Create tables: salesman, customer, orders

create table salesman

(

salesman\_id int Primary Key,

name varchar(50) not null,

city varchar(50) not null,

commission decimal(5,2)

)

create table customer

(

customer\_id int primary key,

cust\_name varchar(50) not null,

city varchar(50) not null,

grade int not null,

salesman\_id int

)

create table orders

(

ord\_no int primary key,

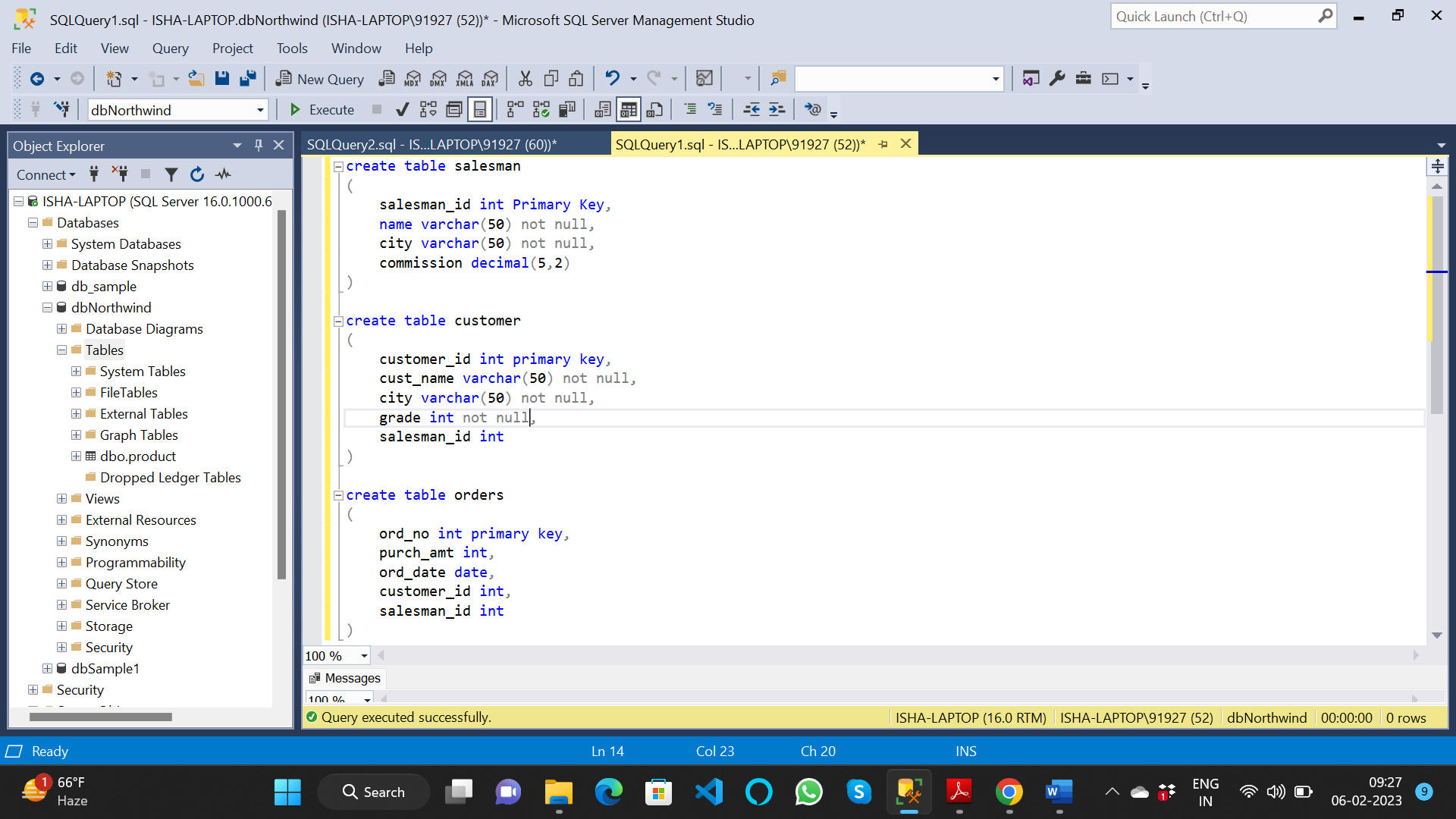
purch\_amt int,

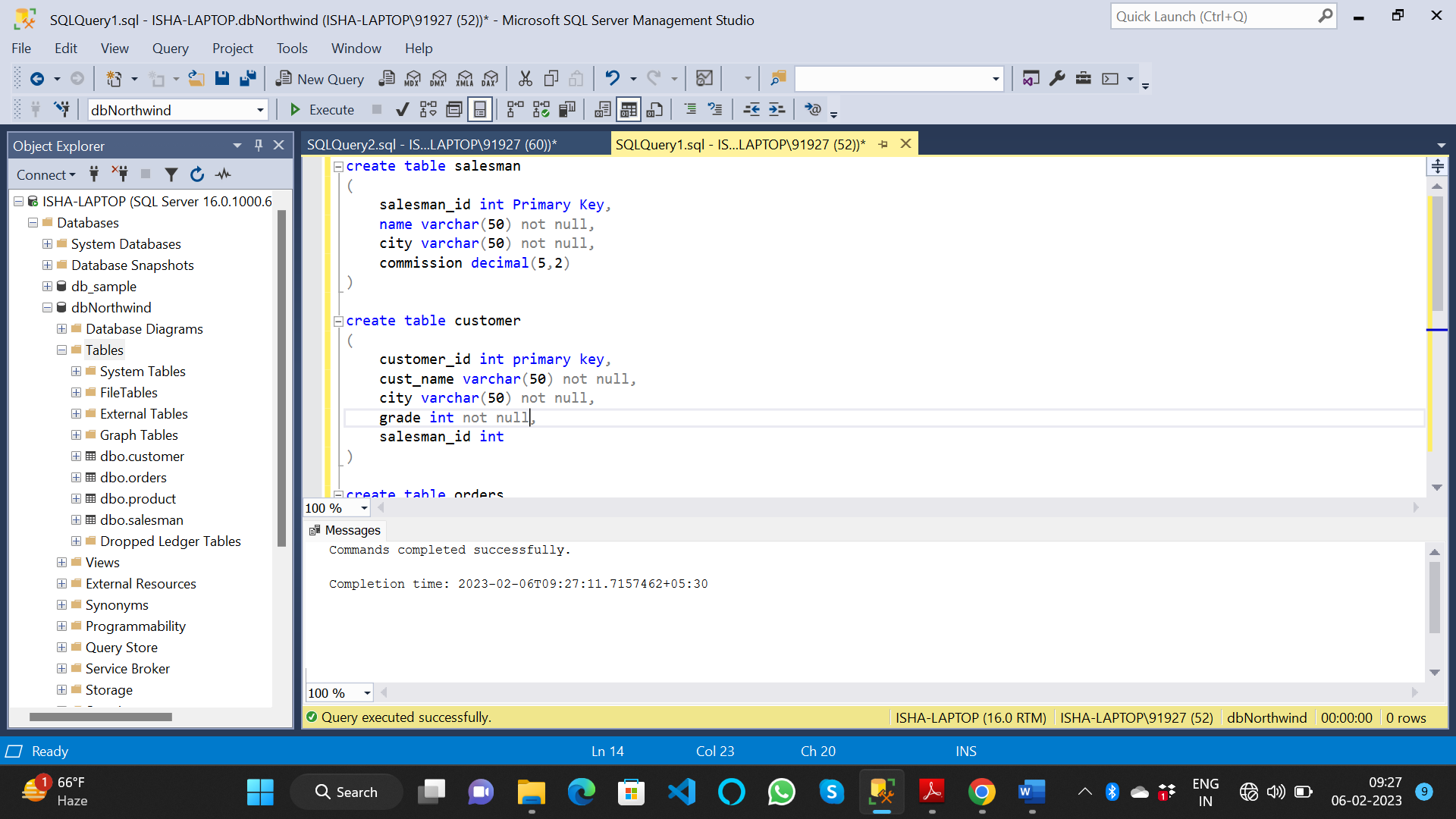
ord\_date date,

customer\_id int,

salesman\_id int

)





Insert values to salesman table

insert into orders values(001,250,'2022-12-25',2007,101)

insert into orders values(002,1050,'2023-01-28',2003,107)

insert into orders values(003,500,'2023-01-15',2008,105)

insert into orders values(004,750,'2022-12-29',2001,106)

insert into orders values(005,2500,'2023-12-10',2009,107)

insert into orders values(006,3080,'2023-01-19',2011,105)

insert into orders values(007,3000,'2023-02-05',2018,107)

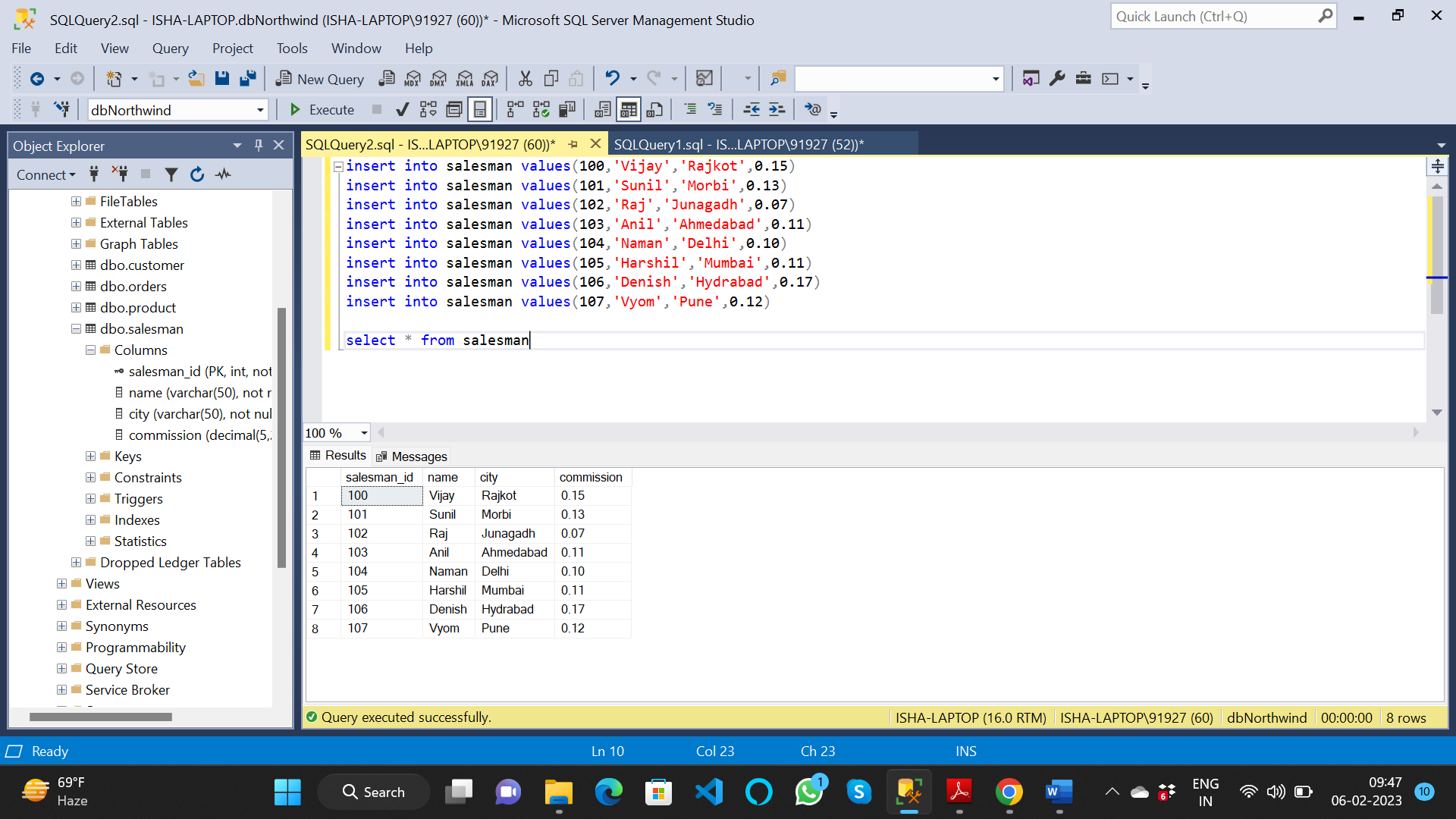
insert into orders values(008,1590,'2023-01-07',2007,101)

insert into orders values(009,210,'2023-01-13',2009,107)

