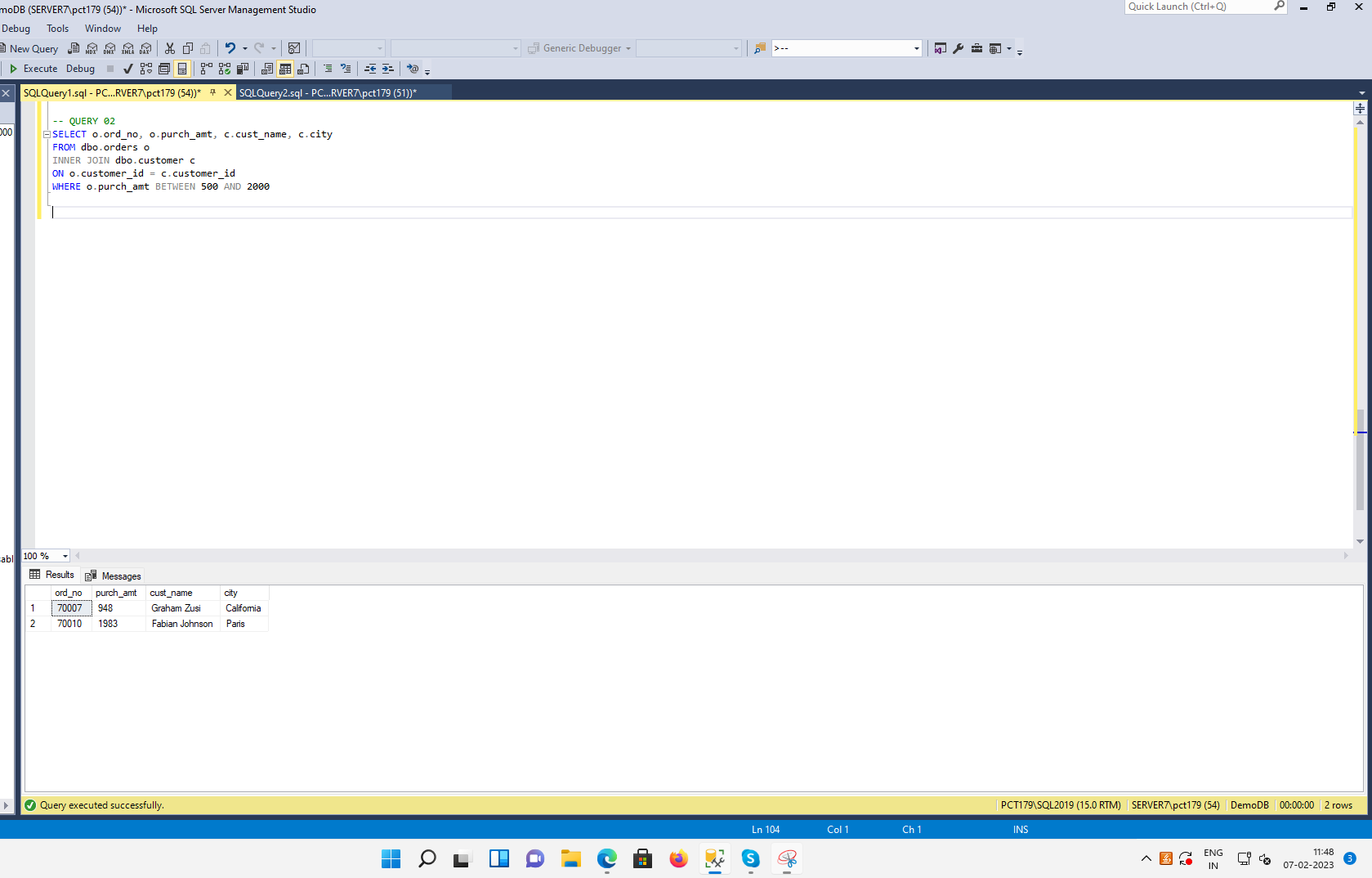
SELECT o.ord\_no, o.purch\_amt, c.cust\_name, c.city

FROM dbo.orders o

INNER JOIN dbo.customer c

ON o.customer\_id = c.customer\_id

WHERE o.purch\_amt BETWEEN 500 AND 2000



=============================================================

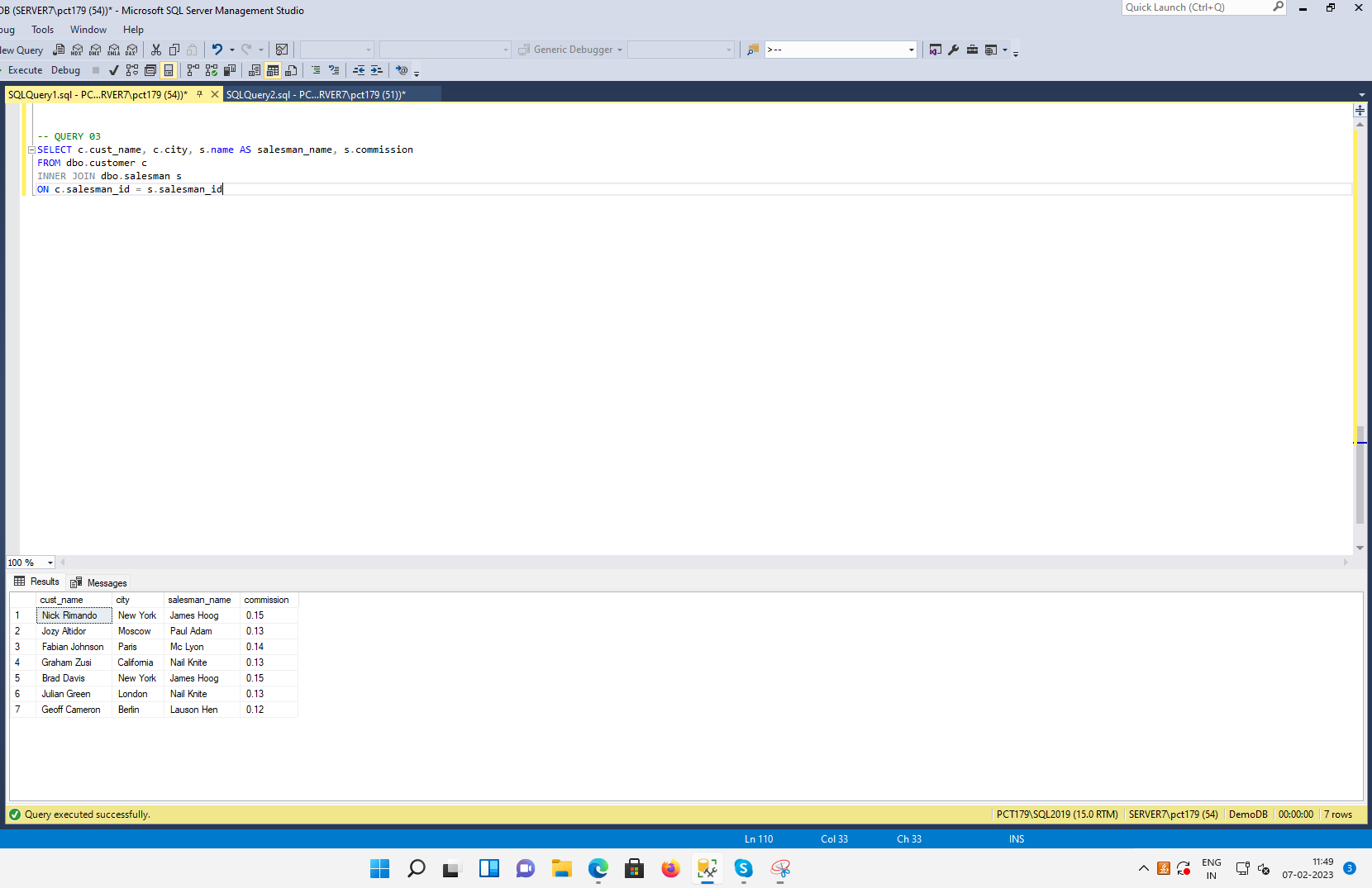
**QUERY 03:** Write a SQL query to find the salesperson(s) and the customer(s) he represents. Return Customer Name, city, Salesman, commission

