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,

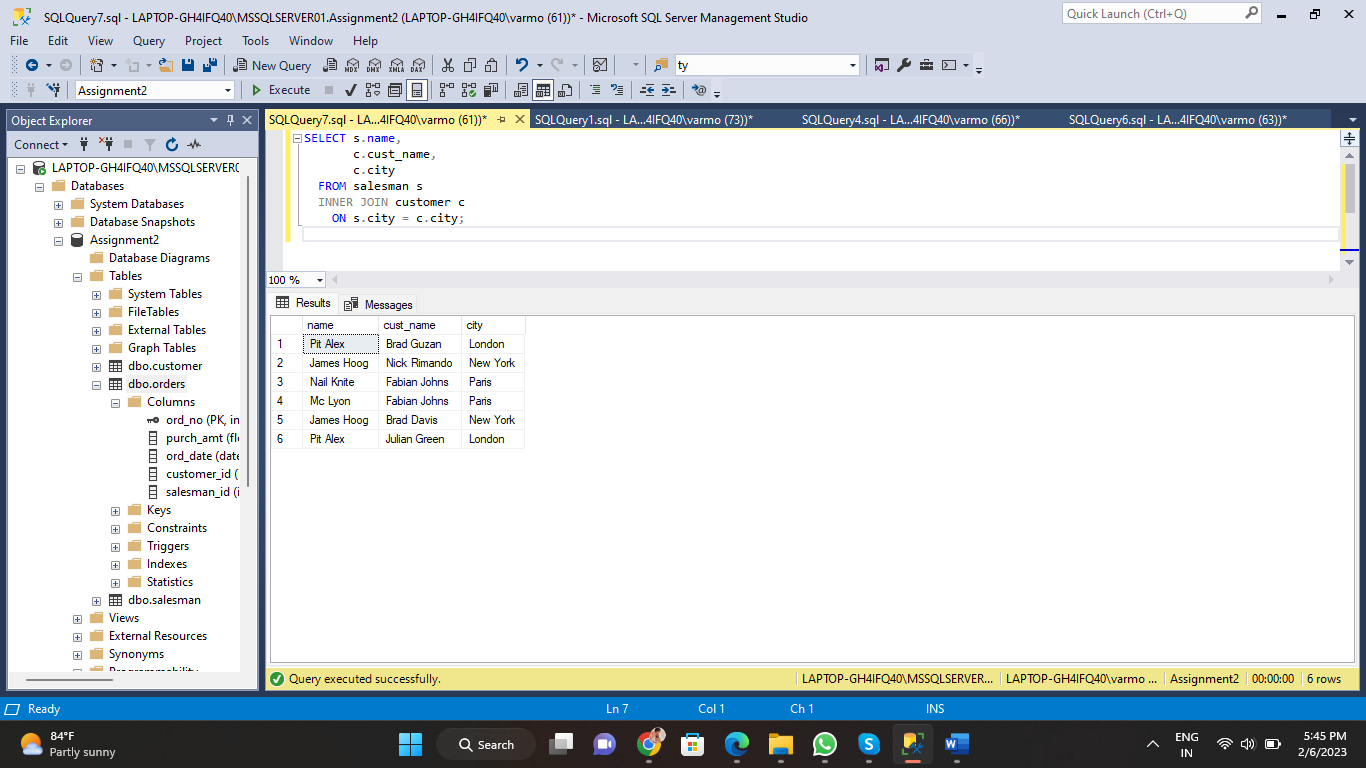
c.cust\_name,

c.city

FROM salesman s

INNER JOIN customer c

ON s.city = c.city;



1. 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,

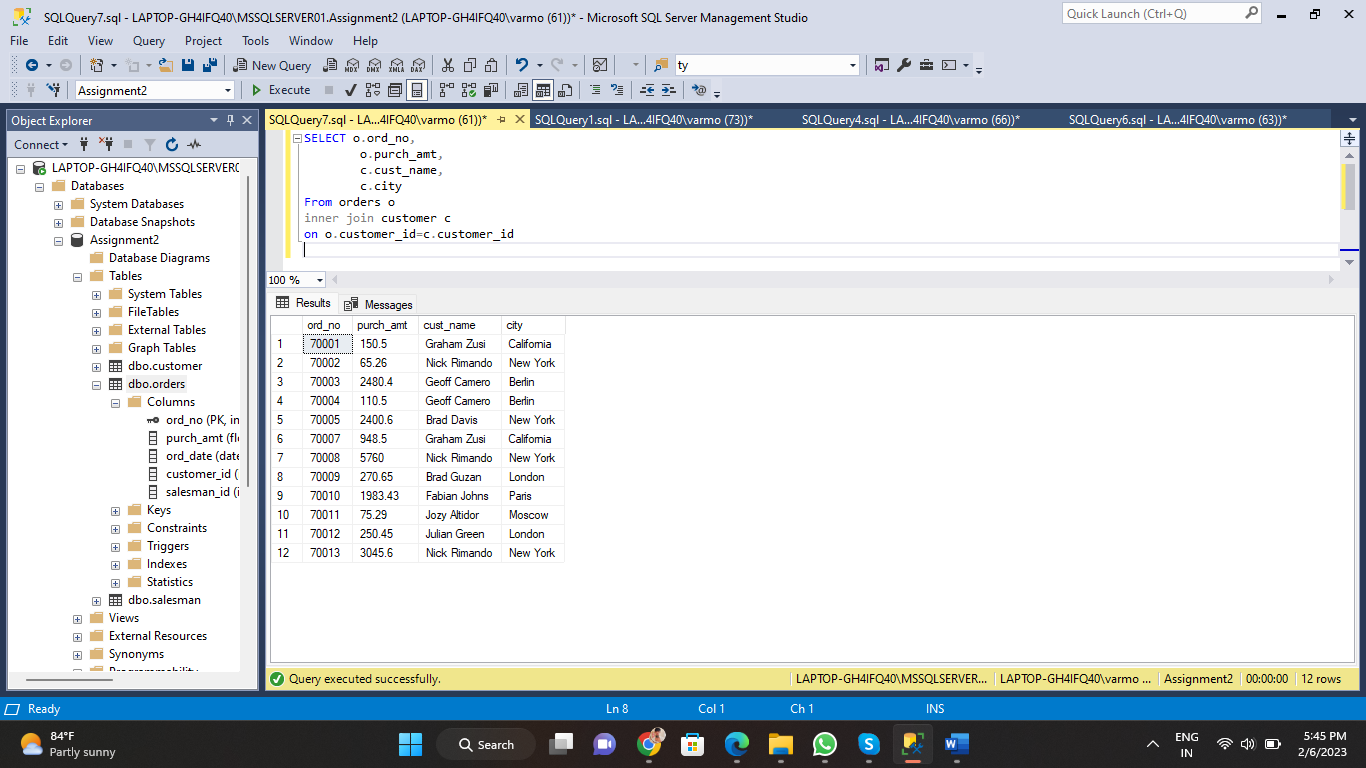
c.cust\_name,

c.city

From orders o

inner join customer c

on o.customer\_id=c.customer\_id;



