---------------------------------------------------------------------

-- LAB 12

--

-- Exercise 1

---------------------------------------------------------------------

USE TSQL;

GO

---------------------------------------------------------------------

-- Task 1

--

-- Write a SELECT statement to return the productid and productname columns from the

Production.Products table.

-- Filter the results to include only products that have a categoryid value 4.

-- Execute the written statement and compare the results that you got with the desired results

-- shown in the file 52 - Lab Exercise 1 - Task 1 Result.txt. Remember the number of rows in the

result.

---------------------------------------------------------------------

---------------------------------------------------------------------

-- Detyra 1

-- Selektoni kolonat productid dhe productname nga tabela Production.Products.

-- Filtroni rezultatin per te perfshire vetem produktet qe kane categoryid me vlere 4.

-- Ekzekutoni veprimet e mesiperme dhe krahasoni rezultatet qe ju moret me rezultatet e deshiruara

-- te treguara ne dokumentin 52 - Lab Exercise 1 - Task 1 Result.txt. Remember the number of rows in the result.

---------------------------------------------------------------------

Select Distinct p.productid,p.productname,p.categoryid

From Production.Products p

CROSS APPLY Production.Categories c

Where p.categoryid = 4

--Rezultati: (10 rreshta)

| Productid | productname |categoryid

-------------------------------------------

1 11 Product QMVUN 4

2 12 Product OSFNS 4

3 31 Product XWOXC 4

4 32 Product NUNAW 4

5 33 Product ASTMN 4

6 59 Product UKXRI 4

7 60 Product WHBYK 4

8 69 Product COAXA 4

9 71 Product MYMOI 4

10 72 Product GEEOO 4

-------------------------------------------

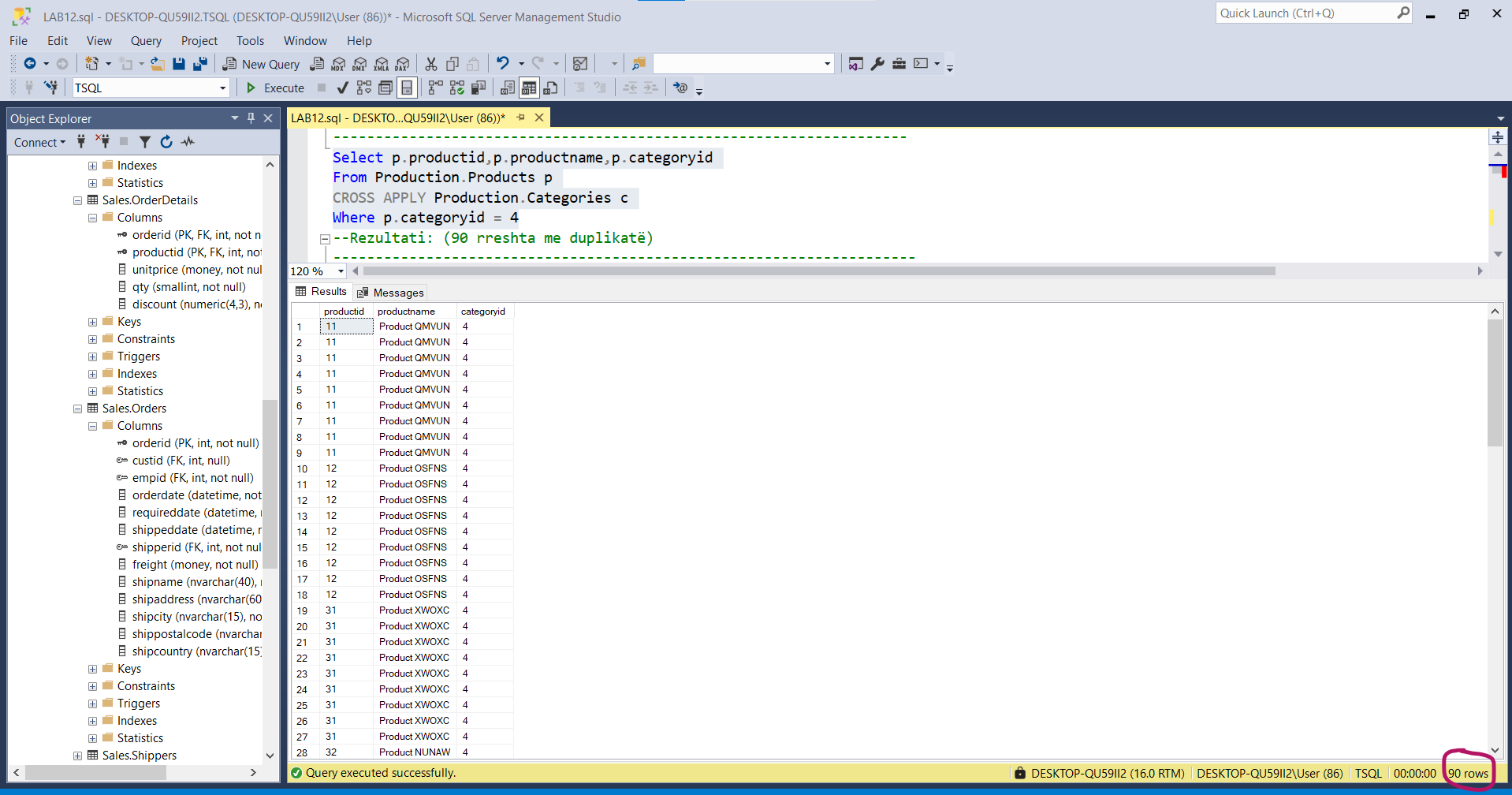
Select p.productid,p.productname,p.categoryid

From Production.Products p

CROSS APPLY Production.Categories c

Where p.categoryid = 4

--Rezultati: (90 rreshta me duplikatë) -------------------------------------



-- Task 2

--

-- Write a SELECT statement to return the productid and productname columns

-- from the Production.Products table. Filter the results to include only products

-- that have a total sales amount of more than $50,000. For the total sales amount,

-- you will need to query the Sales.OrderDetails table and aggregate

-- all order line values (qty \* unitprice) for each product.

-- Execute the written statement and compare the results that you got with the desired results

-- shown in the file 53 - Lab Exercise 1 - Task 2 Result.txt. Remember the number of rows in the result.

---------------------------------------------------------------------

---------------------------------------------------------------------

Select pp.productid ,pp.productname , SUM((od.qty\*od.unitprice)\*(1-od.discount)) as TotalSales

From Production.Products pp

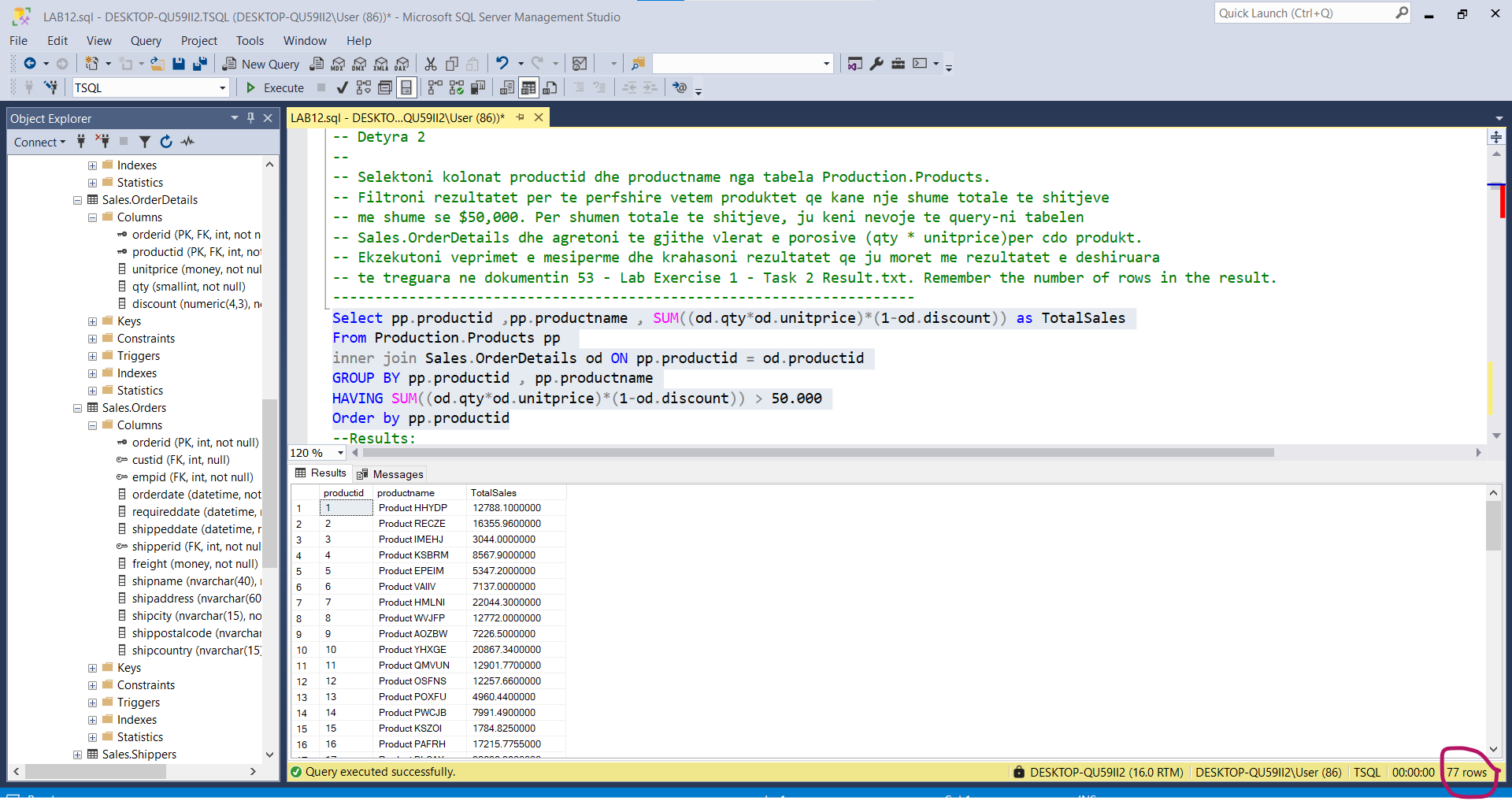
inner join Sales.OrderDetails od ON pp.productid = od.productid

GROUP BY pp.productid , pp.productname

HAVING SUM((od.qty\*od.unitprice)\*(1-od.discount)) > 50.000

Order by pp.productid

Ekzekutimi i Querisë:



