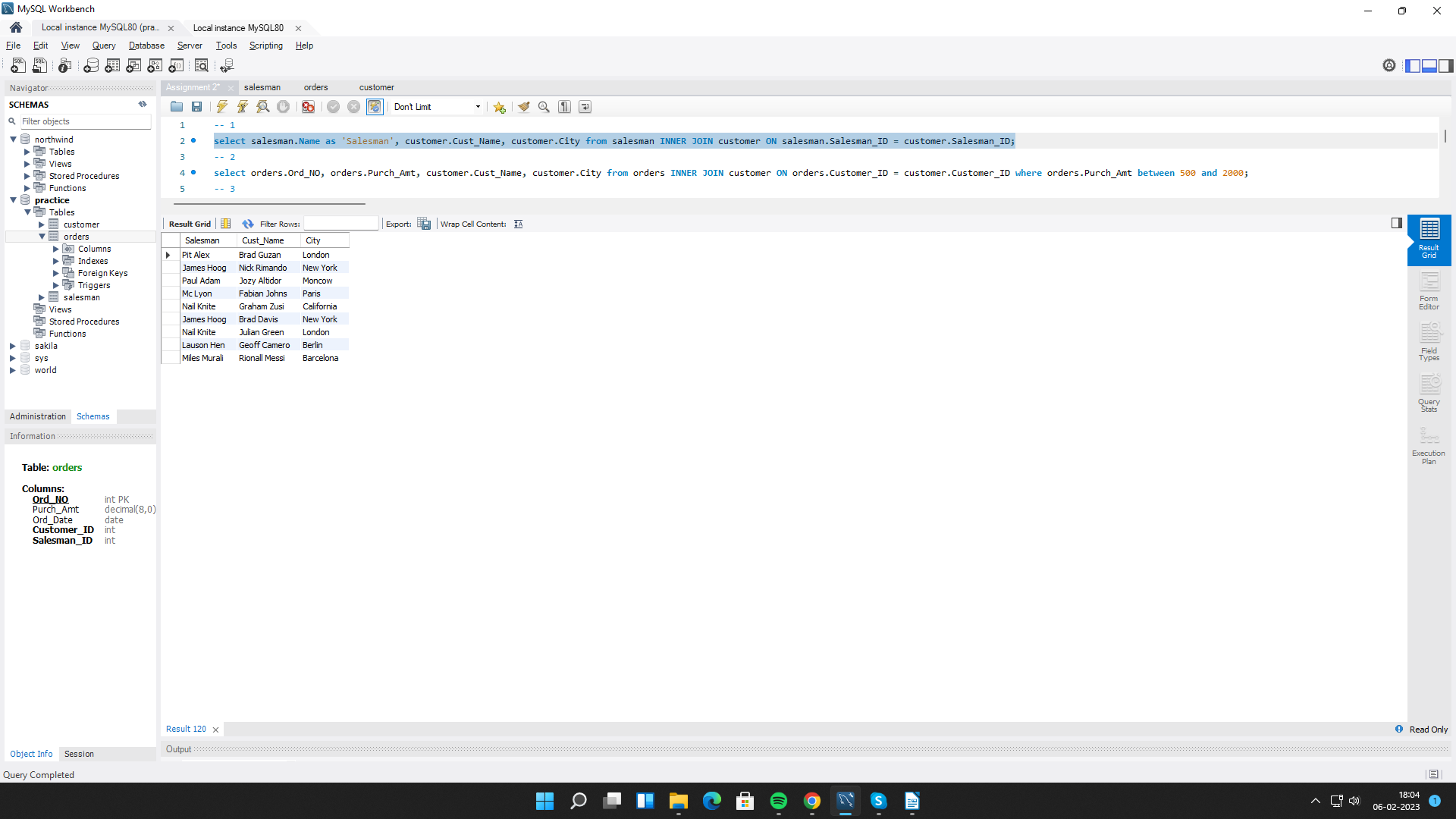
**SQL Assignment 2**

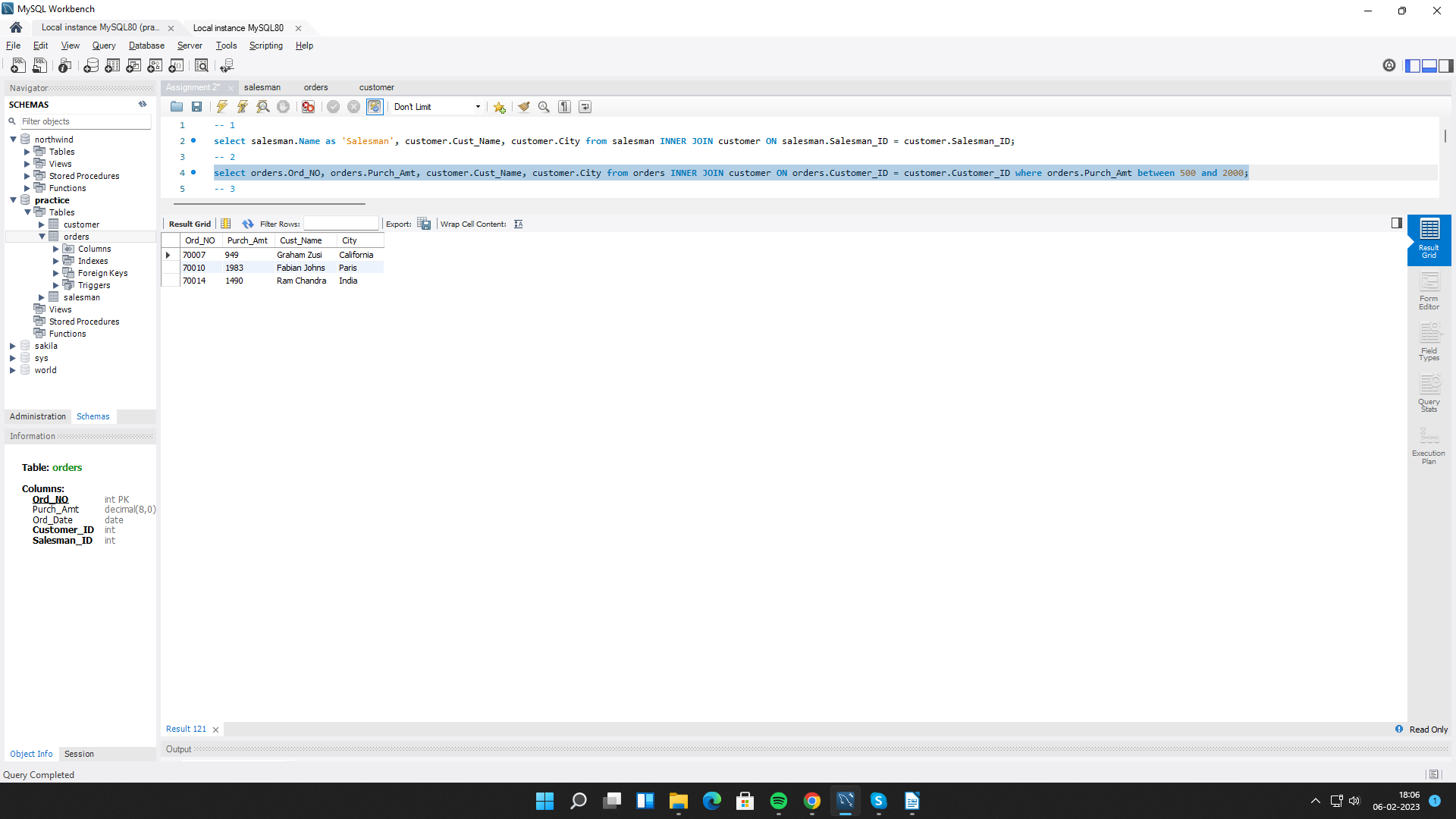
**1. write a SQL query to find the salesperson and customer who reside in the same city. Return Salesman, cust\_name and city.**

select salesman.Name as 'Salesman', customer.Cust\_Name, customer.City from salesman INNER JOIN customer ON salesman.Salesman\_ID = customer.Salesman\_ID;



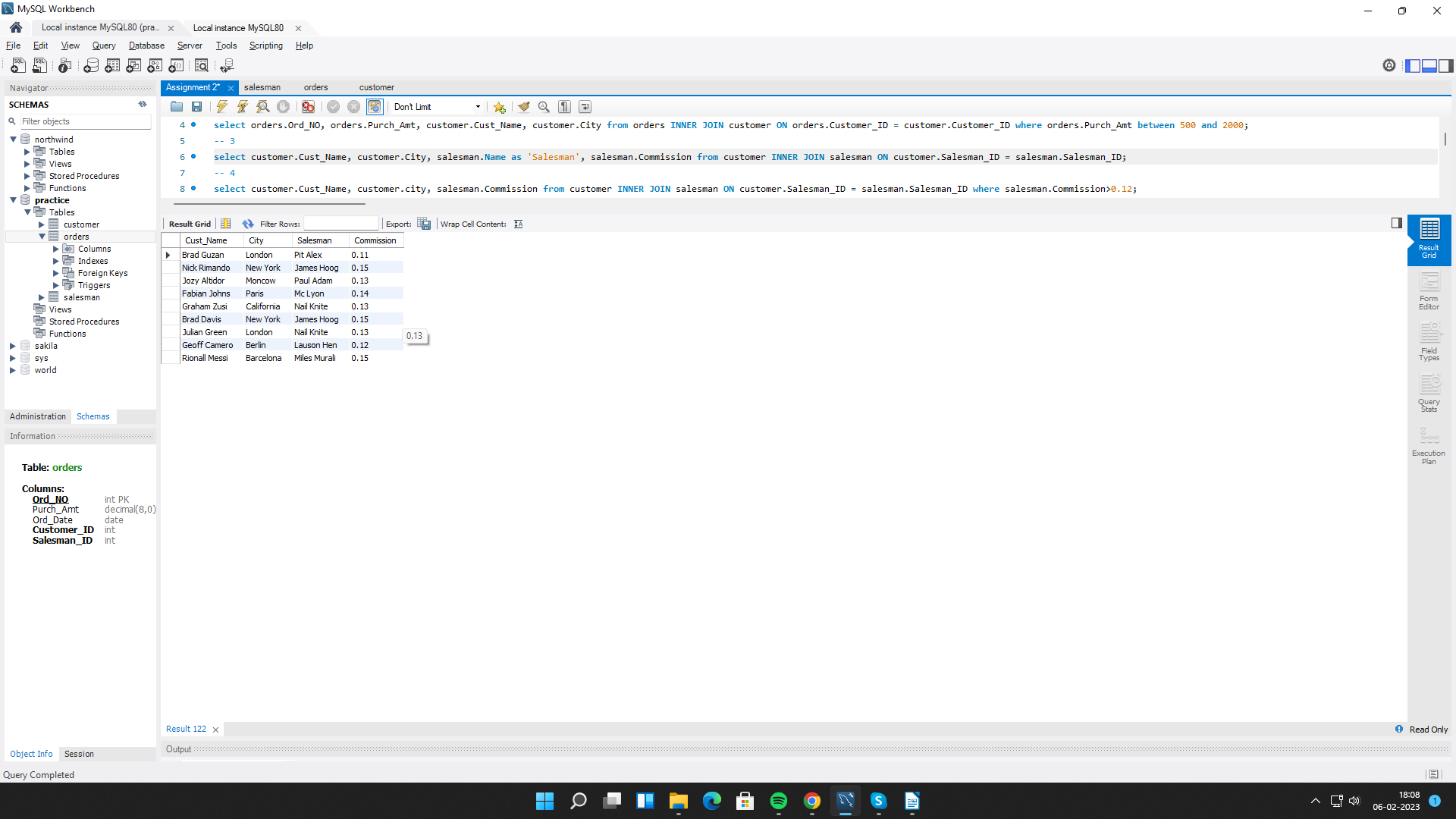
**2. write a SQL query to find those orders where the order amount exists between 500** **and 2000. Return ord\_no, purch\_amt, cust\_name, city.**