1. 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 AS "Customer Name",

c.city,

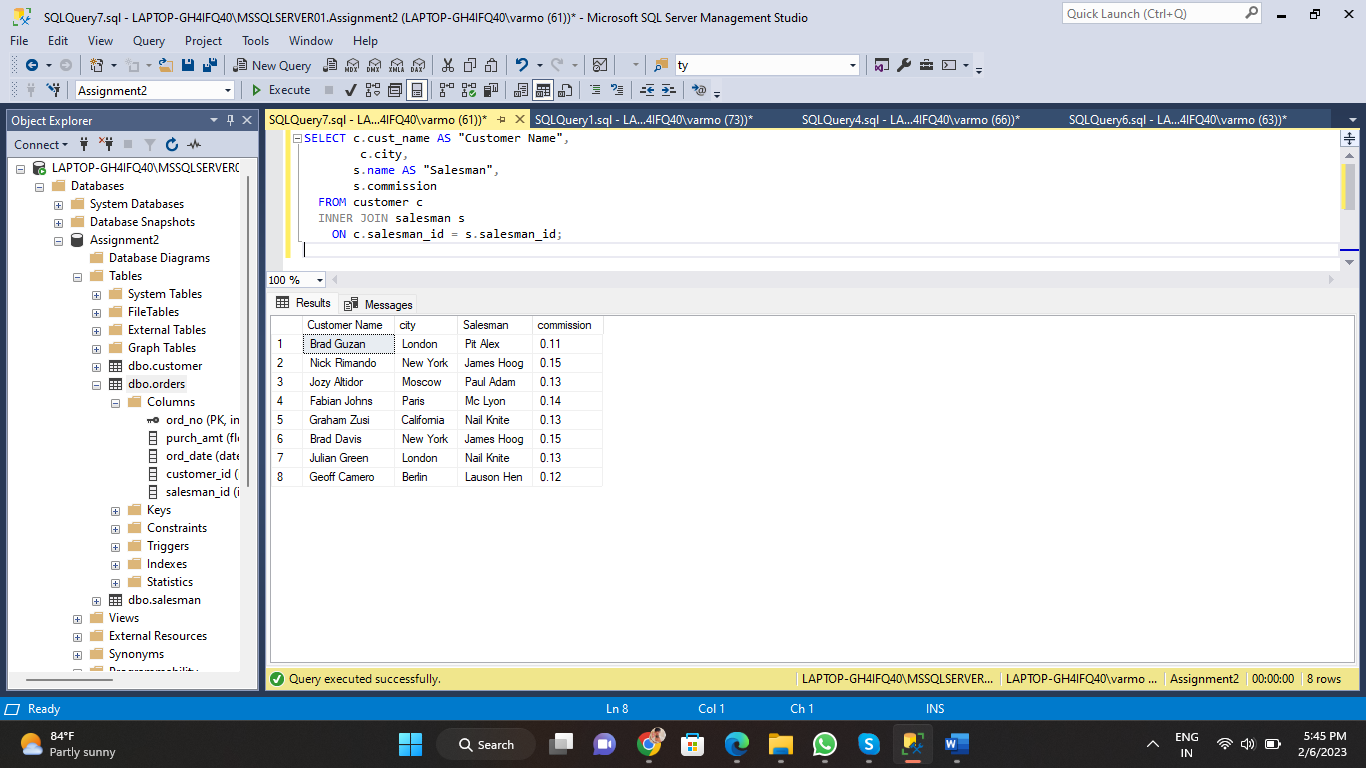
s.name AS "Salesman",

s.commission

FROM customer c

INNER JOIN salesman s

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



1. 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 AS "Customer Name",

c.city,

s.name AS "Salesman",

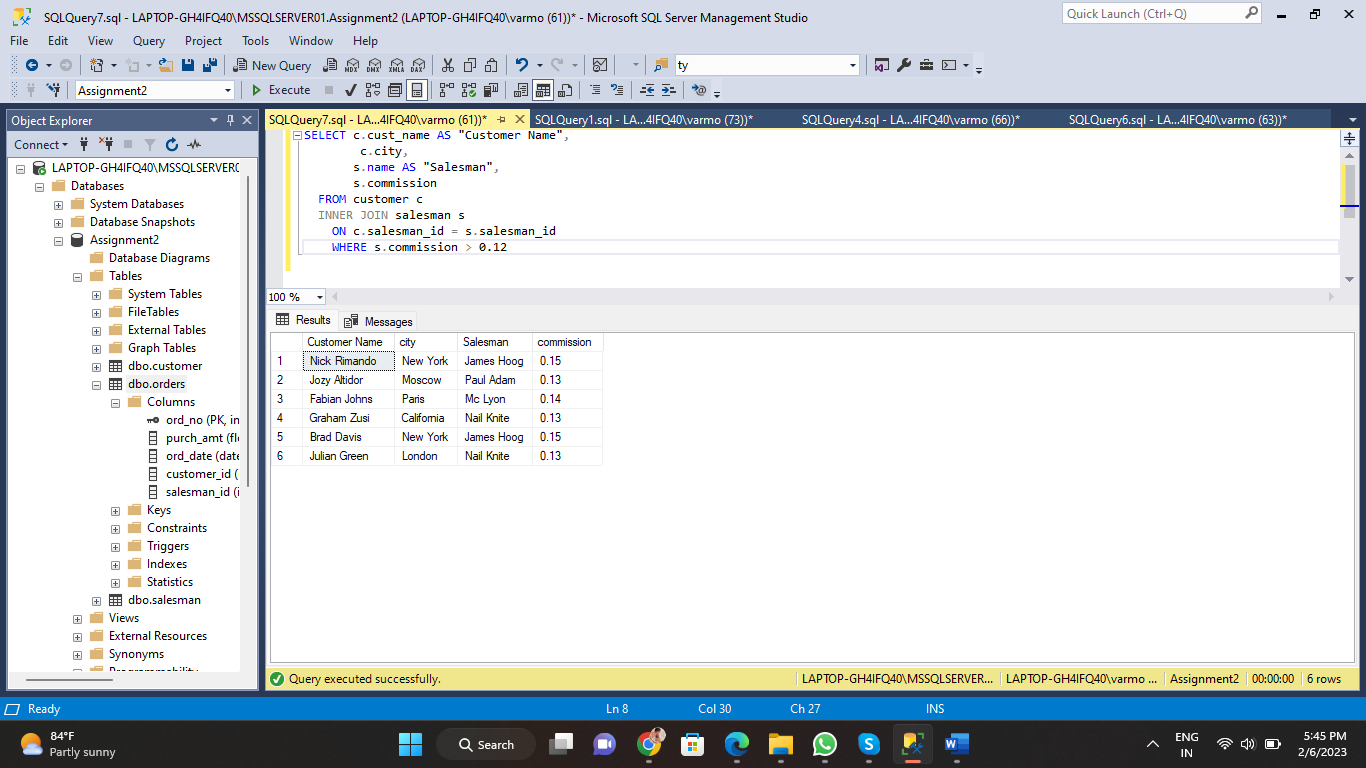
s.commission

FROM customer c

INNER JOIN salesman s

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

WHERE s.commission > 0.12;



1. 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 AS "Customer Name",

c.city AS "Customer City",

s.name AS "Salesman",

s.city AS "Salesman City",

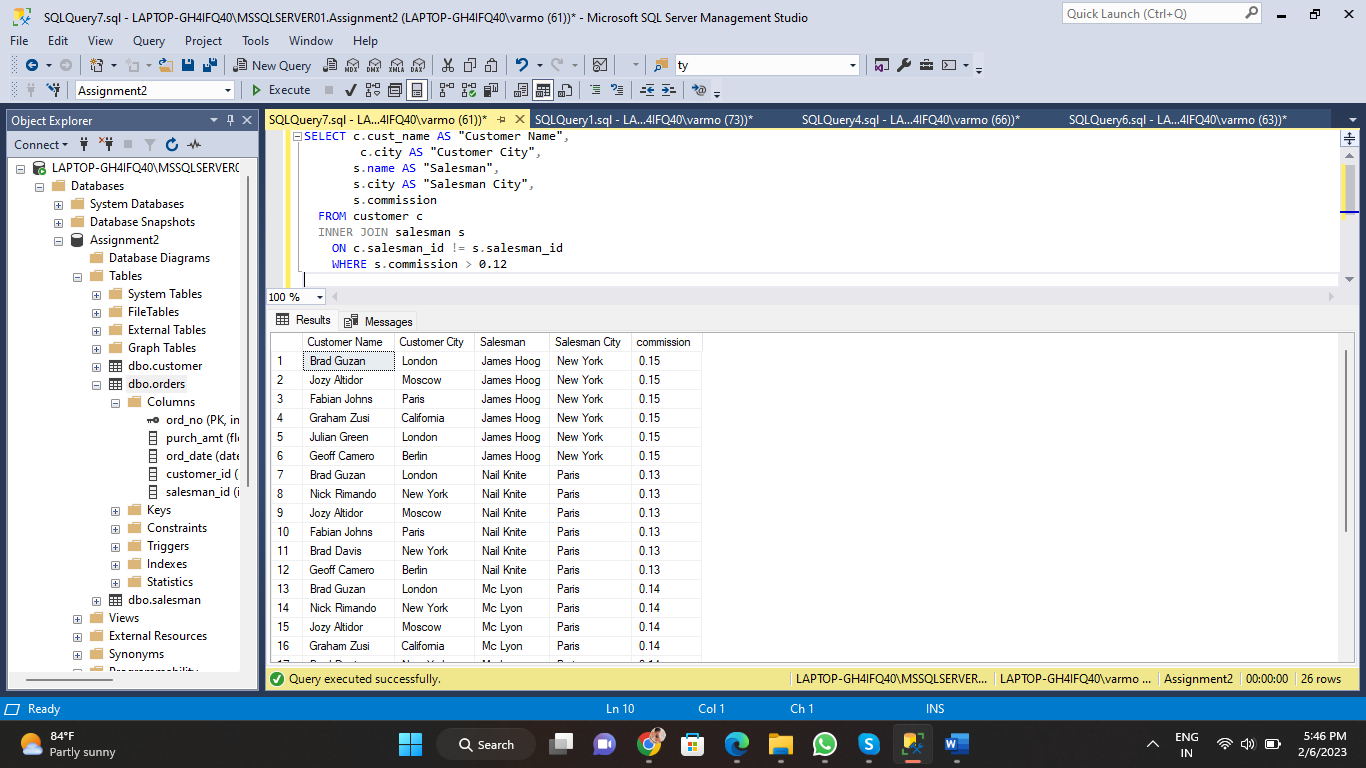
s.commission

FROM customer c

INNER JOIN salesman s

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

WHERE s.commission > 0.12;