--Results:

productid | productname | TotalSales

---------------------------------------------------------------

1 Product HHYDP 12788.1000000

2 Product RECZE 16355.9600000

3 Product IMEHJ 3044.0000000

4 Product KSBRM 8567.9000000

5 Product EPEIM 5347.2000000

6 Product VAIIV 7137.0000000

7 Product HMLNI 22044.3000000

8 Product WVJFP 12772.0000000

9 Product AOZBW 7226.5000000

10 Product YHXGE 20867.3400000

11 Product QMVUN 12901.7700000

12 Product OSFNS 12257.6600000

13 Product POXFU 4960.4400000

14 Product PWCJB 7991.4900000

15 Product KSZOI 1784.8250000

16 Product PAFRH 17215.7755000

17 Product BLCAX 32698.3800000

18 Product CKEDC 29171.8750000

19 Product XKXDO 5862.6200000

20 Product QHFFP 22563.3600000

21 Product VJZZH 9104.0000000

22 Product CPHFY 7122.3600000

23 Product JLUDZ 4601.7000000

24 Product QOGNU 4504.3650000

25 Product LYLNI 3704.4000000

26 Product HLGZA 19849.1445000

27 Product SMIOH 15099.8750000

28 Product OFBNT 25696.6400000

29 Product VJXYN 80368.6720000

30 Product LYERX 13424.1975000

31 Product XWOXC 14920.8750000

32 Product NUNAW 8404.1600000

33 Product ASTMN 1648.1250000

34 Product SWNJY 6350.4000000

35 Product NEVTJ 13644.0000000

36 Product GMKIJ 13458.4600000

37 Product EVFFA 2688.4000000

38 Product QDOMO 141396.7350000

39 Product LSOFL 12294.5400000

40 Product YZIXQ 17910.6300000

41 Product TTEEX 8680.3450000

42 Product RJVNM 8575.0000000

43 Product ZZZHR 23526.7000000

44 Product VJIEO 9915.9450000

45 Product AQOKR 4338.1750000

46 Product CBRRL 5883.0000000

47 Product EZZPR 3958.0800000

48 Product MYNXN 1368.7125000

49 Product FPYPN 9244.6000000

50 Product BIUDV 3437.6875000

51 Product APITJ 41819.6500000

52 Product QSRXF 3232.9500000

53 Product BKGEA 20574.1700000

54 Product QAQRL 4728.2375000

55 Product YYWRT 17426.4000000

56 Product VKCMF 42593.0600000

57 Product OVLQI 7661.5500000

58 Product ACRVI 5881.6750000

59 Product UKXRI 71155.7000000

60 Product WHBYK 46825.4800000

61 Product XYZPE 14352.6000000

62 Product WUXYK 47234.9700000

63 Product ICKNK 16701.0950000

64 Product HCQDE 21957.9675000

65 Product XYWBZ 13869.8900000

66 Product LQMGN 3383.0000000

67 Product XLXQF 2396.8000000

68 Product TBTBL 8714.0000000

69 Product COAXA 21942.3600000

70 Product TOONT 10672.6500000

71 Product MYMOI 19551.0250000

72 Product GEEOO 24900.1300000

73 Product WEUJZ 3997.2000000

74 Product BKAZJ 2432.5000000

75 Product BWRLG 8177.4900000

76 Product JYGFE 15760.4400000

77 Product LUNZZ 9171.6300000

--77 rreshta

---------------------------------------------------------------------

-- Task 3

--

--1-- Write a SELECT statement that uses the UNION operator to retrieve the productid and

-- productname columns from the T-SQL statements in task 1 and task 2.

--2-- Execute the written statement and compare the results that you got with the desired results

-- shown in the file 54 - Lab Exercise 1 - Task 3\_1 Result.txt.

--3-- What is the total number of rows in the result? If you compare this number

-- with an aggregate value of the number of rows from task 1 and task 2 is there any difference?

--4-- Copy the T-SQL statement and modify it to use the UNION ALL operator.

--5-- Execute the written statement and compare the results that you got with

-- the desired results shown in the file 55 - Lab Exercise 1 - Task 3\_2 Result.txt.

--6-- What is the total number of rows in the result?

--7-- What is the difference between the UNION and UNION ALL operators?

---------------------------------------------------------------------

---------------------------------------------------------------------

-- Detyra 3

--

--1-- Selektoni kolonat productid dhe productname duke perdorur operatorin UNION

------ nga veprimet T-SQL ne detyren 1 dhe detyren 2.

(Select pp.productid ,pp.productname , SUM((od.qty\*od.unitprice)\*(1-od.discount)) as TotalSales

From Production.Products pp

inner join Sales.OrderDetails od ON pp.productid = od.productid

GROUP BY pp.productid , pp.productname

HAVING SUM((od.qty\*od.unitprice)\*(1-od.discount)) > 50.000

)UNION

(Select p.productid,p.productname,p.categoryid

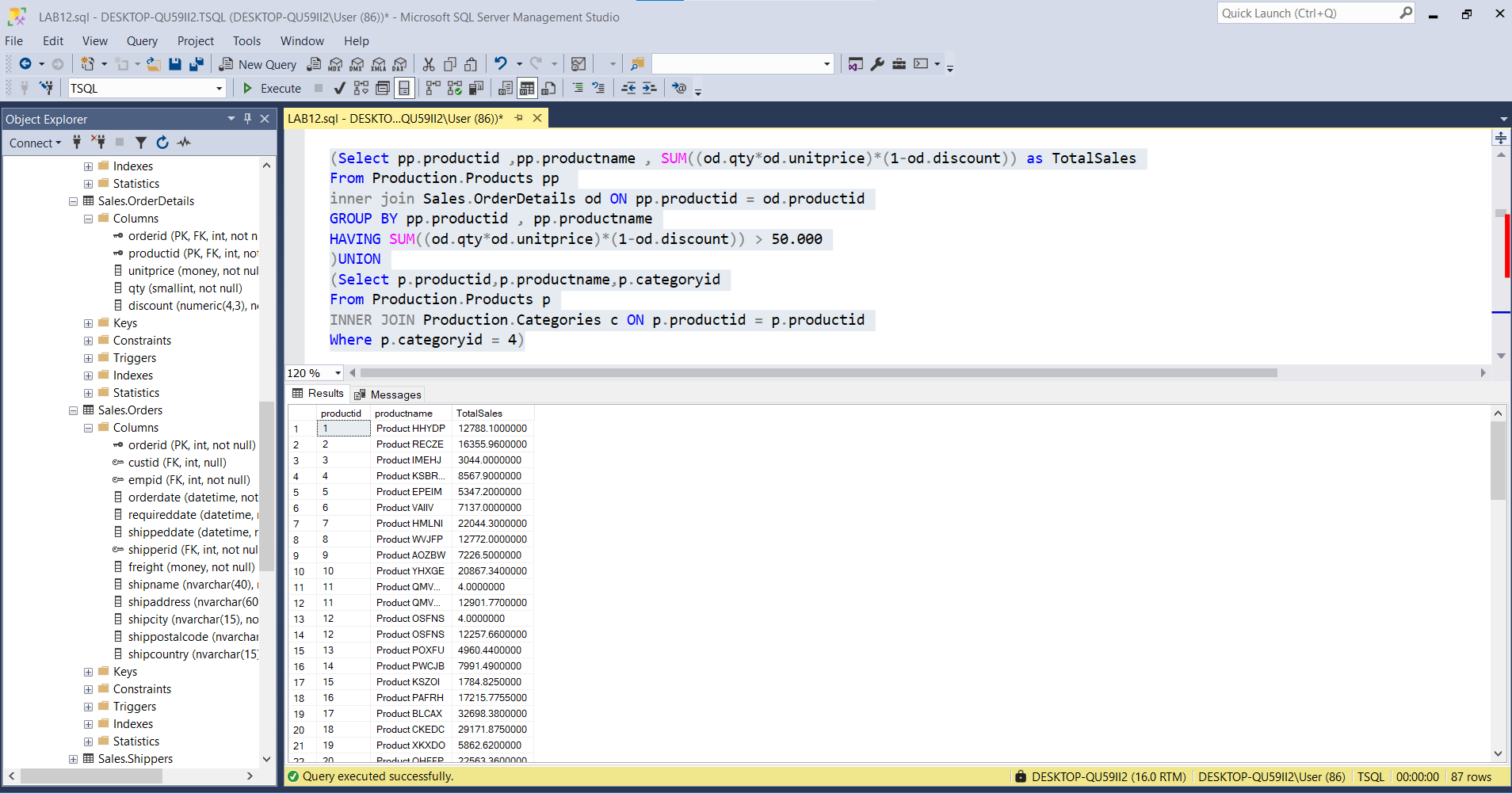
From Production.Products p

INNER JOIN Production.Categories c ON p.productid = p.productid

Where p.categoryid = 4)

--2-- Ekzekutoni veprimet e mesiperme dhe krahasoni rezultatet qe ju moret me rezultatet e deshiruara

------ te treguara ne dokumentin 54 - Lab Exercise 1 - Task 3\_1 Result.txt.



| productid |productname | TotalSales

---------------------------------------------------------------

1 1 Product HHYDP 12788.1000000

2 2 Product RECZE 16355.9600000

3 3 Product IMEHJ 3044.0000000

4 4 Product KSBRM 8567.9000000

5 5 Product EPEIM 5347.2000000

6 6 Product VAIIV 7137.0000000

7 7 Product HMLNI 22044.3000000

8 8 Product WVJFP 12772.0000000

9 9 Product AOZBW 7226.5000000

10 10 Product YHXGE 20867.3400000

11 11 Product QMVUN 4.0000000

12 11 Product QMVUN 12901.7700000

13 12 Product OSFNS 4.0000000

14 12 Product OSFNS 12257.6600000

15 13 Product POXFU 4960.4400000

16 14 Product PWCJB 7991.4900000

17 15 Product KSZOI 1784.8250000

18 16 Product PAFRH 17215.7755000

19 17 Product BLCAX 32698.3800000

20 18 Product CKEDC 29171.8750000

21 19 Product XKXDO 5862.6200000

22 20 Product QHFFP 22563.3600000

23 21 Product VJZZH 9104.0000000

24 22 Product CPHFY 7122.3600000

25 23 Product JLUDZ 4601.7000000

26 24 Product QOGNU 4504.3650000

27 25 Product LYLNI 3704.4000000

28 26 Product HLGZA 19849.1445000

29 27 Product SMIOH 15099.8750000

30 28 Product OFBNT 25696.6400000

31 29 Product VJXYN 80368.6720000

32 30 Product LYERX 13424.1975000

33 31 Product XWOXC 4.0000000

34 31 Product XWOXC 14920.8750000

35 32 Product NUNAW 4.0000000

36 32 Product NUNAW 8404.1600000

37 33 Product ASTMN 4.0000000

38 33 Product ASTMN 1648.1250000

39 34 Product SWNJY 6350.4000000

40 35 Product NEVTJ 13644.0000000

41 36 Product GMKIJ 13458.4600000

42 37 Product EVFFA 2688.4000000

43 38 Product QDOMO 141396.7350000

44 39 Product LSOFL 12294.5400000

45 40 Product YZIXQ 17910.6300000

46 41 Product TTEEX 8680.3450000

47 42 Product RJVNM 8575.0000000

48 43 Product ZZZHR 23526.7000000

49 44 Product VJIEO 9915.9450000

50 45 Product AQOKR 4338.1750000

51 46 Product CBRRL 5883.0000000

52 47 Product EZZPR 3958.0800000

53 48 Product MYNXN 1368.7125000

54 49 Product FPYPN 9244.6000000

55 50 Product BIUDV 3437.6875000

56 51 Product APITJ 41819.6500000

57 52 Product QSRXF 3232.9500000

58 53 Product BKGEA 20574.1700000

59 54 Product QAQRL 4728.2375000

60 55 Product YYWRT 17426.4000000

61 56 Product VKCMF 42593.0600000

62 57 Product OVLQI 7661.5500000

63 58 Product ACRVI 5881.6750000

64 59 Product UKXRI 4.0000000

65 59 Product UKXRI 71155.7000000

66 60 Product WHBYK 4.0000000

67 60 Product WHBYK 46825.4800000

68 61 Product XYZPE 14352.6000000

69 62 Product WUXYK 47234.9700000

70 63 Product ICKNK 16701.0950000

71 64 Product HCQDE 21957.9675000

72 65 Product XYWBZ 13869.8900000

73 66 Product LQMGN 3383.0000000

74 67 Product XLXQF 2396.8000000

75 68 Product TBTBL 8714.0000000

76 69 Product COAXA 4.0000000

77 69 Product COAXA 21942.3600000

78 70 Product TOONT 10672.6500000

79 71 Product MYMOI 4.0000000

80 71 Product MYMOI 19551.0250000

81 72 Product GEEOO 4.0000000

82 72 Product GEEOO 24900.1300000

83 73 Product WEUJZ 3997.2000000

84 74 Product BKAZJ 2432.5000000

85 75 Product BWRLG 8177.4900000

86 76 Product JYGFE 15760.4400000

87 77 Product LUNZZ 9171.6300000

--3-- Cili eshte numri total i rrjeshtave ne rezultat? Ne qoftese ju krahasoni kete numer me nje vlere agregate

-- te numrit te rrjeshtave nga detyra 1 dhe detyra 2 a ka ndonje ndryshim?

