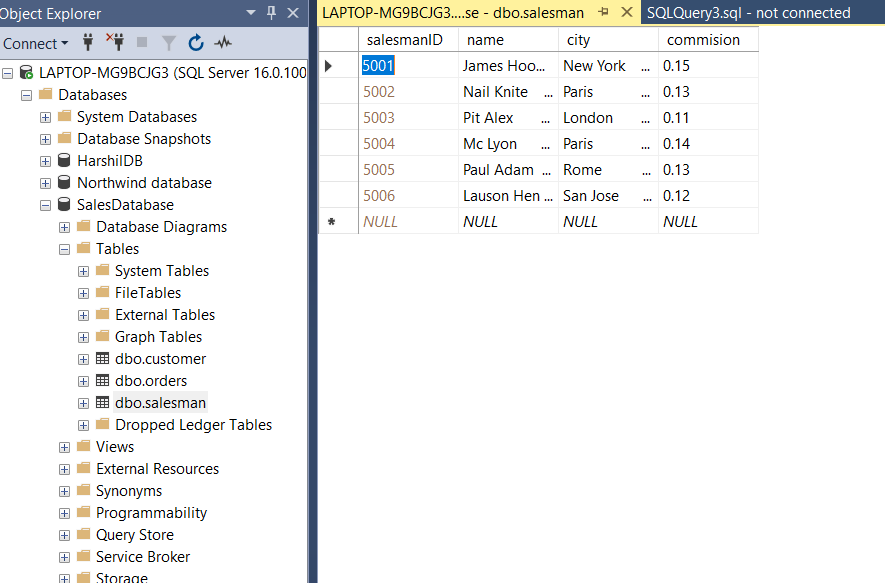
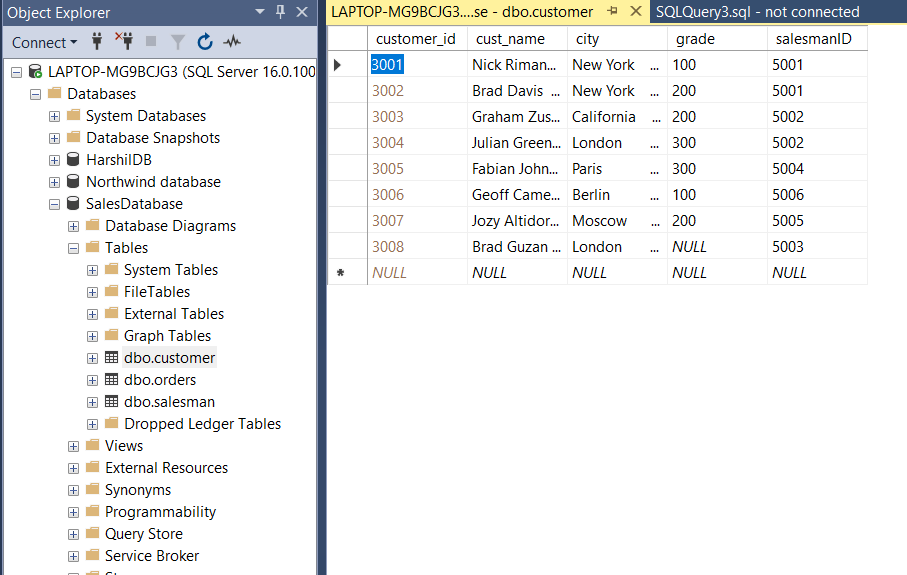
SQL ASSIGNMENT – 2

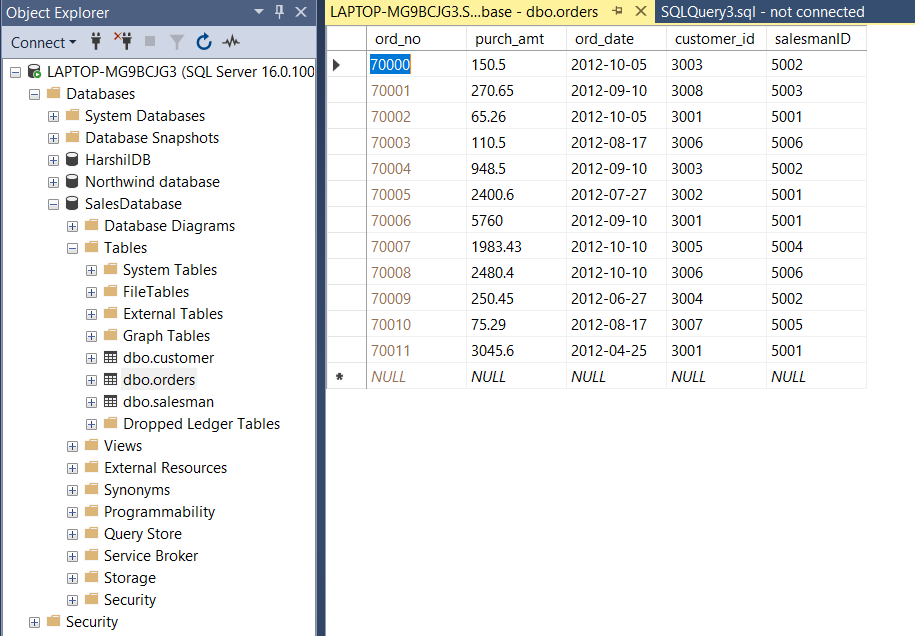
Salesman Table :



Customer table :



Orders table :



Creating and inserting in salesman table :

create table salesman(

salesmanID int PRIMARY KEY identity(5001,1),

name nchar(50) not null,

city nchar(50) not null,

commision float not null

)

insert into salesman values

('James Hoog','New York',0.15),

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

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

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

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

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

Creating and inserting in Customer table :

CREATE TABLE [customer](

customer\_id int PRIMARY KEY identity(3001,1),

cust\_name nchar(50) NOT NULL,

city nchar(50) NOT NULL,

grade int ,

salesmanID int NOT NULL

)

INSERT INTO customer values

('Nick Rimando','New York',100,5001),('Brad Davis','New York',200,5001),('Graham Zusi','California',200,5002),('Julian Green','London',300,5002),('Fabian Johnson','Paris',300,5004),('Geoff Cameron','Berlin',100,5006),('Jozy Altidor','Moscow',200,5005),('Brad Guzan','London',NULL,5003)

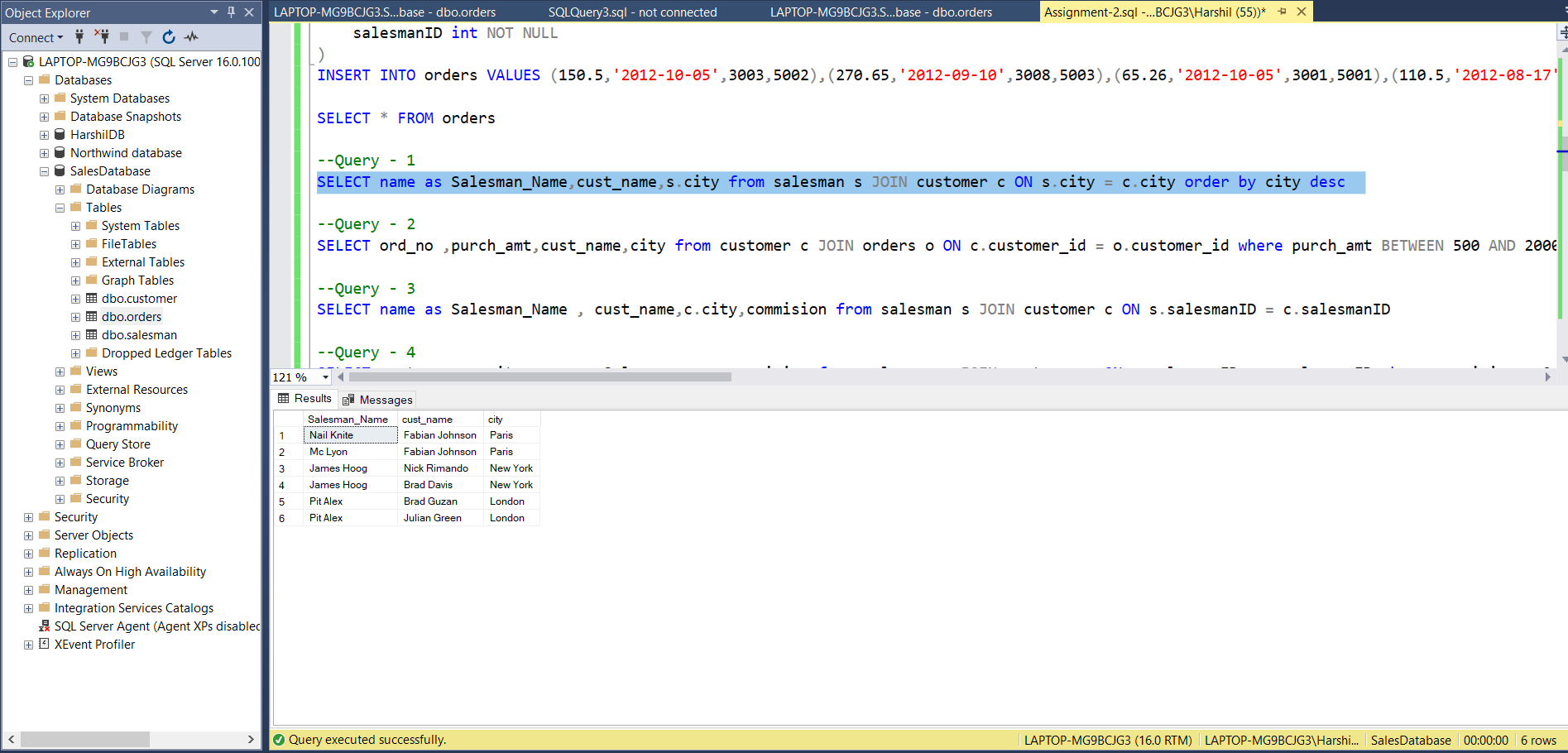
Creating and inserting in Orders table :

