JOBSHEET
PRAKTIKUM BASIS DATA LANJUT

Jurusan Teknologi Informasi

POLITEKNIK NEGERI MALANG

2024

Week 2

SQL SERVER- SELECT, JOIN SORTING DAN FILTERING DATA

Team Teaching:

Habibie Ed Dien, S.Kom., M.T.

Irsyad Arif Mashudi, S.Kom M.Kom

Vit Zuraida, S.Kom., M.Kom.

Rokhimatul Wakhidah, S.Pd., M.T.

Annisa Taufika Firdausi, ST., MT.

Elok Nur Hamdana, S.T., M.T

Information Technology Department, Malang State Polytechnic

Jobsheet- 1 : Introduction to Transact-SQL and Statements SELECT, Join, Sorting, and Filtering data

Advanced Database Course

Supervisor: Advanced Database Teaching Team *September 2024*

SAFRIZAL RAHMAN_19_SIB_2G

Topics

1. Introduction to T-SQL and *Query* Select
2. Querying Multiple Tables
3. Sorting and Filtering Data

Objective

Students are expected to be able to:

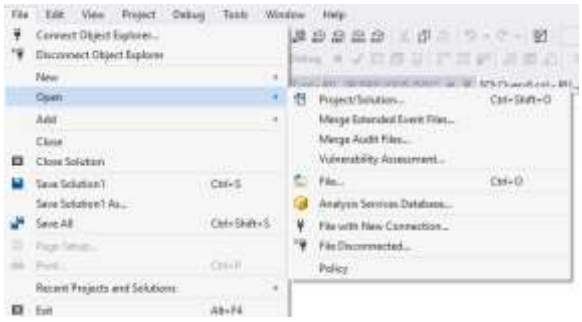
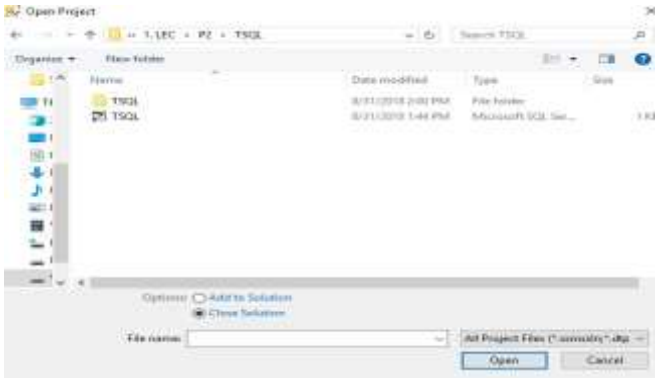
1. Understanding the basic differences between Transact-SQL (T-SQL) and ANSI SQL.
2. Understanding how to create *a database* from an existing SQL file 3. Understand how to execute part or all of a SQL *script* from an existing file.
4. Understanding the concept of using ' *comments* ' in T-SQL.
5. Understand the concept of using the SELECT statement to analyze existing tables in *a database* .
6. Understanding how to display data in a *unique* / *distinct manner* .
7. Understand how to use *ALIAS* for table names and column names.
8. Understand the concept of *CASE* expressions and how to use them.
9. Students understand how to query multiple tables in a SELECT clause using JOIN.
10. Students understand how to write INNER JOIN , OUTER JOIN , SELF-JOIN and CROSS JOIN queries .
11. Students understand how to do Data Sorting , Data Filtering with predicates , Data Filtering with TOP and OFFSET-FETCH
12. Students understand how to handle missing and unknown values in real data.

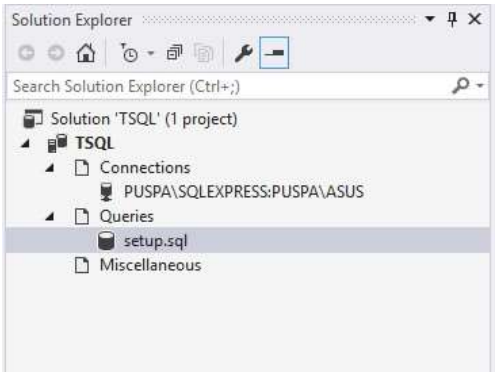
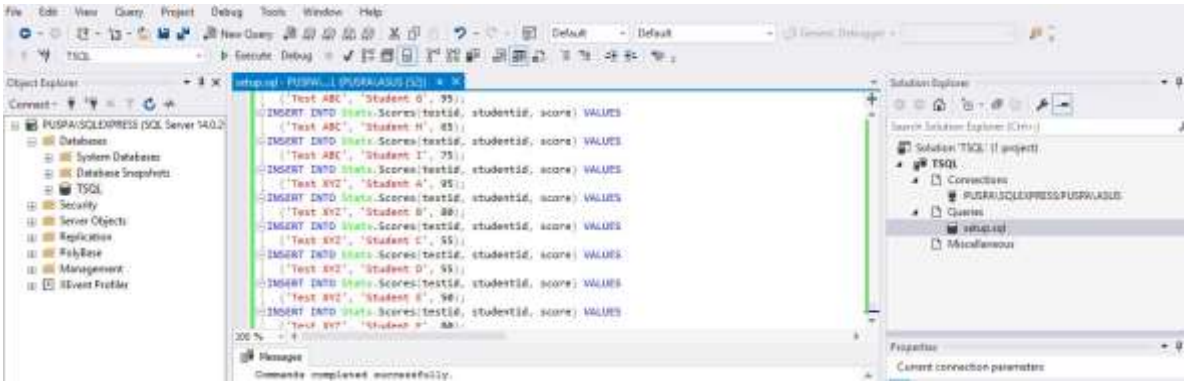
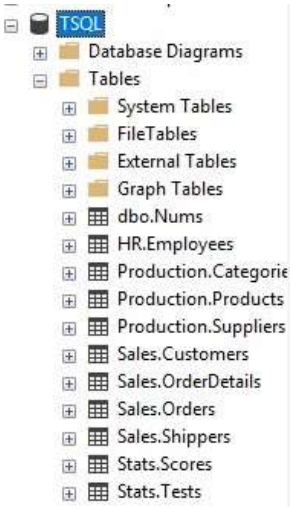
General Instructions

1. Follow the steps in the practical sections in the order given.
2. Answer all questions marked [Question-X] that are found in certain steps in each part of the practicum.
3. In each step of the practicum, there is an explanation that will help you answer the questions in instruction number 3, so read and do all the practicum parts in this jobsheet.

4. Write the answers to the questions in the instructions number 3 in a report that is done using a word processing application (Word, OpenOffice, or other similar). Export as a **PDF file** with the following name format:
   - **BDL_Class_03_YourFullName** .pdf
   - Collect the PDF files as a practical report to the supervising lecturer.
   - In addition to the file name, also include your identity on the first page of the report.

Practical Preparation: Creating a Database from Existing SQL

Step	Information
1	Create a TSQL database `CREATE DATABASE TSQL;`
2	On the **File menu** , click **Open** and click **Project/Solution** . 
3	In the **Open Project** file.  **window** , open the given project

4	Next, the Solution Explorer window will display the following display. Then please open the "Setup" file. This file contains the sql *script* to create the tables needed for this practicum. 
5	After the setup file is opened, a display like the image below will appear. Then click *Execute* and please wait until the process is complete.
	
6	After the process is successful, several tables will be formed, as shown in the image below. 

7

For example, to check *records* in the Sales.Customers table, please execute the command below:

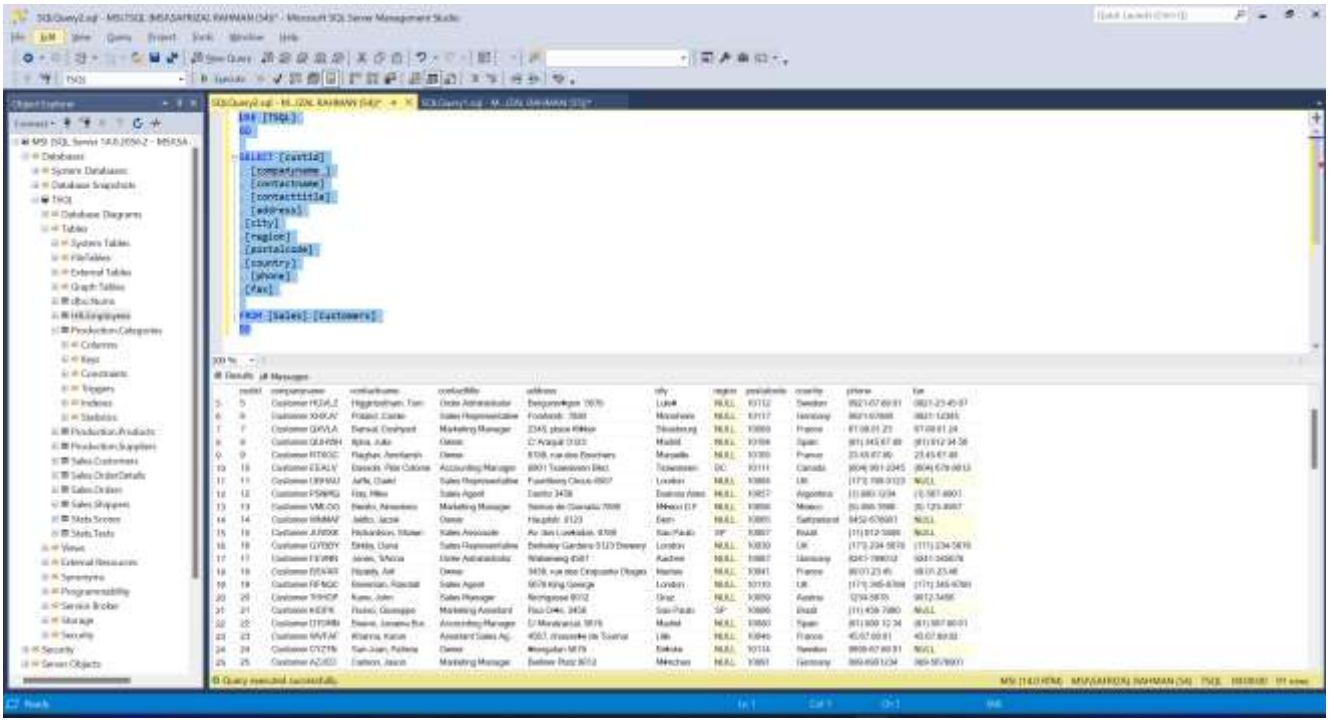
```
USE [TSQL]
GO

SELECT [custid]
      ,[companyname]
      ,[contactname]
      ,[contacttitle]
      ,[address]
      ,[city]
      ,[region]
      ,[postalcode]
      ,[country]
      ,[phone]
      ,[fax]
  FROM [Sales].[Customers]
GO
```

8

The results of the SQL command above are as follows

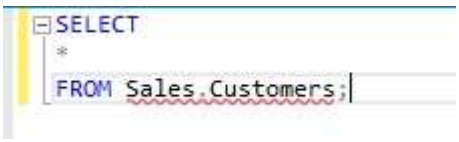
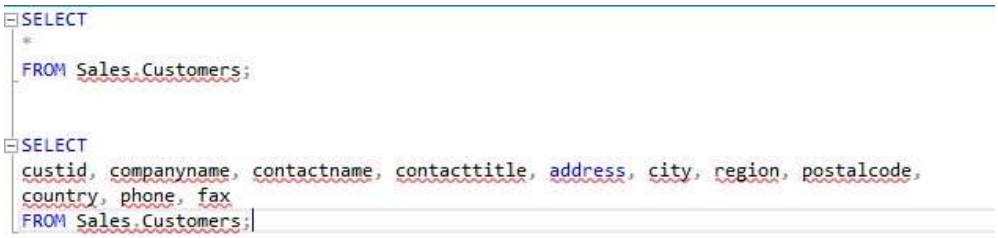
Results | Messages

	custid	companyname	contactname	contacttitle	address	city	region
1	1	Customer NRZBB	Allen, Michael	Sales Representative	Obere Str. 0123	Berlin	NULL
2	2	Customer MLTDN	Hassall, Mark	Owner	Avda. de la Constitución 5678	México D.F.	NULL
3	3	Customer KBUDE	Peoples, John	Owner	Mataderos 7890	México D.F.	NULL
4	4	Customer HFBZG	Arndt, Torsten	Sales Representative	7890 Hanover Sq.	London	NULL
5	5	Customer HGVLZ	Higginbotham, Tom	Order Administrator	Berguvsvägen 5678	Luleå	NULL

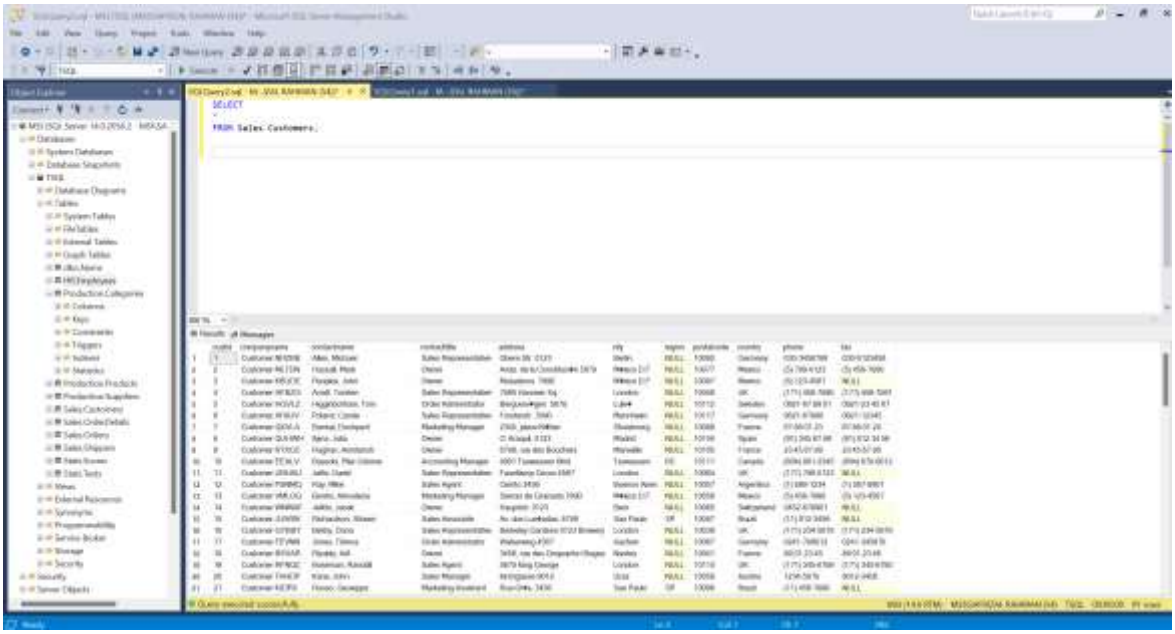


Practical –

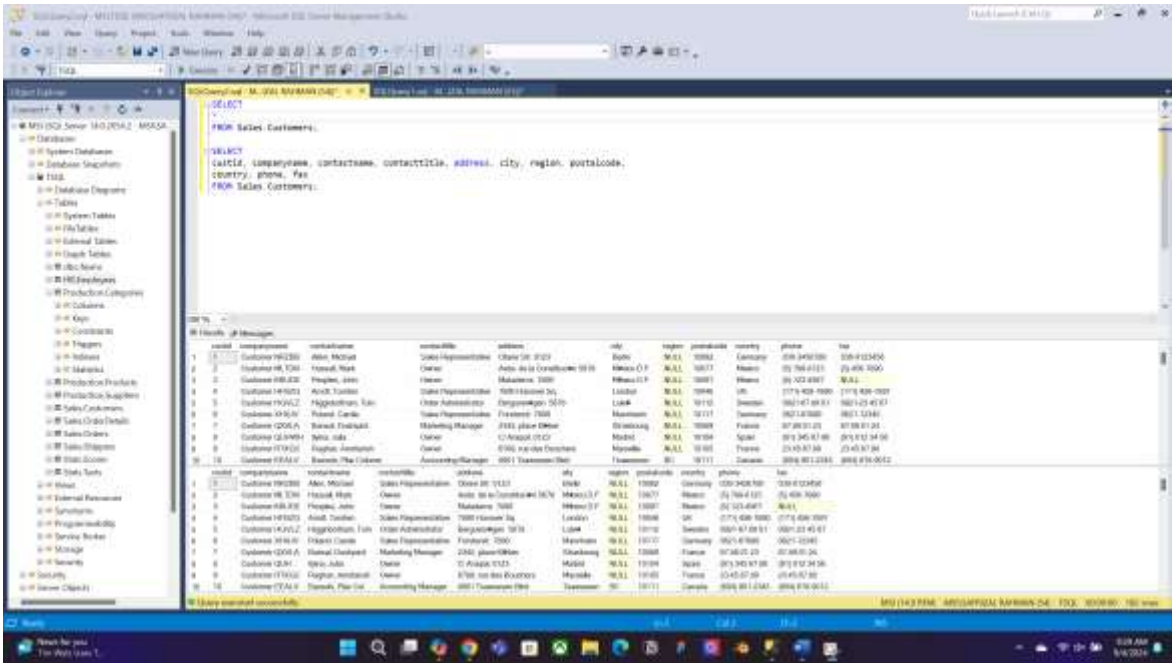
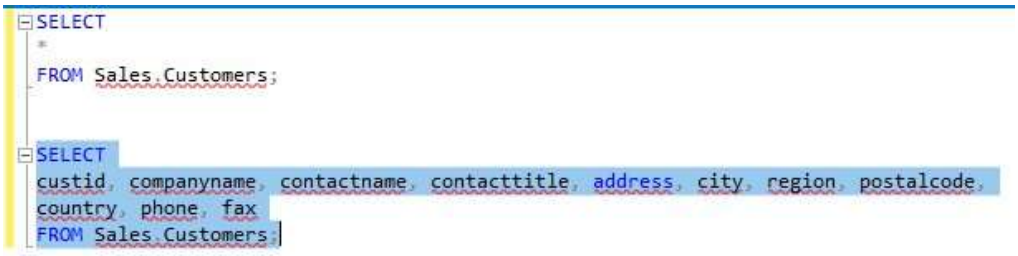
Part 1: Executing part or all of a SQL *script*

Step	Information
1	Please type the following *query in your query* panel then click *execute* . Note the results displayed. 
2	Please add the  following *query to* your *query panel then click execute* . Note the results

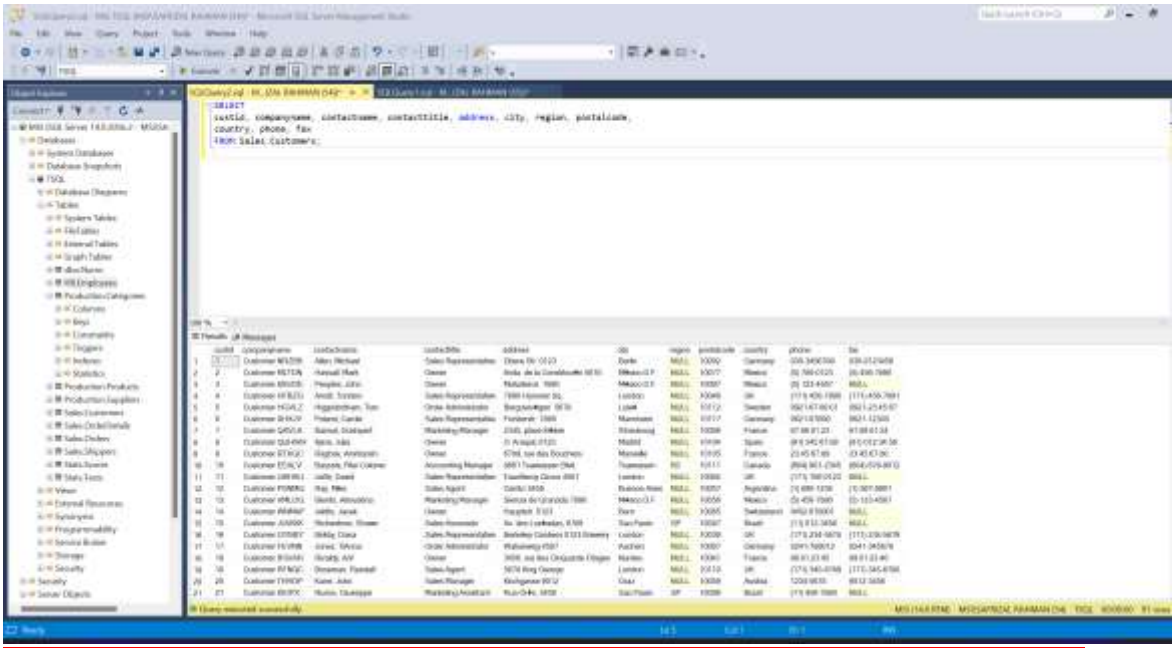
Practical –



Practical –

	 displayed
3	Make a selection on one of the existing *queries then click execute* . Note the results displayed. ***What is the difference with the results in the second step above?*** **(Question 1)** **SELECT** **custid, companyname, contactname, contacttitle, address, city, region, postalcode, country, phone, fax** **FROM Sales.Customers;**  SELECT * FROM Sales.Customers; SELECT custid, companyname, contactname, contacttitle, address, city, region, postalcode, country, phone, fax FROM Sales.Customers;

Practical –



The two SQL queries you've provided are similar but differ in the level of detail they return:

1. First Query:

```
sql
SELECT
FROM Sales.Customers;
```

- This query selects all columns from the Sales.Customers table. The asterisk () is a wildcard that tells the database to return every column for each row in the table.

2. Second Query:

```
sql
SELECT
custid, companyname, contactname, contacttitle, address, city, region, postalcode,
country, phone, fax
FROM Sales.Customers;
```

- This query selects specific columns from the Sales.Customers table. It explicitly lists the columns custid, companyname, contactname, contacttitle, address, city, region,

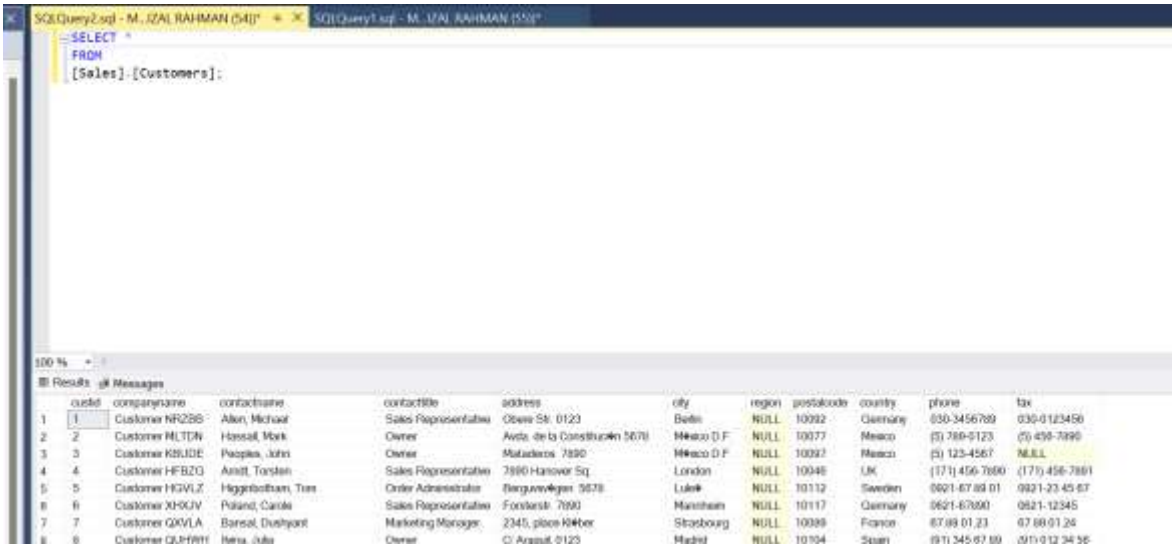
Practical –

	postalcode, country, phone, and fax, meaning only these columns will be returned in the result set. Difference in Results: - The first query (SELECT) will return all columns in the Sales.Customers table, regardless of how many columns the table contains. - The second query will return only the specified columns, which might be fewer than the total number of columns in the table. What to Observe in Execution: When you execute these queries: - For the first query, you'll see a result set that includes every column available in the Sales.Customers table. - For the second query, you'll see a more focused result set, showing only the columns explicitly listed in the query. This difference is important when you want to limit the data returned, especially if you're only interested in certain attributes of the customers and not the entire dataset.