--Me funksion Agregat

(Select count(pp.productid) as NoProducts

From Production.Products pp

INNER JOIN Production.Categories c ON pp.categoryid = c.categoryid

Where EXISTS

(Select pp.productid as pID

From Sales.OrderDetails od

Where od.productid = pp.productid

HAVING SUM((od.qty\*od.unitprice)\*(1-od.discount)) > 50.000

)

)

UNION

(Select COUNT(pp.productid) AS noPrd

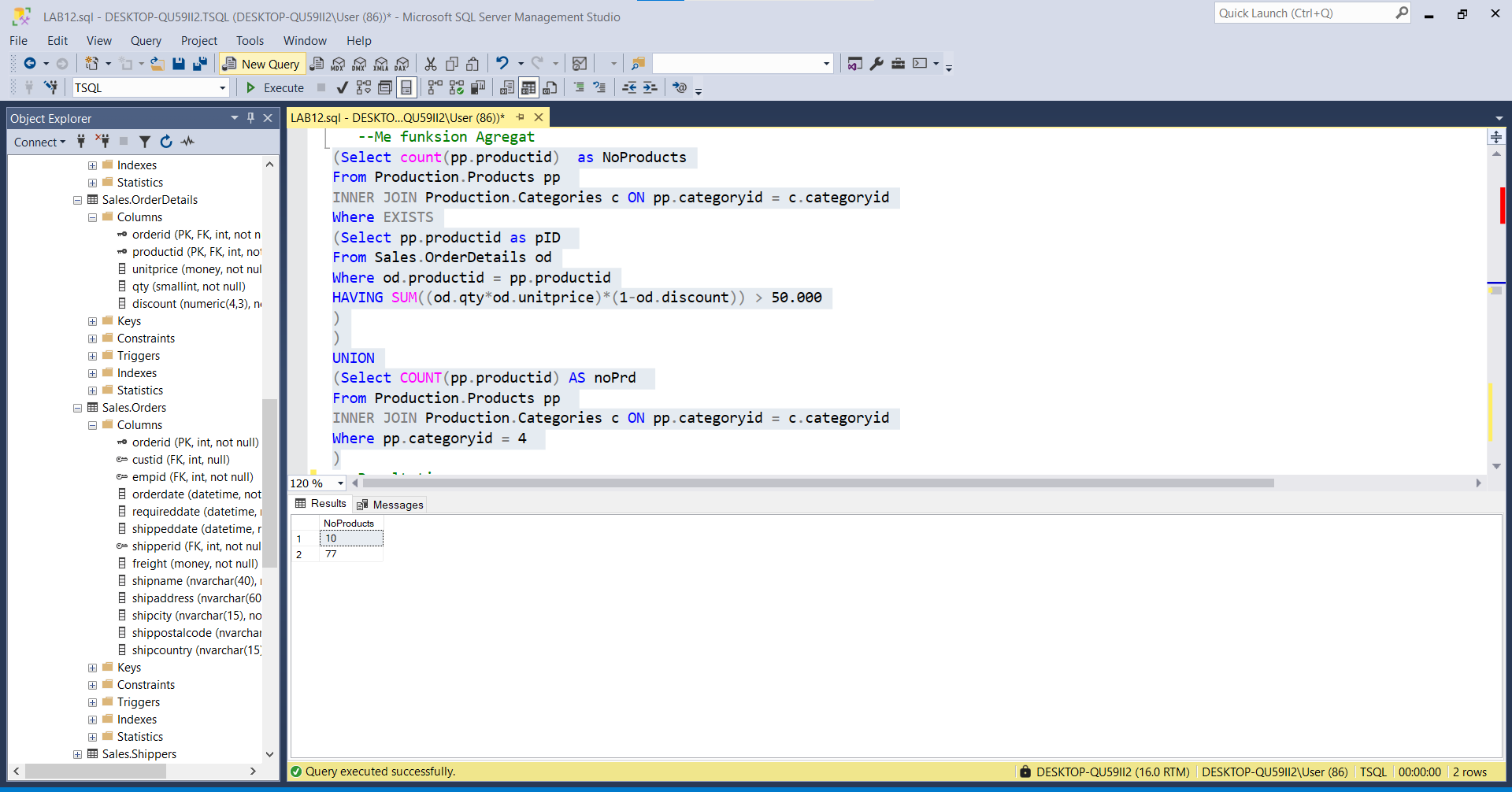
From Production.Products pp

INNER JOIN Production.Categories c ON pp.categoryid = c.categoryid

Where pp.categoryid = 4

)

Ekzekutimi i querisë me sukses :



-- Rezultati

NoProducts

-----------

10

-----------

77

-----------

87 rreshta ne total sikurse ne kerkesen me siper 'pa funksion agregat'-

--4-- Kopjoni veprimet T-SQL dhe modifikoni ate per te perdorur operatorin UNION ALL.

(Select pp.productid ,pp.productname,SUM((od.qty\*od.unitprice)\*(1-od.discount)) as TotalSales

From Production.Products pp

inner join Sales.OrderDetails od ON pp.productid = od.productid

GROUP BY pp.productid , pp.productname

HAVING SUM((od.qty\*od.unitprice)\*(1-od.discount)) > 50.000

)UNION ALL

(Select p.productid,p.productname,p.categoryid

From Production.Products p

INNER JOIN Production.Categories c ON p.productid = p.productid

Where p.categoryid = 4)

--5-- Ekzekutoni veprimet e mesiperme dhe krahasoni rezultatet qe ju moret me rezultatet e deshiruara

-- te treguara ne dokumentin 55 - Lab Exercise 1 - Task 3\_2 Result.txt.

\*\* Rezultojne 'te dublikuara' 167 rreshta ne total

-- Cili eshte numri total i rrjeshtave ne rezultat?

167 rreshta = 77 + 90 (nga rezultatet e mësipërme)

-- Cila eshte ndyshimi midis operatoreve UNION dhe UNION ALL?

UNION ALL afishon rezultatin me dublikata ndryshe nga UNION

---------------------------------------------------------------------

---------------------------------------------------------------------

-- Task 4

--

-- Write a SELECT statement to retrieve the custid and contactname columns from the Sales.Customers table.

-- Display the top 10 customers by sales amount for January 2008 and display the top 10 customers

-- by sales amount for February 2008 (Hint: Write two SELECT statements each joining

-- Sales.Customers and ?Sales.OrderValues and use the appropriate set operator.)

-- Execute the T-SQL code and compare the results that you got with the desired results

-- shown in the file 56 - Lab Exercise 1 - Task 4 Result.txt.

---------------------------------------------------------------------

---------------------------------------------------------------------

-- Detyra 4

--

-- Selektoni kolonat custid dhe contactname nga tabela Sales.Customers.

-- Shfaq top-10 klientet sipas shumes se shitjeve per Janar 2008 dhe shfaq top-10 klientet

-- sipas shumes se shitjeve per Shkurt 2008(Ndihme:Shkruaj dy veprime SELECT secila

-- duke u bashkuar me Sales.Customers dhe ?Sales.OrderValues dhe perdor operatorin e duhur set.)

-- Ekzekutoni kodin T-SQL dhe krahasoni rezultatin qe ju moret me rezultatin e deshiruar

-- te treguar ne dokumentin 56 - Lab Exercise 1 - Task 4 Result.txt.

---------------------------------------------------------------------

--Shfaqim Shumen

WITH CTE\_c20082

AS

(Select TOP(22) sc.custid , sc.contactname ,SUM((od.qty\*od.unitprice)\*(1-od.discount)) as TotalSales\_February ,year(o.orderdate) as yeari

From Sales.Customers as sc inner join Sales.Orders as o ON o.custid = sc.custid

inner join Sales.OrderDetails as od ON o.orderid = od.orderid

Where year(o.orderdate) = 2008 And DATENAME(month,o.orderdate)='February'

Group by sc.custid , sc.contactname, year(o.orderdate)

Order by TotalSales\_February DESC)

,CTE\_c20081

AS

(Select TOP(22) sc.custid , sc.contactname ,SUM((od.qty\*od.unitprice)\*(1-od.discount)) as TotalSales\_JANUARY ,year(o.orderdate) as yeari

From Sales.Customers as sc inner join Sales.Orders as o ON o.custid = sc.custid

inner join Sales.OrderDetails as od ON o.orderid = od.orderid

Where year(o.orderdate) = 2008 And DATENAME(month,o.orderdate) = 'January'

Group by sc.custid , sc.contactname , year(o.orderdate)

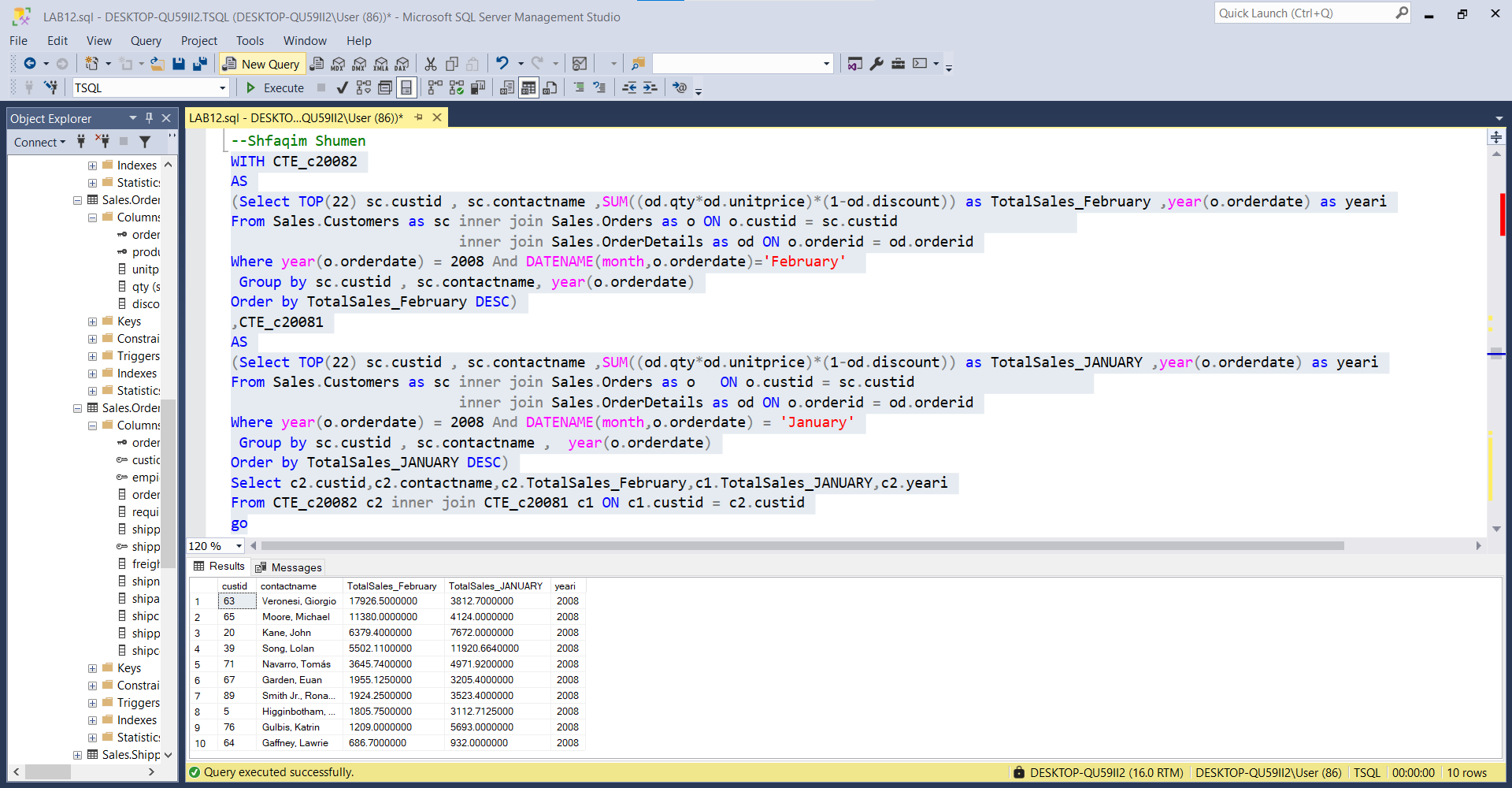
Order by TotalSales\_JANUARY DESC)