INSERT INTO orders VALUES

(150.5,'2012-10-05',3003,5002),(270.65,'2012-09-10',3008,5003),(65.26,'2012-10-05',3001,5001),(110.5,'2012-08-17',3006,5006),(948.5,'2012-09-10',3003,5002),(2400.6,'2012-07-27',3002,5001),(5760,'2012-09-10',3001,5001),(1983.43,'2012-10-10',3005,5004),(2480.4,'2012-10-10',3006,5006),(250.45,'2012-06-27',3004,5002),(75.29,'2012-08-17',3007,5005),(3045.6,'2012-04-25',3001,5001)

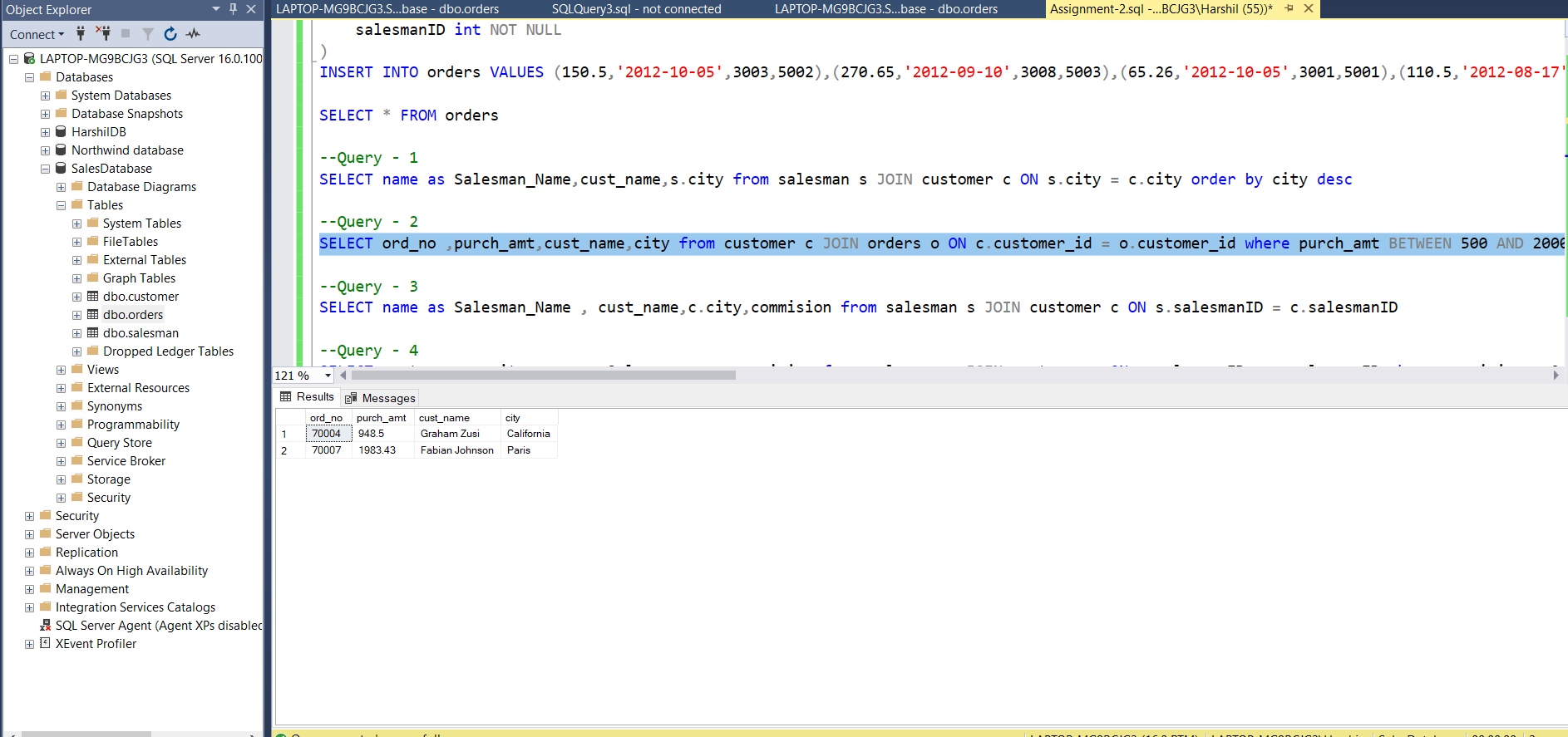
**Q – 1 : write a SQL query to find the salesperson and customer who reside in the same city. Return Salesman, cust\_name and city**

**Query :** SELECT name as Salesman\_Name,cust\_name,s.city from salesman s JOIN customer c ON s.city = c.city order by city desc

****

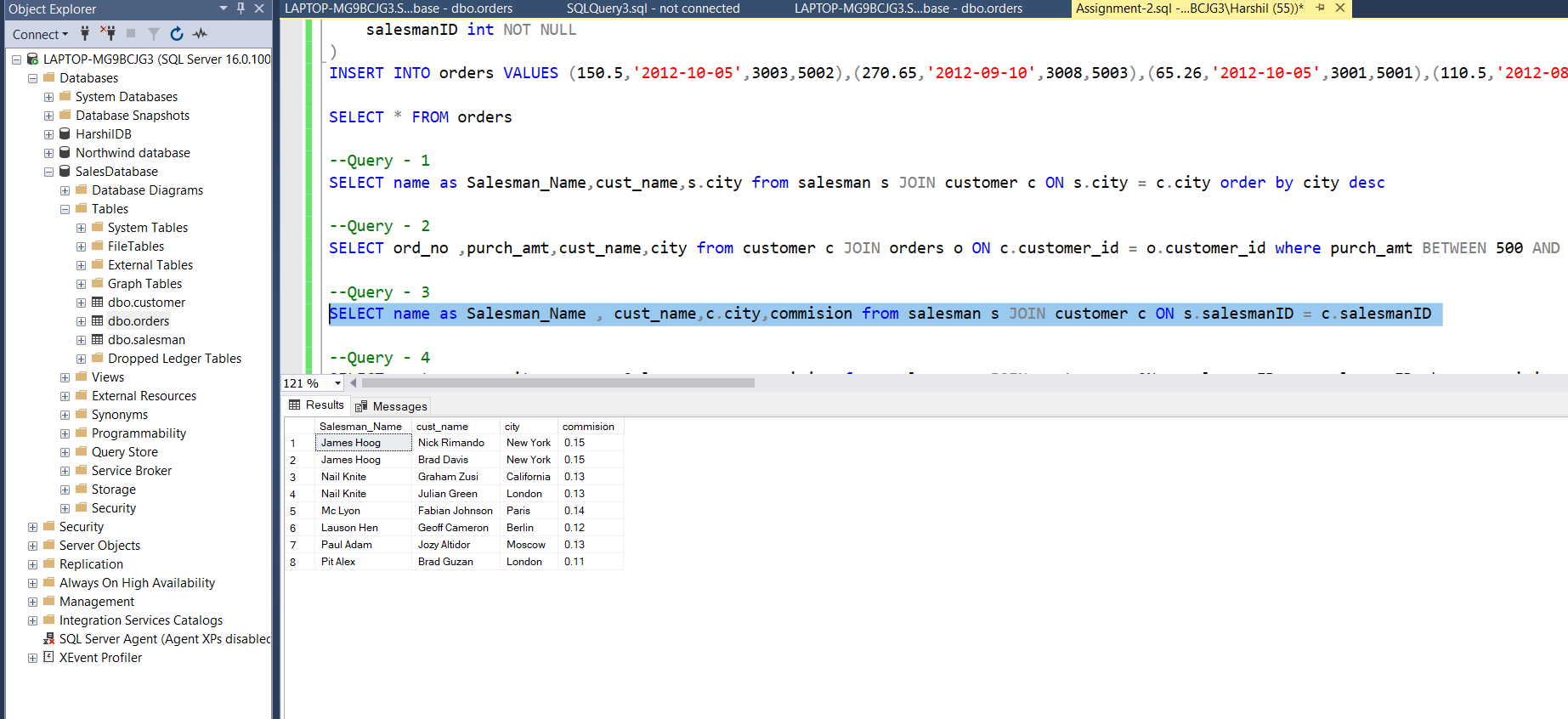
**Q – 2 : write a SQL query to find those orders where the order amount exists between 500 and 2000. Return ord\_no, purch\_amt, cust\_name, city.**

**Query :** SELECT ord\_no ,purch\_amt,cust\_name,city from customer c JOIN orders o ON c.customer\_id = o.customer\_id where purch\_amt BETWEEN 500 AND 2000