1. 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 AS "Order ID",

o.ord\_date AS "Order Date",

o.purch\_amt AS "Purchase Amount",

c.cust\_name AS "Customer Name",

c.grade AS "Grade",

s.name AS "Salesman Name",

s.commission AS "Commission"

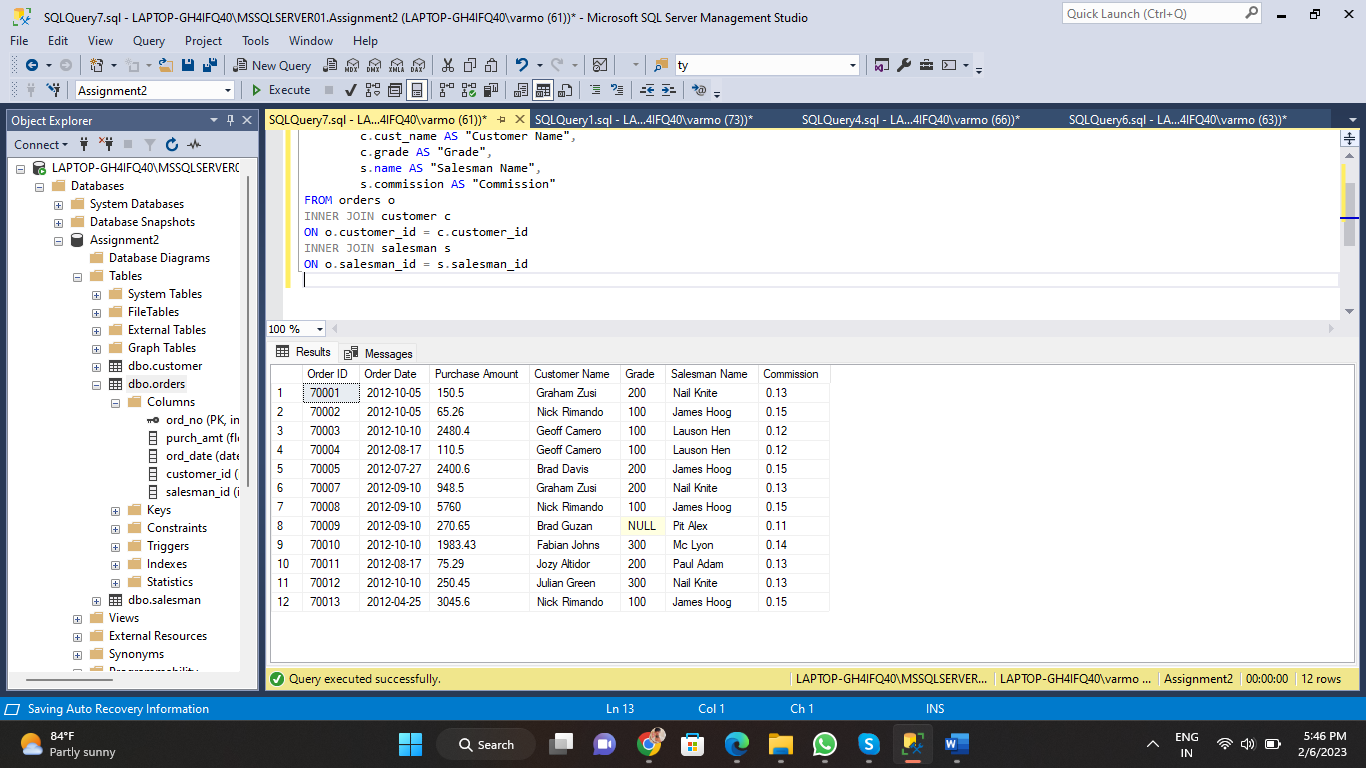
FROM orders o

INNER JOIN customer c

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

INNER JOIN salesman s

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



1. 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.ord\_no AS "Order ID",

o.ord\_date,

o.purch\_amt,

c.customer\_id,

c.cust\_name AS "Customer Name",

c.city AS "Customer City",

c.grade,

s.salesman\_id,

s.name AS "Salesman Name",

s.city AS "Salesman City",

s.commission

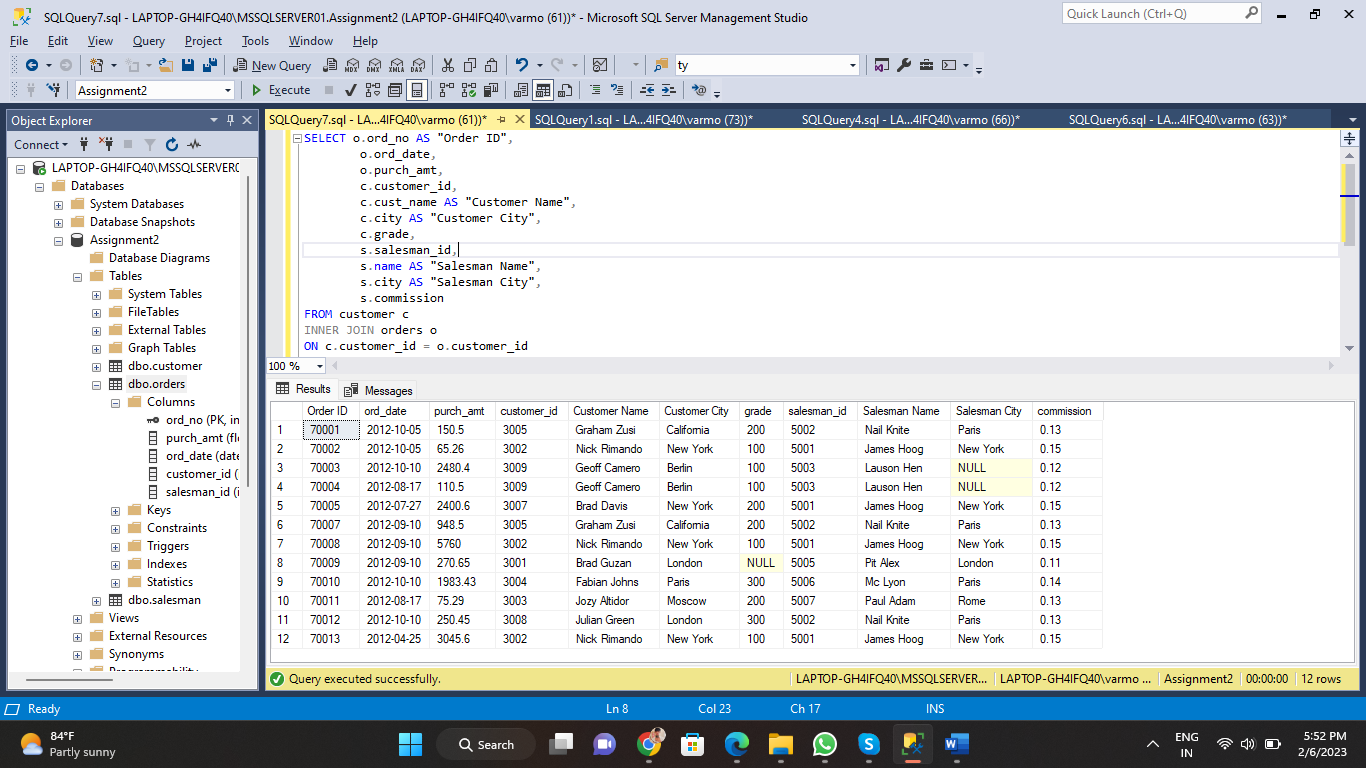
FROM customer c

INNER JOIN orders o

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

INNER JOIN salesman s

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



