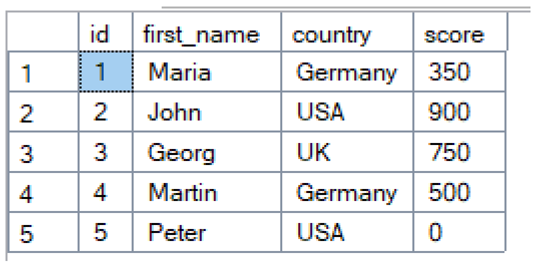
SQL QUERY

The **SELECT** query is used in SQL Server to get data from Table. It's like asking a question to your database and getting the answer in a table format.

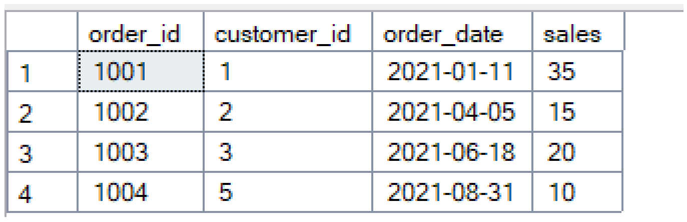
For Example-
Here is a Database of customers and orders

1) Customers:



	id	first_name	country	score
1	1	Maria	Germany	350
2	2	John	USA	900
3	3	Georg	UK	750
4	4	Martin	Germany	500
5	5	Peter	USA	0

2) orders:



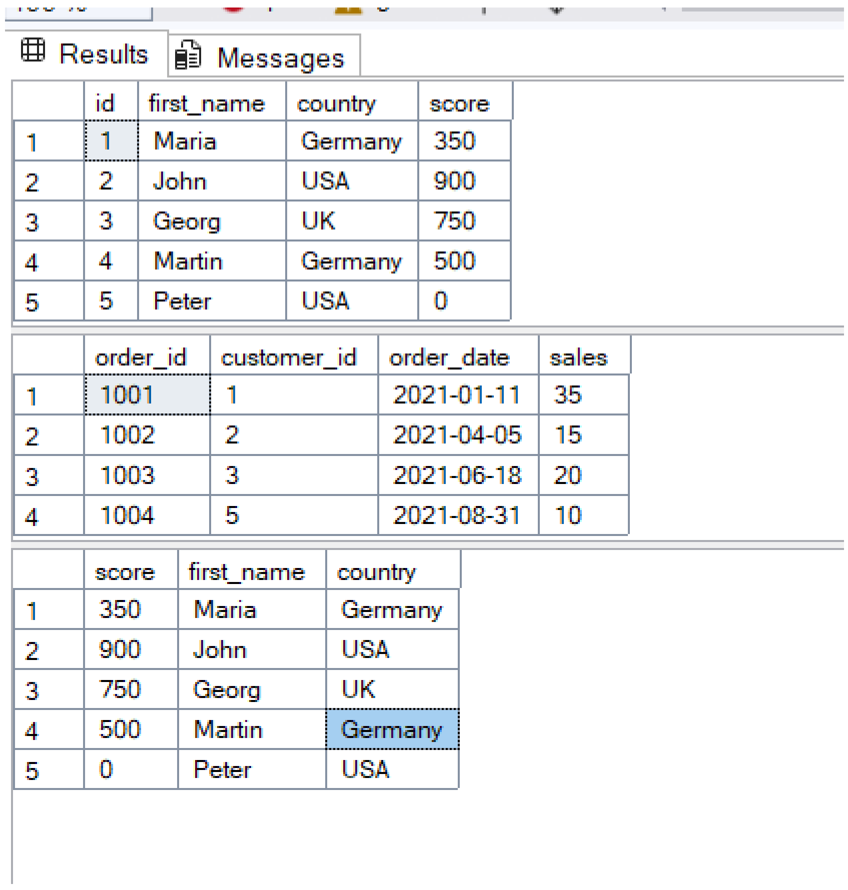
	order_id	customer_id	order_date	sales
1	1001	1	2021-01-11	35
2	1002	2	2021-04-05	15
3	1003	3	2021-06-18	20
4	1004	5	2021-08-31	10

By using -- This we can Do a Single Line Comment.
By using
/* We can
Write a
Multi line
Comment. */

```
USE MyDatabase;
-- This is a Comments--

/*This
is
a
MultiLine
Comment*/

--Retrive All Customer Data
SELECT *
FROM customers

--Retrive All the Order Data
```

```
SELECT *
FROM orders


--- SELECT FEW COLUMNS FROM DATABASE
-- INSTEAD FROM * WRITE THE COLUMN NAME
-- Retrieve Each Customer's Name, Country and Score.

SELECT
  score,
  first_name,
  country
  -- NO comma After a last Column --
FROM customers
```