****

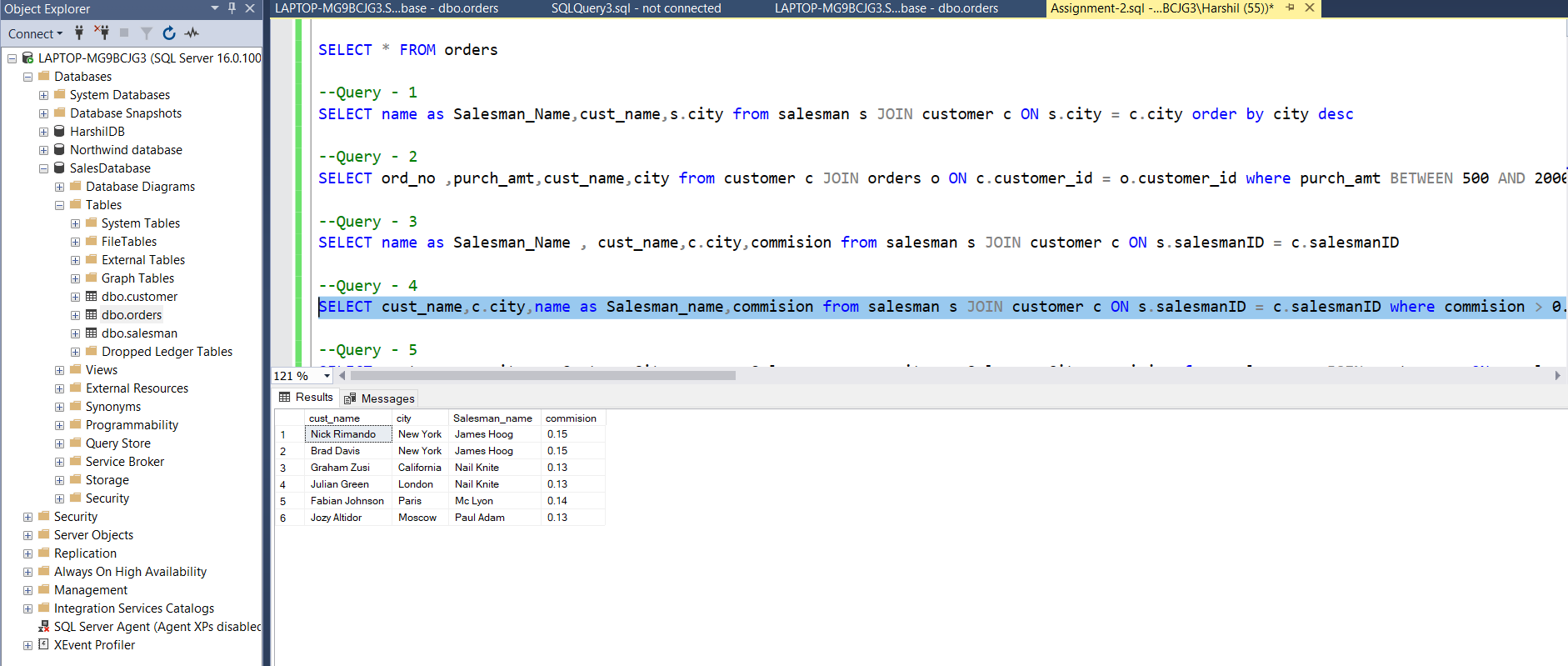
**Q – 3 : write a SQL query to find the salesperson(s) and the customer(s) he represents. Return Customer Name, city, Salesman, commission.**

**Query :** SELECT name as Salesman\_Name , cust\_name,c.city,commision from salesman s JOIN customer c ON s.salesmanID = c.salesmanID

****

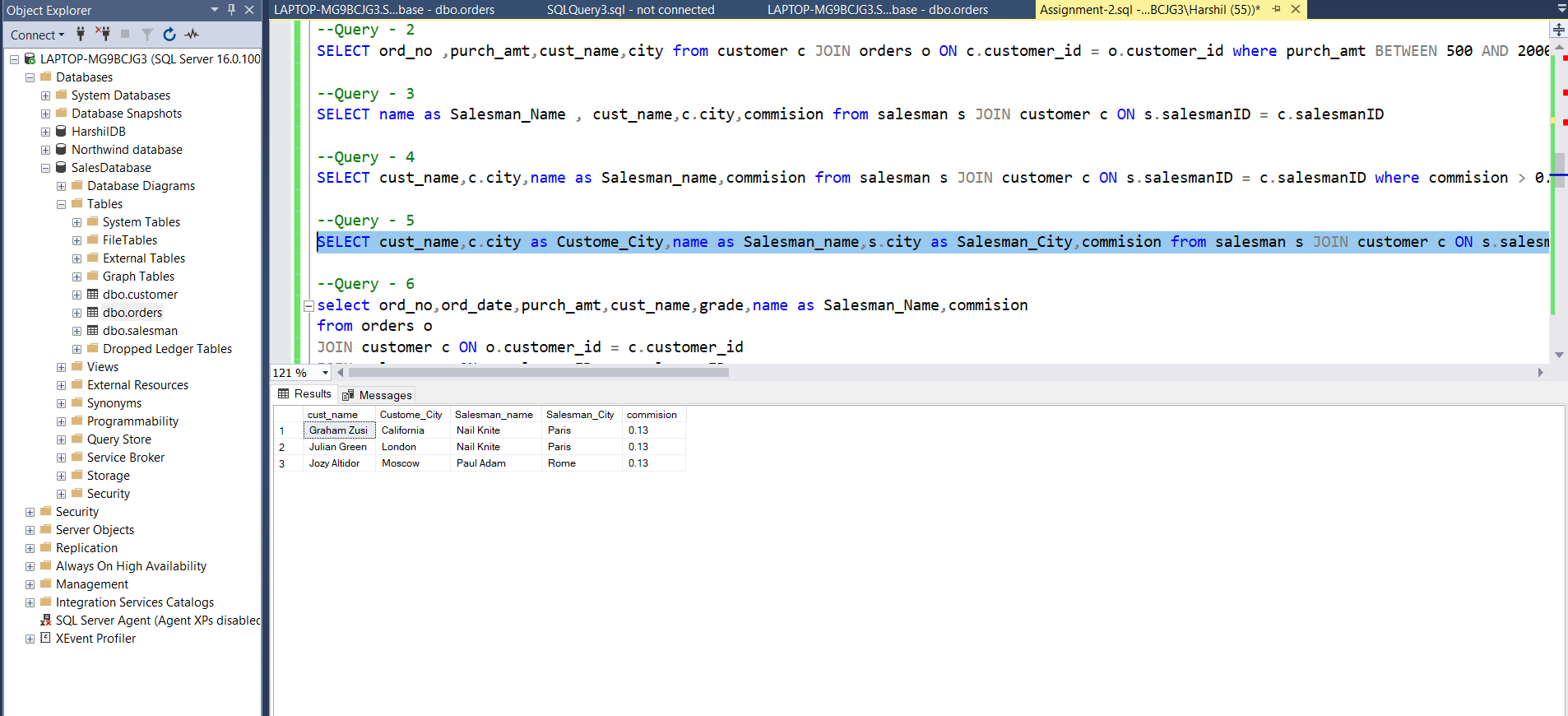
**Q – 4 : write a SQL query to find salespeople who received commissions of more than 12 percent from the company. Return Customer Name, customer city, Salesman,commission.**

**Query :** SELECT cust\_name,c.city,name as Salesman\_name,commision from salesman s JOIN customer c ON s.salesmanID = c.salesmanID where commision > 0.12