4	In *the query* panel please type
	`SELECT *`
	`FROM`

5	then on the Object Explorer tab – Tables please find the Sales.Customers table. Click the table and drag it to the query pane l . The result is as shown below, after that add a semicolon after the name of the table in question and click execute.
	`SELECT *`
	`FROM`
	`[Sales].[Customers];`

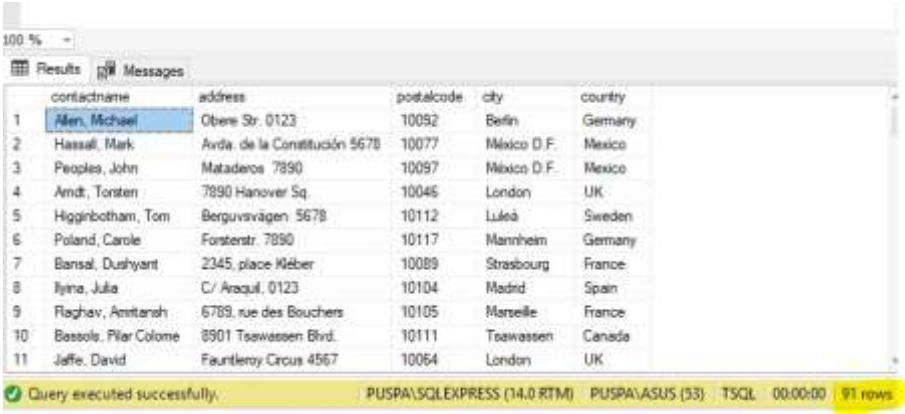
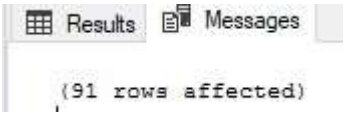
Practical –

Part 2: Using the SELECT statement for specific columns

Step	Information
1	In the query panel, please type the script below `SELECT contactname, address, postalcode, city, country FROM Sales.Customers;`
2	*Highlights query* above and click *execute*

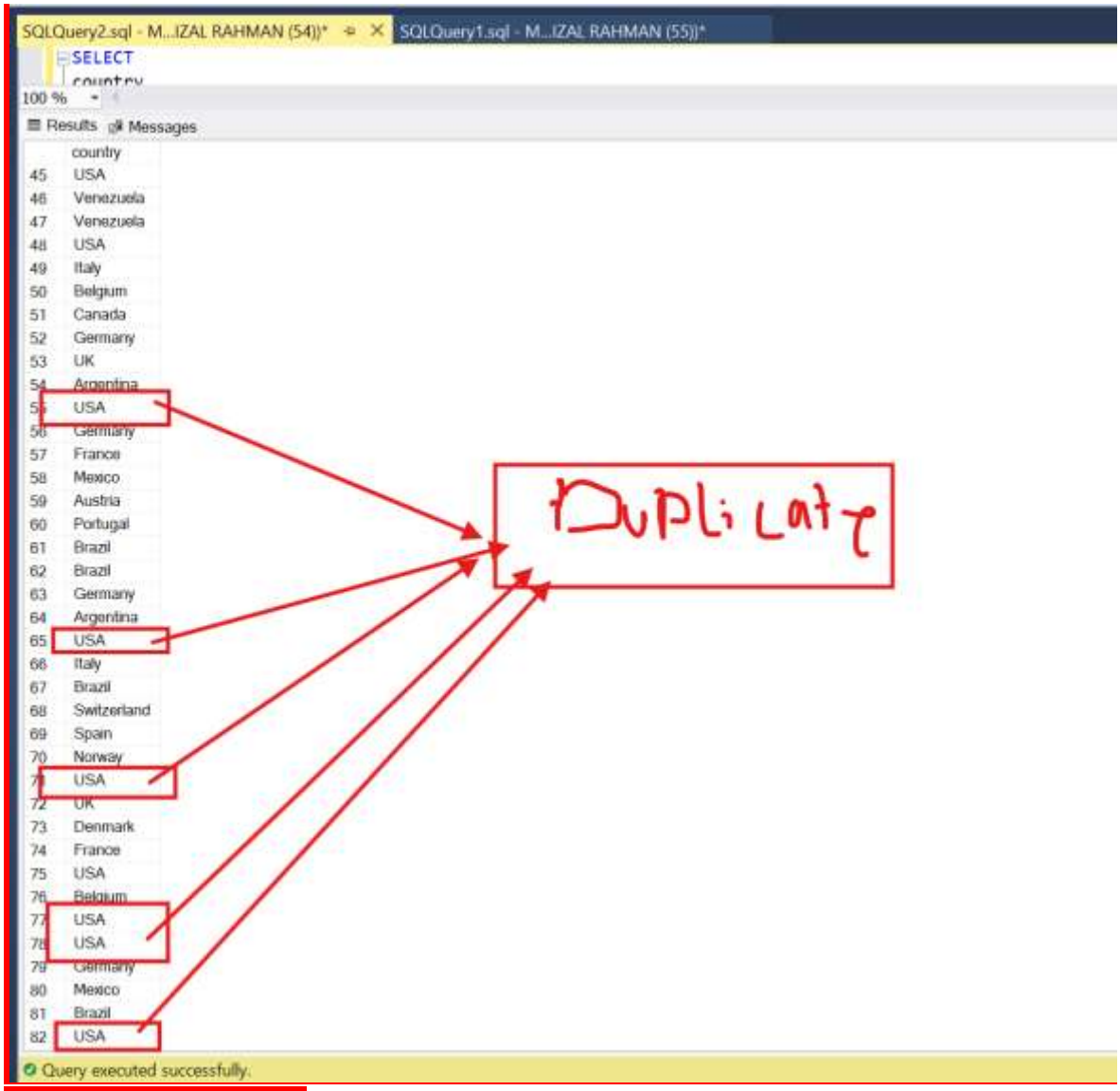
Practical –

3	Please observe the results. How many *rows* are produced? To find out, you can do it on the results tab as shown in the image below  Or you can also go to the messages tab as shown in the image below. 

Part 3: Using the SELECT statement to display data *uniquely / DISTINCT*

Step	Information
1	In *the query* panel, please type *the script* below `SELECT country FROM Sales.Customers;`
2	*Highlights query* above and click *execute*
3	Please observe the results. ***Is there any duplicate data? If YES, why? Capture the results of executing the SQL script above***

Practical –



The query you executed:

sql

```
SELECT country
FROM Sales.Customers;
```

returns a list of all the countries associated with the customers in the Sales.Customers table. The result you've provided shows the country for each customer record in the table.

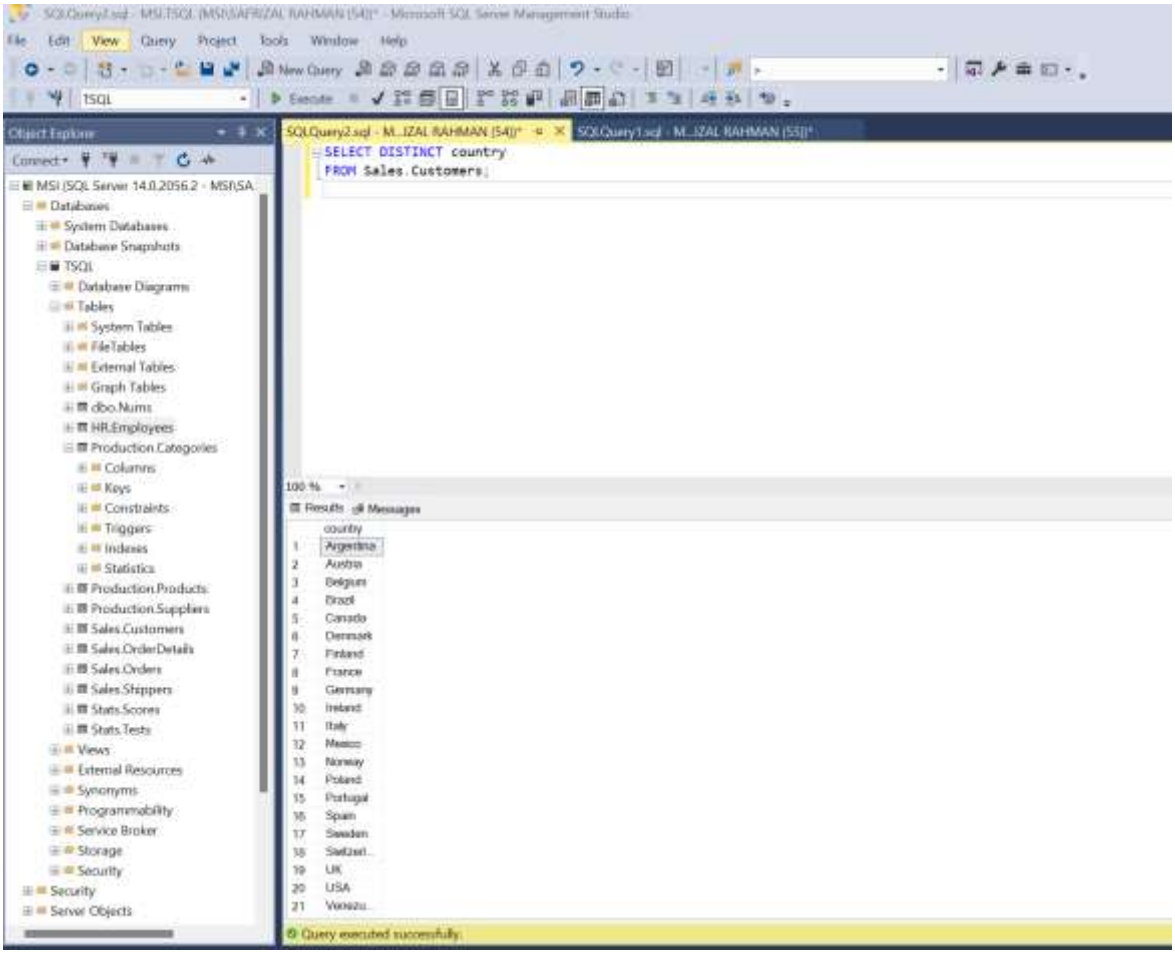
What the Query Does:

Practical –

	- SELECT country: This part specifies that you want to retrieve the country column. - FROM Sales.Customers;: This indicates the table from which you want to retrieve the data. Understanding the Output: - Each row in the result corresponds to a country value from a customer record. - The same country might appear multiple times because multiple customers can be from the same country. Additional Considerations: If you want to see a list of unique countries (i.e., each country appearing only once), you can modify the query using DISTINCT: sql SELECT DISTINCT country FROM Sales.Customers; This will return each country only once, regardless of how many customers are from that country. If you want to count how many customers are from each country, you could use: sql SELECT country, COUNT() as customer_count FROM Sales.Customers GROUP BY country; This query will return each country along with the number of customers from that country.

4	In *the query* pane, please type *the script* below.
	
	`SELECT DISTINCT`
	`  country`
	`FROM Sales.Customers;`
	
	Please click *execute* and observe the results.

Practical –

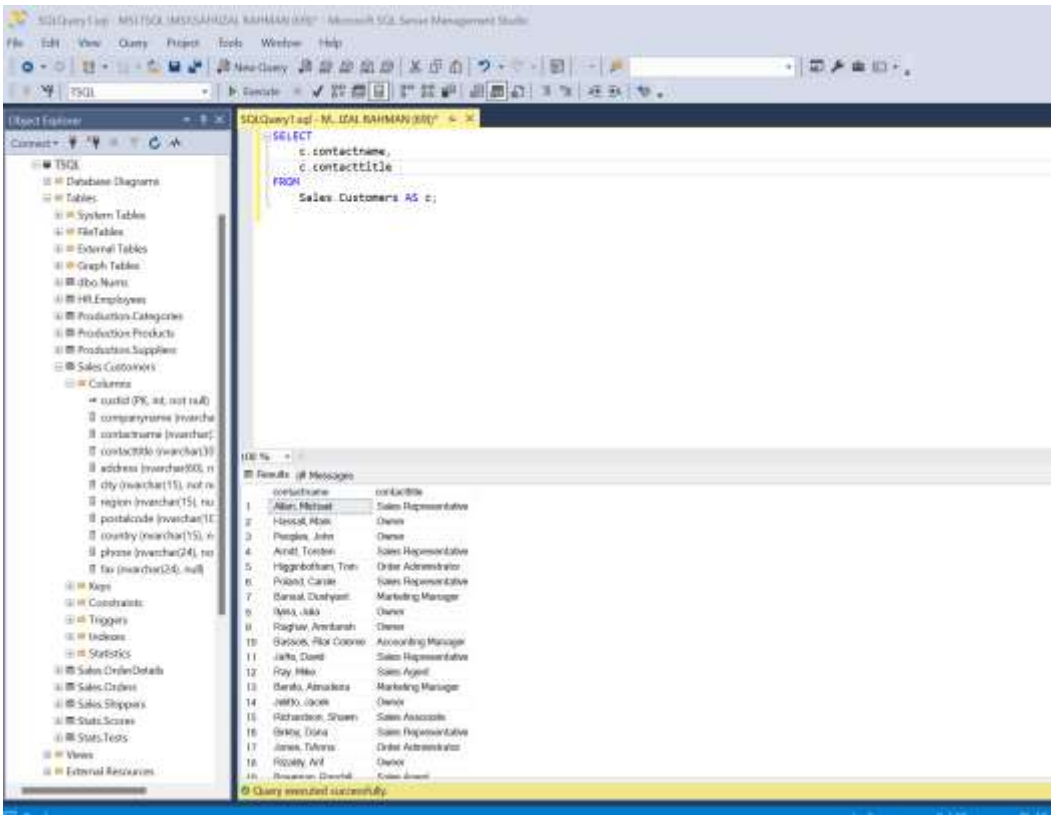
5	***Is there any duplicate data? Explain the difference in results in step 4 and step 3!? What are the benefits of the DISTINCT command? Capture the results of executing the SQL script above*** **(Question 3)** If we use (DISTINCT) returns the unique values of a specified column in the query results. Each row in the results corresponds to a country value from the customer data. The same country may appear multiple times because some customers may be from the same country. Therefore it can be concluded that distinct reduces duplication 

Practical –

Part 4: Using *ALIAS* for table names and column names

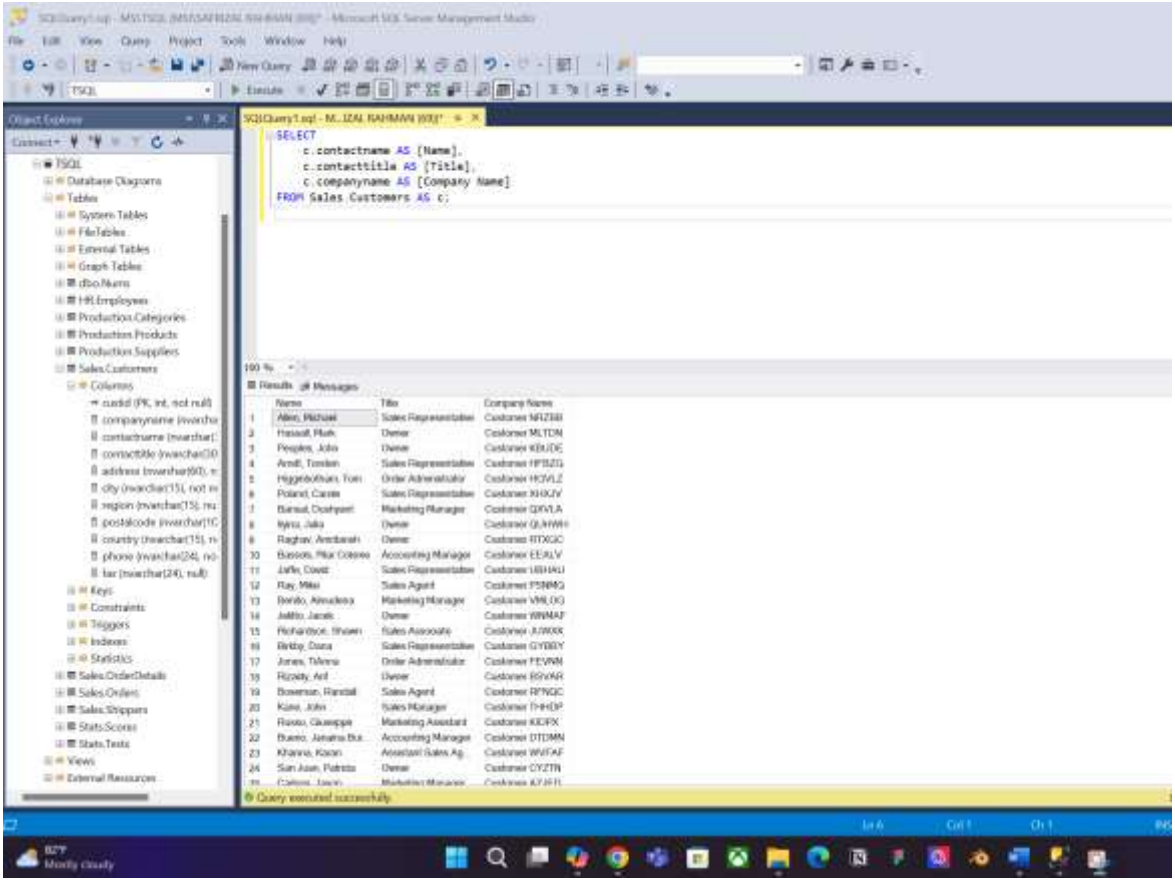
Step	Information

1	In *the query* panel, please type *the script* below
	`SELECT`
	`  c.contactname, c.contacttitle`
	`FROM Sales.Customers AS c;`

2	*Highlights query* above and click *execute.* Observe the results 

Practical –

3	In *the query* panel, please type *the script* below.
	`SELECT`
	`  c.contactname AS Name, c.contacttitle AS Title, c.companyname AS [Company Name]`
	`FROM Sales.Customers AS c;`

4	*Highlights query* above and click *execute* . Observe the results. 
5	***What is the difference between the execution results of the query stage 1 and stage 3 above? What are the benefits of the AS command? Please explain! Capture the results of the SQL script execution above*** **(Question 4)** Differences in Execution Results 1. Query Stage 1: sql

Practical –

```
SELECT
    c.contactname AS [Name],
    c.contacttitle AS [Title],
    c.companyname AS [Company Name]
FROM Sales.Customers AS c;
```

- Columns Selected: contactname, contacttitle, and companyname.
- Aliases Used: The columns are renamed to Name, Title, and Company Name respectively.
- Table Referenced: Sales.Customers.

Result Example:

```
| Name       | Title        | Company Name     |
|------------|--------------|------------------|
| John Doe   | Manager      | ABC Corp         |
| Jane Smith | CEO          | XYZ Inc          |
```

2. Query Stage 3:

```
sql
SELECT c.contactname, c.companyname
FROM Customers AS c;
```

- Columns Selected: contactname and companyname.
- Aliases Used: None.
- Table Referenced: Customers.

Result Example:

```
| contactname | companyname |
|-------------|-------------|
| John Doe    | ABC Corp    |
| Jane Smith  | XYZ Inc     |
```

Benefits of the AS Command

The AS command in SQL is used to create aliases for columns or tables. Here are some benefits:

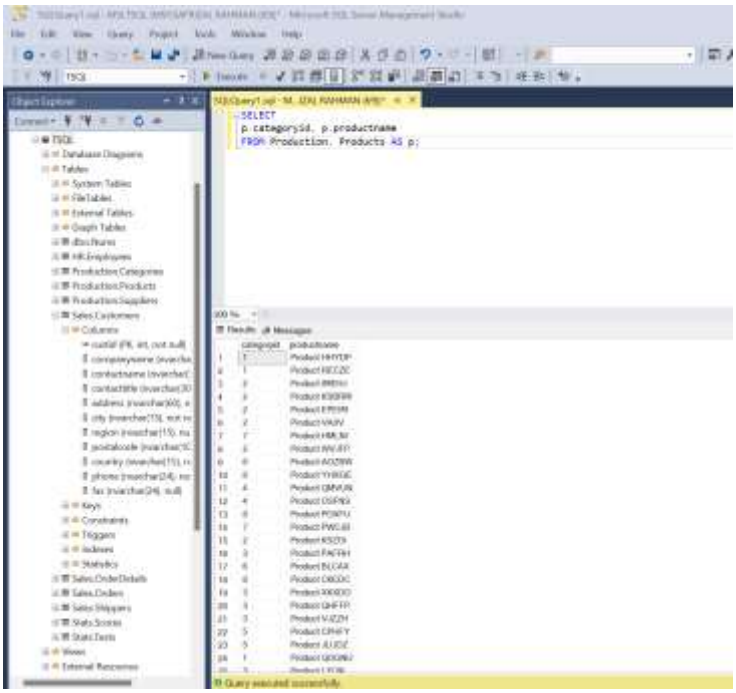
Practical –

	1. Readability: Aliases make the output more readable and understandable, especially when column names are long or not user-friendly. 2. Clarity: They help clarify the purpose of the columns in the result set, making it easier for others to understand the data. 3. Convenience: Aliases can simplify complex queries by providing shorter, more meaningful names. Capturing the Results Since I can't execute SQL queries directly, I recommend running the provided SQL scripts in your SQL environment to capture the actual results. Here are the scripts again for your reference: 1. Query Stage 1: sql SELECT c.contactname AS [Name], c.contacttitle AS [Title], c.companyname AS [Company Name] FROM Sales.Customers AS c; 2. Query Stage 3: sql SELECT c.contactname, c.companyname FROM Customers AS c;

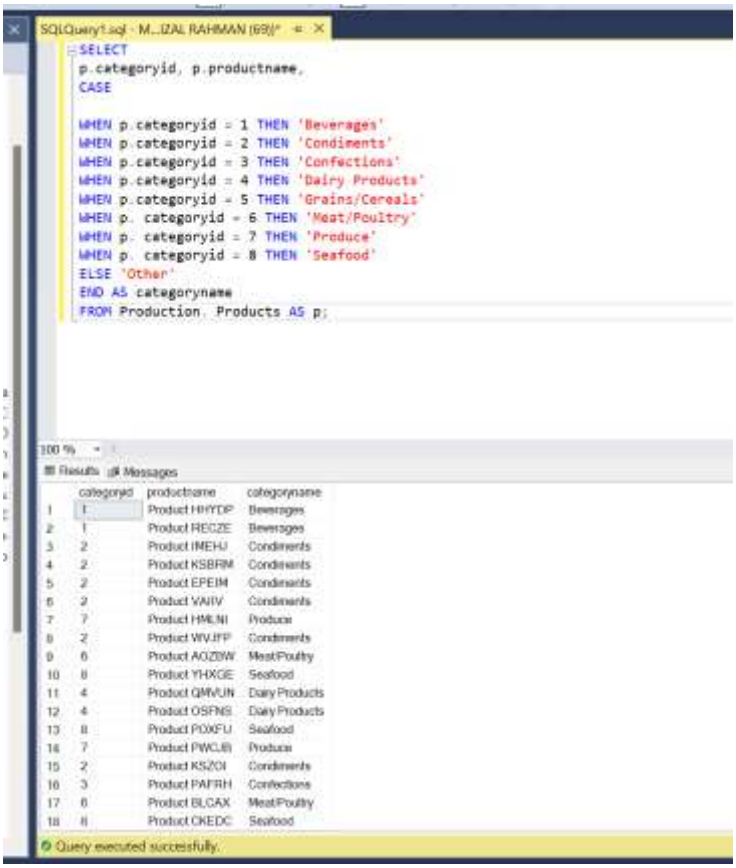
Practicum – Part 5: Use of CASE

Step	Information

1	In *the query* panel, please type *the script* below
	`SELECT`
	`  p.categoryid, p.productname`
	`FROM Production.Products AS p;`

2	*Highlights query* above and click *execute.* Observe the results 

3	In *the query* panel, please type *the script* below.
	`SELECT`
	`  p.categoryid, p.productname,`
	`  CASE`
	`  WHEN p.categoryid = 1 THEN 'Beverages'`
	`  WHEN p.categoryid = 2 THEN 'Condiments'`
	`  WHEN p.categoryid = 3 THEN 'Confections'`
	`  WHEN p.categoryid = 4 THEN 'Dairy Products'`
	`  WHEN p.categoryid = 5 THEN 'Grains/Cereals'`
	`  WHEN p.categoryid = 6 THEN 'Meat/Poultry'`
	`  WHEN p.categoryid = 7 THEN 'Produce'`
	`  WHEN p.categoryid = 8 THEN 'Seafood'`
	`  ELSE 'Other'`
	`  END AS categoryname`
	`FROM Production.Products AS p;`

4	*Highlights query* above and click *execute* . Observe the results.  SQLQuery1.sql - M...IZAL RAHMAN (69))* SELECT p.categoryid, p.productname, CASE WHEN p.categoryid = 1 THEN 'Beverages' WHEN p.categoryid = 2 THEN 'Condiments' WHEN p.categoryid = 3 THEN 'Confections' WHEN p.categoryid = 4 THEN 'Dairy Products' WHEN p.categoryid = 5 THEN 'Grains/Cereals' WHEN p. categoryid = 6 THEN 'Meat/Poultry' WHEN p. categoryid = 7 THEN 'Produce' WHEN p. categoryid = 8 THEN 'Seafood' ELSE 'Other' END AS categoryname FROM Production. Products AS p; 100 % Results Messages categoryid productname categoryname 1 1 Product HHYDP Beverages 2 1 Product REGZE Beverages 3 2 Product IMEHJ Condiments 4 2 Product KSBRM Condiments 5 2 Product EPEIM Condiments 6 2 Product VAIIV Condiments 7 7 Product HMLNI Produce 8 2 Product WVJFP Condiments 9 6 Product AQZBW Meat/Poultry 10 8 Product YHXGE Seafood 11 4 Product QMVUN Dairy Products 12 4 Product OSFNS Dairy Products 13 8 Product POXFU Seafood 14 7 Product PWCJB Produce 15 2 Product KSZOI Condiments 16 3 Product PAFRH Confections 17 6 Product BLCAX Meat/Poultry 18 8 Product CKEDC Seafood Query executed successfully.
5	***What is the difference between the execution results of the query stage 1 and stage 3 above? What are the benefits of the CASE command? Please explain! Capture the results of the SQL script execution above*** **Differences in Execution Results** 1. **Query Stage 1:** 2. SELECT 3. p.categoryid, p.productname 4. FROM Production.Products AS p; o **Columns Selected:** categoryid and productname. o **Aliases Used:** None. o **Table Referenced:** Production.Products. **Result Example:**

categoryid	productname
1	Chai
2	Aniseed Syrup

5. **Query Stage 3:**

```
SELECT
    p.categoryid, p.productname,
    CASE
        WHEN p.categoryid = 1 THEN 'Beverages'
        WHEN p.categoryid = 2 THEN 'Condiments'
        WHEN p.categoryid = 3 THEN 'Confections'
        WHEN p.categoryid = 4 THEN 'Dairy Products'
        WHEN p.categoryid = 5 THEN 'Grains/Cereals'
        WHEN p.categoryid = 6 THEN 'Meat/Poultry'
        WHEN p.categoryid = 7 THEN 'Produce'
        WHEN p.categoryid = 8 THEN 'Seafood'
        ELSE 'Other'
    END AS categoryname
FROM Production.Products AS p;
```

- **Columns Selected:** categoryid, productname, and a new column categoryname generated by the CASE statement.
- **Aliases Used:** categoryname for the result of the CASE statement.
- **Table Referenced:** Production.Products.

