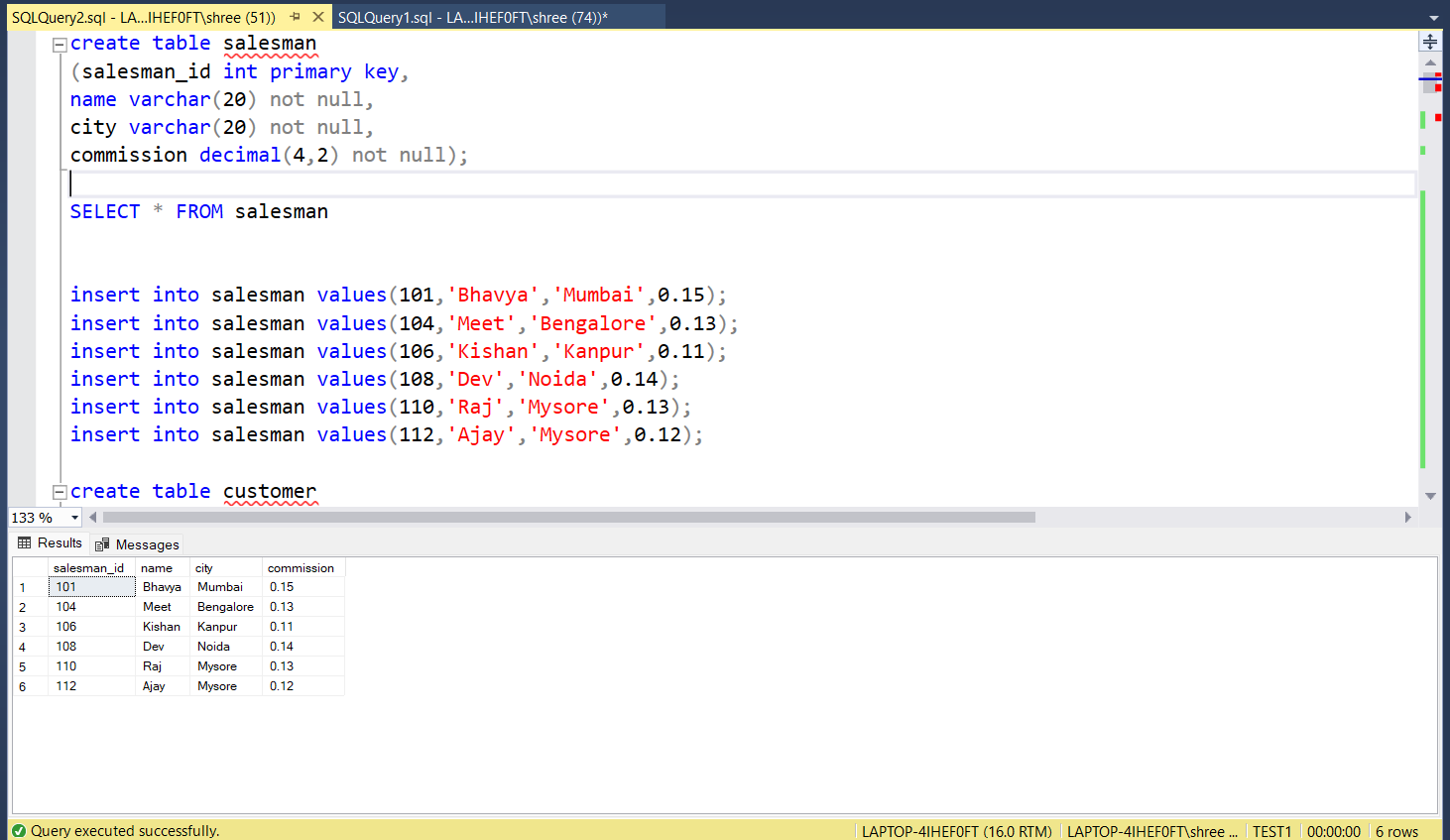
**SQL ASSIGNMENT-2**

**Creating Database: -**

****

**SQL-Queries**

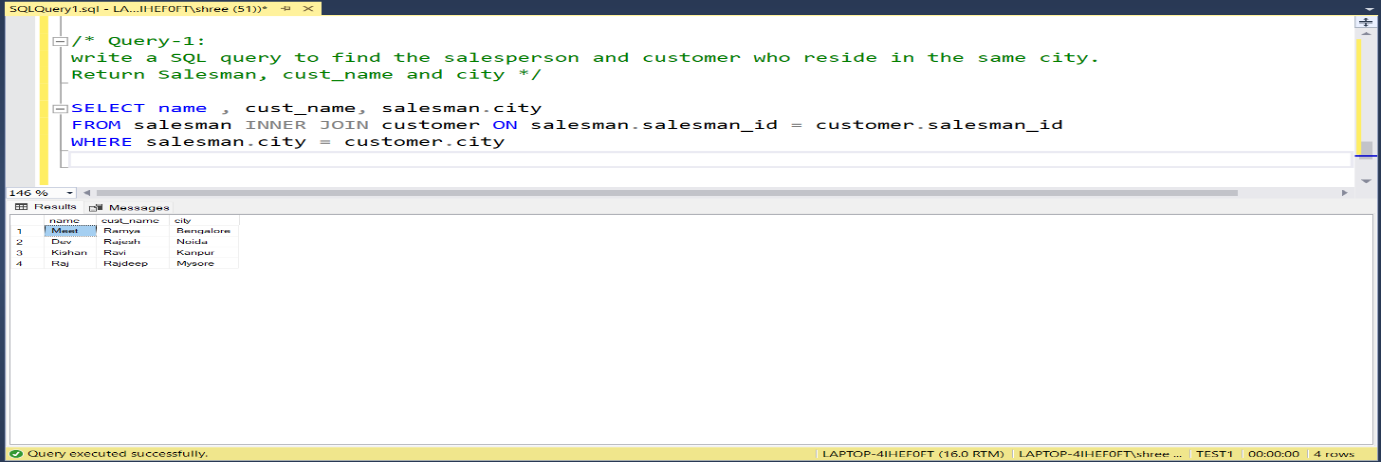
**1) Write a SQL query to find the salesperson and customer who reside in the same city. Return Salesman, cust\_name and city.**

Answer: -

SELECT name , cust\_name, salesman.city

FROM salesman INNER JOIN customer ON salesman.salesman\_id = customer.salesman\_id

WHERE salesman.city = customer.city



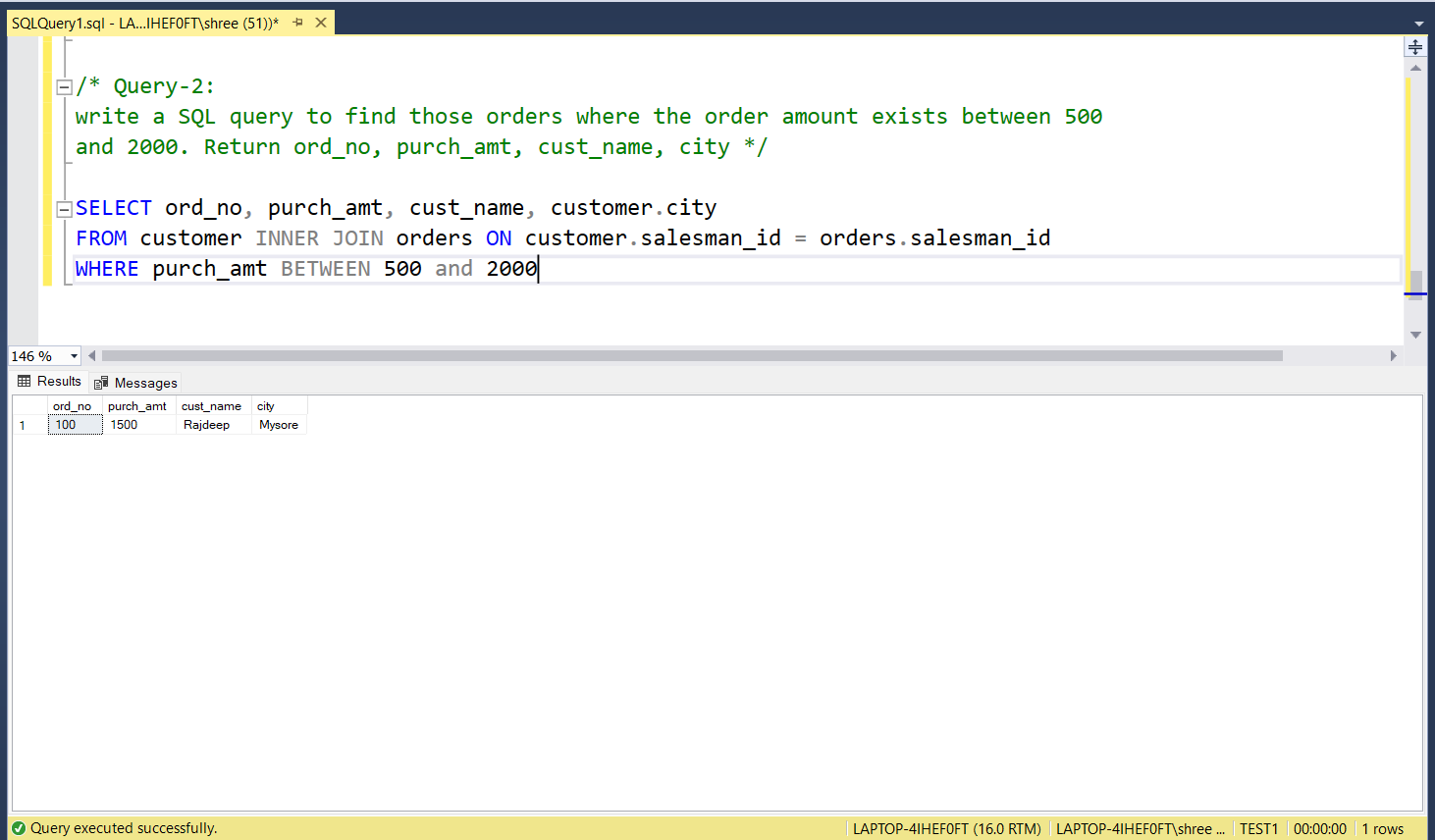
**2) Write a SQL query to find those orders where the order amount exists between 500 and 2000. Return ord\_no, purch\_amt, cust\_name, city.**

