CREATE DATABASE Assignment2;

use Assignment2;

CREATE TABLE salesman(

salesman\_id INT primary key,

name NVARCHAR(20),

city NVARCHAR(30),

commission FLOAT

);

CREATE TABLE customer(

customer\_id INT PRIMARY KEY,

cust\_name NVARCHAR(30),

city NVARCHAR(30),

grade INT,

salesman\_id INT,

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

);

CREATE TABLE orders(

order\_no INT PRIMARY KEY,

purch\_amt INT,

ord\_date DATE,

customer\_id INT,

salesman\_id INT,

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

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

);

INSERT INTO salesman VALUES (5001,'James Hoog','New York',0.15),

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

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

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

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

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

INSERT INTO customer VALUES

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

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

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

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

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

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

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

(3001 , 'Brad Guzan' , 'London' , 0 , 5005);

INSERT INTO orders VALUES

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

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

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

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

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

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

(70008, 5760.0, '2012-09-10', 3002, 5001),

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

(70003, 2480.4, '2012-10-10', 3009, 5003),

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

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

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

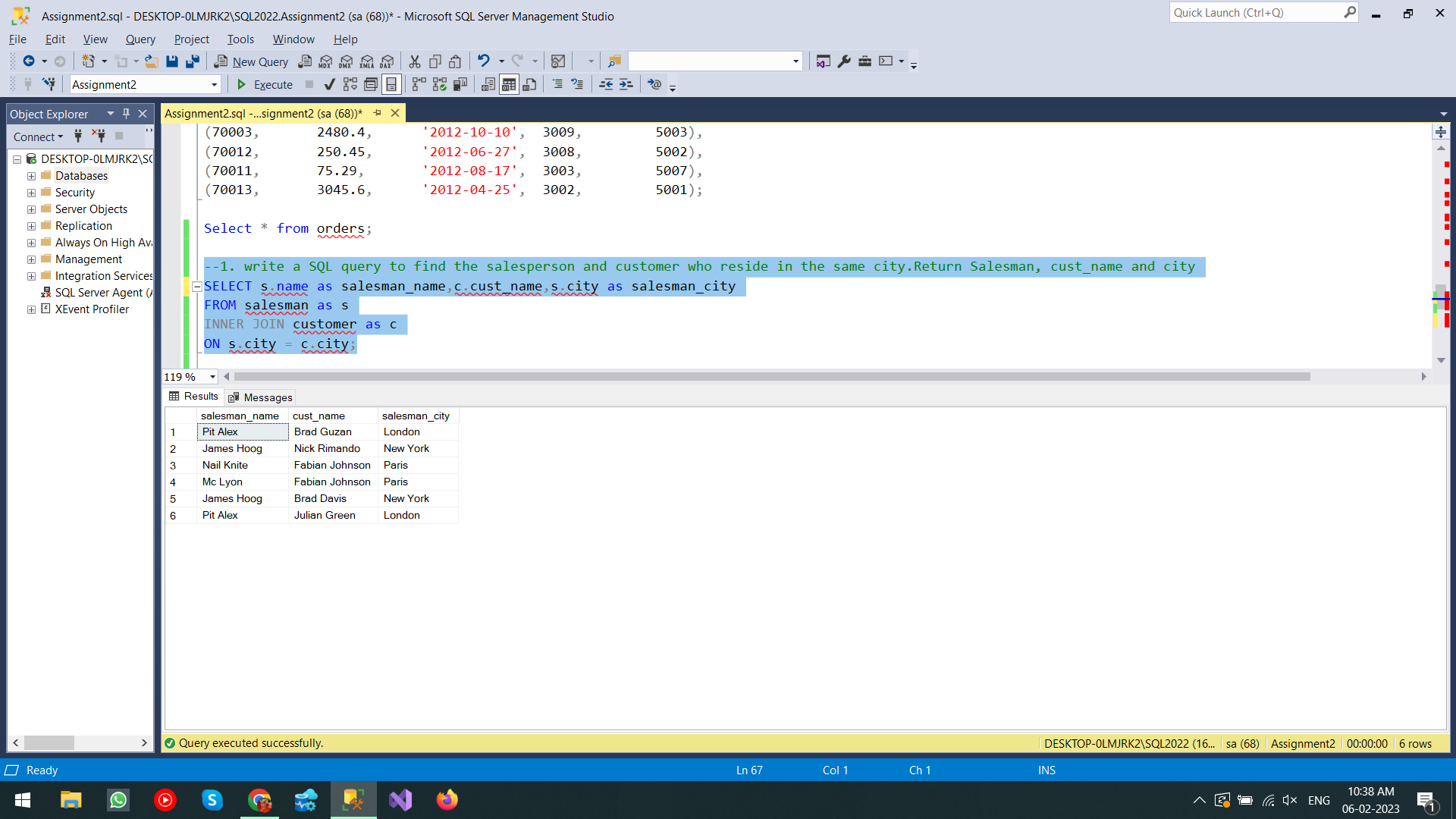
--1. write a SQL query to find the salesperson and customer who reside in the same city.Return Salesman, cust\_name and city.

SELECT s.name as salesman\_name,c.cust\_name,s.city as salesman\_city

FROM salesman as s

INNER JOIN customer as 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.order\_no,o.purch\_amt,c.cust\_name, c.city

FROM orders as o

INNER JOIN customer as c

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

WHERE o.purch\_amt BETWEEN 500 AND 2000;



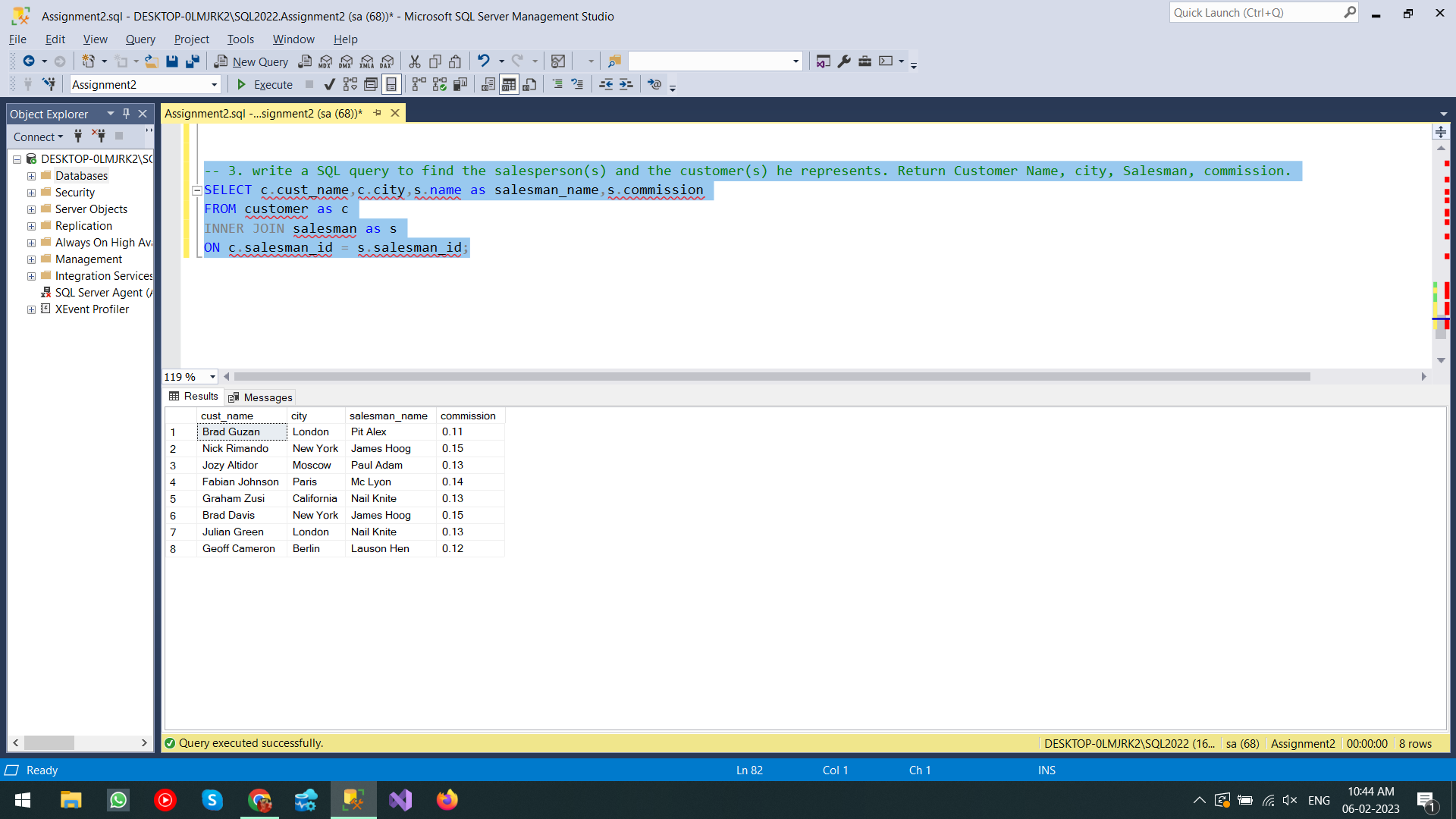
-- 3. write a SQL query to find the salesperson(s) and the customer(s) he represents. Return Customer Name, city, Salesman, commission.