<u>Here is a Output:-</u>

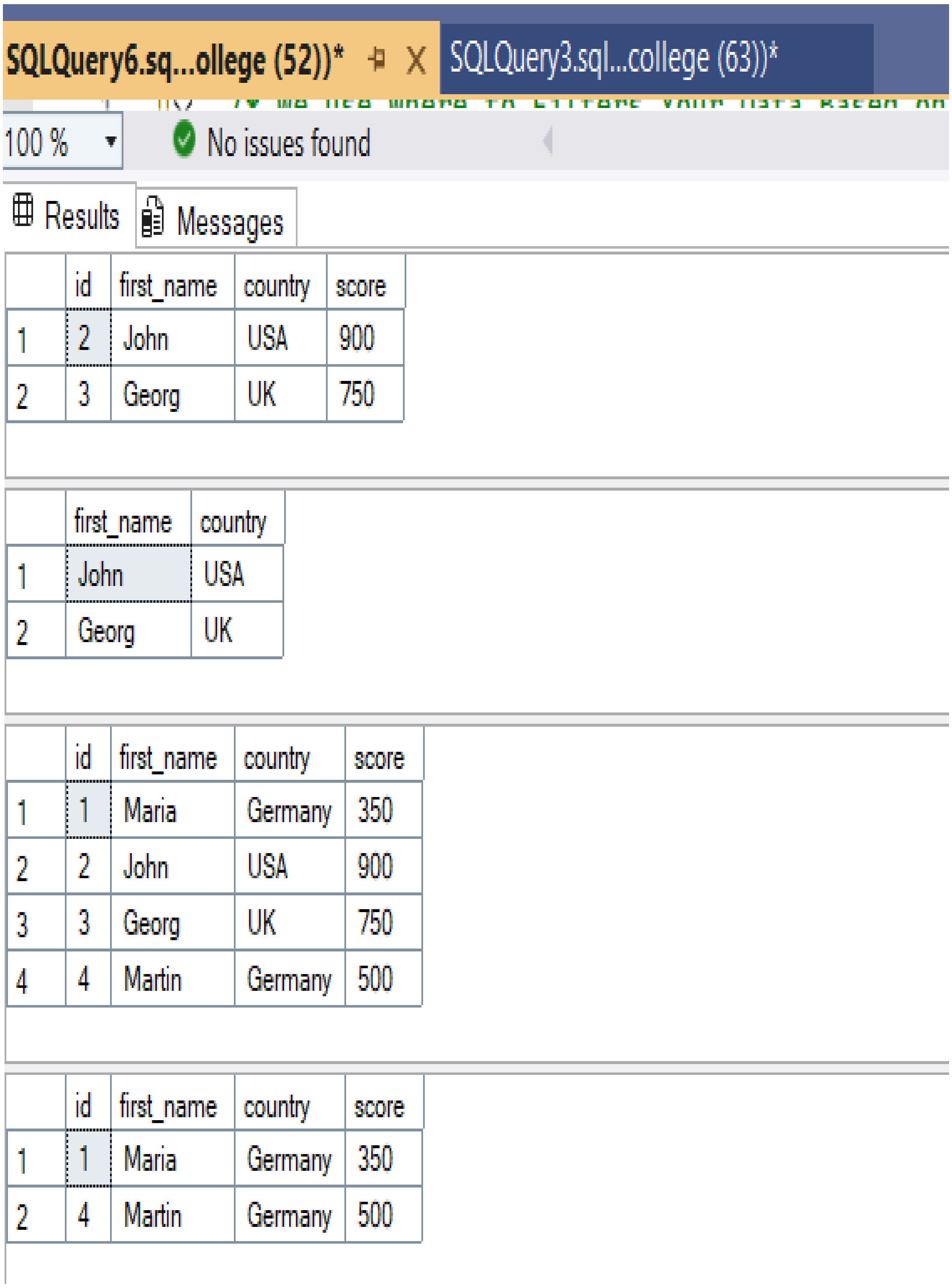


	id	first_name	country	score
1	1	Maria	Germany	350
2	2	John	USA	900
3	3	Georg	UK	750
4	4	Martin	Germany	500
5	5	Peter	USA	0

	order_id	customer_id	order_date	sales
1	1001	1	2021-01-11	35
2	1002	2	2021-04-05	15
3	1003	3	2021-06-18	20
4	1004	5	2021-08-31	10

	score	first_name	country
1	350	Maria	Germany
2	900	John	USA
3	750	Georg	UK
4	500	Martin	Germany
5	0	Peter	USA

```
/* We use Where to Filters Your Data Based on a
For ex-  Score Higher than 500
*/
Select *
From customers
where score > 500

-- Select Firstname and Country whose Score Higher than 500
Select
first_name,
country
From customers
where score > 500

-- Retrieve Customers With a Score not Equal to 0

Select *
From customers
where score != 0

-- Retrieve Customer From Germany
Select *
From customers
where country = 'Germany'
```


SQLQuery6.sq...ollege (52))* SQLQuery3.sql...college (63))*

100 % No issues found

Results Messages

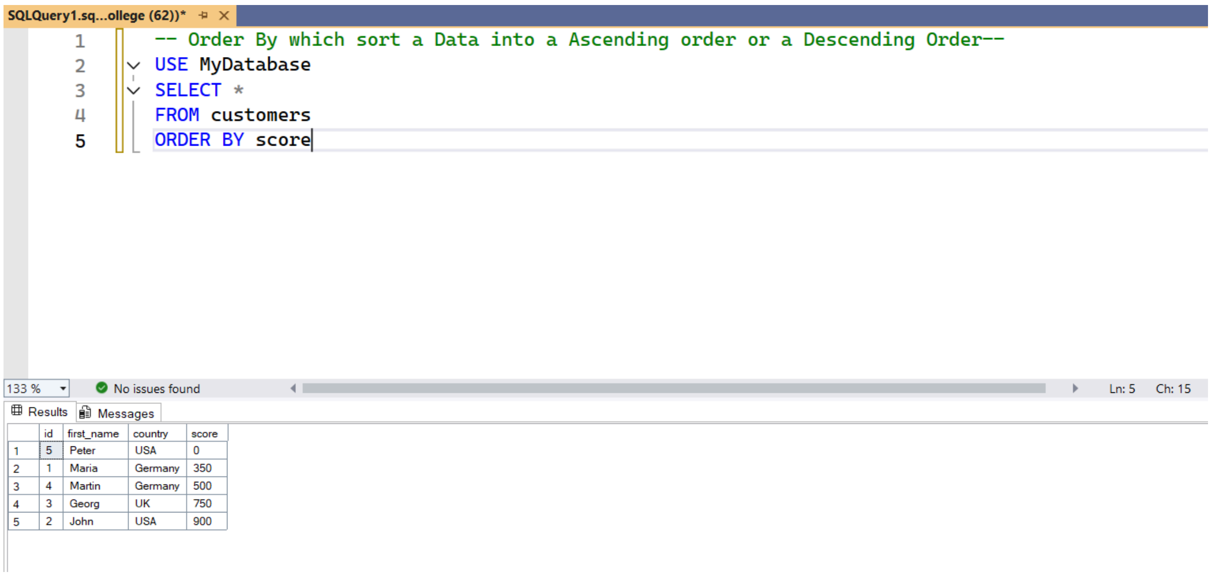
	id	first_name	country	score
1	2	John	USA	900
2	3	Georg	UK	750

	first_name	country
1	John	USA
2	Georg	UK

	id	first_name	country	score
1	1	Maria	Germany	350
2	2	John	USA	900
3	3	Georg	UK	750
4	4	Martin	Germany	500

	id	first_name	country	score
1	1	Maria	Germany	350
2	4	Martin	Germany	500

ORDER BY - (by Default it will do in the Ascending order but we can also use ASC)