Answer: -

SELECT ord\_no, purch\_amt, cust\_name, customer.city

FROM customer INNER JOIN orders ON customer.salesman\_id = orders.salesman\_id

WHERE purch\_amt BETWEEN 500 and 2000

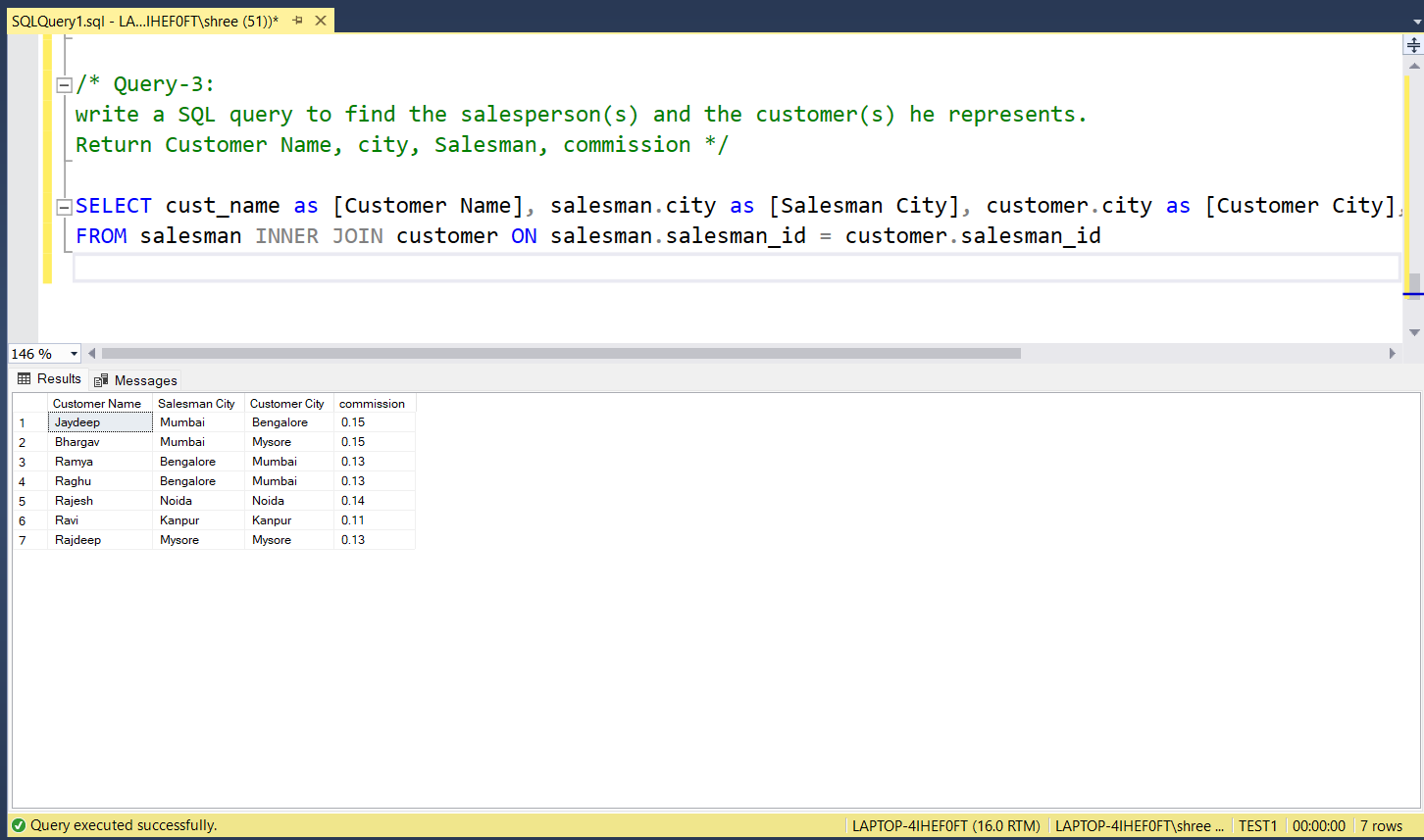


**3) Write a SQL query to find the salesperson(s) and the customer(s) he represents.Return Customer Name, city, Salesman, commission.**

Answer: -

SELECT cust\_name as [Customer Name], salesman.city as [Salesman City], customer.city as [Customer City], commission

FROM salesman INNER JOIN customer ON salesman.salesman\_id = customer.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.**

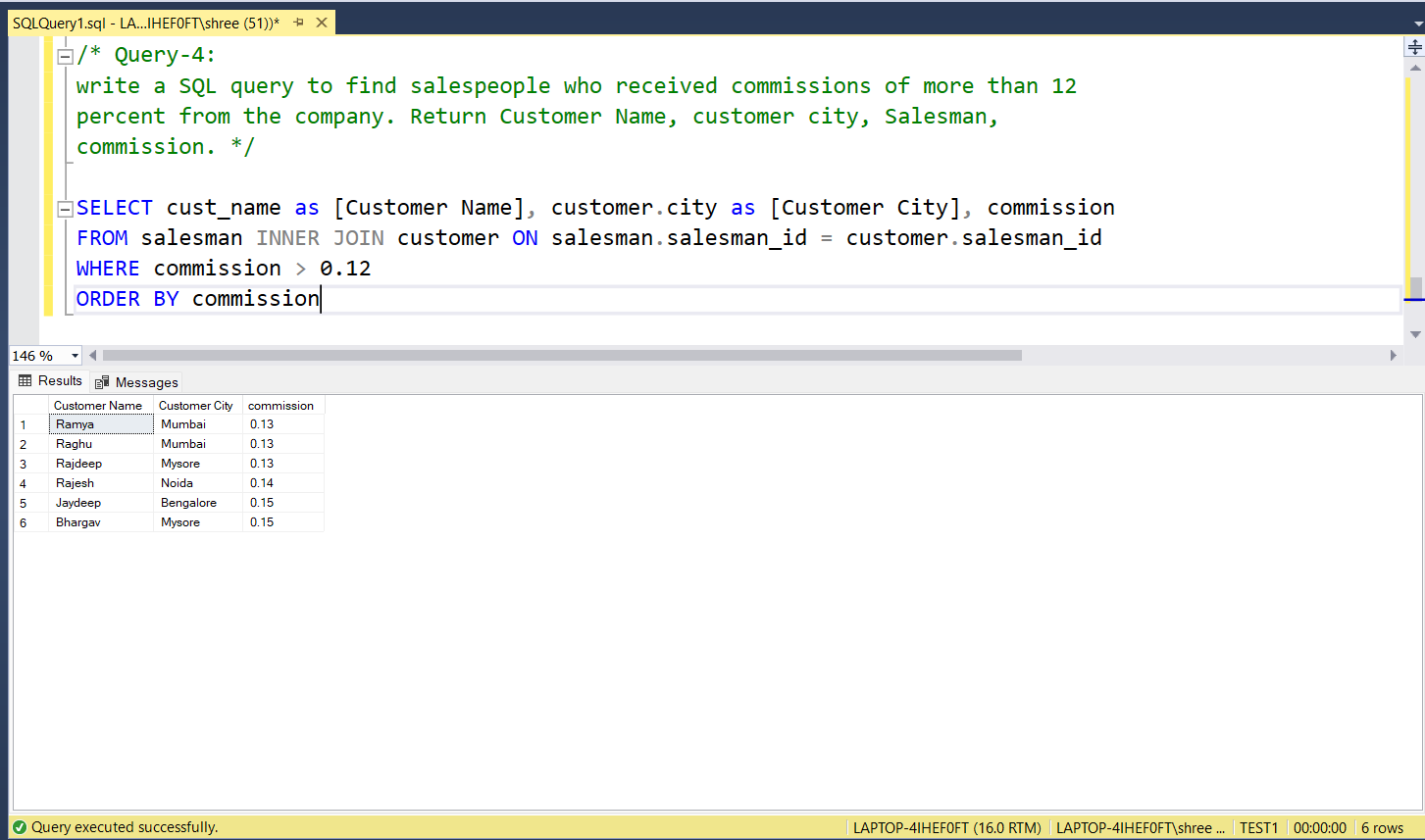
Answer: -

SELECT cust\_name as [Customer Name], customer.city as [Customer City], commission

FROM salesman INNER JOIN customer ON salesman.salesman\_id = customer.salesman\_id