****

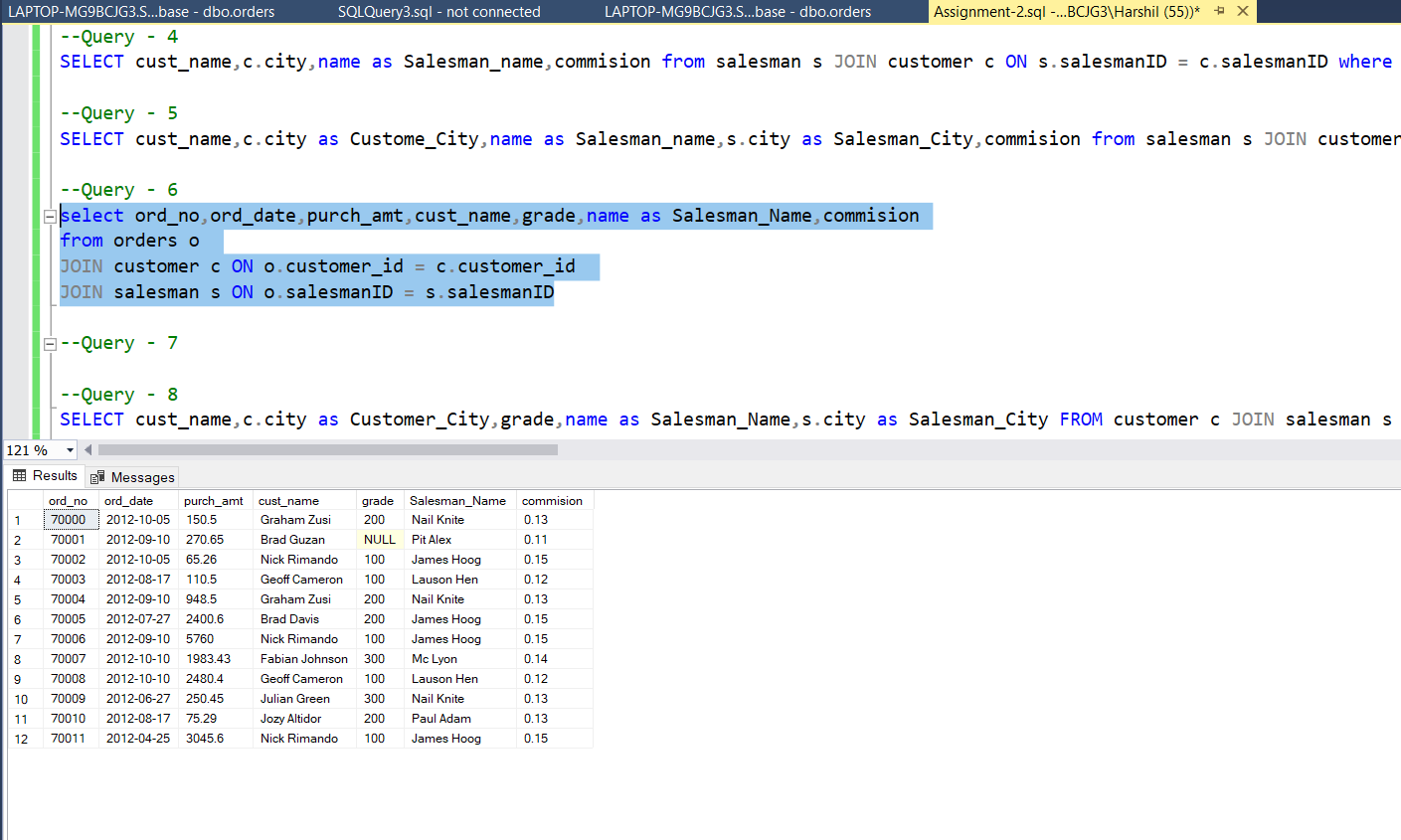
**Q – 5 :** **write a SQL query to locate those salespeople who do not live in the same city where their customers live and have received a commission of more than 12% from the company. Return Customer Name, customer city, Salesman, salesman city,commission.**

**Query :** SELECT cust\_name,c.city as Custome\_City,name as Salesman\_name,s.city as Salesman\_City,commision from salesman s JOIN customer c ON s.salesmanID = c.salesmanID where commision > 0.12 AND s.city != c.city

****

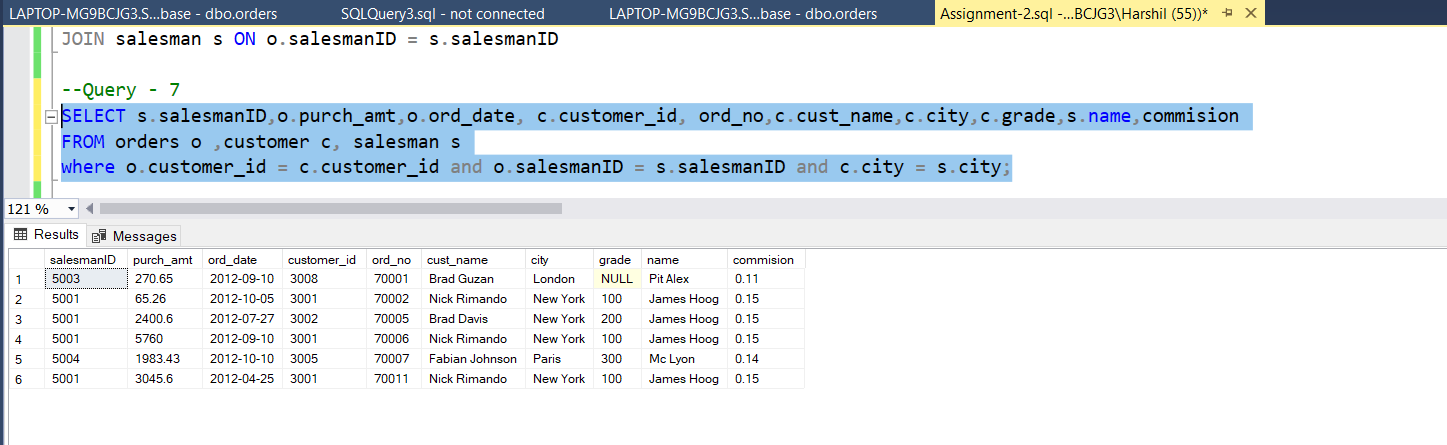
**Q – 6 : write a SQL query to find the details of an order. Return ord\_no, ord\_date, purch\_amt, Customer Name, grade, Salesman, commission.**

**Query :** select ord\_no,ord\_date,purch\_amt,cust\_name,grade,name as Salesman\_Name,commission from orders o JOIN customer c ON o.customer\_id = c.customer\_id JOIN salesman s ON o.salesmanID = s.salesmanID



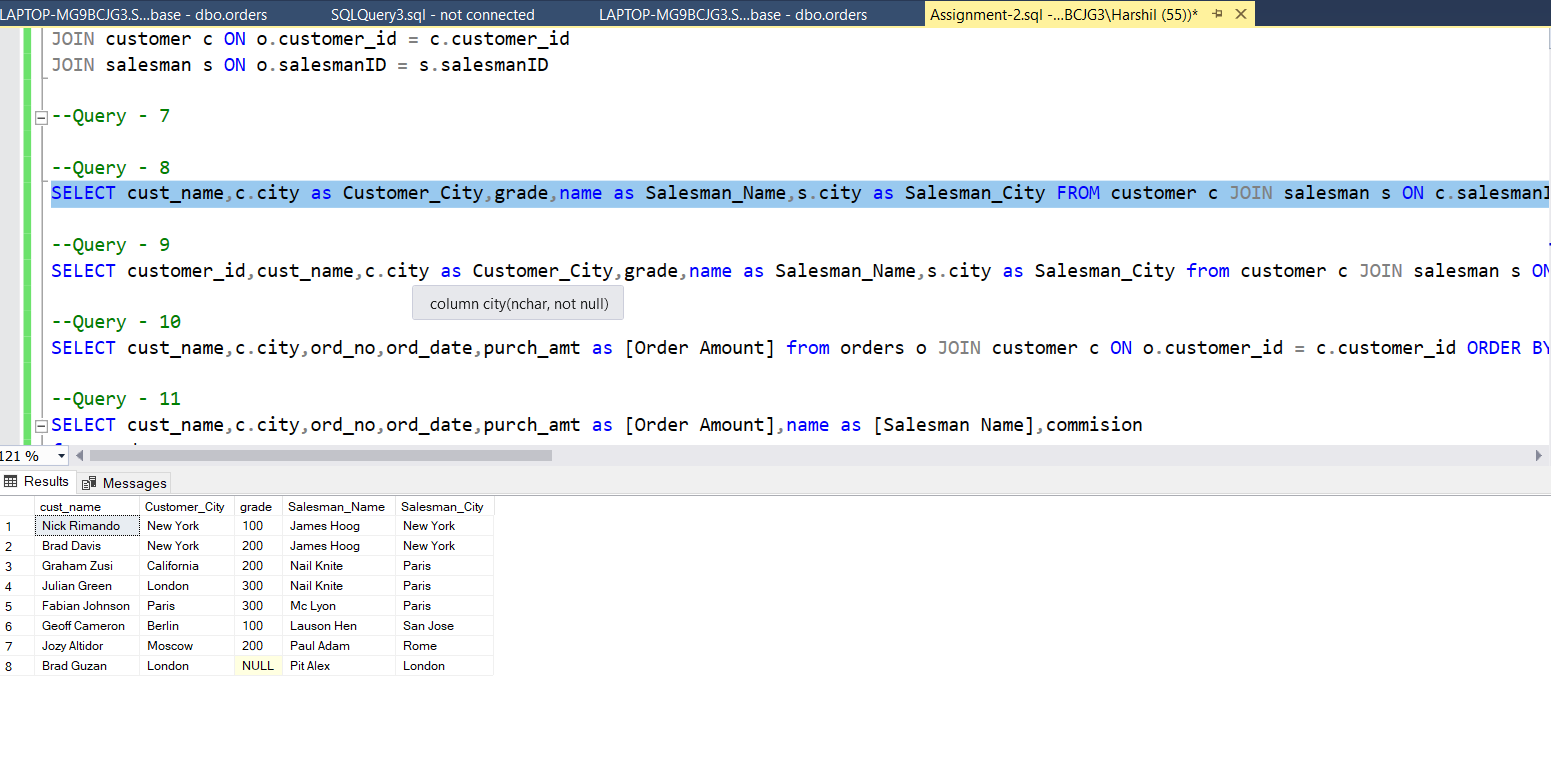
**Q – 7 : Write a SQL statement to join the tables salesman, customer and orders so that the same column of each table appears once and only the relational rows are returned.**

**Query :** SELECT s.salesmanID,o.purch\_amt,o.ord\_date, c.customer\_id, ord\_no,c.cust\_name,c.city,c.grade,s.name,commission FROM orders o ,customer c, salesman s where o.customer\_id = c.customer\_id and o.salesmanID = s.salesmanID and c.city = s.city;