SQLQuery1.sq...ollege (62))*

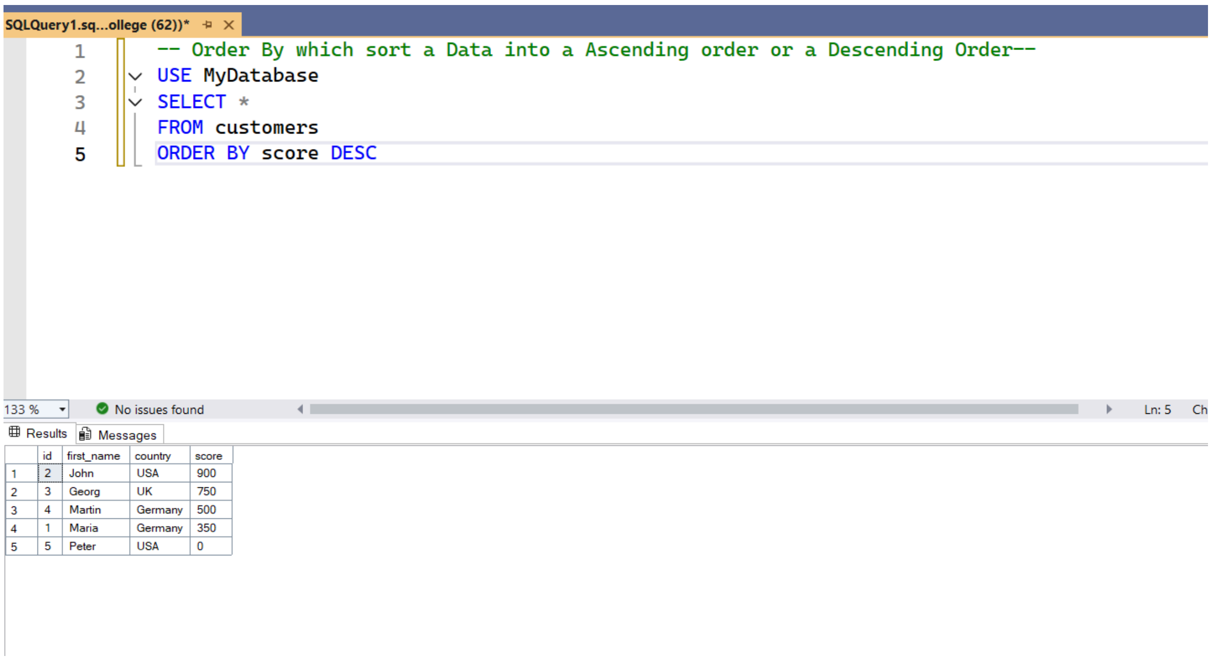
```sql
-- Order By which sort a Data into a Ascending order or a Descending Order--
USE MyDatabase
SELECT *
FROM customers
ORDER BY score
```

133 %   No issues found   Ln: 5   Ch: 15

Results | Messages

	id	first_name	country	score
1	5	Peter	USA	0
2	1	Maria	Germany	350
3	4	Martin	Germany	500
4	3	Georg	UK	750
5	2	John	USA	900

In this we have use a Descending order using DESC



SQLQuery1.sq...ollege (62))*

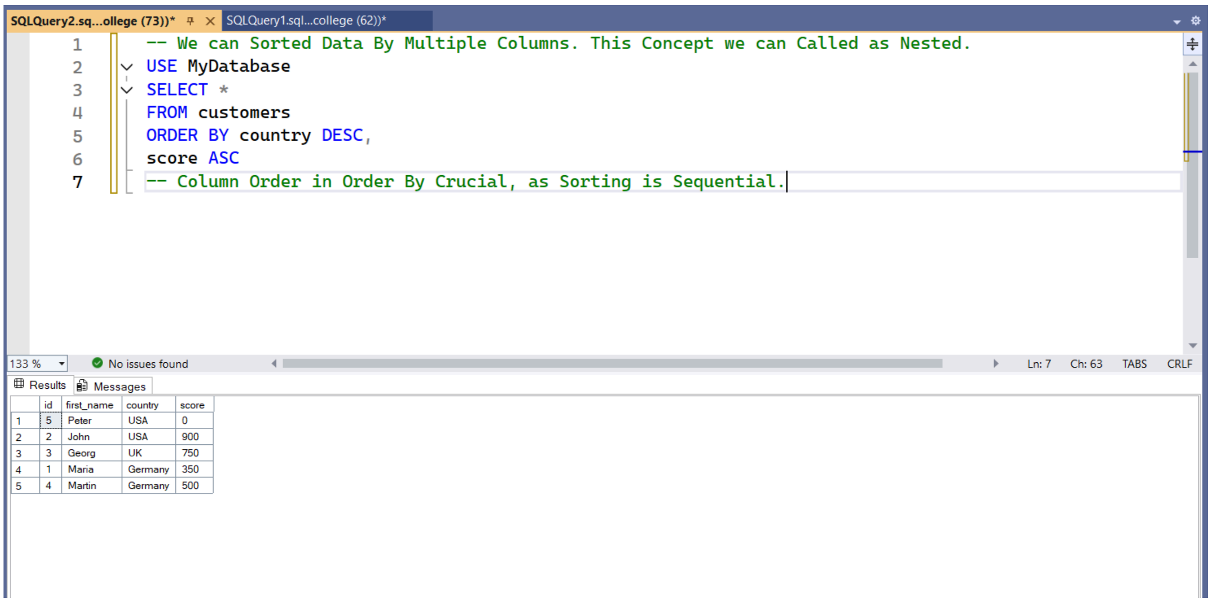
```sql
-- Order By which sort a Data into a Ascending order or a Descending Order--
USE MyDatabase
SELECT *
FROM customers
ORDER BY score DESC
```

133 %   No issues found   Ln: 5   Ch

Results | Messages

	id	first_name	country	score
1	2	John	USA	900
2	3	Georg	UK	750
3	4	Martin	Germany	500
4	1	Maria	Germany	350
5	5	Peter	USA	0

We have Sorted Data by Multiple Columns.