WHERE commission > 0.12

ORDER BY 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.**

Answer: -

SELECT

cust\_name as [Customer Name],

customer.city as [Customer City],

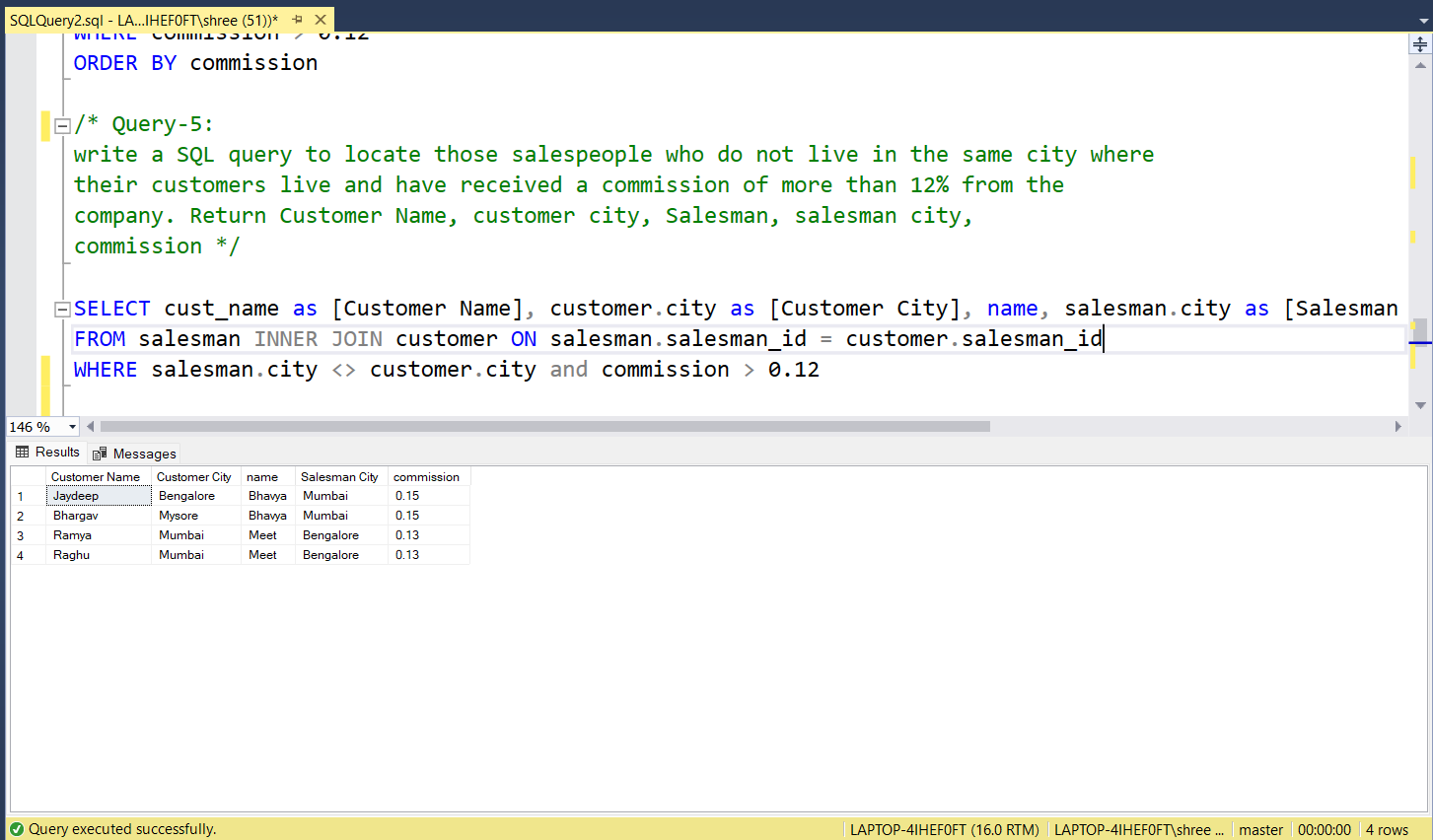
name,

salesman.city as [Salesman City],

commission

FROM salesman INNER JOIN customer ON salesman.salesman\_id = customer.salesman\_id

WHERE salesman.city <> customer.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.**

Answer: -

SELECT

ord\_no,

ord\_date,

purch\_amt,

cust\_name as [Customer Name],

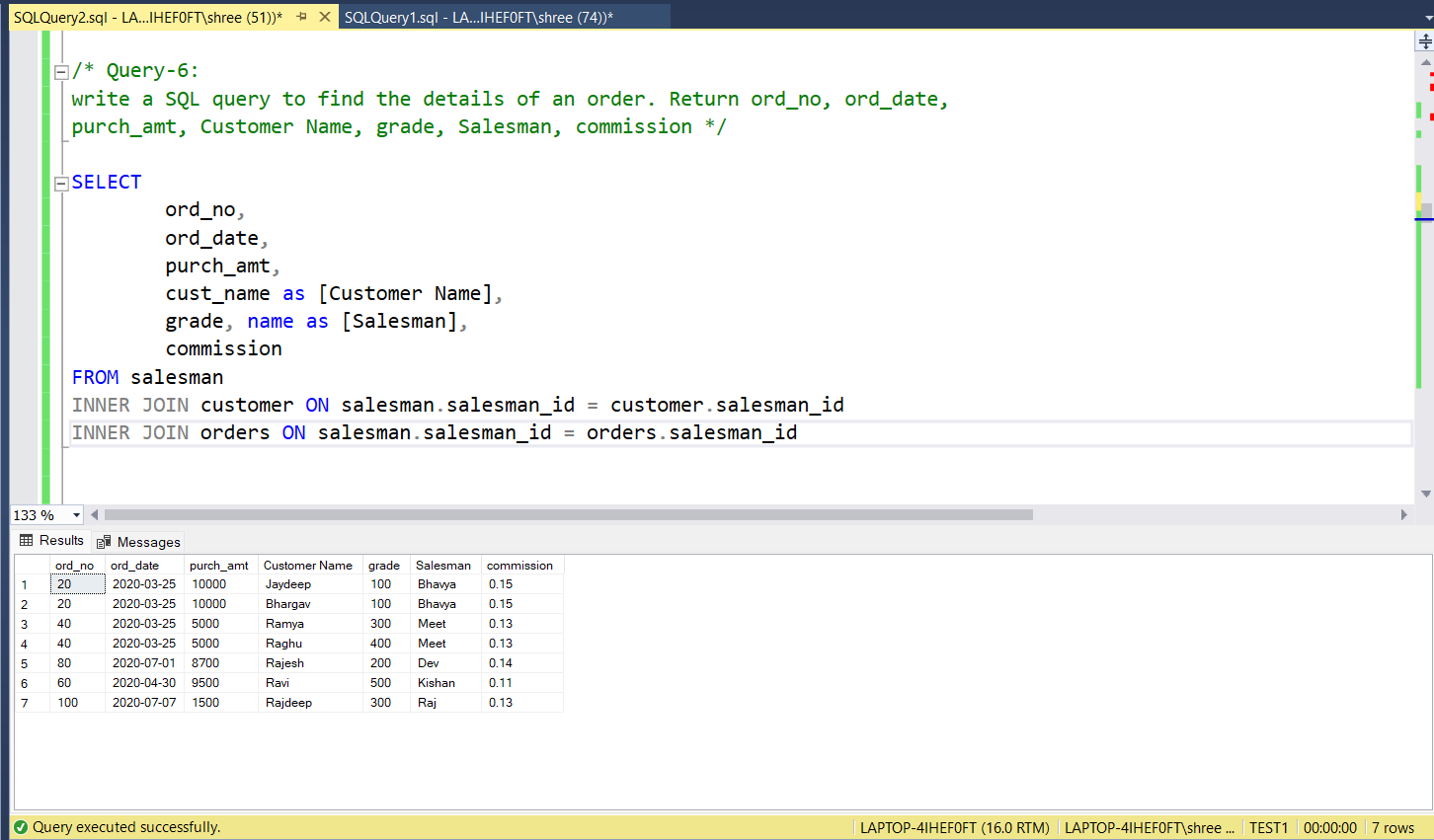
grade, name as [Salesman],

commission

FROM salesman

INNER JOIN customer ON salesman.salesman\_id = customer.salesman\_id

INNER JOIN orders ON salesman.salesman\_id = orders.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.**

Answer: -

SELECT

name,

salesman.city,

commission

cust\_name,

customer.city,

grade,

ord\_no,

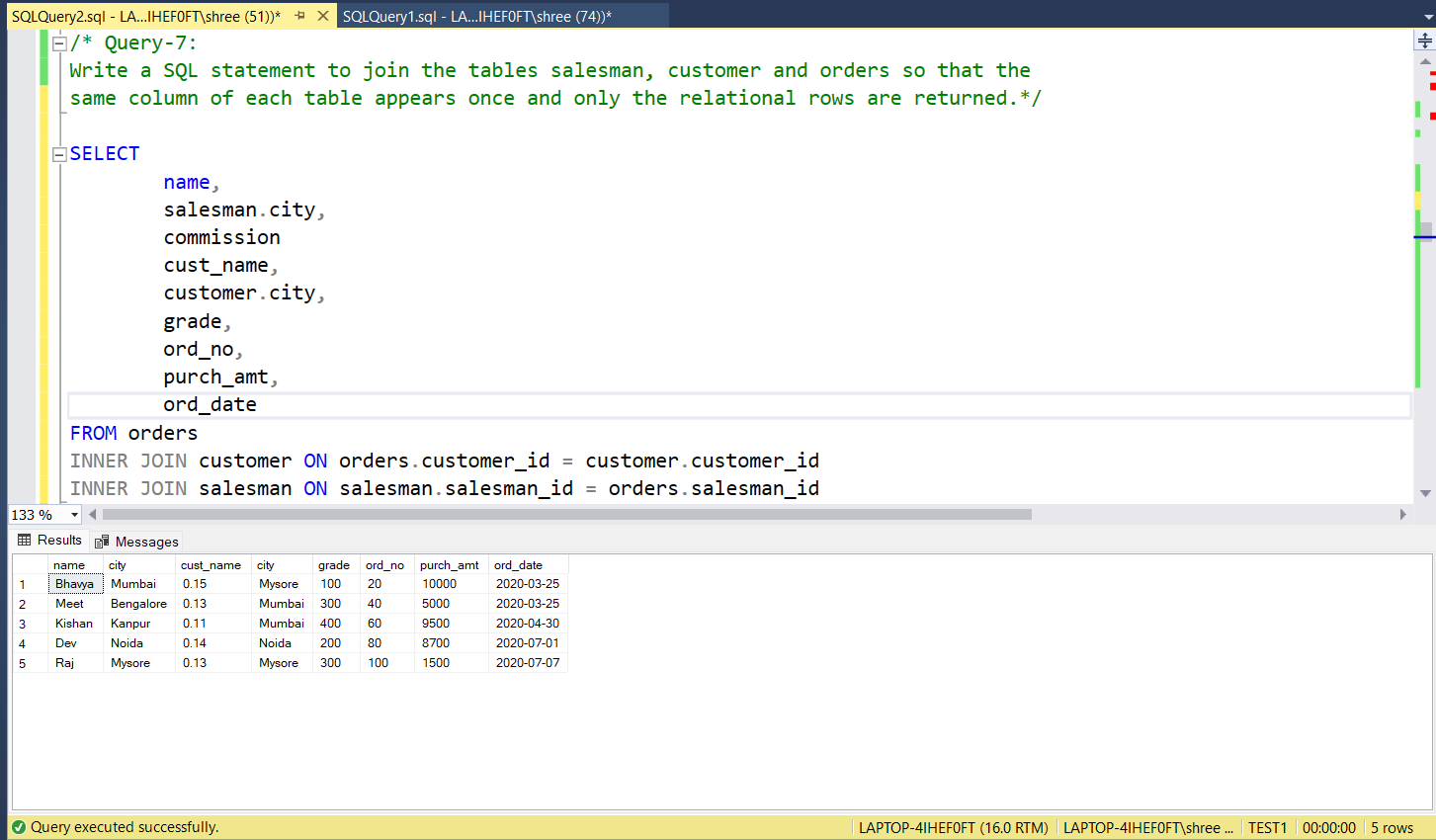
purch\_amt,

ord\_date

FROM orders

INNER JOIN customer ON orders.customer\_id = customer.customer\_id

INNER JOIN salesman ON salesman.salesman\_id = orders.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.**