Result Example:

categoryid	productname	categoryname
1	Chai	Beverages
2	Aniseed Syrup	Condiments

Benefits of the CASE Command

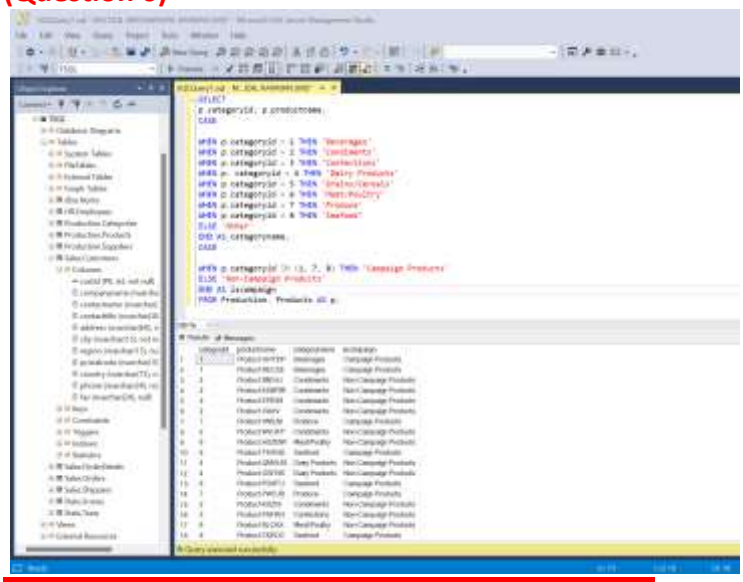
The CASE command in SQL is used to implement conditional logic within queries. Here are some benefits:

	1. **Conditional Logic:** It allows you to apply different conditions and return specific values based on those conditions, similar to an IF-THEN-ELSE statement in programming. 2. **Data Transformation:** You can transform data dynamically within your query, making it more readable and meaningful. 3. **Simplifies Complex Queries:** It helps in simplifying complex queries by avoiding multiple SELECT statements or JOIN operations. 4. **Flexibility:** It provides flexibility to handle various scenarios directly within the SQL query. **Capturing the Results** Since I can't execute SQL queries directly, I recommend running the provided SQL scripts in your SQL environment to capture the actual results. Here are the scripts again for your reference: 1. **Query Stage 1:** 2. SELECT 3. p.categoryid, p.productname 4. FROM Production.Products AS p; 5. **Query Stage 3:** 6. SELECT 7. p.categoryid, p.productname, 8. CASE 9. WHEN p.categoryid = 1 THEN 'Beverages' 10. WHEN p.categoryid = 2 THEN 'Condiments' 11. WHEN p.categoryid = 3 THEN 'Confections' 12. WHEN p.categoryid = 4 THEN 'Dairy Products' 13. WHEN p.categoryid = 5 THEN 'Grains/Cereals' 14. WHEN p.categoryid = 6 THEN 'Meat/Poultry' 15. WHEN p.categoryid = 7 THEN 'Produce' 16. WHEN p.categoryid = 8 THEN 'Seafood' 17. ELSE 'Other' 18. END AS categoryname 19. FROM Production.Products AS p;
6	In *the query* panel, please type *the script* below.

```
SELECT
   p.categoryid, p.productname,
   CASE
          WHEN p.categoryid = 1 THEN 'Beverages'
          WHEN p.categoryid = 2 THEN 'Condiments'
          WHEN p.categoryid = 3 THEN 'Confections'
          WHEN p.categoryid = 4 THEN 'Dairy Products'
          WHEN p.categoryid = 5 THEN 'Grains/Cereals'
          WHEN p.categoryid = 6 THEN 'Meat/Poultry'
          WHEN p.categoryid = 7 THEN 'Produce'
          WHEN p.categoryid = 8 THEN 'Seafood'
          ELSE 'Other'
   END AS categoryname,
   CASE
          WHEN p.categoryid IN (1, 7, 8) THEN 'Campaign Products'
          ELSE 'Non-Campaign Products'
   END AS iscampaign
FROM Production.Products AS p;
```

7

Please capture the results, what data is obtained from the query command above? Explain (Question 6)



Question 6: Capturing the Results and Explanation

The provided query categorizes products by their categoryid and indicates whether they are part of a "Campaign Product" or "Non-Campaign Product." The results obtained from the query include:

Product Category: Identified by the categoryid and translated into readable category names such as 'Beverages', 'Condiments', 'Seafood', etc.

Product Name: The name of the product.

Category Name: This is an alias column created using the CASE statement to map the categoryid to the actual category name.

Is Campaign: Another alias column that determines if a product is a "Campaign Product" based on its categoryid.

	Example Result Data: From the provided data, some rows might look like this: mathematica Copy code 1 Product HHYDP Beverages Campaign Products 2 Product IMEHJ Condiments Non-Campaign Products 8 Product YHXGE Seafood Campaign Products
8	Based on question number 6, please display data that is in the 'seafood' category only and use the *ALIAS command* to change the column name as shown in the image below. **Capture your SQL command and how many *rows* are produced**  **Question 7: Filtering Data for 'Seafood' Category Only and Renaming Columns** To filter only the 'Seafood' category and use the alias command to rename the columns, the SQL query would look like this: sql SELECT p.categoryid AS Category_ID, p.productname AS Product_Name, CASE WHEN p.categoryid = 8 THEN 'Seafood' END AS Category_Name, CASE WHEN p.categoryid IN (1, 7, 8) THEN 'Campaign Products' ELSE 'Non-Campaign Products' END AS Is_Campaign FROM Production.Products AS p WHERE

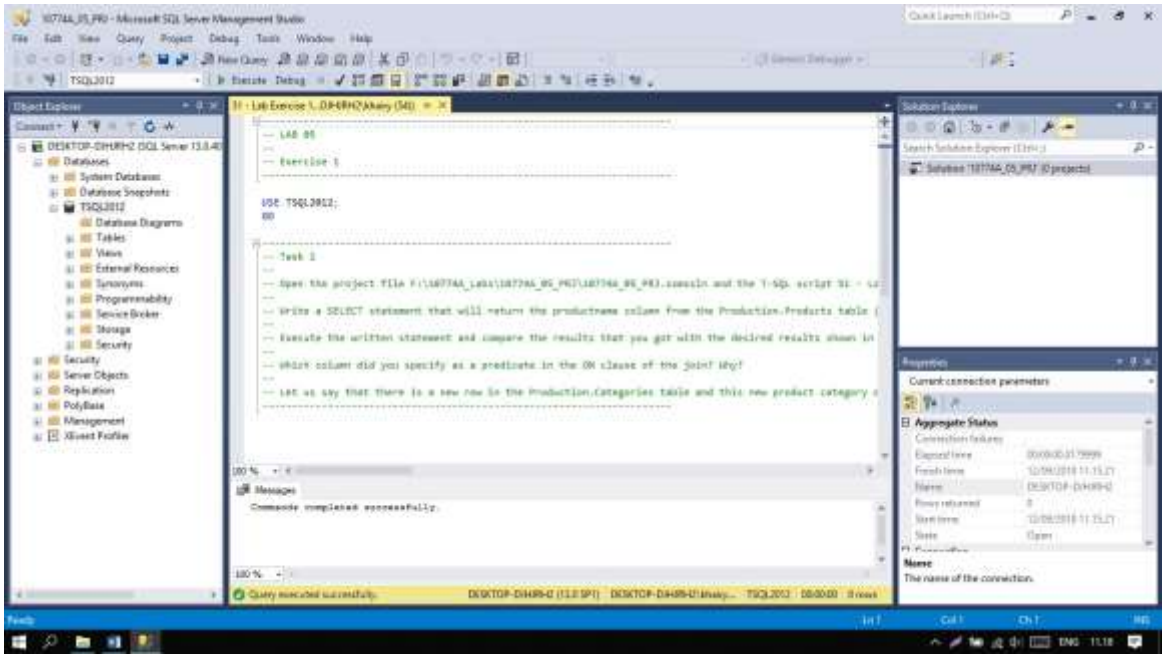
	ID_KATEGORI	NAMA_PRODUK	NAMA_KATEGORI	STATUS
1	8	Product ACRVI	Seafood	Campaign Products
2	8	Product AQOKR	Seafood	Campaign Products
3	8	Product CBRRL	Seafood	Campaign Products
4	8	Product CKEDC	Seafood	Campaign Products
5	8	Product EVFFA	Seafood	Campaign Products
6	8	Product GMKIJ	Seafood	Campaign Products
7	8	Product LYERX	Seafood	Campaign Products
8	8	Product POXFU	Seafood	Campaign Products
9	8	Product TTEEX	Seafood	Campaign Products

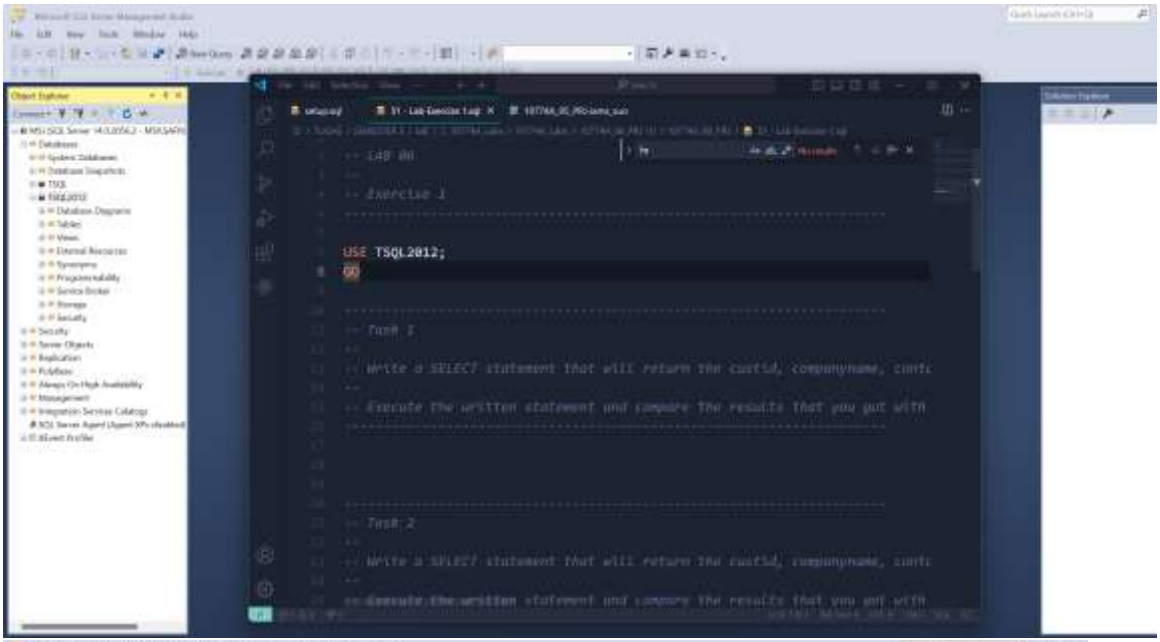
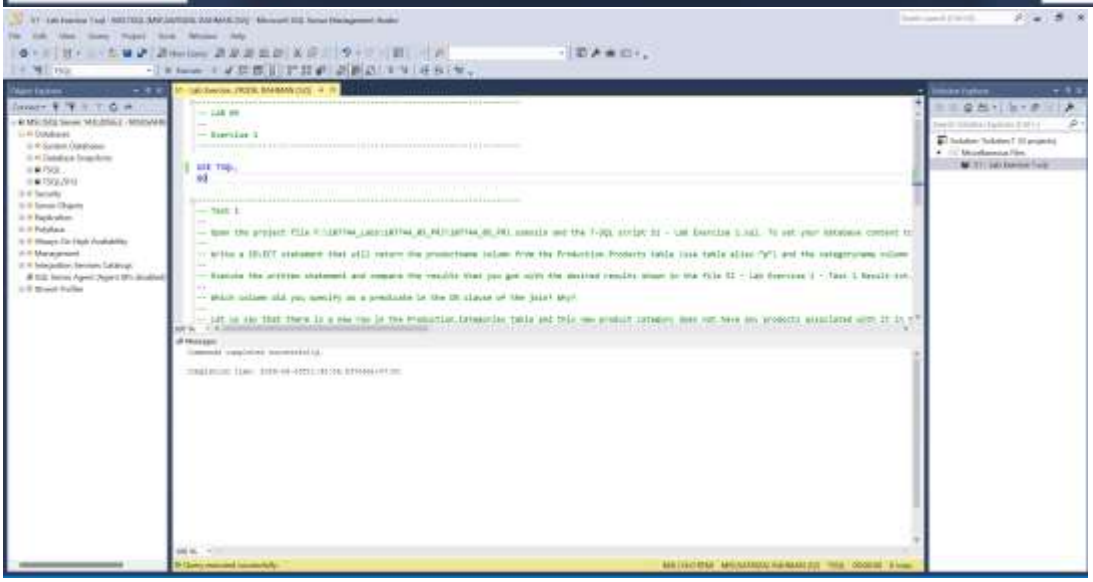
	p.categoryid = 8; **Explanation:** • **Filter**: The WHERE p.categoryid = 8 condition ensures that only products in the 'Seafood' category are selected. • **Alias**: The AS keyword is used to rename the columns to Category_ID, Product_Name, Category_Name, and Is_Campaign. **Rows Produced:** To determine the number of rows produced by this query, you would use the following: sql Copy code SELECT COUNT(*) AS Row_Count FROM Production.Products WHERE categoryid = 8; **Example Result:** Based on the provided data: mathematica Copy code 8 Product YHXGE Seafood Campaign Products 8 Product POXFU Seafood Campaign Products ... In your dataset, there are **12 rows** in the 'Seafood' category. **Captured Command and Output:** If this SQL query is executed, the result will display only the 'Seafood' products with renamed columns and indicate if they are part of a campaign.
9	Display employee data from HR.Employees table that comes from country 'USA' and city 'Seattle', use ALIAS command to change column name as shown below. Capture your SQL command (Question 8) 

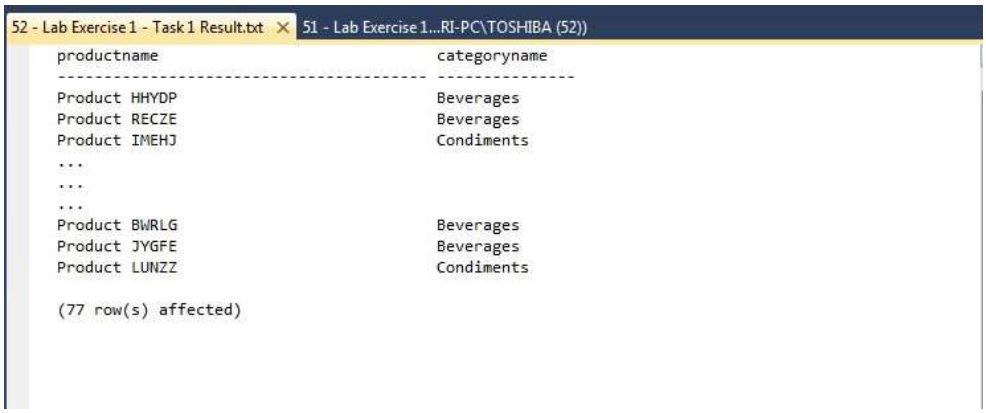
	FIRST_NAME	LAST_NAME	CITY	COUNTRY
1	Sara	Davis	Seattle	USA
2	Maria	Cameron	Seattle	USA

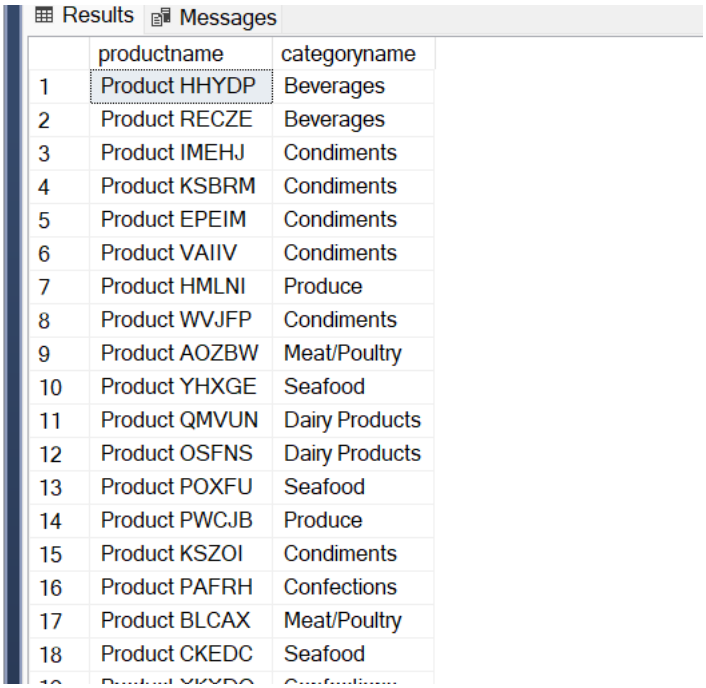
	

Practical – Part 6 : Creating an Inner Join Query

Step	Information
1	To experiment on this jobsheet, first log in to SQL Server Management Studio (SSMS). Then open the project \10774A Labs\10774A_05_PRJ\10774A_05_PRJ.ssmssln and the T-SQL script 51 - Lab Exercise 1.sql. Make sure the database is connected to " **TSQL** ".
	

	 
2	**[Question- 9]** Write a T-SQL SELECT that will display the productname column from the Production.Products table (use the alias table ''p'') and the categoryname column from the Production.Categories table (use the alias table ''c'') using inner join. SELECT p.productname, c.categoryname FROM Production.Products AS p INNER JOIN Production.Categories AS c ON p.categoryid = c.categoryid;

3	Compare the results in step 2 with the file 52 - Lab Exercise 1 - Task 1 Result.txt. If they are the same then the T-SQL you wrote is correct. 



Results Messages

	productname	categoryname
1	Product HHYDP	Beverages
2	Product RECZE	Beverages
3	Product IMEHJ	Condiments
4	Product KSBRM	Condiments
5	Product EPEIM	Condiments
6	Product VAIIV	Condiments
7	Product HMLNI	Produce
8	Product WVJFP	Condiments
9	Product AOZBW	Meat/Poultry
10	Product YHXGE	Seafood
11	Product QMVUN	Dairy Products
12	Product OSFNS	Dairy Products
13	Product POXFU	Seafood
14	Product PWCJB	Produce
15	Product KSZOI	Condiments
16	Product PAFRH	Confections
17	Product BLCAX	Meat/Poultry
18	Product CKEDC	Seafood

4

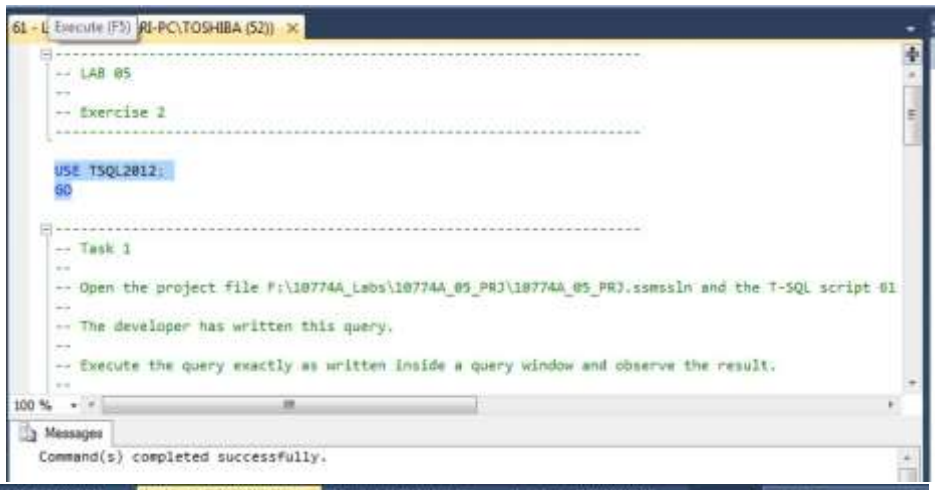
[Question- 10] Which column is specified as a predicate in the ON join clause? Why?

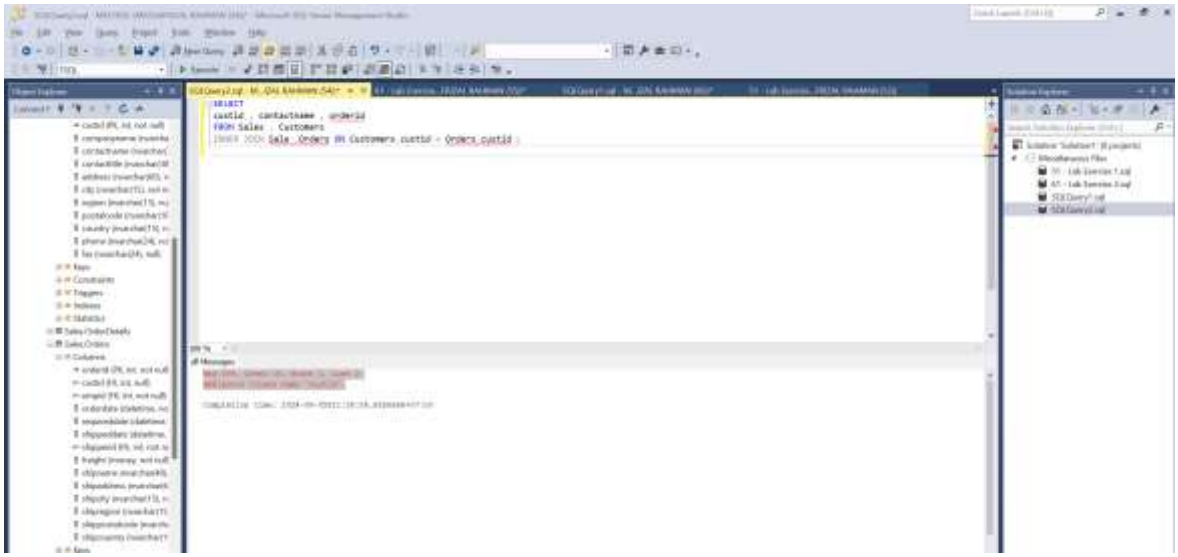
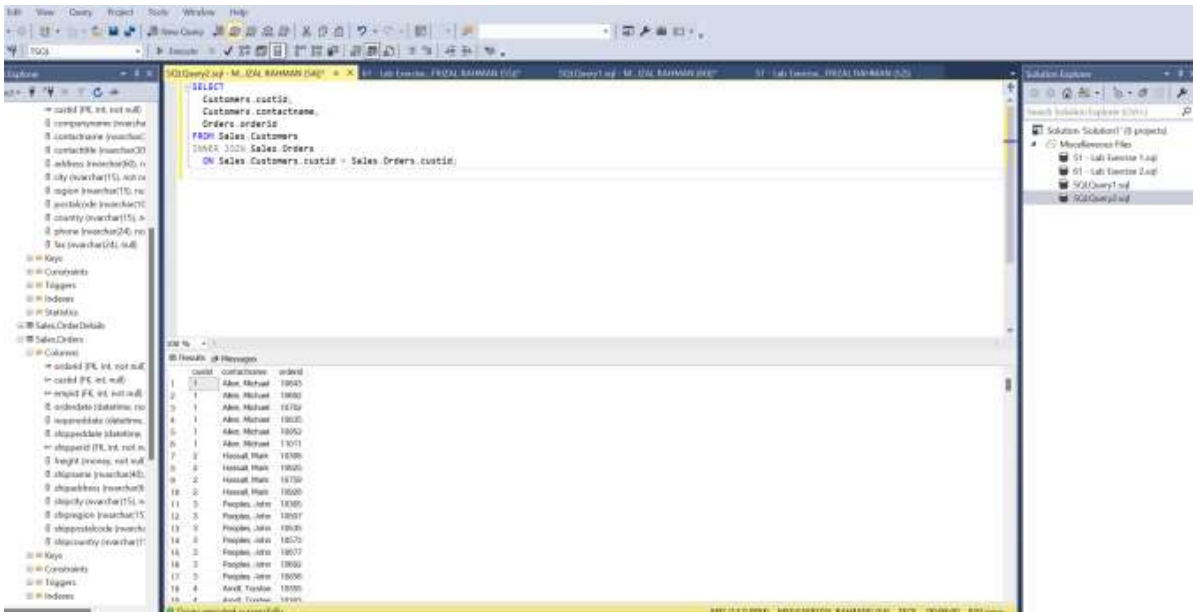
```
SELECT p.productname, c.categoryname
FROM Production.Products AS p
INNER JOIN Production.Categories AS c ON p.categoryid = c.categoryid;
```

Common Key: The categoryid column is a common key that exists in both the Production.Products table and the Production.Categories table.
Relationship: It represents the relationship between products and their categories. Each product has a categoryid that links it to a specific category in the Categories table.
Data Integrity: Using categoryid ensures that the join operation correctly matches each product with its corresponding category, maintaining data integrity and providing meaningful results.