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

FROM customer as c

INNER JOIN salesman as 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.

INSERT INTO salesman values(5008 ,'Tyson Knite','Los Angeles',13);

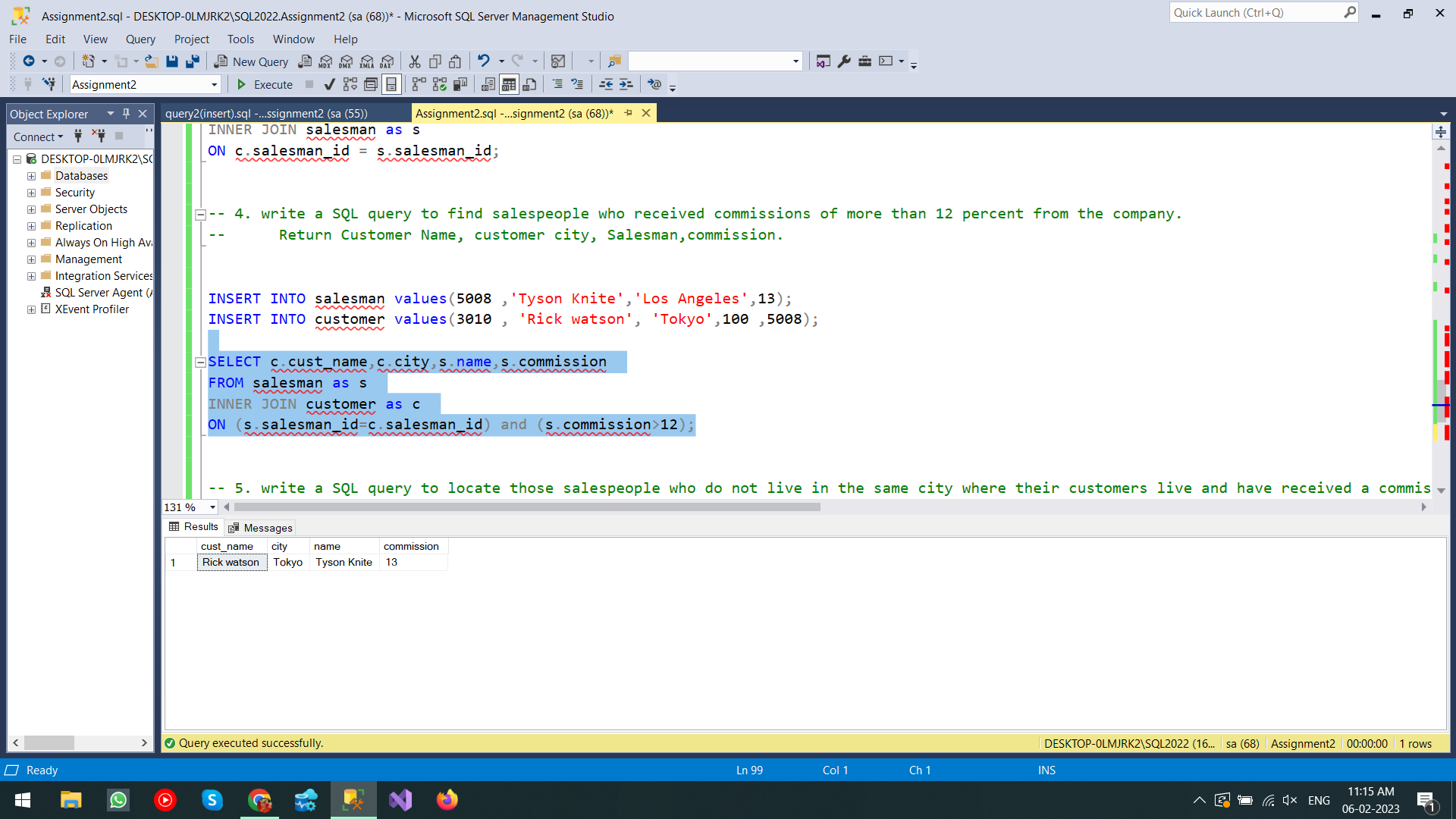
INSERT INTO customer values(3010 , 'Rick watson', 'Tokyo',100 ,5008);

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

FROM salesman as s

INNER JOIN customer as c

ON (s.salesman\_id=c.salesman\_id) and (s.commission>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

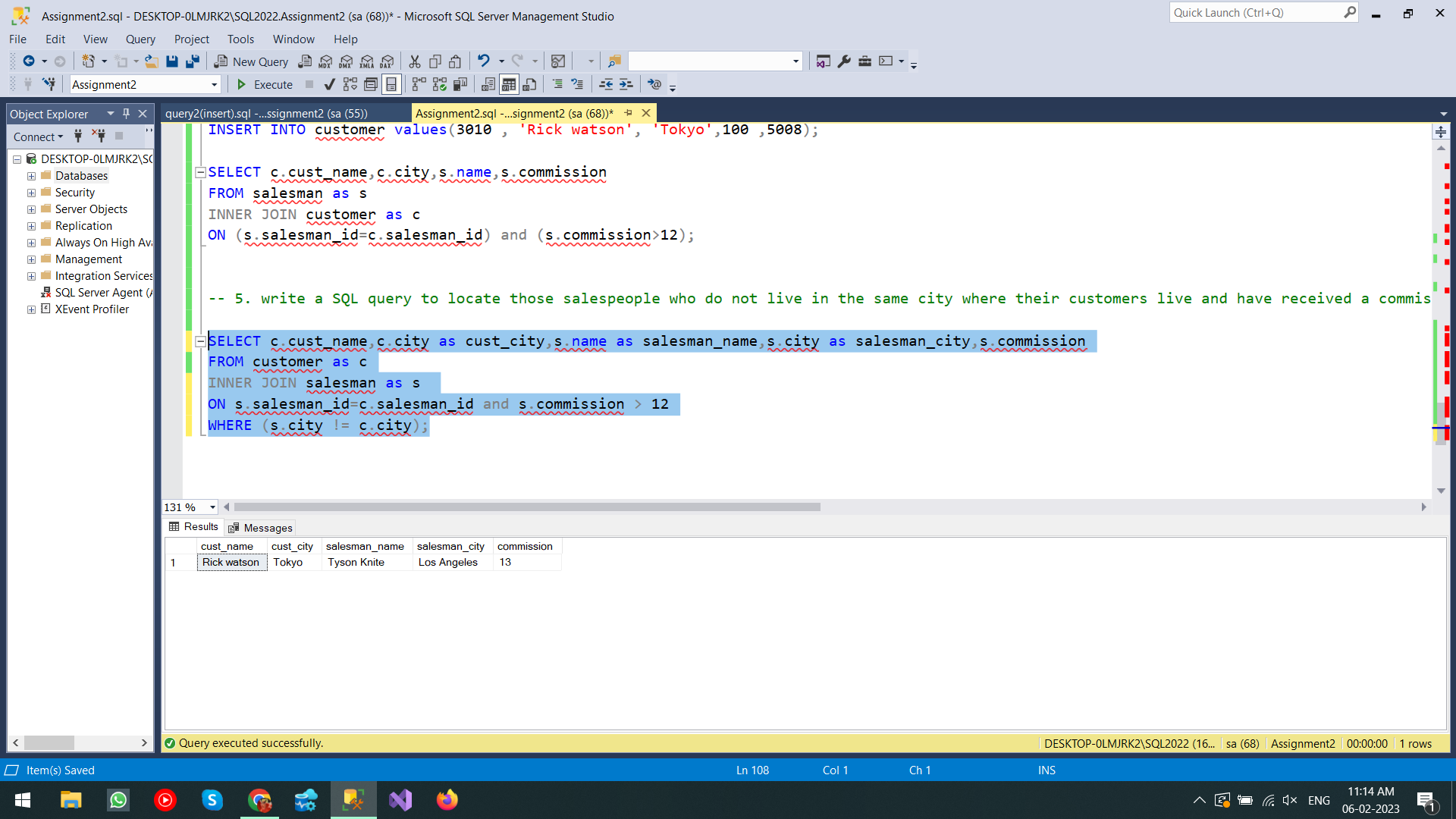
SELECT c.cust\_name,c.city as cust\_city,s.name as salesman\_name,s.city as salesman\_city,s.commission

FROM customer as c

INNER JOIN salesman as s

ON s.salesman\_id=c.salesman\_id and s.commission > 12

WHERE (s.city != c.city);



-- 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.order\_no , o.ord\_date,o.purch\_amt,c.cust\_name,c.grade as customer\_grade,s.name as salesman\_name,s.commission as salesman\_commission