select orders.Ord\_NO, orders.Purch\_Amt, customer.Cust\_Name, customer.City from orders INNER JOIN customer ON orders.Customer\_ID = customer.Customer\_ID where orders.Purch\_Amt between 500 and 2000;



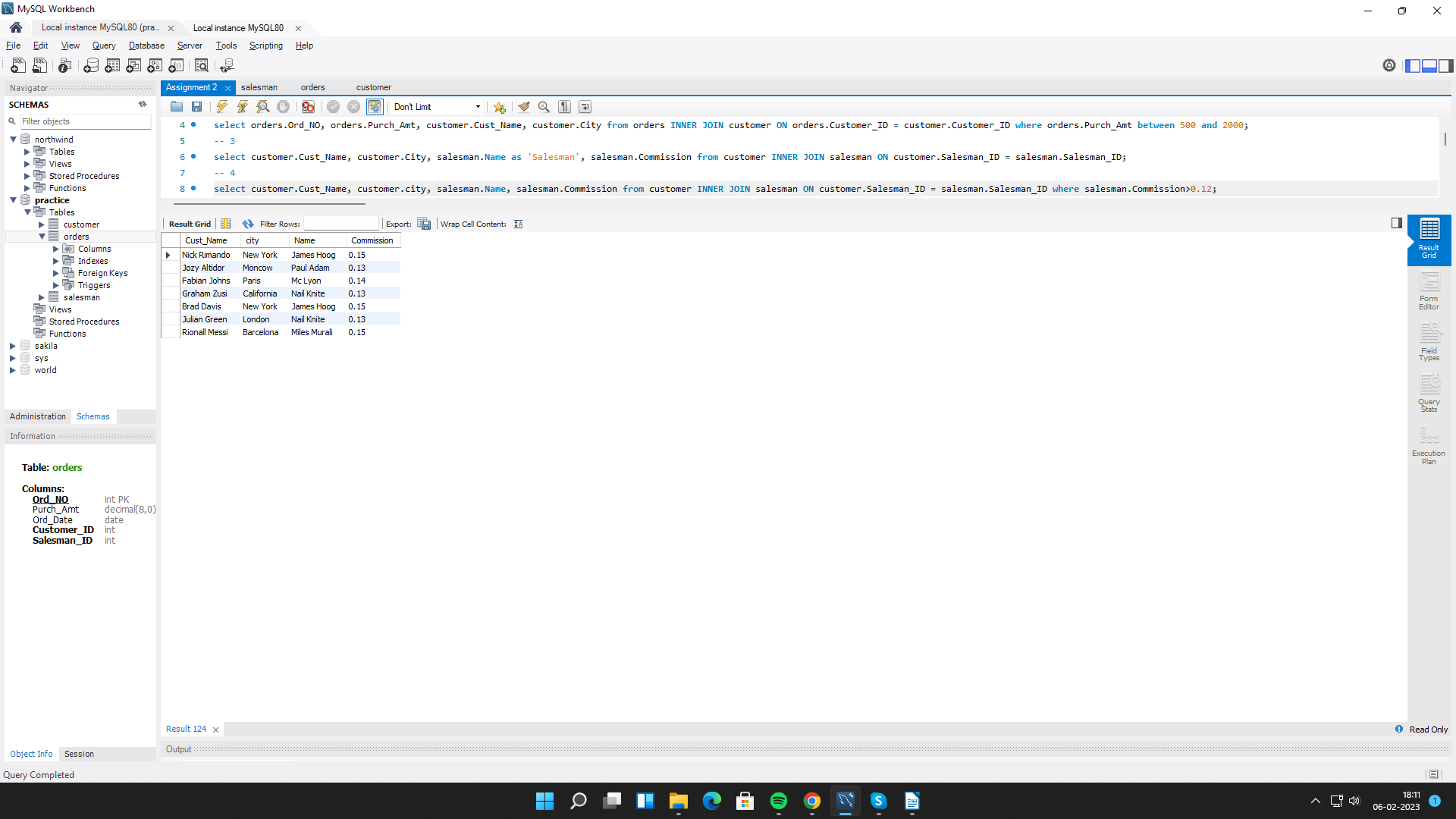
**3. write a SQL query to find the salesperson(s) and the customer(s) he represents. Return Customer Name, city, Salesman, commission.**

select customer.Cust\_Name, customer.City, salesman.Name as 'Salesman', salesman.Commission from customer INNER JOIN salesman ON customer.Salesman\_ID = salesman.Salesman\_ID;



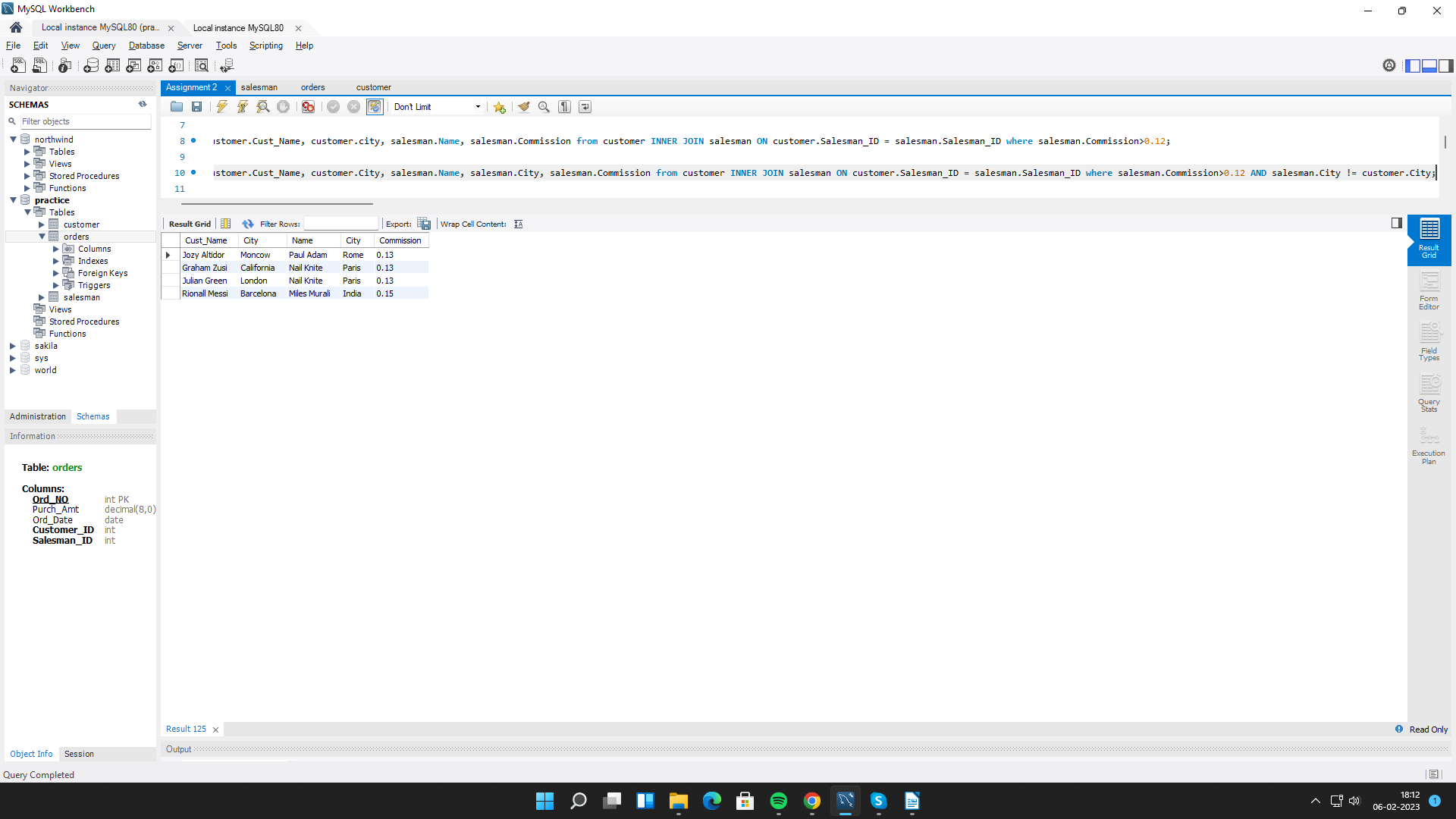
**4. write a SQL query to find salespeople who received commissions of more than 12 percent from the company. Return Customer Name, customer city, Salesman, commission.**

select customer.Cust\_Name, customer.city, salesman.Name, salesman.Commission from customer INNER JOIN salesman ON customer.Salesman\_ID = salesman.Salesman\_ID where salesman.Commission>0.12;