Answer: -

SELECT

cust\_name as [Customer Name],

customer.city as [Customer City],

grade,

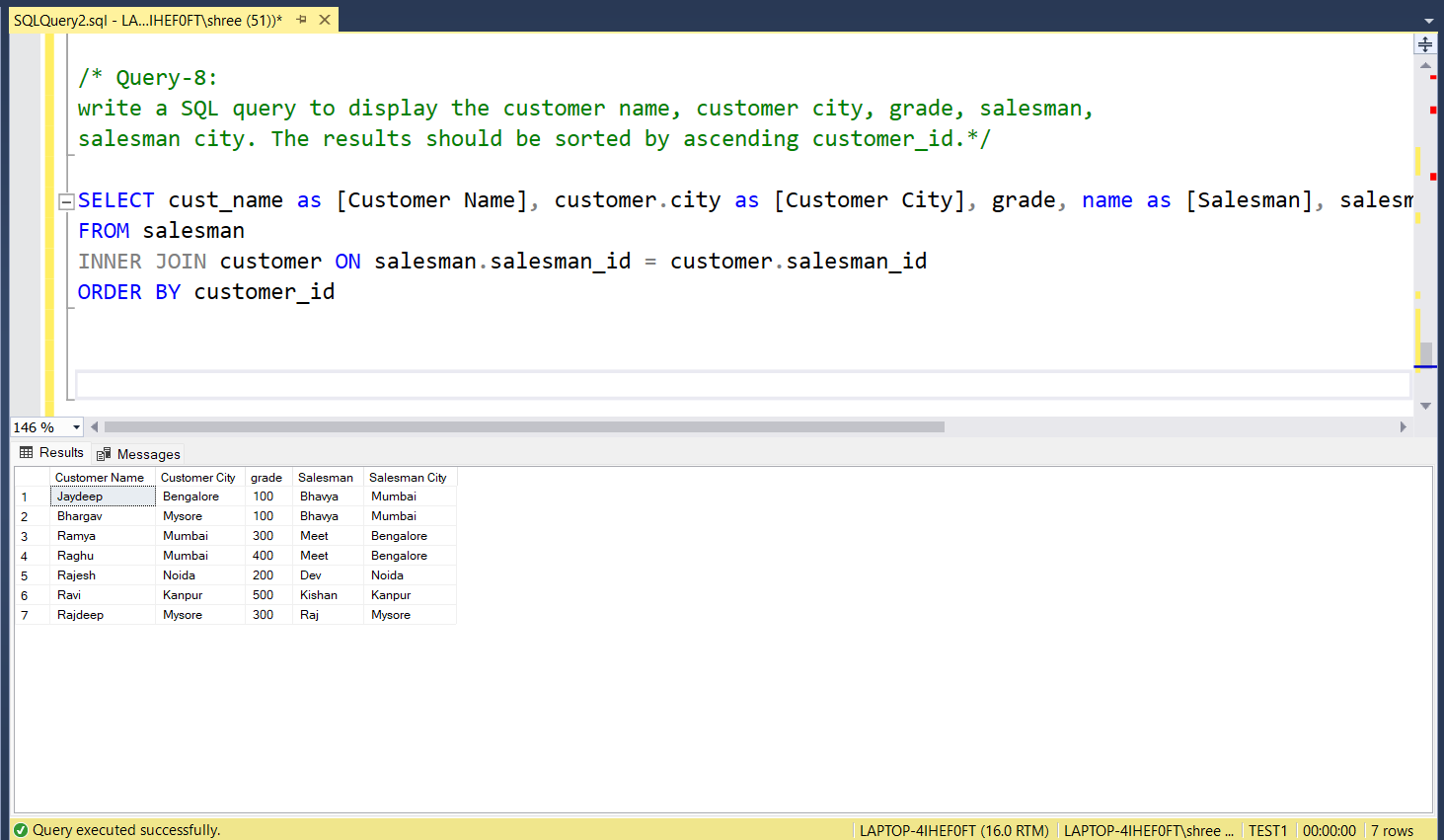
name as [Salesman],

salesman.city as [Salesman City]

FROM salesman

INNER JOIN customer ON salesman.salesman\_id = customer.salesman\_id

ORDER BY 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.**

Answer: -

SELECT

cust\_name as [Customer Name],

customer.city as [Customer City],

grade, name as [Salesman],

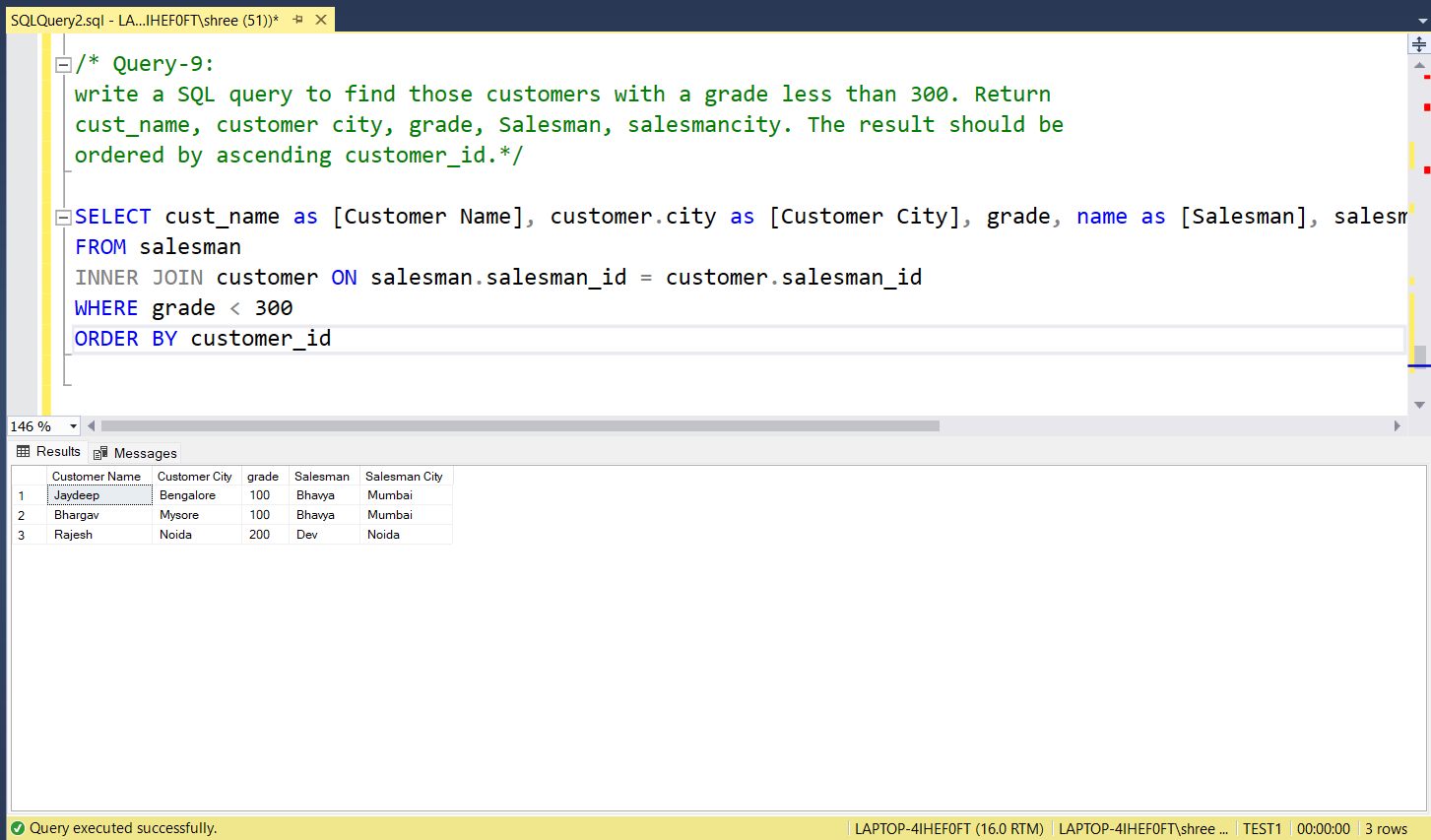
salesman.city as [Salesman City]