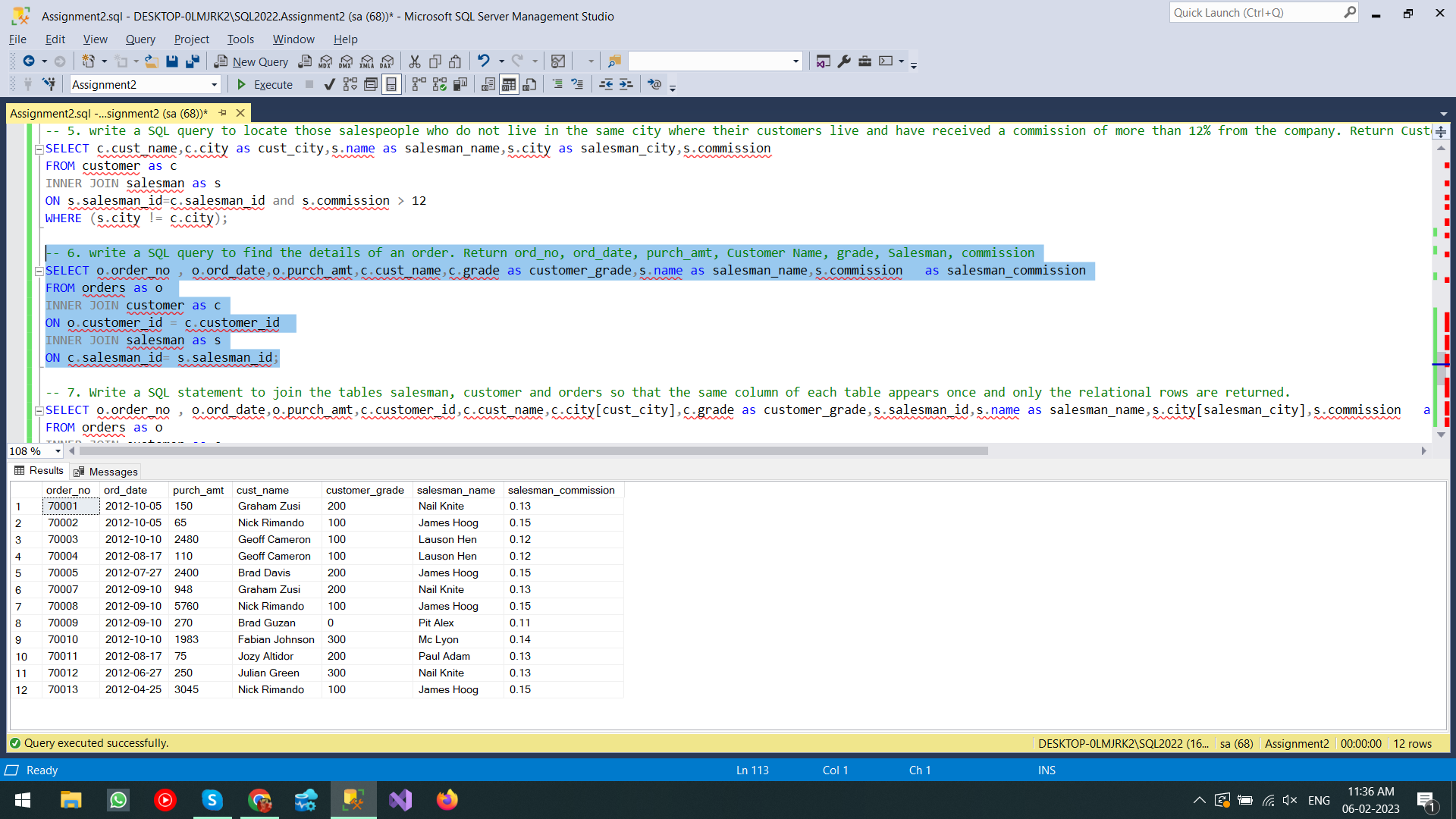
FROM orders as o

INNER JOIN customer as c

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

INNER JOIN salesman as s

ON c.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 o.order\_no , o.ord\_date,o.purch\_amt,c.customer\_id,c.cust\_name,c.city[cust\_city],c.grade as customer\_grade,s.salesman\_id,s.name as salesman\_name,s.city[salesman\_city],s.commission as salesman\_commission