Select c2.custid,c2.contactname,c2.TotalSales\_February,c1.TotalSales\_JANUARY,c2.yeari

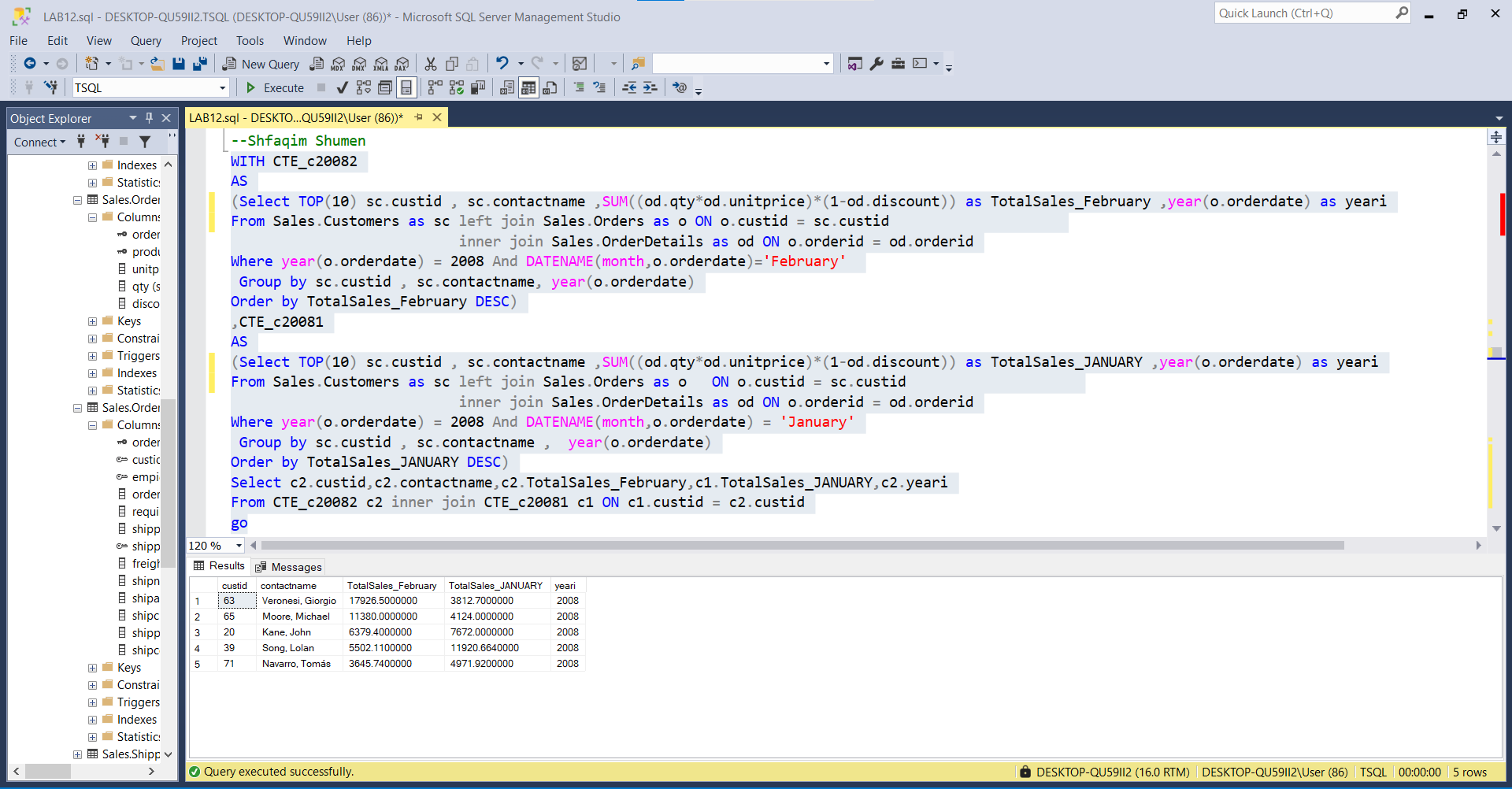
From CTE\_c20082 c2 inner join CTE\_c20081 c1 ON c1.custid = c2.custid

Go

Ekzekutimi me Sukses i Querisë është :



Nqs marrim TOP(10) ,rezultati I ekzekutuar eshte si me poshtë në pese rreshta :



---------------------------------------------------------------------

-- LAB 12

-- Exercise 2

---------------------------------------------------------------------

USE TSQL;

GO

---------------------------------------------------------------------

-- Task 1

--

-- Write a SELECT statement to retrieve the productid and productname columns from

-- the Production.Products table. In addition, for each product, retrieve

-- the last two rows from the Sales.OrderDetails table based on orderid number.

-- Use the CROSS APPLY operator and a correlated subquery. Order the result by the column productid.

-- Execute the written statement and compare the results that you got

-- with the desired results shown in the file 62 - Lab Exercise 2 - Task 1 Result.txt.

---------------------------------------------------------------------

---------------------------------------------------------------------

-- Detyre 1

--

-- Selektoni kolonat productid dhe productname nga tabela Production.Products.

-- Pastaj, per cdo produkt, nxjerr dy rrjeshtat e fundit nga tabela Sales.OrderDetails bazuar ne orderid number.

-- Perdor operatorin CROSS APPLY dhe nje subquery te koreluar. Rendit rezultatin sipas kolones productid.

-- Ekzekutoni veprimet e mesiperme dhe krahasoni rezultatet qe ju moret me

-- rezultatet e deshiruara te treguara ne dokumentin 62 - Lab Exercise 2 - Task 1 Result.txt.

---------------------------------------------------------------------

Select pp.productid , pp.productname

From Sales.OrderDetails od CROSS APPLY Production.Products pp

WHERE od.orderid IN (Select TOP(2) odd.orderid

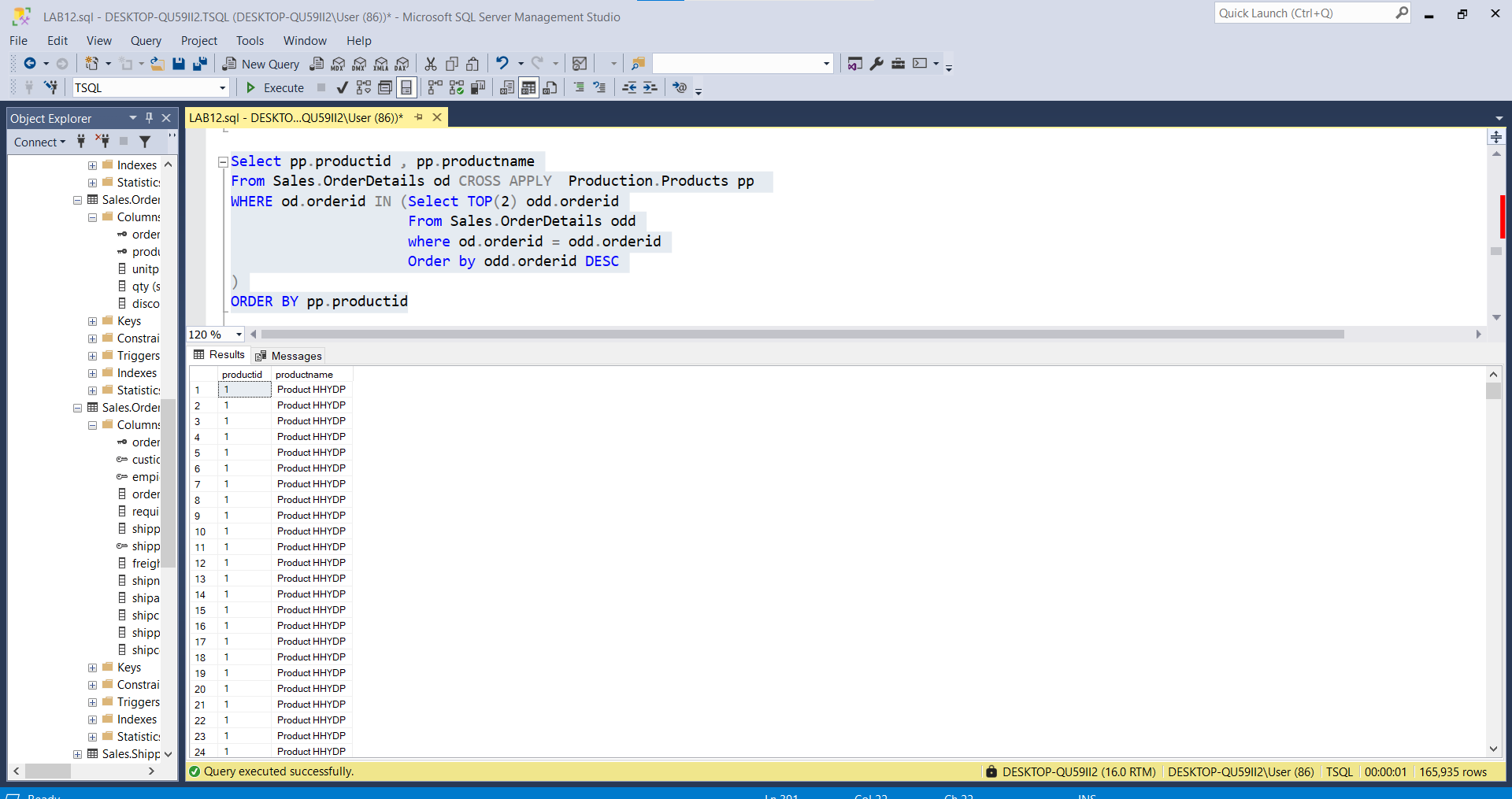
From Sales.OrderDetails odd

where od.orderid = odd.orderid

Order by odd.orderid DESC

)

ORDER BY pp.productid



---------------------------------------------------------------------

-- Task 2

--

-- Execute the provided T-SQL code to create the inline table-valued function

-- fnGetTop3ProductsForCustomer, as you did in the previous module:

-- Write a SELECT statement to retrieve the custid and contactname columns from

-- the Sales.Customers table. Use the CROSS APPLY operator with the dbo.fnGetTop3ProductsForCustomer

-- function to retrieve productid, productname, and totalsalesamount columns for each customer.

-- Execute the written statement and compare the results that you got with the recommended

-- result shown in the file 63 - Lab Exercise 2 - Task 2 Result.txt. Remember the number of rows in the result.