**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.**

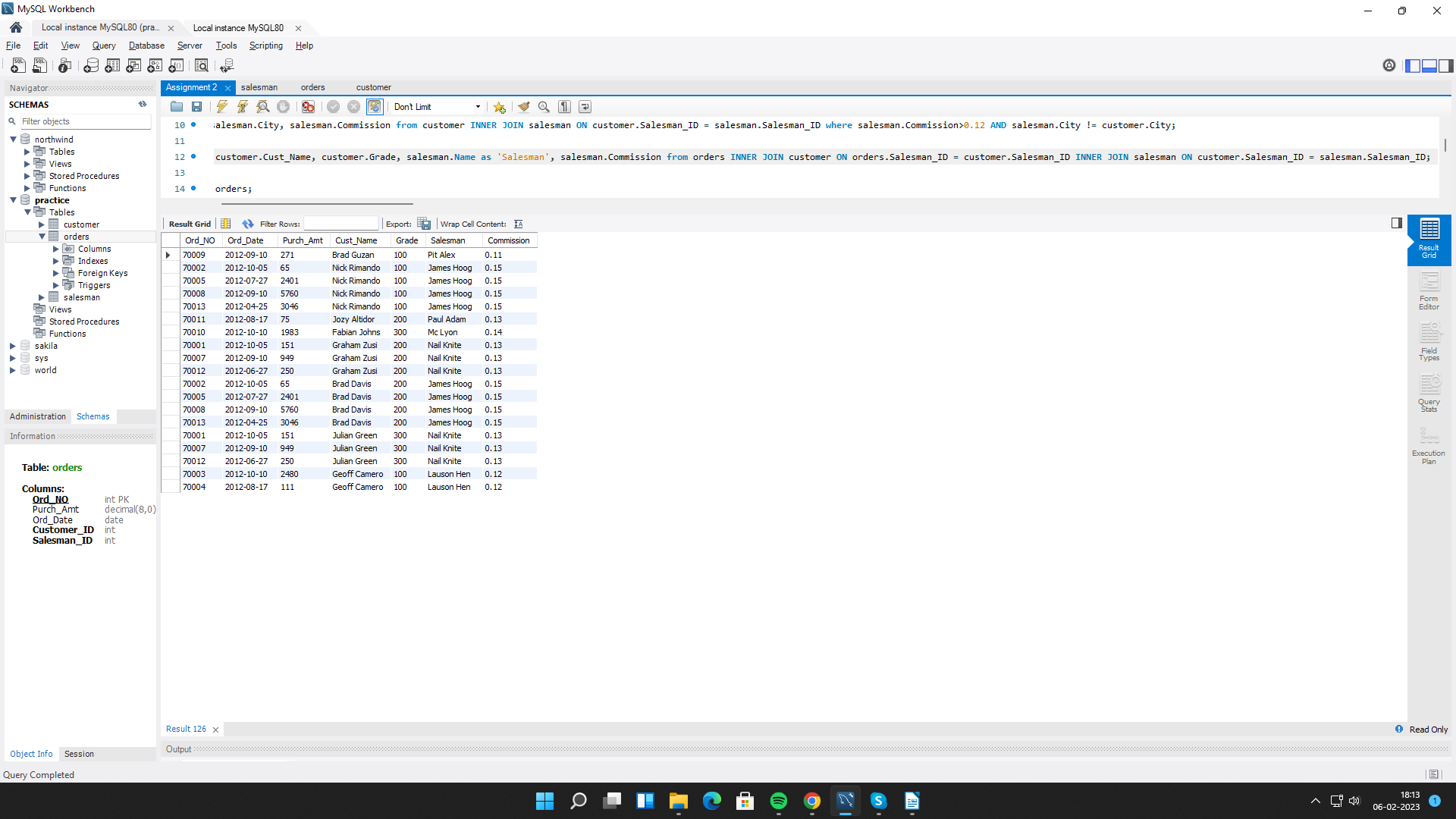
select customer.Cust\_Name, customer.City, salesman.Name, salesman.City, salesman.Commission from customer INNER JOIN salesman ON customer.Salesman\_ID = salesman.Salesman\_ID where salesman.Commission>0.12 AND salesman.City != customer.City;



**6. write a SQL query to find the details of an order. Return ord\_no, ord\_date,**

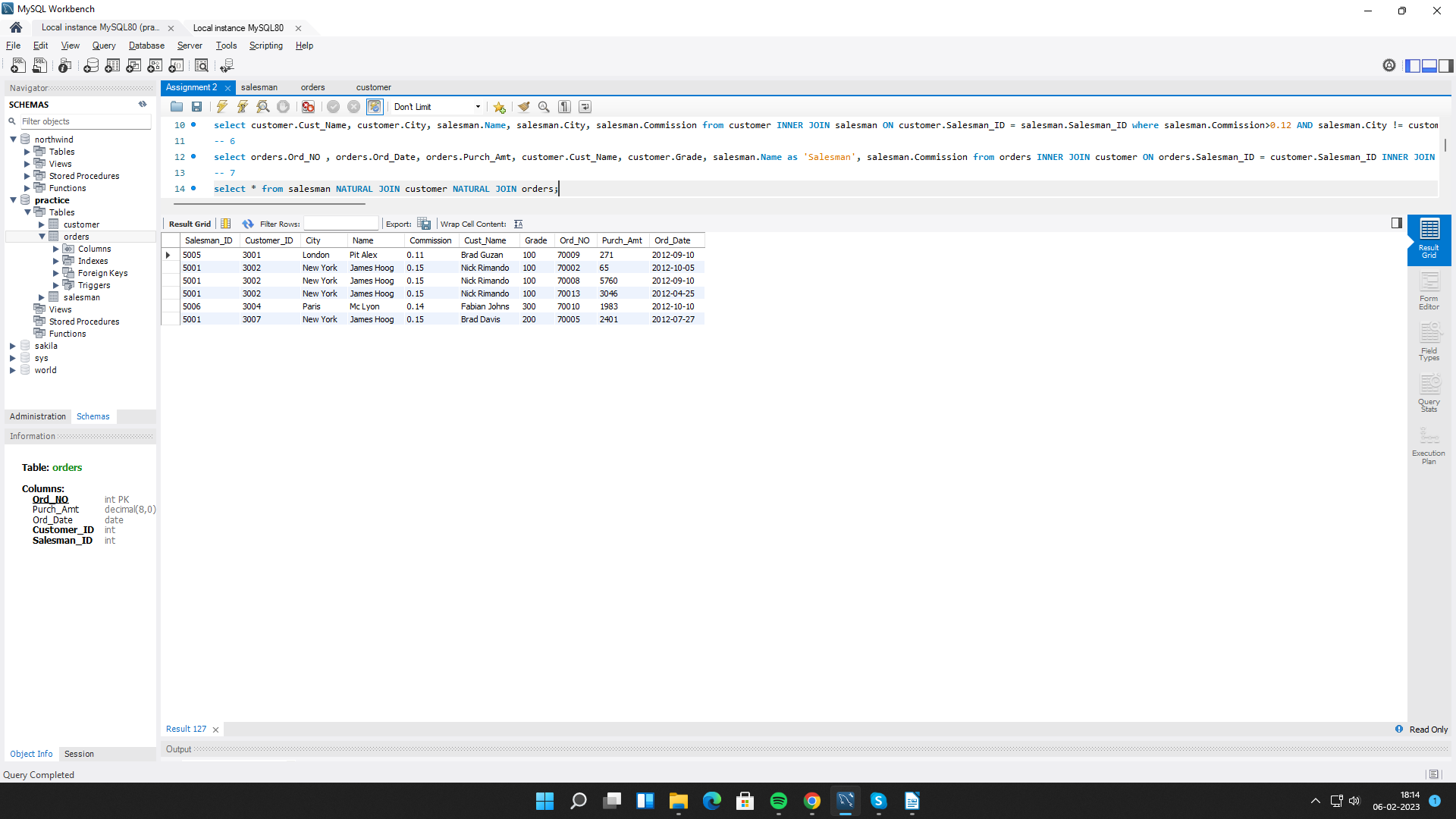
**purch\_amt, Customer Name, grade, Salesman, commission.**

select orders.Ord\_NO , orders.Ord\_Date, orders.Purch\_Amt, customer.Cust\_Name, customer.Grade, salesman.Name as 'Salesman', salesman.Commission from orders INNER JOIN customer ON orders.Salesman\_ID = customer.Salesman\_ID INNER JOIN salesman ON customer.Salesman\_ID = salesman.Salesman\_ID;