SELECT c.cust\_name, c.city, s.name AS salesman\_name, s.commission

FROM dbo.customer c

INNER JOIN dbo.salesman s

ON c.salesman\_id = s.salesman\_id



=============================================================

**QUERY 04:** 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.

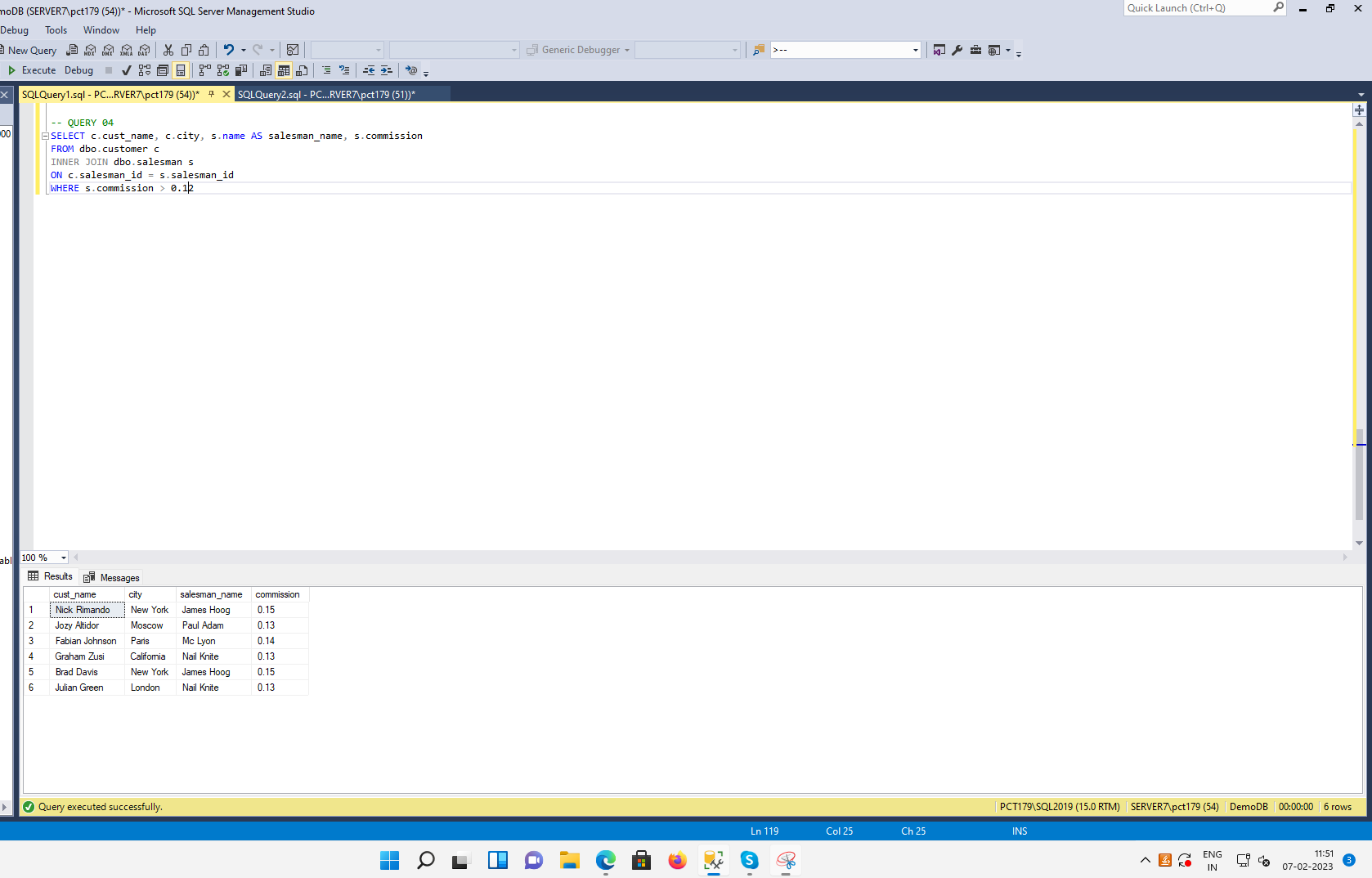
SELECT c.cust\_name, c.city, s.name AS salesman\_name, s.commission

FROM dbo.customer c

INNER JOIN dbo.salesman s

ON c.salesman\_id = s.salesman\_id

WHERE s.commission > 0.12



=============================================================

**QUERY 05:** 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

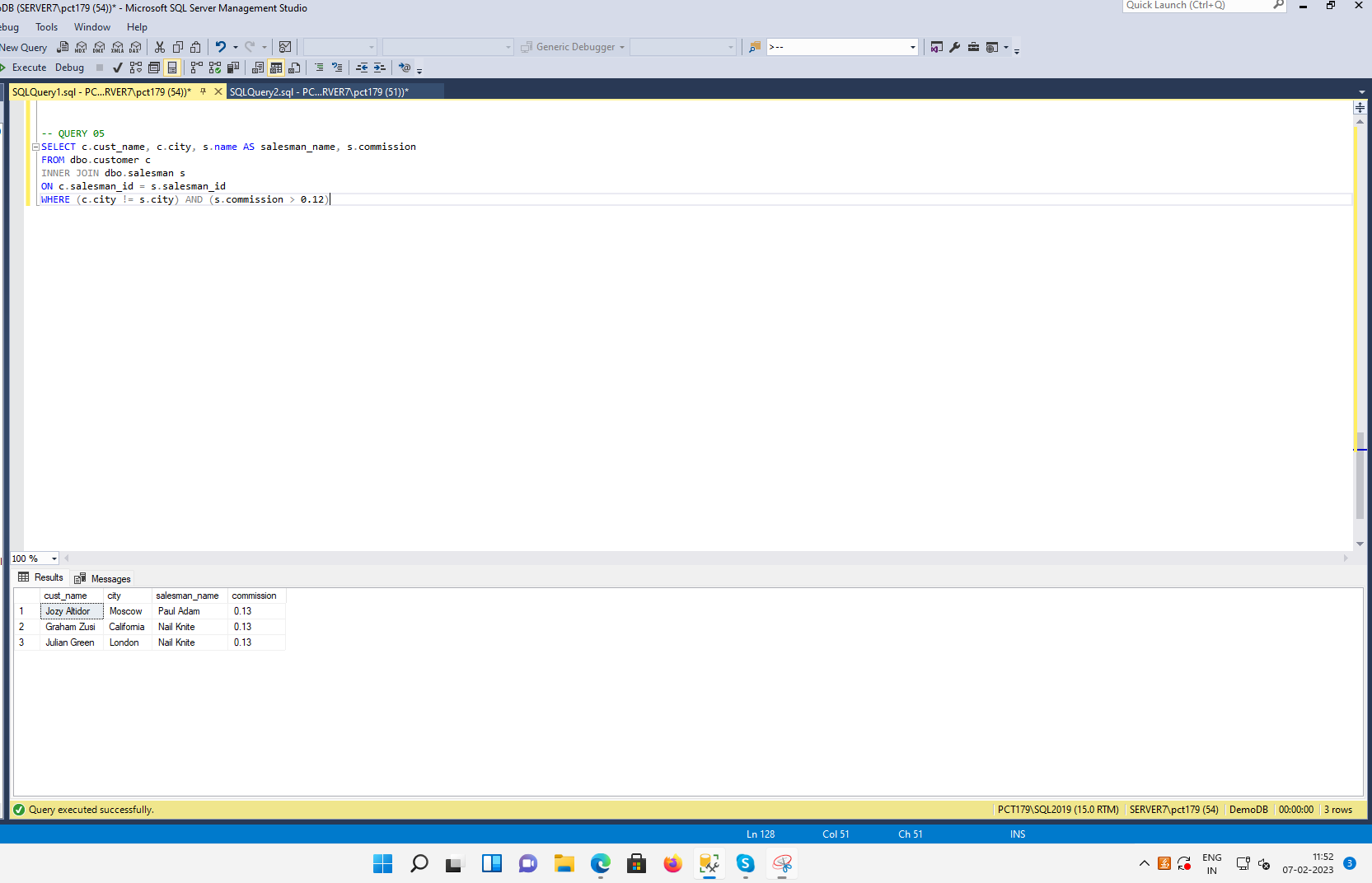
SELECT c.cust\_name, c.city, s.name AS salesman\_name, s.commission

FROM dbo.customer c

INNER JOIN dbo.salesman s

ON c.salesman\_id = s.salesman\_id

WHERE (c.city != s.city) AND (s.commission > 0.12)