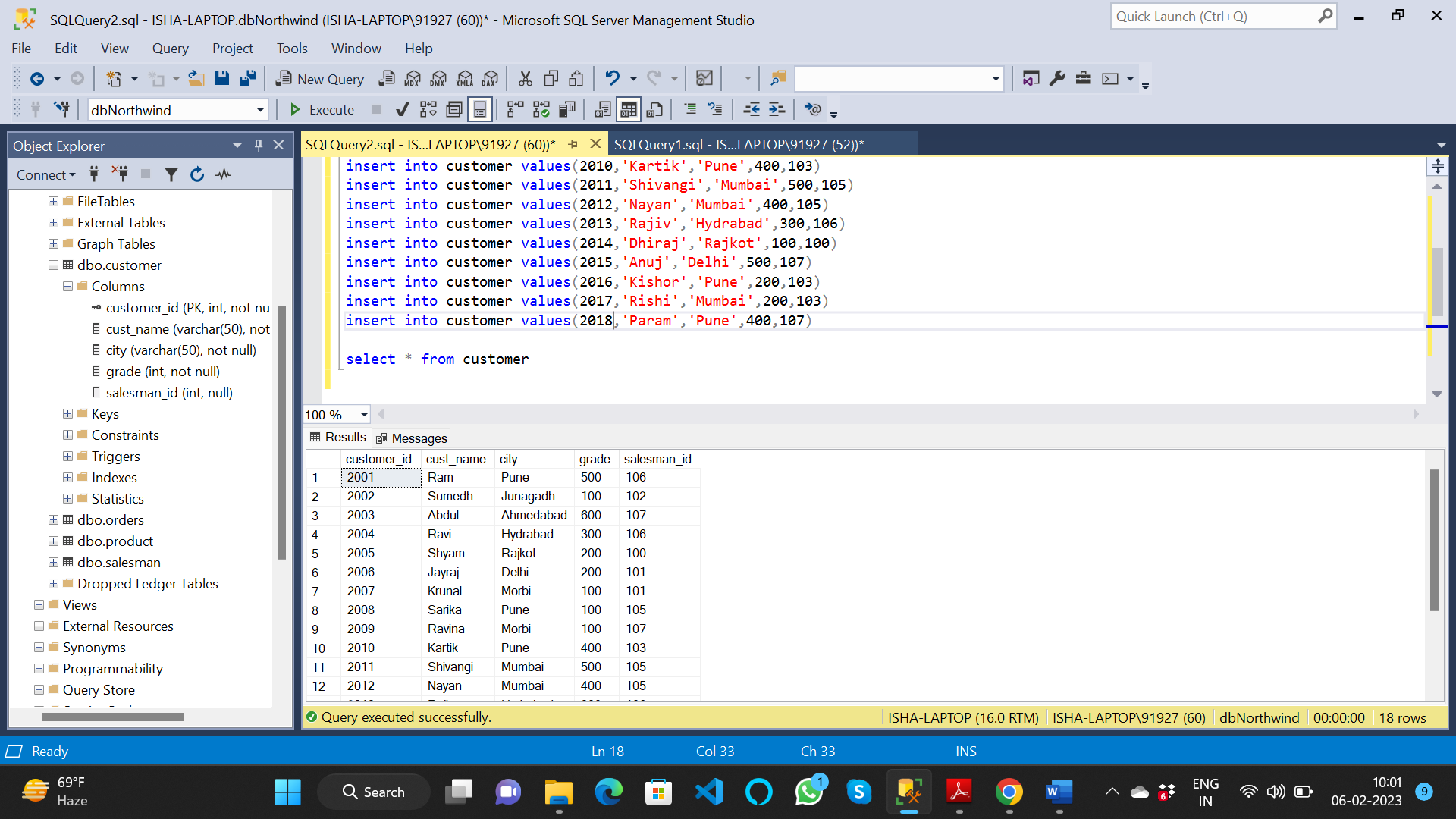
insert into orders values(010,690,'2023-01-29',2002,102)

insert into orders values(011,1350,'2023-02-01',2018,107)

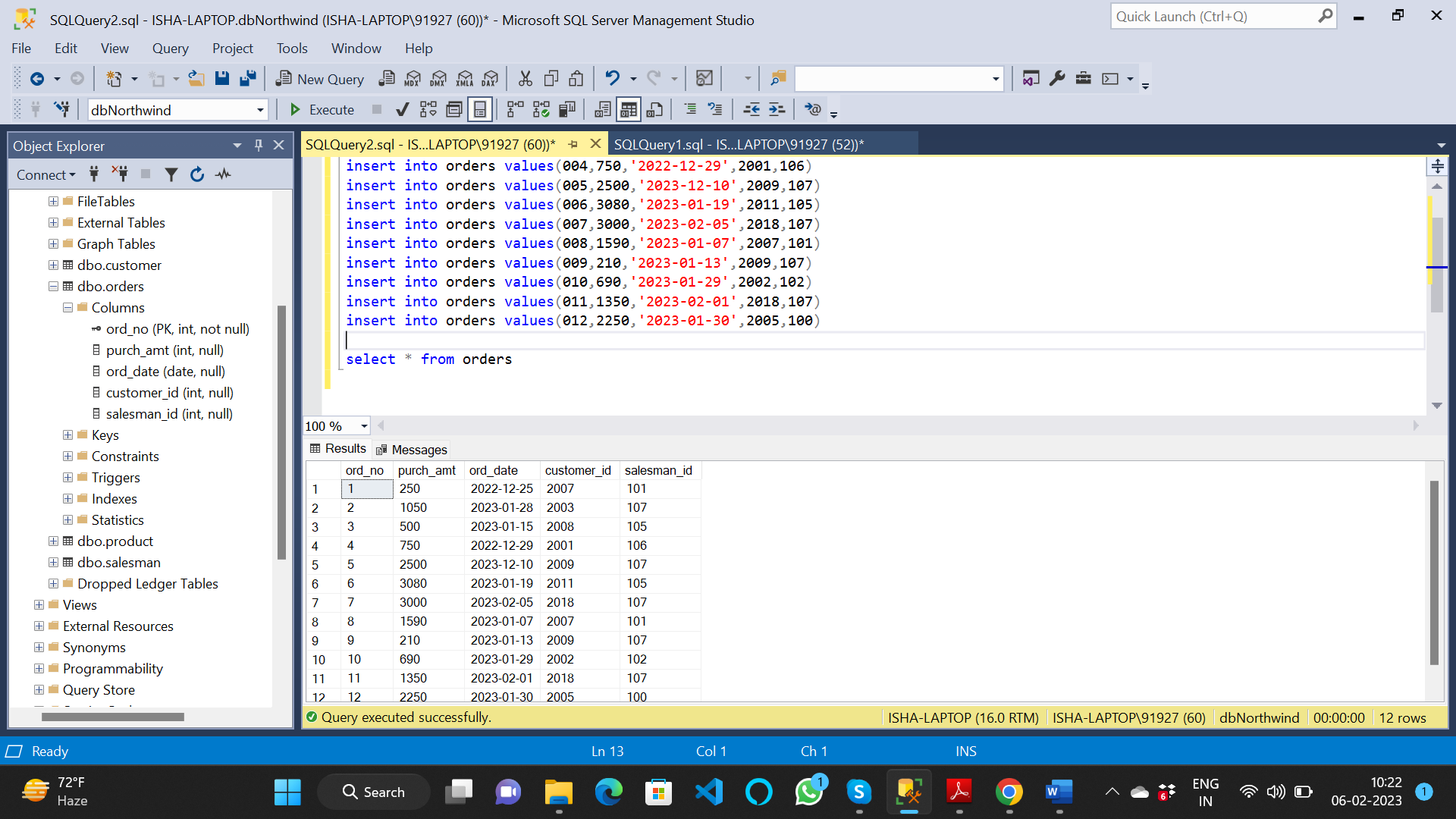
insert into orders values(012,2250,'2023-01-30',2005,100)



Insert values to customer table



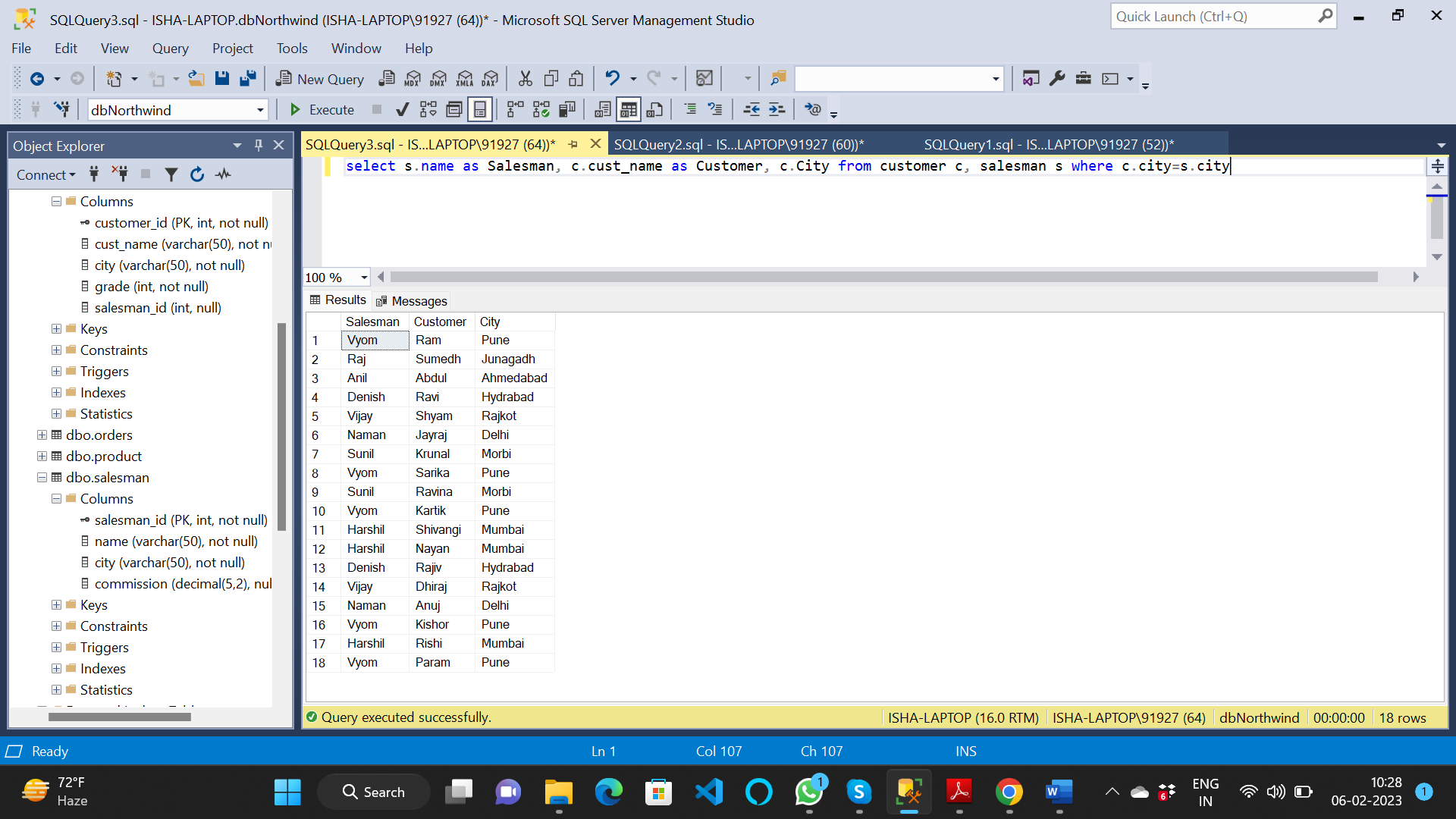
Insert values to orders table



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, c.cust\_name as Customer, c.city from customer c, salesman s where c.city=s.city