5	**Conclusion :** After carrying out this part of the practicum, students know and understand how to perform an INNER JOIN on two tables.

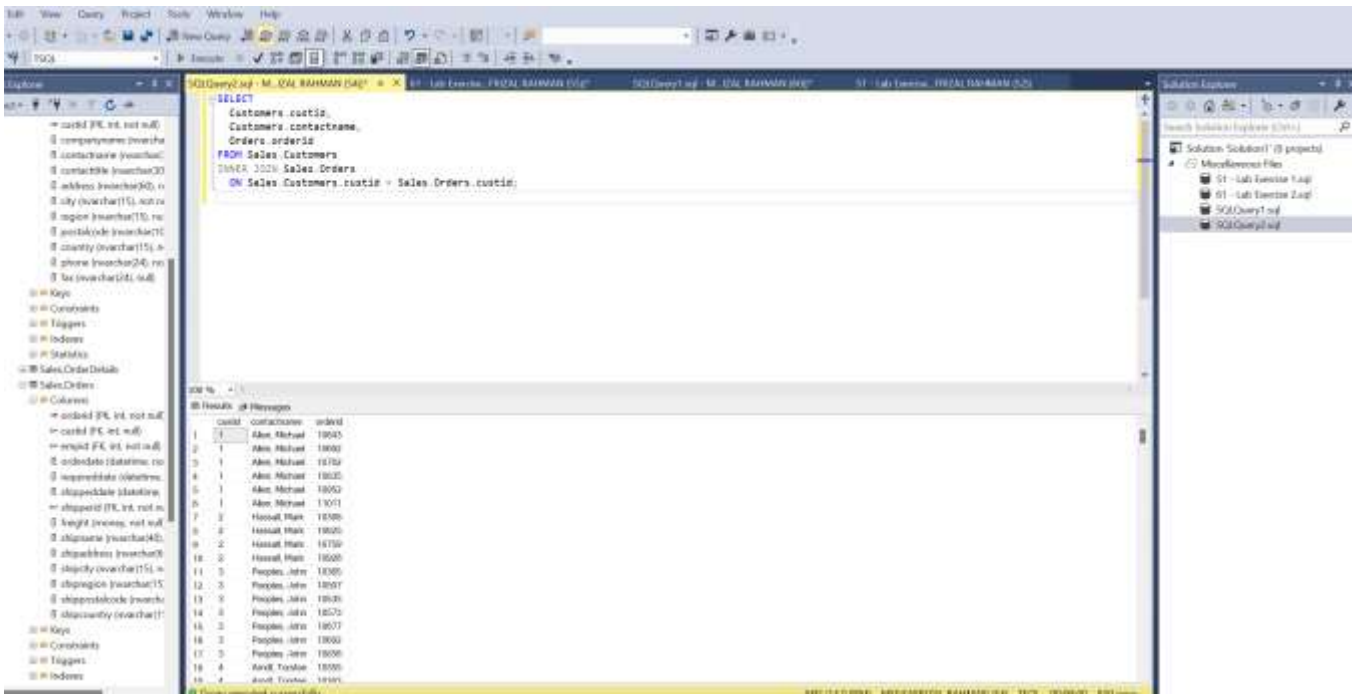
Practical – Part 7 : Creating an Inner Join Query on Multiple Tables

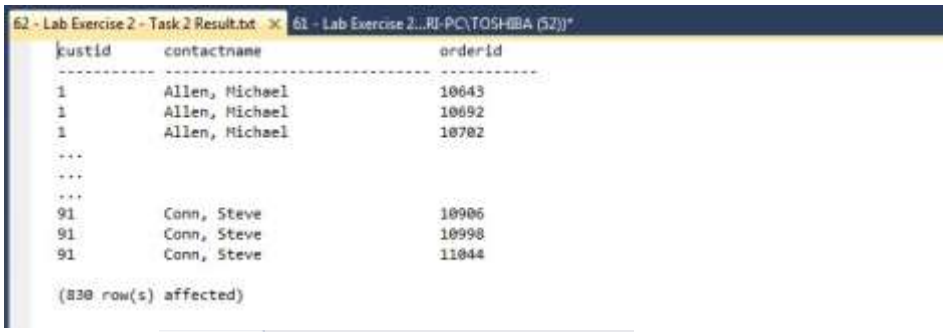
Step	Information
1	A *developer* will often be asked to run T-SQL files obtained from various departments . For example, the sales department wants a sales report of all customers for at least one order , with detailed information about each order. Then *the developer* will prepare the initialization of the SELECT statement to retrieve the custid and contactname columns in the Sales.Orders table. In accordance with the case study, this part 2 practicum will be carried out. Open the project \10774A Labs\10774A_05_PRJ\10774A_05_PRJ.ssmssln and the T-SQL script 61 - Lab Exercise 2.sql. Make sure the database is connected with “TSQL”.
	 

2	*The developer* will write T-SQL:
	
	`SELECT`
	`custid , contactname , orderid`
	`FROM Sales . Customers`
	`INNER JOIN Sales . Orders ON Customers . custid = Orders . custid ;`
	Execute the T-SQL , and observe the results!
	 
	
	NO eror
	 

3	**[Question- 11]** After the 2nd stage of the experiment is carried out, an error will appear. What is the content of the error message? Why can this error occur? Explain! The error message will be "Ambiguous column name 'custid'". This error happens because both the Sales.Customers and Sales.Orders tables have a custid column. SQL Server doesn't know which one to

Practical – Part

	use in the SELECT query since both tables are part of the JOIN. Thus, you need to explicitly reference the table from which you want to retrieve custid by using the table name or alias.
4	**[Question- 12]** In this 4th trial, fix the error that occurred in the 3rd stage trial which explains that all table names have their own table identities. In the 4th stage, you resolved the ambiguity in the custid column by explicitly referring to the table names in the SELECT query. SELECT Customers.custid, Customers.contactname, Orders.orderid FROM Sales.Customers INNER JOIN Sales.Orders ON Sales.Customers.custid = Sales.Orders.custid; 
5	Observe and compare the results of the 4th stage trial with the file 62 - Lab Exercise 2 - Task 2 Result.txt. If the results are the same, then your answer is correct.



```
custid      contactname                    orderid
----------- ------------------------------ -----------
1           Allen, Michael                 10643
1           Allen, Michael                 10692
1           Allen, Michael                 10702
...
...
...
91          Conn, Steve                    10906
91          Conn, Steve                    10998
91          Conn, Steve                    11044

(830 row(s) affected)
```

00 %

Results | Messages

	custid	contactname	orderid
1	1	Allen, Michael	10643
2	1	Allen, Michael	10692
3	1	Allen, Michael	10702
4	1	Allen, Michael	10835
5	1	Allen, Michael	10952
6	1	Allen, Michael	11011
7	2	Hassall, Mark	10308
8	2	Hassall, Mark	10625
9	2	Hassall, Mark	10759
10	2	Hassall, Mark	10926
11	3	Peoples, John	10365
12	3	Peoples, John	10507
13	3	Peoples, John	10535
14	3	Peoples, John	10573
15	3	Peoples, John	10677
16	3	Peoples, John	10682
17	3	Peoples, John	10856
18	4	Arndt, Torsten	10355
19	4	Arndt, Torsten	10383

Query executed successfully.

6

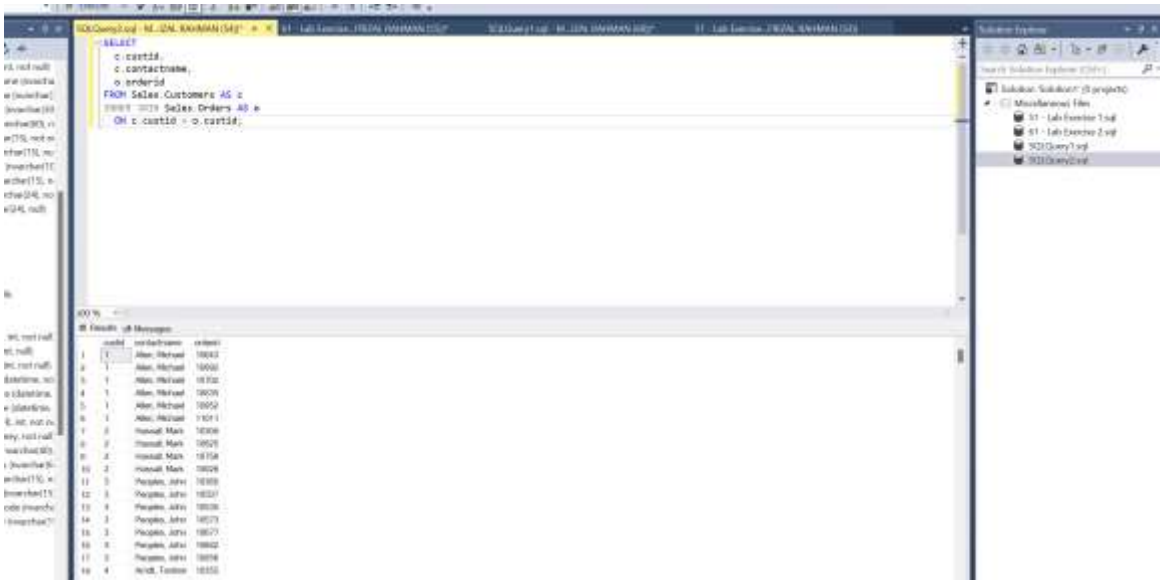
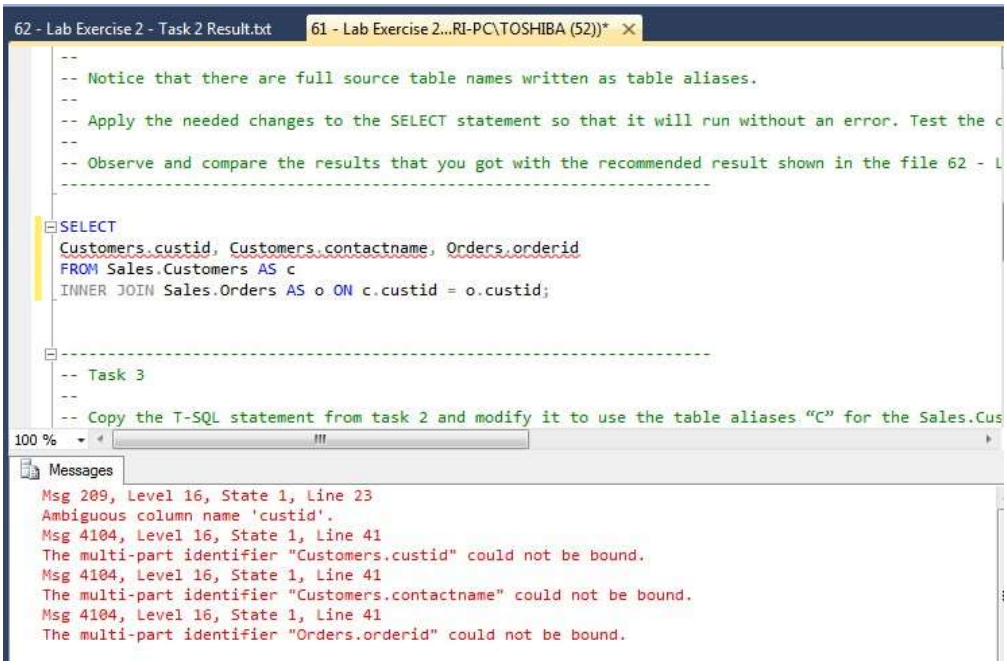
[Question- 13] Copy the T-SQL in the 4th stage of the test and modify it by using the alias table '' c '' to
Sales.Custumers table and '' o '' for Sales.Orders table.

sql
Copy code

```
SELECT
  c.custid,
  c.contactname,
  o.orderid
FROM Sales.Customers AS c
INNER JOIN Sales.Orders AS o
  ON c.custid = o.custid;
```

This query now uses table aliases c for Sales.Customers and o for Sales.Orders, which makes the query more concise and easier to read.

Practical – Part

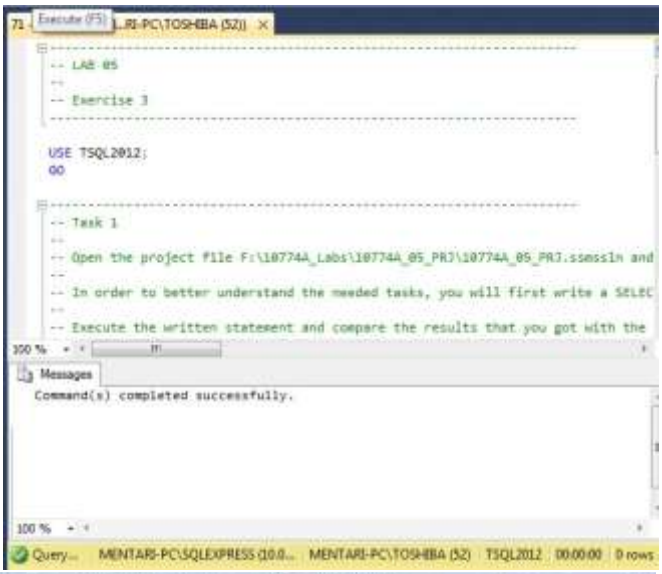
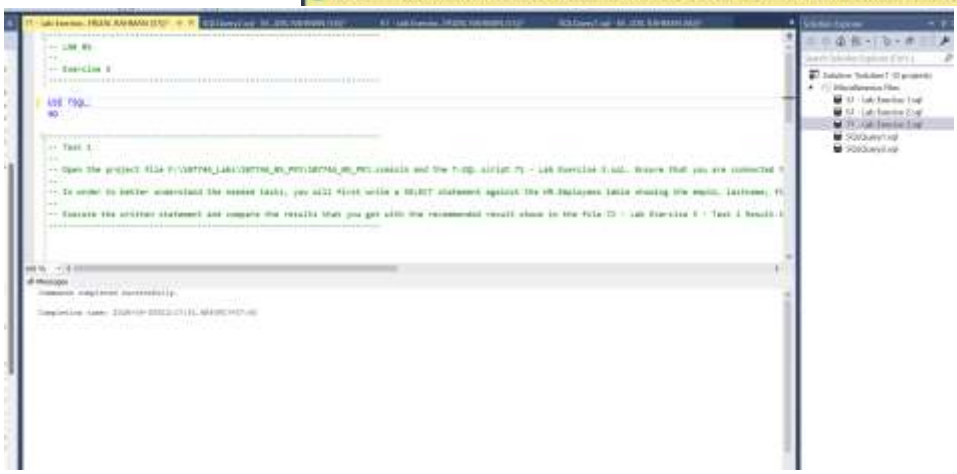
	
7	Execute T-SQL on stage-6 test and compare the result with the result of stage 4 execution! If the result is the same then your T-SQL is correct.
8	Change the column prefix in the SELECT clause to the full name, then execute the T-SQL! 
9	**[Question- 14]** Why does the execution result of T-SQL stage 8 produce an error? 1. **Ambiguous Column Name:** o Error message **"Ambiguous column name 'custid'"** suggests that the column custid exists in both Sales.Customers and Sales.Orders tables, and without further clarification, SQL does not know which table you are referring to.

	2. **Multi-part Identifier Error:** ○ The error **"The multi-part identifier 'Customers.custid' could not be bound"** means that SQL Server cannot resolve the reference to the Customers.custid, Customers.contactname, and Orders.orderid because the table Customers and Orders are aliased as c and o, respectively. **To fix this:** 1. **Change the column references in the SELECT clause** to use the table aliases c and o for Customers and Orders, as you did in the FROM clause. You should update your query as follows: sql Copy code SELECT c.custid, c.contactname, o.orderid FROM Sales.Customers AS c INNER JOIN Sales.Orders AS o ON c.custid = o.custid; 2. This query should execute without any errors because the column references in the SELECT clause now correctly correspond to the aliases provided in the FROM clause. The error occurred because the columns were referenced with full table names (Customers and Orders) instead of their aliases (c and o) defined in the FROM clause. SQL Server could not bind those full table names as no such references existed in the query.
10	**[Question- 15]** Change the column name prefix in the T-SQL test step 8 with its alias name, then display the execution results! You need to replace the full table names with their corresponding aliases (as shown in the corrected query above), and then re-run the query to get the expected result.
11	**Conclusion** : After carrying out this part of the practicum, you should now know and understand the importance of using table alias names and how to JOIN multiple tables (more than two tables).

8 : Creating a Self-Join Query

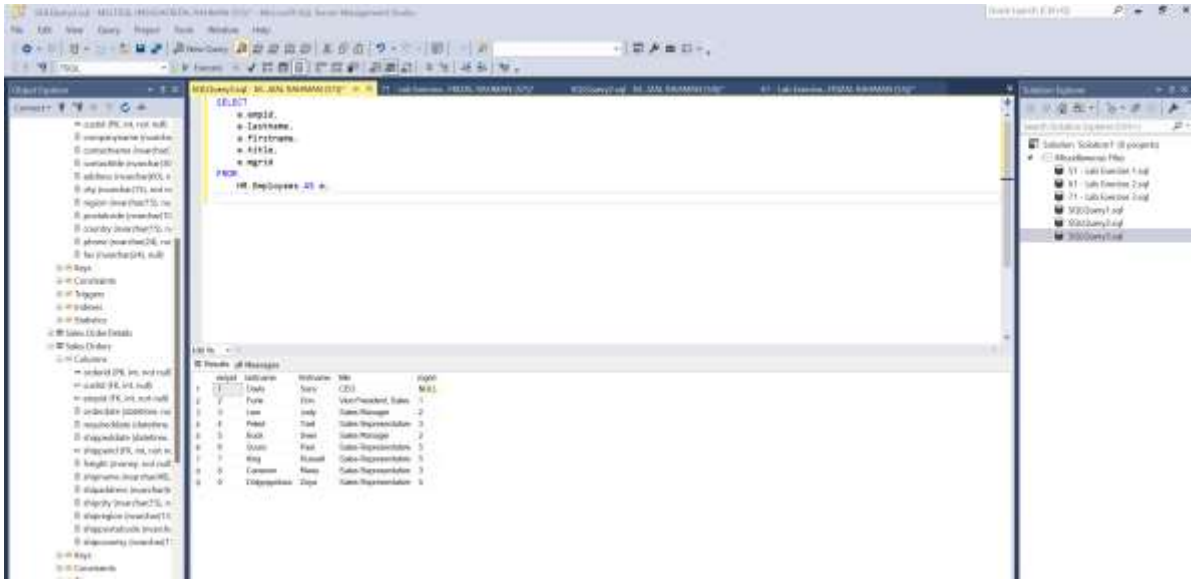
Step	Information

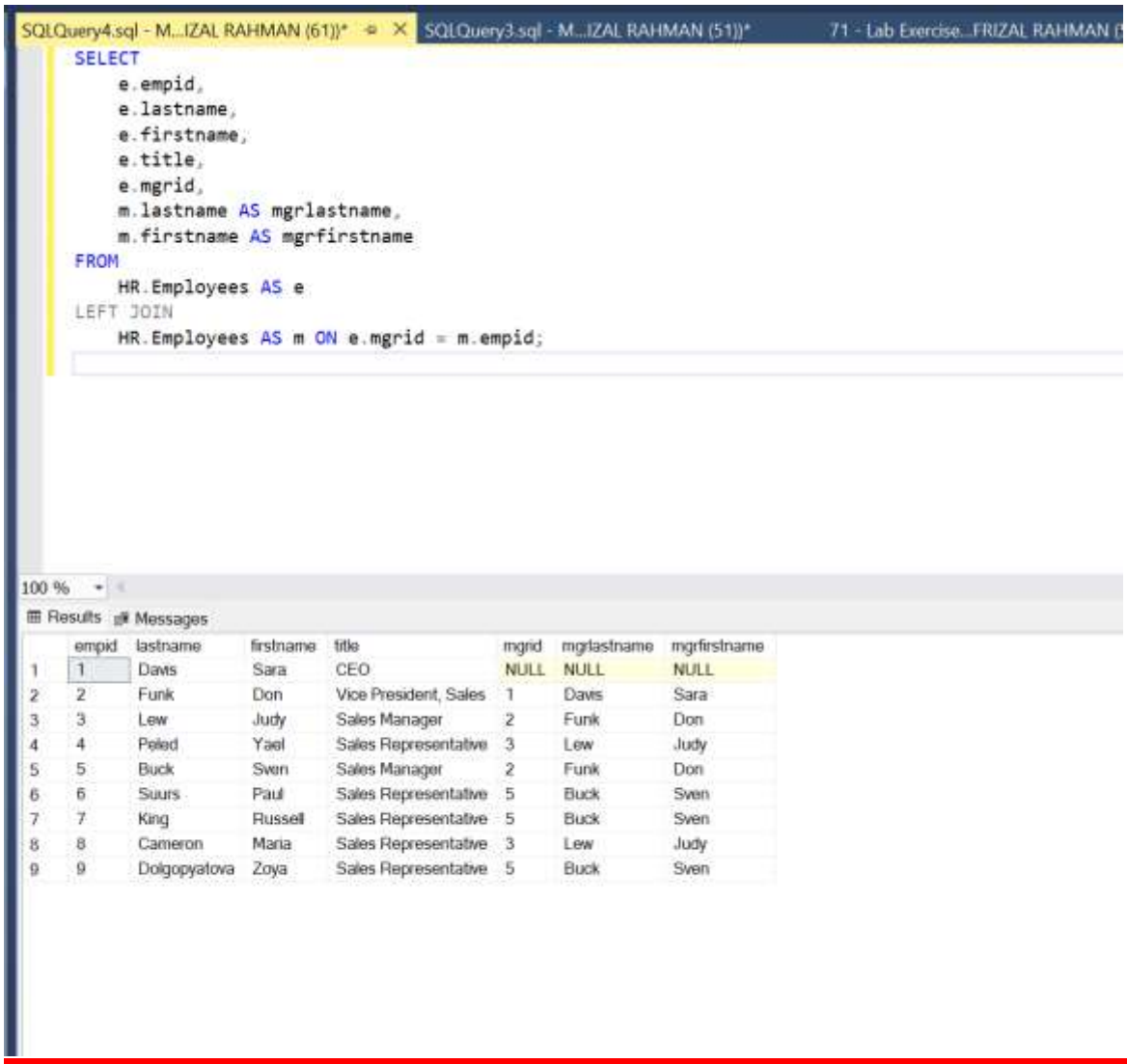
Practical – Part

	This practicum uses a case study in an HR department that wants to display reports on employees and managers. Some of the things that want to be displayed are the lastname, firstname, and title columns of the HR.Employees table for employees and managers. Open the project \10774A Labs\10774A_05_PRJ\10774A_05_PRJ.ssmssln and the T-SQL script 71 - Lab Exercise 3.sql. Make sure the database is connected with "TSQL".
1	 
2	**[Question- 16]** Write T-SQL using SELECT clause to display empid, lastname, firstname, title, and mgrid columns. on the table HR.Employees by giving the alias name "e" for the HR.Employees table. To display the columns empid, lastname, firstname, title, and mgrid from the HR.Employees table using the alias e, you can write the T-SQL SELECT query as follows: sql Copy code SELECT e.empid, e.lastname, e.firstname,

	e.title, e.mgrid FROM HR.Employees AS e; This query will retrieve the specified columns from the HR.Employees table with the alias e for the table, making the query easier to read, especially when joining multiple tables or working with large datasets. 
3	**[Question- 17]** Execute the 2nd stage of the test and compare it with 72 - Lab Exercise 3 - Task 1 Result.txt . If the results are the same, then your test is correct. SELECT e.empid, e.lastname, e.firstname, e.title, e.mgrid FROM HR.Employees AS e;