=============================================================

**QUERY 06:** 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 AS salesman\_name, s.commission

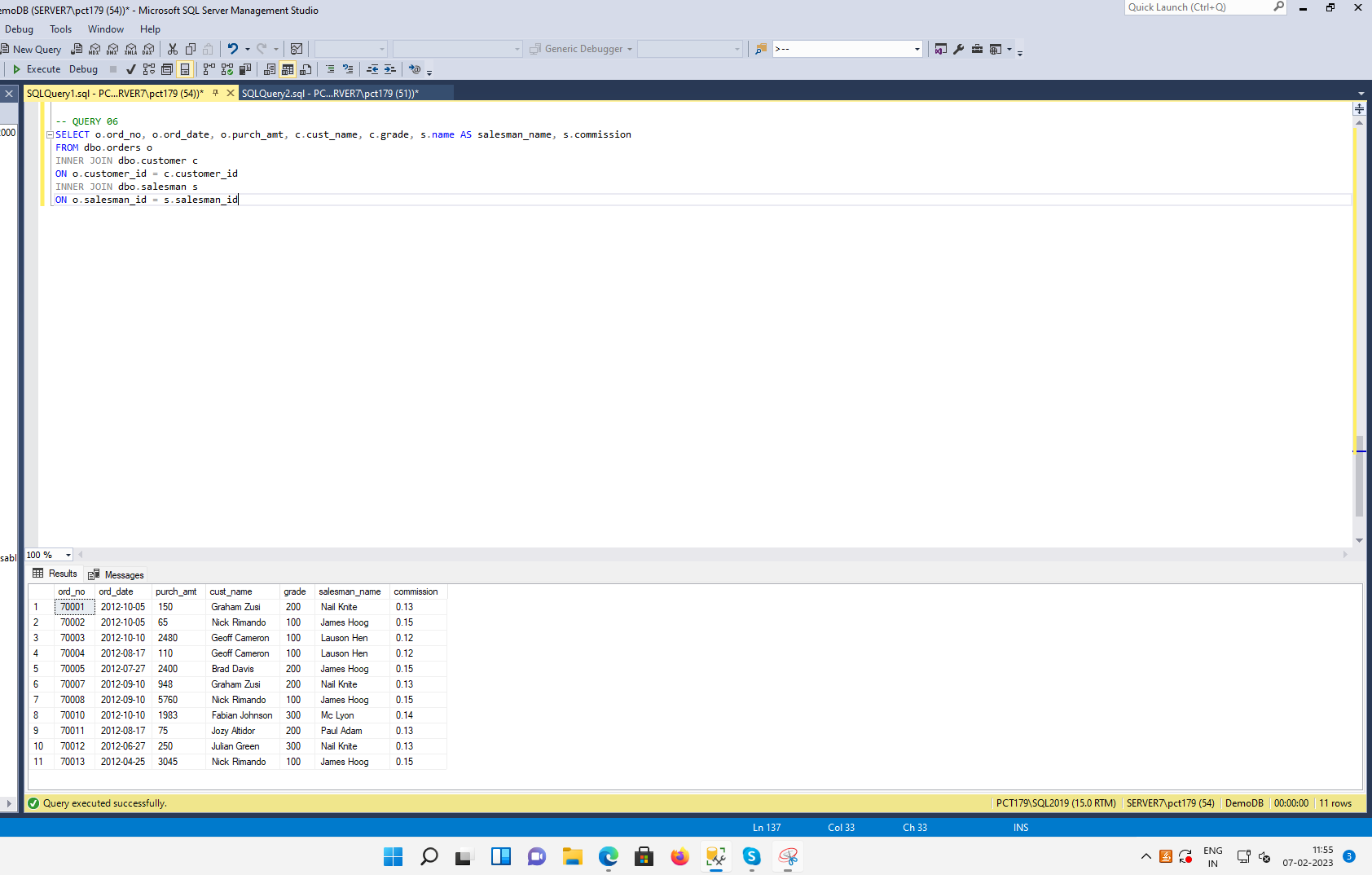
FROM dbo.orders o

INNER JOIN dbo.customer c

ON o.customer\_id = c.customer\_id

INNER JOIN dbo.salesman s

ON o.salesman\_id = s.salesman\_id



=============================================================

**QUERY 07:** 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 \*

FROM orders

NATURAL JOIN customer

NATURAL JOIN salesman

=============================================================

**QUERY 08:** Write a SQL query to display the customer name, customer city, grade, salesman, salesman city. The results should be sorted by ascending customer\_id.

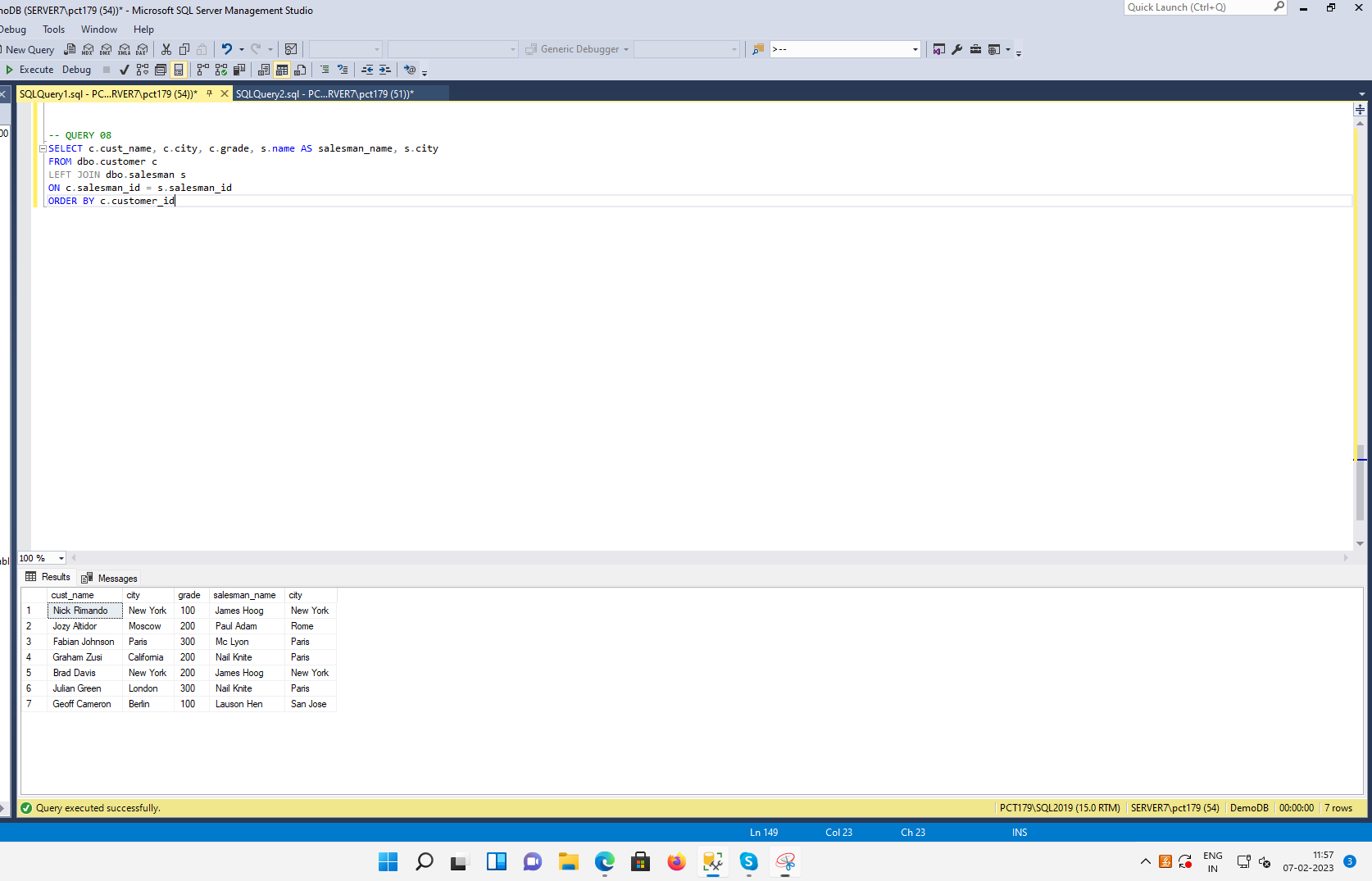
SELECT c.cust\_name, c.city, c.grade, s.name AS salesman\_name, s.city