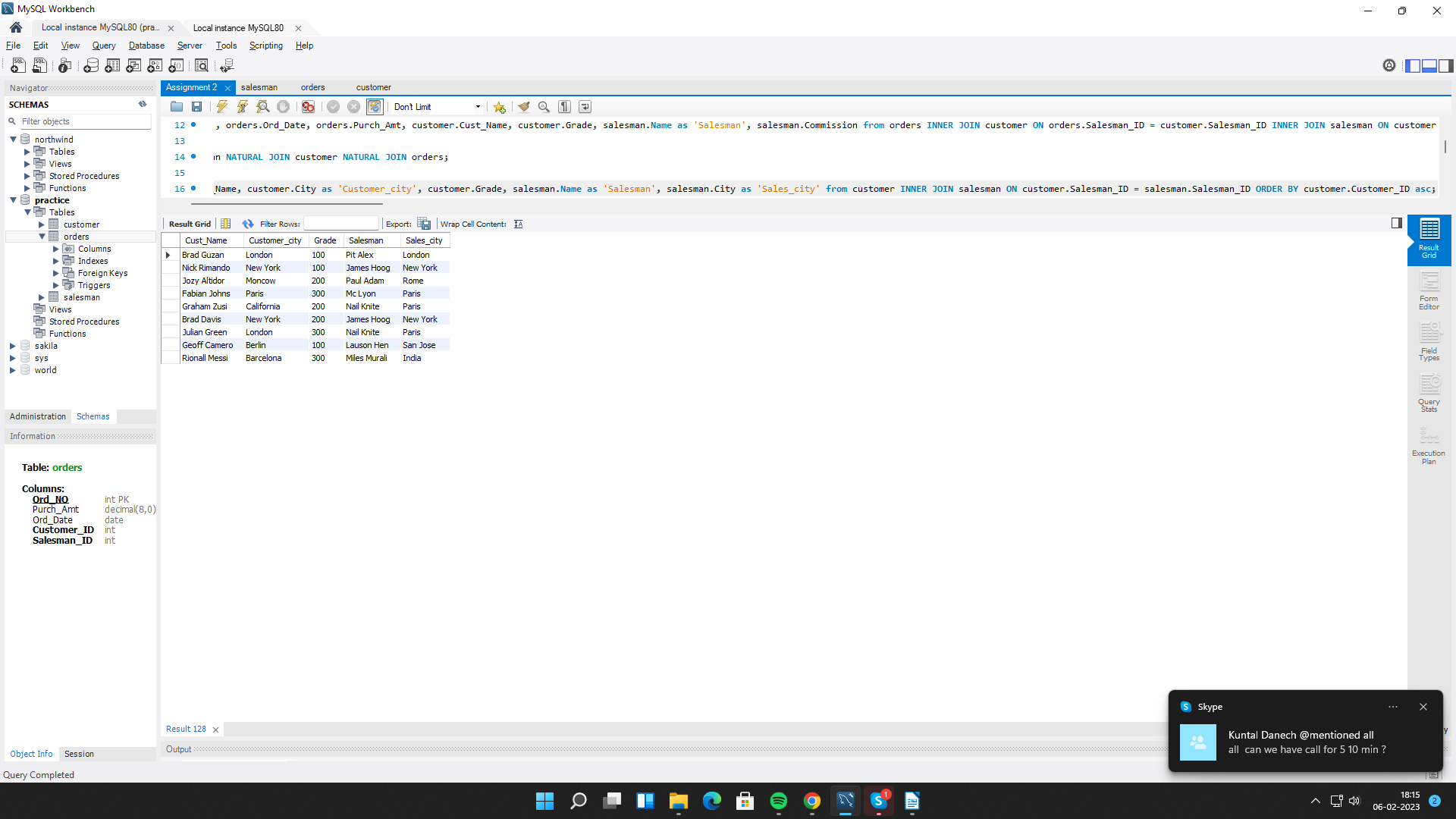
**7. Write a SQL statement to join the tables salesman, customer and orders so that the same column of each table appears once and only the relational rows are returned.**

select \* from salesman NATURAL JOIN customer NATURAL JOIN orders;



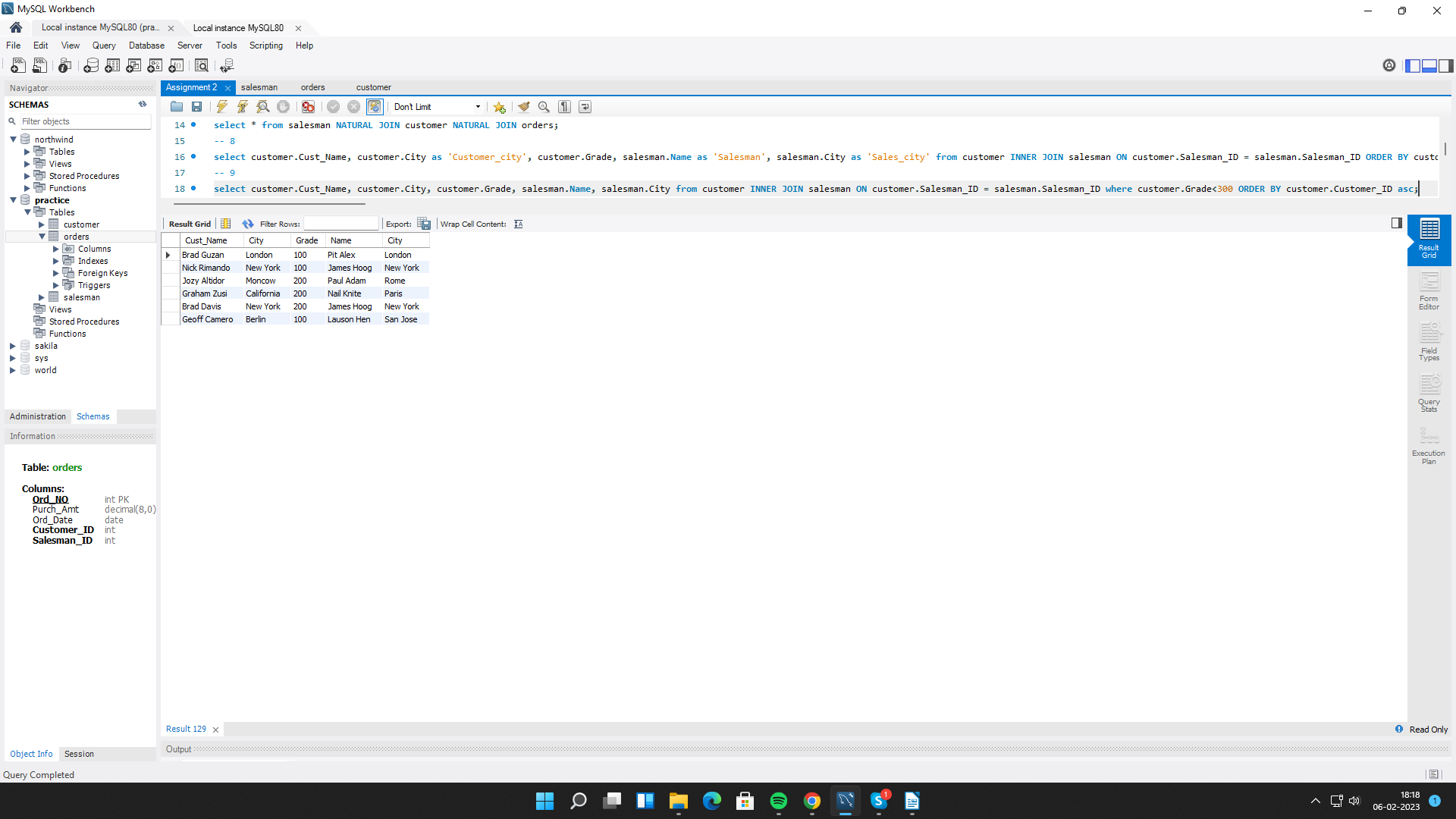
**8. write a SQL query to display the customer name, customer city, grade, salesman, salesman city. The results should be sorted by ascending customer\_id.**

select customer.Cust\_Name, customer.City as 'Customer\_city', customer.Grade, salesman.Name as 'Salesman', salesman.City as 'Sales\_city' from customer INNER JOIN salesman ON customer.Salesman\_ID = salesman.Salesman\_ID ORDER BY customer.Customer\_ID asc;



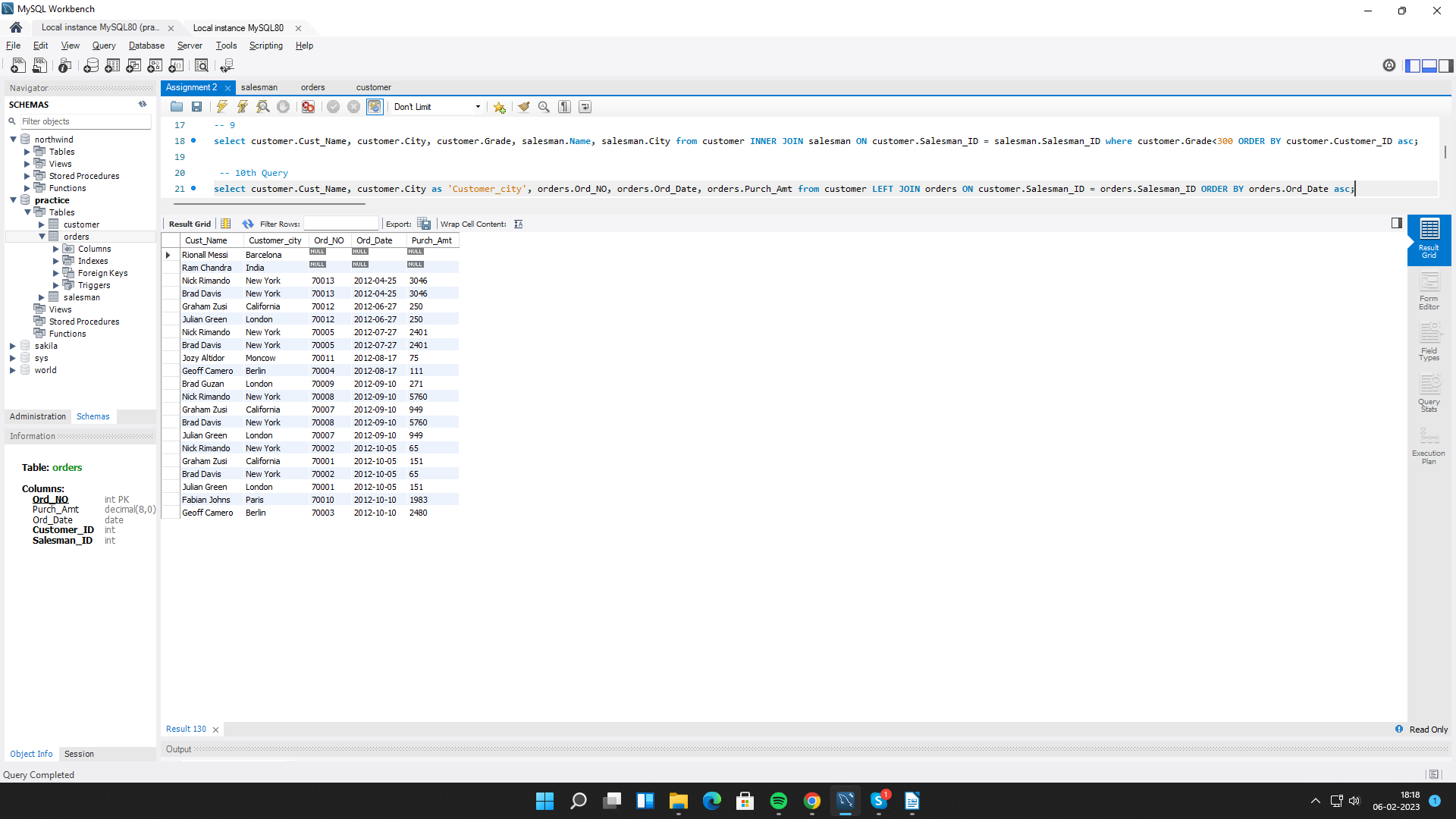
**9. write a SQL query to find those customers with a grade less than 300. Return cust\_name, customer city, grade, Salesman, salesmancity. The result should be ordered by ascending customer\_id.**

select customer.Cust\_Name, customer.City, customer.Grade, salesman.Name, salesman.City from customer INNER JOIN salesman ON customer.Salesman\_ID = salesman.Salesman\_ID where customer.Grade<300 ORDER BY customer.Customer\_ID asc;