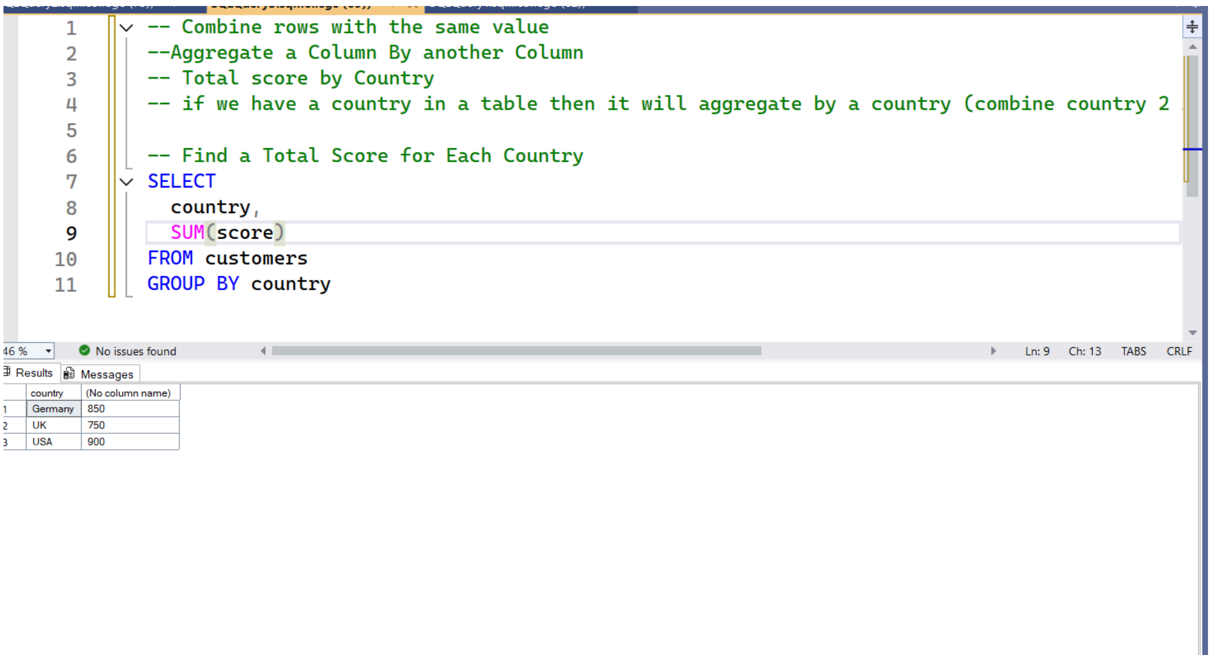

SQLQuery2.sq...ollege (73))* SQLQuery1.sql...college (62))*

```sql
-- We can Sorted Data By Multiple Columns. This Concept we can Called as Nested.
USE MyDatabase
SELECT *
FROM customers
ORDER BY country DESC,
score ASC
-- Column Order in Order By Crucial, as Sorting is Sequential.
```

133 % No issues found Ln: 7 Ch: 63 TABS CRLF

Results Messages

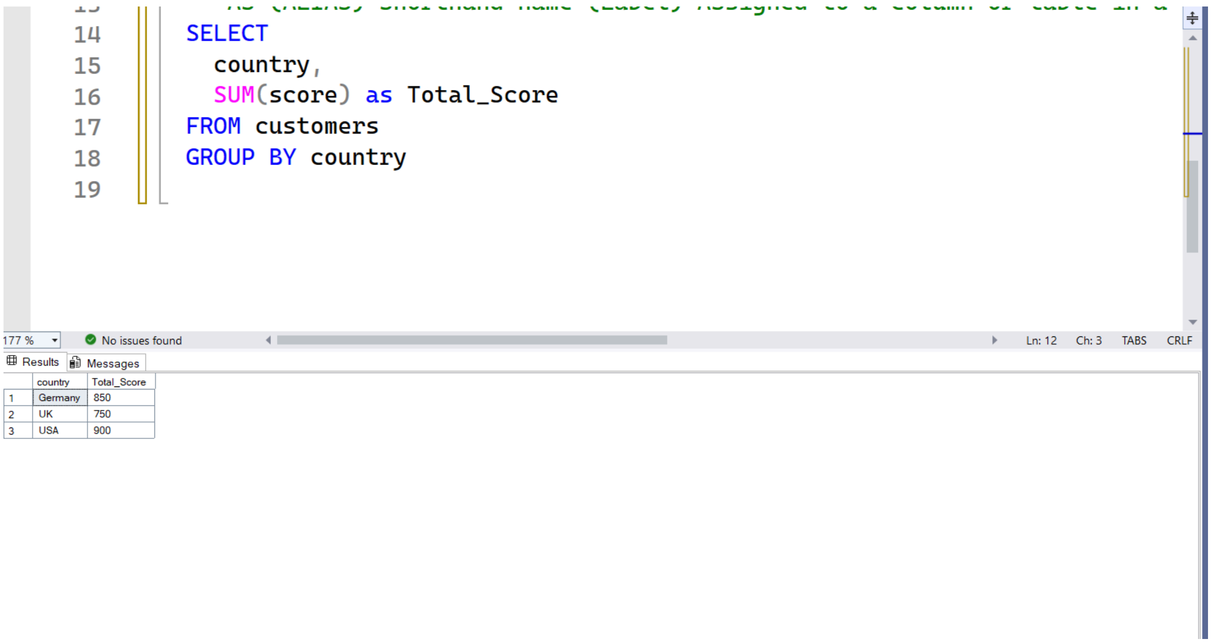
	id	first_name	country	score
1	5	Peter	USA	0
2	2	John	USA	900
3	3	Georg	UK	750
4	1	Maria	Germany	350
5	4	Martin	Germany	500


```sql
-- Combine rows with the same value
--Aggregate a Column By another Column
-- Total score by Country
-- if we have a country in a table then it will aggregate by a country (combine country 2

-- Find a Total Score for Each Country
SELECT
  country,
  SUM(score)
FROM customers
GROUP BY country
```