FROM dbo.customer c

LEFT JOIN dbo.salesman s

ON c.salesman\_id = s.salesman\_id

ORDER BY c.customer\_id



=============================================================

**QUERY 09:** 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, c.city, c.grade, s.name AS salesman\_name, s.city

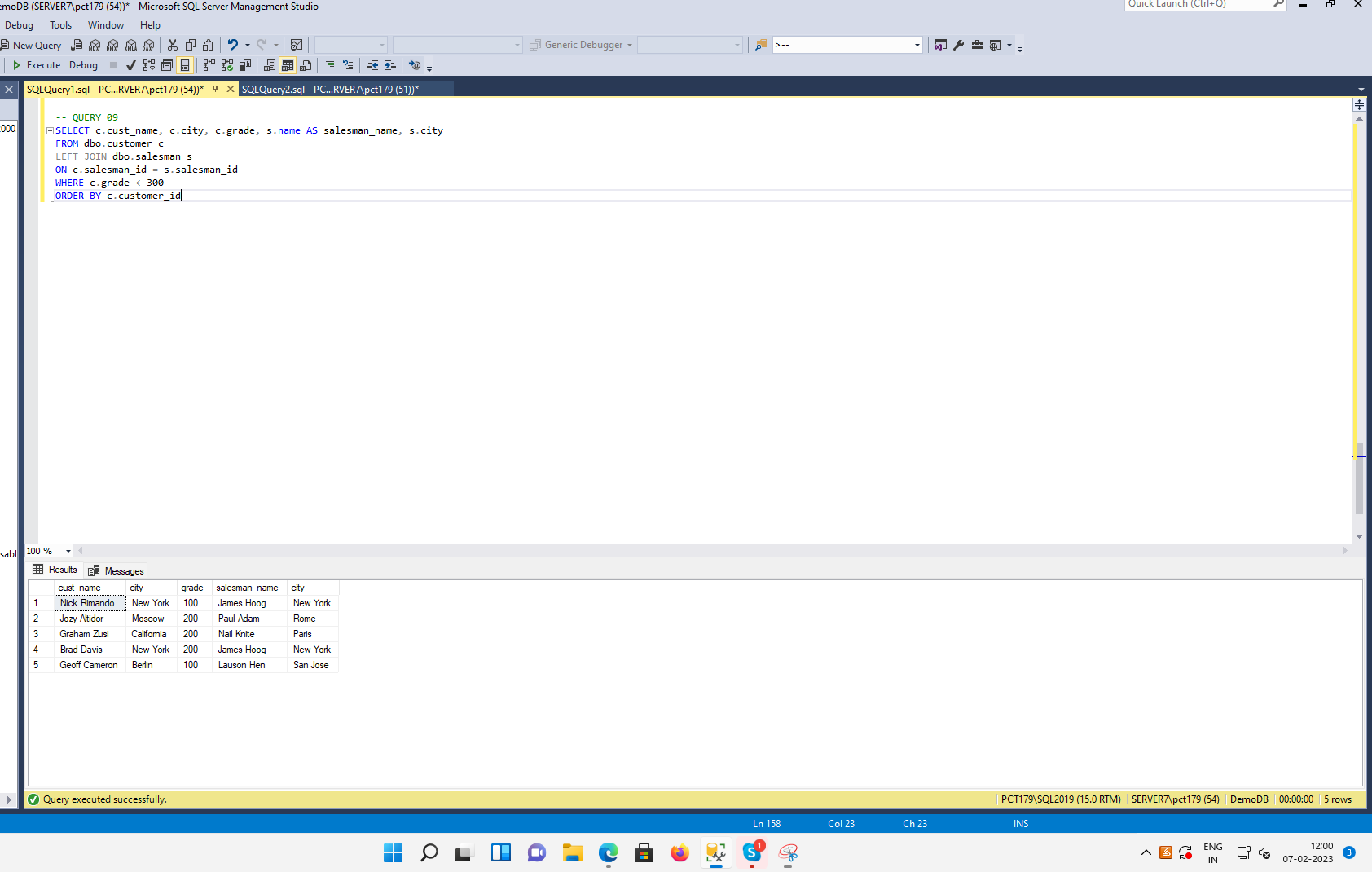
FROM dbo.customer c

LEFT JOIN dbo.salesman s

ON c.salesman\_id = s.salesman\_id

WHERE c.grade < 300

ORDER BY c.customer\_id



=============================================================

**QUERY 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.

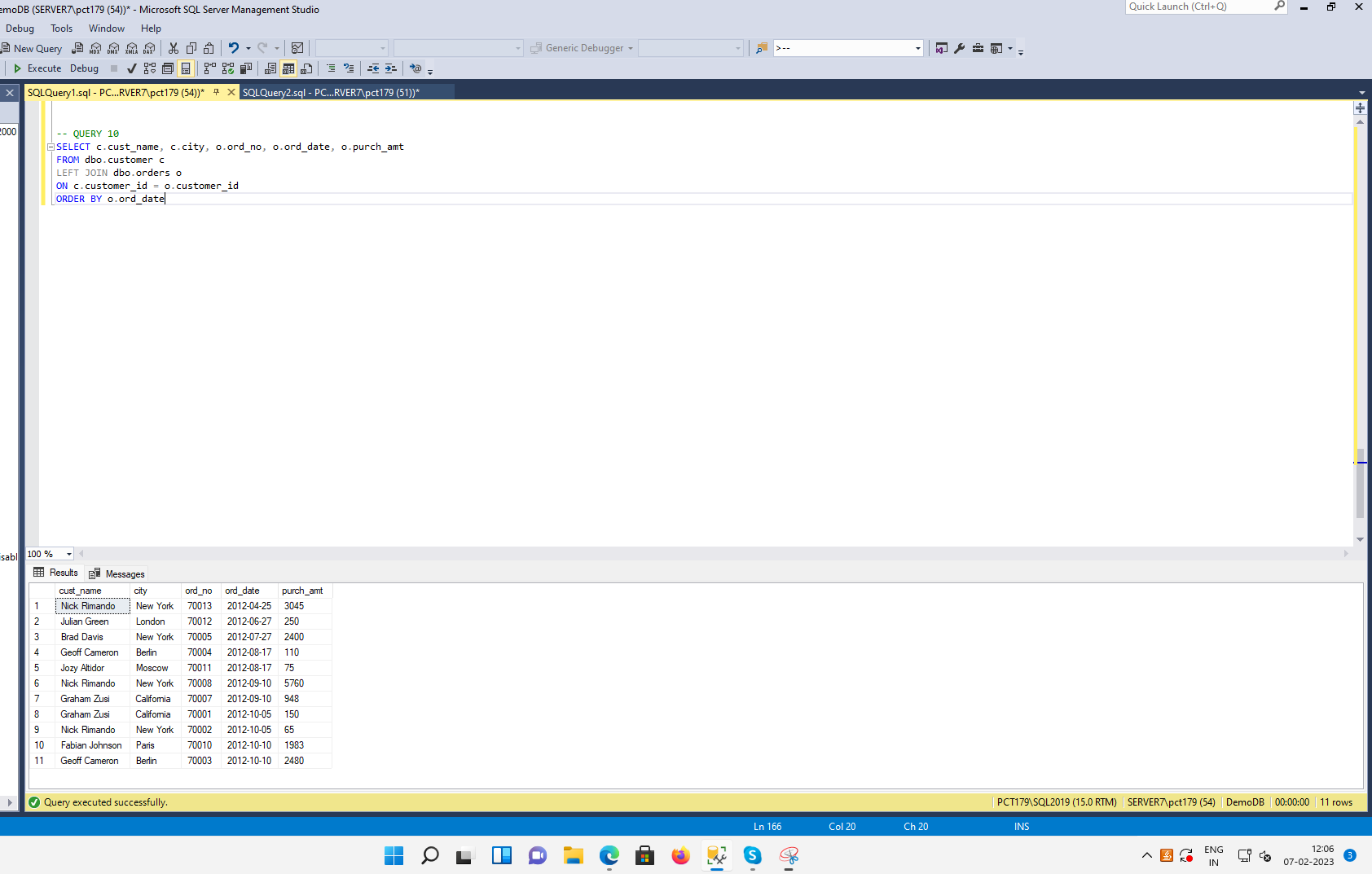
SELECT c.cust\_name, c.city, o.ord\_no, o.ord\_date, o.purch\_amt

FROM dbo.customer c

LEFT JOIN dbo.orders o

ON c.customer\_id = o.customer\_id

ORDER BY o.ord\_date



=============================================================

**QUERY 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, c.city, o.ord\_no, o.ord\_date, o.purch\_amt, s.name AS salesman\_name, s.commission