1. 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 AS "Customer Name",

c.city AS "Customer City",

c.grade AS "Grade",

s.name AS "Salesman Name",

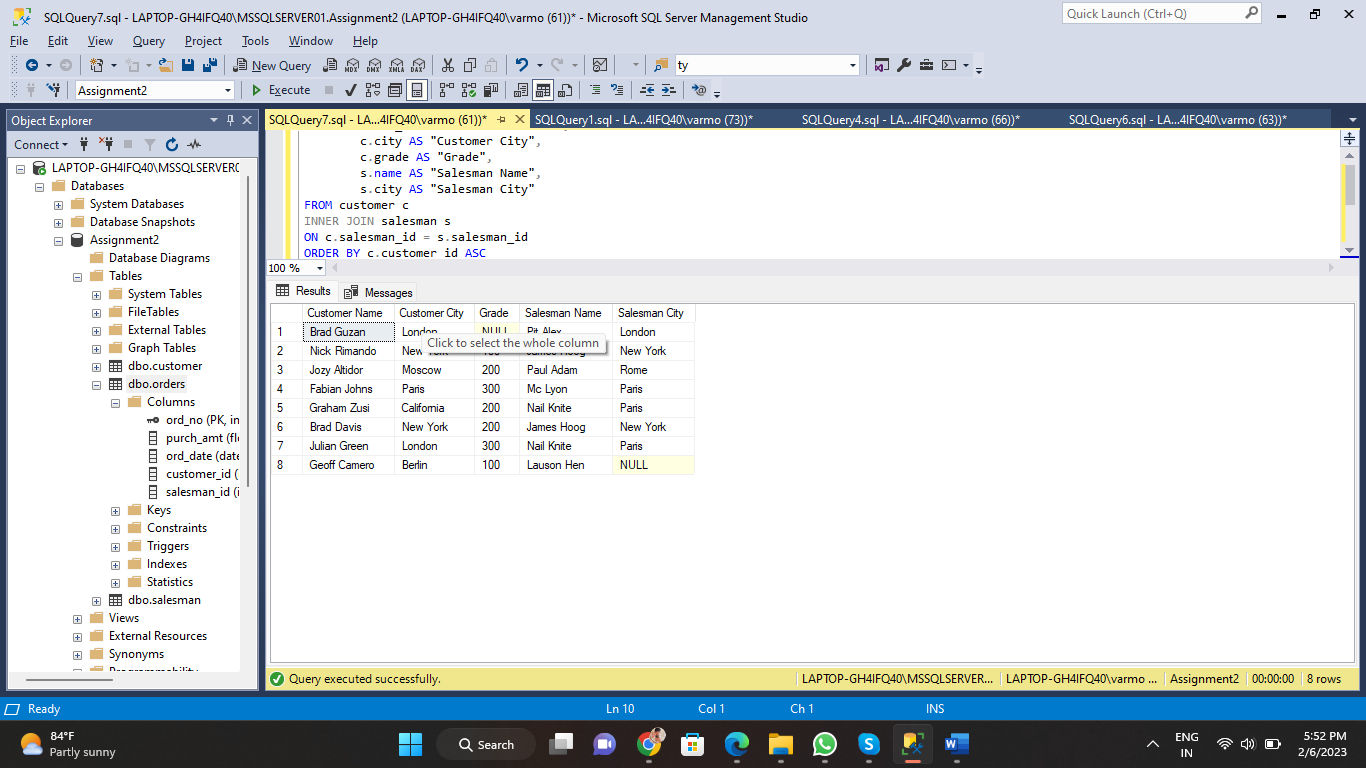
s.city AS "Salesman City"

FROM customer c

INNER JOIN salesman s

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

ORDER BY c.customer\_id ASC;



1. 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 AS "Customer Name",

c.city AS "Customer City",

c.grade AS "Grade",

s.name AS "Salesman Name",

s.city AS "Salesman City"

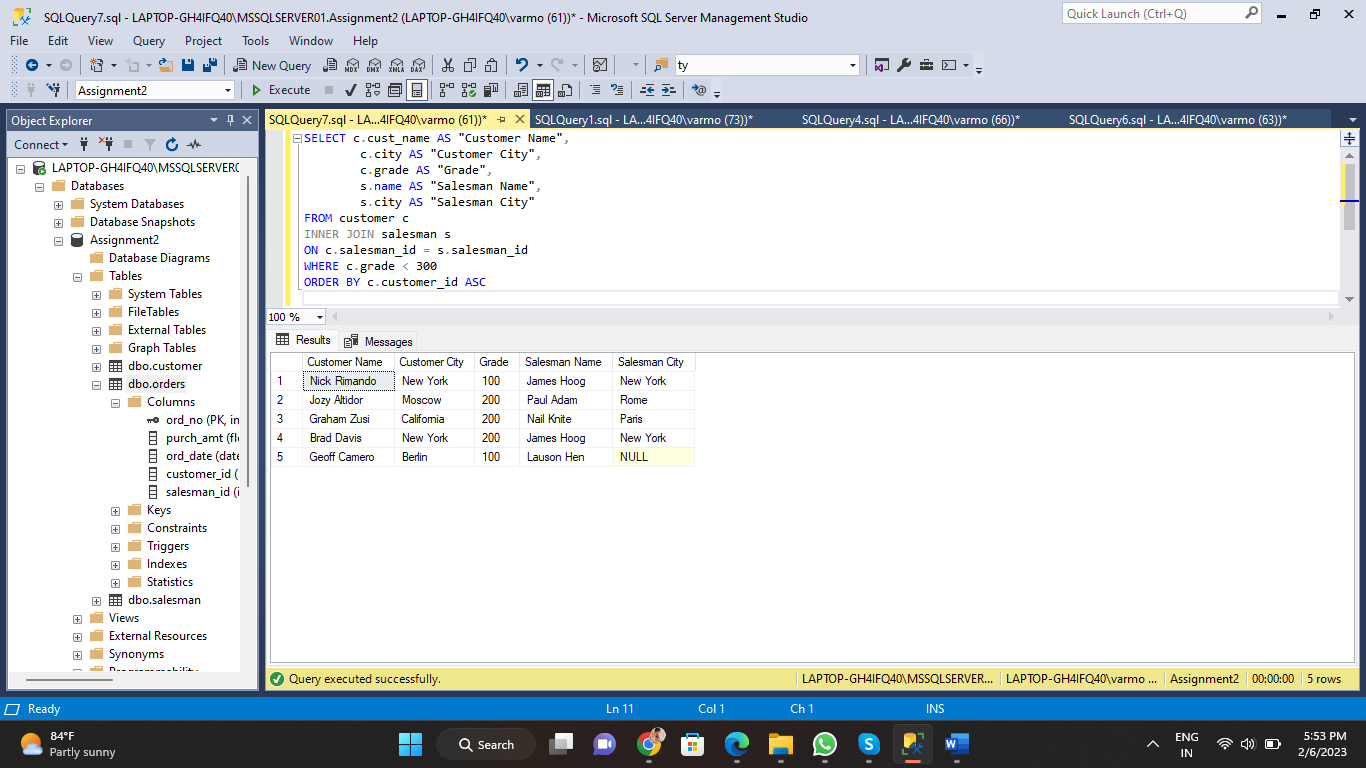
FROM customer c

INNER JOIN salesman s

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

WHERE c.grade < 300

ORDER BY c.customer\_id ASC;



1. 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 AS "Customer Name",

c.city AS "Customer City",

o.ord\_no AS "Order Number",

o.ord\_date AS "Order Date",

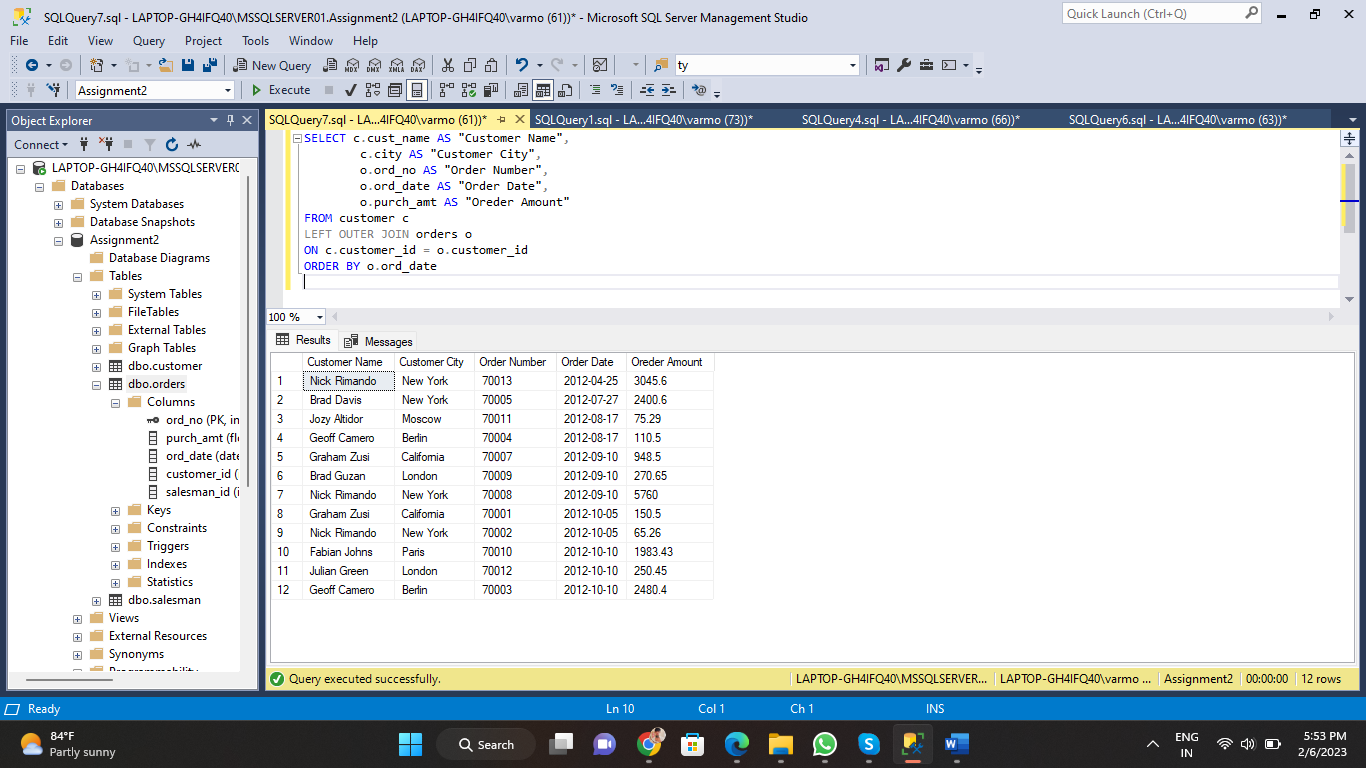
o.purch\_amt AS "Oreder Amount"