FROM salesman

INNER JOIN customer ON salesman.salesman\_id = customer.salesman\_id

WHERE 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.**

Answer: -

SELECT

cust\_name as [Customer Name],

customer.city as [Customer City],

ord\_no as [Order No],

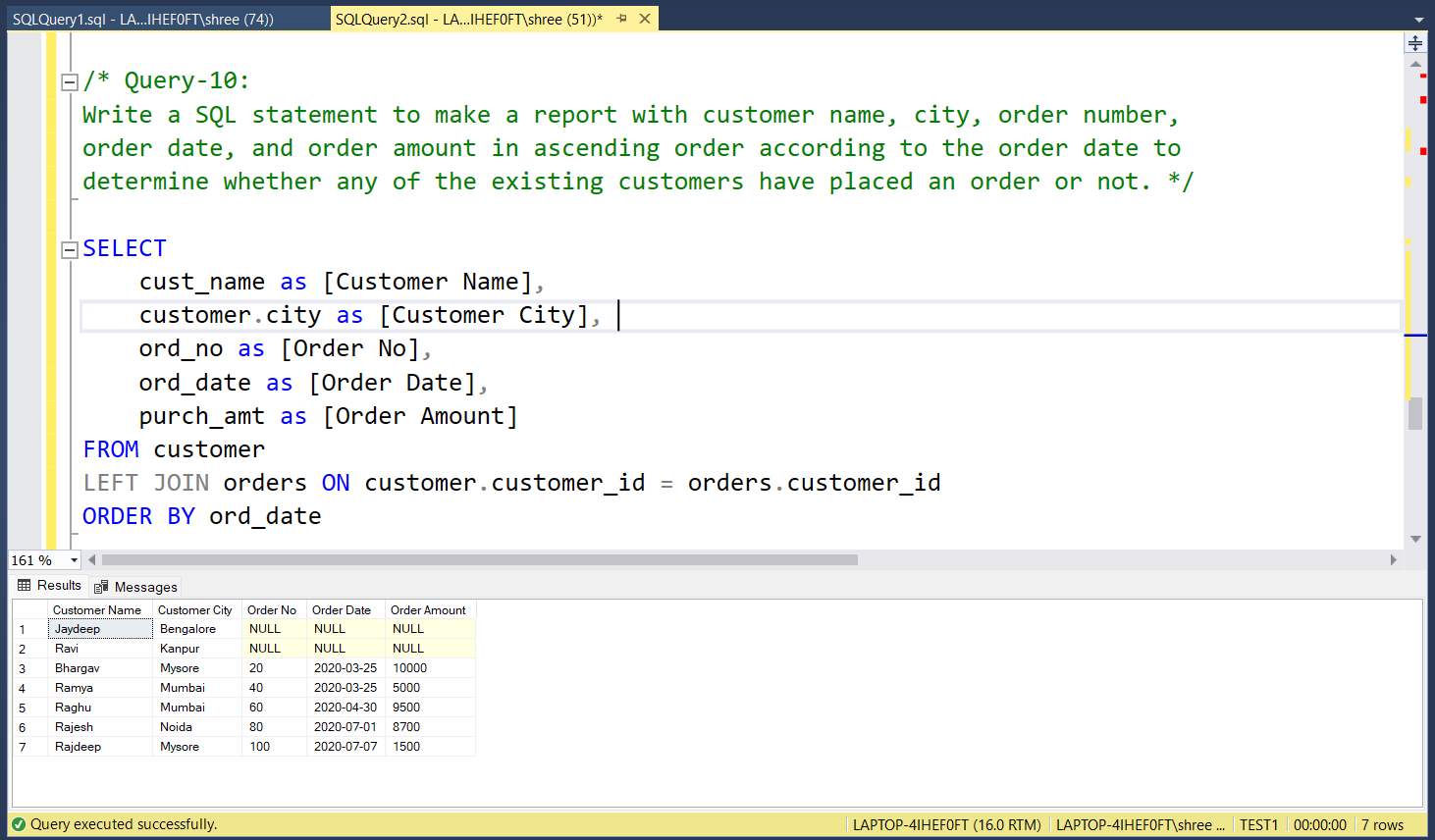
ord\_date as [Order Date],

purch\_amt as [Order Amount]

FROM customer

LEFT JOIN orders ON customer.customer\_id = orders.customer\_id

ORDER BY 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.**

Answer: -

SELECT

ord\_no,

ord\_date,

purch\_amt,

cust\_name as [Customer Name],

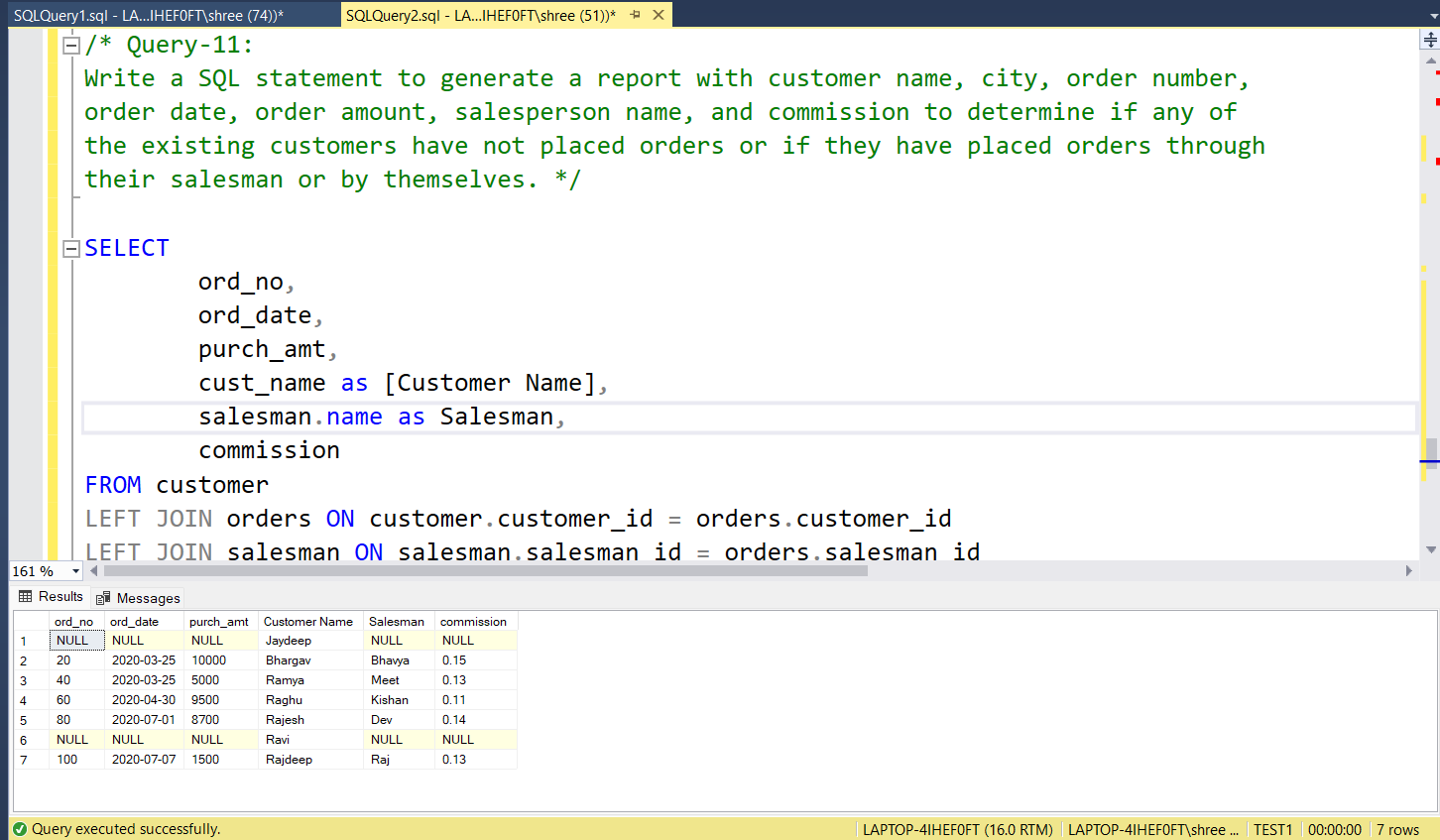
salesman.name as Salesman,

commission

FROM customer

LEFT JOIN orders ON customer.customer\_id = orders.customer\_id

LEFT JOIN salesman ON salesman.salesman\_id = orders.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.**

Answer: -

SELECT

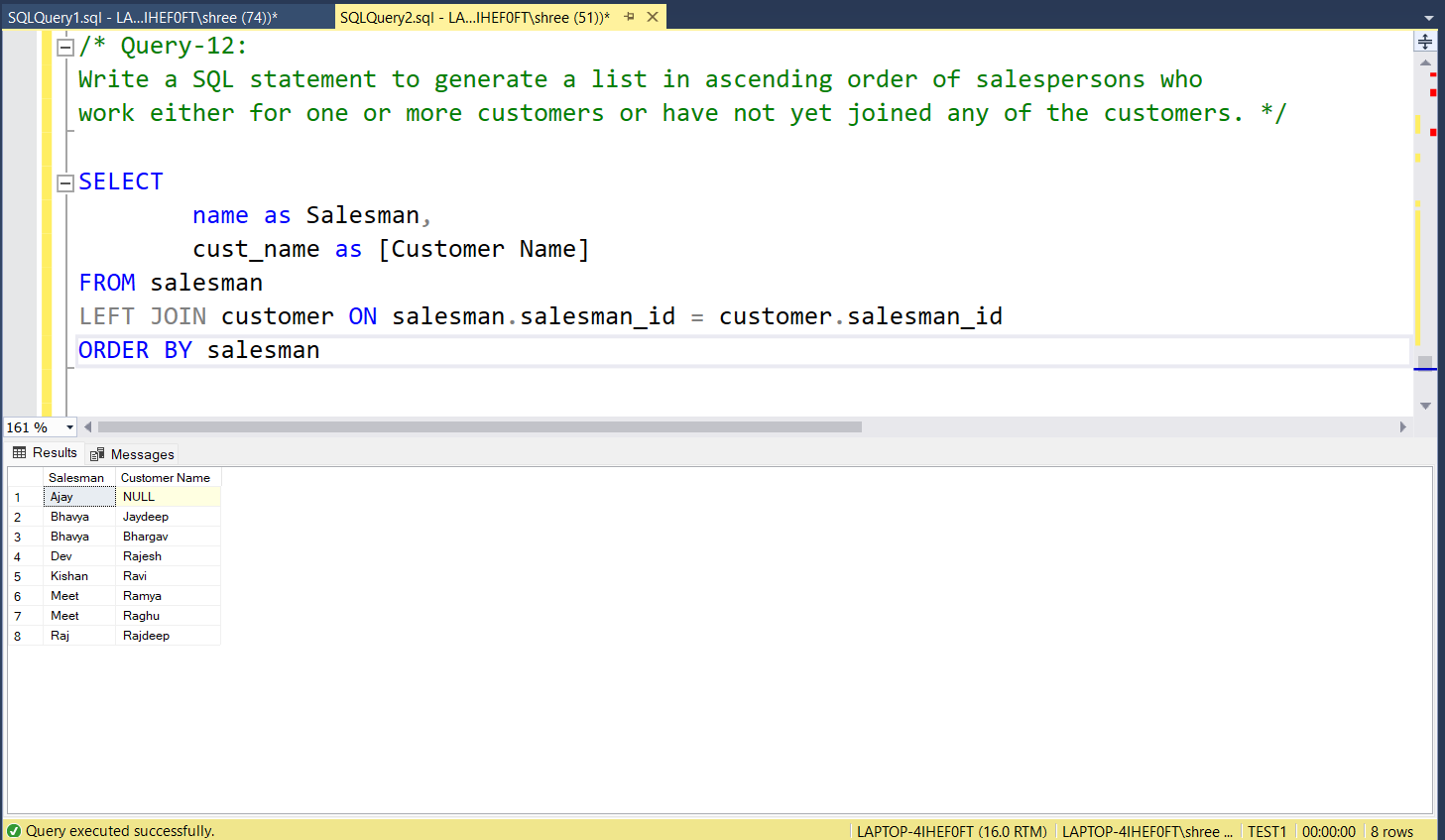
name as Salesman,

cust\_name as [Customer Name]