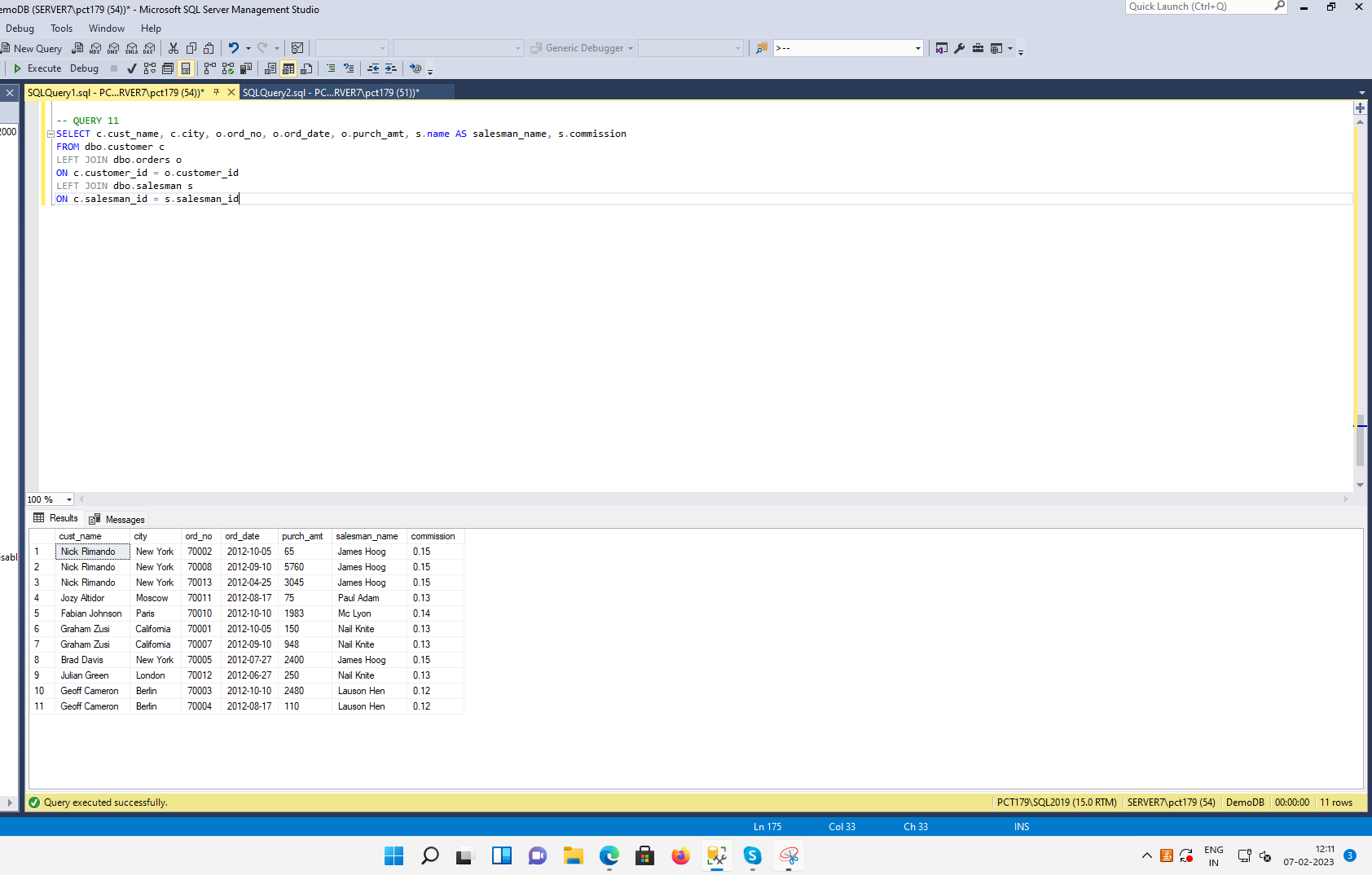
FROM dbo.customer c

LEFT JOIN dbo.orders o

ON c.customer\_id = o.customer\_id

LEFT JOIN dbo.salesman s

ON c.salesman\_id = s.salesman\_id



=============================================================

**QUERY 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

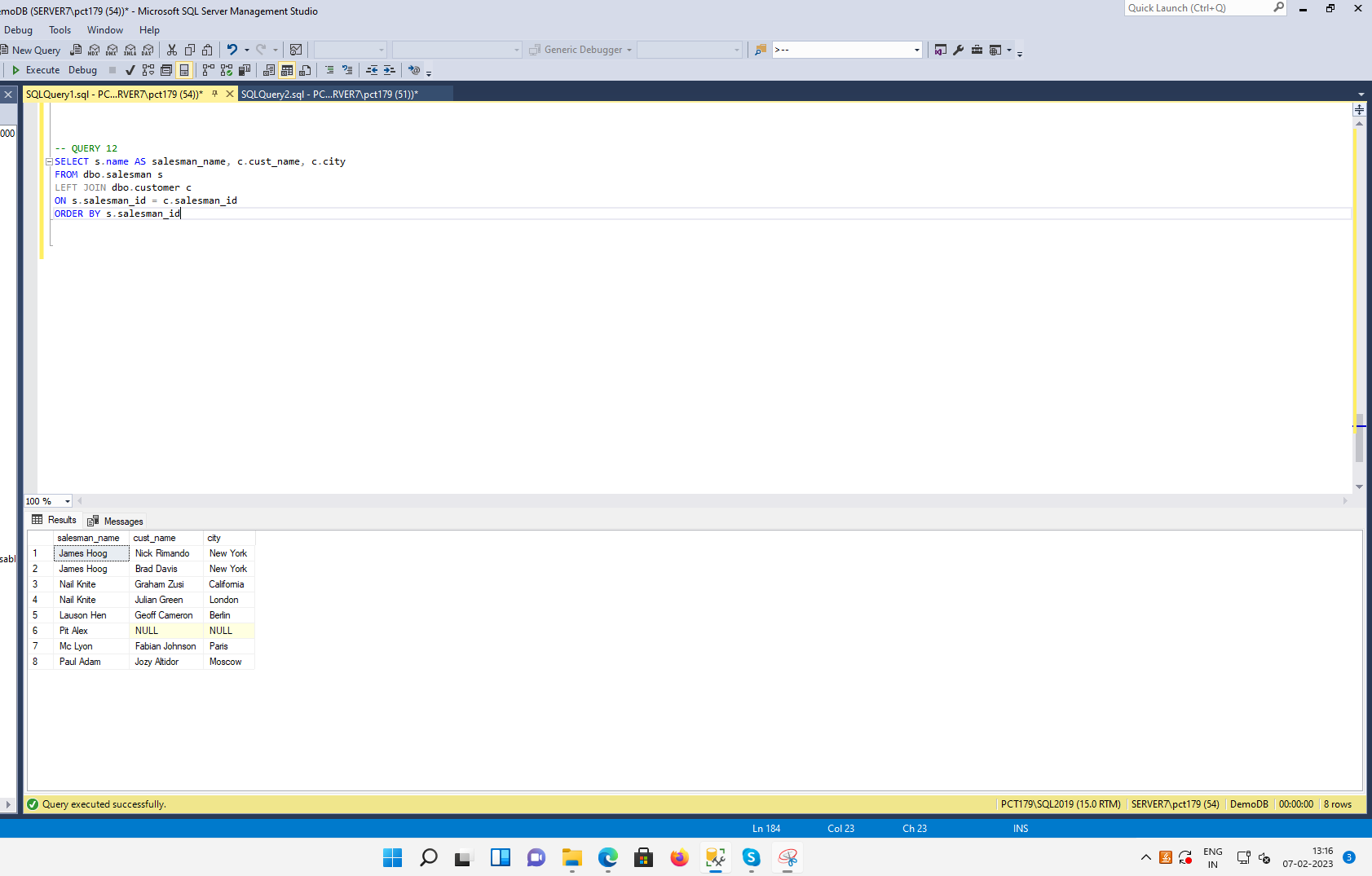
SELECT s.name AS salesman\_name, c.cust\_name, c.city

FROM dbo.salesman s

LEFT JOIN dbo.customer c

ON s.salesman\_id = c.salesman\_id

ORDER BY s.salesman\_id



=============================================================

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

SELECT s.name AS salesman\_name, c.cust\_name, c.city, c.grade, o.ord\_no, o.ord\_date, o.purch\_amt

