



SELECT *Category*
Country, *Aggregation*
SUM(score)
FROM Table
GROUP BY *Country*



	id	name	Country	Score
1	Maria	Germany	350	
2	John	USA	900	
3	Georg	UK	750	
4	Martin	Germany	500	
5	Peter	USA	0	

SELECT

Country,

SUM(score)

① **FROM Table**

② **GROUP By Country**

Germany	350
Germany	500

	id	name	Country	Score
1	Maria	Germany	350	
2	John	USA	900	
3	Georg	UK	750	
4	Martin	Germany	500	
5	Peter	USA	0	

Database

	id	name	Country	Score
1	Maria	Germany	350	
2	John	USA	900	
3	Georg	UK	750	
4	Martin	Germany	500	
5	Peter	USA	0	

SELECT Country, SUM(score)

① FROM Table
② Group By Country

The diagram illustrates a query process. A yellow arrow points from the 'Country' column of the original table to a sub-table where rows for 'Germany' and 'USA' are grouped. Another yellow arrow points from the 'Score' column of the original table to a sub-table where rows for 'Germany' and 'UK' are grouped. Handwritten annotations show the sum of scores for each group: '350+500' for the Germany group and '350+500' for the USA group.

Database

	id	name	Country	Score
1	Maria	Germany	350	
2	John	USA	900	
3	Georg	UK	750	
4	Martin	Germany	500	
5	Peter	USA	0	

SELECT Country, SUM(score)

① FROM Table
② Group By Country

This diagram shows a similar query process to the first one. A yellow arrow points from the 'Country' column to a sub-table with 'Germany' and 'USA'. Another yellow arrow points from the 'Score' column to a sub-table with 'Germany' and 'UK'. Handwritten annotations show the sum of scores for each group: '350+500' for the Germany group and '350+500' for the USA group.

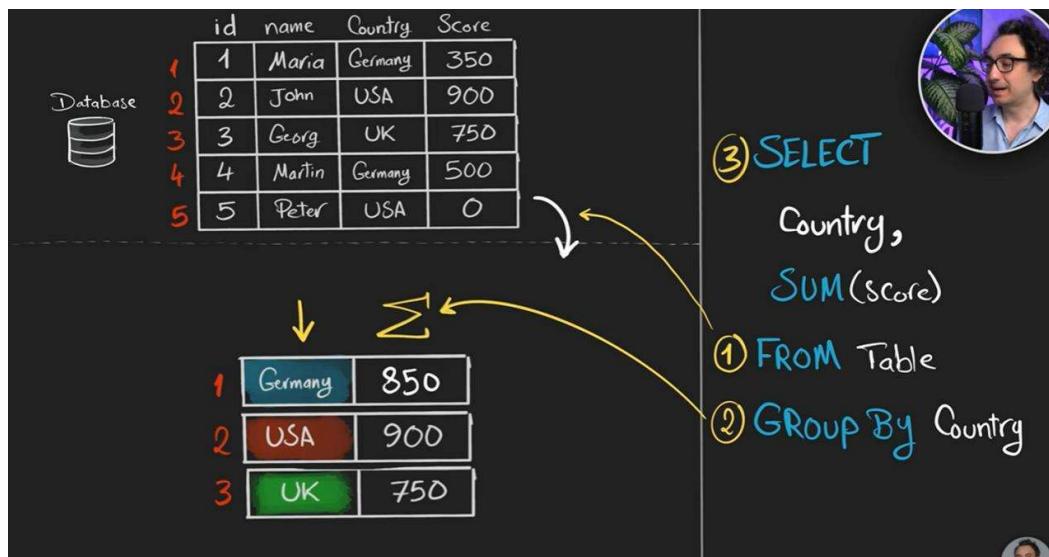
Database

	id	name	Country	Score
1	Maria	Germany	350	
2	John	USA	900	
3	Georg	UK	750	
4	Martin	Germany	500	
5	Peter	USA	0	

SELECT Country, SUM(score)

① FROM Table
② Group By Country

This diagram shows the final step of the query process. A yellow arrow points from the 'Country' column to a sub-table with 'Germany', 'USA', and 'UK'. Another yellow arrow points from the 'Score' column to a sub-table with the same three countries. Handwritten annotations show the sum of scores for each group: '850' for the Germany group, '900' for the USA group, and '750' for the UK group.



```
-- Find the total score for each country

SELECT
    country,
    SUM(score)
FROM customers
GROUP BY country
```