Practical – Part

	
4	**[Question- 18]** Copy the T-SQL in step 2 then modify it by adding columns about manager information, namely lastname, firstname using SELF-JOIN. Use the aliases mgrlastname and mgrfirstname to distinguish the names of managers and employees. SELECT e.empid, e.lastname, e.firstname, e.title, e.mgrid, m.lastname AS mgrlastname, m.firstname AS mgrfirstname FROM HR.Employees AS e LEFT JOIN HR.Employees AS m ON e.mgrid = m.empid;



	empid	lastname	firstname	title	mgrid	mgrlastname	mgrfirstname
1	1	Davis	Sara	CEO	NULL	NULL	NULL
2	2	Funk	Don	Vice President, Sales	1	Davis	Sara
3	3	Lew	Judy	Sales Manager	2	Funk	Don
4	4	Peled	Yael	Sales Representative	3	Lew	Judy
5	5	Buck	Sven	Sales Manager	2	Funk	Don
6	6	Suurs	Paul	Sales Representative	5	Buck	Sven
7	7	King	Russell	Sales Representative	5	Buck	Sven
8	8	Cameron	Maria	Sales Representative	3	Lew	Judy
9	9	Dolgopyatova	Zoya	Sales Representative	5	Buck	Sven

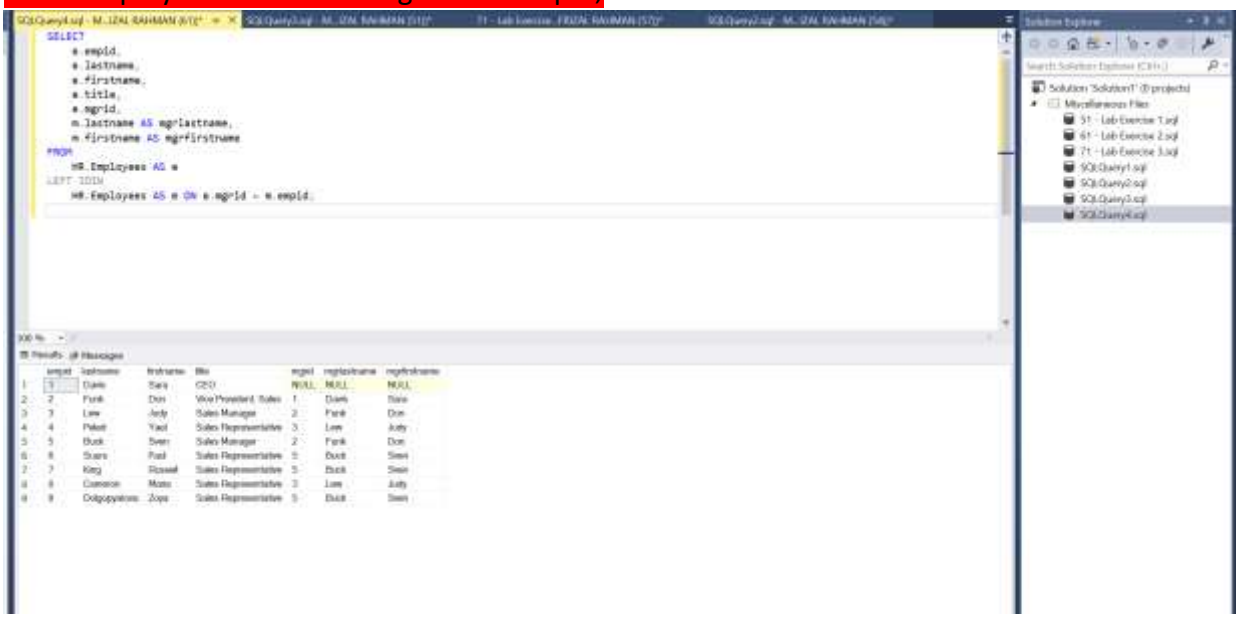
This query will display the employee's data along with the manager's name (mgrlastname and mgrfirstname). It uses a **LEFT JOIN** because some employees (like the CEO) may not have a manager (mgrid is NULL).

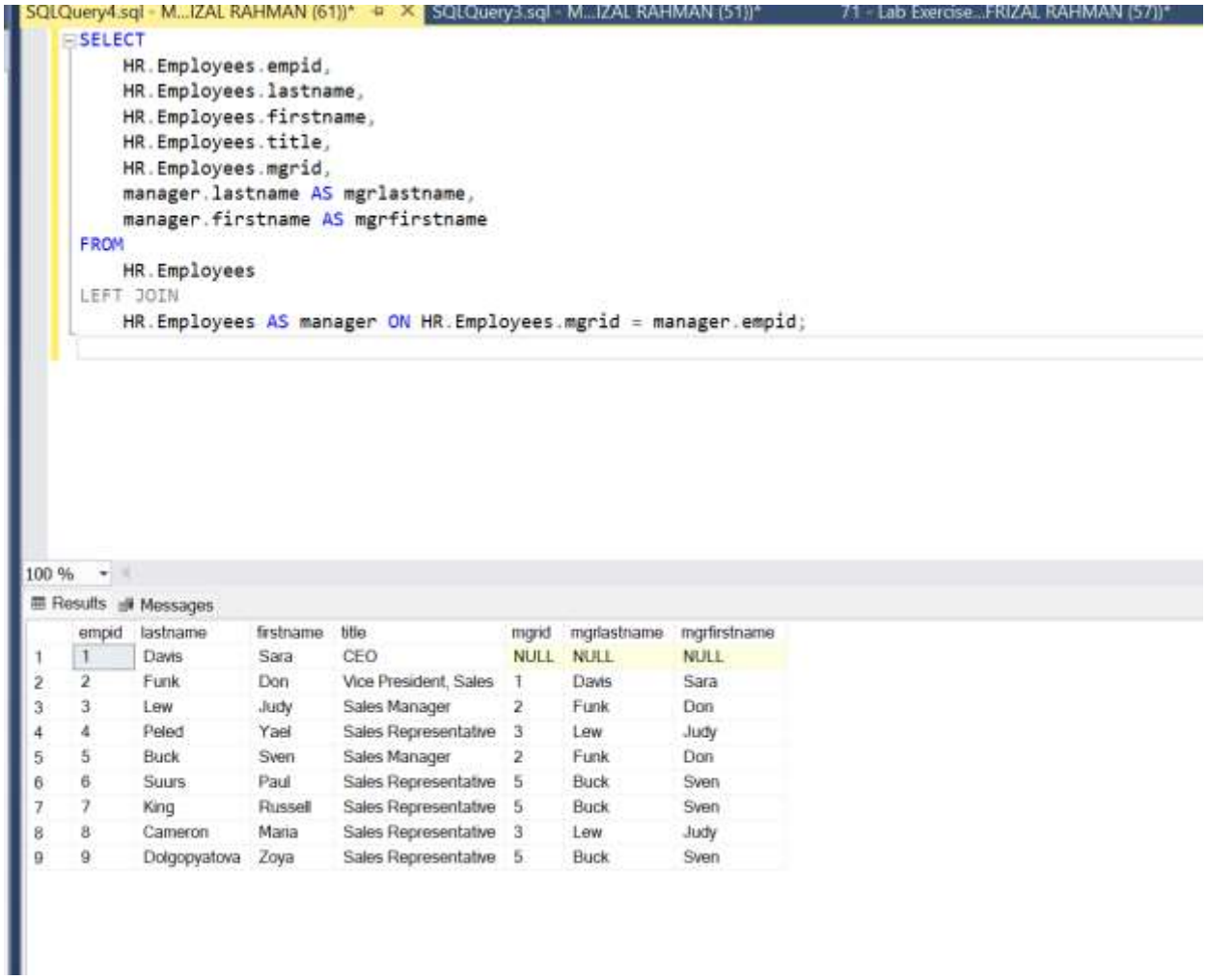
5

Execute the 2nd stage of the test and compare it with 7 3 - Lab Exercise 3 - Task 2 Result.txt . If the results are the same, then your test is correct.

```
SELECT
   e.empid,
   e.lastname,
   e.firstname,
   e.title,
   e.mgrid,
   m.lastname AS mgrlastname,
   m.firstname AS mgrfirstname
FROM
   HR.Employees AS e
LEFT JOIN
```

Practical – Part

	HR.Employees AS m ON e.mgrid = m.empid; 
6	[Question- 20] Is it mandatory to write the table alias name when executing the SELF-JOIN command? Can the original table name be used as an alias name? Explain! **Is it mandatory to use an alias in a SELF-JOIN?** • **No, it is not mandatory** to use a table alias in a SELF-JOIN, but it is highly recommended to do so. When you join a table to itself, using aliases helps distinguish between the different instances of the same table, making your query more readable and understandable. **Can the original table name be used as an alias?** • **Yes**, the original table name can be used as an alias, but it's uncommon and can lead to confusion. Using a different alias (like e for employees and m for managers) is more readable and makes it clear which instance of the table you're referring to in the query.



```
SELECT
    HR.Employees.empid,
    HR.Employees.lastname,
    HR.Employees.firstname,
    HR.Employees.title,
    HR.Employees.mgrid,
    manager.lastname AS mgrlastname,
    manager.firstname AS mgrfirstname
FROM
    HR.Employees
LEFT JOIN
    HR.Employees AS manager ON HR.Employees.mgrid = manager.empid;
```

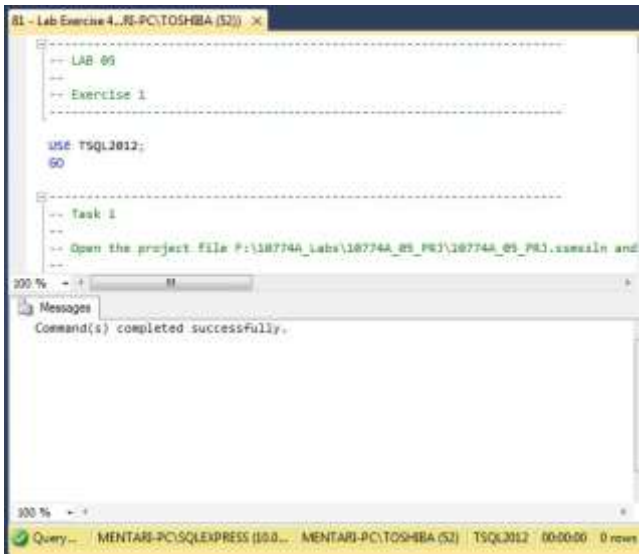
100 %

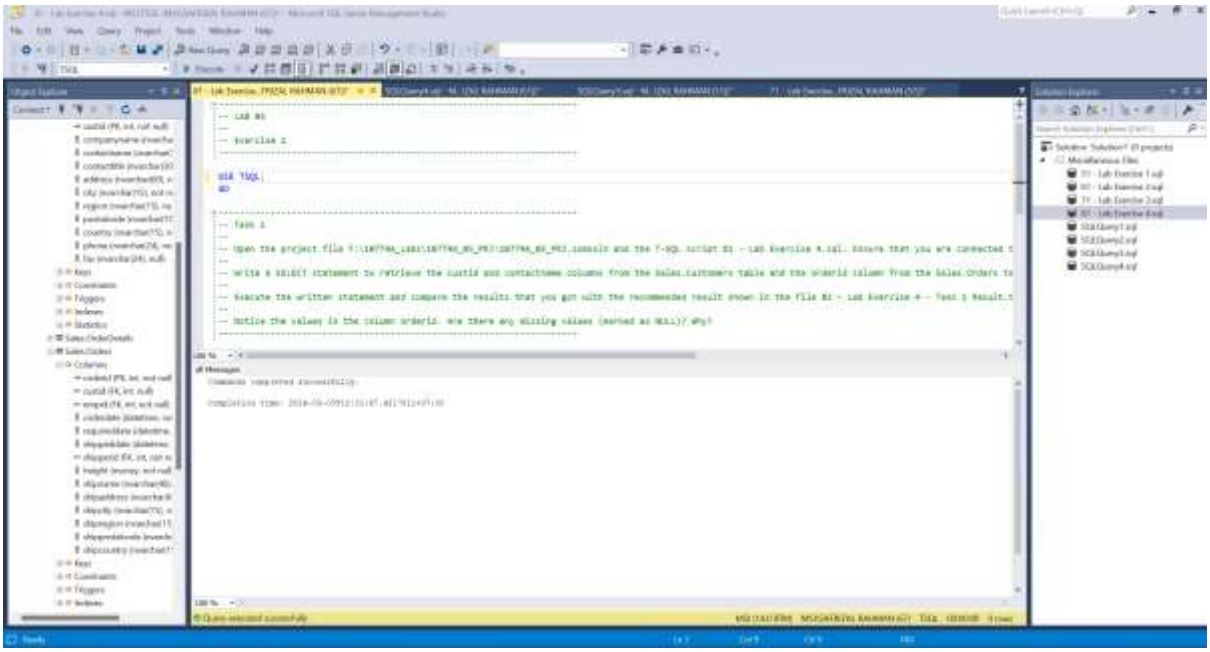
Results | Messages

	empid	lastname	firstname	title	mgrid	mgrlastname	mgrfirstname
1	1	Davis	Sara	CEO	NULL	NULL	NULL
2	2	Funk	Don	Vice President, Sales	1	Davis	Sara
3	3	Lew	Judy	Sales Manager	2	Funk	Don
4	4	Peled	Yael	Sales Representative	3	Lew	Judy
5	5	Buck	Sven	Sales Manager	2	Funk	Don
6	6	Suurs	Paul	Sales Representative	5	Buck	Sven
7	7	King	Russell	Sales Representative	5	Buck	Sven
8	8	Cameron	Maria	Sales Representative	3	Lew	Judy
9	9	Dolgopyatova	Zoya	Sales Representative	5	Buck	Sven

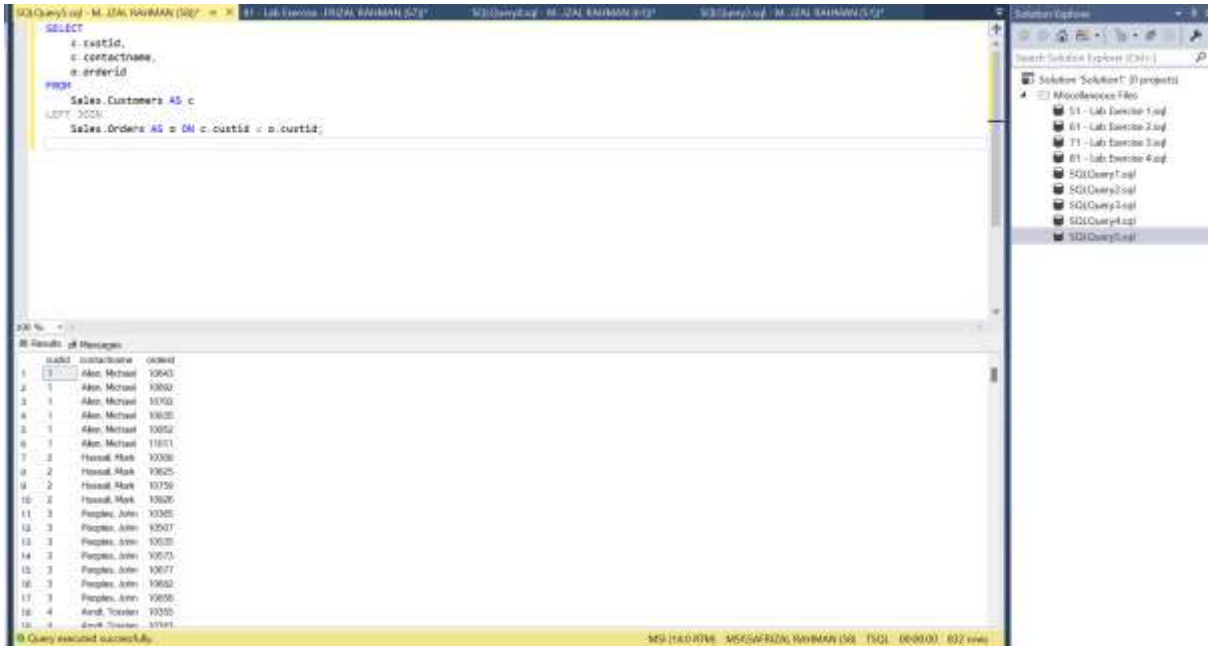
7	**Conclusion** : After doing this part of the practicum, you should understand how to write a T-SQL SELF-JOIN statement.

Practical – Part 9 : Creating Outer-Join Query

Step	Information
1	The case study used in this practicum part 4 continues the practicum in part 3. The sales department is quite satisfied with the report that has been made. Then the sales department wants to change the report to show all customers, even though the customer does not have an order history or customers who have an order history. Therefore, a SELECT clause is needed to retrieve all rows from the Sales.Customers table (custid and contactname columns) and the orderid column From the Sales.Orders table. Open the project \10774A Labs\10774A_05_PRJ\10774A_05_PRJ.ssmssln and the T-SQL script 81 - Lab Exercise 4.sql. Make sure the database is connected with “TSQL”.
	

	
2	**[Question- 21]** Write a T-SQL command with a SELECT clause to retrieve the custid and contactname columns from the table Sales.Customers and the orderid column from the Sales.Orders table . The command created must retrieve all rows from the Sales.Customers table . To retrieve all rows from the Sales.Customers table, including those customers who might not have any orders, you should use a **LEFT JOIN**. This ensures that even customers without orders (where orderid might be NULL) are included in the result. Here's the T-SQL command: sql Copy code SELECT c.custid, c.contactname, o.orderid FROM Sales.Customers AS c LEFT JOIN Sales.Orders AS o ON c.custid = o.custid; • **c**: Alias for Sales.Customers table. • **o**: Alias for Sales.Orders table. • **LEFT JOIN**: Ensures that all rows from the Sales.Customers table are retrieved, even if there are no corresponding orderid values in Sales.Orders.

Practical – Part

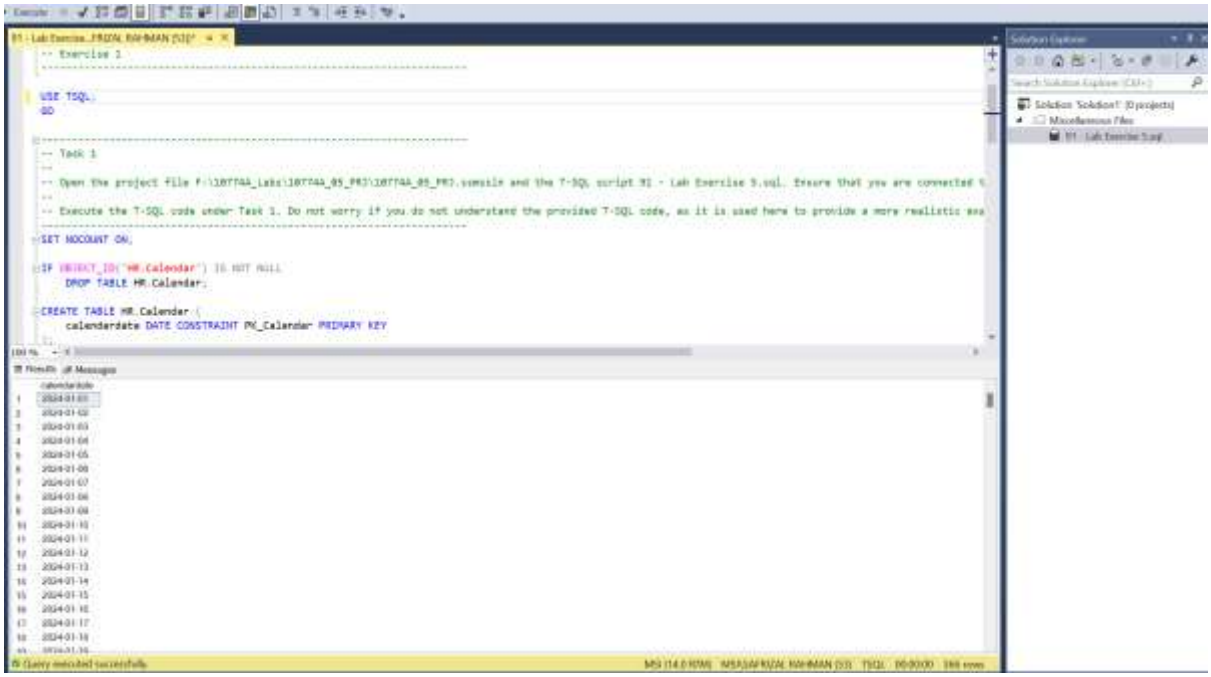
	
3	**[Question- 22]** Execute the 2nd stage of the test and compare it with 82 - Lab Exercise 4 - Task 1 Result.txt . If the results are the same, then your test is correct. 
4	**[Question- 23]** Pay attention to the values in the orderid column . Are there any missing values (NULL)? Why? **Are there any missing values (NULL)?** • **Yes**, there might be NULL values in the orderid column. These would occur for customers who have not placed any orders. **Why are there NULL values?**

	• The query uses a **LEFT JOIN** between Sales.Customers and Sales.Orders. A **LEFT JOIN** retrieves all rows from the left table (Sales.Customers) and matches them with the right table (Sales.Orders). If a customer does not have any corresponding orders, the orderid column will be NULL for that customer.
5	**Conclusion** : After doing this part of the practicum, you should understand how to write the TSQL OUTER-JOIN statement .

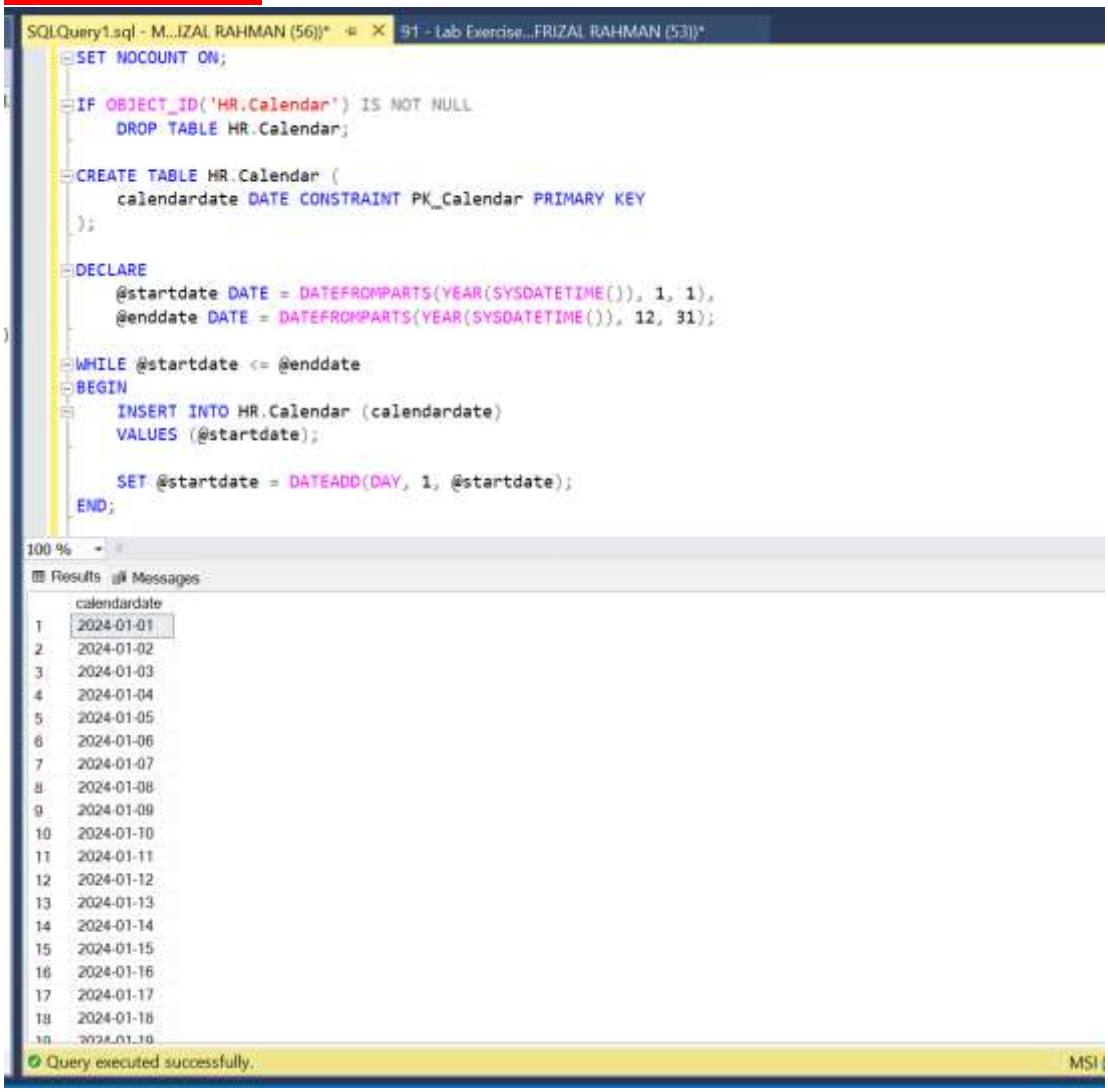
10 : Creating a Cross-Join Query

Step	Information
1	This case study begins with the HR department wanting to set up a personal calendar for each employee. The IT department will provide a T-SQL code that generates all days in the past year. Therefore, *the developer* will use the SELECT clause to return all rows from the calendar table for each row in the HR.Employees table. Open the project \10774A Labs\10774A_05_PRJ\10774A_05_PRJ.ssmssln and the T-SQL script 91 - Lab Exercise 5.sql. Make sure the database is connected with "TSQL".
	