FROM salesman

LEFT JOIN customer ON salesman.salesman\_id = customer.salesman\_id

ORDER BY salesman



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

Answer: -

SELECT

name as Salesman,

cust\_name as [Customer Name],

customer.city as [Customer City],

grade,

ord\_no as [Order No],

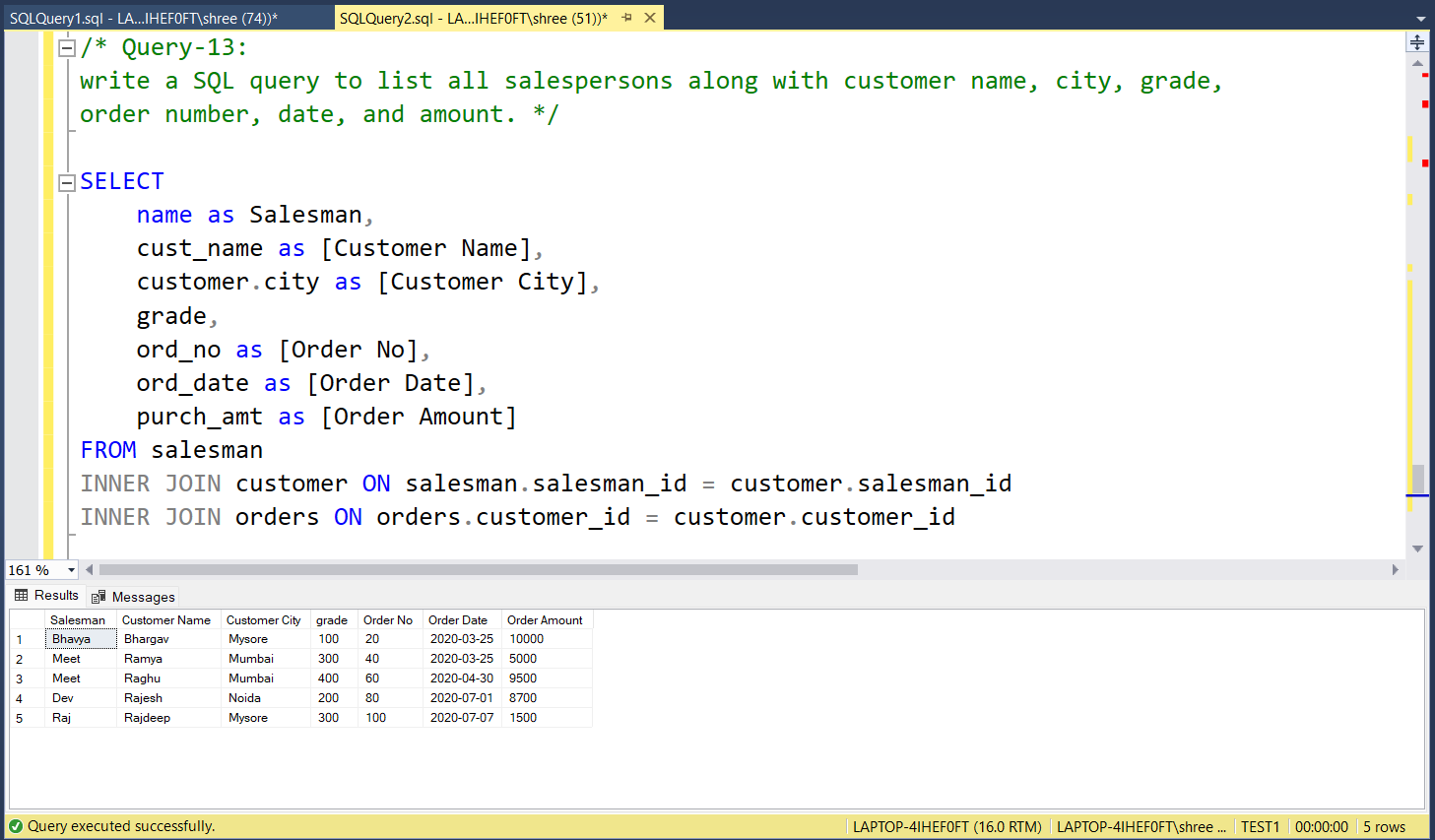
ord\_date as [Order Date],

purch\_amt as [Order Amount]

FROM salesman

INNER JOIN customer ON salesman.salesman\_id = customer.salesman\_id

INNER JOIN orders ON orders.customer\_id = customer.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.**

Answer: -

SELECT

cust\_name,

customer.city,

grade,

name AS Salesman,

ord\_no,

ord\_date,

purch\_amt

FROM customer

RIGHT OUTER JOIN salesman

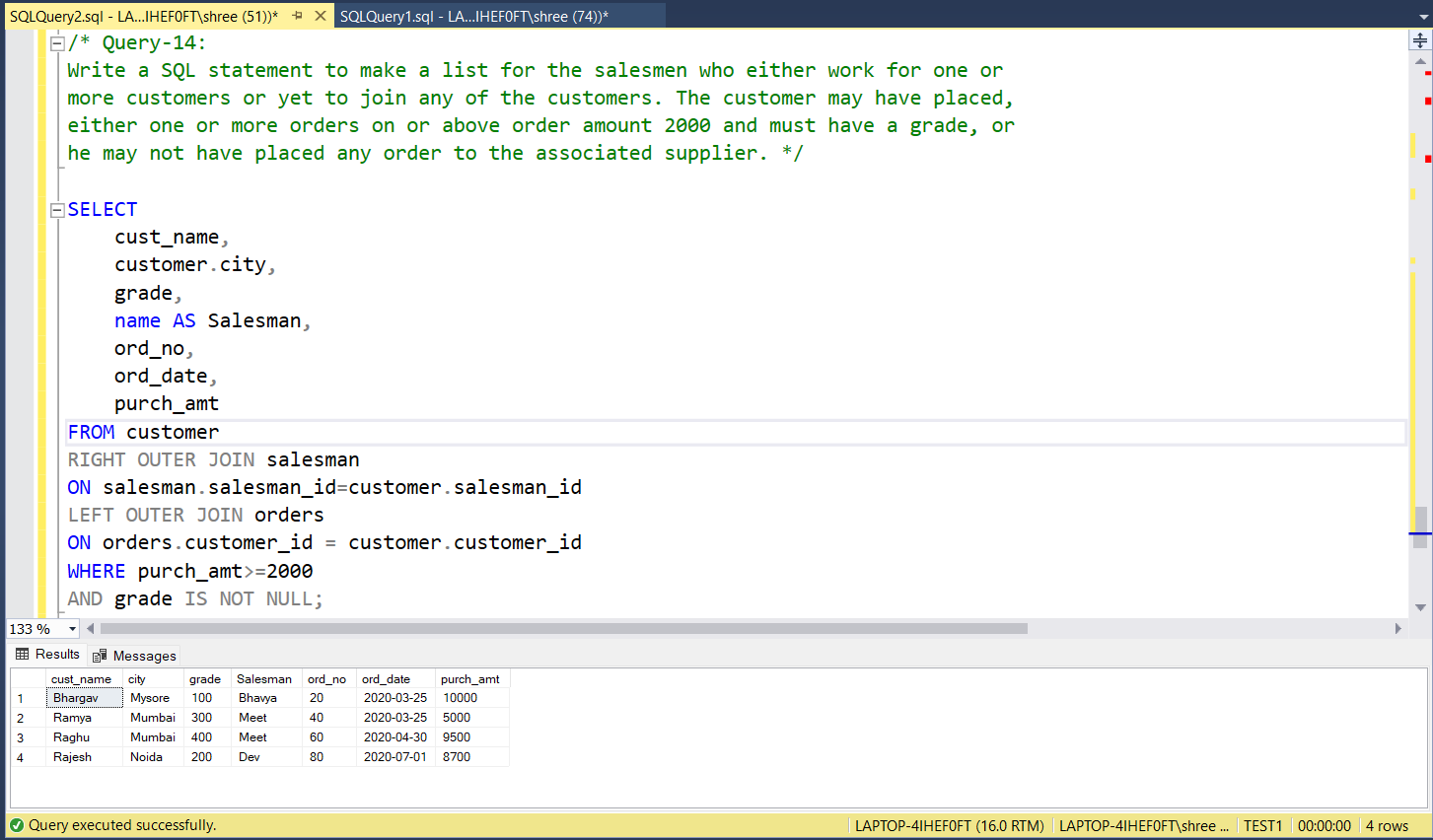
ON salesman.salesman\_id=customer.salesman\_id