---------------------------------------------------------------------

IF OBJECT\_ID('dbo.fnGetTop3ProductsForCustomer') IS NOT NULL

DROP FUNCTION dbo.fnGetTop3ProductsForCustomer;

GO

CREATE FUNCTION dbo.fnGetTop3ProductsForCustomer1

(@custid AS INT) RETURNS TABLE

AS

RETURN

SELECT TOP(3) p.productid,p.productname AS ProductName, SUM(d.qty \* d.unitprice) AS totalsalesamount,

c.custid , c.contactname

FROM Sales.Customers c CROSS APPLY Sales.Orders AS o

INNER JOIN Sales.OrderDetails AS d ON d.orderid = o.orderid

INNER JOIN Production.Products AS p ON p.productid = d.productid

WHERE c.custid = @custid

GROUP BY c.custid , c.contactname , p.productid,p.productname

ORDER BY totalsalesamount DESC;

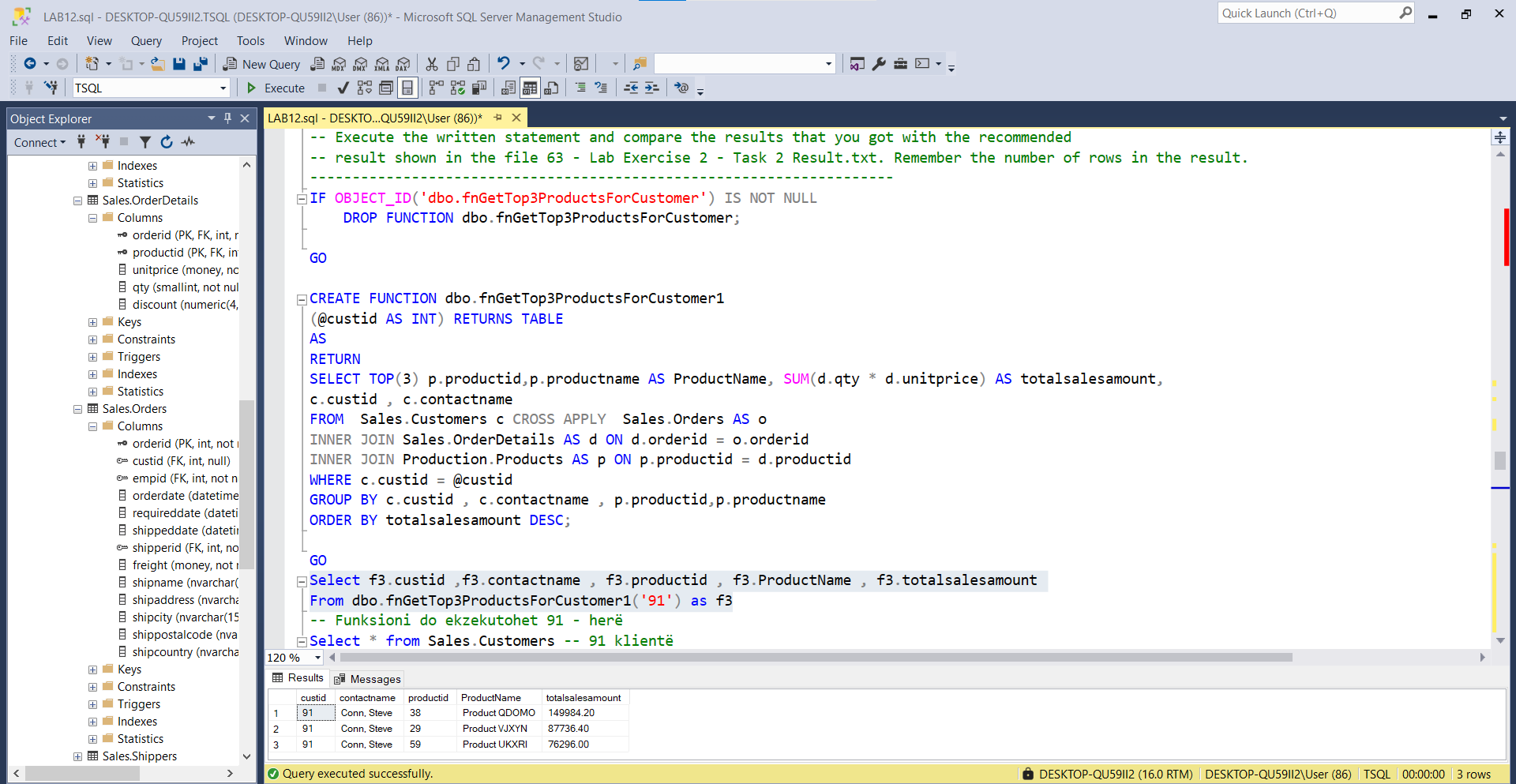
GO

Select f3.custid ,f3.contactname , f3.productid , f3.ProductName , f3.totalsalesamount

From dbo.fnGetTop3ProductsForCustomer1('91') as f3

-- Funksioni do ekzekutohet 91 - herë

Select \* from Sales.Customers -- 91 klientë



---------------------------------------------------------------------

-- Task 3

--

-- Copy the T-SQL statement from the previous task and modify it by replacing

-- the CROSS APPLY operator with the OUTER APPLY operator.

-- Execute the written statement and compare the results that you got with

-- the recommended result shown in the file 64 - Lab Exercise 2 -

-- Task 3 Result.txt. Notice that you got more rows than in the previous task.

---------------------------------------------------------------------

---------------------------------------------------------------------

-- Detyra 3

--

-- Kopjoni veprimet T-SQL nga detyra e meparshme dhe modifikoni ate

-- duke zevendesuar operatorin CROSS APPLY me operatorin OUTER APPLY.

-- Ekzekutoni veprimet e mesiperme dhe krahasoni rezultatet qe ju moret

-- me rezultatet e deshiruara te treguara ne dokumentin 64 - Lab Exercise 2 -

-- Task 3 Result.txt. Notice that you got more rows than in the previous task.

---------------------------------------------------------------------

CREATE FUNCTION dbo.fnGetTop3ProductsForCustomer12

(@custid AS INT) RETURNS TABLE

AS

RETURN

SELECT TOP(3) p.productid,p.productname AS ProductName, SUM(d.qty \* d.unitprice) AS totalsalesamount,

c.custid , c.contactname

FROM Sales.Customers c OUTER APPLY Sales.Orders AS o

INNER JOIN Sales.OrderDetails AS d ON d.orderid = o.orderid

INNER JOIN Production.Products AS p ON p.productid = d.productid

WHERE c.custid = @custid

GROUP BY c.custid , c.contactname , p.productid,p.productname

ORDER BY totalsalesamount DESC;

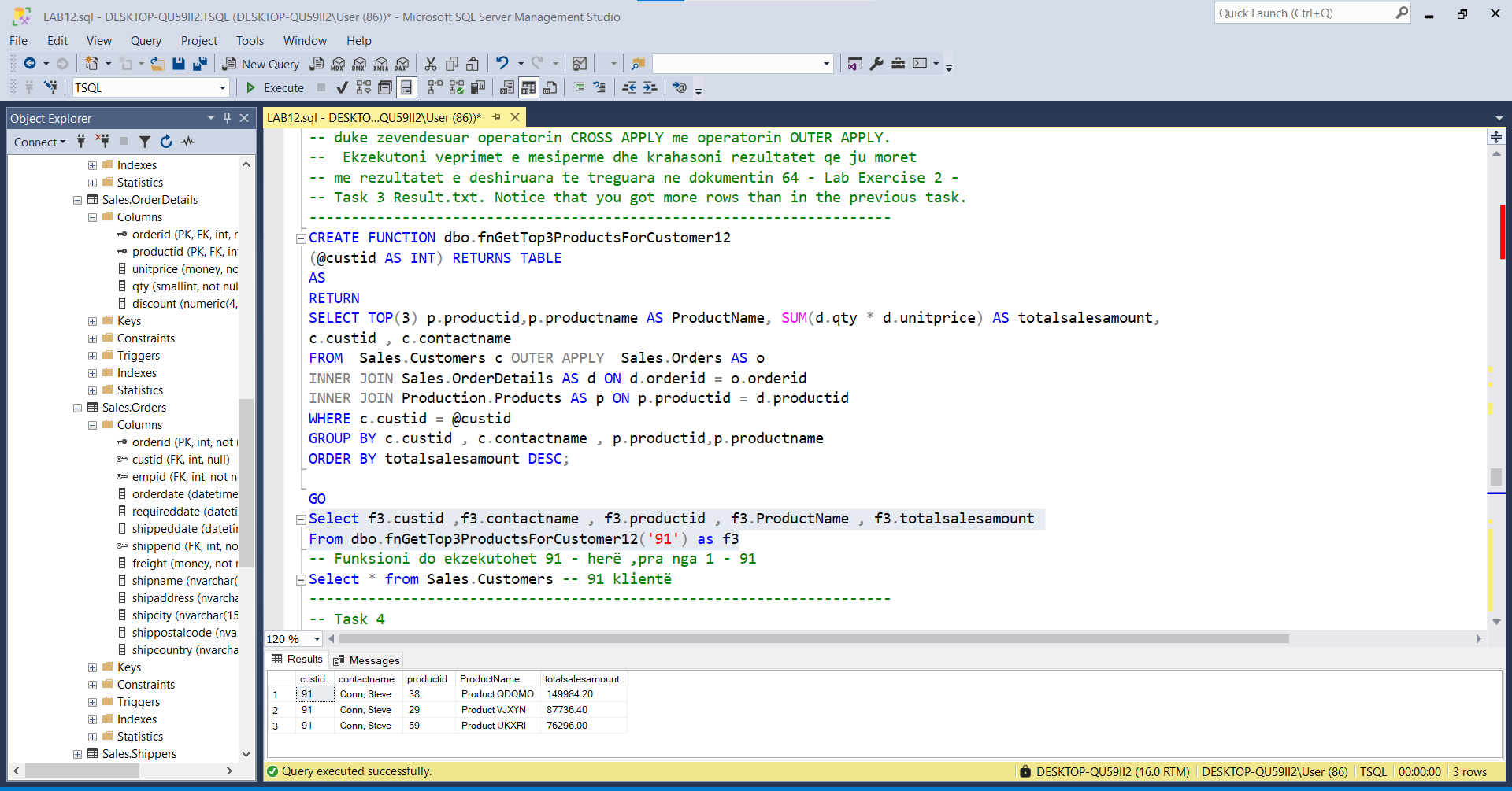
GO

Select f3.custid ,f3.contactname , f3.productid , f3.ProductName , f3.totalsalesamount

From dbo.fnGetTop3ProductsForCustomer12('91') as f3

-- Funksioni do ekzekutohet 91 - herë ,pra nga 1 - 91

Select \* from Sales.Customers -- 91 klientë ---------------------------------------------------------------------



-- Task 4

--

-- Copy the T-SQL statement from the previous task and modify it

-- by filtering the results to show only customers without products.

-- (Hint: In a WHERE clause, look for any column returned by the inline table-valued function that is NULL.)

-- Execute the written statement and compare the results that you got with the

-- recommended result shown in the file 65 - Lab Exercise 2 - Task 4 Result.txt.

-- What is the difference between the CROSS APPLY and OUTER APPLY operators?

---------------------------------------------------------------------

CREATE FUNCTION dbo.fnGetTop3ProductsForCustomer123

(@custid AS INT) RETURNS TABLE

AS

RETURN

SELECT Top(3) SUM(d.qty \* d.unitprice) AS totalsalesamount,

c.custid , c.contactname

FROM Sales.Customers c OUTER APPLY Sales.Orders AS o

INNER JOIN Sales.OrderDetails AS d ON d.orderid = o.orderid

INNER JOIN Production.Products AS p ON p.productid = d.productid

WHERE c.custid = @custid

GROUP BY c.custid , c.contactname

ORDER BY totalsalesamount DESC;