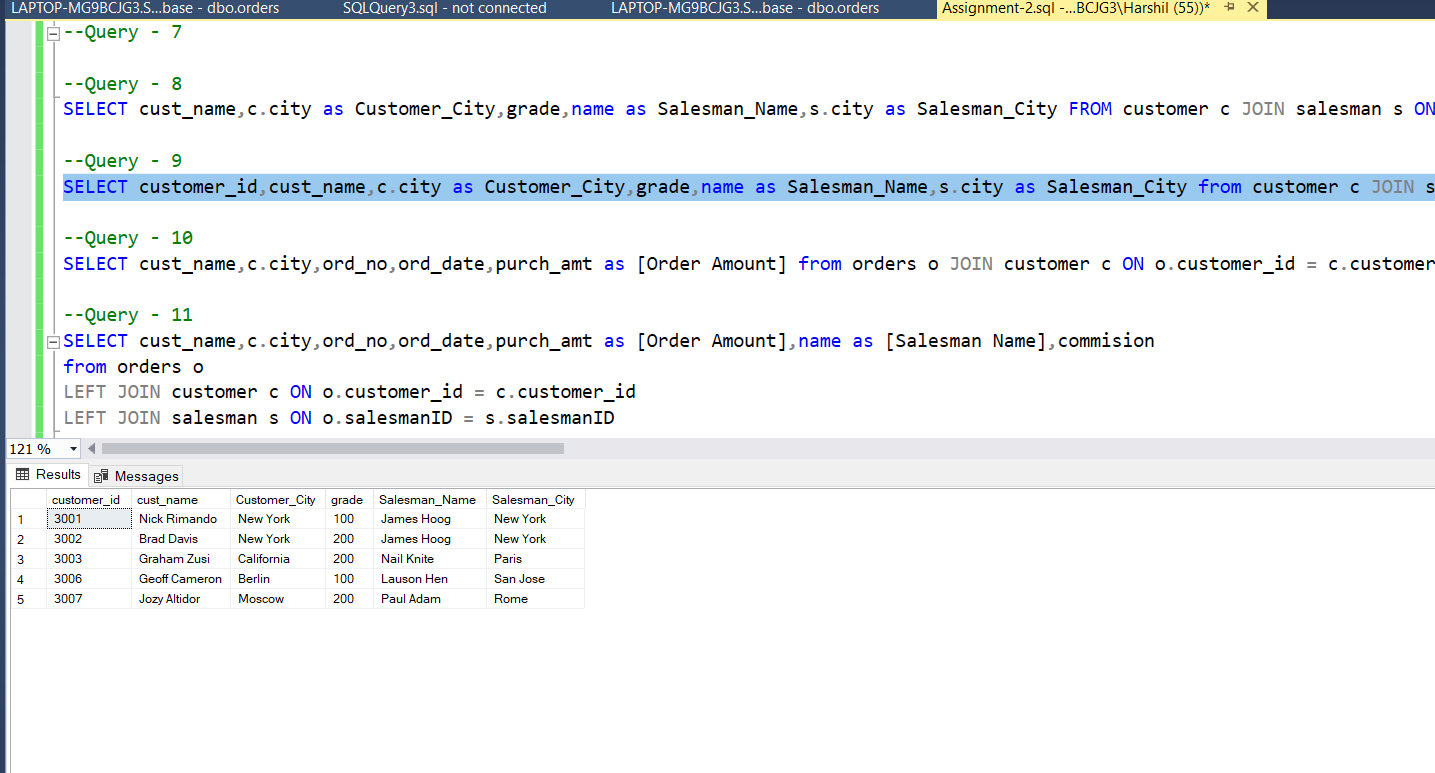
**Q – 8 :** **write a SQL query to display the customer name, customer city, grade, salesman, salesman city. The results should be sorted by ascending customer\_id.**

**Query :** SELECT cust\_name,c.city as Customer\_City,grade,name as Salesman\_Name,s.city as Salesman\_City FROM customer c JOIN salesman s ON c.salesmanID = s.salesmanID ORDER BY customer\_id asc



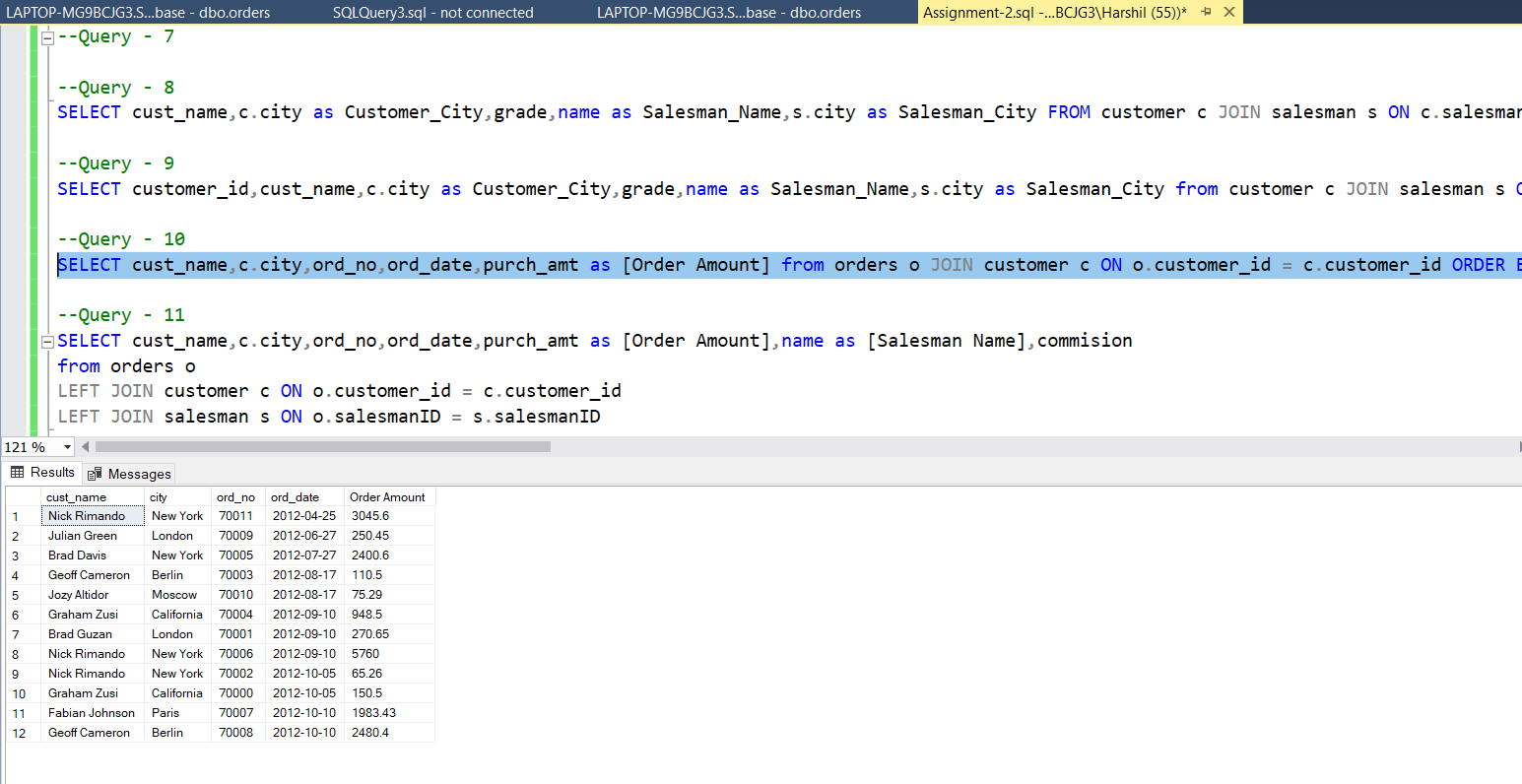
**Q – 9: write a SQL query to find those customers with a grade less than 300. Return cust\_name, customer city, grade, Salesman, salesmancity. The result should be ordered by ascending customer\_id.**

**Query :** SELECT customer\_id,cust\_name,c.city as Customer\_City,grade,name as Salesman\_Name,s.city as Salesman\_City from customer c JOIN salesman s ON c.salesmanID = s.salesmanID where grade < 300 ORDER BY customer\_id asc



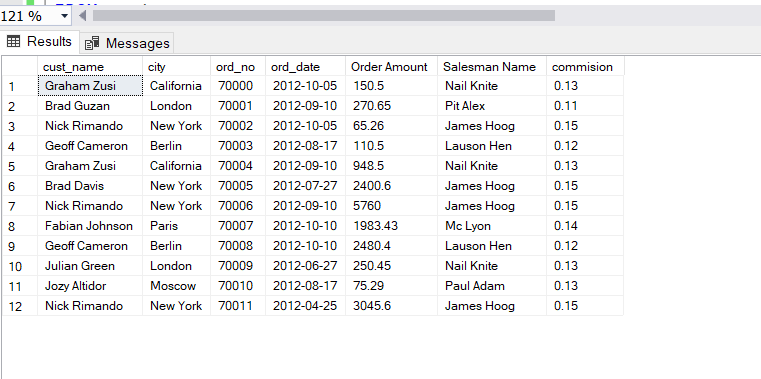
**Q – 10: Write a SQL statement to make a report with customer name, city, order number,order date, and order amount in ascending order according to the order date to determine whether any of the existing customers have placed an order or not.**