LEFT OUTER JOIN orders

ON orders.customer\_id = customer.customer\_id

WHERE purch\_amt>=2000

AND 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.**

Answer: -

SELECT

cust\_name,

customer.city,

grade,

name AS Salesman,

ord\_no,

ord\_date,

purch\_amt

FROM customer

RIGHT OUTER JOIN salesman

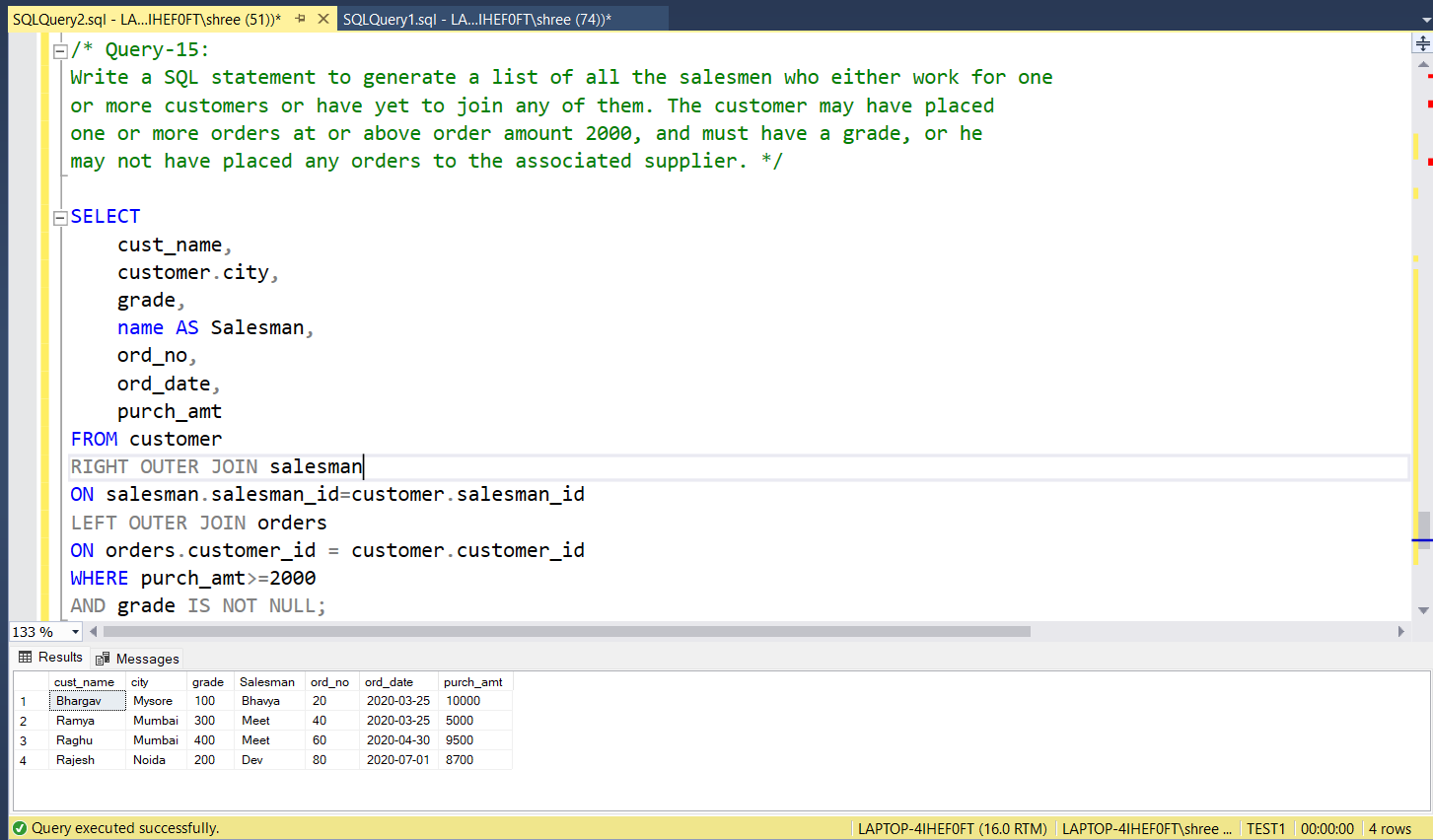
ON salesman.salesman\_id=customer.salesman\_id

LEFT OUTER JOIN orders

ON orders.customer\_id = customer.customer\_id

WHERE purch\_amt>=2000

AND 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.**

Answer: -

SELECT

cust\_name,

customer.city,

ord\_no,

ord\_date,

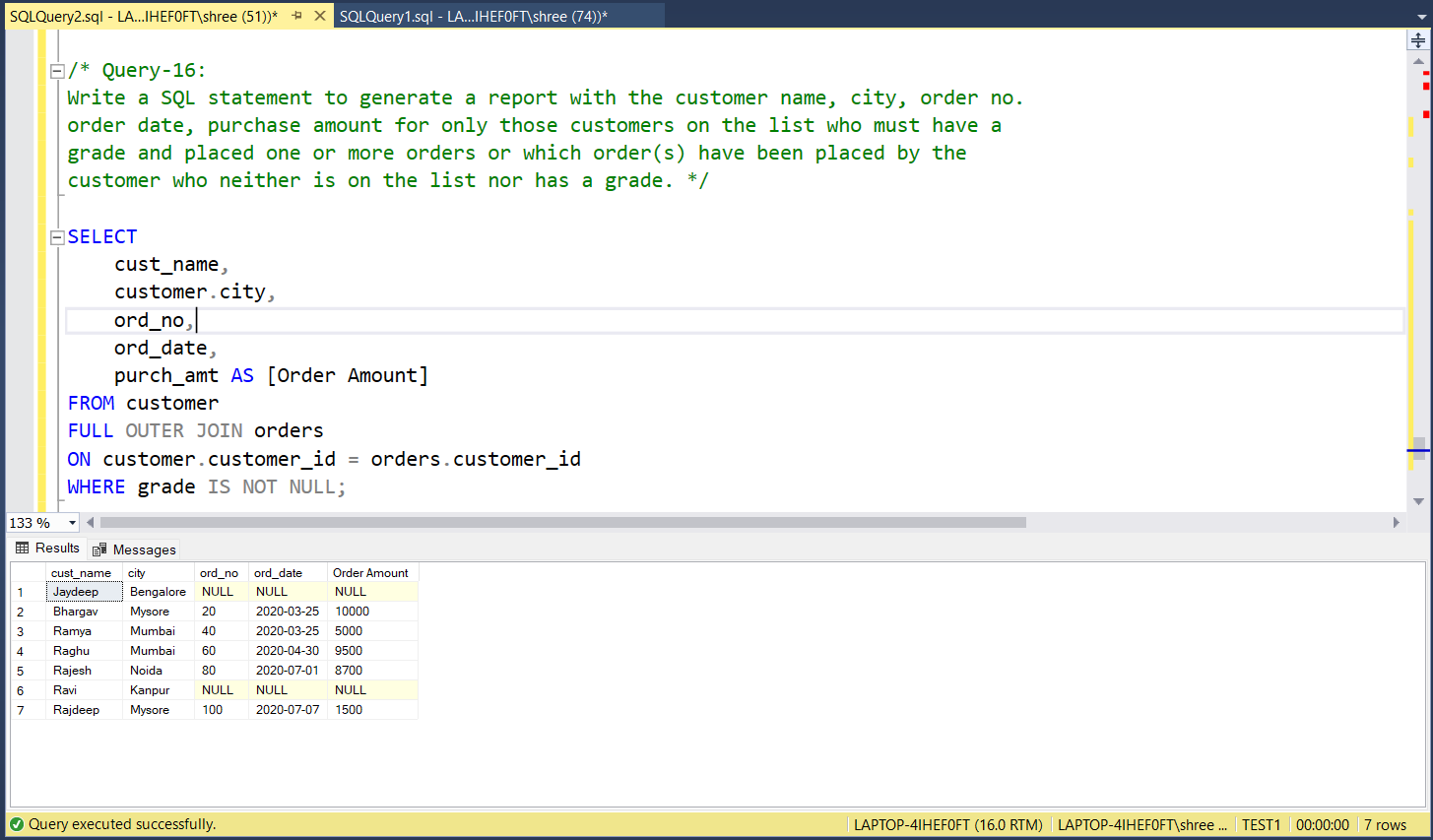
purch\_amt AS [Order Amount]

FROM customer

FULL OUTER JOIN orders

ON customer.customer\_id = orders.customer\_id

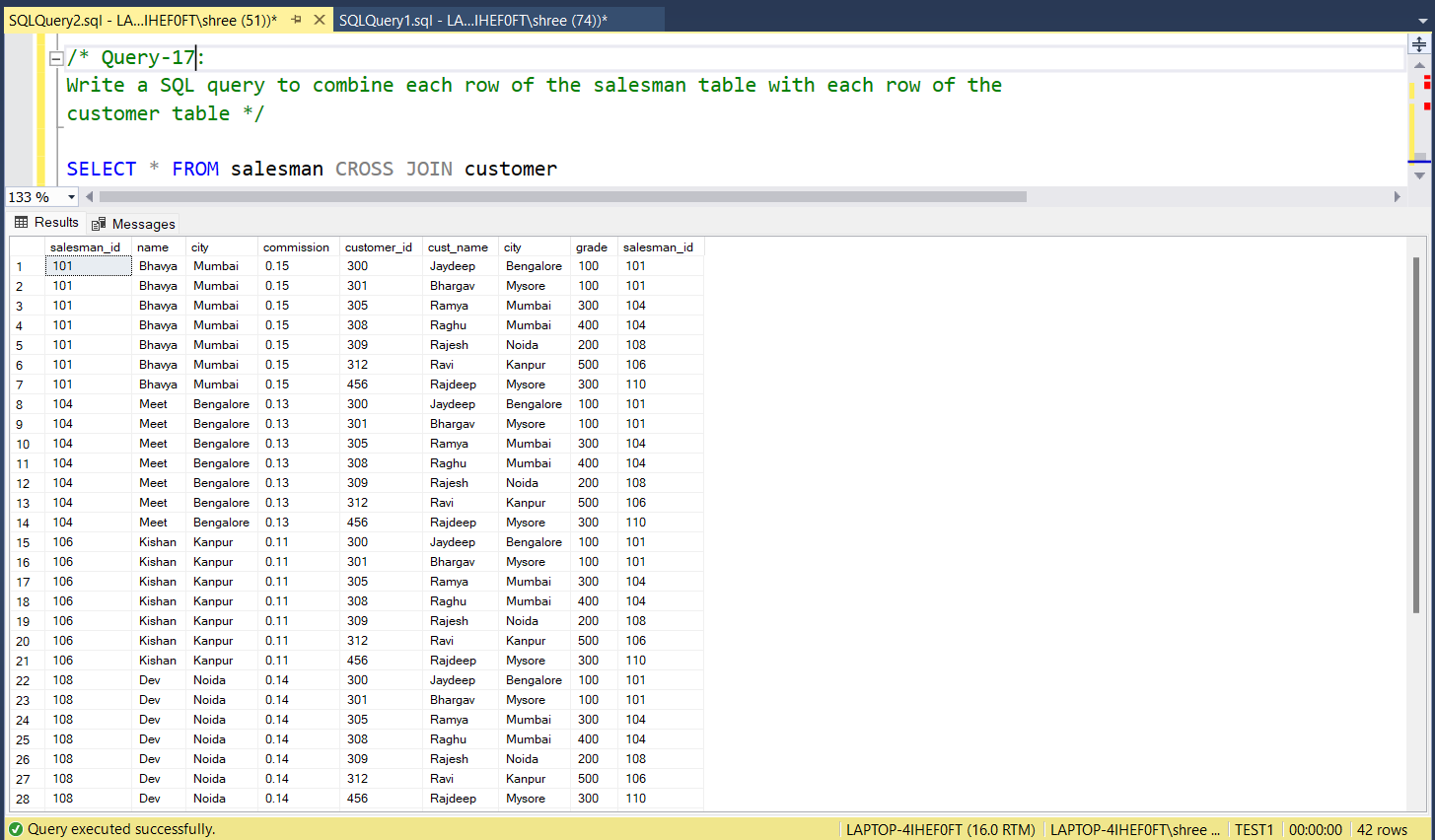
WHERE grade IS NOT NULL;