FROM customer c

LEFT OUTER JOIN orders o

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

ORDER BY o.ord\_date;



1. 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 AS "Customer Name",

c.city AS "Customer City",

o.ord\_no AS "Order Number",

o.ord\_date AS "Order Date",

o.purch\_amt AS "Oreder Amount",

s.name AS "Salesperson Name",

s.commission AS "SAles Commision"

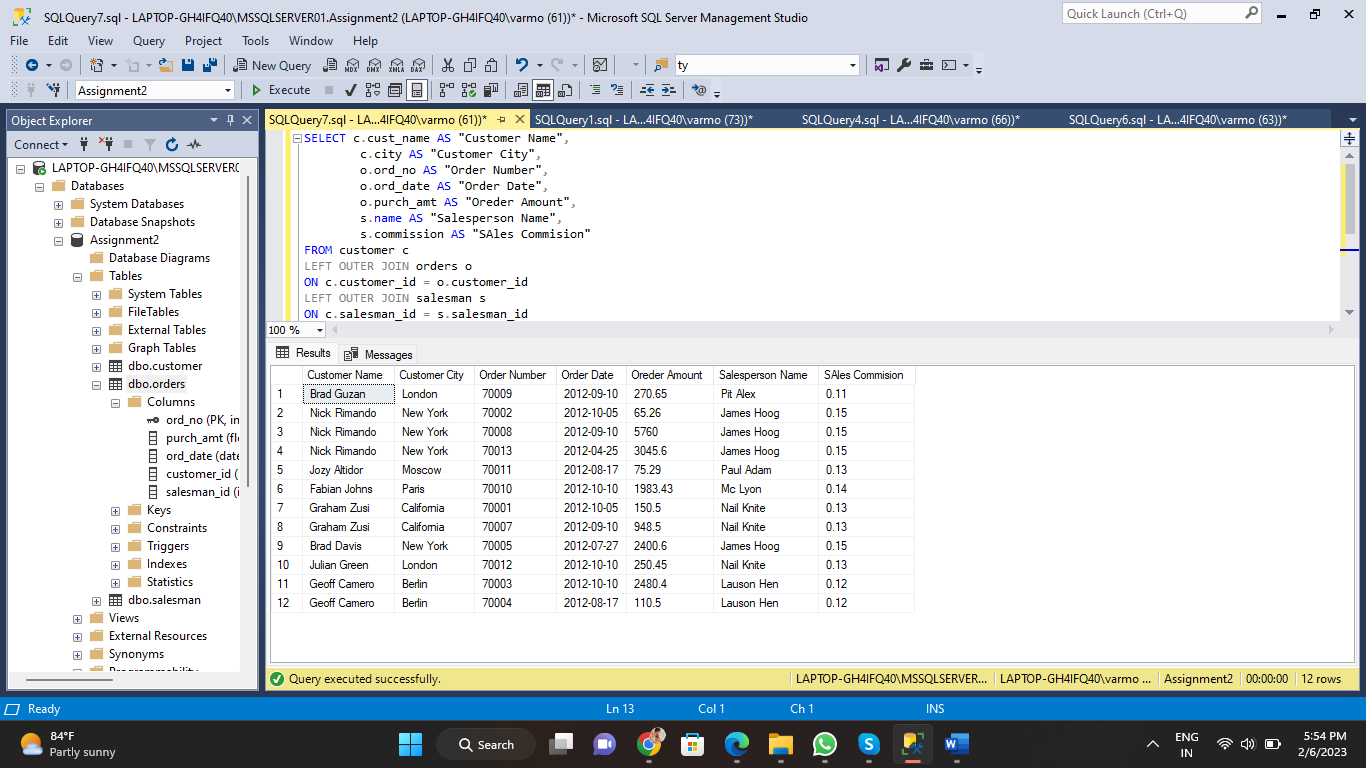
FROM customer c

LEFT OUTER JOIN orders o

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

LEFT OUTER JOIN salesman s

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



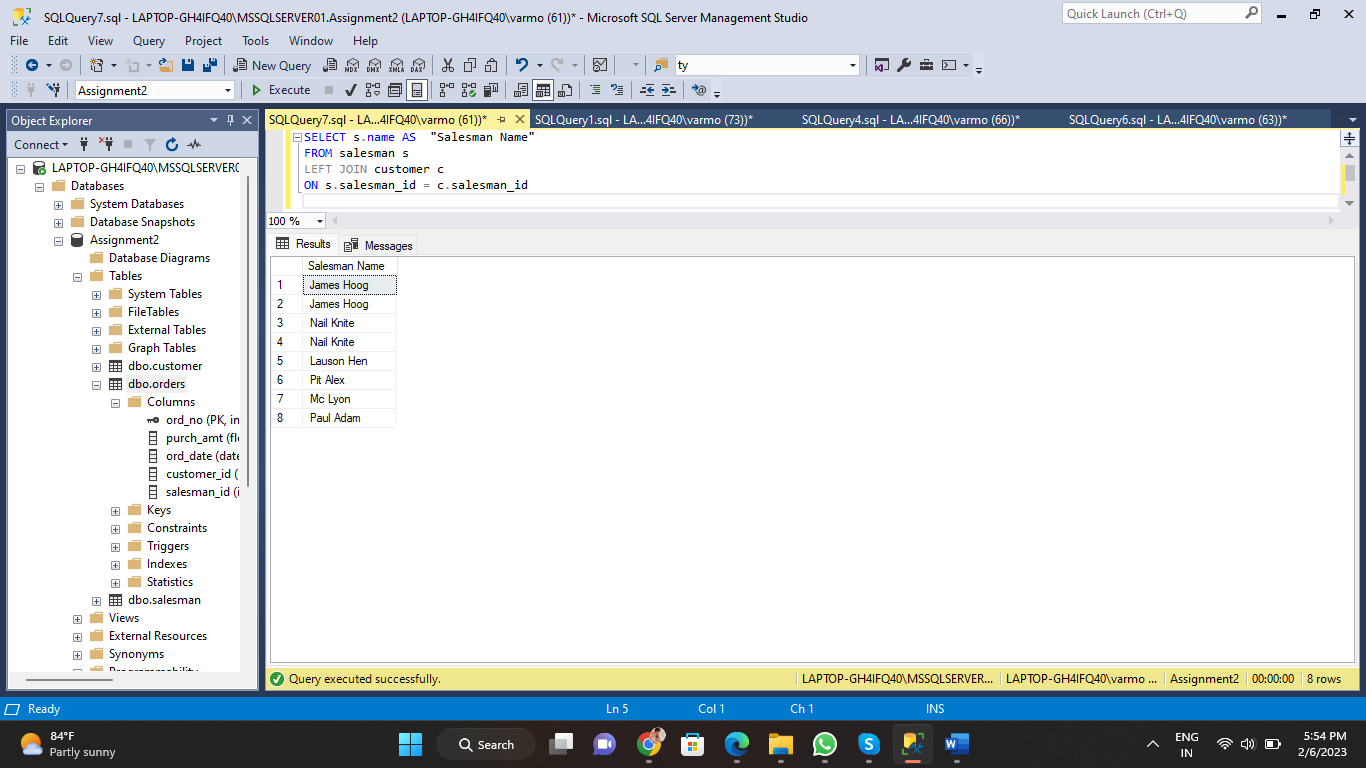
1. Write a SQL statement to generate a list in ascending order of salespersons who work either for one or more customers or have not yet joined any of the customers