**Query :** SELECT cust\_name,c.city,ord\_no,ord\_date,purch\_amt as [Order Amount] from orders o JOIN customer c ON o.customer\_id = c.customer\_id ORDER BY ord\_date asc

****

**Q-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.**

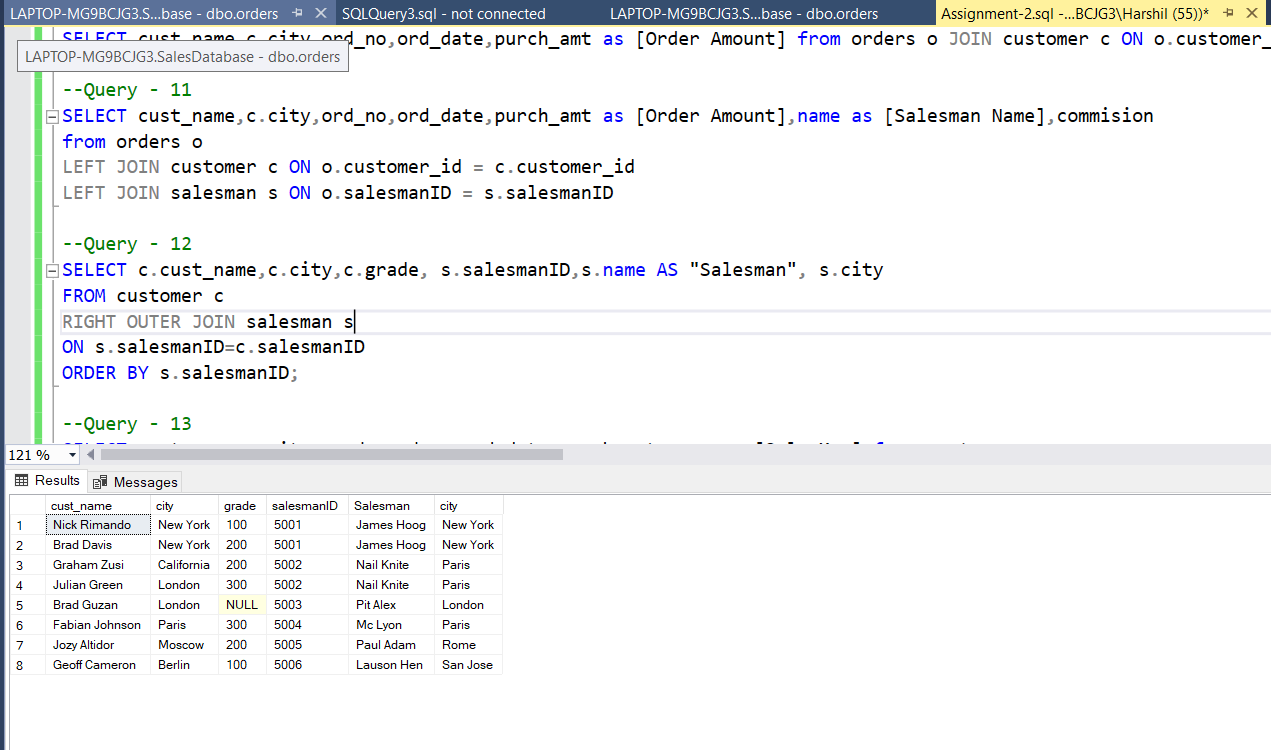
**Query :** SELECT cust\_name,c.city,ord\_no,ord\_date,purch\_amt as [Order Amount],name as [Salesman Name],commission from orders o LEFT JOIN customer c ON o.customer\_id = c.customer\_id LEFT JOIN salesman s ON o.salesmanID = s.salesmanID

****

**Q-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.**

**Query :** SELECT c.cust\_name,c.city,c.grade, s.salesmanID,s.name AS "Salesman", s.city

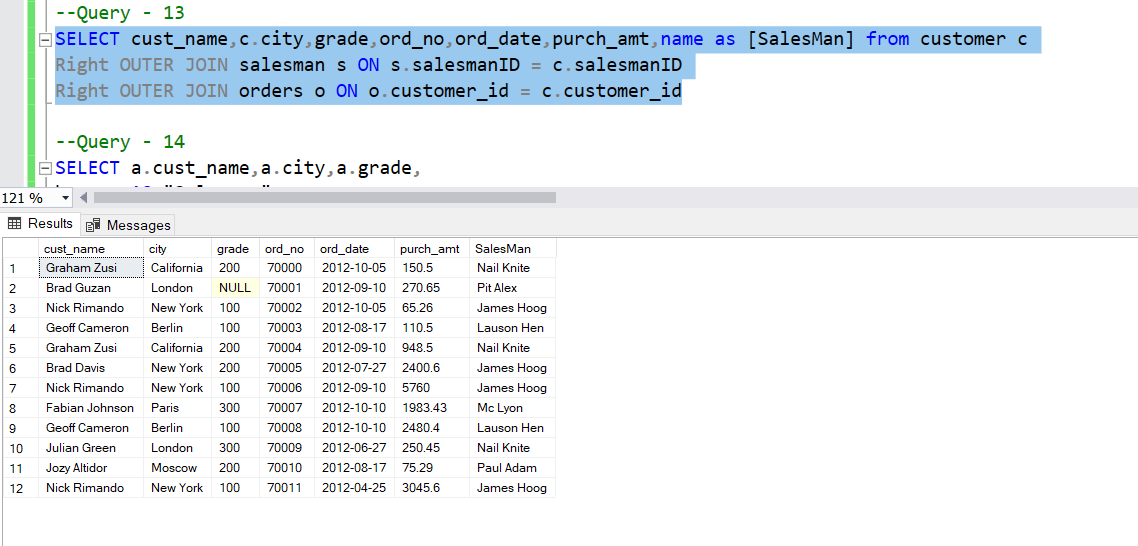
FROM customer c RIGHT OUTER JOIN salesman s ON s.salesmanID=c.salesmanID ORDER BY s.salesmanID;

****

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

**Query :** SELECT cust\_name,c.city,grade,ord\_no,ord\_date,purch\_amt,name as [SalesMan] from customer cRight OUTER JOIN salesman s ON s.salesmanID = c.salesmanID