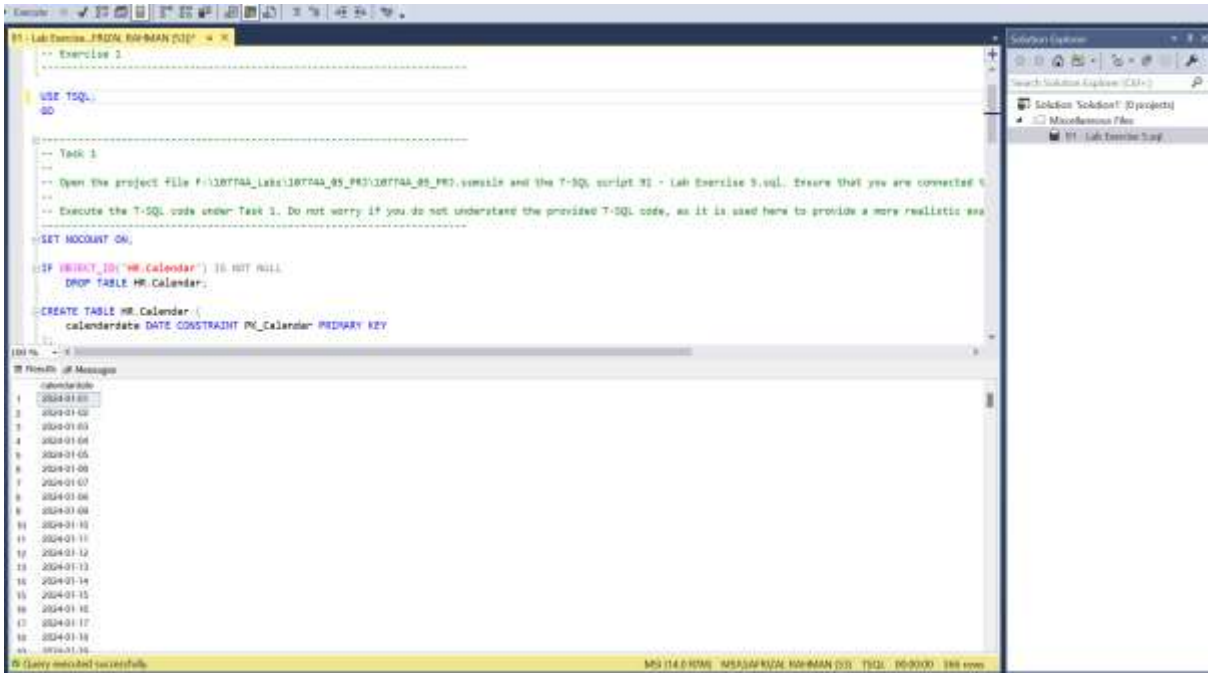
Practical – Part

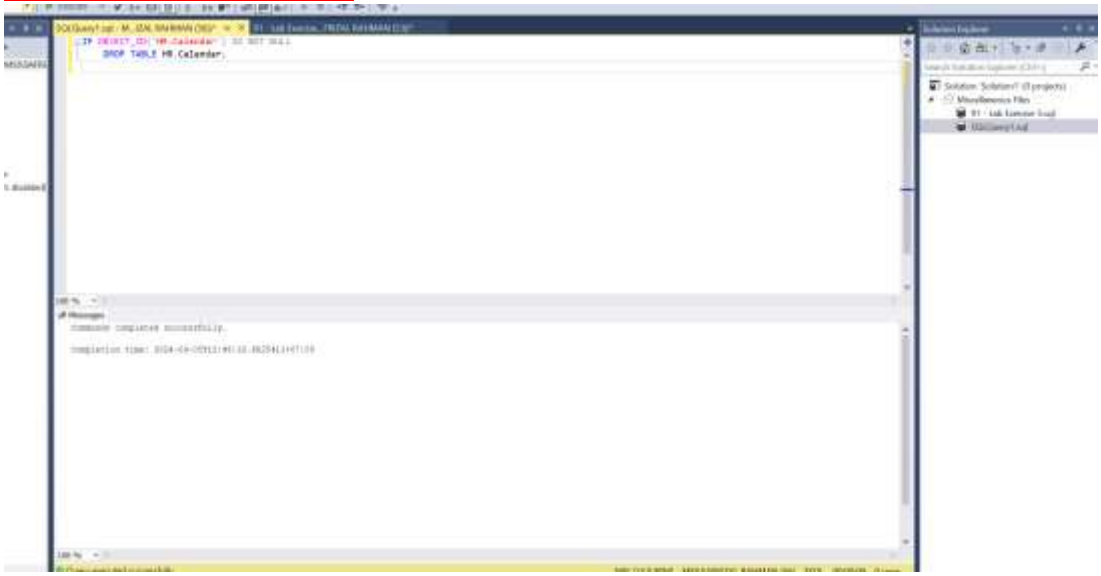
	
2	[Question- 24] Run the T-SQL code under task 1. Display the output! (Don't worry if you don't understand the T-SQL code. The next step will provide a more concrete example of how CROSSJOIN is implemented.) The T-SQL code under Task 1 creates and populates the HR.Calendar table with dates for the current year. Here is the code: **SQL** SET NOCOUNT ON; IF OBJECT_ID('HR.Calendar') IS NOT NULL DROP TABLE HR.Calendar; CREATE TABLE HR.Calendar (calendardate DATE CONSTRAINT PK_Calendar PRIMARY KEY); DECLARE @startdate DATE = DATEFROMPARTS(YEAR(SYSDATETIME()), 1, 1), @enddate DATE = DATEFROMPARTS(YEAR(SYSDATETIME()), 12, 31); WHILE @startdate <= @enddate BEGIN INSERT INTO HR.Calendar (calendardate) VALUES (@startdate); SET @startdate = DATEADD(DAY, 1, @startdate);

```
END;

SET NOCOUNT OFF;

GO

-- Observe the HR.Calendar table
SELECT
   calendardate
FROM HR.Calendar;
```

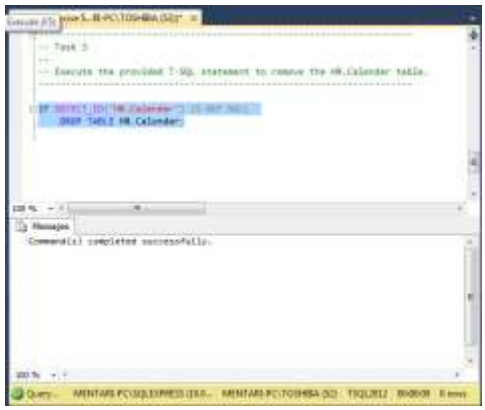


Practical – Part

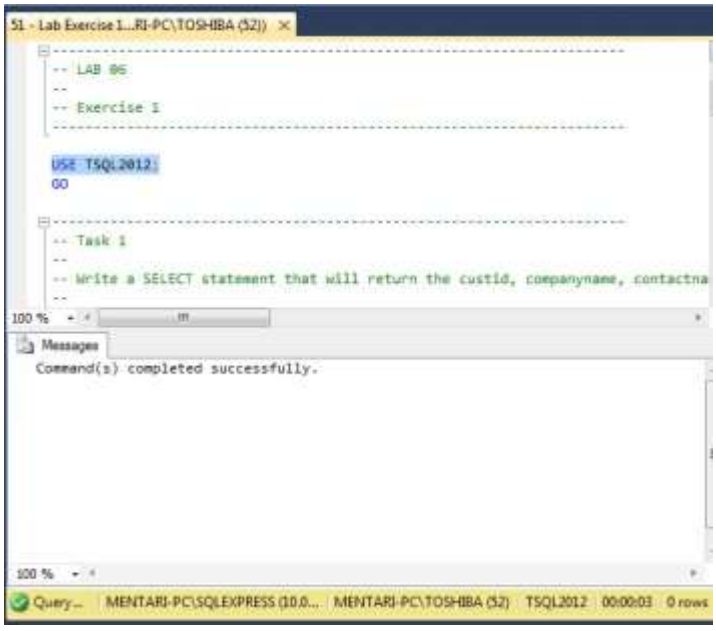
	
3	**[Question- 25]** Write a SELECT command to retrieve values from the empid, firstname, and lastname columns from the HR.Employees table and the calendardate column from the HR.Calendar table . SELECT e.empid, e.firstname, e.lastname, c.calendardate FROM HR.Employees e CROSS JOIN HR.Calendar c;

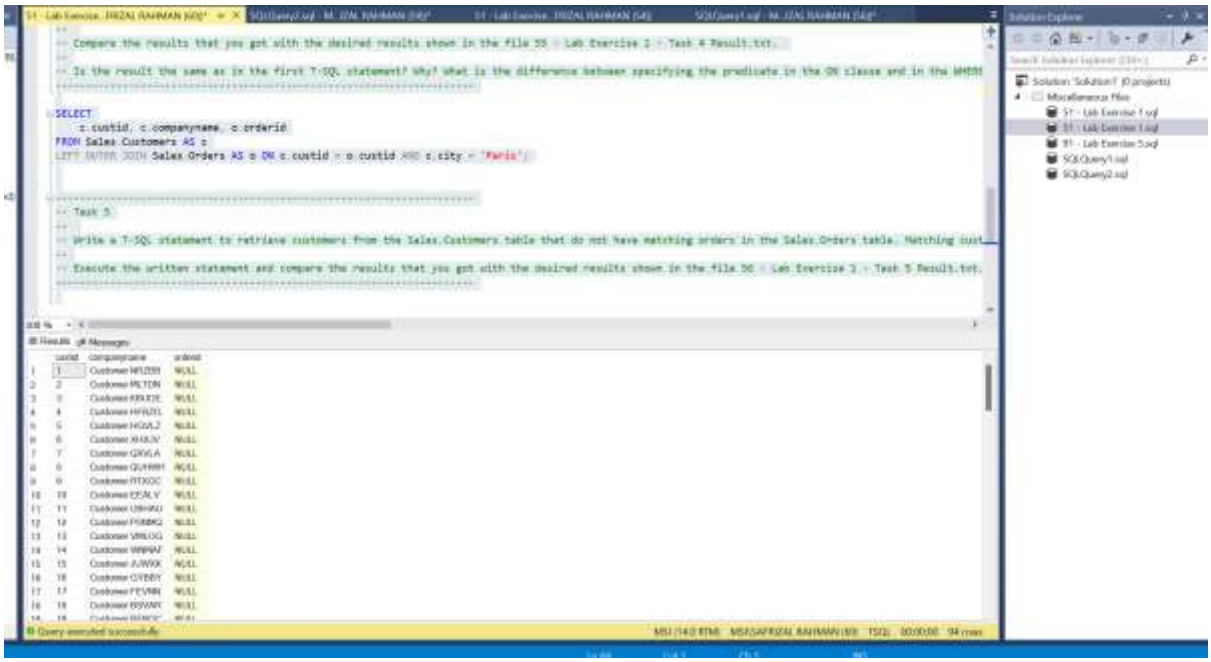
	
4	**[Question-2 6]** Execute the 3rd stage test and compare it with the file 92 - Lab Exercise 5 - Task 2 Result.txt . If the results are the same, then your test is correct. By completing these tasks, you've demonstrated your understanding of how to write and execute T-SQL CROSS JOIN queries. If you have any further questions or need additional assistance, feel free to ask! 
5	Drop the HR.Calendar table by executing the T-SQL code below task 3.

Practical – Part

	
6	**Conclusion** : After completing this practical section, you will understand how to write T-SQL CROSS-JOIN code .

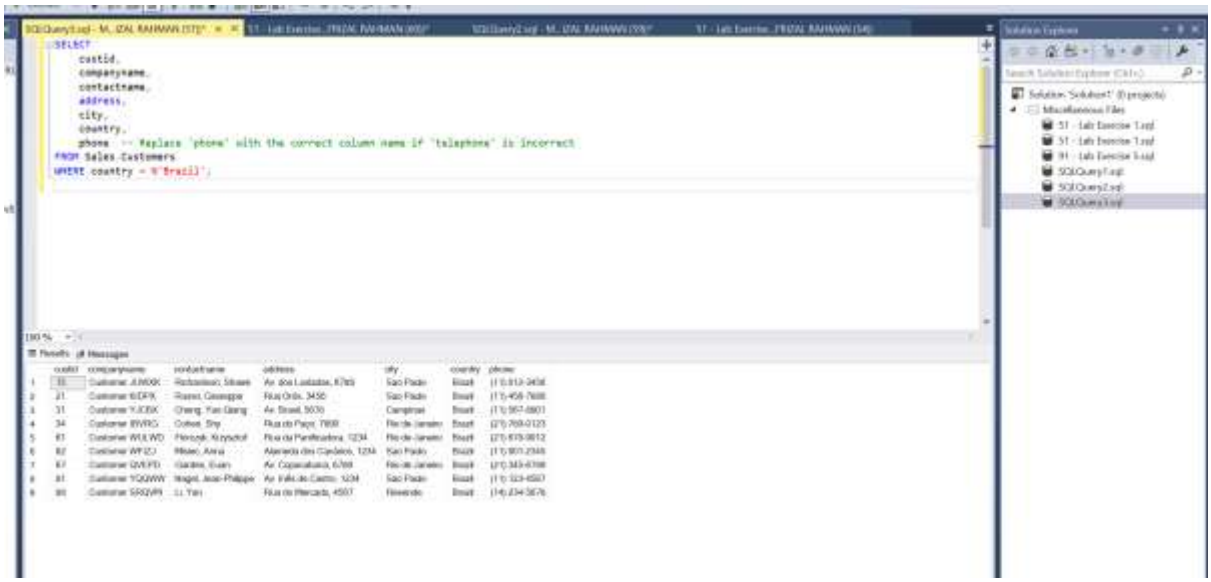
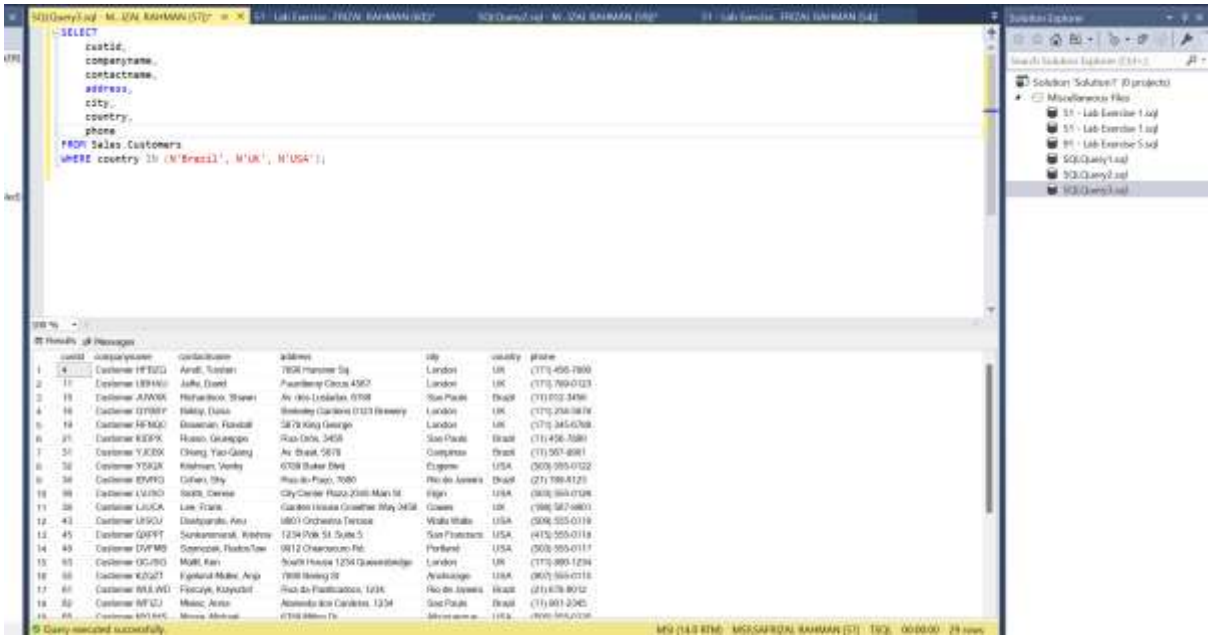
Practical – Part 11 : Writing Queries Who Will Filter Data with WHERE clause

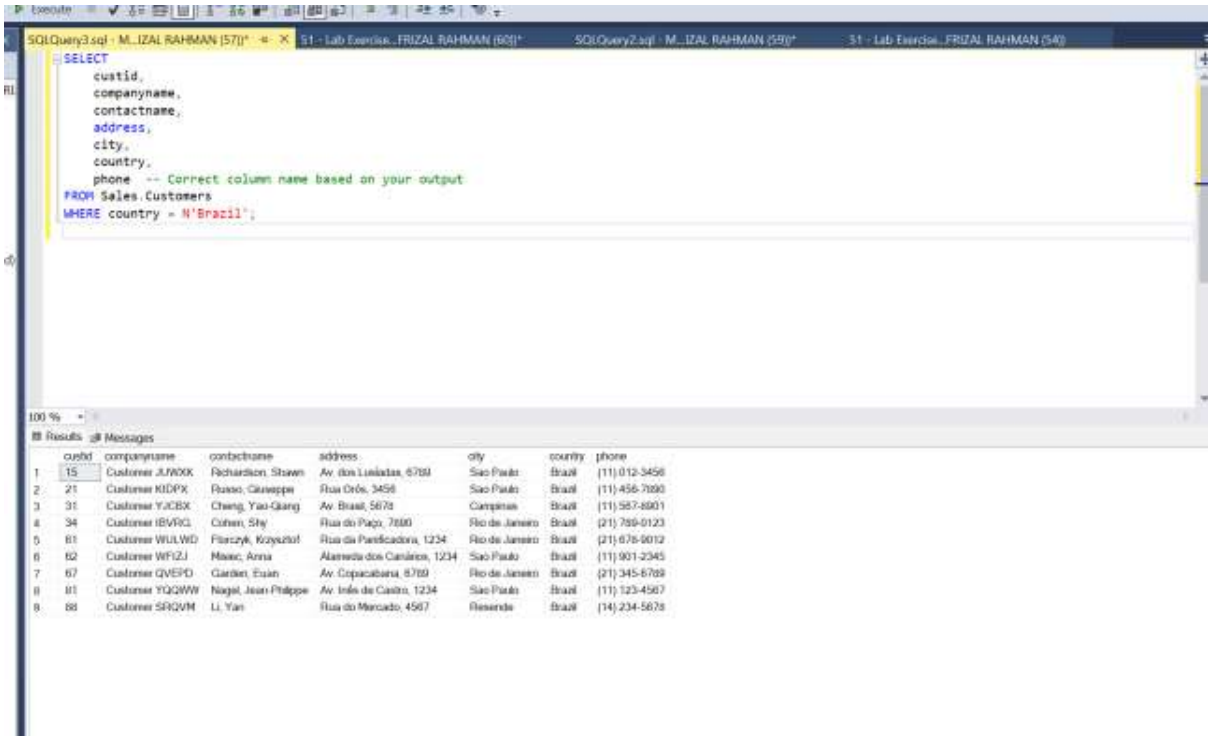
Step	Information
1	The scenario in this practicum uses the problems in the marketing department. The marketing department is working on several campaigns for old customers. The marketing staff needs a different customer list according to several business rules. Therefore, *the developer* will write a SELECT command to retrieve the desired rows from the Sales.Customers table. Open the project \10774A Labs\10774A_06_PRJ\10774A_06_PRJ.ssmssln and the T-SQL script 51 - Lab Exercise 1.sql. Make sure the database is connected with “TSQL”.
	

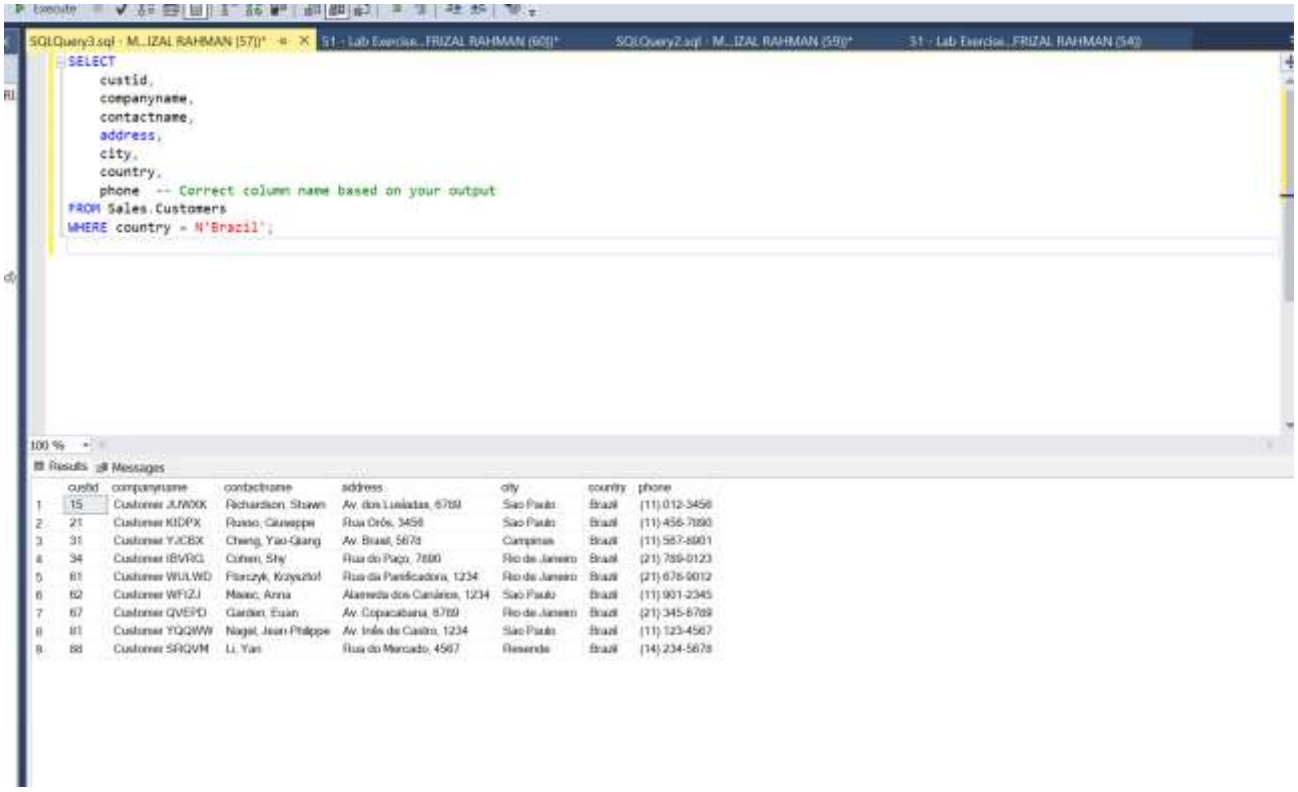
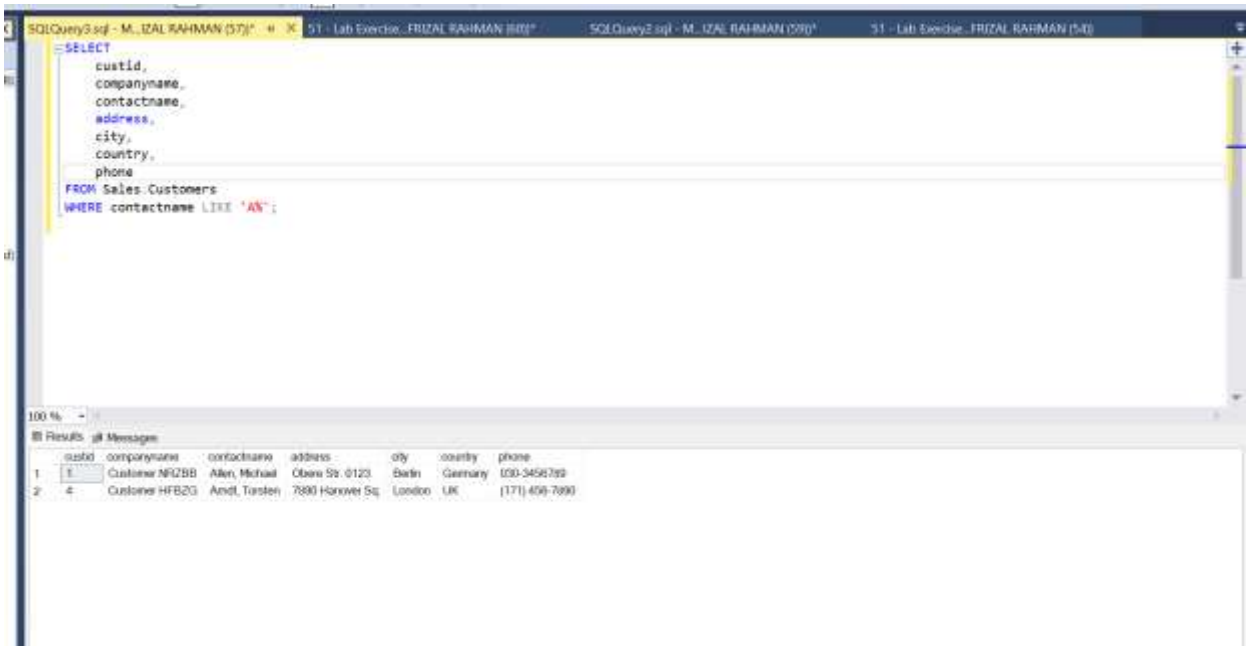


2	Write a SELECT statement that will return the column values from a table, Then filter the results to only customers who are from “Brazil”!
	