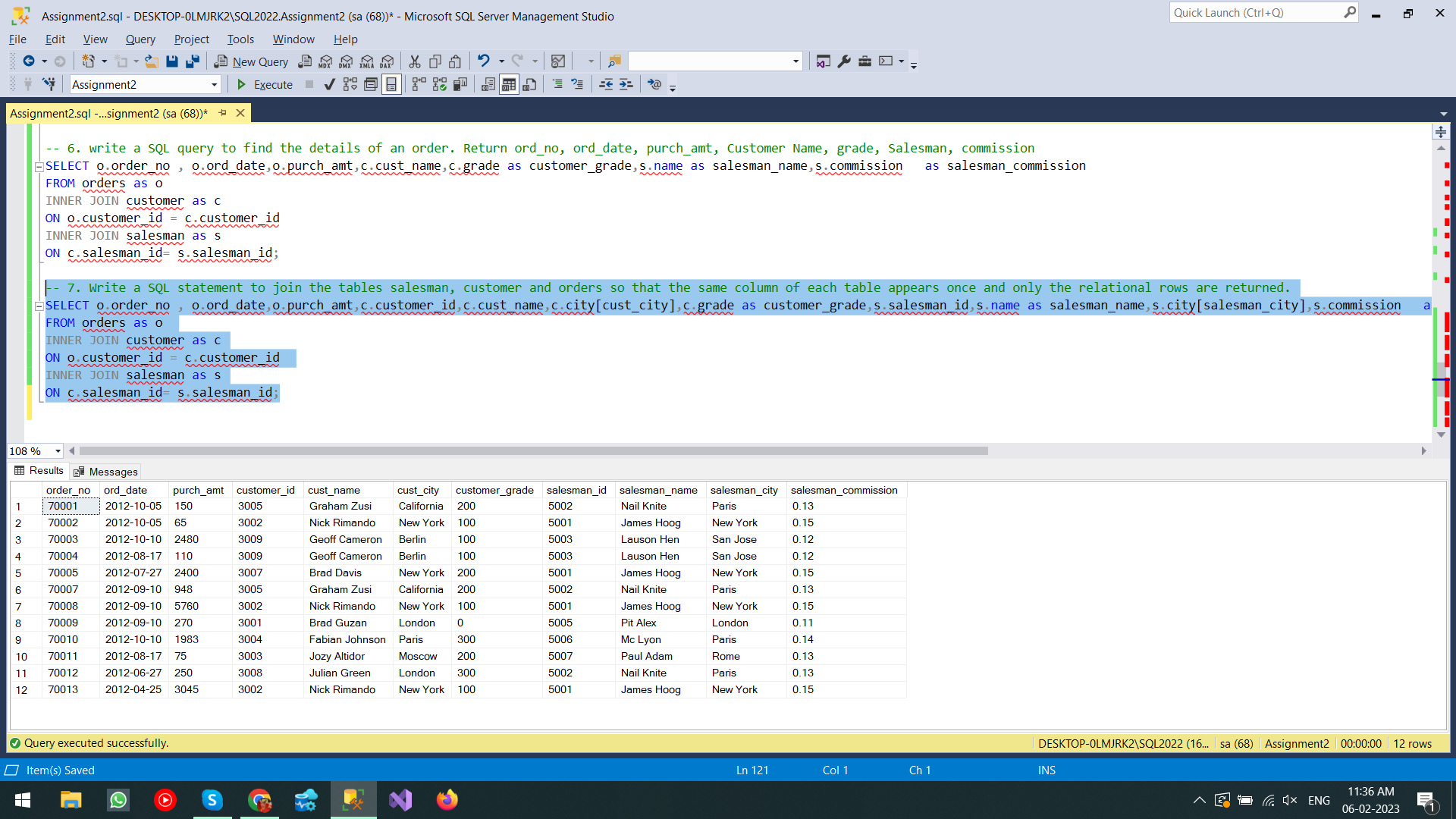
FROM orders as o

INNER JOIN customer as c

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

INNER JOIN salesman as s

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



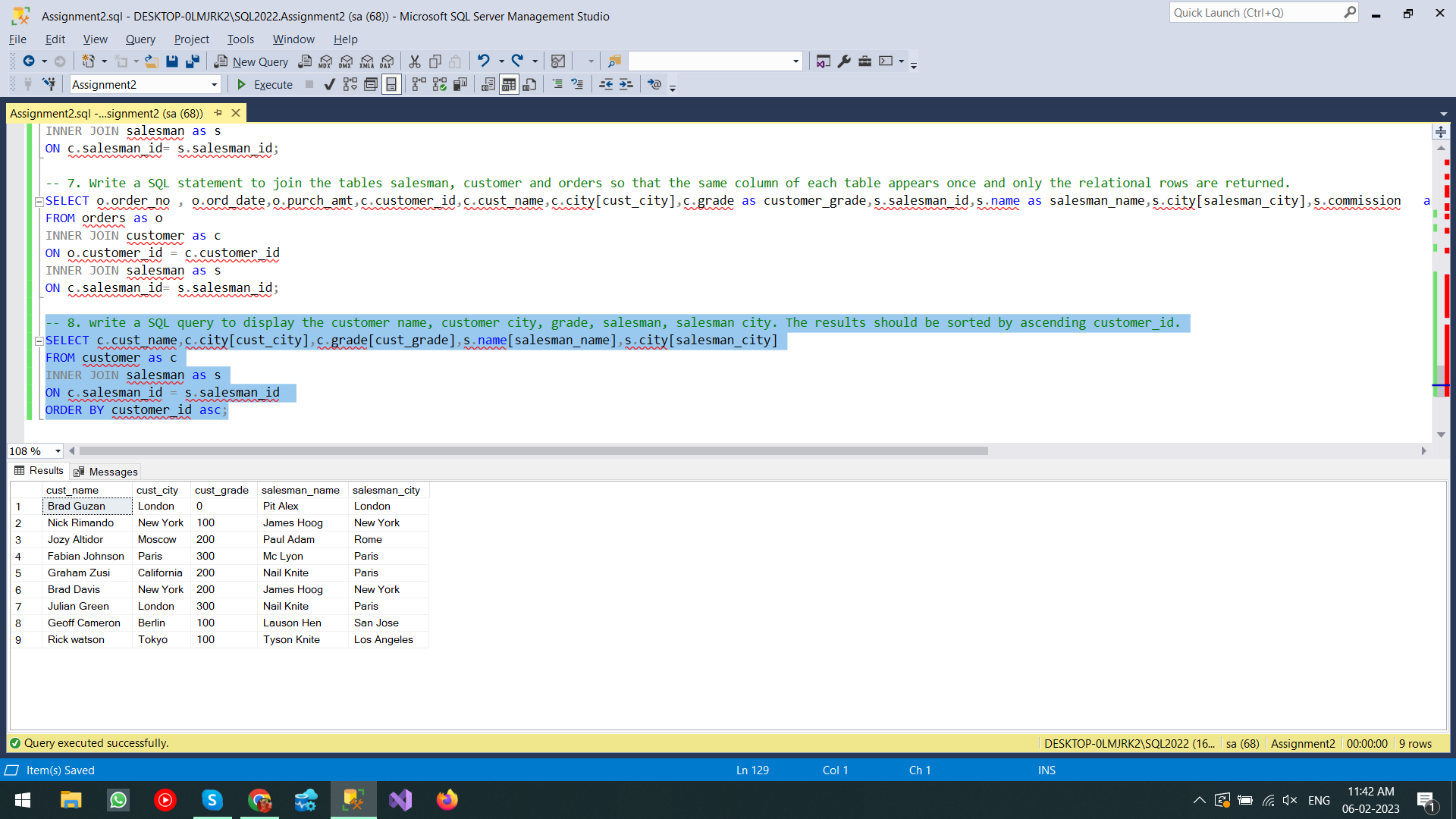
-- 8. write a SQL query to display the customer name, customer city, grade, salesman, salesman city. The results should be sorted by ascending customer\_id.

SELECT c.cust\_name,c.city[cust\_city],c.grade[cust\_grade],s.name[salesman\_name],s.city[salesman\_city]FROM customer as c

INNER JOIN salesman as s

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

ORDER BY customer\_id asc;



--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.

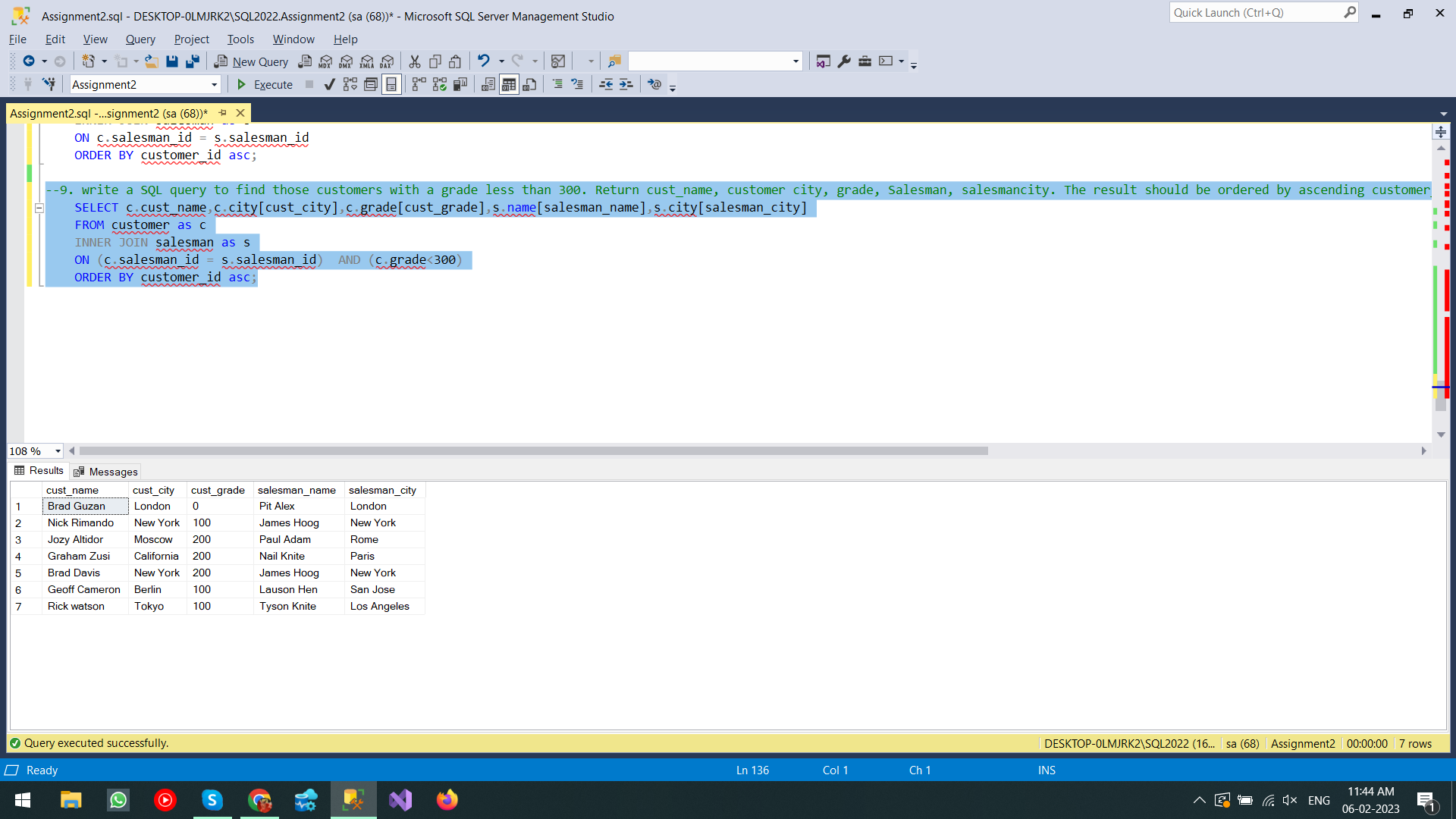
SELECT c.cust\_name,c.city[cust\_city],c.grade[cust\_grade],s.name[salesman\_name],s.city[salesman\_city]

FROM customer as c

INNER JOIN salesman as s

ON (c.salesman\_id = s.salesman\_id) AND (c.grade<300)

ORDER BY customer\_id asc;



-- 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

insert into customer values(3011 , 'Eric Dickson', 'New York',50 ,5003);