Right OUTER JOIN orders o ON o.customer\_id = c.customer\_id

****

**Q-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.**

**Query :** 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

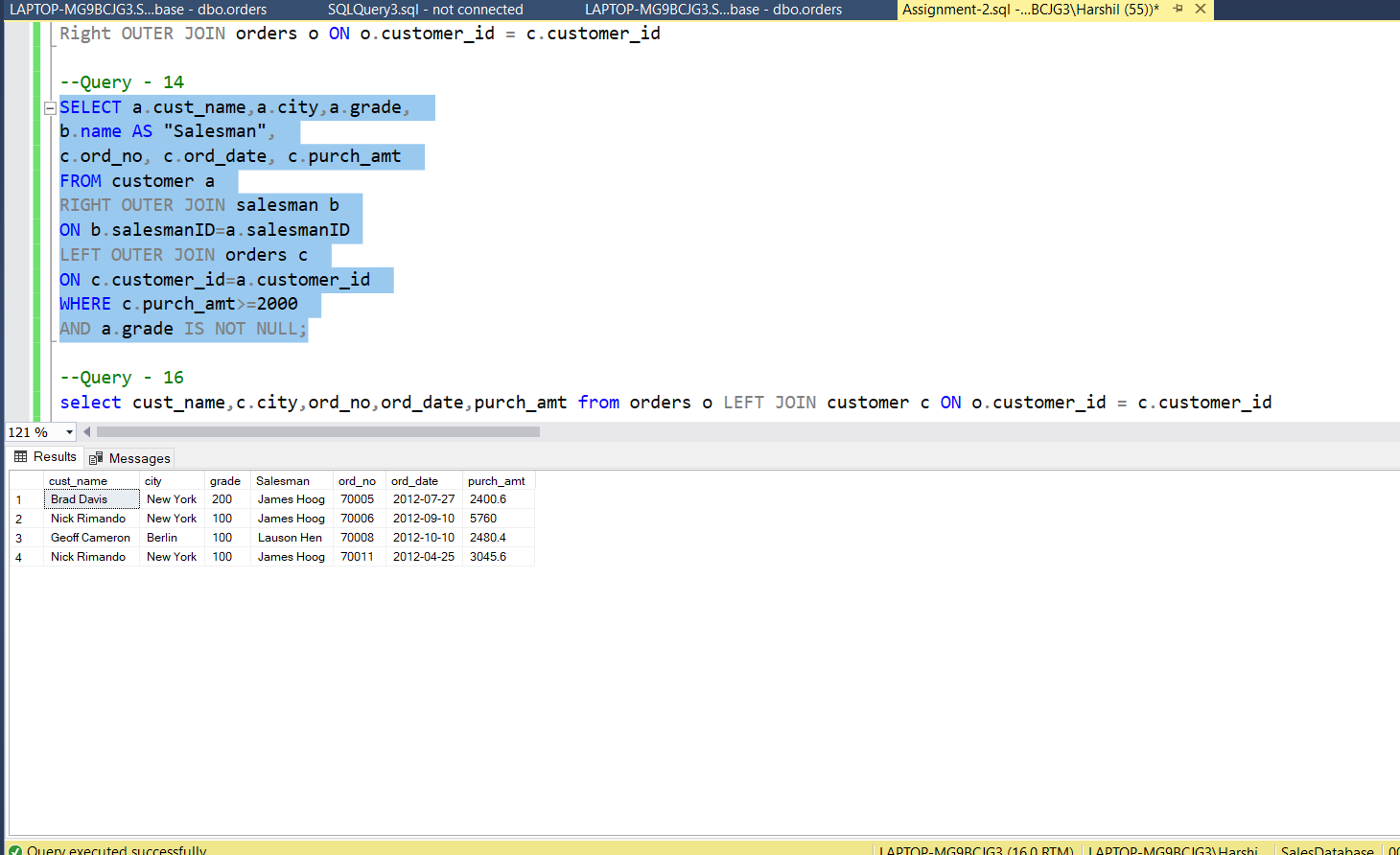
ON b.salesmanID=a.salesmanID

LEFT OUTER JOIN orders c

ON c.customer\_id=a.customer\_id

WHERE c.purch\_amt>=2000

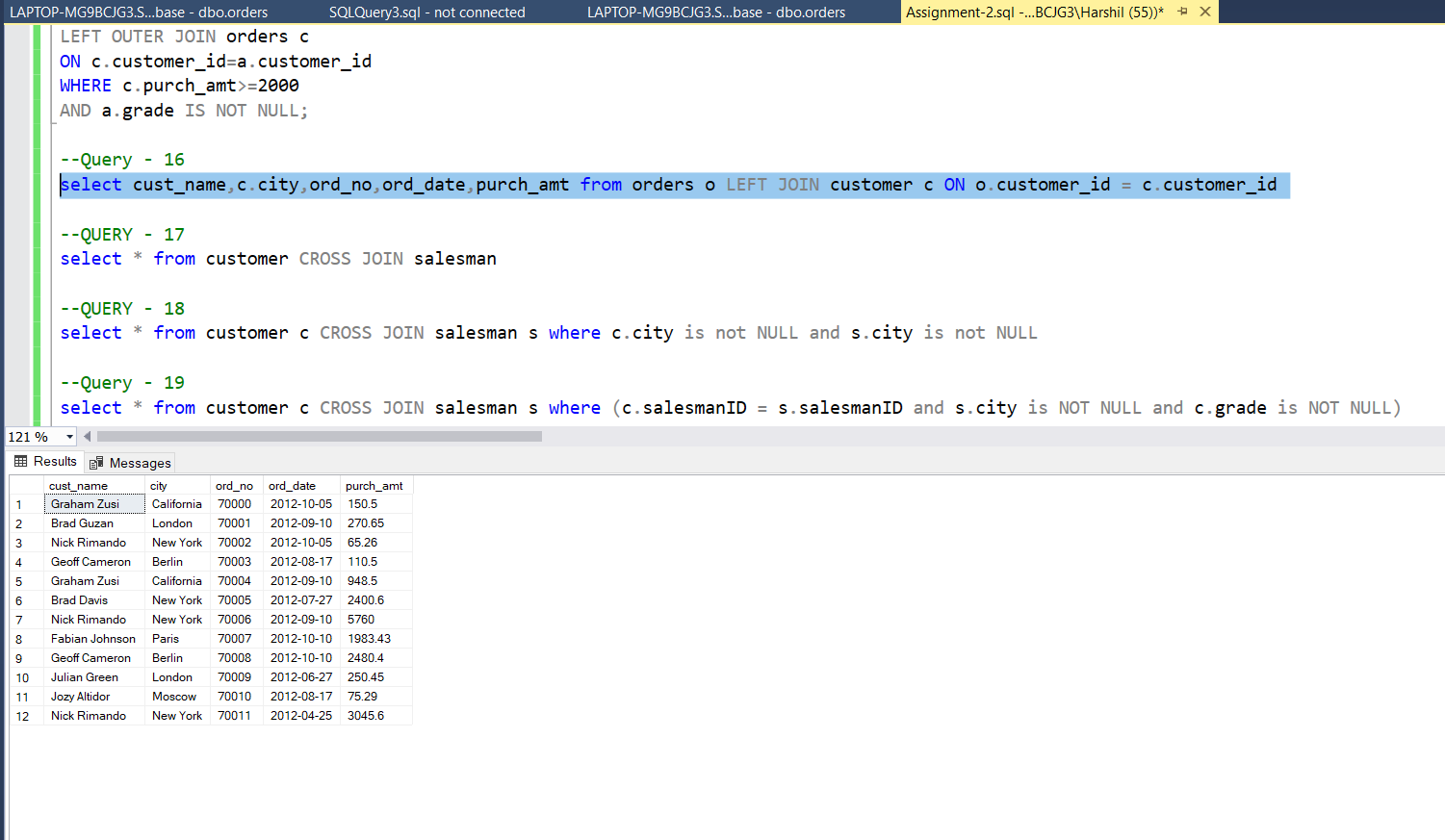
AND a.grade IS NOT NULL;

****

**Q – 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.**

**Query :**

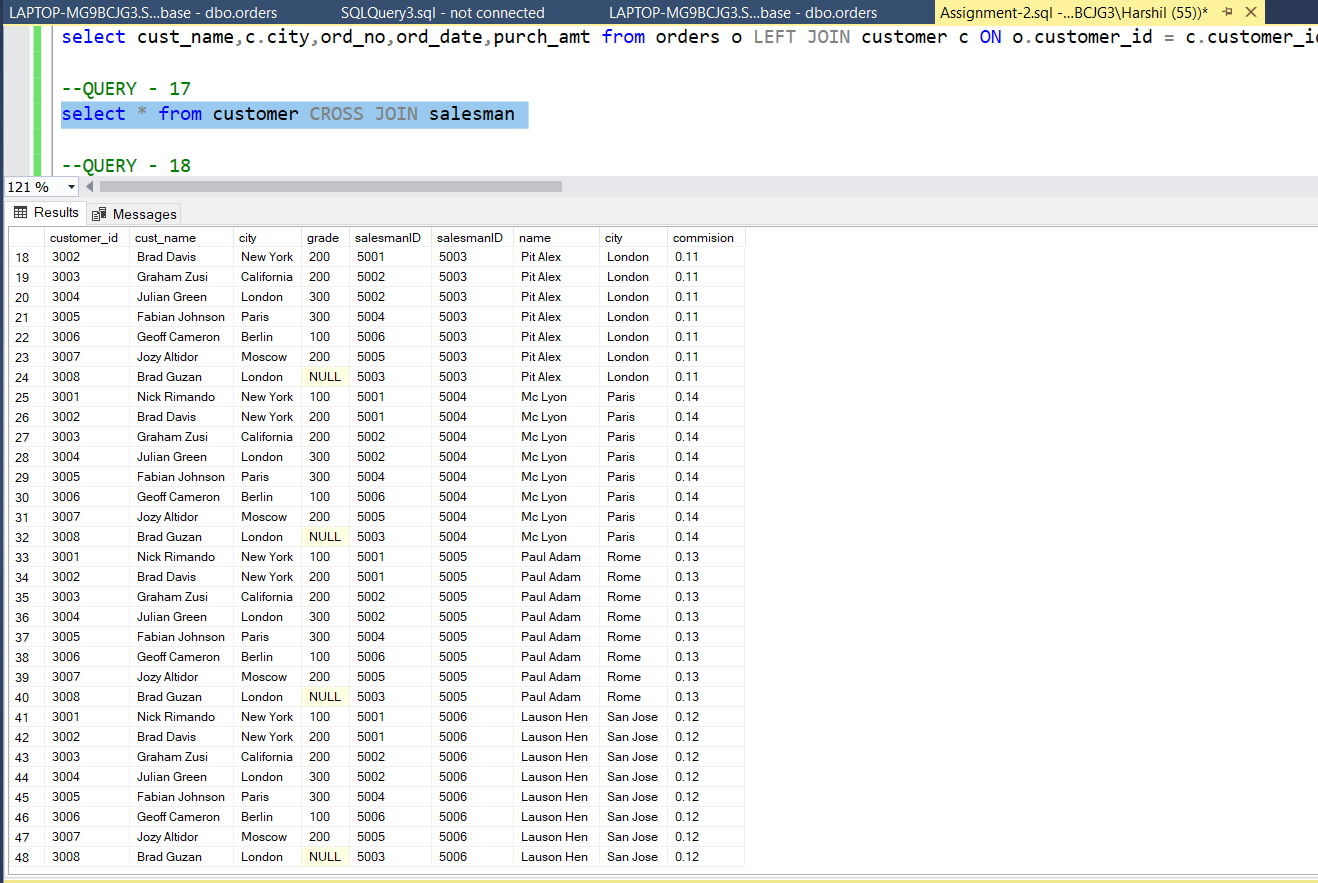
select cust\_name,c.city,ord\_no,ord\_date,purch\_amt from orders o LEFT JOIN customer c ON o.customer\_id = c.customer\_id