FROM dbo.orders o

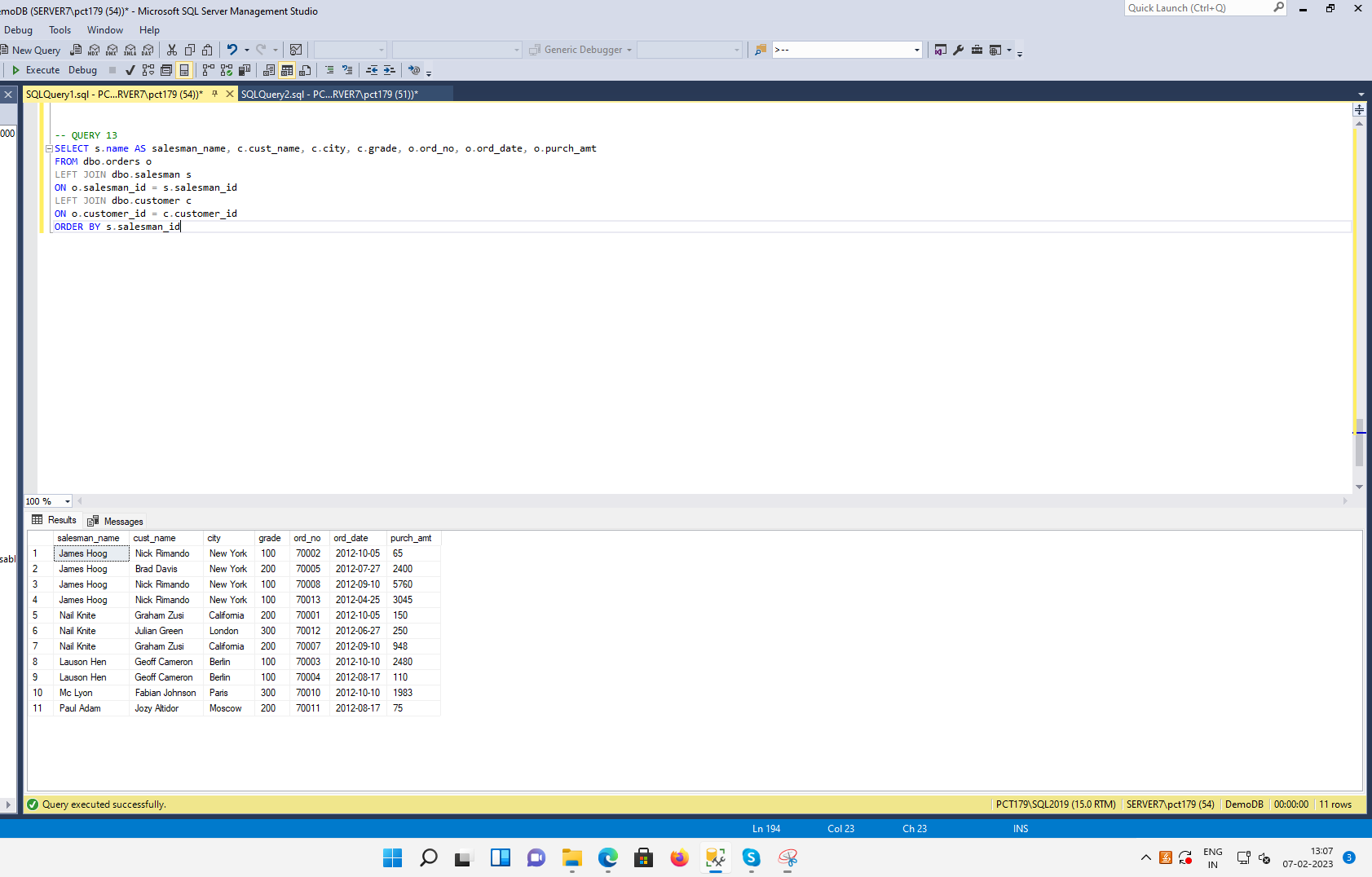
LEFT JOIN dbo.salesman s

ON o.salesman\_id = s.salesman\_id

LEFT JOIN dbo.customer c

ON o.customer\_id = c.customer\_id

ORDER BY s.salesman\_id



=============================================================

**QUERY 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 AS salesman\_name, c.cust\_name, c.grade, o.ord\_no, o.purch\_amt

FROM dbo.salesman s

LEFT JOIN dbo.customer c

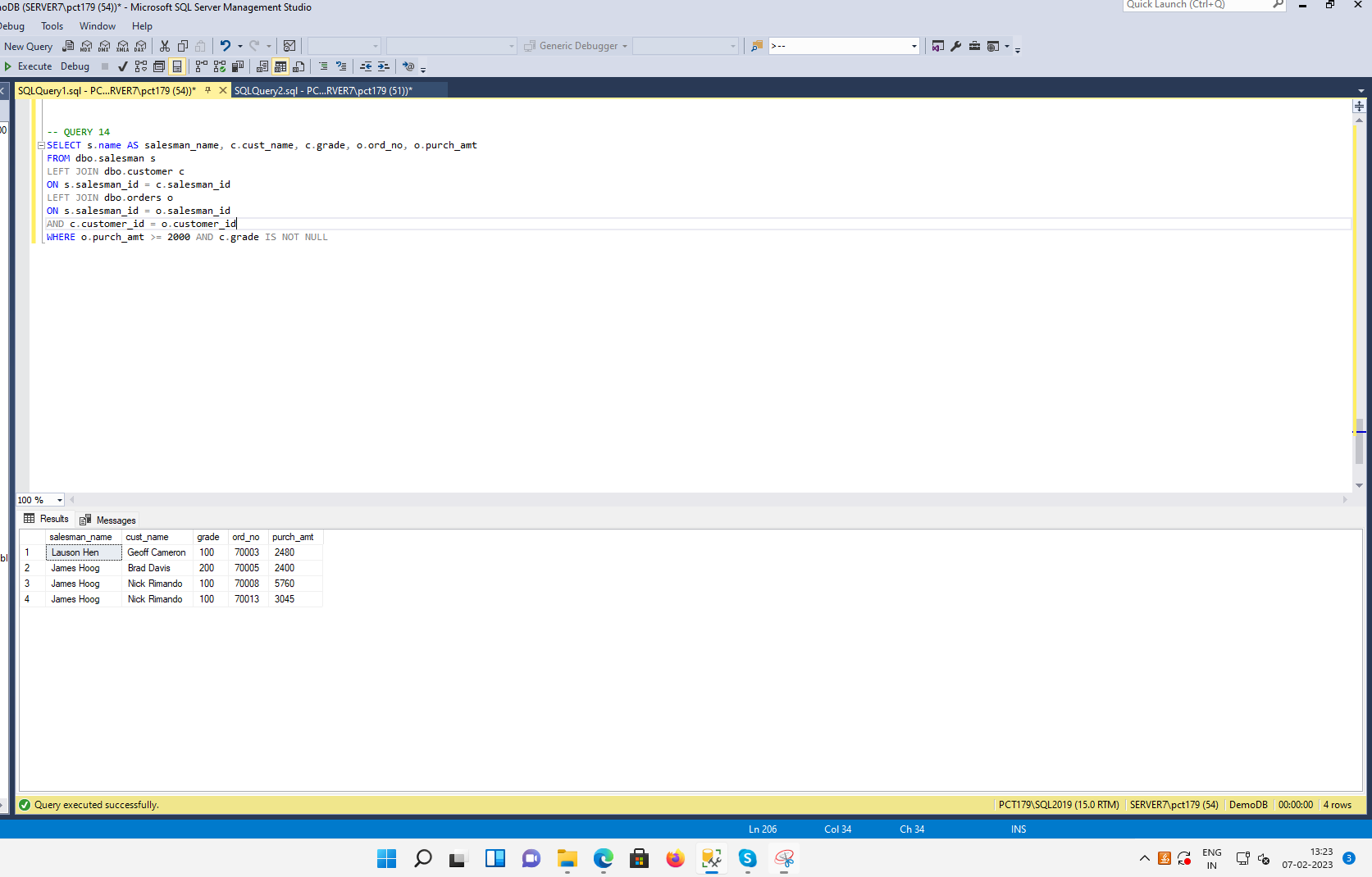
ON s.salesman\_id = c.salesman\_id

LEFT JOIN dbo.orders o

ON s.salesman\_id = o.salesman\_id

AND c.customer\_id = o.customer\_id

WHERE o.purch\_amt >= 2000 AND c.grade IS NOT NULL



=============================================================

**QUERY 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 AS salesman\_name, c.cust\_name, c.grade, o.ord\_no, o.purch\_amt