	`SELECT custid , companyname , contactname , address , city , country , telephone FROM Sales . Customers WHERE country = Brazil' ;`
	
	Use of the N prefix for literal characters (`N'Brazil'`). This prefix is used because the country column is a Unicode data type. When expressing Unicode characters literally, the N character (for National) is specified as the prefix.
	`SELECT custid, companyname, contactname, address, city, country, telephone FROM Sales.Customers WHERE country = N'Brazil';`
	
	`SELECT custid, companyname, contactname, address,`

Practical – Part

	city, country, phone -- Replace 'phone' with the correct column name if 'telephone' is incorrect FROM Sales.Customers WHERE country = N'Brazil'; 
3	**[Question- 27]** Execute the 2nd stage of the test and compare it with the file 52 - Lab Exercise 1 - Task 1 Result.txt . If the results are the same, then your test is correct. 


SELECT
custid,
companyname,
contactname,
address,
city,
country,
phone -- Correct column name based on your output
FROM Sales.Customers
WHERE country = N'Brazil';
Results
Messages
custid companyname contactname address city country phone
1 15 Customer JUWXK Richardson, Shawn Av. dos Lusíadas, 6789 Sao Paulo Brazil (11) 012-3456
2 21 Customer KIDPX Russo, Giuseppe Rua Orós, 3456 Sao Paulo Brazil (11) 456-7890
3 31 Customer YJCBX Cheng, Yao-Qiang Av. Brasil, 5678 Campinas Brazil (11) 567-8901
4 34 Customer IBVRG Cohen, Shy Rua do Paço, 7890 Rio de Janeiro Brazil (21) 789-0123
5 61 Customer WULWD Filarczyk, Krzysztof Rua da Panificadora, 1234 Rio de Janeiro Brazil (21) 678-9012
6 62 Customer WFIZJ Mares, Anna Alameda dos Canários, 1234 Sao Paulo Brazil (11) 901-2345
7 67 Customer QVEPD Garden, Evan Av. Copacabana, 6789 Rio de Janeiro Brazil (21) 345-6789
8 81 Customer YQQWW Nagel, Jean-Philippe Av. Inês de Castro, 1234 Sao Paulo Brazil (11) 123-4567
9 88 Customer SRQVM Li, Yan Rua do Mercado, 4567 Resende Brazil (14) 234-5678

4	**[Question- 28]** Write a SELECT command that will return values in the custid, companyname, contactname, address, city, columns. country, and phone in the Sales.Customers table , then filter the results only for “Brazil, UK and USA” (Use the IN predicate in the WHERE clause). 
5	**[Question-2 9]** Execute the 3rd stage test and compare it with file 53 - Lab Exercise 1 - Task 2 Result.txt . If the results are the same, then your test is correct. 

```
SELECT
    custid,
    companyname,
    contactname,
    address,
    city,
    country,
    phone
FROM Sales.Customers
WHERE country IN (N'Brazil', N'UK', N'USA');
```

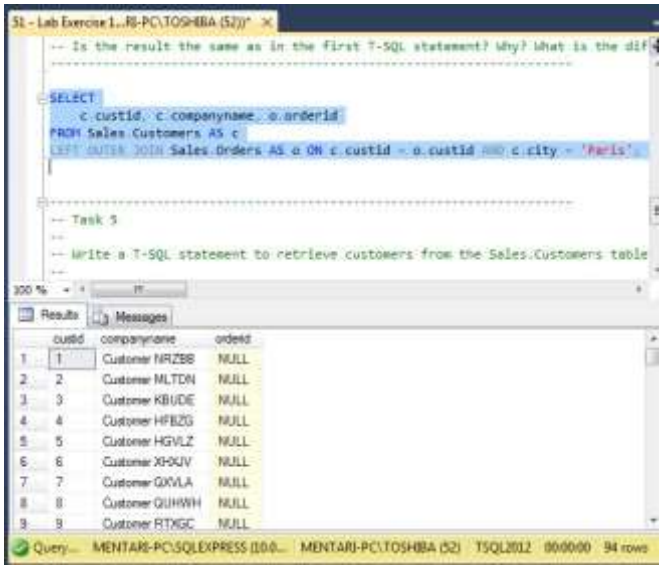
Results | Messages

	custid	companyname	contactname	address	city	country	phone
1	4	Customer HFBZG	Arndt, Torsten	7890 Hanover Sq.	London	UK	(171) 456-7890
2	11	Customer UBHAU	Jaffe, David	Faunillercy Circus 4567	London	UK	(171) 789-0123
3	15	Customer JUWXK	Richardson, Shawn	Av. dos Lusíadas, 6789	Sao Paulo	Brazil	(11) 012-3456
4	16	Customer GYBBY	Bikby, Dana	Berkeley Gardens 0123 Brewery	London	UK	(171) 234-5678
5	19	Customer RFNQC	Boseman, Randall	5678 King Georgs	London	UK	(171) 345-6789
6	21	Customer KIDPX	Russo, Giuseppe	Rua Orós, 3456	Sao Paulo	Brazil	(11) 456-7890
7	31	Customer YJCBX	Cheng, Yao-Qiang	Av. Brasil, 5678	Campinas	Brazil	(11) 567-6901
8	32	Customer YSIQX	Kristman, Vonky	6789 Baker Blvd.	Eugene	USA	(503) 555-0122
9	34	Customer IBVRG	Cohen, Shy	Rua do Paço, 7890	Rio de Janeiro	Brazil	(21) 789-0123
10	36	Customer LVJSO	Smith, Denise	City Center Plaza 2345 Main St.	Elgin	USA	(503) 555-0126
11	38	Customer LJUCA	Lee, Frank	Garden House Crowther Way 3456	Cowes	UK	(198) 567-8901
12	43	Customer UISOJ	Deshpande, Anu	8901 Orchestra Terrace	Walla Walla	USA	(509) 555-0119
13	45	Customer QXPPT	Sunkammurali, Krishna	1234 Polk St. Suite 5	San Francisco	USA	(415) 555-0118
14	48	Customer DVFMB	Szymczak, Radosław	8012 Chiaroscuro Rd.	Portland	USA	(503) 555-0117
15	53	Customer GCJSG	Mallit, Ken	South House 1234 Queensbridge	London	UK	(171) 890-1234
16	55	Customer KZQZT	Egelund-Muller, Anja	7890 Bering St.	Anchorage	USA	(907) 555-0115
17	61	Customer WULWD	Floryczyk, Krzysztof	Rua da Panificadora, 1234	Rio de Janeiro	Brazil	(21) 678-9012
18	62	Customer WFIZJ	Mlsec, Anna	Alameda dos Canários, 1234	Sao Paulo	Brazil	(11) 901-2345

6

The IT department has written T-SQL code to return values in the custid, companyname columns in the Sales.Customers table and the orderid column. in the Sales.Orders table as below:

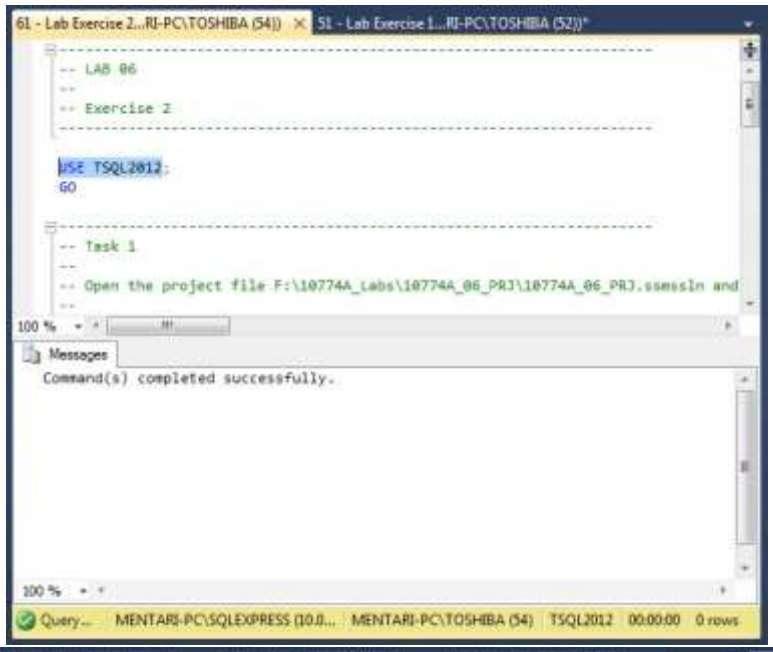
```
SELECT
c . custid , c . companyname , o . orderid
FROM Sales . Customers AS c
LEFT OUTER JOIN Sales . Orders AS o ON c . custid = o . custid AND c . city = 'Paris' ;
```

7	Query execution in the 7th stage of the trial. Note two things, first the query will retrieve all rows in the Sales.Customers table . Second, the use of the comparison operator with the ON clause makes the city column more specific, namely the same as the value "Paris".
	
8	**[Question-30]** Copy the T-SQL Code in step 7 then modify it with the comparison operator for the city column in the WHERE clause. After that execute the code, show the result!
	

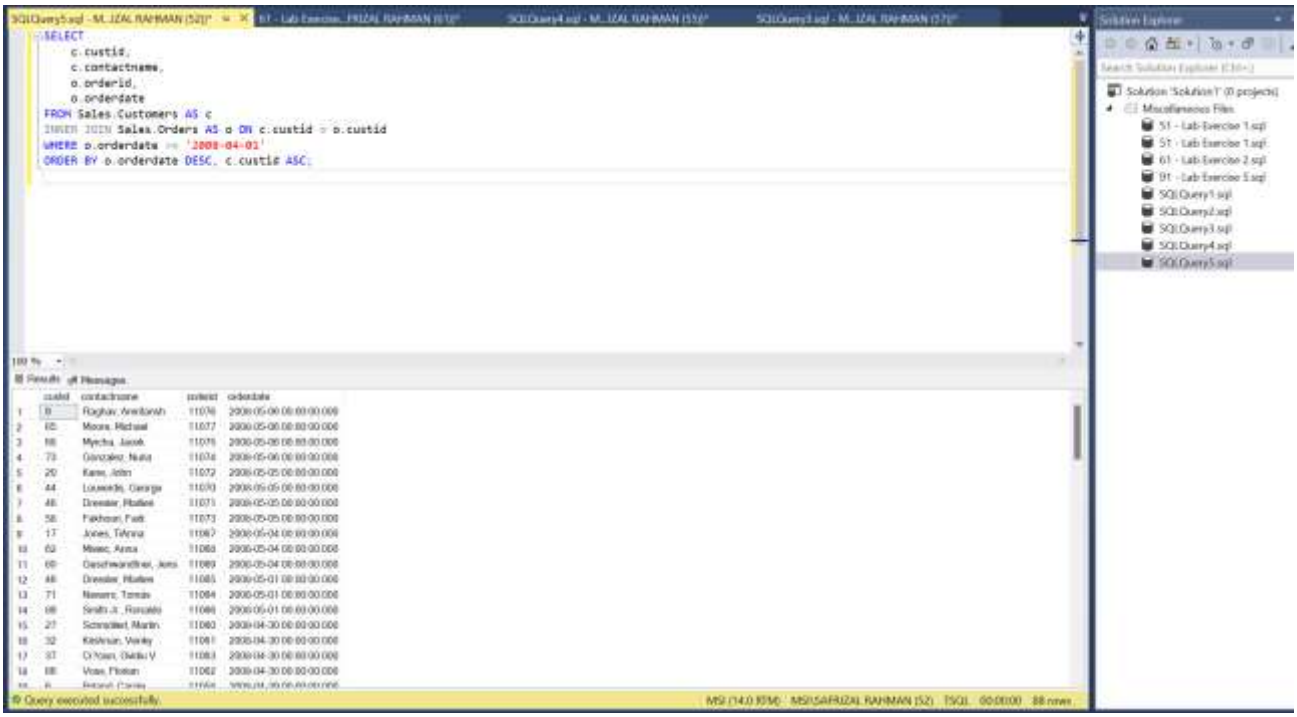
	
9	Compare the results of step 9 with file 55 - Lab Exercise 1 - Task 4 Result.txt . If the results are the same, then your test is correct. 
10	**Conclusion** : After completing the practicum and answering the questions in this section, you should understand how to filter data rows from one or more tables using the WHERE clause with logical operator predicates.

Practical – Part 11 : Writing Queries Which Will Sort Data with clause ORDER BY

Step	Information
1	The case study in this lab is based on a problem in the sales department. The sales department wants to create a report that shows all orders with some customer information. In addition, there is an additional request to sort the data based on order dates and the customer IDs. The order rows in the previous lab were displayed without using the ORDER BY clause, therefore specifically for this lab section the WHERE command will be followed by the ORDER BY clause. Open the project \10774A Labs\10774A_06_PRJ\10774A_06_PRJ.ssmssln and the T-SQL script 61 - Lab Exercise 2.sql . Make sure the database is connected with “TSQL”.

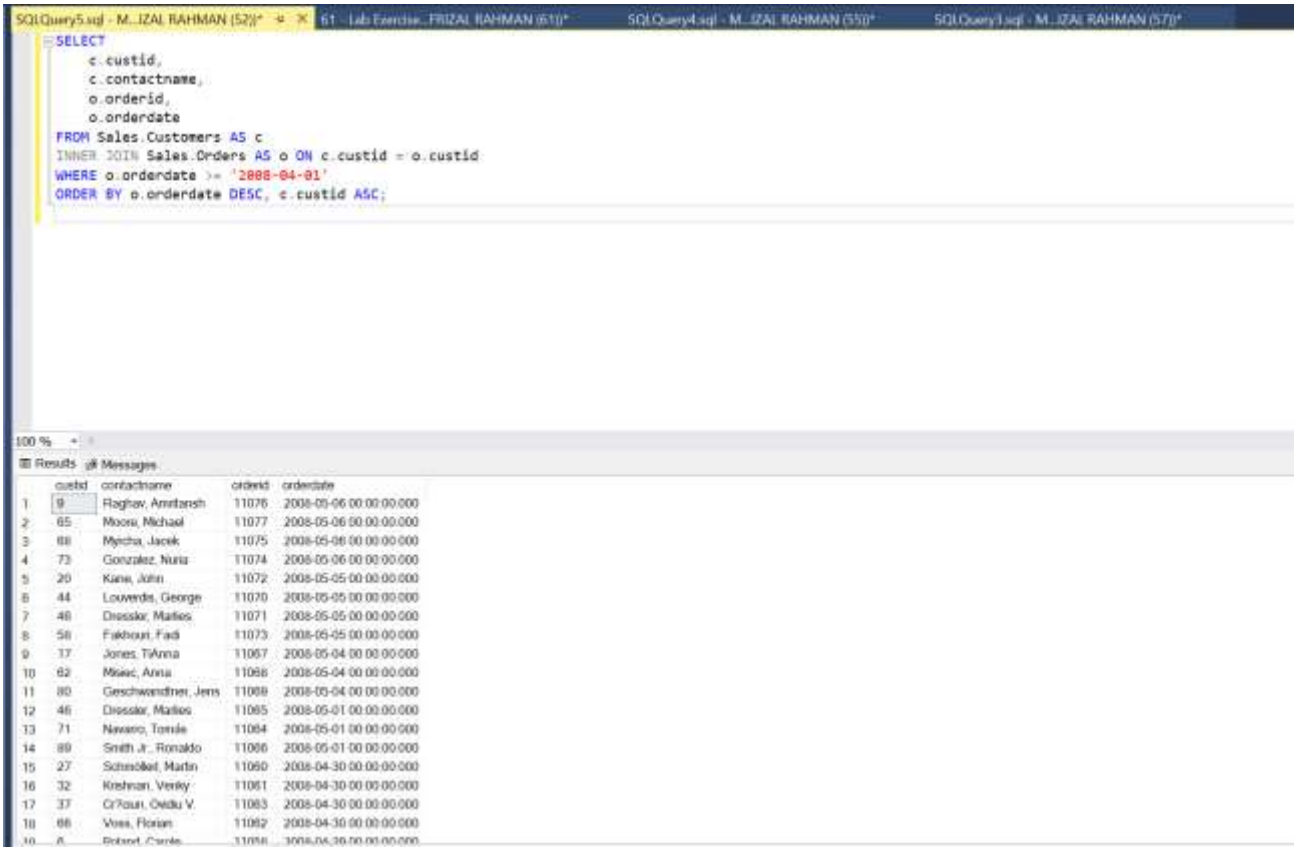


2	**[Question- 31]** Write a SELECT command to retrieve the custid, custname columns from the Sales.Customers table and the orderid, orderdate columns from the Sales.Orders table ! Filter the results only for orders on or after April 1, 2008. Then sort the results based on orderdate in descending order and custid in ascending order!



3

[Question- 32] Execute the 2nd stage of the test and compare it with the file 62 - Lab Exercise 2 - Task 1 Result.txt . If the results are the same, then your test is correct.



custid	contactname	orderid	orderdate
9	Raghav, Amritansh	11076	2008-05-06 00:00:00.000
65	Moore, Michael	11077	2008-05-06 00:00:00.000
88	Mycha, Jacek	11075	2008-05-06 00:00:00.000
73	Gonzalez, Nuria	11074	2008-05-06 00:00:00.000
20	Kane, John	11072	2008-05-05 00:00:00.000
44	Louwerds, George	11070	2008-05-05 00:00:00.000
46	Dressler, Marlies	11071	2008-05-05 00:00:00.000
58	Fakhouri, Fadi	11073	2008-05-05 00:00:00.000
17	Jones, TiAnna	11067	2008-05-04 00:00:00.000
62	Misiec, Anna	11068	2008-05-04 00:00:00.000
80	Geschwandtner, Jens	11069	2008-05-04 00:00:00.000
46	Dressler, Marlies	11065	2008-05-01 00:00:00.000
71	Navarro, Tomás	11064	2008-05-01 00:00:00.000
89	Smith Jr., Ronaldo	11066	2008-05-01 00:00:00.000
27	Schmidbel, Martin	11060	2008-04-30 00:00:00.000
32	Krishnan, Venky	11061	2008-04-30 00:00:00.000
37	O'Toure, Ovidiu V.	11063	2008-04-30 00:00:00.000
66	Voss, Florian	11062	2008-04-30 00:00:00.000

The T-SQL command from the previous practicum followed by the WHERE command is as follows:

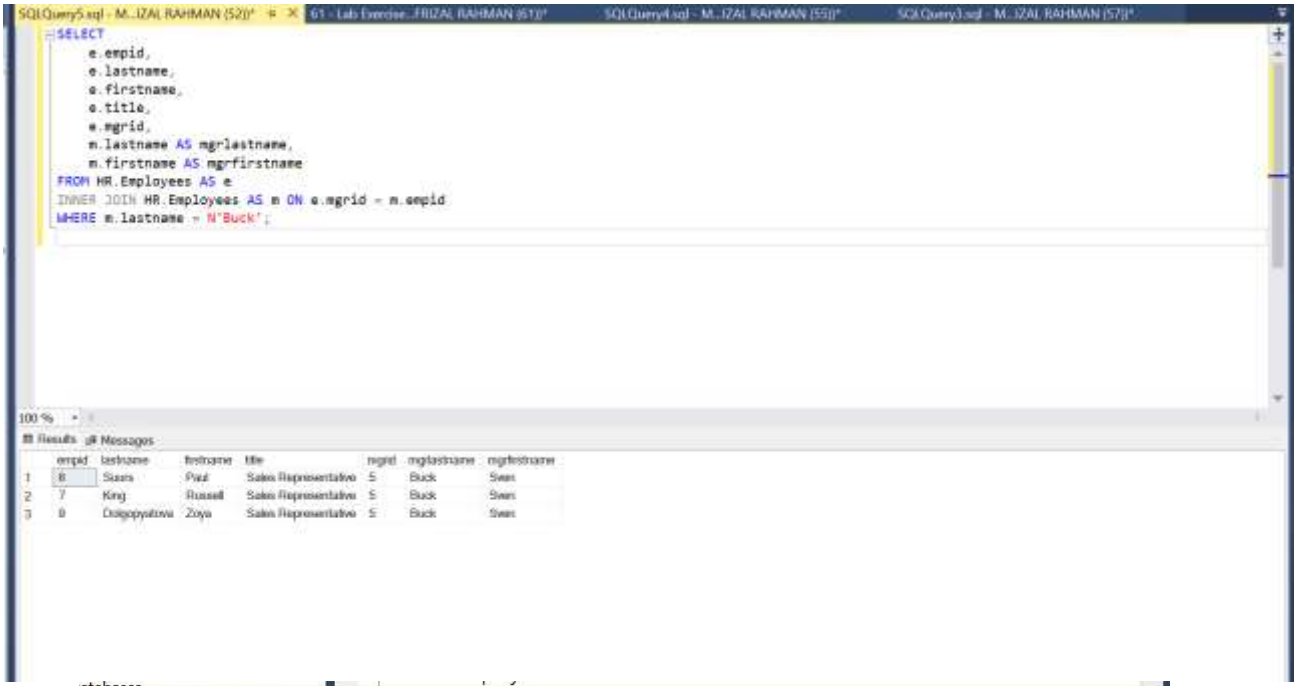
```
SELECT
e . empid , e . lastname , e . firstname , e . title , e . mgrid , m
. lastname AS mgrlastname , m . firstname AS mgrfirstname FROM HR .
Employees AS e
INNER JOIN HR . Employees AS m ON e . mgrid = m . empid WHERE
mgrlastname = N'Buck' ;
```

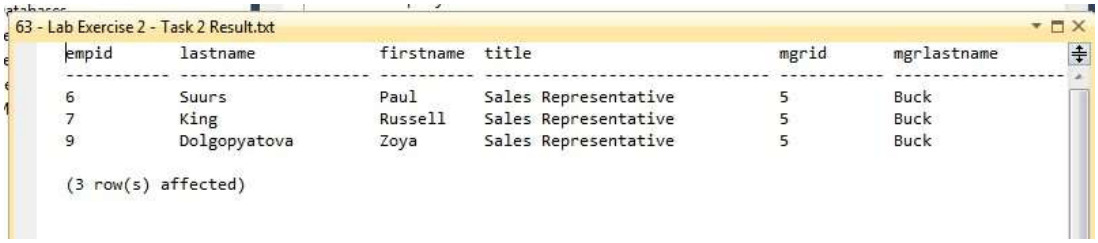
[Question- 33] Execute the T-SQL command at stage 3. Did an error occur? What is the error message? What do you think is the cause?