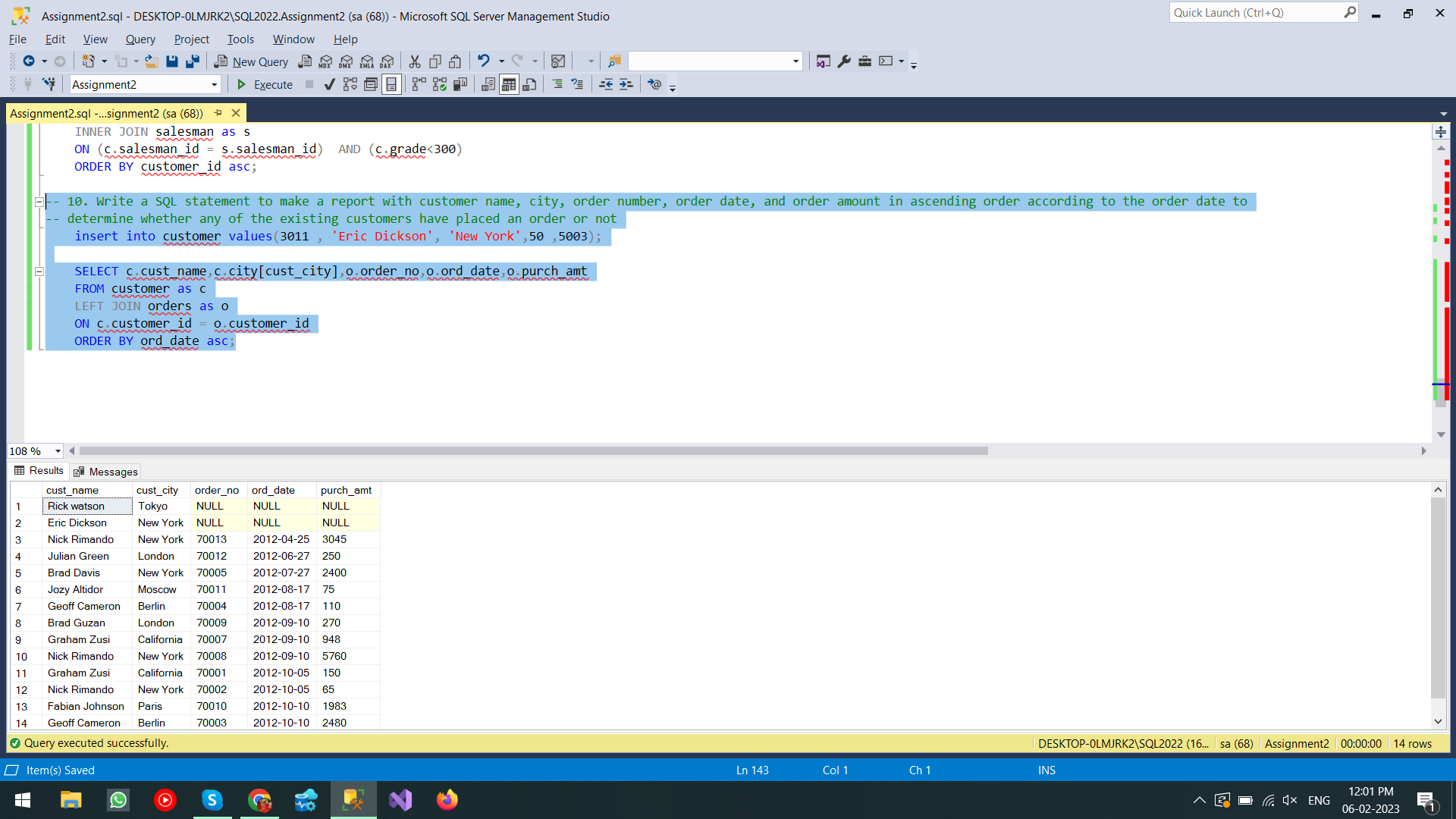
SELECT c.cust\_name,c.city[cust\_city],o.order\_no,o.ord\_date,o.purch\_amt

FROM customer as c

LEFT JOIN orders as o

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

ORDER BY ord\_date asc;



-- 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

insert into orders values(70015,25000,'2012-5-15',3010,null);

SELECT c.cust\_name , c.city[cust\_city],o.order\_no,o.ord\_date,o.purch\_amt,s.name[salesman\_name],s.commission

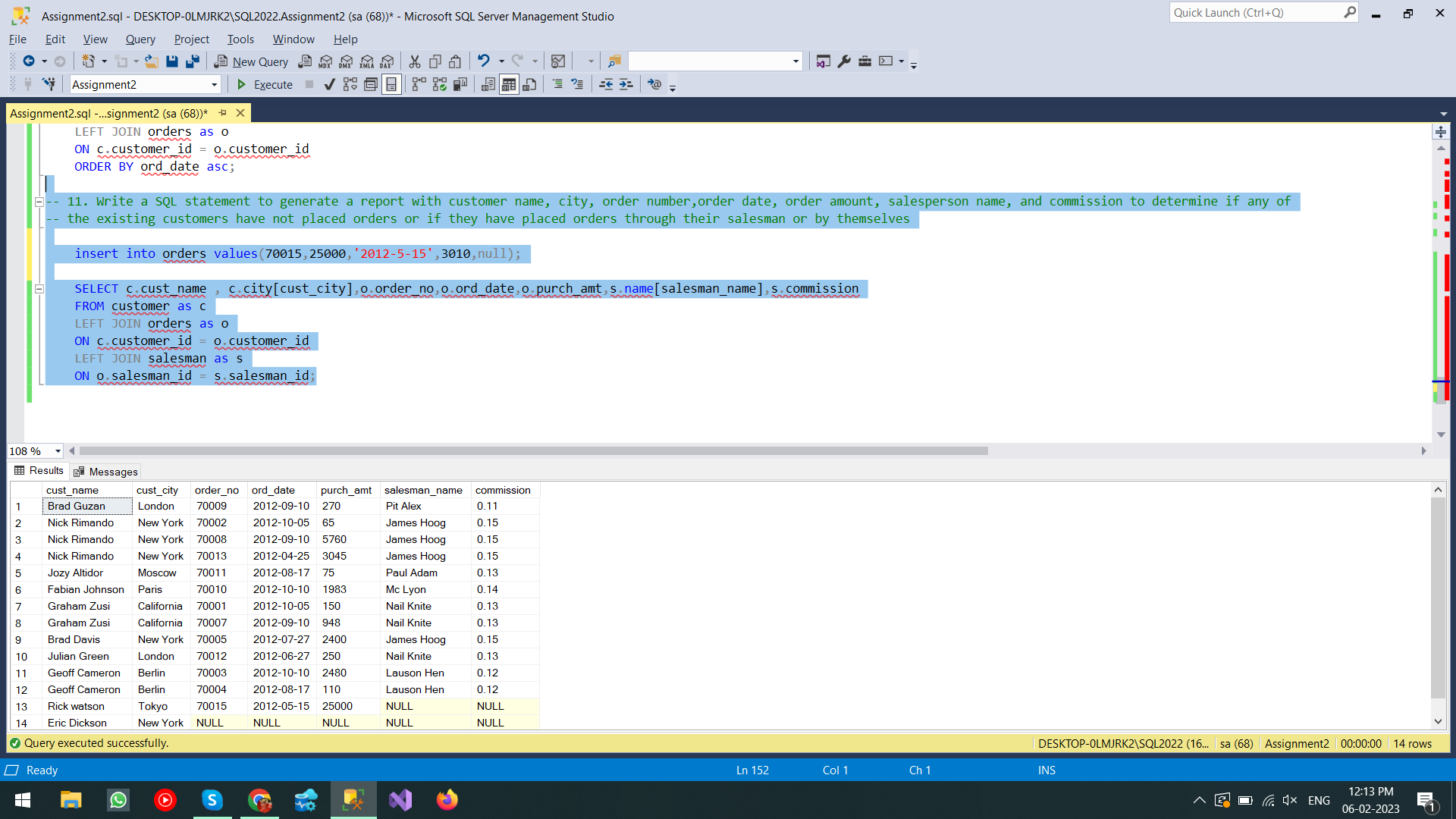
FROM customer as c

LEFT JOIN orders as o

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

LEFT JOIN salesman as s

ON o.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

insert into salesman values(5009,'Mr. Bhushan','Tokyo',20);

SELECT s.name[Salesman name],COUNT(customer\_id)[No\_of\_customers]