46 % No issues found Ln: 9 Ch: 13 TABS CRLF

Results Messages

	country	(No column name)
1	Germany	850
2	UK	750
3	USA	900



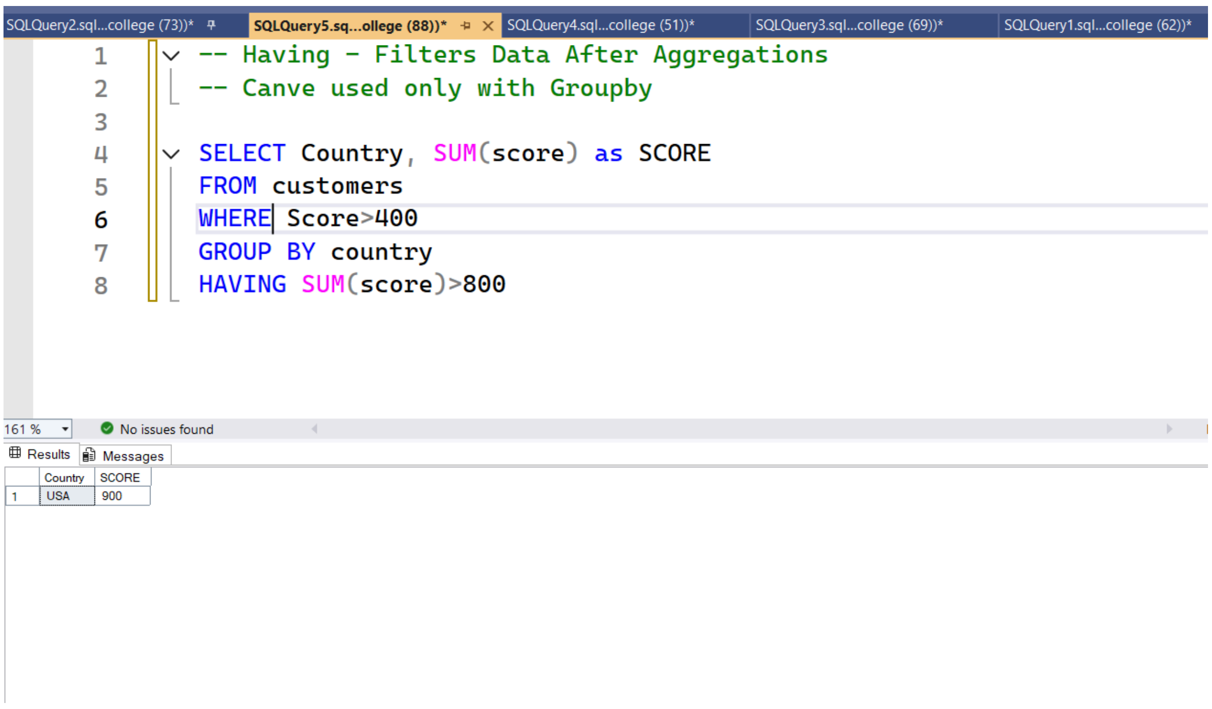
```sql
    AS (ALIAS) Shorthand name (Label) Assigned to a Column or table in a
SELECT
  country,
  SUM(score) as Total_Score
FROM customers
GROUP BY country
```

177 % No issues found Ln: 12 Ch: 3 TABS CRLF

Results Messages

```
   country   Total_Score
1  Germany   850
2  UK        750
3  USA       900
```

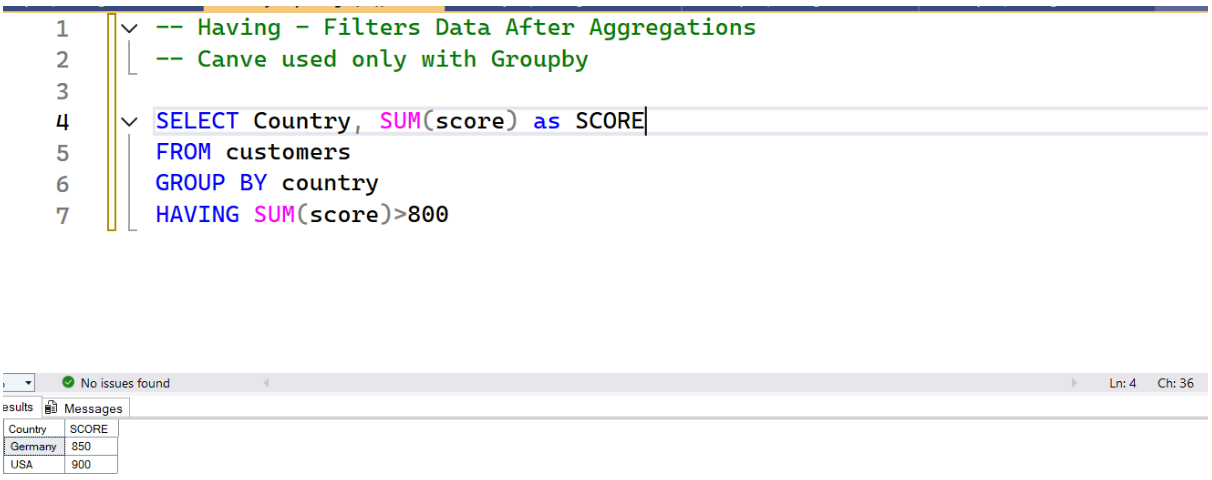
Q) Find the Total Score and Total numbers of Customers for Each Category



SQLQuery2.sql...college (73))*
SQLQuery5.sq...ollege (88))*
SQLQuery4.sql...college (51))*
SQLQuery3.sql...college (69))*
SQLQuery1.sql...college (62))*
161 %
No issues found
Results
Messages

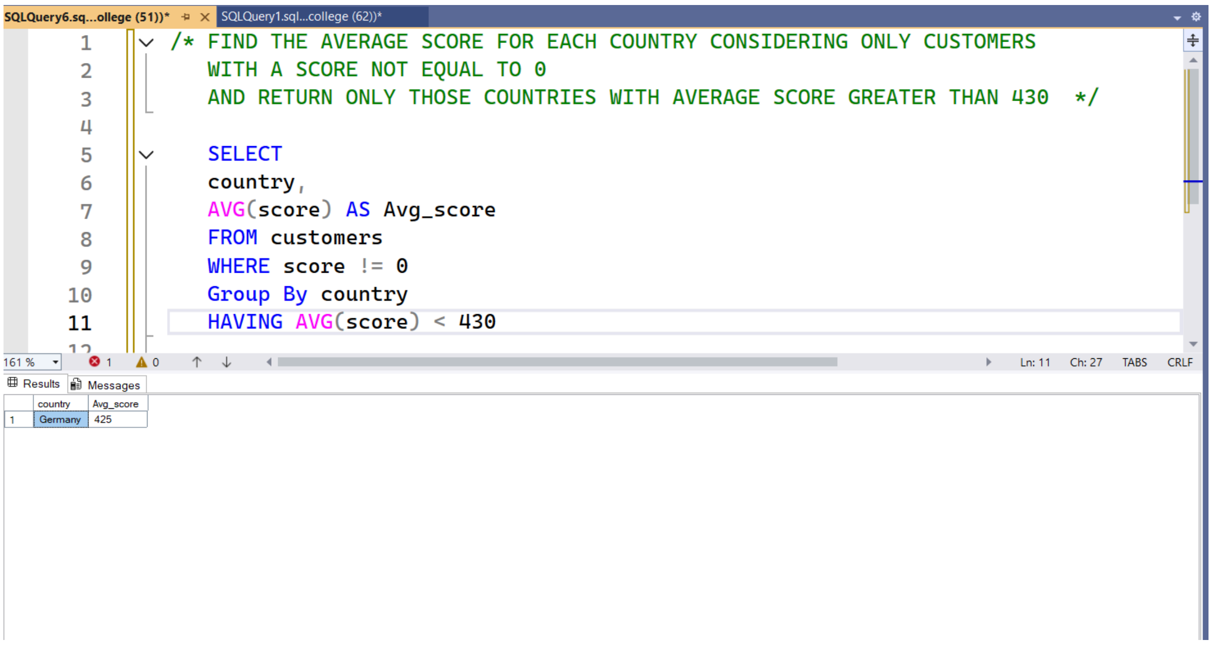
```
-- Having - Filters Data After Aggregations
-- Canve used only with Groupby

SELECT Country, SUM(score) as SCORE
FROM customers
WHERE Score>400
GROUP BY country
HAVING SUM(score)>800
```