SELECT s.name AS "Salesman Name"

FROM salesman s

LEFT JOIN customer c

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



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

SELECT c.cust\_name AS "Customer Name",

c.city AS "Customer City",

c.grade AS "Grade",

o.ord\_no AS "Order Number",

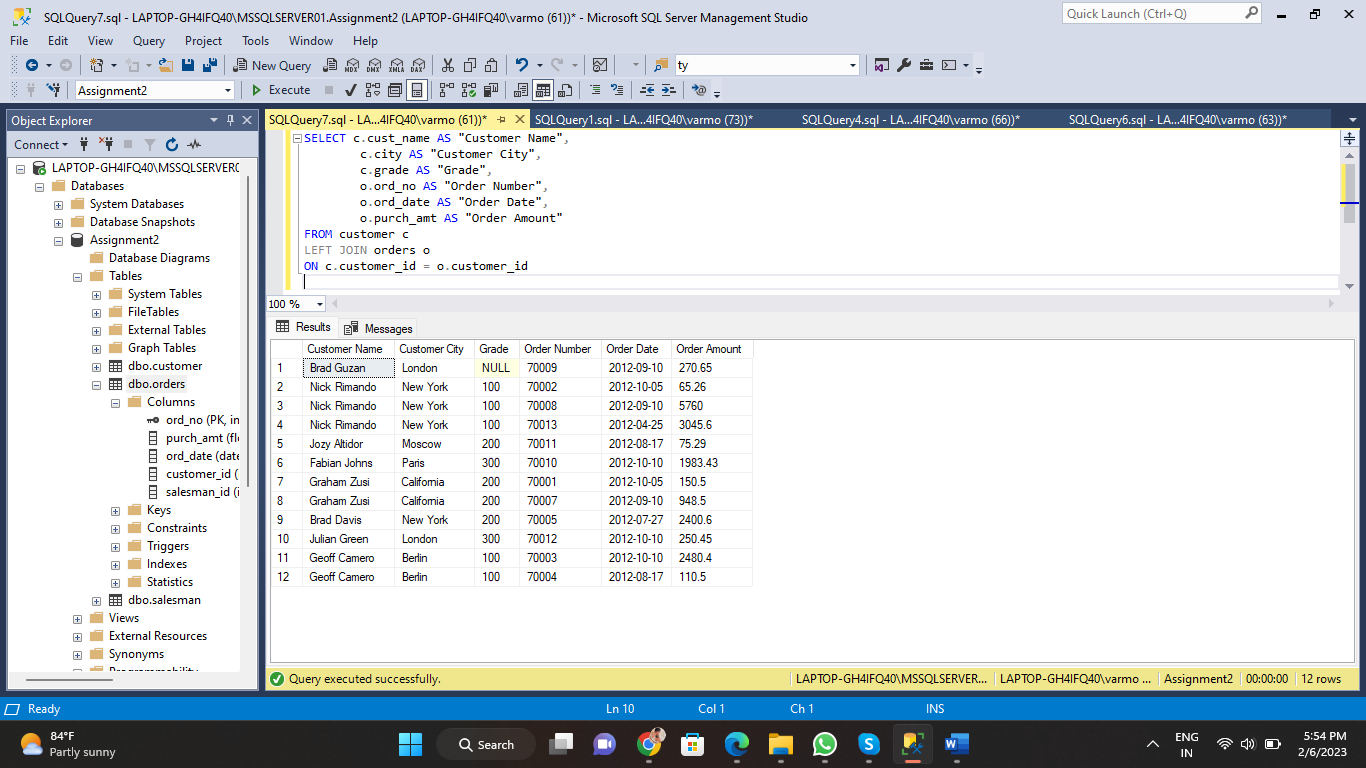
o.ord\_date AS "Order Date",

o.purch\_amt AS "Order Amount"

FROM customer c

LEFT JOIN orders o

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



1. 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"

FROM salesman s

LEFT OUTER JOIN customer c

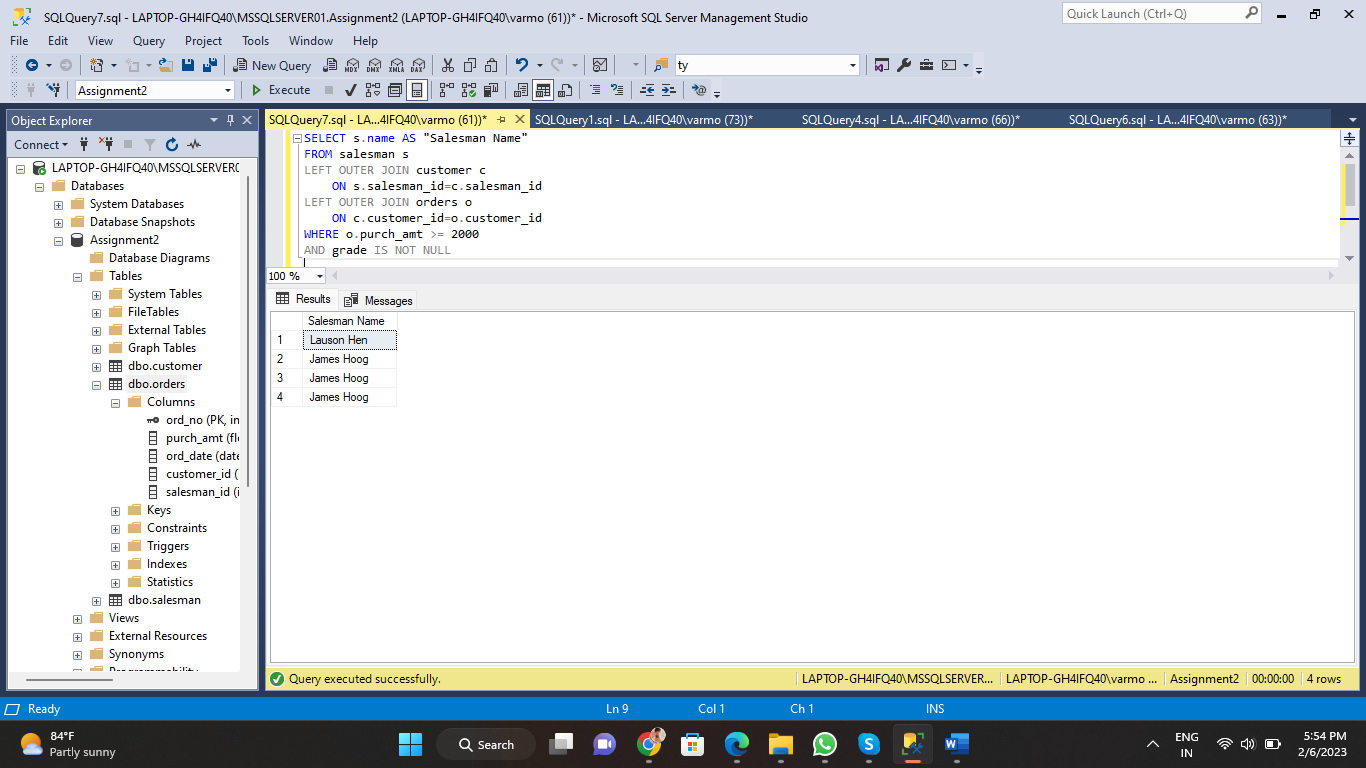
ON s.salesman\_id=c.salesman\_id

LEFT OUTER JOIN orders o

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

WHERE o.purch\_amt >= 2000

AND grade IS NOT NULL;



1. 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"