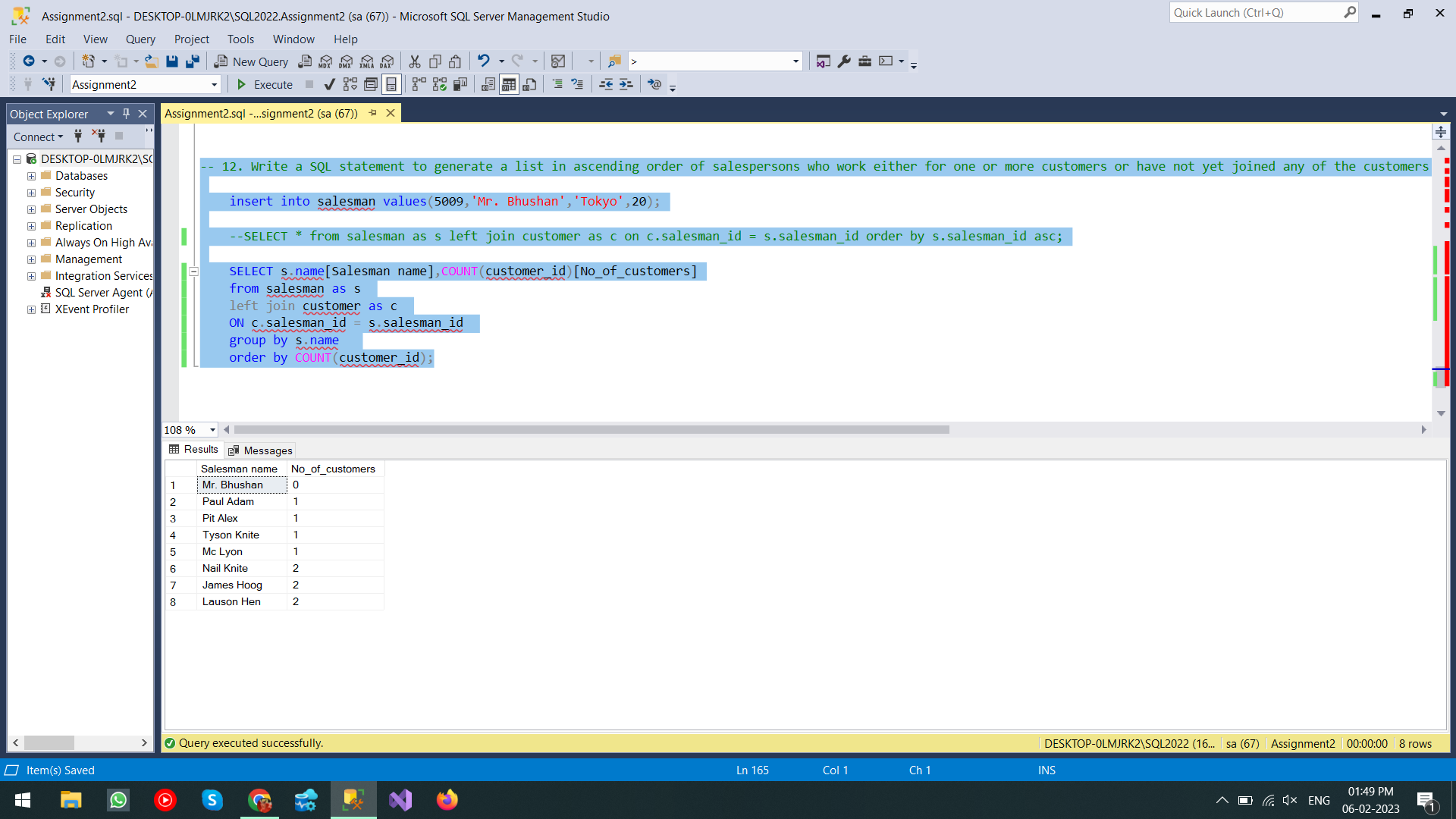
from salesman as s

left join customer as c

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

group by s.name

order by COUNT(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,c.city,c.grade,o.order\_no,o.ord\_date,o.purch\_amt

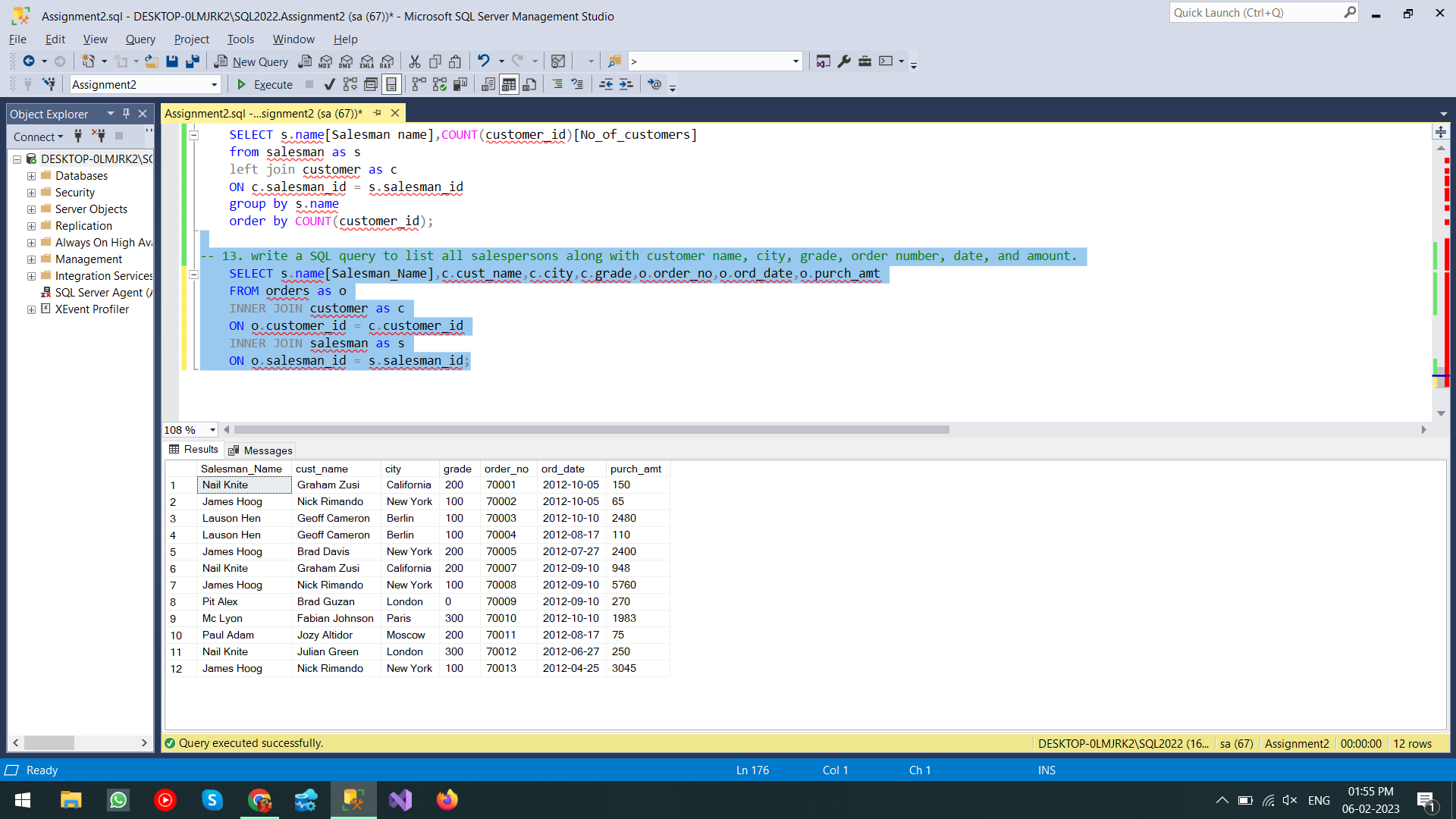
FROM orders as o

INNER JOIN customer as c

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

INNER JOIN salesman as s

ON o.salesman\_id = s.salesman\_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,COUNT(c.customer\_id)[No of Customers]

FROM salesman as s

LEFT JOIN customer as c

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

WHERE c.customer\_id in (

SELECT c.customer\_id

FROM customer as c

INNER JOIN orders as o ON (c.customer\_id=o.customer\_id)

WHERE (((c.customer\_id in

(SELECT c.customer\_id FROM customer as c

INNER JOIN orders as o ON o.customer\_id=c.customer\_id

GROUP BY c.customer\_id having COUNT(o.order\_no)>=1))

or (o.purch\_amt>2000))

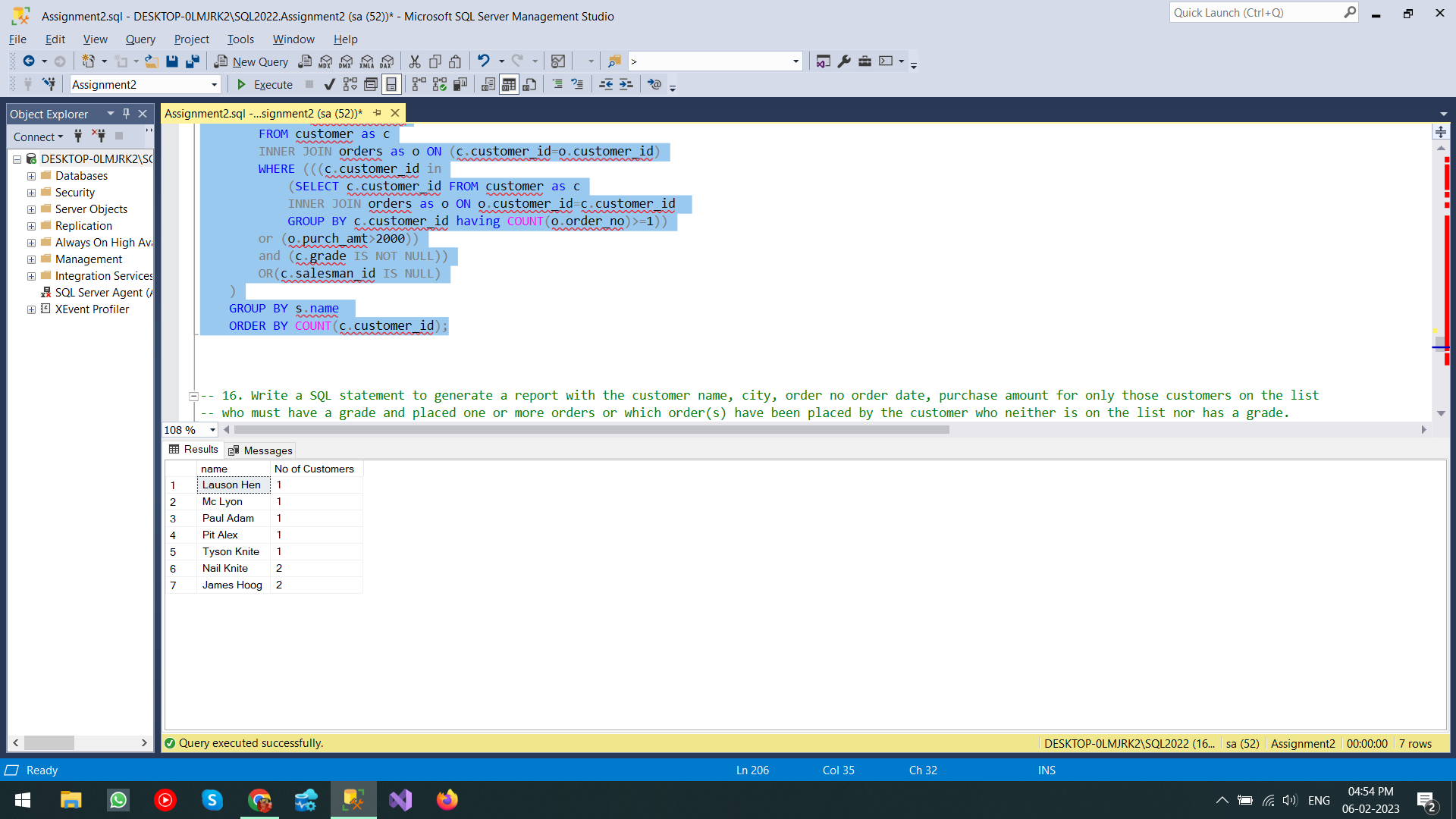
and (c.grade IS NOT NULL))