2. write a SQL query to find those orders where the order amount exists between 500

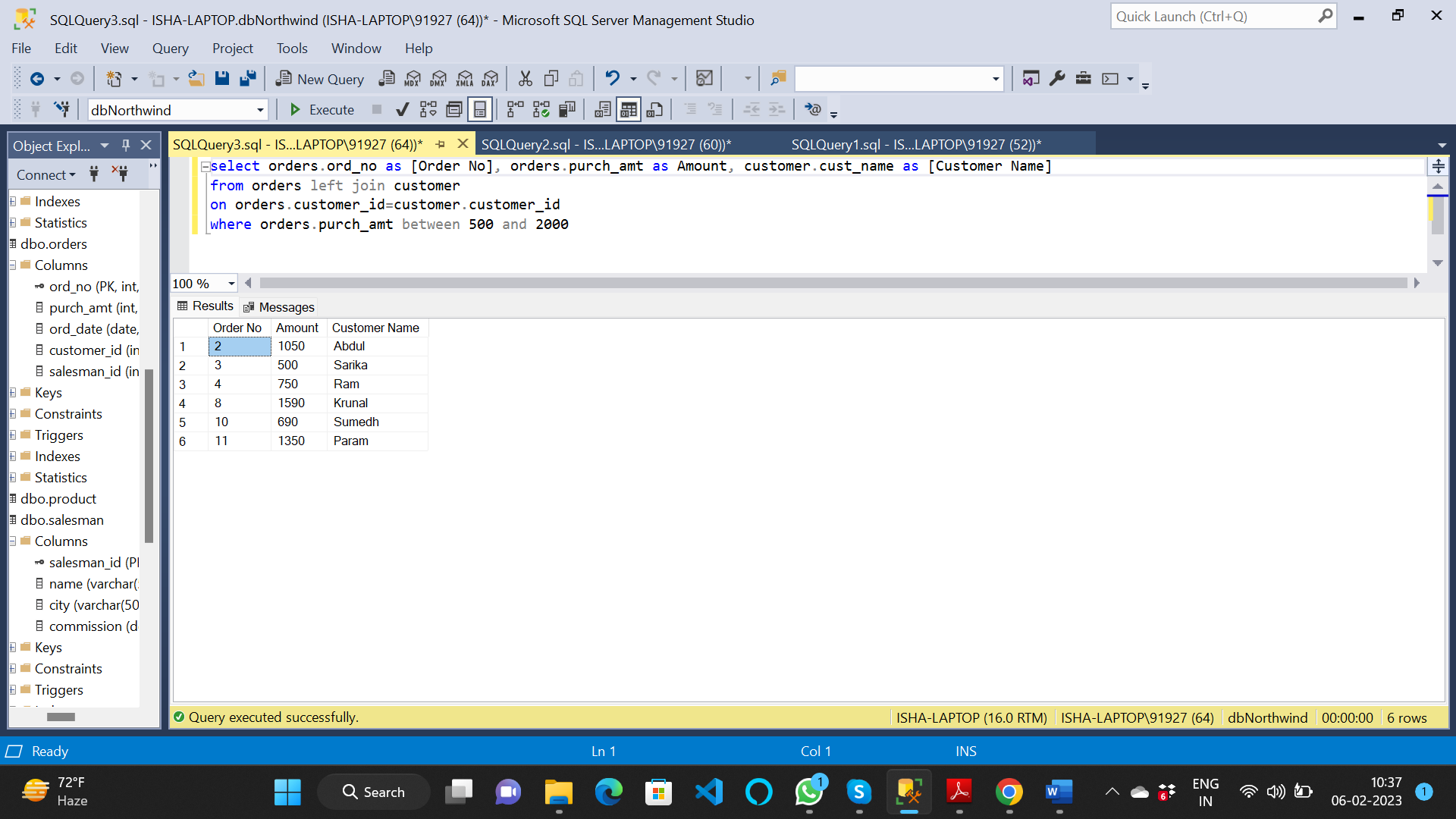
and 2000. Return ord\_no, purch\_amt, cust\_name, city

select orders.ord\_no as [Order No], orders.purch\_amt as Amount, customer.cust\_name as [Customer Name]

from orders left join customer

on orders.customer\_id=customer.customer\_id

where orders.purch\_amt between 500 and 2000



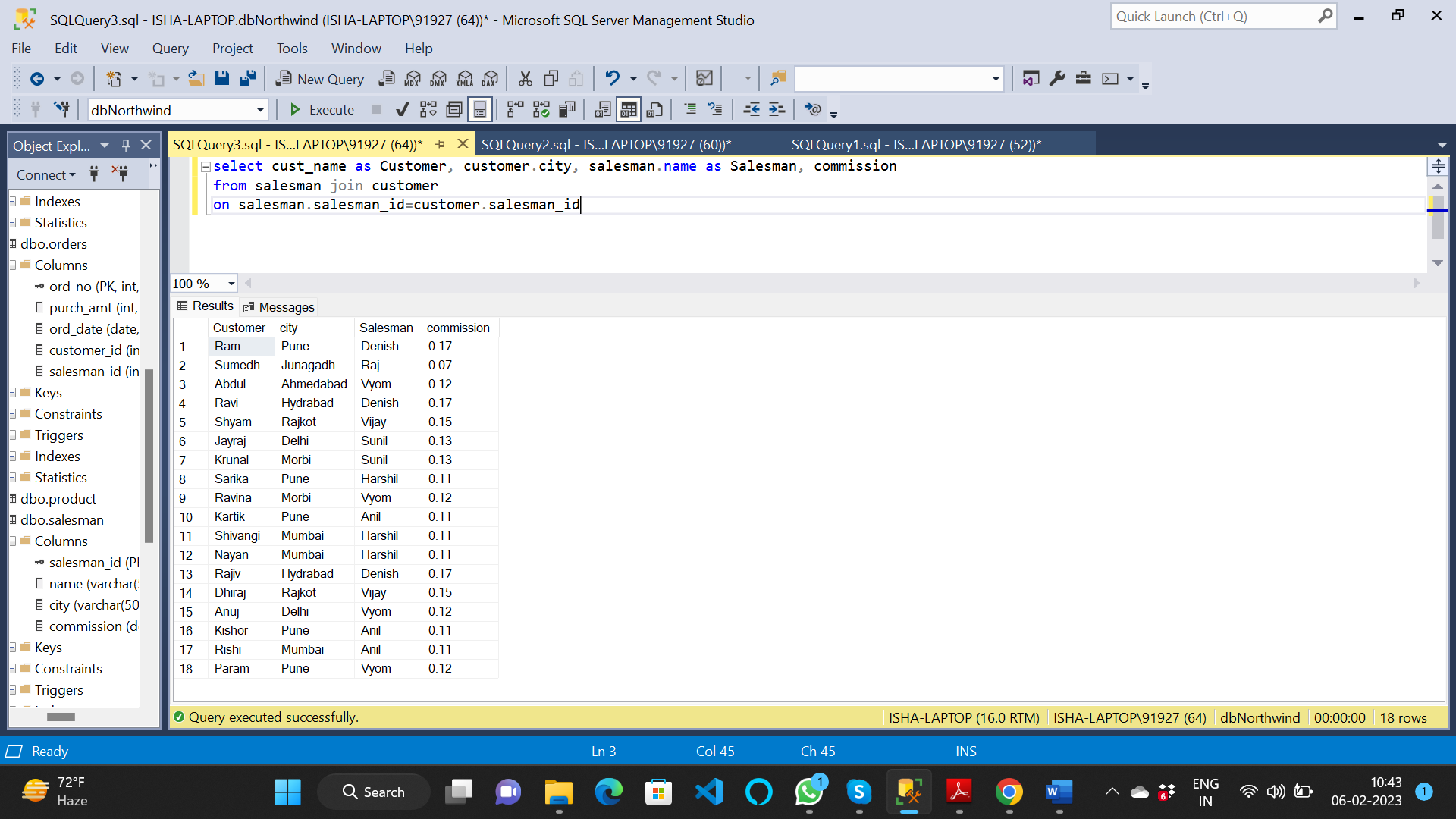
3. write a SQL query to find the salesperson(s) and the customer(s) he represents.

Return Customer Name, city, Salesman, commission

select cust\_name as Customer, customer.city, salesman.name as Salesman, commission

from salesman 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,

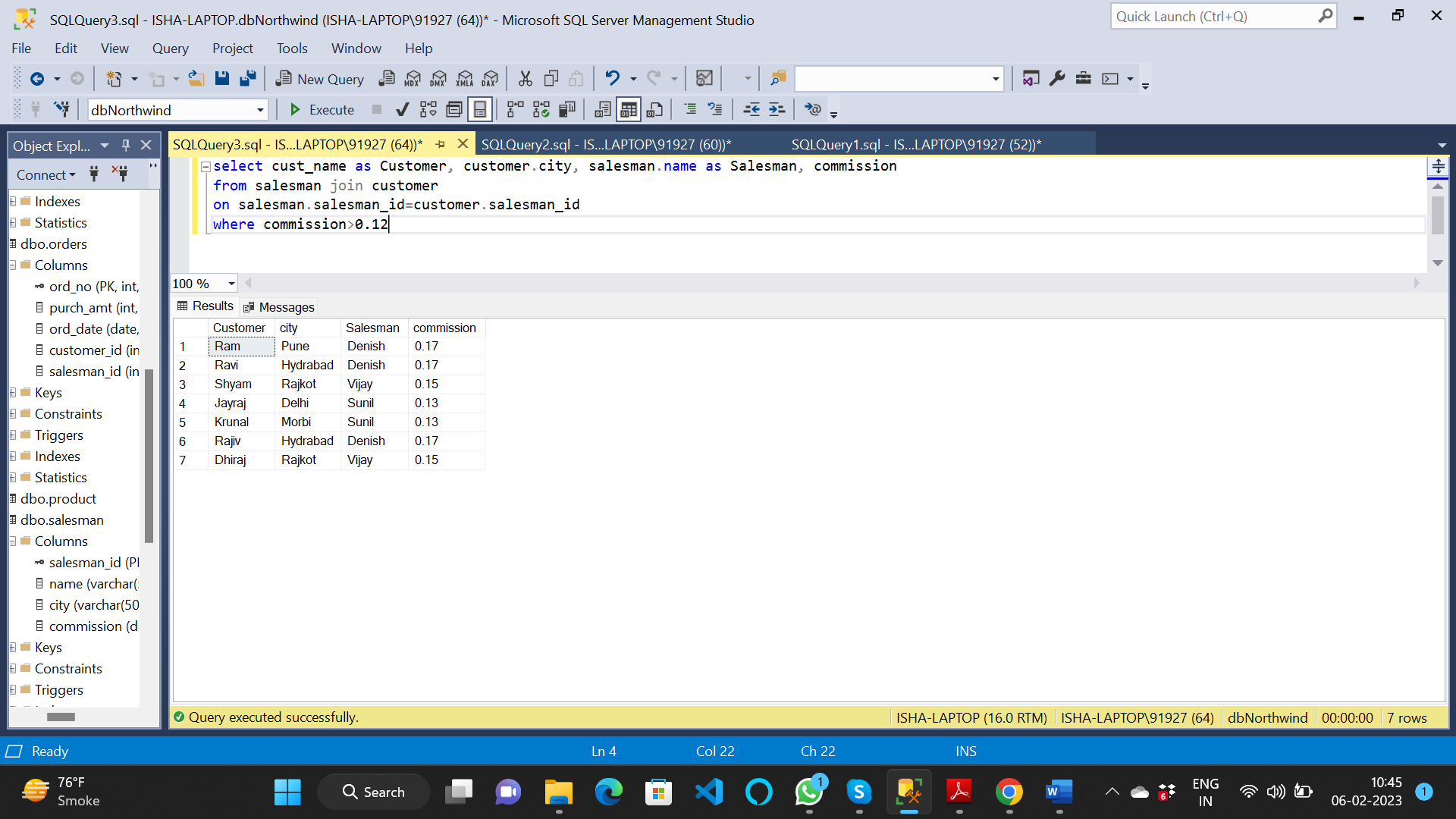
commission.