FROM dbo.salesman s

LEFT JOIN dbo.customer c

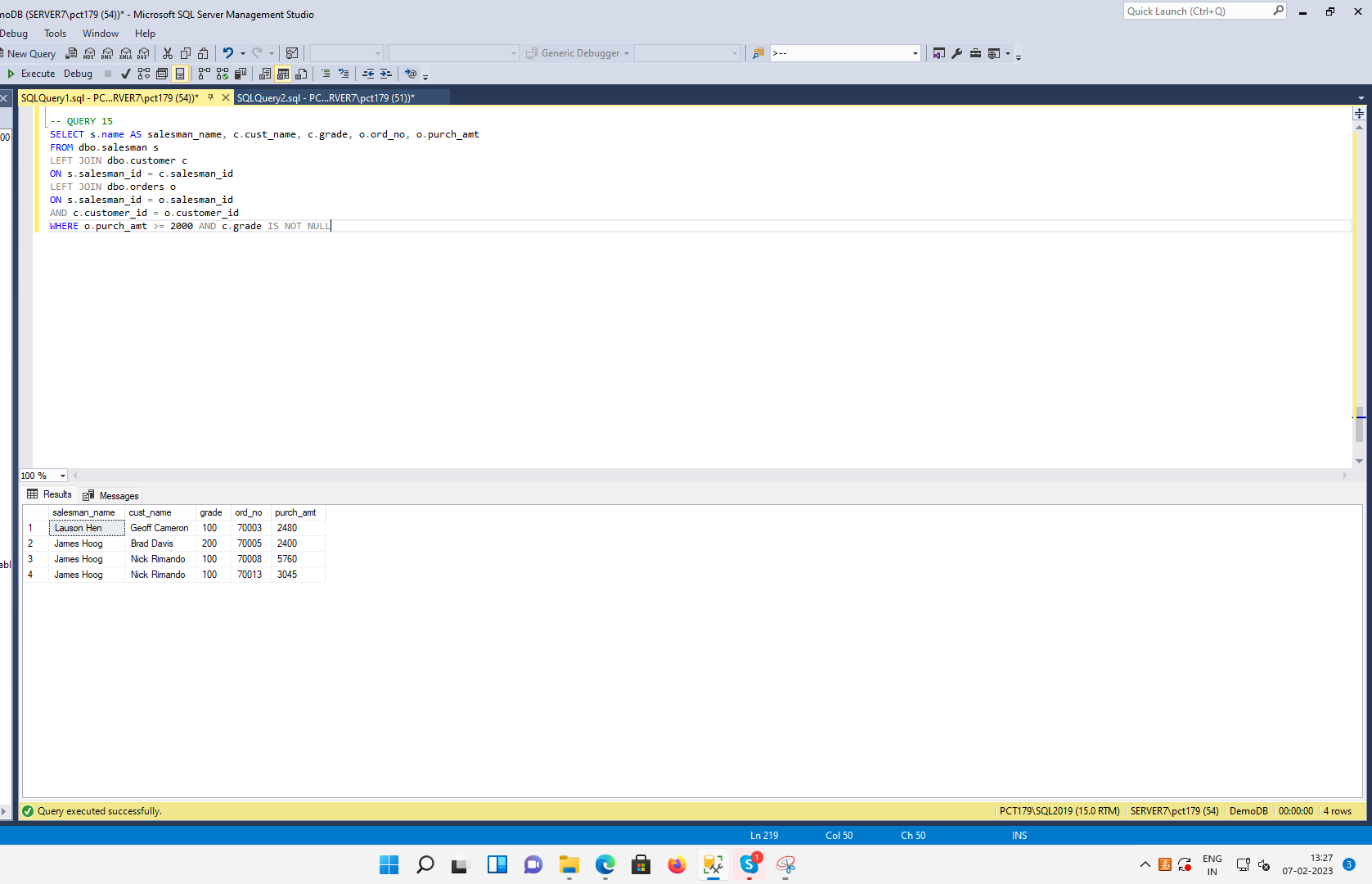
ON s.salesman\_id = c.salesman\_id

LEFT JOIN dbo.orders o

ON s.salesman\_id = o.salesman\_id

AND c.customer\_id = o.customer\_id

WHERE o.purch\_amt >= 2000 AND c.grade IS NOT NULL



=============================================================

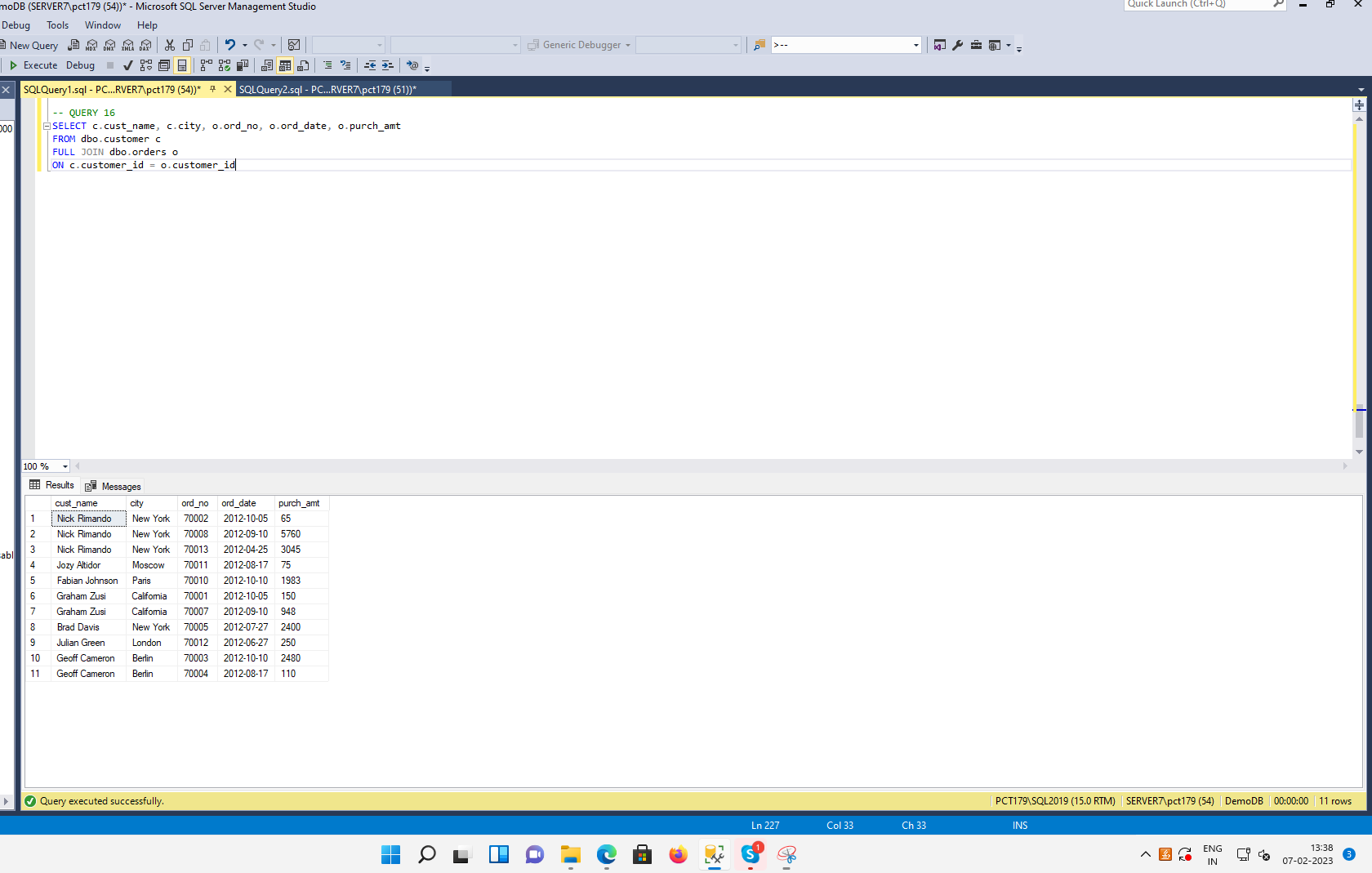
**QUERY 16:** Write a SQL statement to generate a report with the customer name, city, order no., order date, purchase amount for only those customers on the list who must have a grade and placed one or more orders or which order(s) have been placed by the customer who neither is on the list nor has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

SELECT c.cust\_name, c.city, o.ord\_no, o.ord\_date, o.purch\_amt

FROM dbo.customer c

FULL JOIN dbo.orders o

ON c.customer\_id = o.customer\_id



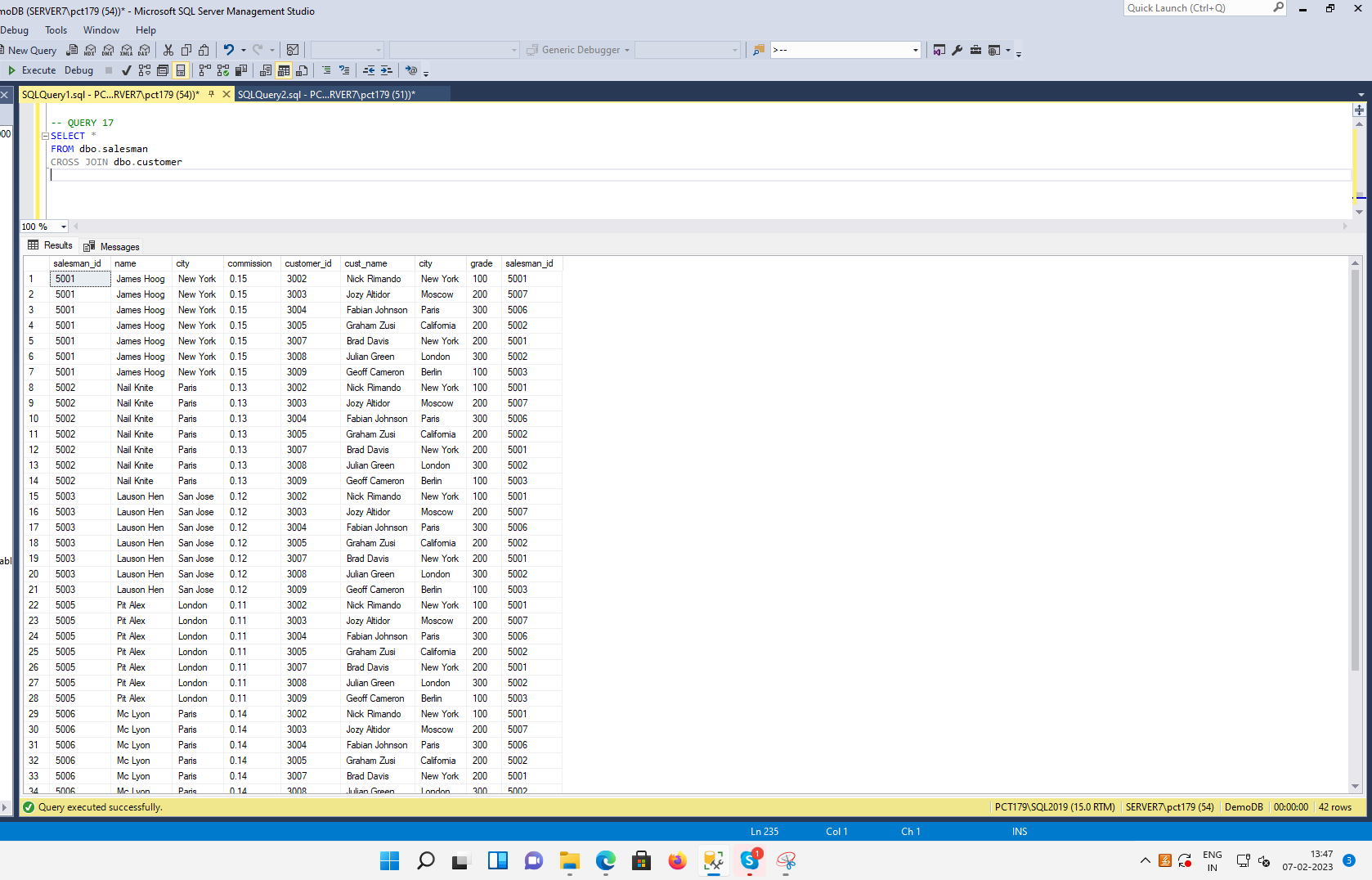
=============================================================