	Country	SCORE
1	USA	900



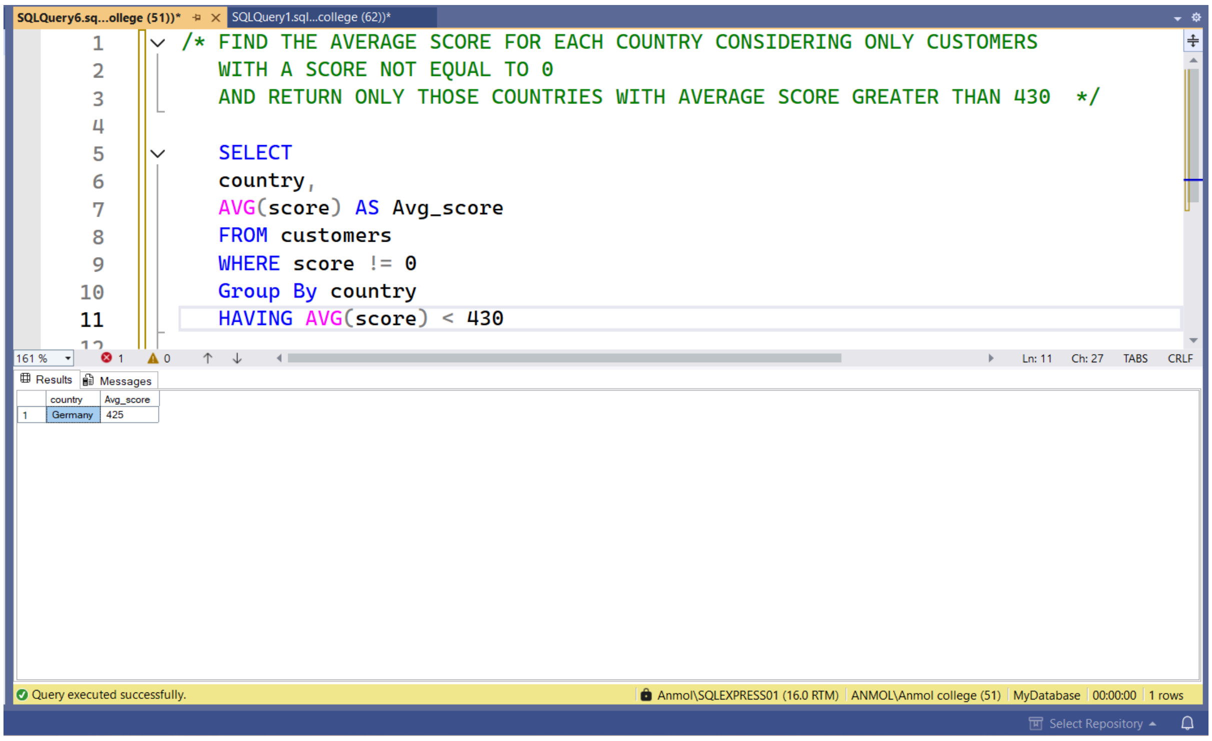
```
-- Having - Filters Data After Aggregations
-- Canve used only with Groupby

SELECT Country, SUM(score) as SCORE
FROM customers
GROUP BY country
HAVING SUM(score)>800
```

Country	SCORE
Germany	850
USA	900



```sql
/* FIND THE AVERAGE SCORE FOR EACH COUNTRY CONSIDERING ONLY CUSTOMERS
   WITH A SCORE NOT EQUAL TO 0
   AND RETURN ONLY THOSE COUNTRIES WITH AVERAGE SCORE GREATER THAN 430  */

   SELECT
   country,
   AVG(score) AS Avg_score
   FROM customers
   WHERE score != 0
   Group By country
   HAVING AVG(score) < 430
```

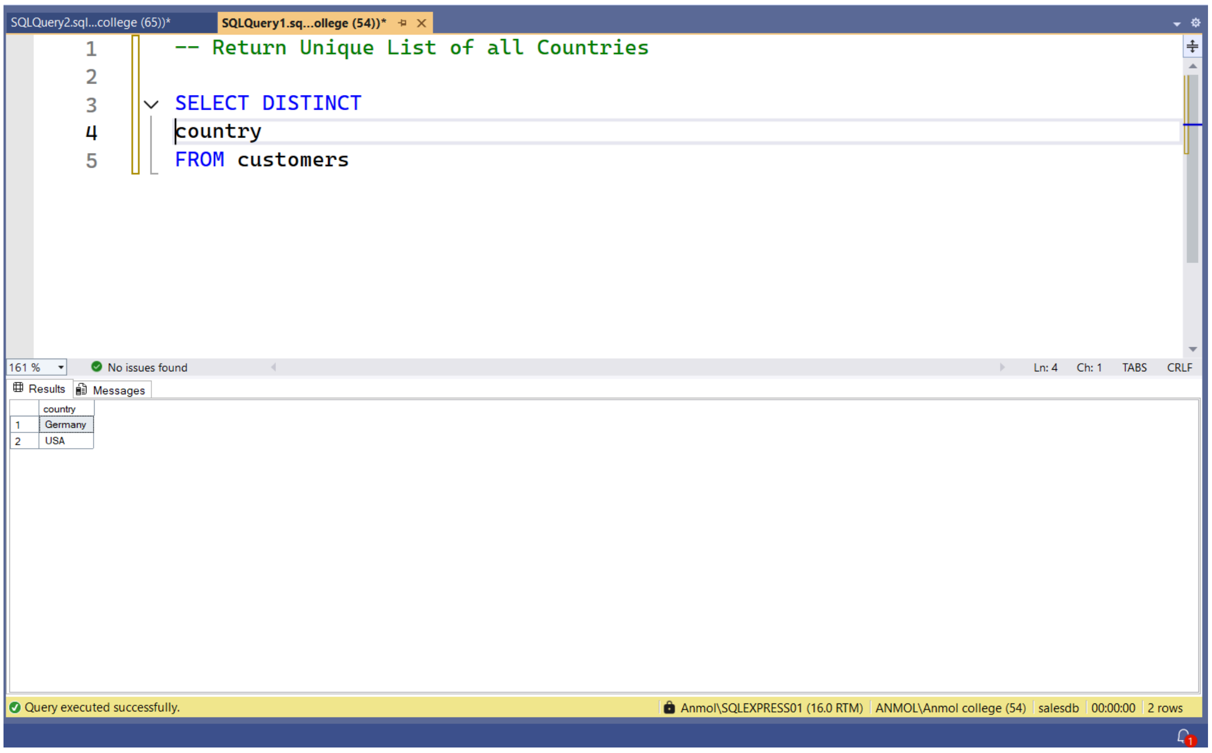
	country	Avg_score
1	Germany	425



```sql
/* FIND THE AVERAGE SCORE FOR EACH COUNTRY CONSIDERING ONLY CUSTOMERS
   WITH A SCORE NOT EQUAL TO 0
   AND RETURN ONLY THOSE COUNTRIES WITH AVERAGE SCORE GREATER THAN 430  */

   SELECT
   country,
   AVG(score) AS Avg_score
   FROM customers
   WHERE score != 0
   Group By country
   HAVING AVG(score) < 430
```