select cust\_name as Customer, customer.city, salesman.name as Salesman, commission

from salesman join customer

on salesman.salesman\_id=customer.salesman\_id

where 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,

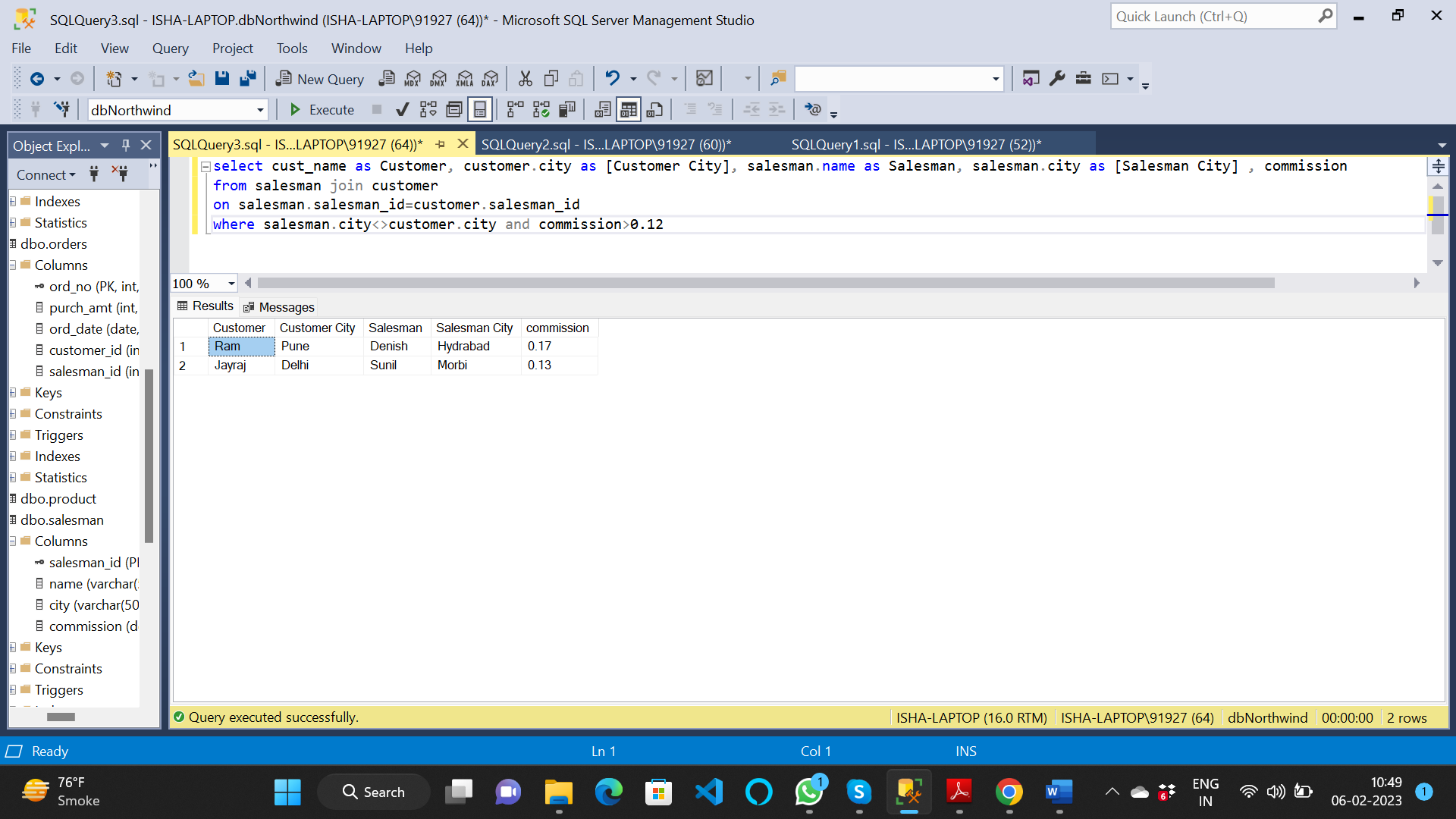
commission

select cust\_name as Customer, customer.city as [Customer City], salesman.name as Salesman, salesman.city as [Salesman City] , commission

from salesman 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

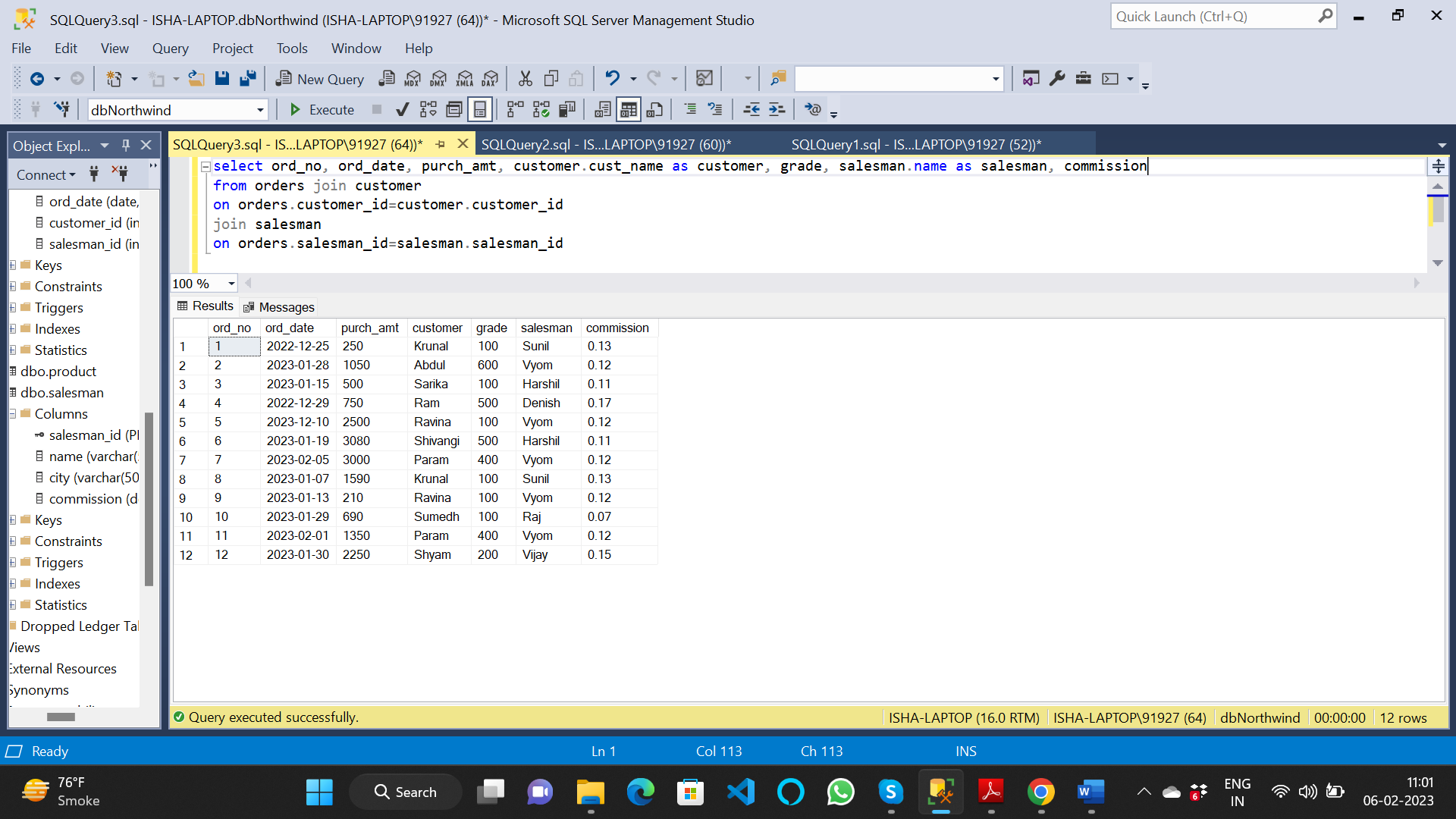
select ord\_no, ord\_date, purch\_amt, customer.cust\_name as customer, grade, salesman.name as salesman, commission

from orders join customer

on orders.customer\_id=customer.customer\_id

join salesman

on orders.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 orders.\*, customer.cust\_name as customer, customer.city as [customer city], grade as [customer grade],

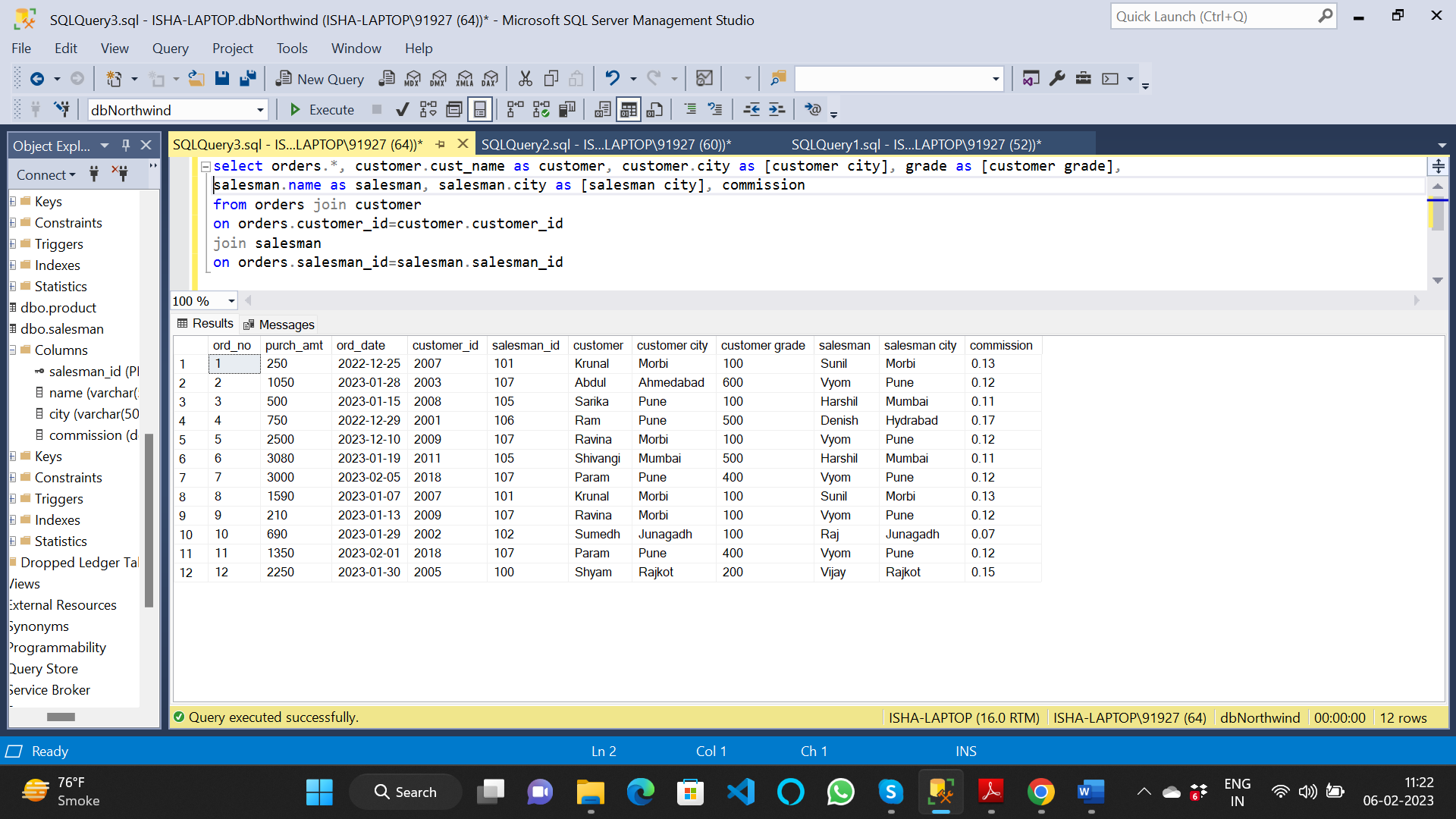
salesman.name as salesman, salesman.city as [salesman city], commission

from orders join customer

on orders.customer\_id=customer.customer\_id

join salesman

on orders.salesman\_id=salesman.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.