****

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

**Query :**

select \* from customer CROSS JOIN salesman

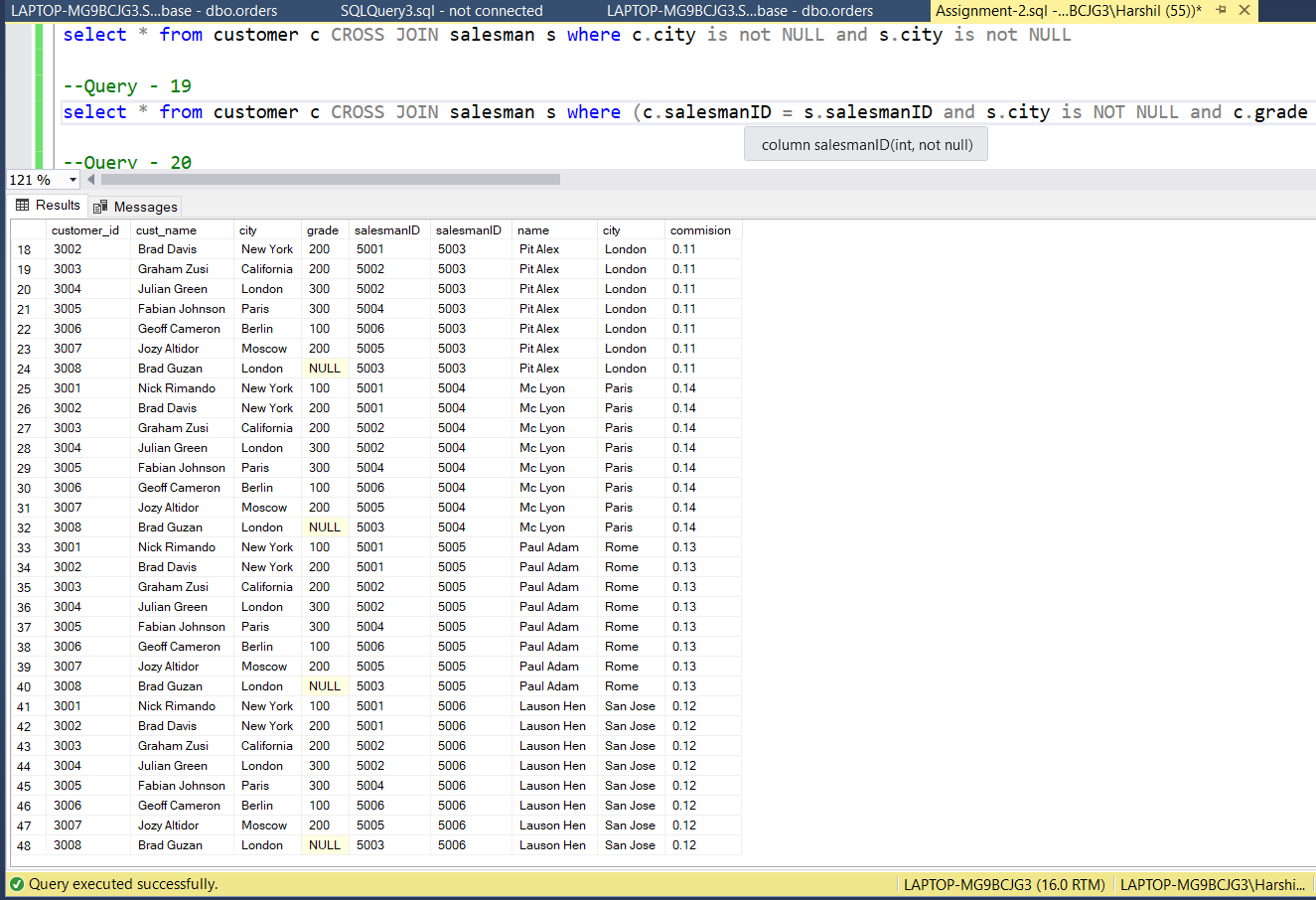
****

**Q – 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.**

**Query :**

select \* from customer c CROSS JOIN salesman s where c.city is not NULL and s.city is not NULL

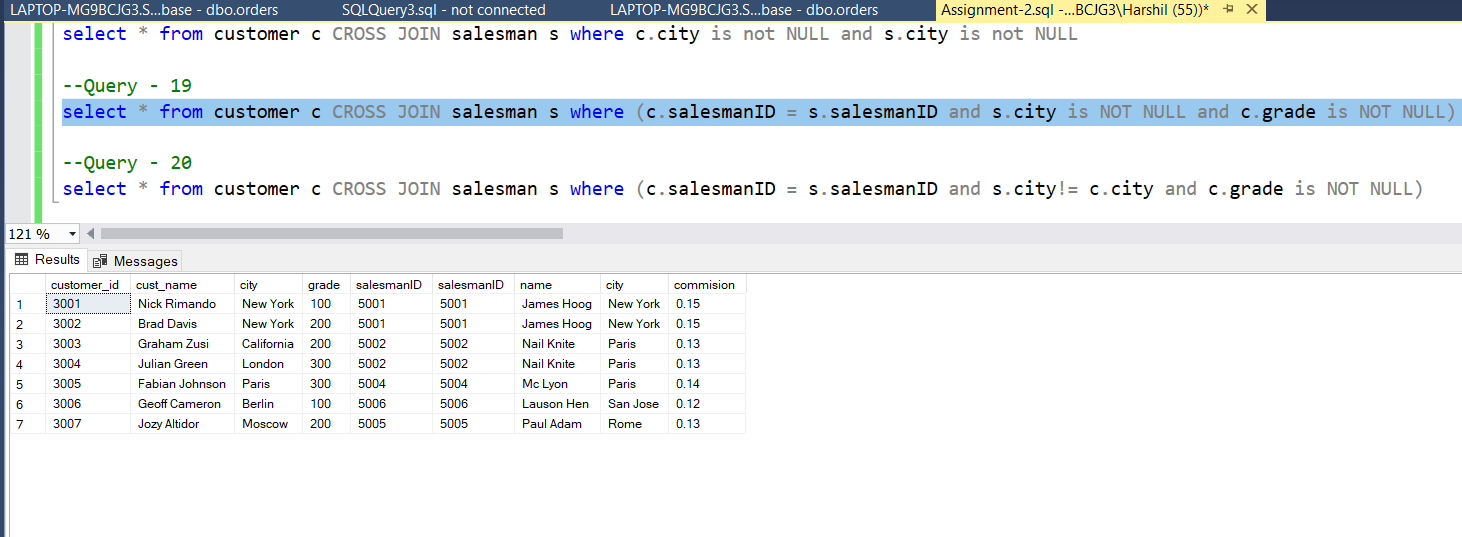
****

**Q – 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.**

**Query :**

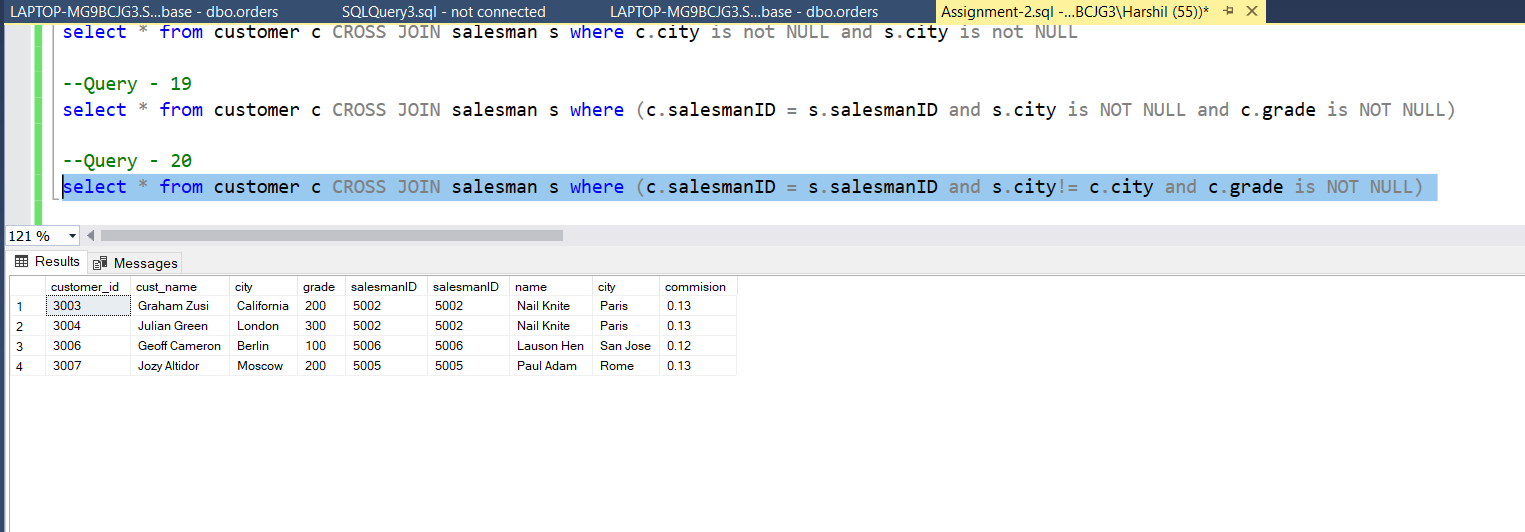
select \* from customer c CROSS JOIN salesman s where (c.salesmanID = s.salesmanID and s.city is NOT NULL and c.grade is NOT NULL)

****

**Q – 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.**

**Query :**

select \* from customer c CROSS JOIN salesman s where (c.salesmanID = s.salesmanID and s.city!= c.city and c.grade is NOT NULL)

****