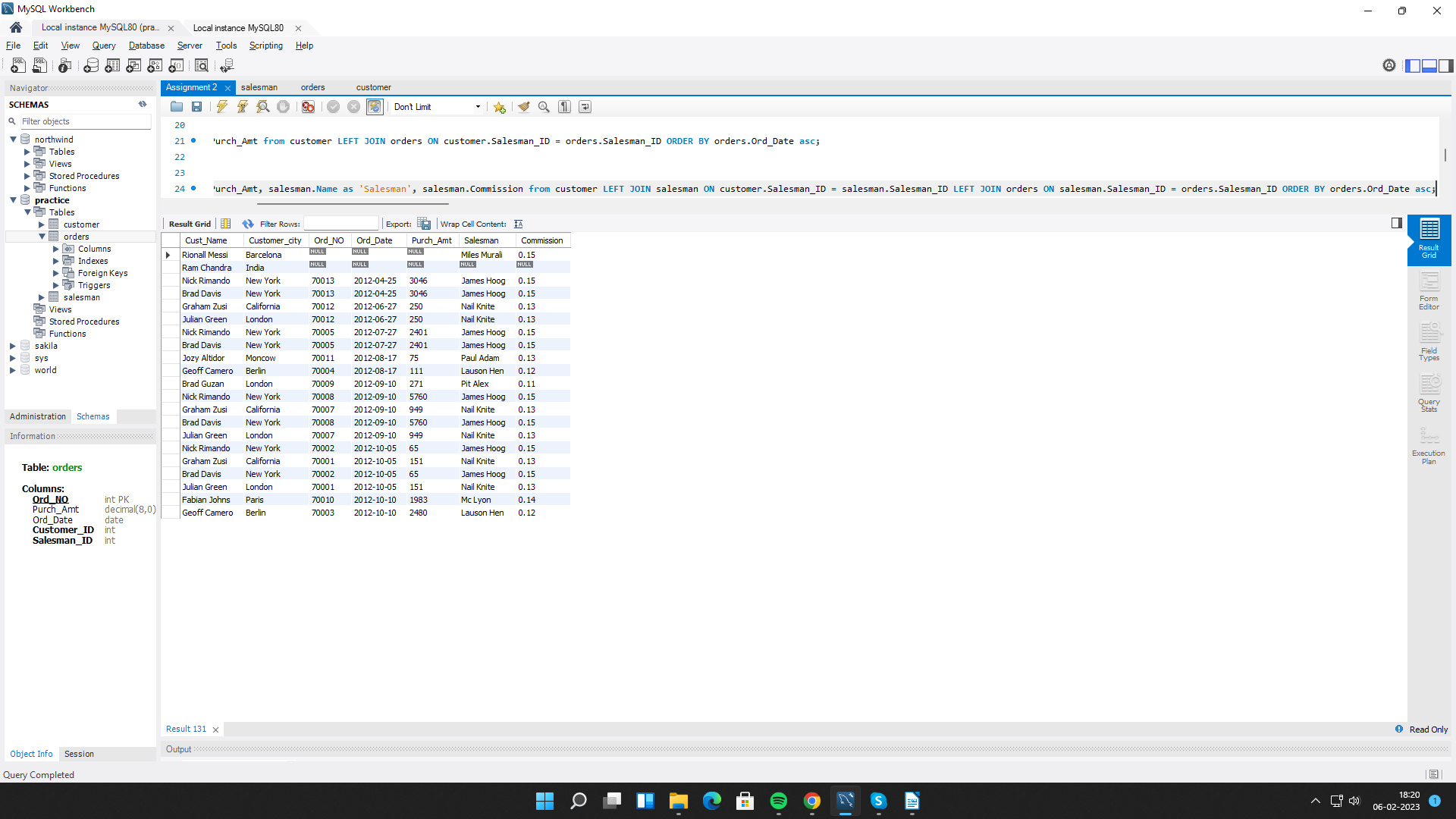
**10. Write a SQL statement to make a report with customer name, city, order number, order date, and order amount in ascending order according to the order date to determine whether any of the existing customers have placed an order or not.**

select customer.Cust\_Name, customer.City as 'Customer\_city', orders.Ord\_NO, orders.Ord\_Date, orders.Purch\_Amt from customer LEFT JOIN orders ON customer.Salesman\_ID = orders.Salesman\_ID ORDER BY orders.Ord\_Date asc;



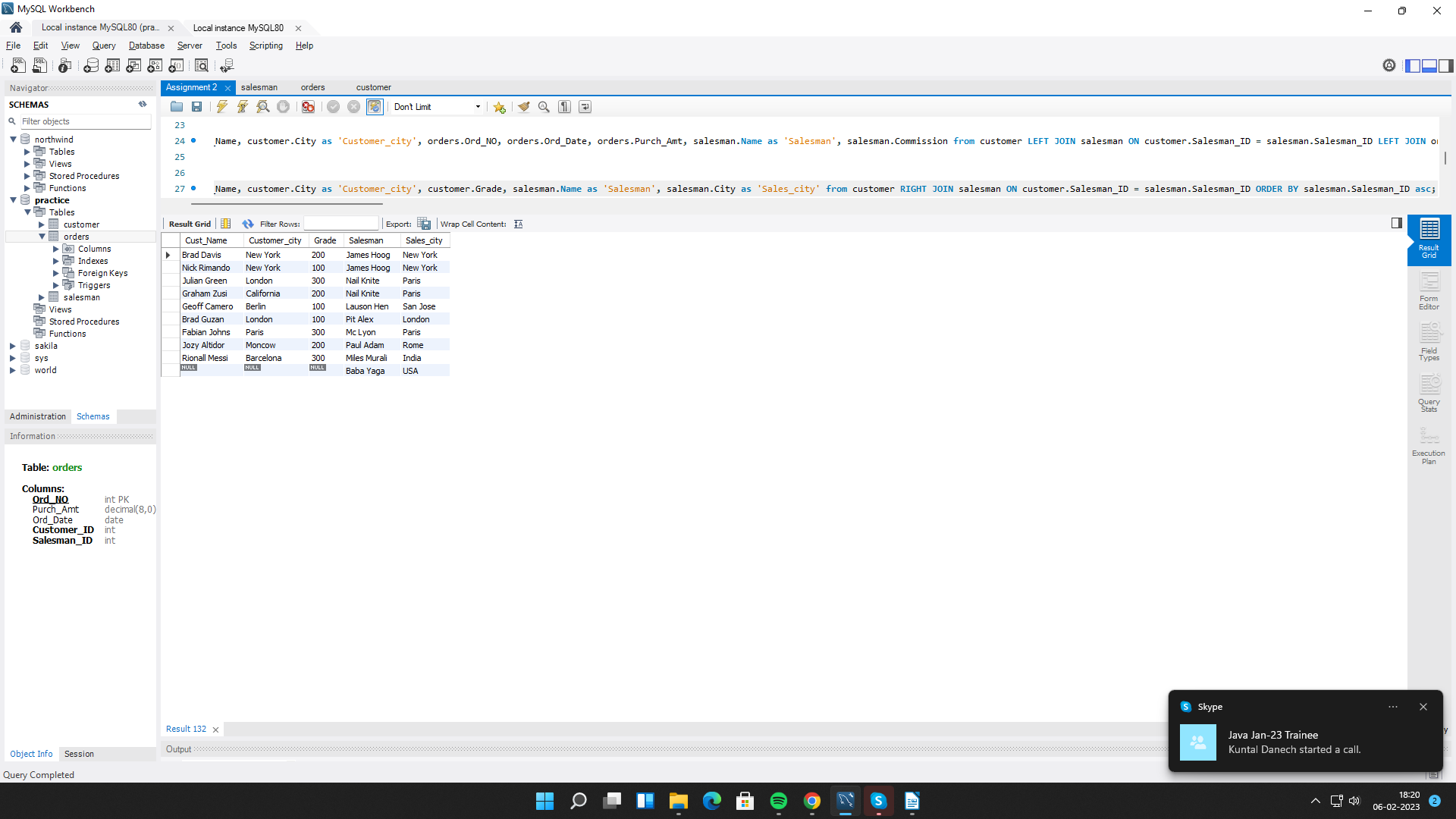
**11. Write a SQL statement to generate a report with customer name, city, order number, order date, order amount, salesperson name, and commission to determine if any of the existing customers have not placed orders or if they have placed orders through their salesman or by themselves.**

select customer.Cust\_Name, customer.City as 'Customer\_city', orders.Ord\_NO, orders.Ord\_Date, orders.Purch\_Amt, salesman.Name as 'Salesman', salesman.Commission from customer LEFT JOIN salesman ON customer.Salesman\_ID = salesman.Salesman\_ID LEFT JOIN orders ON salesman.Salesman\_ID = orders.Salesman\_ID ORDER BY orders.Ord\_Date asc;