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

SELECT \*

FROM dbo.salesman

CROSS JOIN dbo.customer



=============================================================

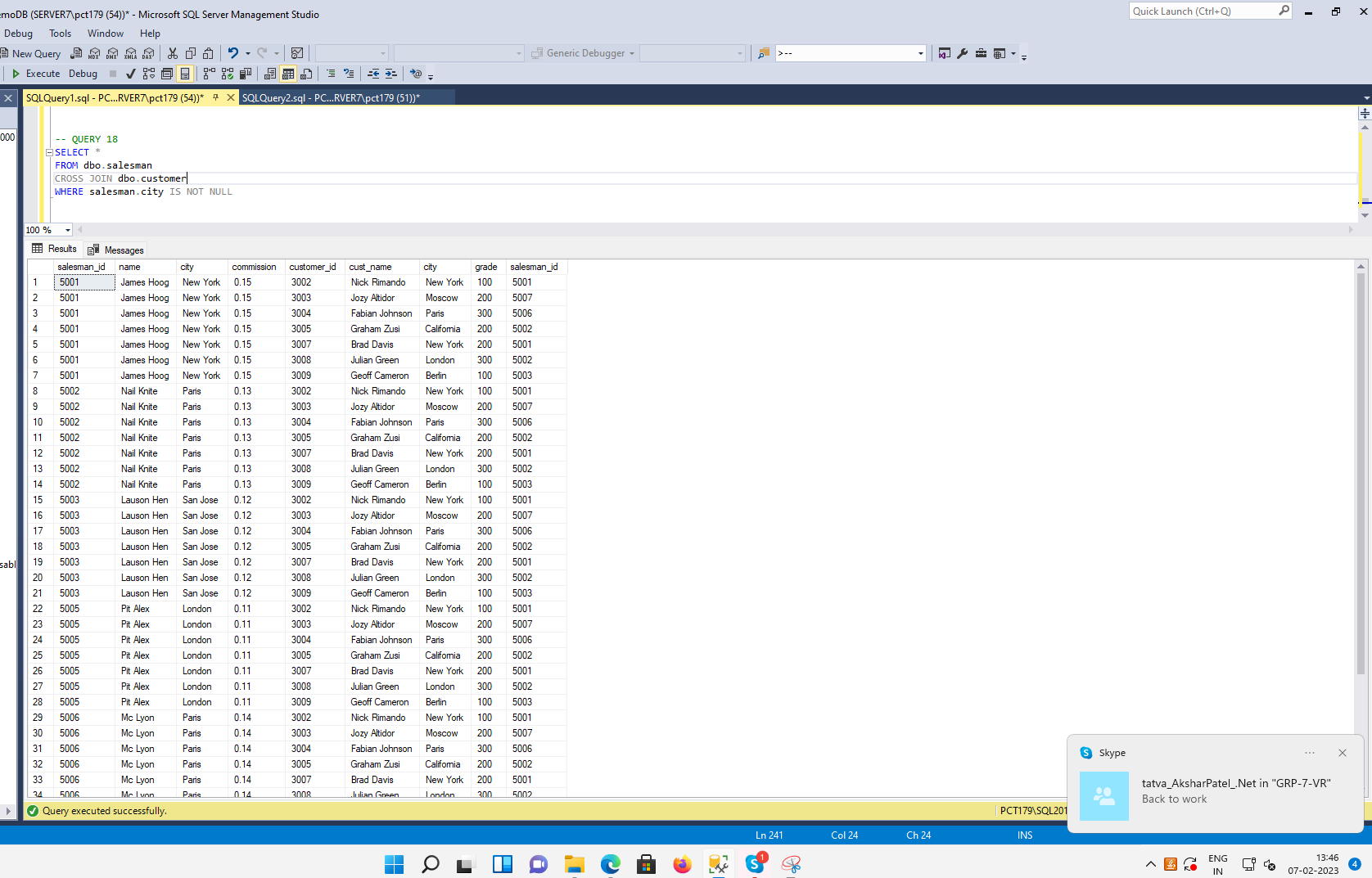
**QUERY 18:** Write a SQL statement to create a Cartesian product between salesperson and customer, i.e. each salesperson will appear for all customers and vice versa for that salesperson who belongs to that city

SELECT \*

FROM dbo.salesman

CROSS JOIN dbo.customer

WHERE salesman.city IS NOT NULL



=============================================================

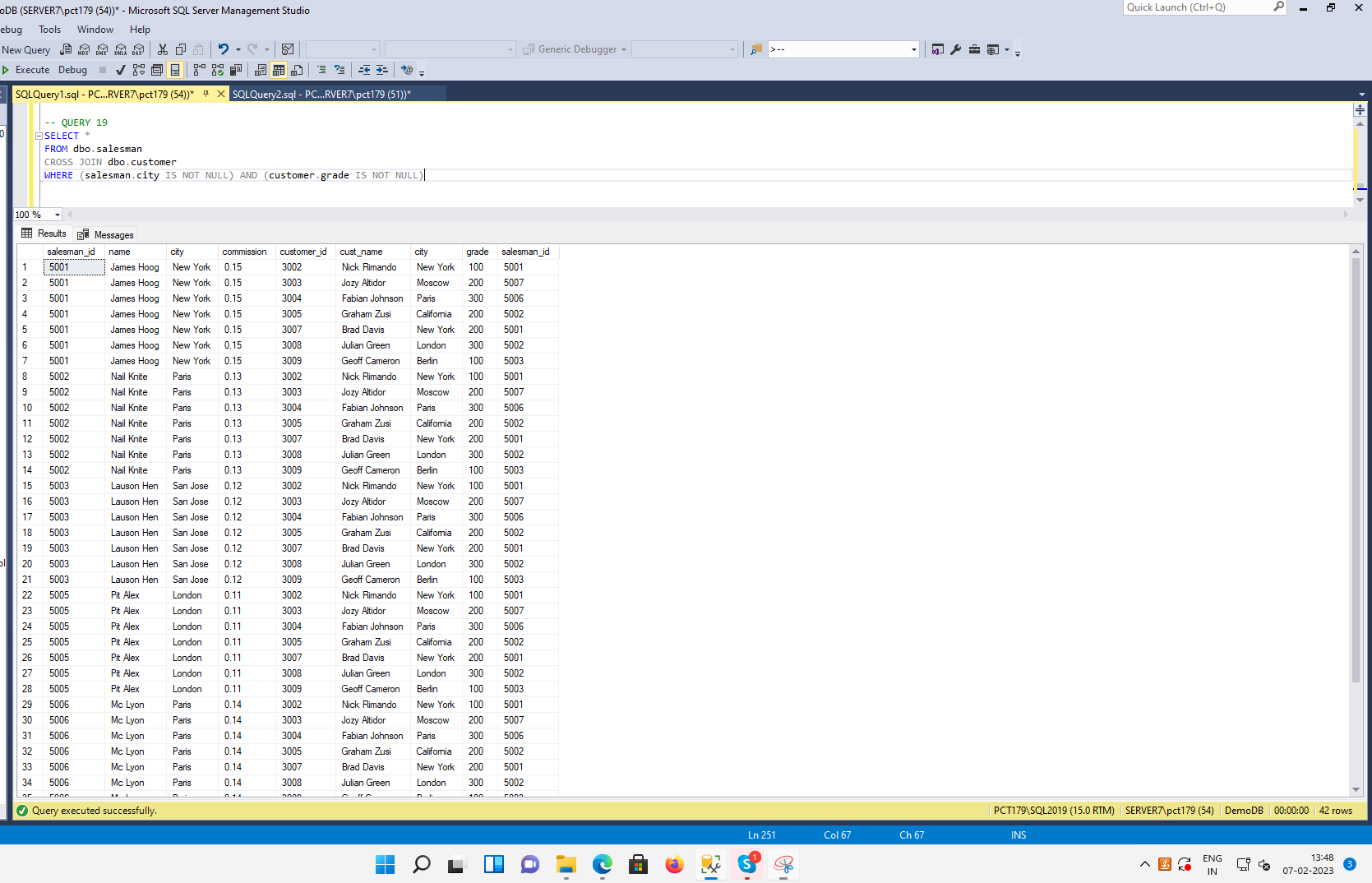
**QUERY 19:** Write a SQL statement to create a Cartesian product between salesperson and customer, i.e. each salesperson will appear for every customer and vice versa for those salesmen who belong to a city and customers who require a grade

SELECT \*

FROM dbo.salesman

CROSS JOIN dbo.customer

WHERE (salesman.city IS NOT NULL) AND (customer.grade IS NOT NULL)



=============================================================

**QUERY 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 dbo.salesman s

CROSS JOIN dbo.customer c

WHERE (s.city IS NOT NULL) AND (s.city != c.city) AND (c.grade IS NOT NULL)