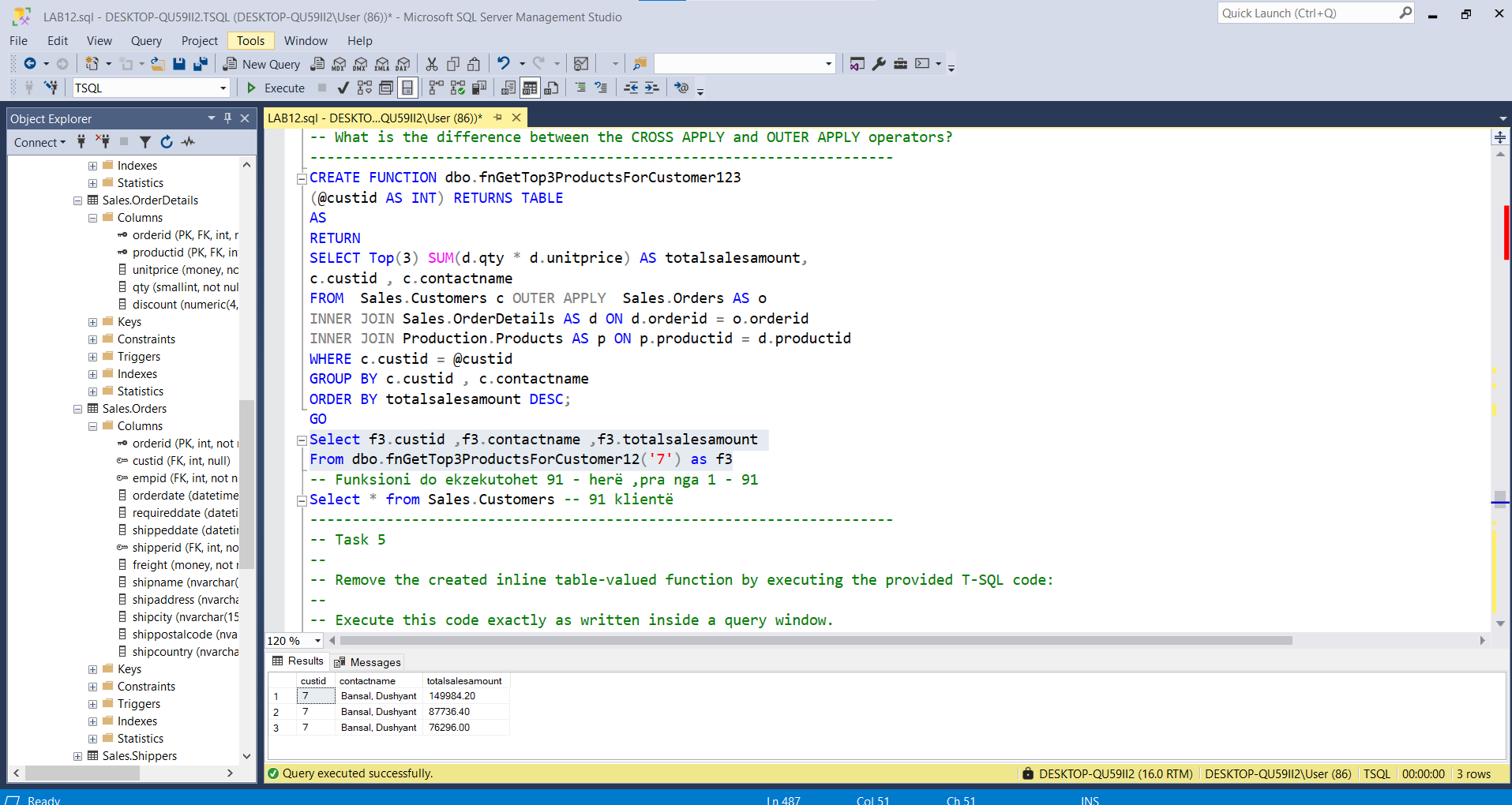
GO

Select f3.custid ,f3.contactname ,f3.totalsalesamount

From dbo.fnGetTop3ProductsForCustomer12('7') as f3

-- Funksioni do ekzekutohet 91 - herë ,pra nga 1 - 91

Select \* from Sales.Customers -- 91 klientë



---------------------------------------------------------------------

-- Task 5

--

-- Remove the created inline table-valued function by executing the provided T-SQL code:

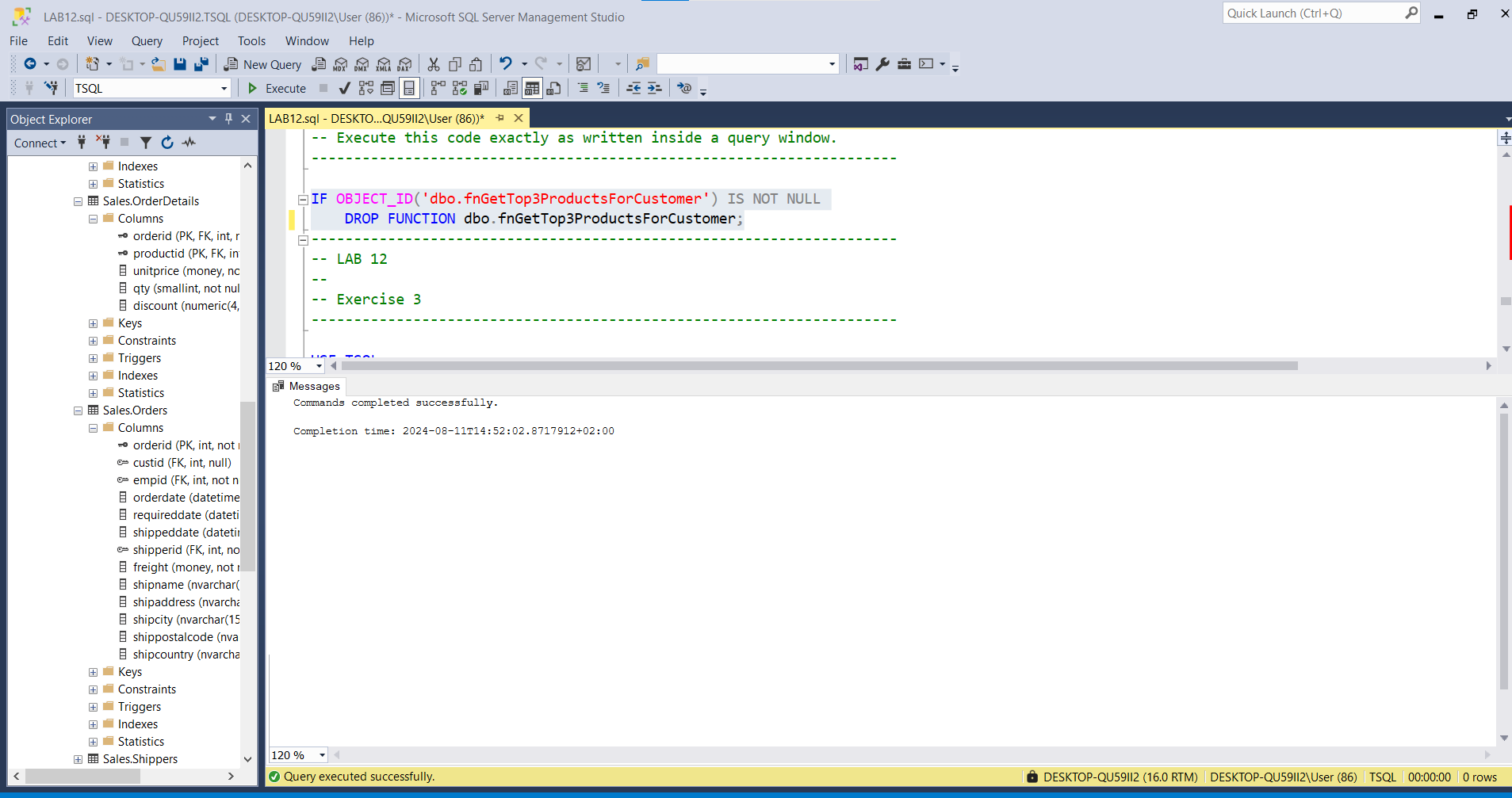
--

-- Execute this code exactly as written inside a query window.

---------------------------------------------------------------------

IF OBJECT\_ID('dbo.fnGetTop3ProductsForCustomer') IS NOT NULL

DROP FUNCTION dbo.fnGetTop3ProductsForCustomer;



---------------------------------------------------------------------

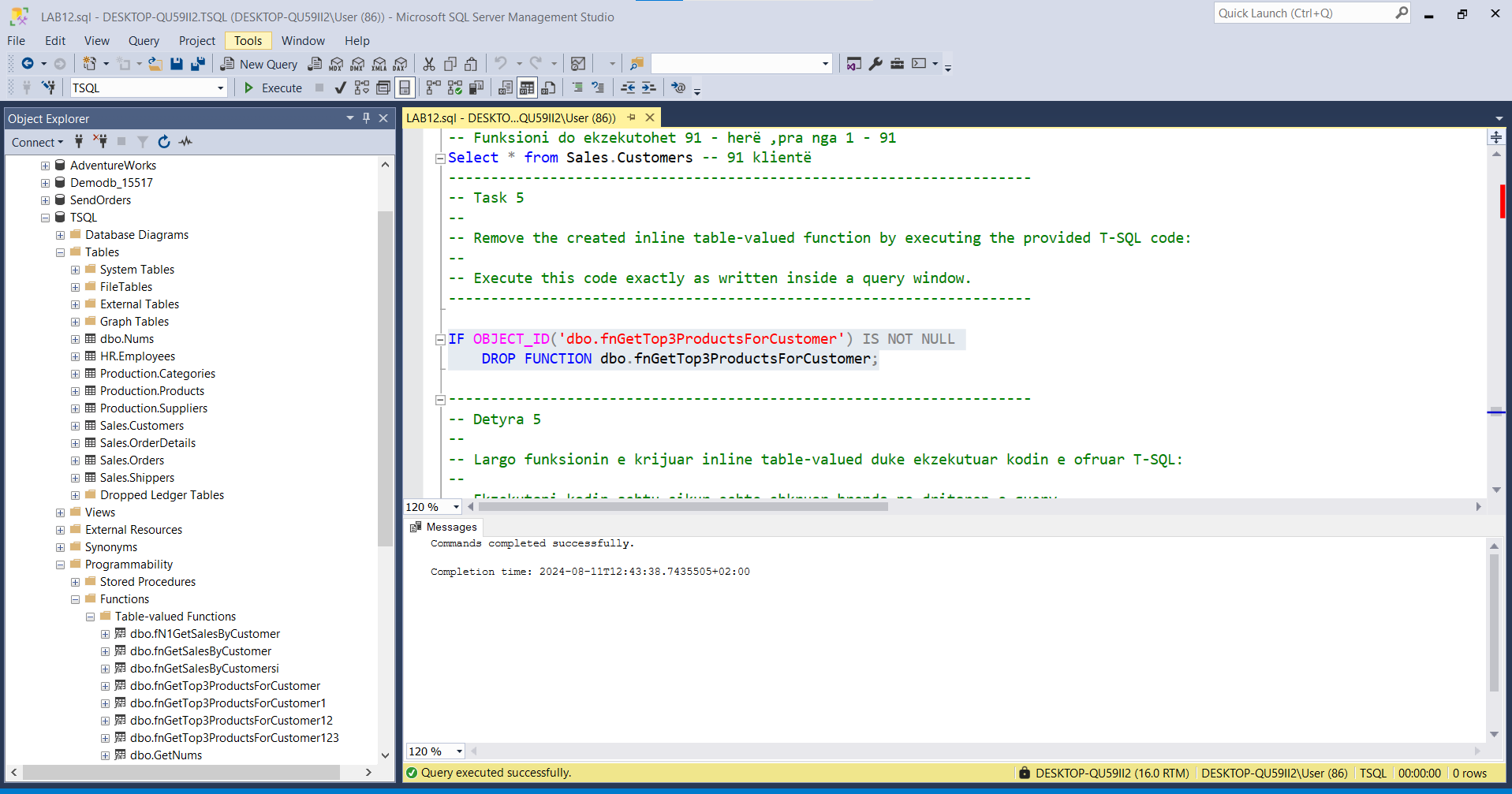
-- Detyra 5

-- Largo funksionin e krijuar inline table-valued duke ekzekutuar kodin e ofruar T-SQL:

-- Ekzekutoni kodin ashtu sikur eshte shkruar brenda ne dritaren e query.