OR(c.salesman\_id IS NULL)

)

GROUP BY s.name

ORDER BY COUNT(c.customer\_id);



-- 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.order\_no,o.ord\_date,o.purch\_amt

FROM customer as c

INNER JOIN orders as o

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

where ((c.grade IS NOT NULL) AND (c.customer\_id in

(

SELECT c.customer\_id FROM

customer as c inner join orders as o

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

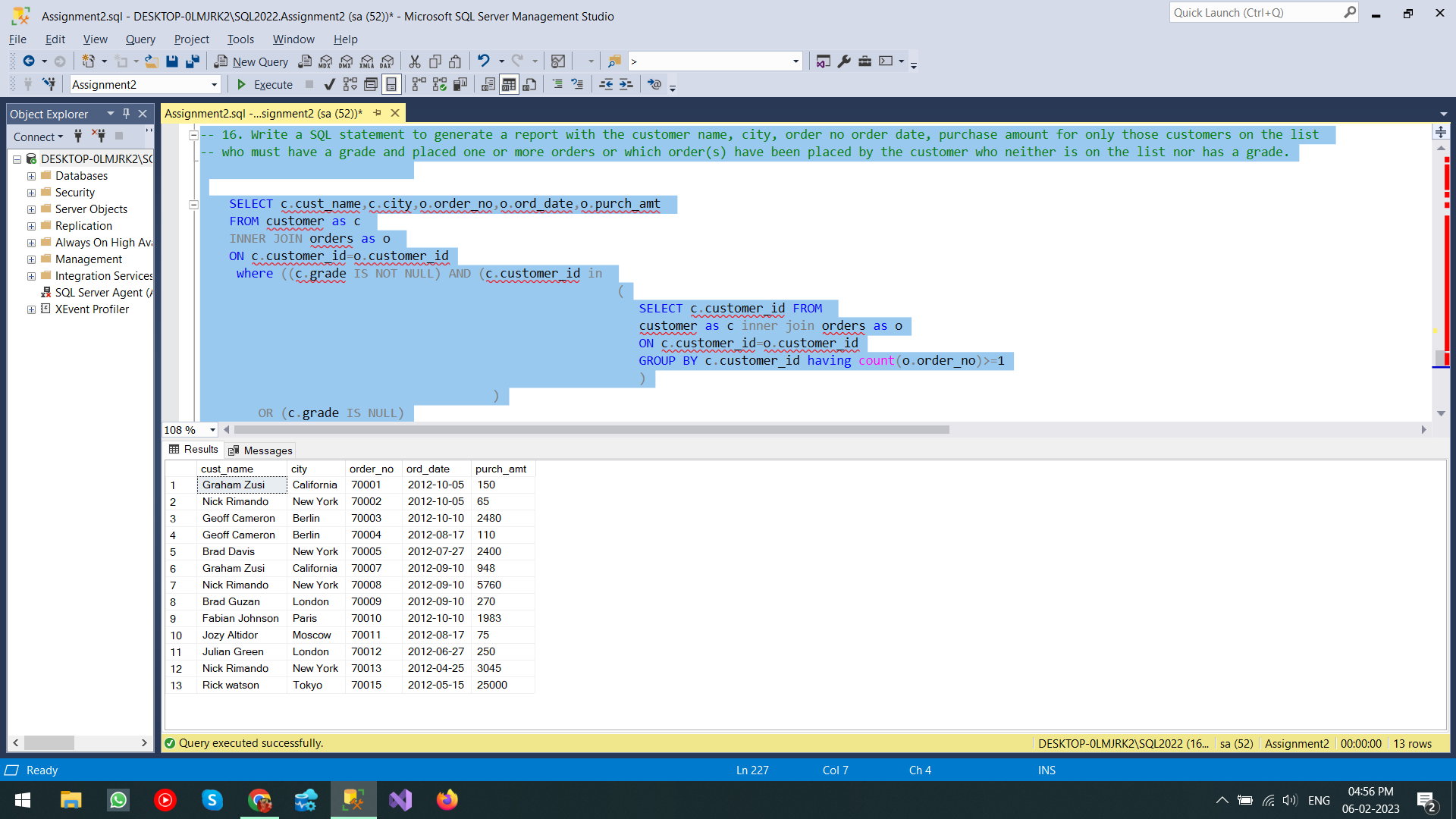
GROUP BY c.customer\_id having count(o.order\_no)>=1

)

)

OR (c.grade IS NULL)

);

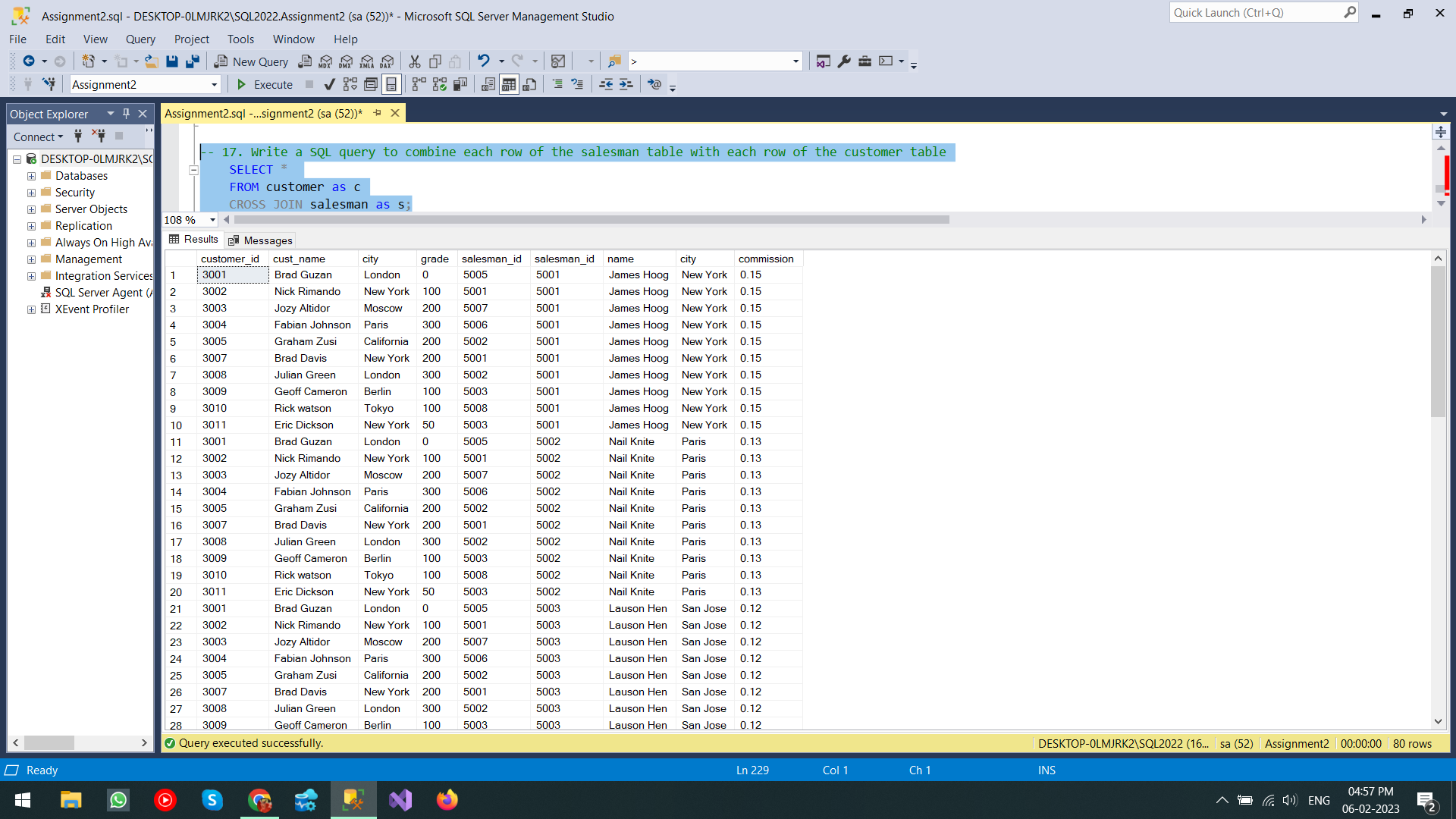


-- 17. Write a SQL query to combine each row of the salesman table with each row of the customer table