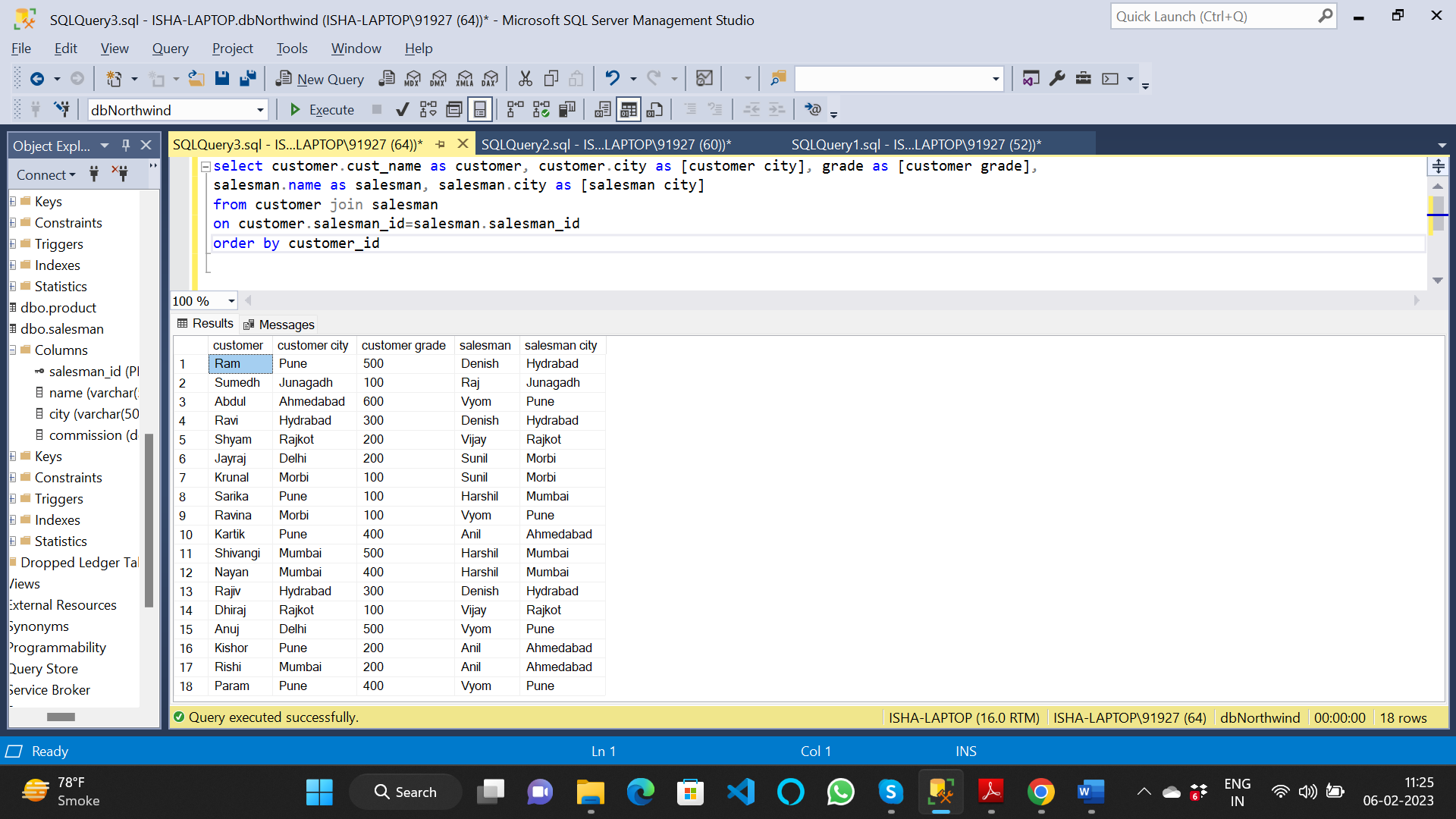
select customer.cust\_name as customer, customer.city as [customer city], grade as [customer grade],

salesman.name as salesman, salesman.city as [salesman city]

from customer join salesman

on customer.salesman\_id=salesman.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.

select customer.cust\_name as customer, customer.city as [customer city], grade as [customer grade],

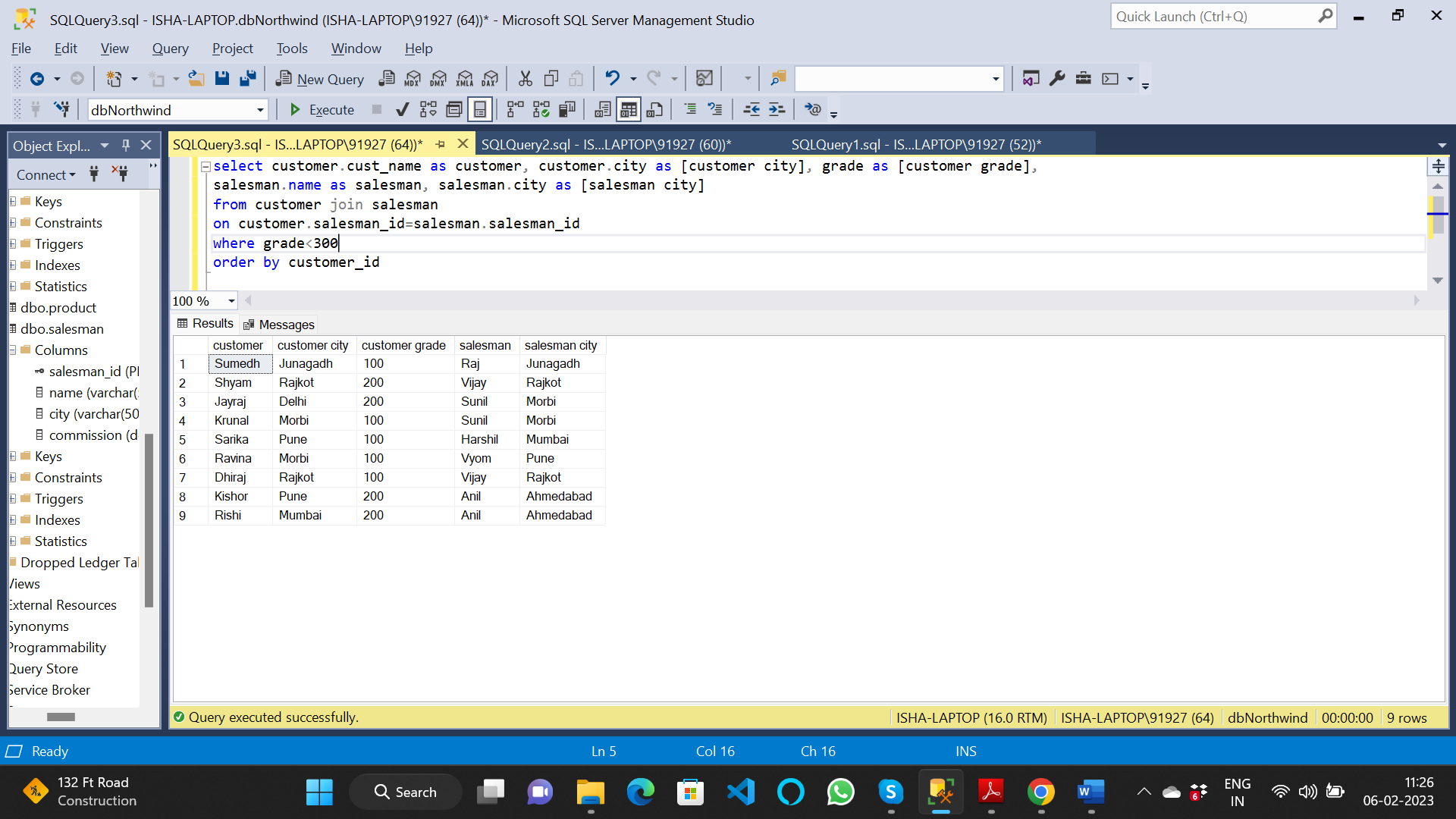
salesman.name as salesman, salesman.city as [salesman city]

from customer join salesman

on customer.salesman\_id=salesman.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

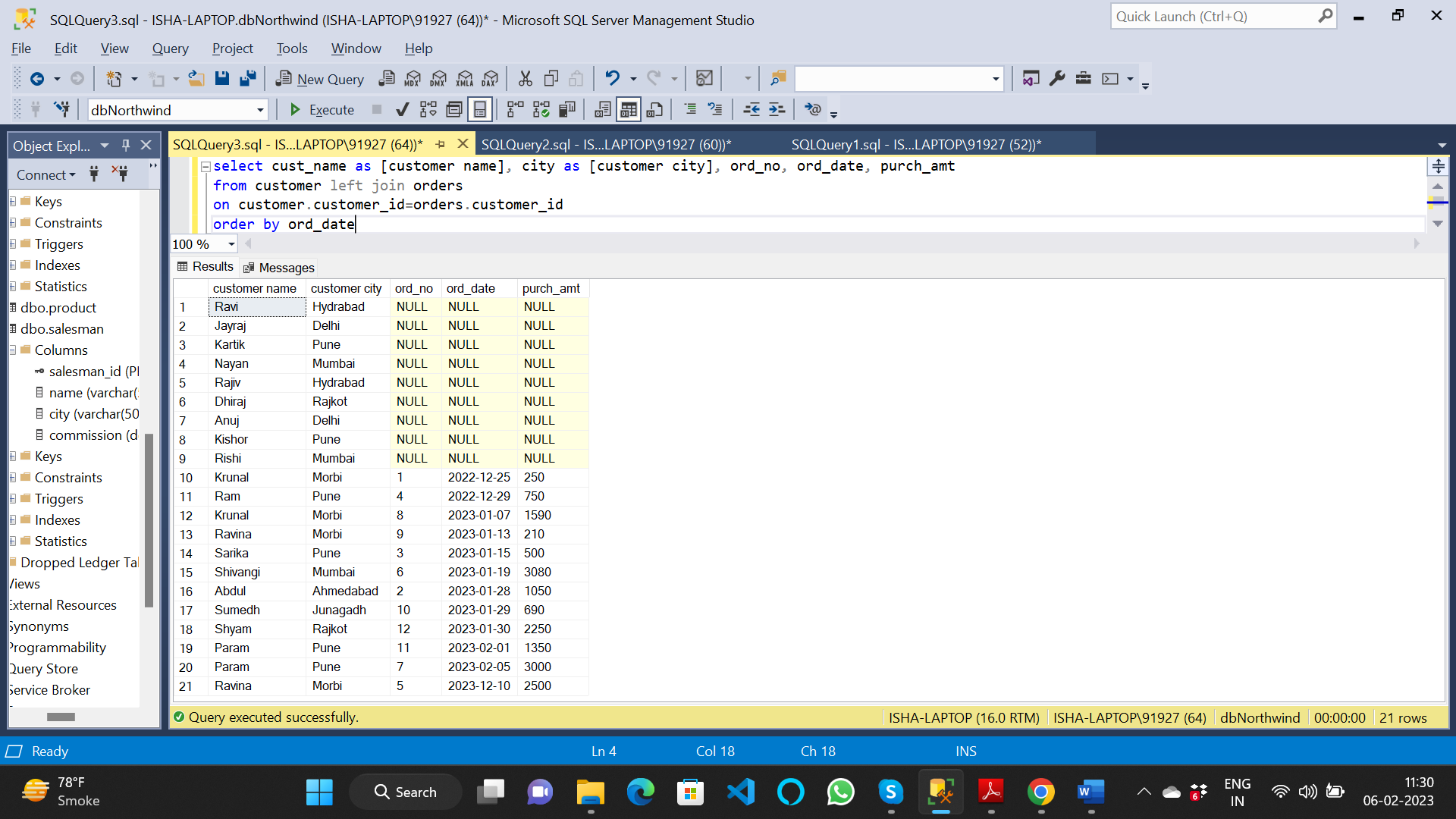
determine whether any of the existing customers have placed an order or not

select cust\_name as [customer name], city as [customer city], ord\_no, ord\_date, purch\_amt

from orders left join customer

on orders.customer\_id=customer.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