FROM salesman s

LEFT OUTER JOIN customer c

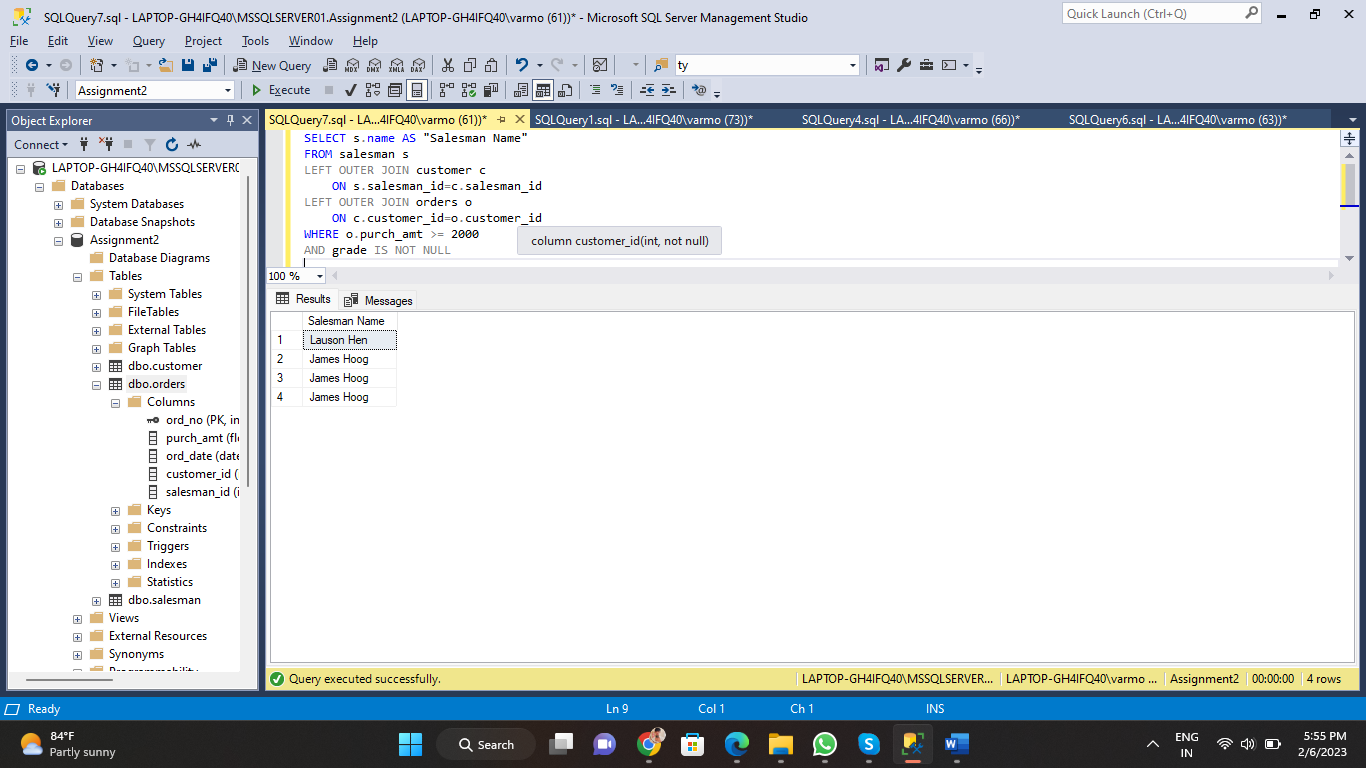
ON s.salesman\_id=c.salesman\_id

LEFT OUTER JOIN orders o

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

WHERE o.purch\_amt >= 2000

AND grade IS NOT NULL;



1. 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 AS "Customer Name",

c.city AS "Customer City",

o.ord\_no AS "Order Number",

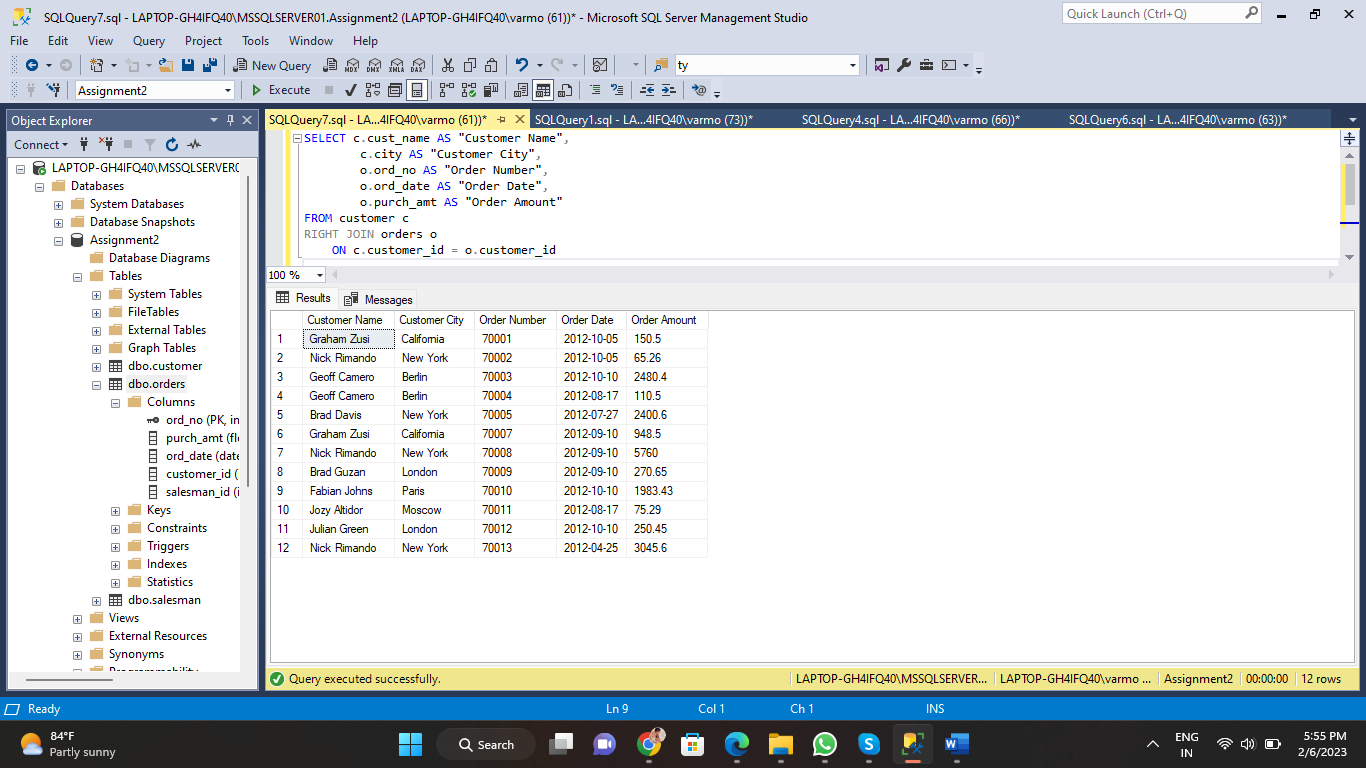
o.ord\_date AS "Order Date",

o.purch\_amt AS "Order Amount"