	country	Avg_score
1	Germany	425

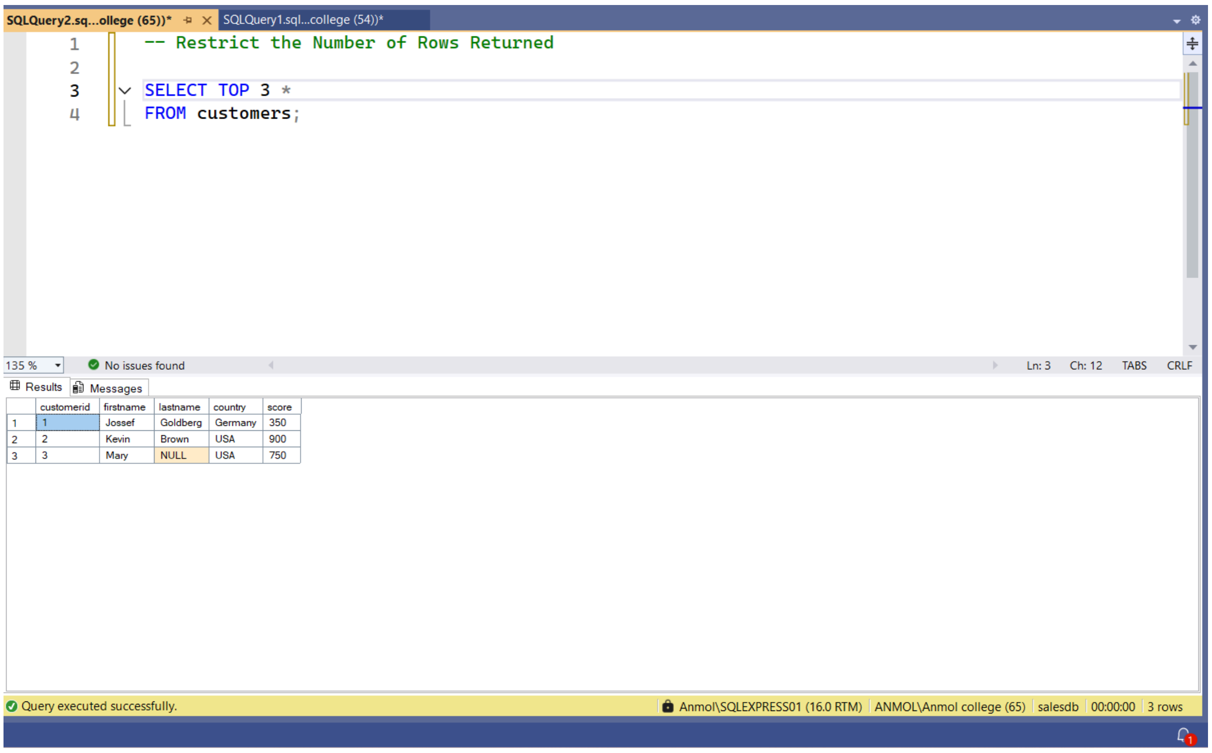
Query executed successfully. Anmol\SQLEXPRESS01 (16.0 RTM) ANMOL\Anmol college (51) MyDatabase 00:00:00 1 rows



```sql
-- Return Unique List of all Countries

SELECT DISTINCT
country
FROM customers
```

	country
1	Germany
2	USA

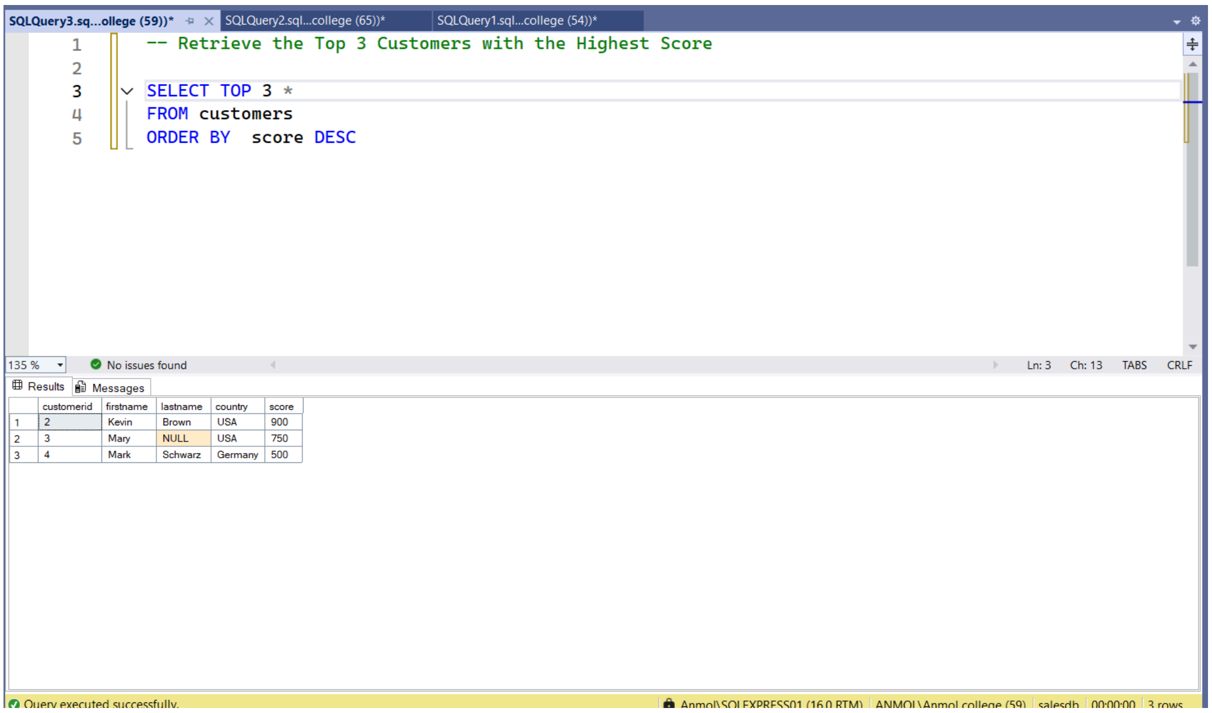
Query executed successfully. Anmol\SQLEXPRESS01 (16.0 RTM) | ANMOL\Anmol college (54) | salesdb | 00:00:00 | 2 rows



```sql
-- Restrict the Number of Rows Returned

SELECT TOP 3 *
FROM customers;
```