---------------------------------------------------------------------

IF OBJECT\_ID('dbo.fnGetTop3ProductsForCustomer') IS NOT NULL DROP FUNCTION dbo.fnGetTop3ProductsForCustomer;



---------------------------------------------------------------------

-- LAB 12

--

-- Exercise 3

-----------------------------------------------------------------------------------------------------------

USE TSQL;

GO

------------------------------------------------------------------------------------------------------

-- Task 1

--

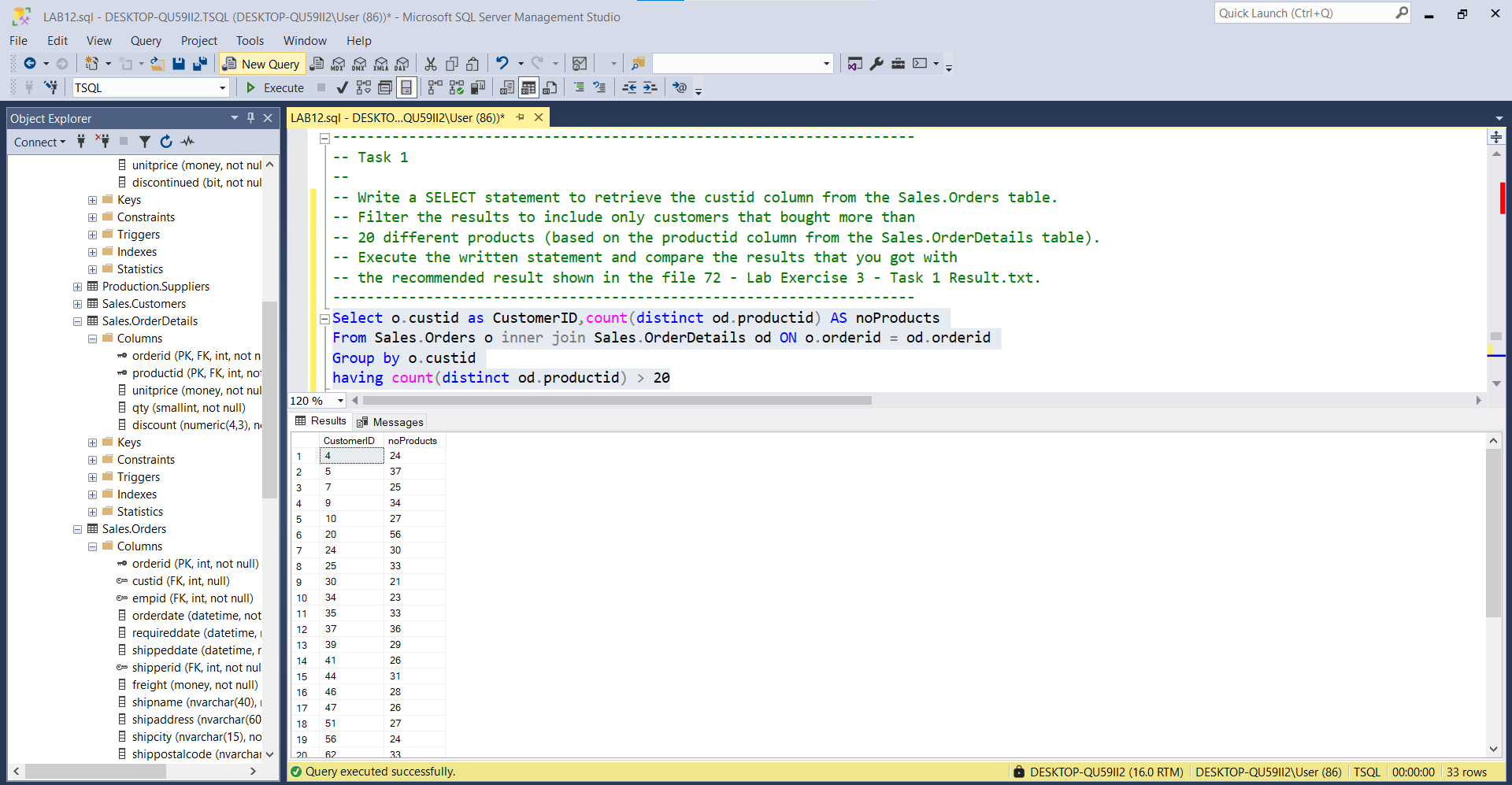
-- Write a SELECT statement to retrieve the custid column from the Sales.Orders table. Filter the results to include only customers that bought more than 20 different products (based on the productid column from the Sales.OrderDetails table).

Select o.custid as CustomerID,count(distinct od.productid) AS noProducts

From Sales.Orders o inner join Sales.OrderDetails od ON o.orderid = od.orderid

Group by o.custid

having count(distinct od.productid) > 20 -- Execute the written statement and compare the results that you got with the recommended result shown in the file 72 - Lab Exercise 3 - Task 1 Result.txt.



-------------------------------------------------------------------------------------------------------

Result:

CustomerID |noProducts

4 24

5 37

7 25

9 34

10 27

20 56

24 30

25 33

30 21

34 23

35 33

37 36

39 29

41 26

44 31

46 28

47 26

51 27

56 24

62 33

63 49

65 45

67 21

68 26

71 53

72 23

76 31

80 24

83 23

84 22

86 22

87 28

89 32

-- Task 2

-------------------------

-- Write a SELECT statement to retrieve the custid column from the Sales.Orders table. Filter the results to include only customers from the country USA and exclude all customers from the previous (task 1) result. (Hint: Use the EXCEPT operator and the previous query.)

(Select c.custid ,c.country

From Sales.Customers c inner join Sales.Orders o ON c.custid = o.custid

Where c.country = N'USA')

EXCEPT

(Select o.custid as CustomerID,noProducts = CAST(count(distinct od.productid) AS nvarchar(25))

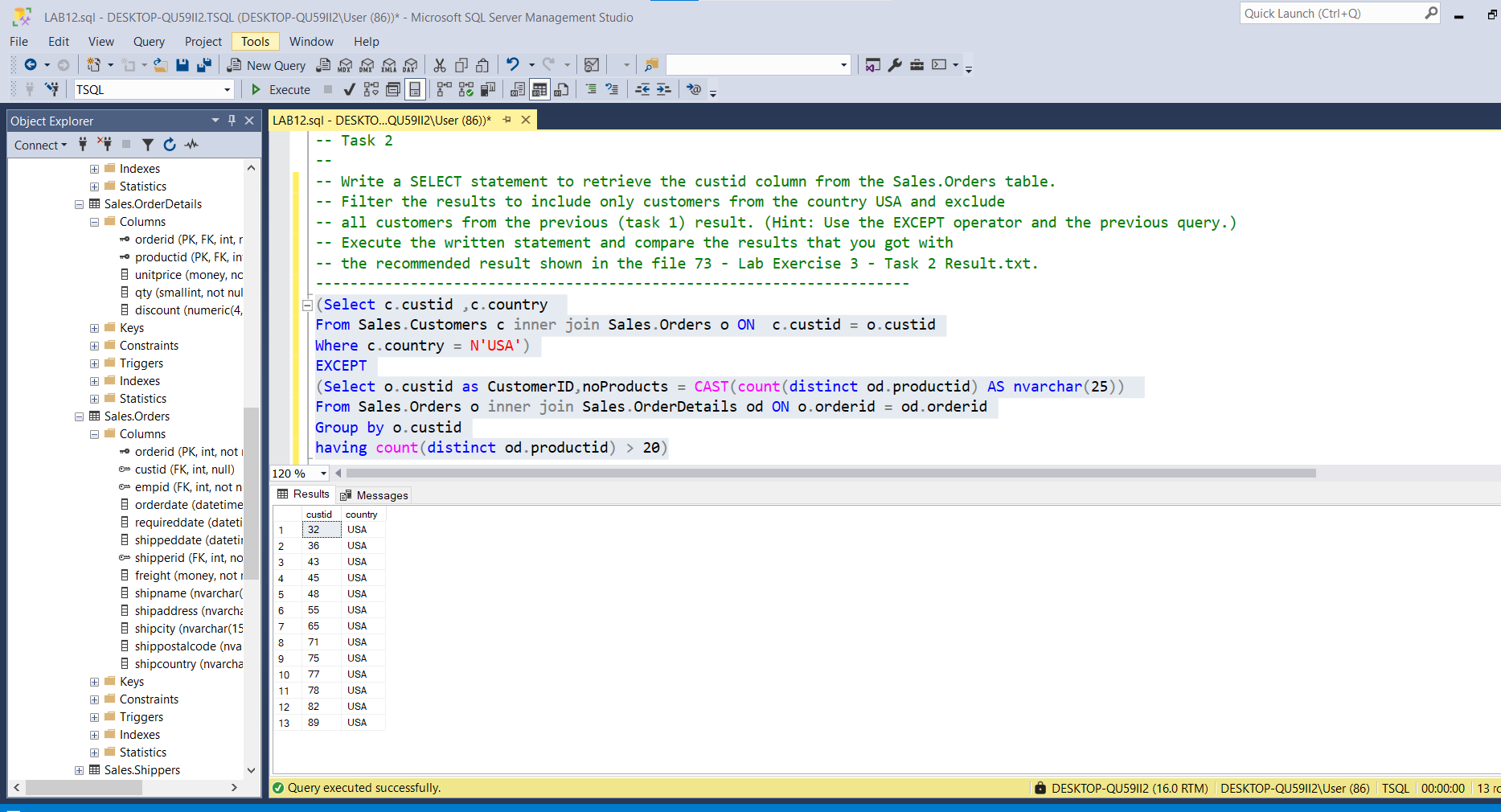
From Sales.Orders o inner join Sales.OrderDetails od ON o.orderid = od.orderid

Group by o.custid

having count(distinct od.productid) > 20)

-- Execute the written statement and compare the results that you got with the recommended result

shown in the file 73 - Lab Exercise 3 - Task 2 Result.txt.



--------------------------------------------------------------------

---------------------------------------------------------------------

-- Task 3

--

-- Write a SELECT statement to retrieve the custid column from the Sales.Orders table. Filter only customers that have a total sales value greater than $10,000. Calculate the sales value using the qty and unitprice columns from the Sales.OrderDetails table.

Select c.custid as CustomerID,SUM(sum1.totalSalesValue) TotalValue

From Sales.Customers c inner join Sales.Orders o ON c.custid = o.custid

inner join (Select od.orderid ,SUM((od.qty\*od.unitprice)\*(1-od.discount)) as totalSalesValue

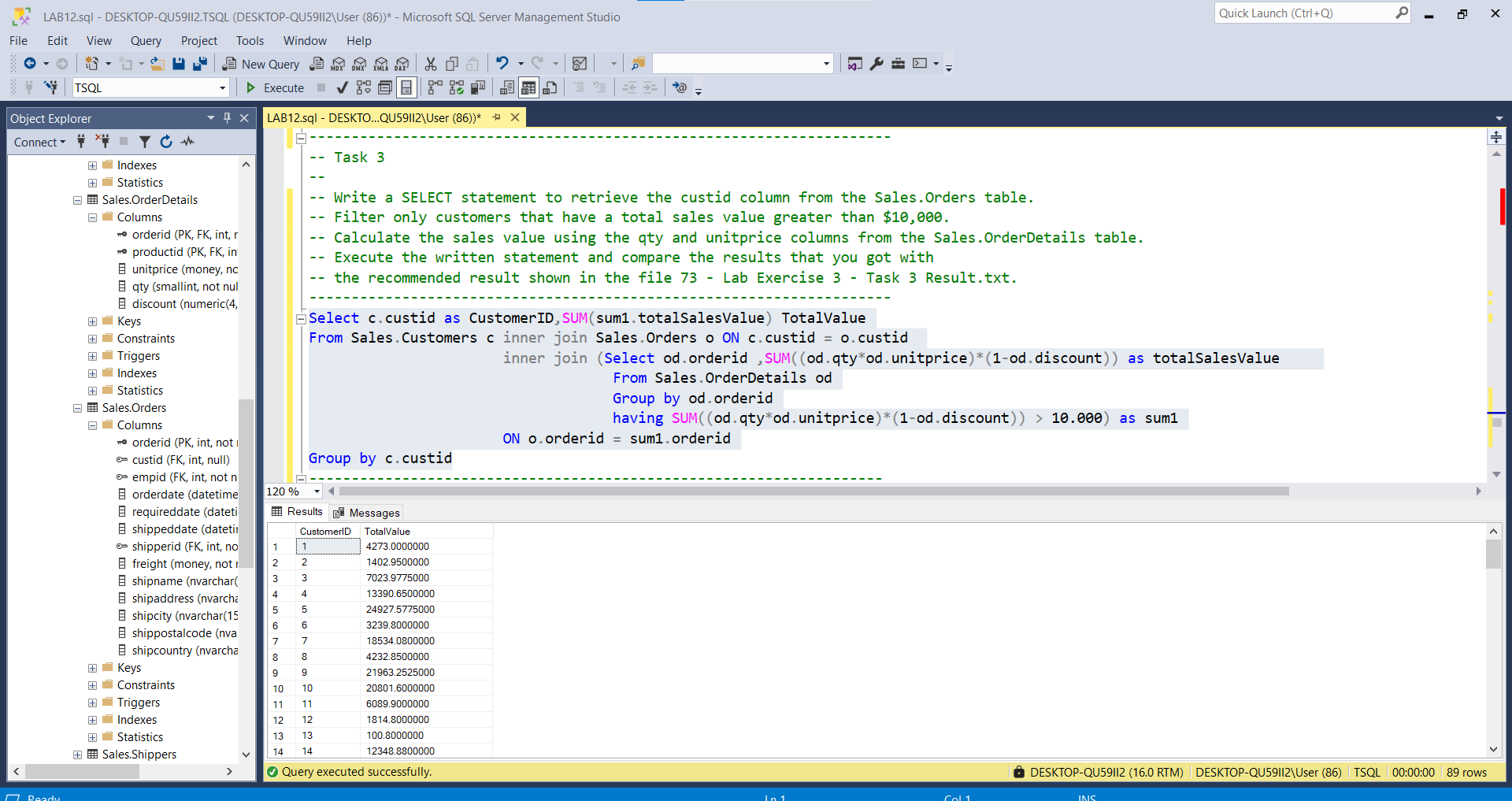
From Sales.OrderDetails od

Group by od.orderid

having SUM((od.qty\*od.unitprice)\*(1-od.discount)) > 10.000) as sum1

ON o.orderid = sum1.orderid

Group by c.custid



-- Execute the written statement and compare the results that you got with the recommended result shown in the file 73 - Lab Exercise 3 - Task 3 Result.txt.