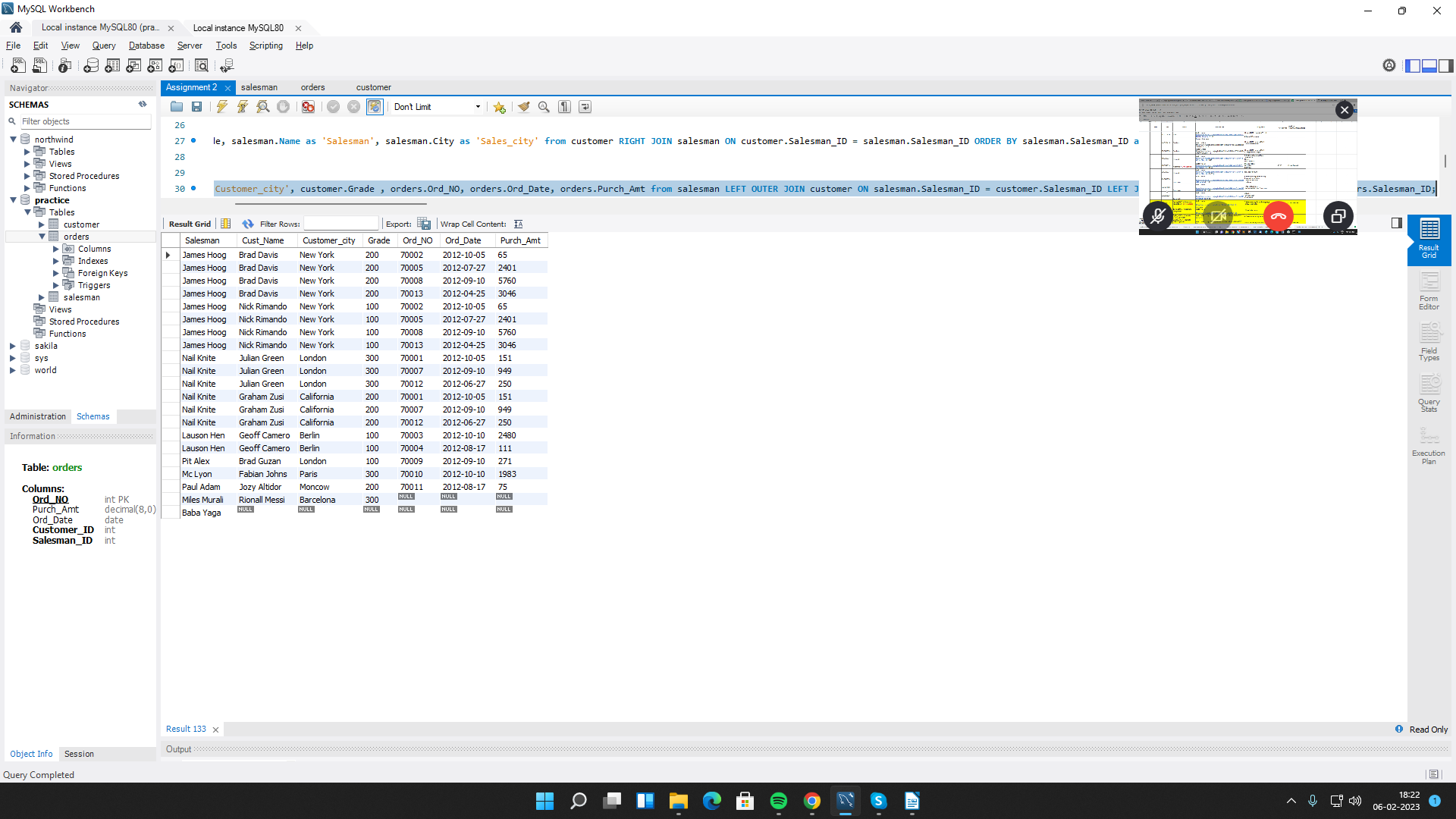
**12. Write a SQL statement to generate a list in ascending order of salespersons who work either for one or more customers or have not yet joined any of the customers.**

select customer.Cust\_Name, customer.City as 'Customer\_city', customer.Grade, salesman.Name as 'Salesman', salesman.City as 'Sales\_city' from customer RIGHT JOIN salesman ON customer.Salesman\_ID = salesman.Salesman\_ID ORDER BY salesman.Salesman\_ID asc;



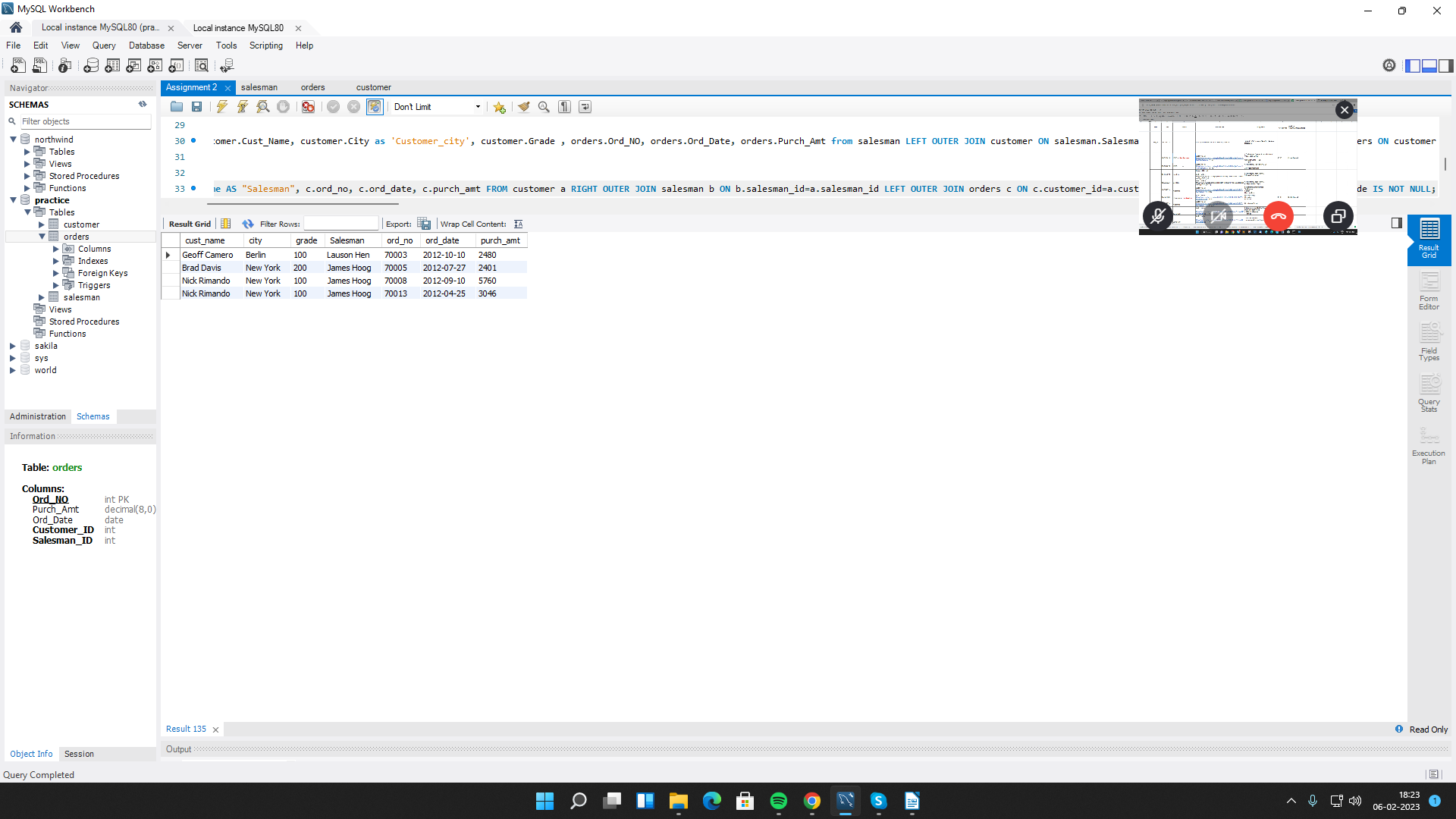
**13. write a SQL query to list all salespersons along with customer name, city, grade, order number, date, and amount.**

select salesman.Name as 'Salesman', customer.Cust\_Name, customer.City as 'Customer\_city', customer.Grade , orders.Ord\_NO, orders.Ord\_Date, orders.Purch\_Amt from salesman LEFT OUTER JOIN customer ON salesman.Salesman\_ID = customer.Salesman\_ID LEFT JOIN orders ON customer.Salesman\_ID = orders.Salesman\_ID;