	customerid	firstname	lastname	country	score
1	1	Jossef	Goldberg	Germany	350
2	2	Kevin	Brown	USA	900
3	3	Mary	NULL	USA	750

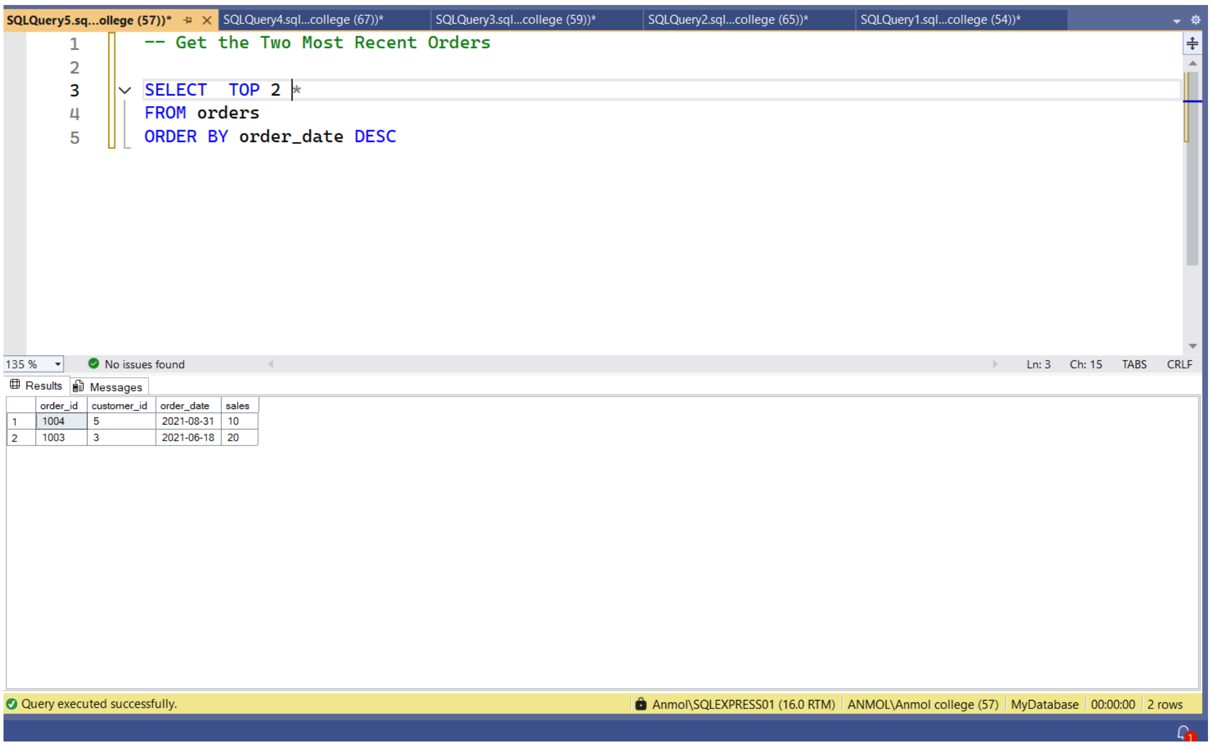
Query executed successfully. Anmol\SQLEXPRESS01 (16.0 RTM) | ANMOL\Anmol college (65) | salesdb | 00:00:00 | 3 rows



```sql
-- Retrieve the Top 3 Customers with the Highest Score

SELECT TOP 3 *
FROM customers
ORDER BY  score DESC
```

	customerid	firstname	lastname	country	score
1	2	Kevin	Brown	USA	900
2	3	Mary	NULL	USA	750
3	4	Mark	Schwarz	Germany	500

```
Query executed successfully.    Anmol\SQLEXPRESS01 (16.0 RTM) | ANMOL\Anmol college (59) | salesdb | 00:00:00 | 3 rows
```



```sql
-- Get the Two Most Recent Orders

SELECT  TOP 2 *
FROM orders
ORDER BY order_date DESC
```

	order_id	customer_id	order_date	sales
1	1004	5	2021-08-31	10
2	1003	3	2021-06-18	20

```
Query executed successfully.    Anmol\SQLEXPRESS01 (16.0 RTM) | ANMOL\Anmol college (57) | MyDatabase | 00:00:00 | 2 rows
```

SQL QUERY

- SELECT
- DISTINCT
- TOP
- FROM
- WHERE
- GROUP BY
- HAVING
- ORDER BY

Execution Order and Coding Order

```
SELECT DISTINCT TOP2
   Col 1,
   SUM(col 2)
FROM Table
WHERE Col = 10
GROUP BY Col 1
HAVING SUM(Col2)>30
ORDER BY Col1 ASC
```

CODING ORDER

Execute Order

1) FROM
2) WHERE
3) GROUP BY
4) HAVING
5) SELECT DISTINCT
6) ORDER BY
7) TOP

If we Select a particular part of a code then only that selected part will be Execute.