---------------------------------------------------------------------

---------------------------------------------------------------------

---------------------------------------------------------------------

-- Task 4

--

-- Copy the T-SQL statement from task 2. Add the INTERSECT operator at the end of the statement. After the INTERSECT operator, add the T-SQL statement from task 3.

(Select c.custid ,c.country

From Sales.Customers c inner join Sales.Orders o ON c.custid = o.custid

Where c.country = N'USA')

EXCEPT

(Select o.custid as CustomerID,noProducts = CAST(count(distinct od.productid) AS nvarchar(25))

From Sales.Orders o inner join Sales.OrderDetails od ON o.orderid = od.orderid

Group by o.custid

having count(distinct od.productid) > 20)

INTERSECT

(Select c.custid as CustomerID,CONVERT(NVARCHAR(25),SUM(sum1.totalSalesValue)) AS TotalValue

From Sales.Customers c inner join Sales.Orders o ON c.custid = o.custid

inner join (Select od.orderid ,SUM((od.qty\*od.unitprice)\*(1-od.discount)) as totalSalesValue

From Sales.OrderDetails od

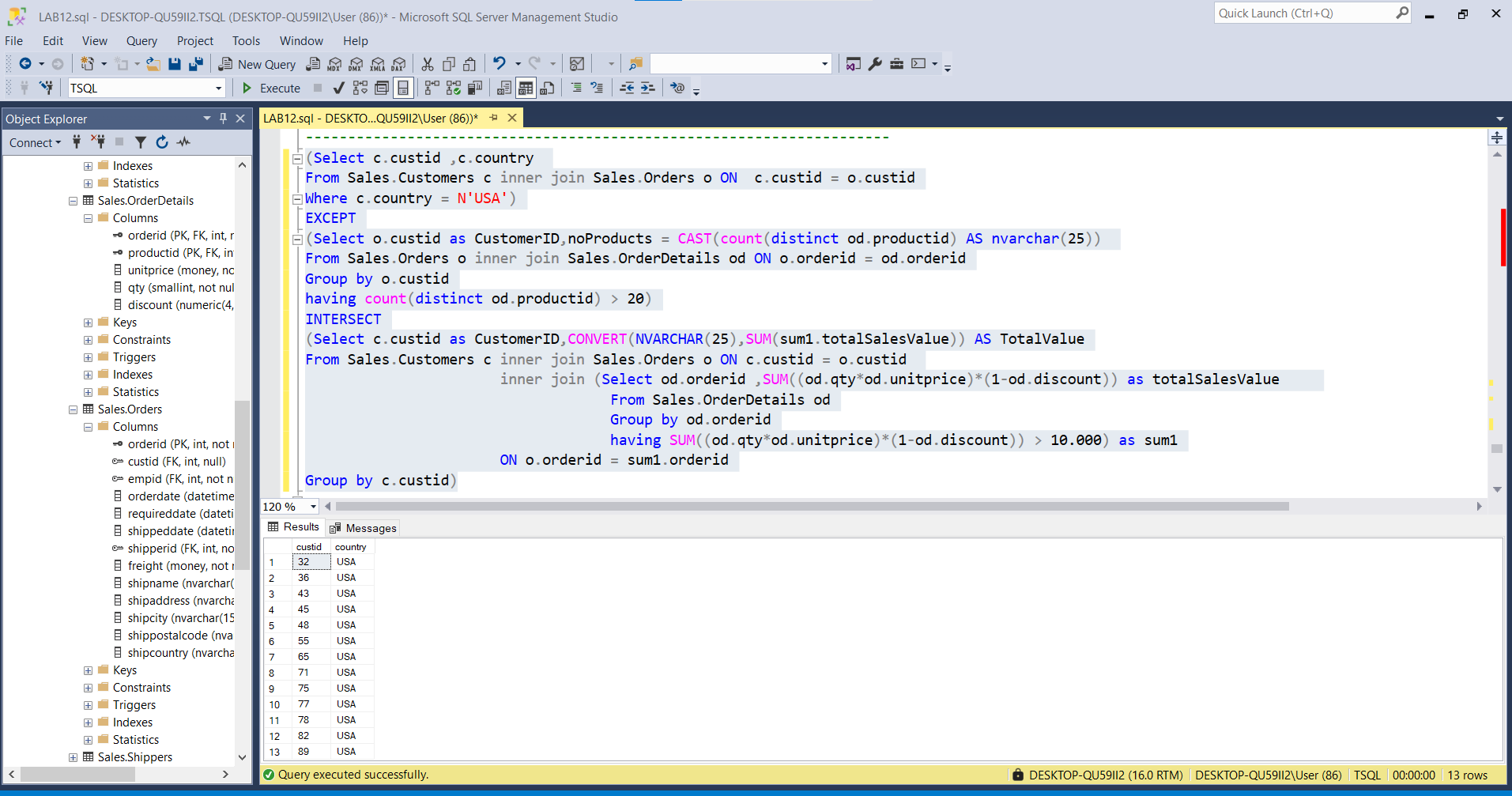
Group by od.orderid

having SUM((od.qty\*od.unitprice)\*(1-od.discount)) > 10.000) as sum1

ON o.orderid = sum1.orderid

Group by c.custid)

-- Execute the T-SQL statement and compare the results that you got with the recommended result shown in the file 74 - Lab Exercise 3 - Task 4 Result.txt. Notice the total number of rows in the result.



-- Can you explain in business terms which customers are part of the result?

---------------------------------------------------------------------

---------------------------------------------------------------------

-- Task 5

--

-- Copy the T-SQL statement from the previous task and add parentheses around the first two SELECT statements (from the beginning until the INTERSECT operator).

((Select c.custid ,c.country

From Sales.Customers c inner join Sales.Orders o ON c.custid = o.custid

Where c.country = N'USA')

EXCEPT

(Select o.custid as CustomerID,noProducts = CAST(count(distinct od.productid) AS nvarchar(25))

From Sales.Orders o inner join Sales.OrderDetails od ON o.orderid = od.orderid

Group by o.custid

having count(distinct od.productid) > 20))

INTERSECT

(Select c.custid as CustomerID,CONVERT(NVARCHAR(25),SUM(sum1.totalSalesValue)) AS TotalValue

From Sales.Customers c inner join Sales.Orders o ON c.custid = o.custid

inner join (Select od.orderid ,SUM((od.qty\*od.unitprice)\*(1-od.discount)) as totalSalesValue

From Sales.OrderDetails od

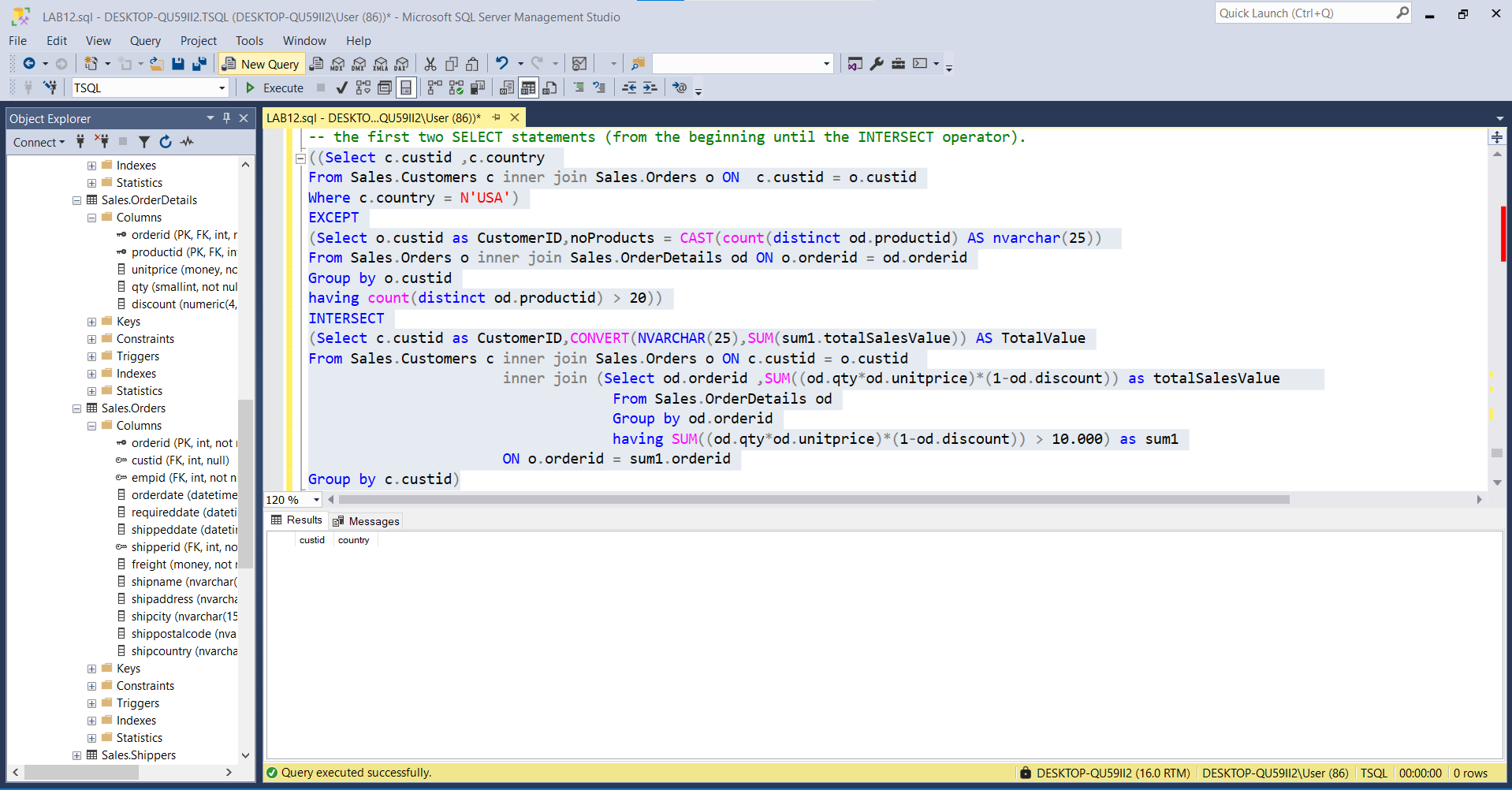
Group by od.orderid

having SUM((od.qty\*od.unitprice)\*(1-od.discount)) > 10.000) as sum1

ON o.orderid = sum1.orderid

Group by c.custid)

-- Execute the T-SQL statement and compare the results that you got with the recommended result shown in the file 75 - Lab Exercise 3 - Task 5 Result.txt. Notice the total number of rows in the result.



-- Is the result different than the result from task 4? Please explain why.

\*\*Result from the previous TASK(task 2) doesn’t have any mutual attribute’s tuples

-- What is the precedence among the set operators? First EXCEPT Second INTERCEPT

---------------------------------------------------------------------