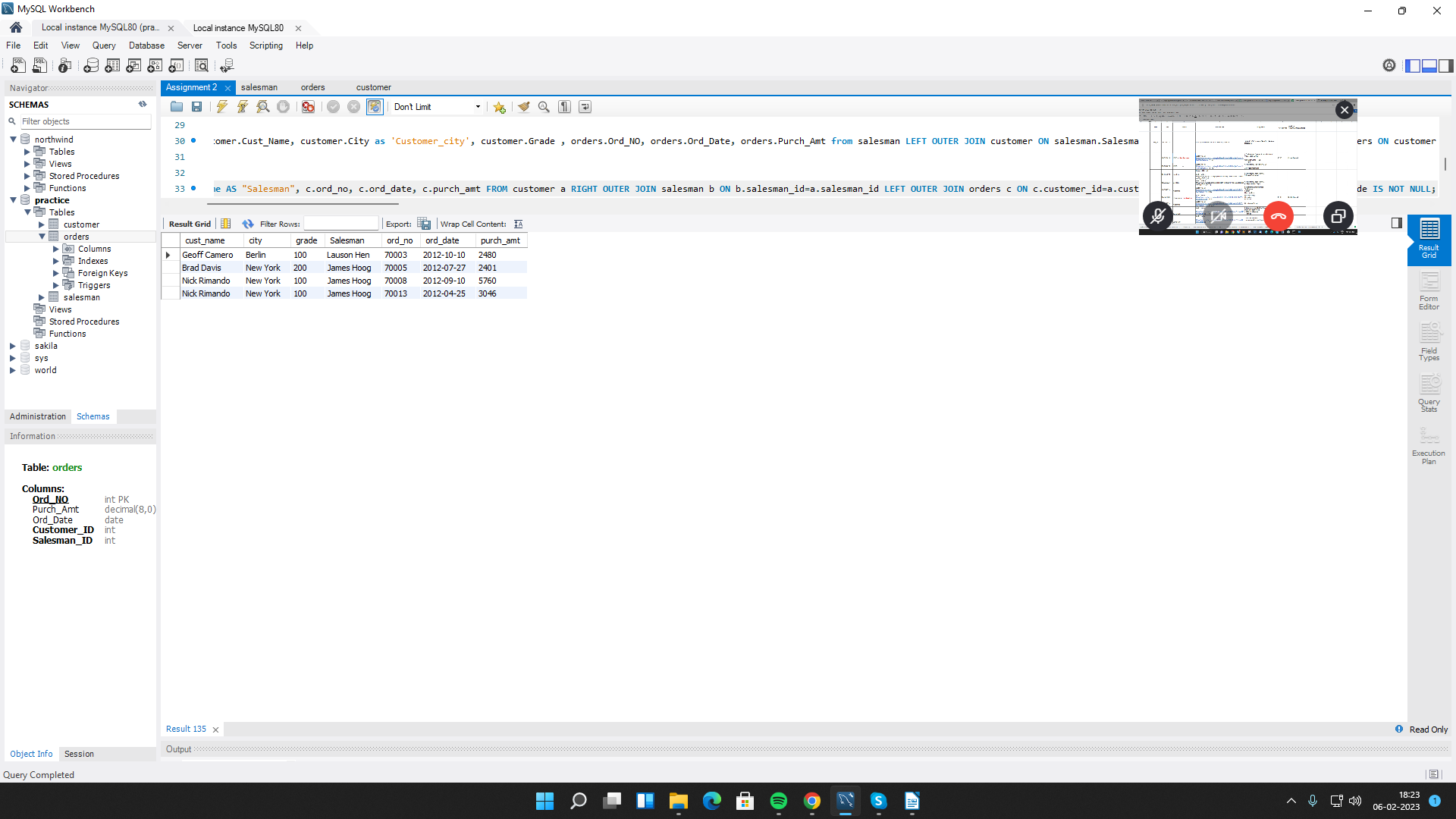
**14. Write a SQL statement to make a list for the salesmen who either work for one or more customers or yet to join any of the customers. The customer may have placed, either one or more orders on or above order amount 2000 and must have a grade, or he may not have placed any order to the associated supplier.**

SELECT a.cust\_name,a.city,a.grade, b.name AS "Salesman", c.ord\_no, c.ord\_date, c.purch\_amt FROM customer a RIGHT OUTER JOIN salesman b ON b.salesman\_id=a.salesman\_id LEFT OUTER JOIN orders c ON c.customer\_id=a.customer\_id WHERE c.purch\_amt>=2000 AND a.grade IS NOT NULL;



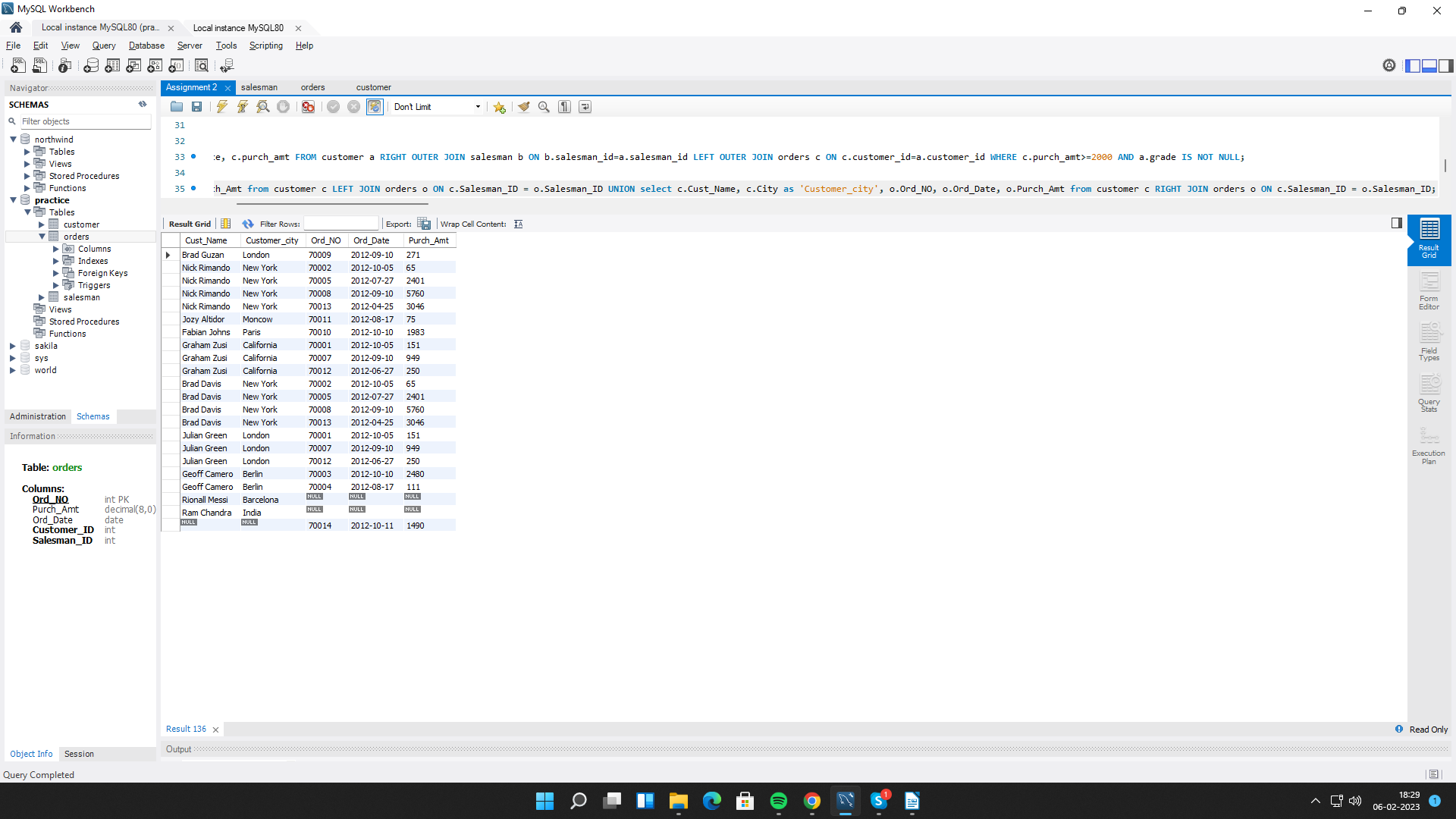
**15. Write a SQL statement to generate a list of all the salesmen who either work for one or more customers or have yet to join any of them. The customer may have placed one or more orders at or above order amount 2000, and must have a grade, or he may not have placed any orders to the associated supplier.**

SELECT a.cust\_name,a.city,a.grade, b.name AS "Salesman", c.ord\_no, c.ord\_date, c.purch\_amt FROM customer a RIGHT OUTER JOIN salesman b ON b.salesman\_id=a.salesman\_id LEFT OUTER JOIN orders c ON c.customer\_id=a.customer\_id WHERE c.purch\_amt>=2000 AND a.grade IS NOT NULL;



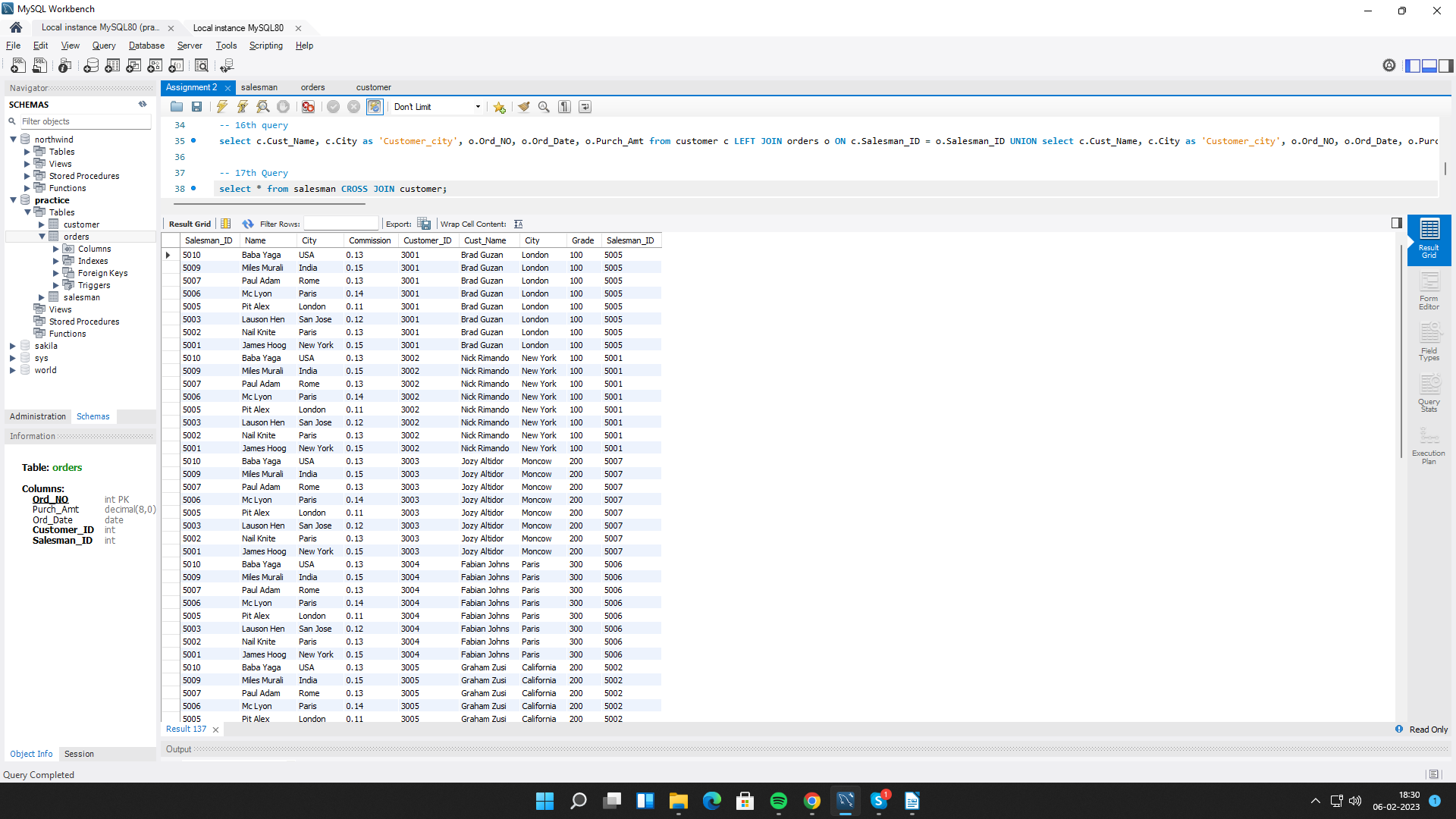
**16. Write a SQL statement to generate a report with the customer name, city, order no. order date, purchase amount for only those customers on the list who must have a grade and placed one or more orders or which order(s) have been placed by the customer who neither is on the list nor has a grade.**