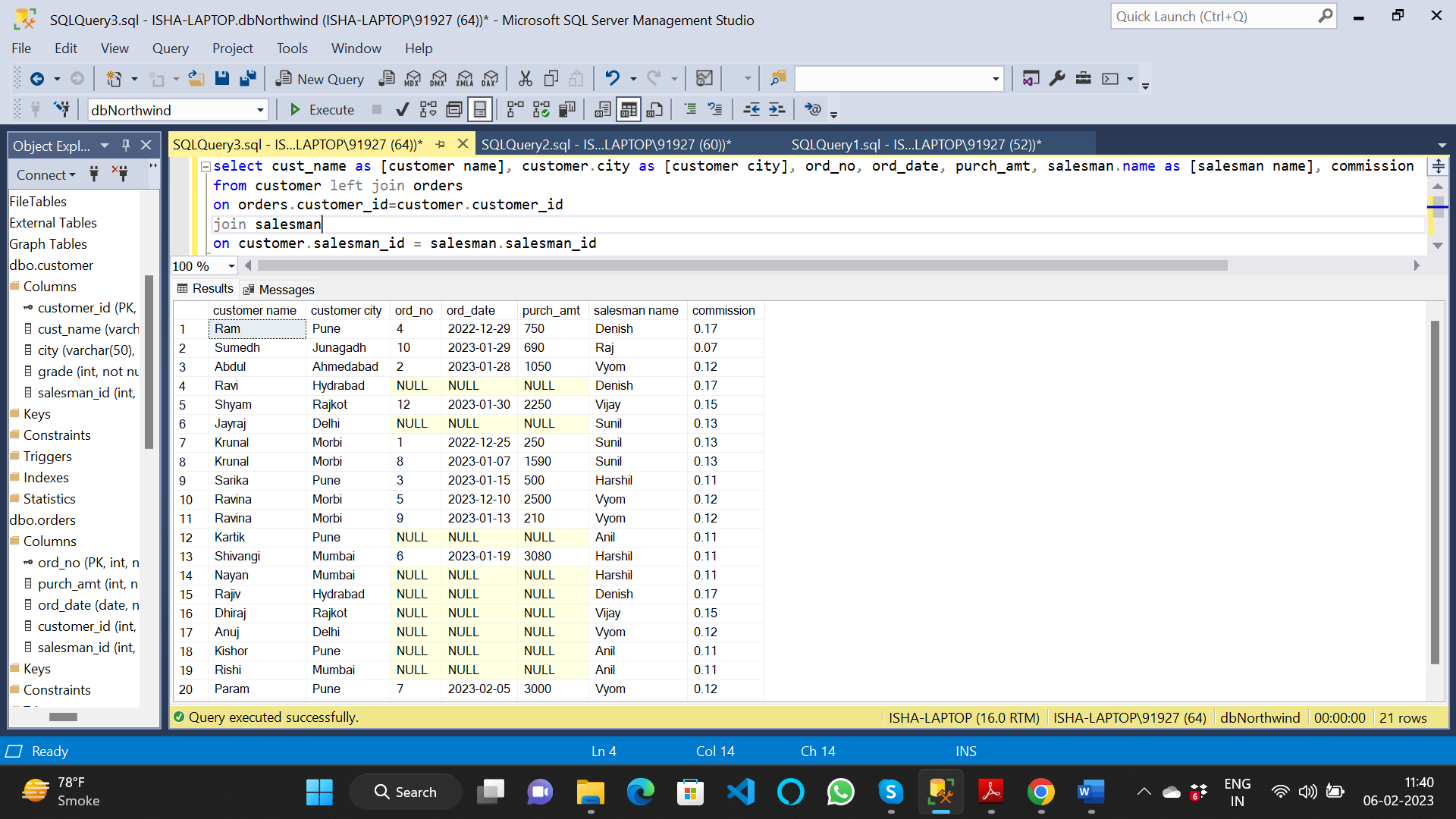
select cust\_name as [customer name], customer.city as [customer city], ord\_no, ord\_date, purch\_amt, salesman.name as [salesman name], commission

from customer left join orders

on orders.customer\_id=customer.customer\_id

join salesman

on customer.salesman\_id = salesman.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

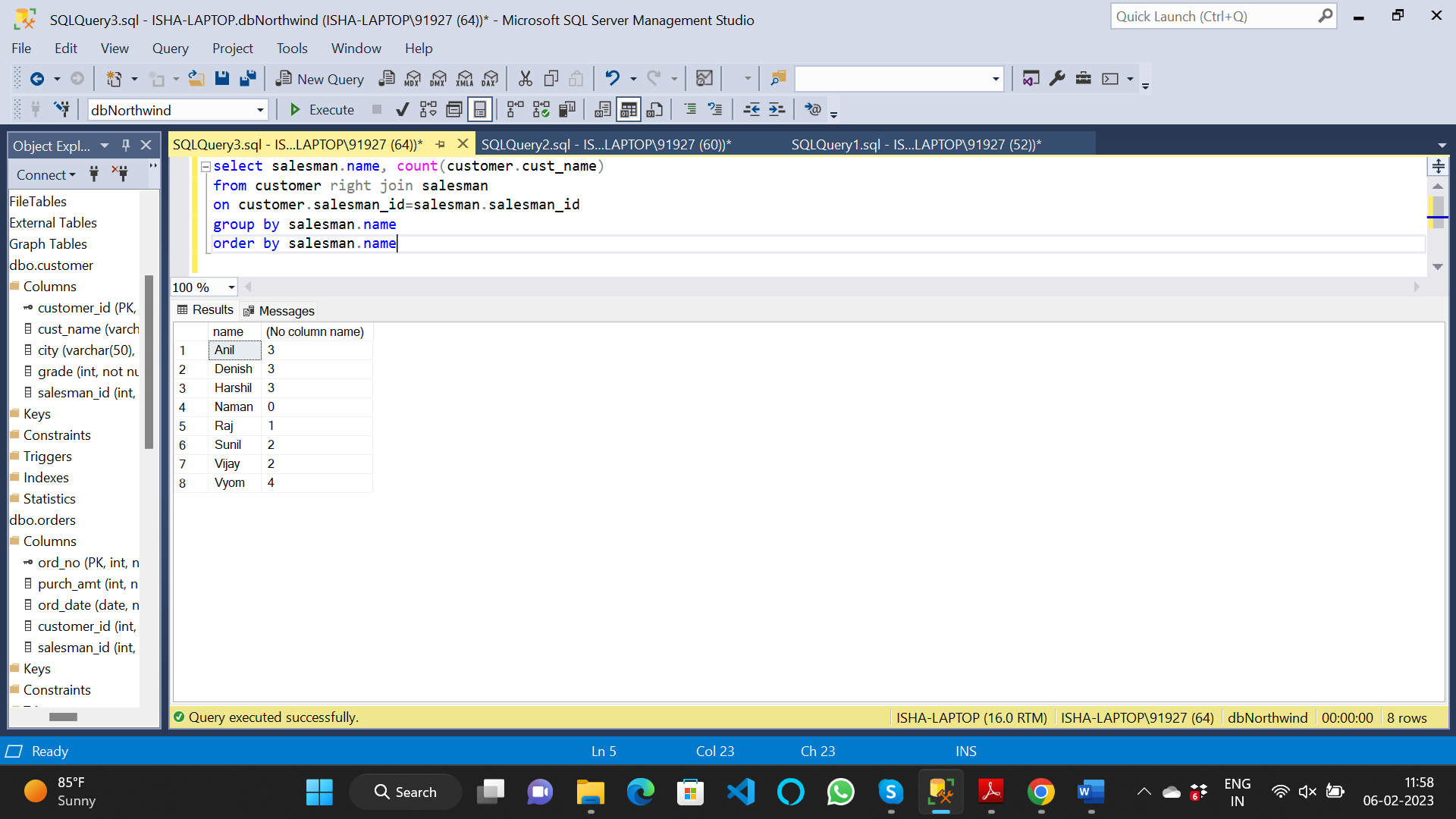
select salesman.name, count(customer.cust\_name)

from customer right join salesman

on customer.salesman\_id=salesman.salesman\_id

group by salesman.name

order by salesman.name



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, cust\_name as customer, customer.city as [customer city], grade, ord\_no, ord\_date, purch\_amt

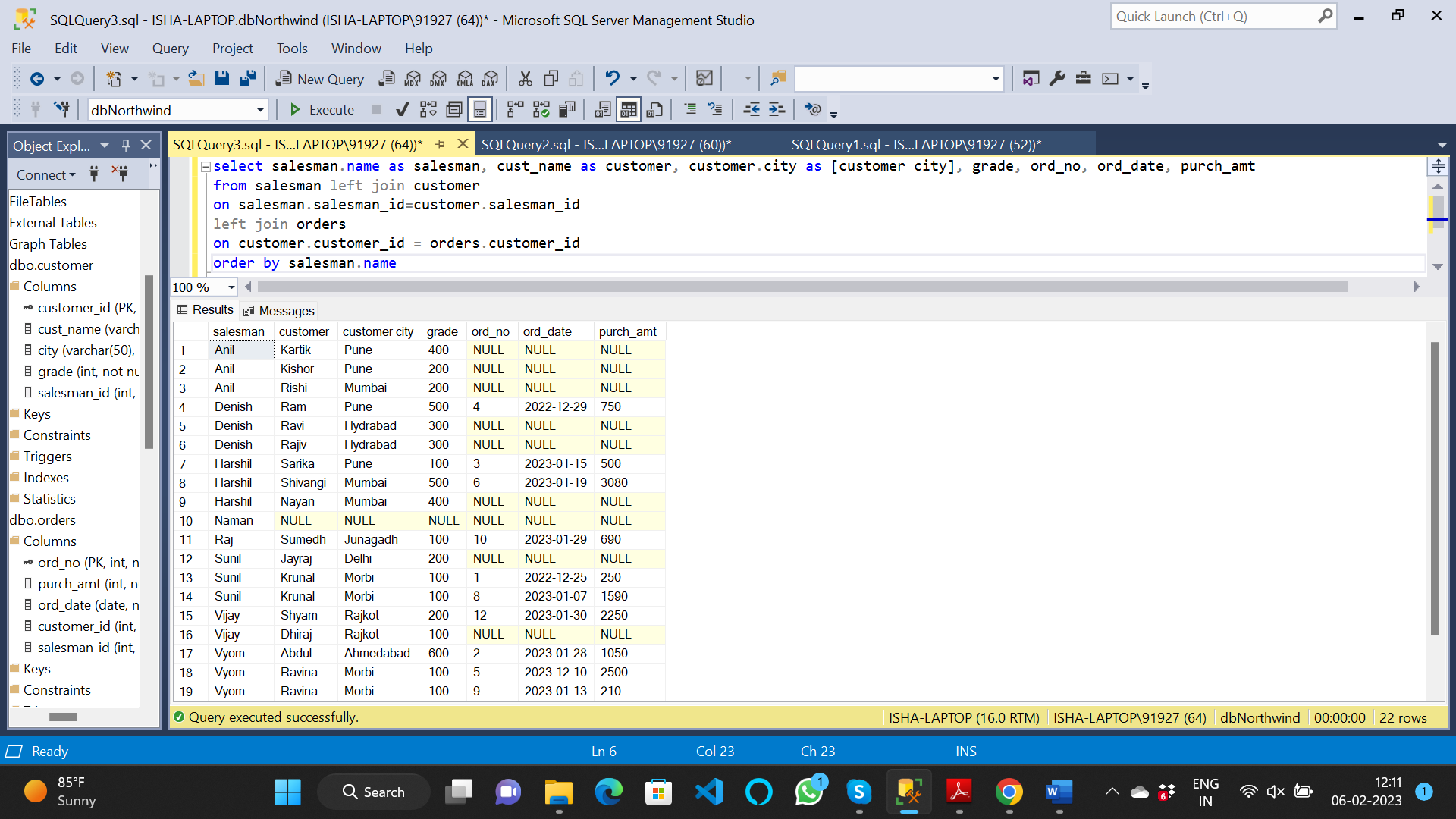
from salesman left join customer

on salesman.salesman\_id=customer.salesman\_id

left join orders

on customer.customer\_id = orders.customer\_id

order by salesman.name



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 salesman.name as salesman, cust\_name as customer, customer.city as [customer city], grade, ord\_no, ord\_date, purch\_amt