FROM customer c

RIGHT JOIN orders o

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



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

SELECT s.salesman\_id,

s.name,

s.city,

s.commission,

c.customer\_id,

c.cust\_name,

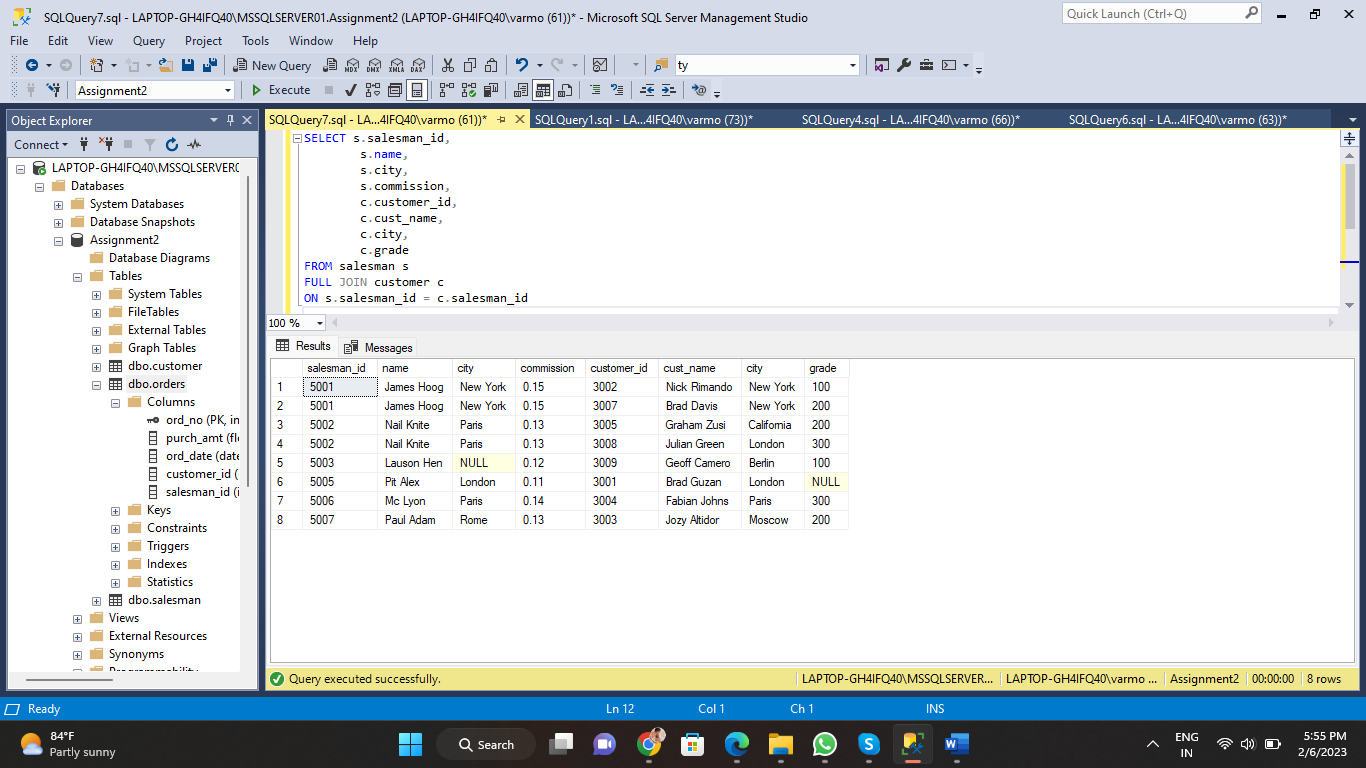
c.city,

c.grade

FROM salesman s

FULL JOIN customer c

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



1. 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

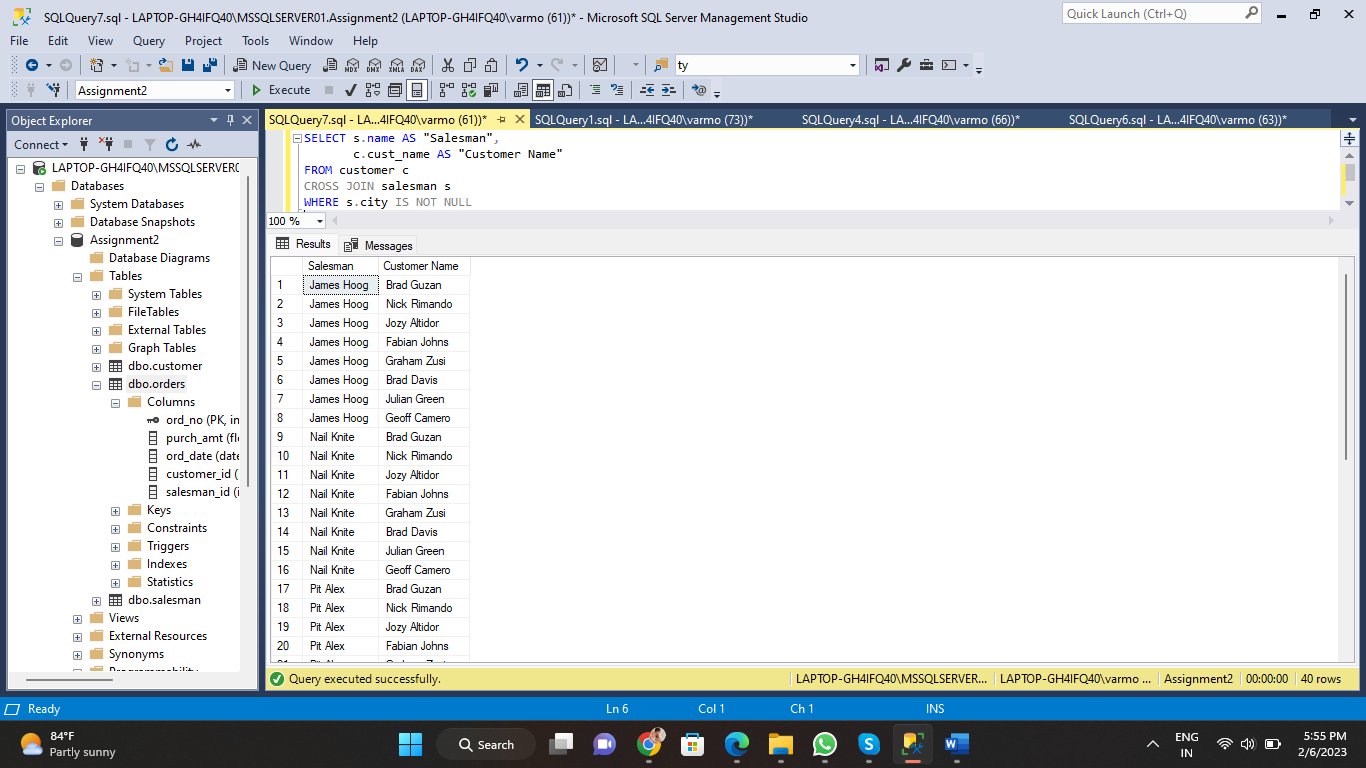
SELECT s.name AS "Salesman",

c.cust\_name AS "Customer Name"

FROM customer c

CROSS JOIN salesman s

WHERE s.city IS NOT NULL;



1. 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 s.name AS "Salesman",

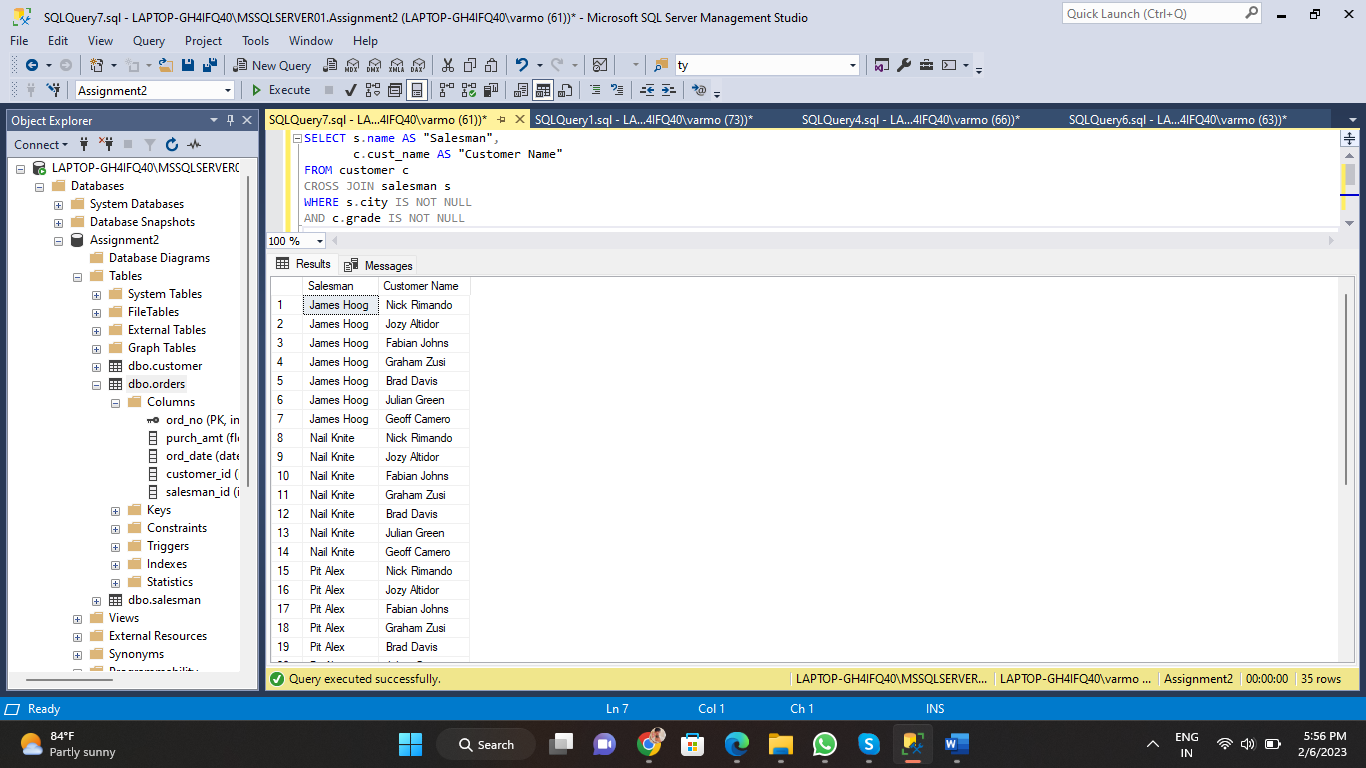
c.cust\_name AS "Customer Name"

FROM customer c

CROSS JOIN salesman s

WHERE s.city IS NOT NULL

AND c.grade IS NOT NULL;



1. 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 s.name AS "Salesman",

c.cust\_name AS "Customer Name"

FROM customer c

CROSS JOIN salesman s

WHERE s.city IS NOT NULL

AND c.city != s.city

AND c.grade IS NOT NULL;