SELECT \*

FROM customer as c

CROSS JOIN salesman as s;



-- 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

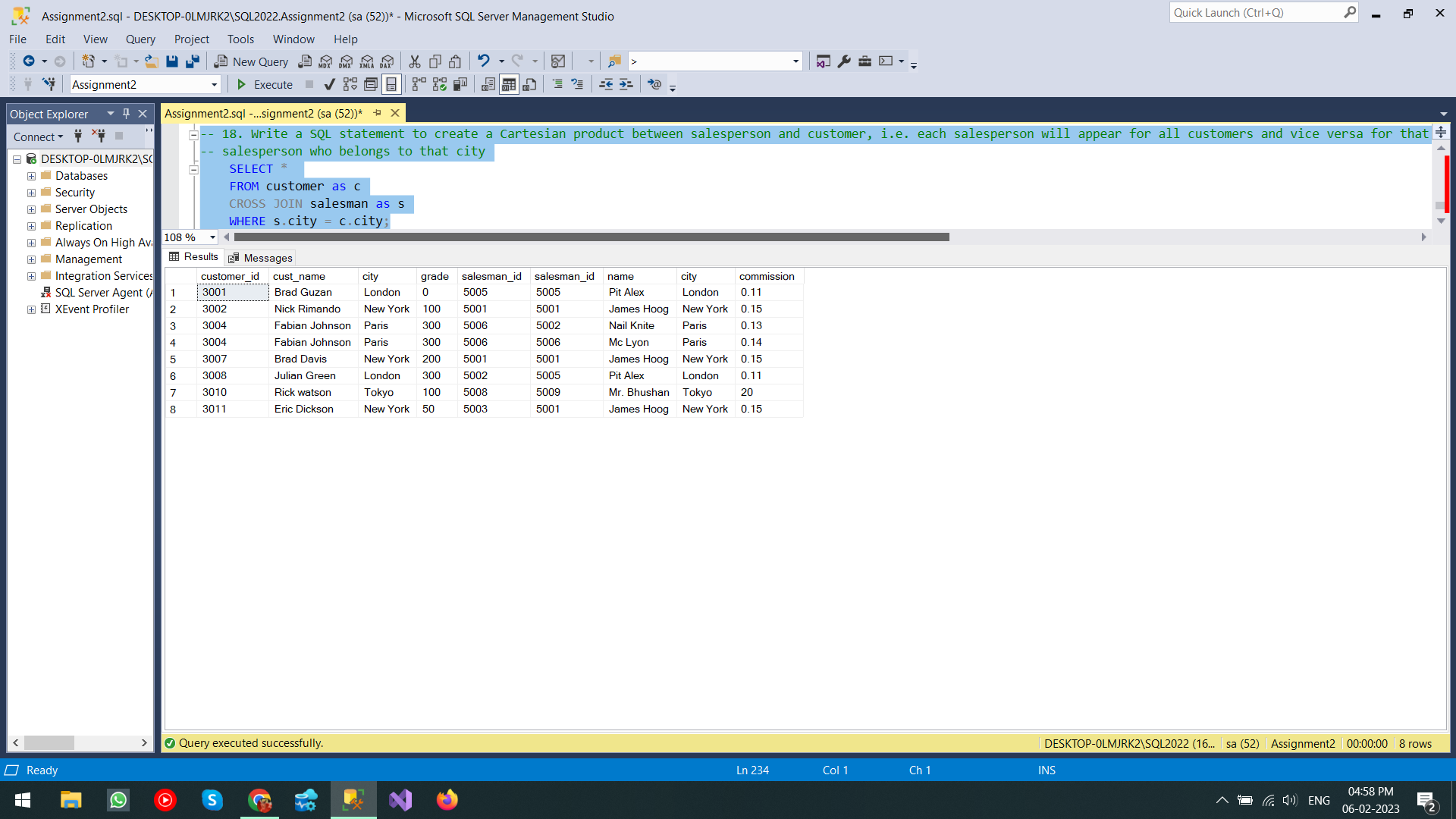
-- salesperson who belongs to that city

SELECT \*

FROM customer as c

CROSS JOIN salesman as s

WHERE s.city = c.city;



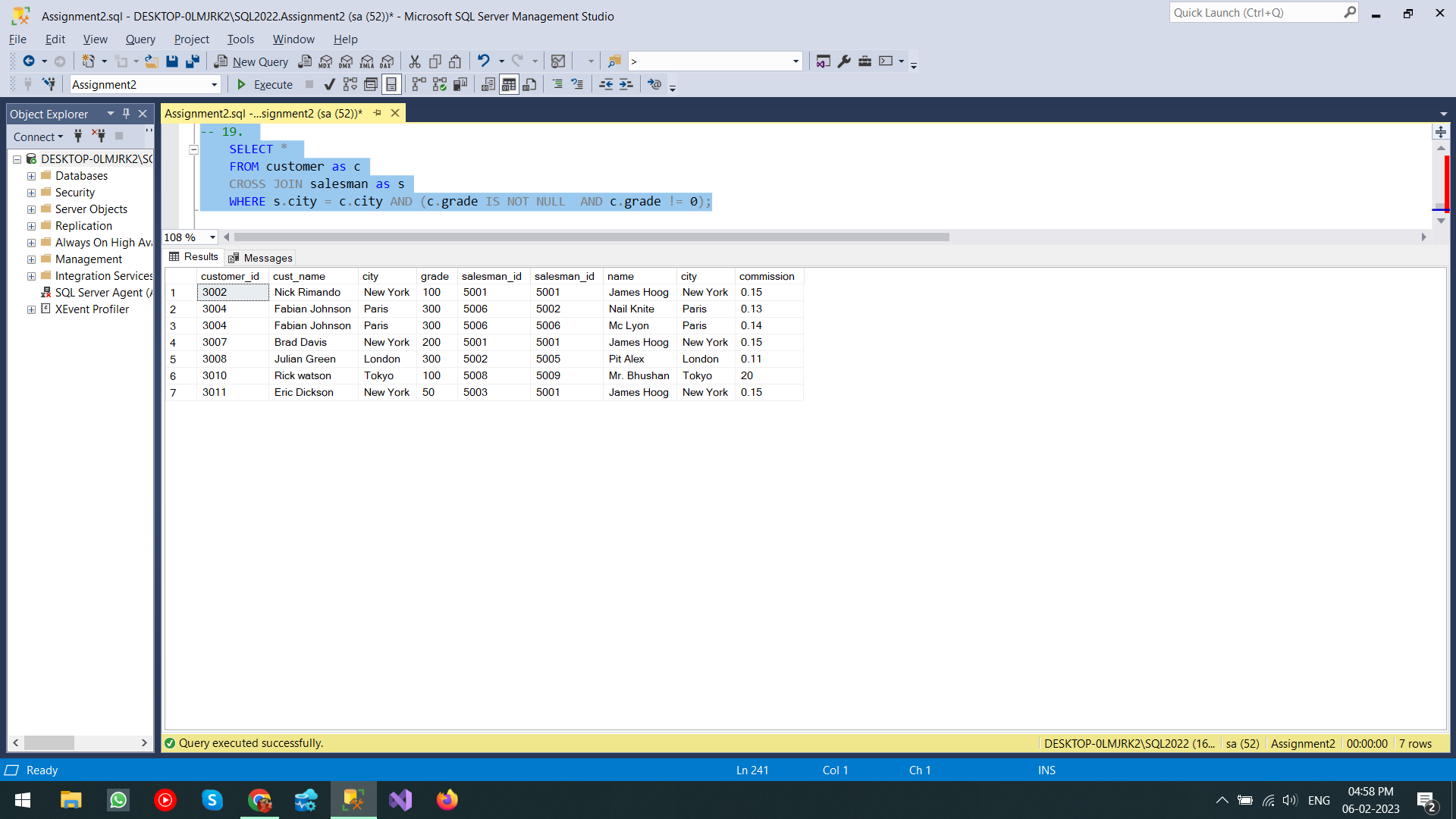
-- 19.

SELECT \*

FROM customer as c

CROSS JOIN salesman as s

WHERE s.city = c.city AND (c.grade IS NOT NULL AND c.grade != 0);



-- 20.

SELECT \*

FROM customer as c

CROSS JOIN salesman as s

WHERE s.city != c.city AND (c.grade IS NOT NULL AND c.grade != 0);