from salesman left join customer

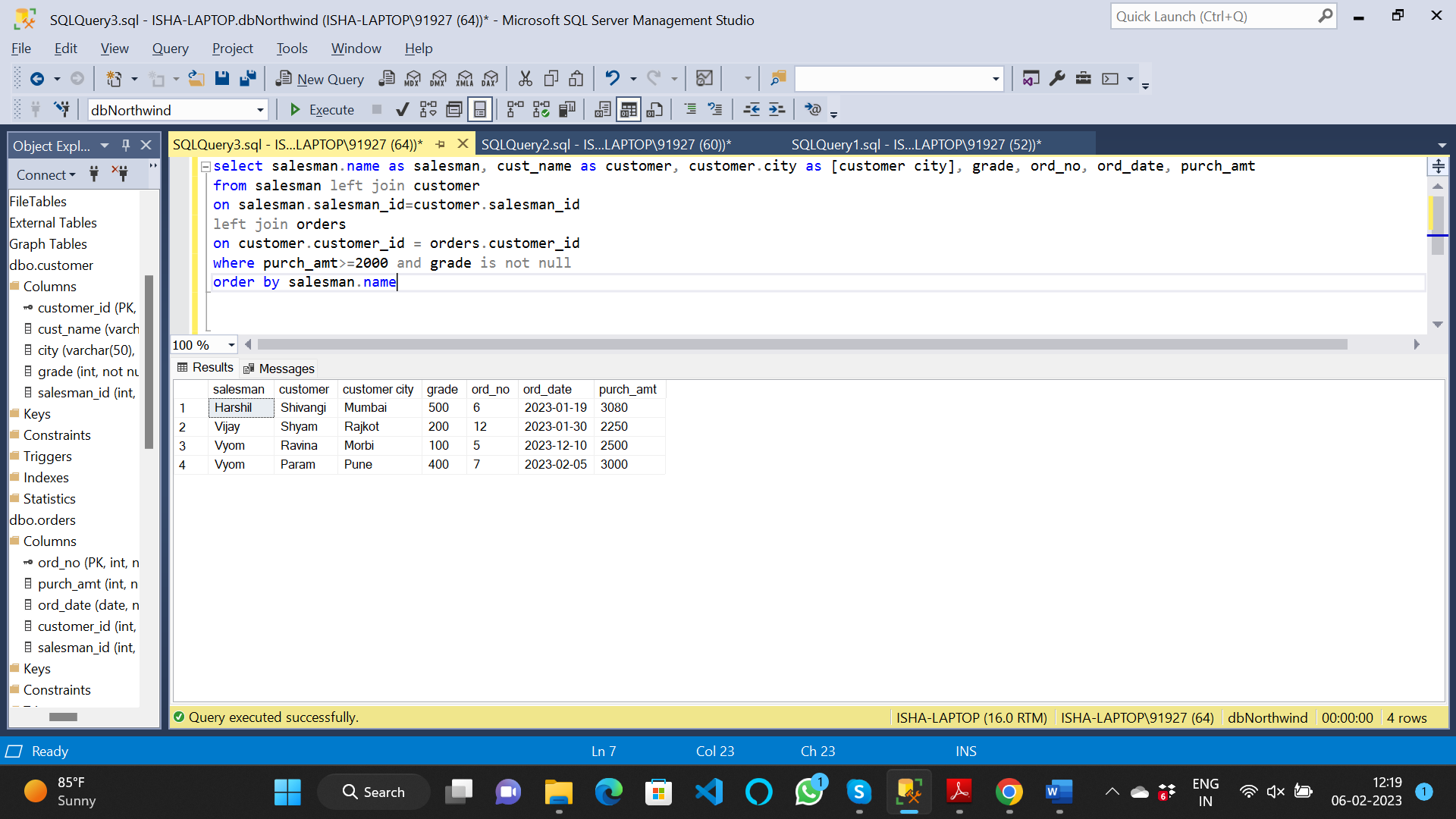
on salesman.salesman\_id=customer.salesman\_id

left join orders

on customer.customer\_id = orders.customer\_id

where purch\_amt>=2000 and grade is not null

order by salesman.name



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 salesman.name as salesman, cust\_name as customer, customer.city as [customer city], grade, ord\_no, ord\_date, purch\_amt

from salesman left join customer

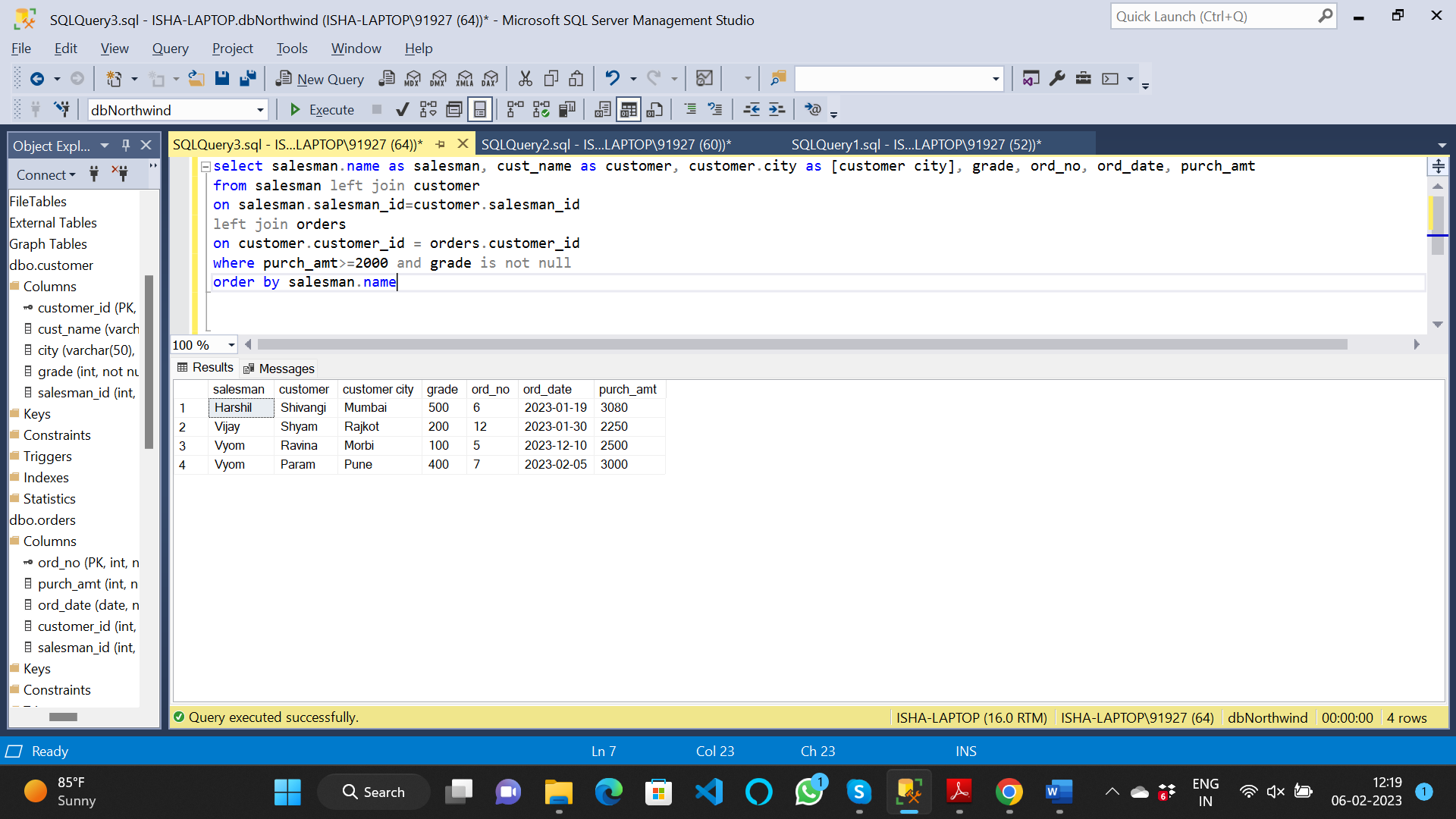
on salesman.salesman\_id=customer.salesman\_id

left join orders

on customer.customer\_id = orders.customer\_id

where purch\_amt>=2000 and grade is not null

order by salesman.name



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

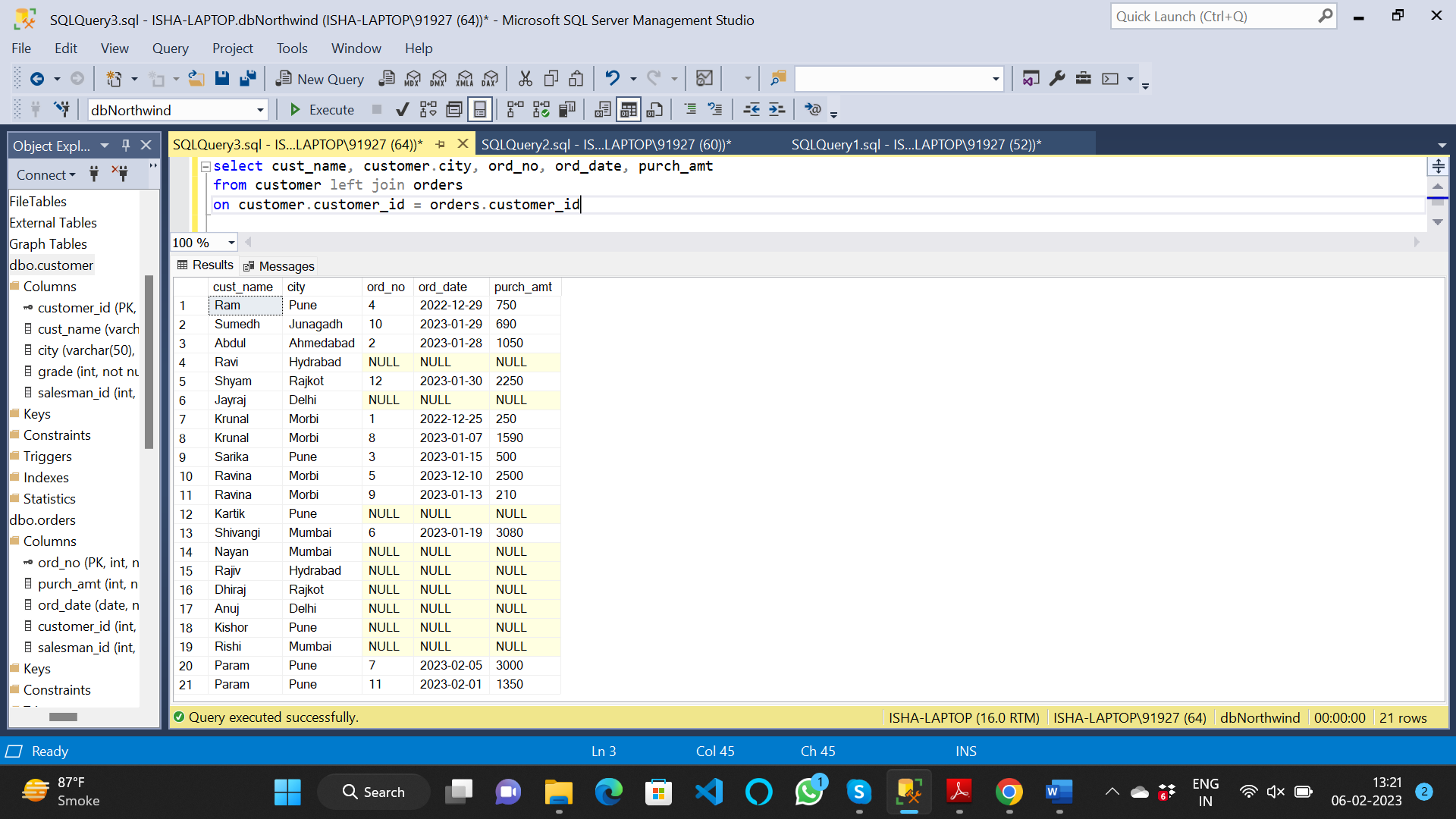
grade and placed one or more orders or which order(s) have been placed by the

customer who neither is on the list nor has a grade.