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

Answer: -

SELECT \* FROM salesman CROSS JOIN customer

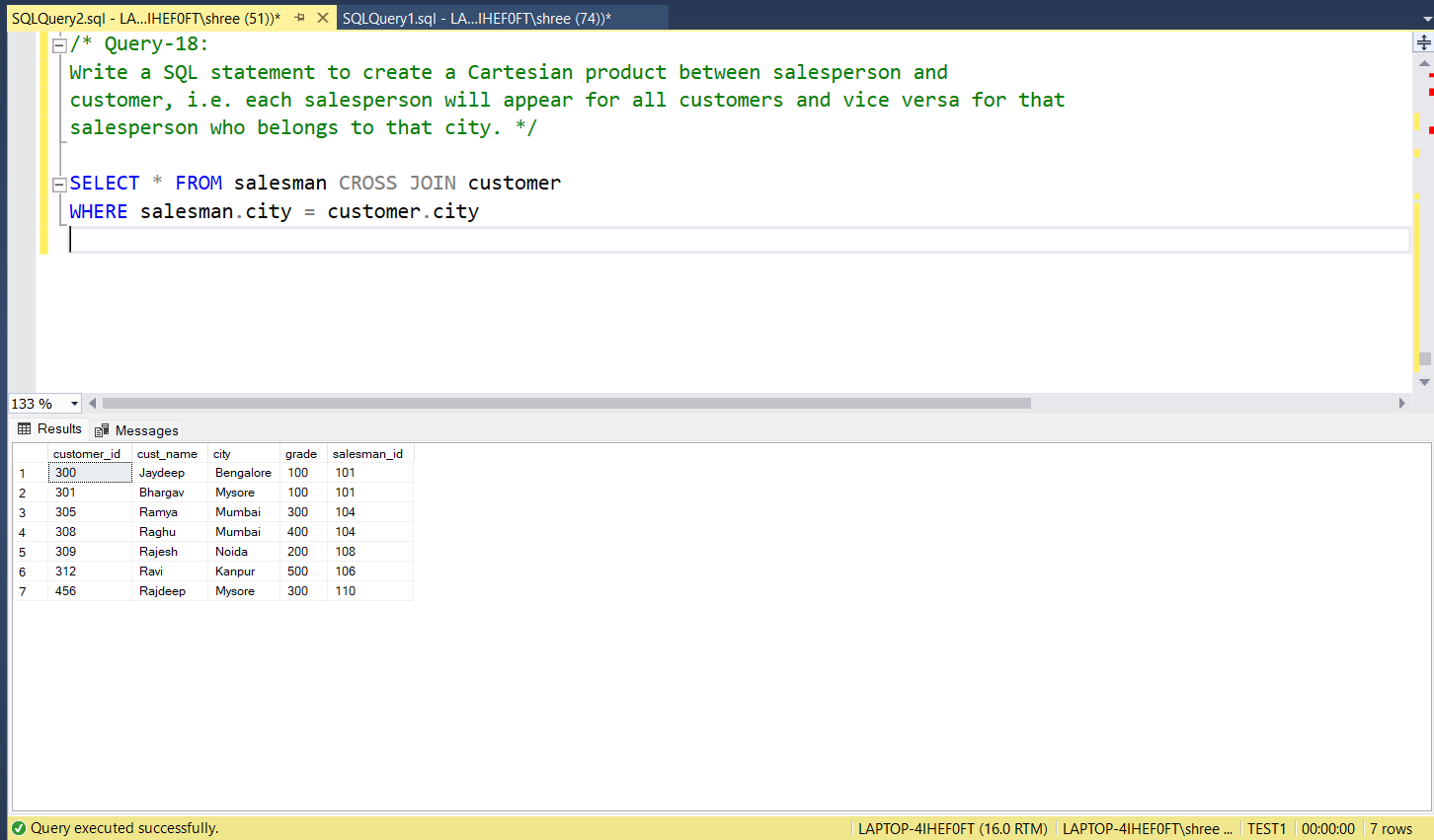


**18) Write a SQL statement to create a Cartesian product between salesperson and customer, i.e. each salesperson will appear for all customers and vice versa for that salesperson who belongs to that city.**

Answer: -

SELECT \* FROM salesman CROSS JOIN customer

WHERE salesman.city = customer.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.**

Answer: -

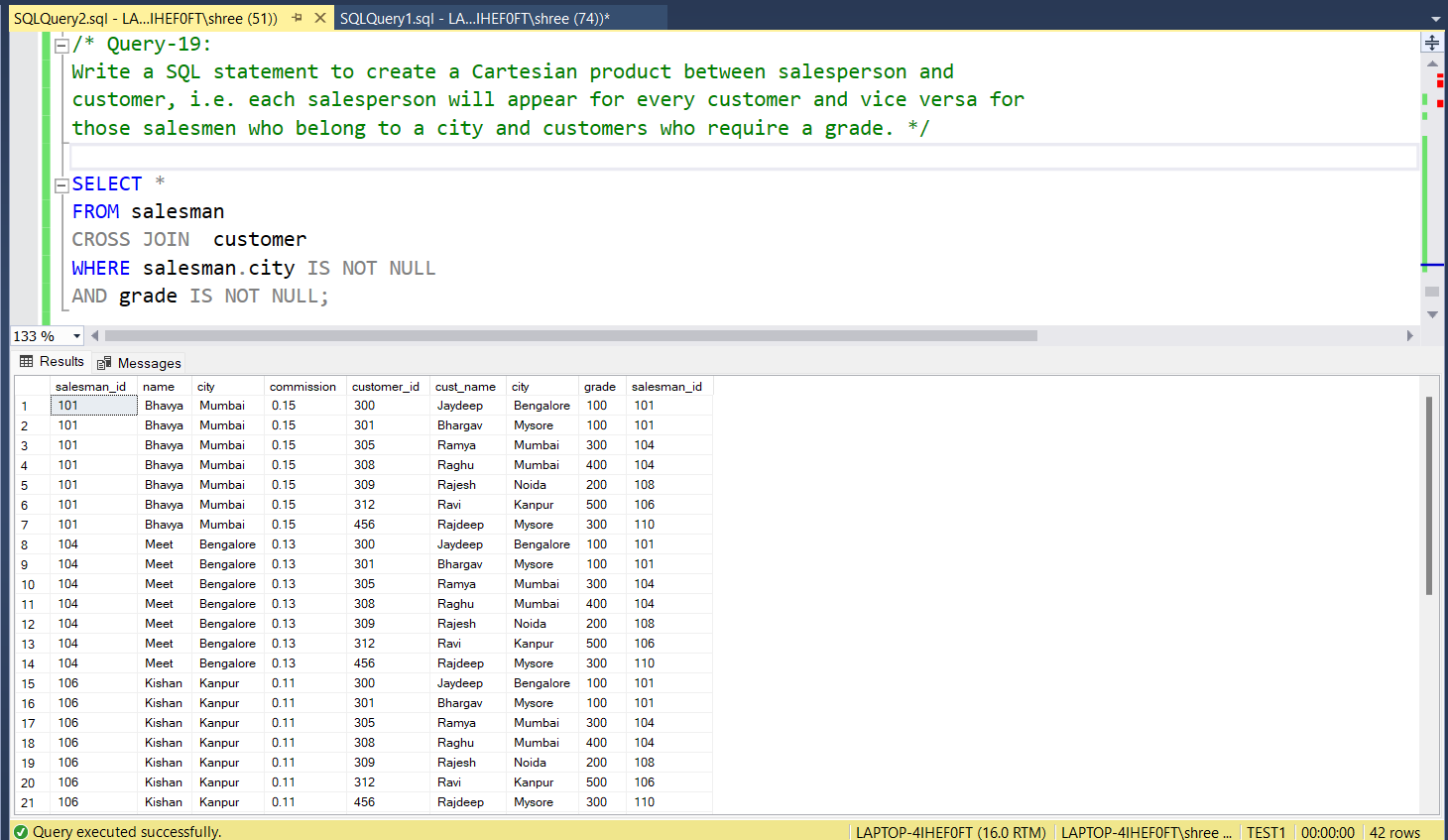
SELECT \*

FROM salesman

CROSS JOIN customer

WHERE salesman.city IS NOT NULL

AND 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.**

Answer: -

SELECT \*

FROM salesman

CROSS JOIN customer

WHERE salesman.city IS NOT NULL

AND salesman.city <> customer.city;