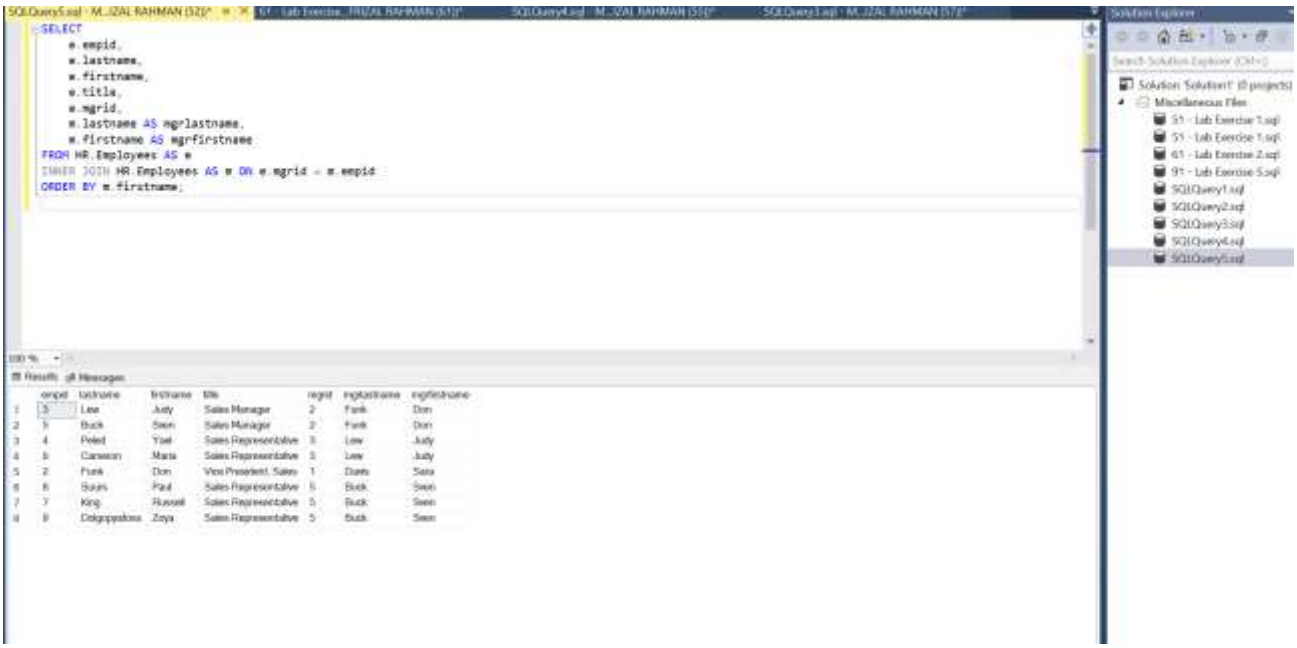
Msg 207, Level 16, State 1, Line 4
Invalid column name 'mgrlastname'.

4

[Question-3 4] Make changes to the T-SQL command to fix the error in the 3rd trial, then execute it! Compare the execution results with the file 63 - Lab Exercise 2 - Task 2 Result.txt. If the same, then the test result is correct.

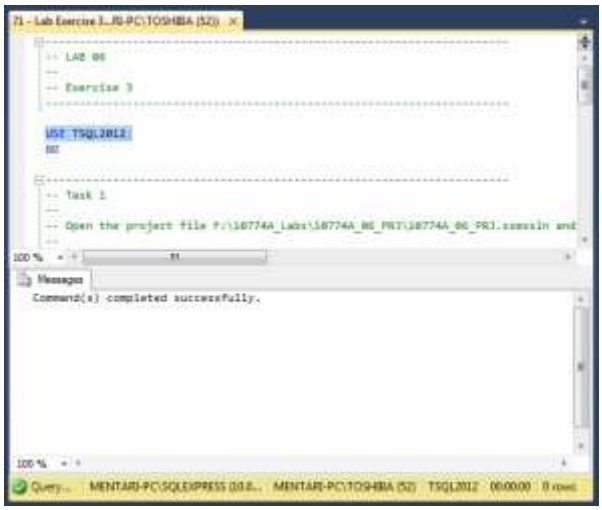




5	**[Question- 35]** Copy the T-SQL command in experiment 4, and modify it to produce all employees ORDER BY manager's first name. Initially test using the table's original name, then test using the table's alias name! Execute the T-SQL and compare the results to the 64 - Lab Exercise 2 - Task 3 Result.txt file . If the results are the same, then the experiment wa correct.  64 - Lab Exercise 2 - Task 3 Result.txt empid lastname firstname title mgrid mgrlastname 3 Lew Judy Sales Manager 2 Funk 5 Buck Sven Sales Manager 2 Funk 4 Peled Yael Sales Representative 3 Lew 8 Cameron Maria Sales Representative 3 Lew 2 Funk Don Vice President, Sales 1 Davis 6 Suurs Paul Sales Representative 5 Buck 7 King Russell Sales Representative 5 Buck 9 Dolgopyatova Zoya Sales Representative 5 Buck (8 row(s) affected)  SELECT e.empid, e.lastname, e.firstname, e.title, e.mgrid, e.lastname AS mgrlastname, e.firstname AS mgrfirstname FROM HR.Employees AS e INNER JOIN HR.Employees AS e ON e.mgrid = e.empid ORDER BY e.firstname;
6	**[Question-3 6]** Why can we use column names according to the original table name or use table alias names? SELECT e.empid, e.lastname,

	e.firstname, e.title, e.mgrid, m.lastname AS mgrlastname, m.firstname AS mgrfirstname FROM HR.Employees AS e INNER JOIN HR.Employees AS m ON e.mgrid = m.empid ORDER BY mgrfirstname; **Clarity and Precision**: Using table aliases helps make your queries clearer, especially when joining multiple tables. It ensures that the columns are correctly referenced and avoids ambiguity. **Flexibility**: SQL allows both approaches for flexibility. If table aliases are used, they provide a more concise way to write queries, and they can be used interchangeably with original table names if needed.
7	**Conclusion** : After working on the practical work and questions in this section, you should now understand how to use the ORDER BY clause .

Practical – Part 12 : Writing Queries Who Will Do Data Filtering with clauses TOP

Step	Information
1	Part 8 of the lab uses a case study on the sales department. The sales department wants to create an additional report that shows the order invoices and the 10 percent of the most expensive products that have been sold. Open the project \10774A Labs\10774A_06_PRJ\10774A_06_PRJ.ssmssln and the T-SQL script 71 - Lab Exercise 3.sql . Make sure the database is connected with “TSQL”.
	

2	**[Question- 37]** Write a SELECT command to display the productname and unitprice columns in the Production.Products table sorted descending by unitprice! Show the execution results!

```
SELECT
    productname,
    unitprice
FROM Production.Products
ORDER BY unitprice DESC;
```

	productname	unitprice
1	Product QDOMO	263.50
2	Product VJXYN	123.79
3	Product AOZBW	97.00
4	Product QHFFP	81.00
5	Product CKEDC	62.50
6	Product UKXRI	55.00
7	Product APITJ	53.00
8	Product WUXYK	49.30
9	Product ZZZHR	48.00
10	Product OFBNT	45.60
11	Product SMIOH	43.90
12	Product ICKNK	43.90
13	Product WVJFP	40.00
14	Product BLCAX	38.00
15	Product OSFNS	38.00
16	Product VKOMF	38.00
17	Product COAXA	38.00
18	Product GEEOO	34.80

Query executed successfully.

3	**[Question- 38]** Copy and modify the T-SQL command in trial 2 with the limitation that only 10 percent of the child products are displayed based on unitprice ordering! Execute the command, and compare whether it is in accordance with the file 73 - Lab Exercise 3 - Task 2 Result.txt.
4	**[Question- 39]** Is it possible to implement the 5 trial T-SQL command using the OFFSET-FETCH clause?

```
73 - Lab Exercise 3 - Task 2 Result.txt
productname                              unitprice
---------------------------------------- ---------------------
Product QDOMO                            263.50
Product V3XYN                            123.79
Product AOZBW                            97.00
Product QHFFP                            81.00
Product CKEDC                            62.50
Product UKXRI                            55.00
Product APITJ                            53.00
Product WUXYK                            49.30

(8 row(s) affected)
```

```
;WITH SortedProducts AS (
  SELECT
    productname,
    unitprice,
    ROW_NUMBER() OVER (ORDER BY unitprice DESC) AS RowNum,
    COUNT(*) OVER () AS TotalRows
  FROM Production.Products
)
SELECT
  productname,
  unitprice
FROM SortedProducts
WHERE RowNum <= (TotalRows * 0.10);
```

Yes, you can use the OFFSET-FETCH clause to achieve similar results, but it works with a fixed number of rows or pages, not percentages directly. However, you can combine it with a row count calculation. For instance, if you know the total number of rows, you can calculate the number of rows to fetch for 10% and use OFFSET-FETCH to retrieve that subset:

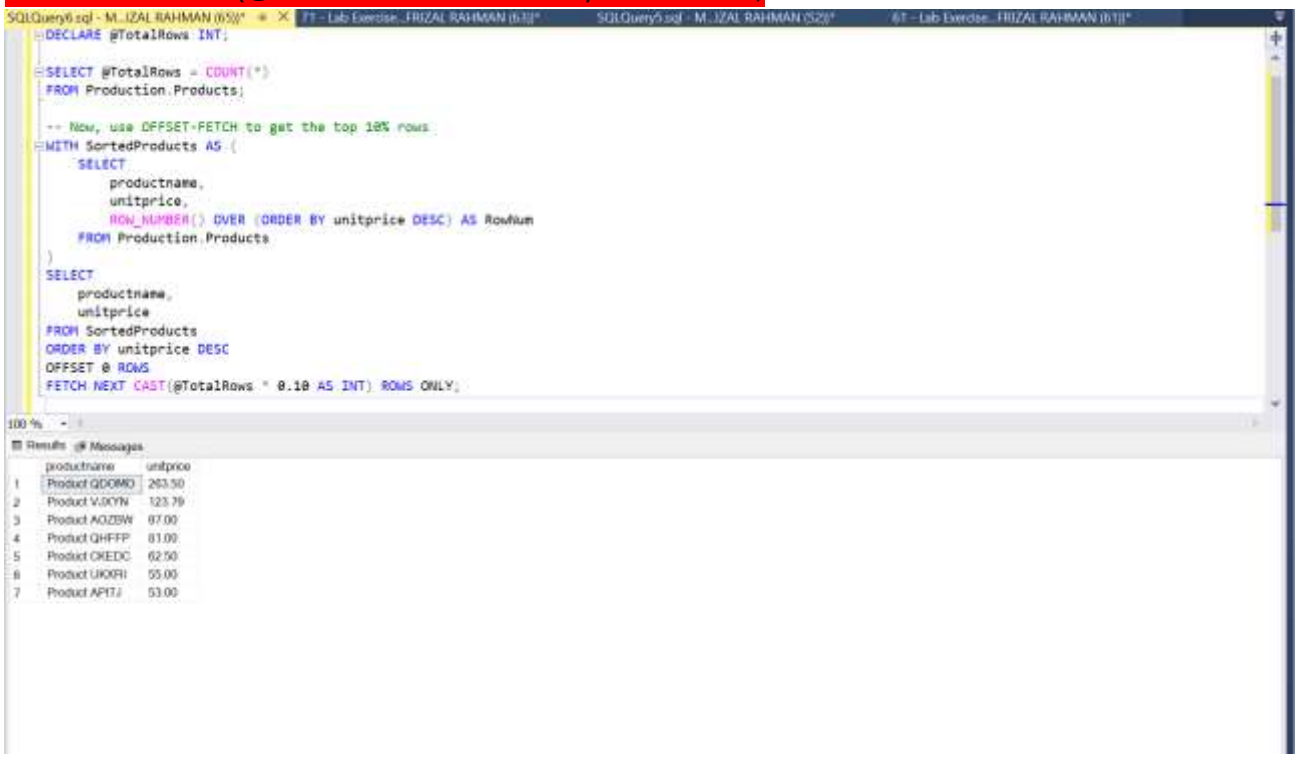
sql

Copy code

```
;WITH SortedProducts AS (
    SELECT
        productname,
        unitprice,
        ROW_NUMBER() OVER (ORDER BY unitprice DESC) AS RowNum,
        COUNT(*) OVER () AS TotalRows
    FROM Production.Products
)
SELECT
    productname,
    unitprice
FROM SortedProducts
ORDER BY unitprice DESC
OFFSET 0 ROWS
FETCH NEXT CAST((TotalRows * 0.10) AS INT) ROWS ONLY;
```

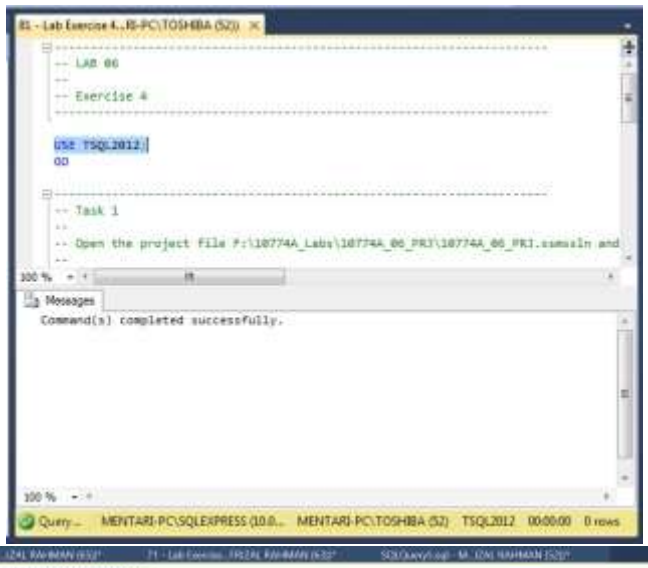
In this example, OFFSET 0 ROWS starts at the beginning, and FETCH NEXT CAST((TotalRows * 0.10) AS INT) ROWS ONLY limits the number of rows fetched to approximately 10% of the total rows. Note that FETCH works with exact row numbers, so you need to compute the number of rows based on the total rows available.

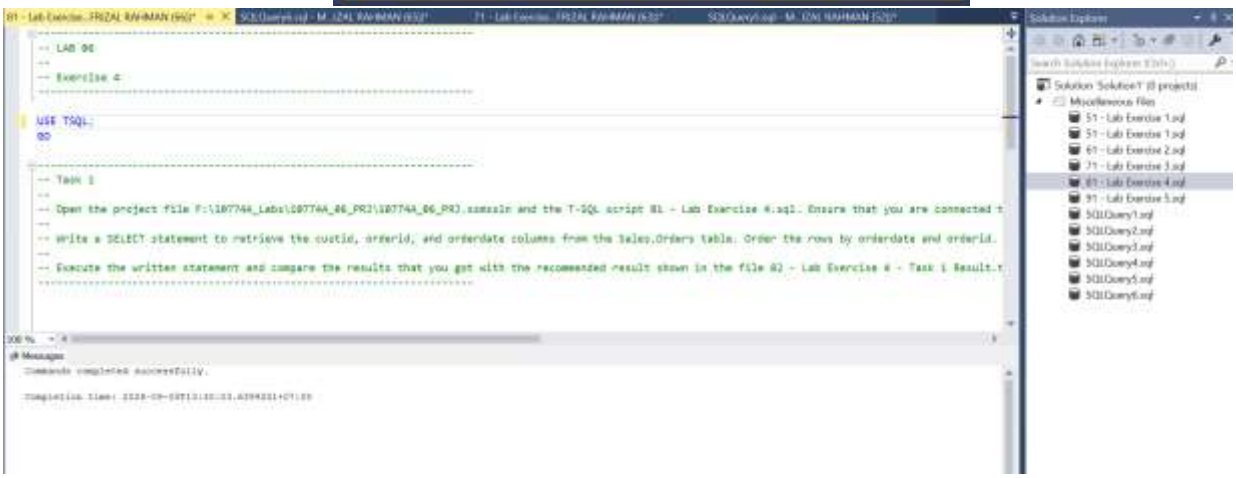
```
-- First, get the total number of rows
DECLARE @TotalRows INT;

SELECT @TotalRows = COUNT(*)
FROM Production.Products;

-- Now, use OFFSET-FETCH to get the top 10% rows
WITH SortedProducts AS (
    SELECT
        productname,
        unitprice,
        ROW_NUMBER() OVER (ORDER BY unitprice DESC) AS RowNum
    FROM Production.Products
)
SELECT
    productname,
```

	unitprice FROM SortedProducts ORDER BY unitprice DESC OFFSET 0 ROWS FETCH NEXT CAST(@TotalRows * 0.10 AS INT) ROWS ONLY; 
5	**Conclusion** : After completing the practical work and questions in this section, you should now understand how to apply the TOP option to the SELECT clause of the T-SQL command.

Practical – Part 13 : Writing Queries Who Will Filter Data with OFFSET-FETCH clause

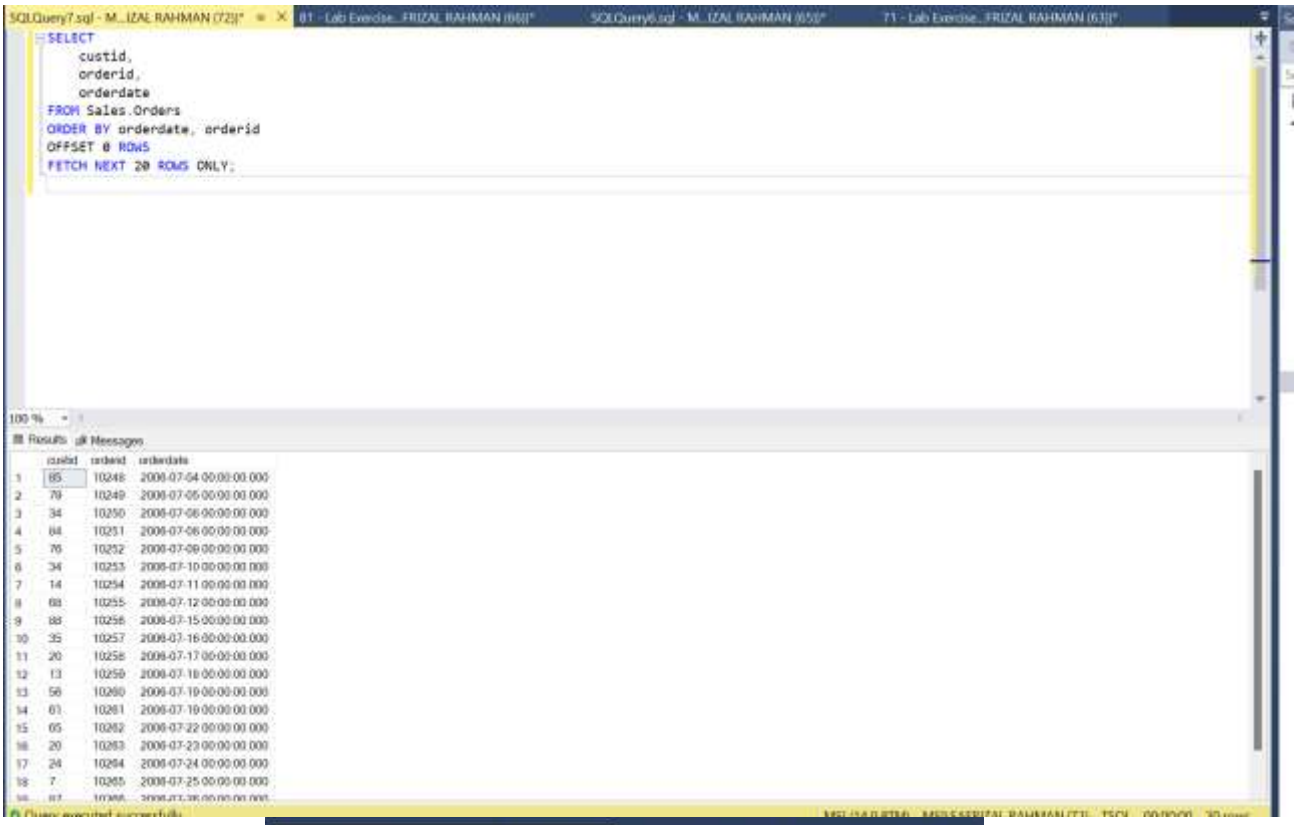
Step	Information
1	Practical part 9 will implement paging solution to display rows from Sales.Orders table , because the number of rows is too many. On each report page, user can only see 20 rows. Open the project \10774A Labs\10774A_06_PRJ\10774A_06_PRJ.ssmssln and the T-SQL script 81 - Lab Exercise 4.sql . Make sure the database is connected with “TSQL”.





2

[Question- 40] Write a SELECT command to display the custid, orderid, and orderdate columns in the Sales.Orders table . Sort the rows by orderdate and orderid. Take the first 20 rows. Execute the command and compare the results with the file 82 - Lab Exercise 4 - Task 1 Result.txt. If the results are the same, then your test is correct.



82 - Lab Exercise 4 - Task 1 Result.txt | 81 - Lab Exercise 4...RI-PC\TOSHIBA (52))

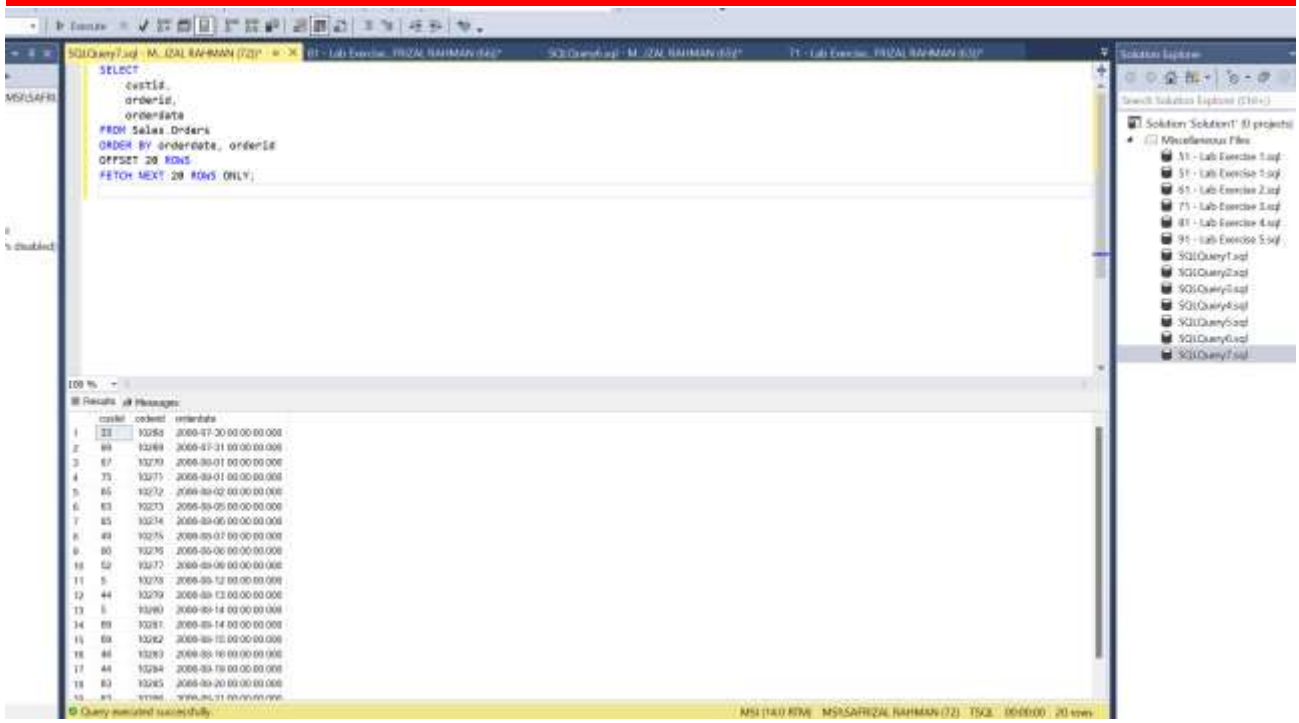
```
custid      orderid     orderdate
----------- ----------- -----------------------
85          10248       2006-07-04 00:00:00.000
79          10249       2006-07-05 00:00:00.000
34          10250       2006-07-08 00:00:00.000
...
...
...
7           10265       2006-07-25 00:00:00.000
87          10266       2006-07-26 00:00:00.000
25          10267       2006-07-29 00:00:00.000

(20 row(s) affected)
```

- ORDER BY orderdate, orderid sorts the result by orderdate first and then by orderid.
- OFFSET 0 ROWS skips no rows.
- FETCH NEXT 20 ROWS ONLY retrieves the next 20 rows after the offset.

3

[Question- 41] Write a SELECT statement to display the same results as question no. 43, skip the first 20 rows, and continue with the next 20 rows using the OFFSET-FETCH clause! Execute the statement and compare 83 - Lab Exercise 4 - Task 2 Result.txt. If the results are the same, then your test is correct.

- ORDER BY orderdate, orderid sorts the result by orderdate and orderid.
- OFFSET 20 ROWS skips the first 20 rows.

	? FETCH NEXT 20 ROWS ONLY retrieves the next 20 rows after the offset. 
	83 - Lab Exercise 4 - Task 2 Result.txt custid orderid orderdate ---------- ---------- ----------------------- 33 10268 2006-07-30 00:00:00.000 89 10269 2006-07-31 00:00:00.000 87 10270 2006-08-01 00:00:00.000 ... ... 63 10285 2006-08-20 00:00:00.000 63 10286 2006-08-21 00:00:00.000 67 10287 2006-08-22 00:00:00.000 (20 row(s) affected)
4	**Conclusion** : After working on the practical work and questions in this section, you should now understand how to use the OFFSET-FETCH clause in T-SQL commands.

-- Have a great time doing it -