select cust\_name, customer.city, ord\_no, ord\_date, purch\_amt

from customer left join orders

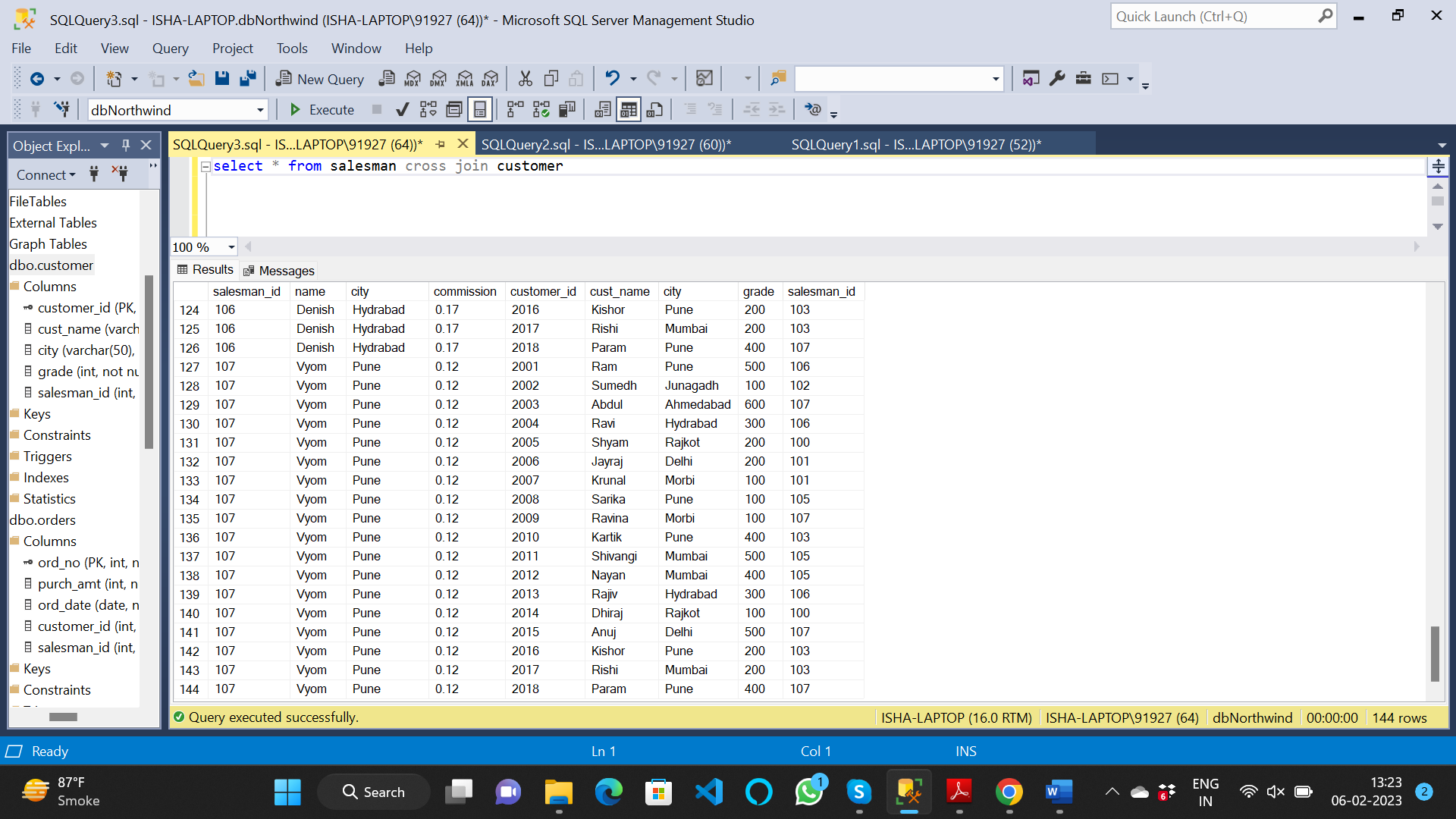
on customer.customer\_id = orders.customer\_id



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

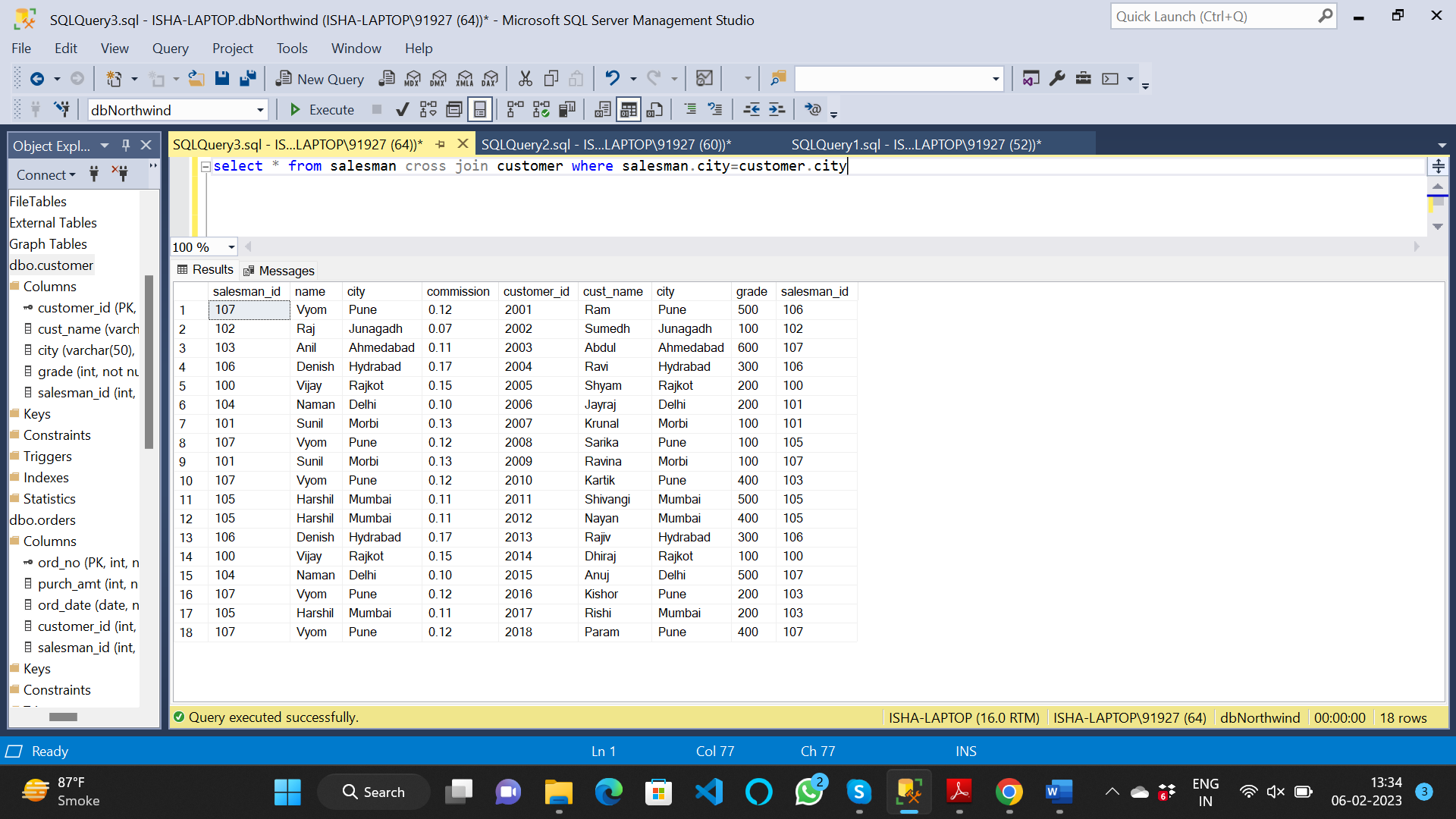


18. Write a SQL statement to create a Cartesian product between salesperson and

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=customer.city

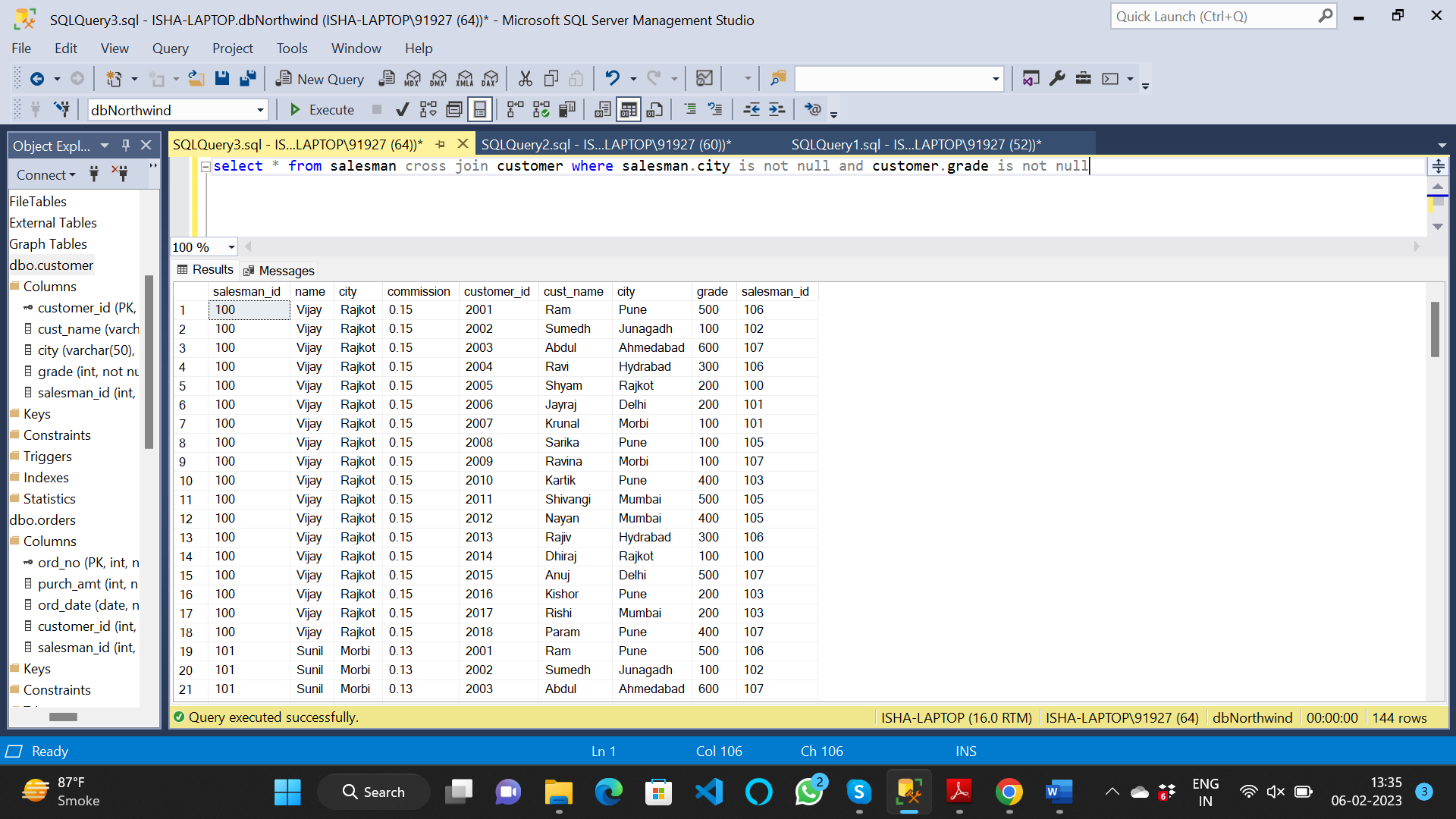


19. Write a SQL statement to create a Cartesian product between salesperson and

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

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<>customer.city) and customer.grade is not null