select c.Cust\_Name, c.City as 'Customer\_city', o.Ord\_NO, o.Ord\_Date, o.Purch\_Amt from customer c LEFT JOIN orders o ON c.Salesman\_ID = o.Salesman\_ID UNION select c.Cust\_Name, c.City as 'Customer\_city', o.Ord\_NO, o.Ord\_Date, o.Purch\_Amt from customer c RIGHT JOIN orders o ON c.Salesman\_ID = o.Salesman\_ID;



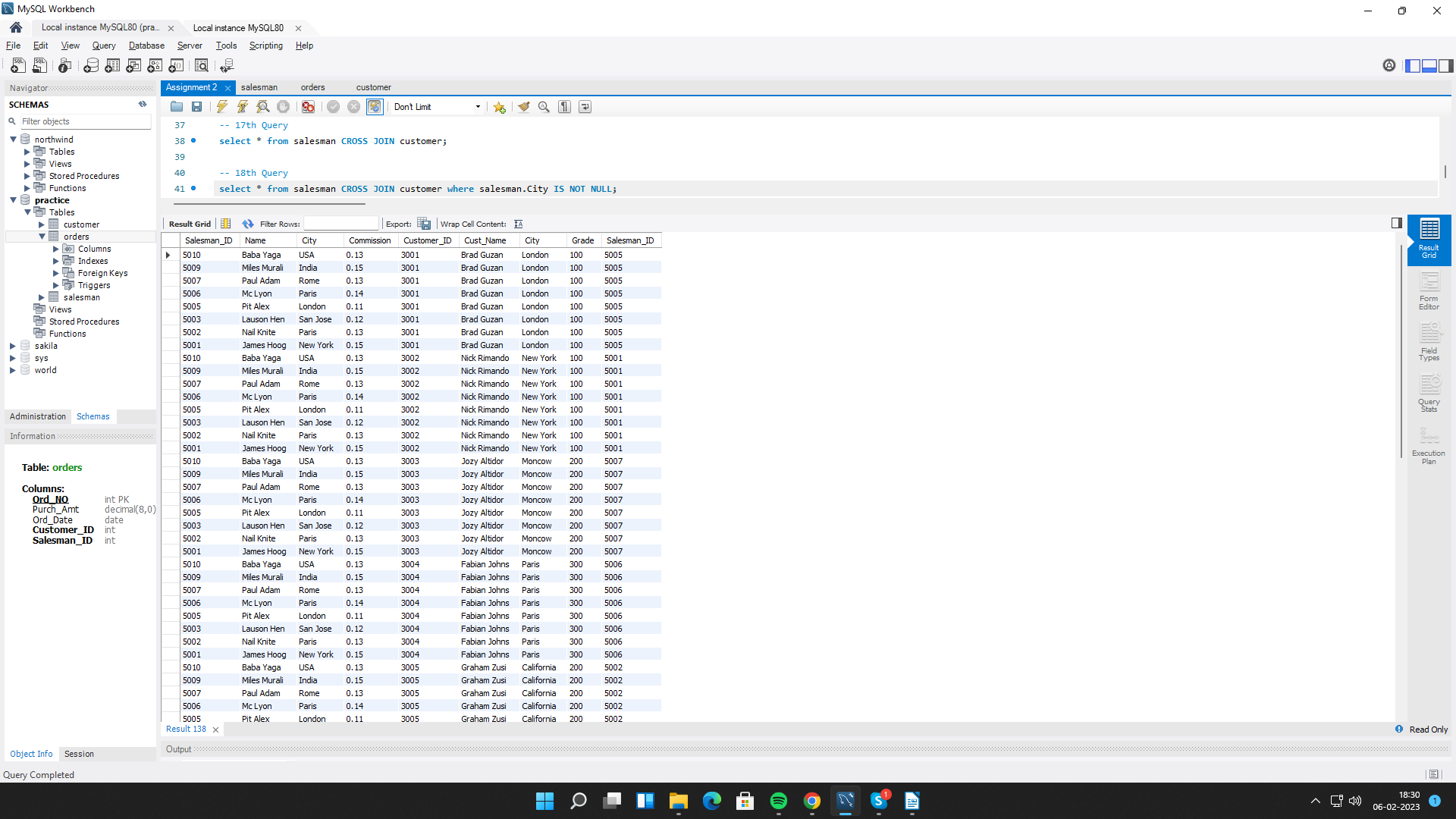
**17. Write a SQL query to combine each row of the salesman table with each row of the customer table .**

select \* from salesman CROSS JOIN customer;



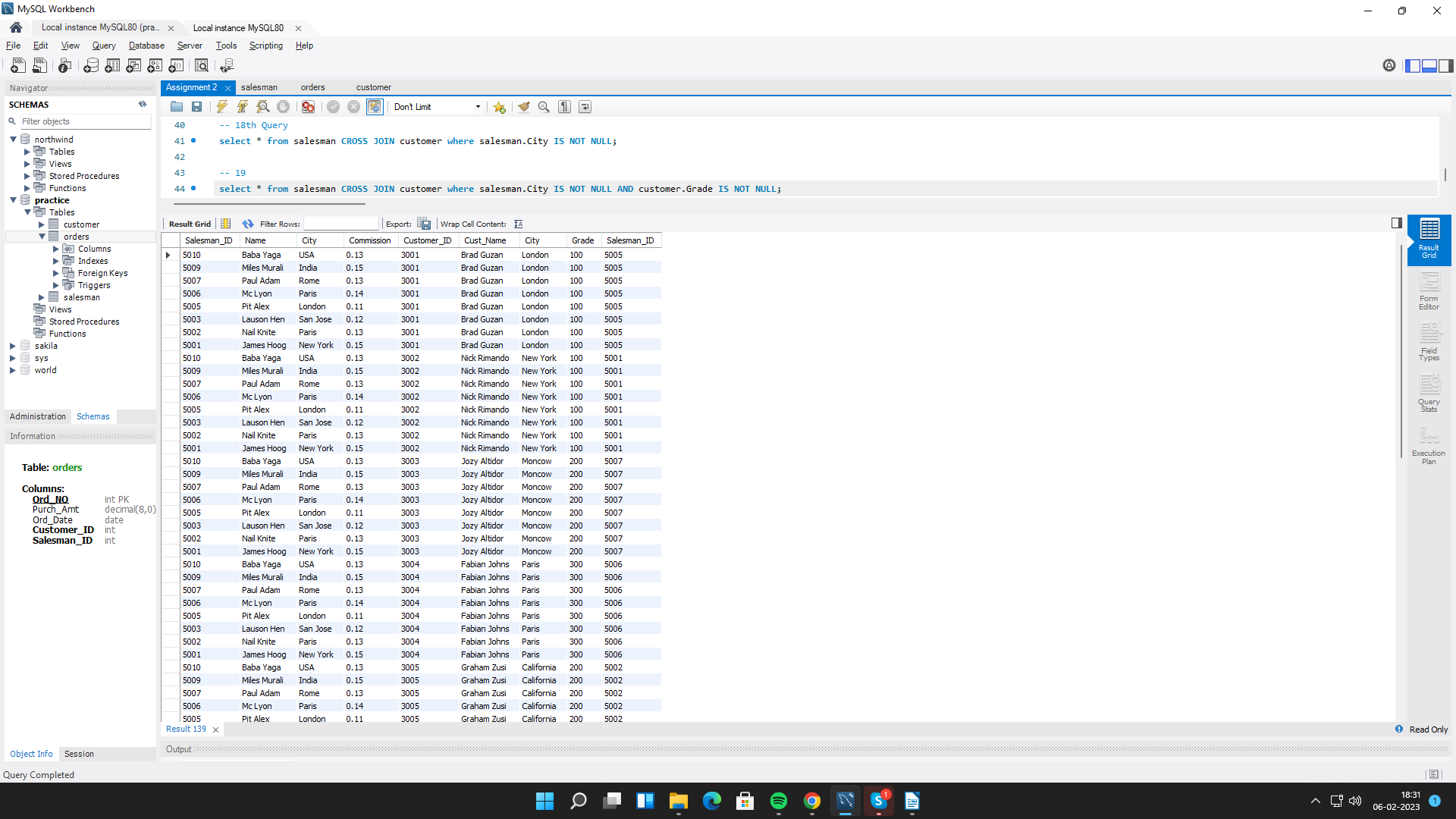
**18. Write a SQL statement to create a Cartesian product between salesperson and customer, i.e. each salesperson will appear for all customers and vice versa for that salesperson who belongs to that city.**

select \* from salesman CROSS JOIN customer where salesman.City IS NOT NULL;



**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 CROSS JOIN customer where salesman.City IS NOT NULL AND customer.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.**

select \* from salesman CROSS JOIN customer where salesman.City IS NOT NULL AND customer.Grade IS NOT NULL AND salesman.City != customer.City;