Results

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

Object Explorer

Connect ▾ DESKTOP-848QBU\SQLEXPRESS (SQL Server 1)

Databases

System Databases

AdventureWorks2022

AdventureWorksDW2022

MyDatabase

Tables

System Tables

FileTables

External Tables

Graph Tables

dbo.customers

dbo.orders

Views

External Resources

Synonyms

Programmability

Query Store

Service Broker

Storage

Security

SalesDB

Se

Se

Re

-- Find the total score for each country

AS (ALIAS)

```
SELECT
    country,
    SUM(score)
FROM customers
GROUP BY country
```

Results Messages

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

shorthand name (Label) assigned to a column or table in a query

-- Find the total score for each country

```
1
-- Find the total score for each country

SELECT
    country,
    SUM(score) AS total_score
FROM customers
GROUP BY country
```

Results Messages

	country	total_score
1	Germany	850
2	UK	750
3	USA	900

```
-- Find the total score for each country
```

```
SELECT
    country AS customer_country,
    SUM(score) AS total_score
FROM customers
GROUP BY country
```

Results Messages

customer_country	total_score
Germany	850
UK	750
USA	900

New Query Execute

Object Explorer

DESKTOP-84B8QBU\SQLEXPRESS (SQL Server 1)

Databases

MyDatabase

Tables

customer

first_name

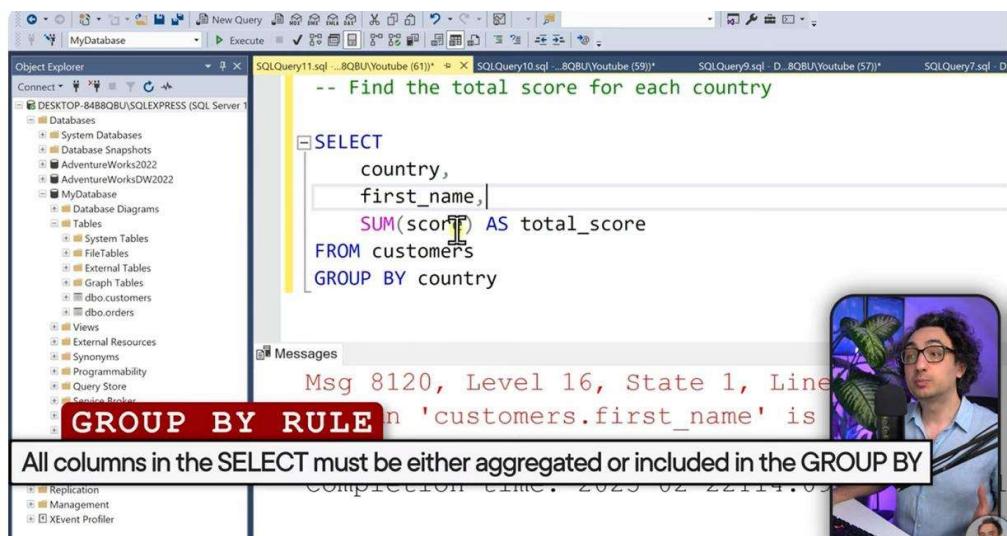
score

-- Find the total score for each country

```
SELECT
    country,
    first_name,
    SUM(score) AS total_score
FROM customers
GROUP BY country
```

Msg 8120, Level 16, State 1, Line 1
All columns in the SELECT must be either aggregated or included in the GROUP BY

Completion time: 2023-02-22T14:05:20



```

SQLQuery11.sql - ...8QBU\Youtube (61)*  SQLQuery10.sql - ...8QBU\Youtube (59)*  SQLQuery9.sql - D...8QBU\Youtube (57)*
-- Find the total score for each country

SELECT
    country,
    first_name,
    SUM(score) AS total_score
FROM customers
GROUP BY country,first_name

```

Results

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

```

SQLQuery12.sql - ...8QBU\Youtube (56)  SQLQuery11.sql - ...8QBU\Youtube (61)*  SQLQuery10.sql - ...8QBU\Youtube (59)*  SQLQuery9.sql - D...8QBU\Youtube (57)*
-- Find the total score and total number of customers for each country

SELECT
    country,
    SUM(score) AS total_score,
    COUNT(id) AS total_customers
FROM customers
GROUP BY country

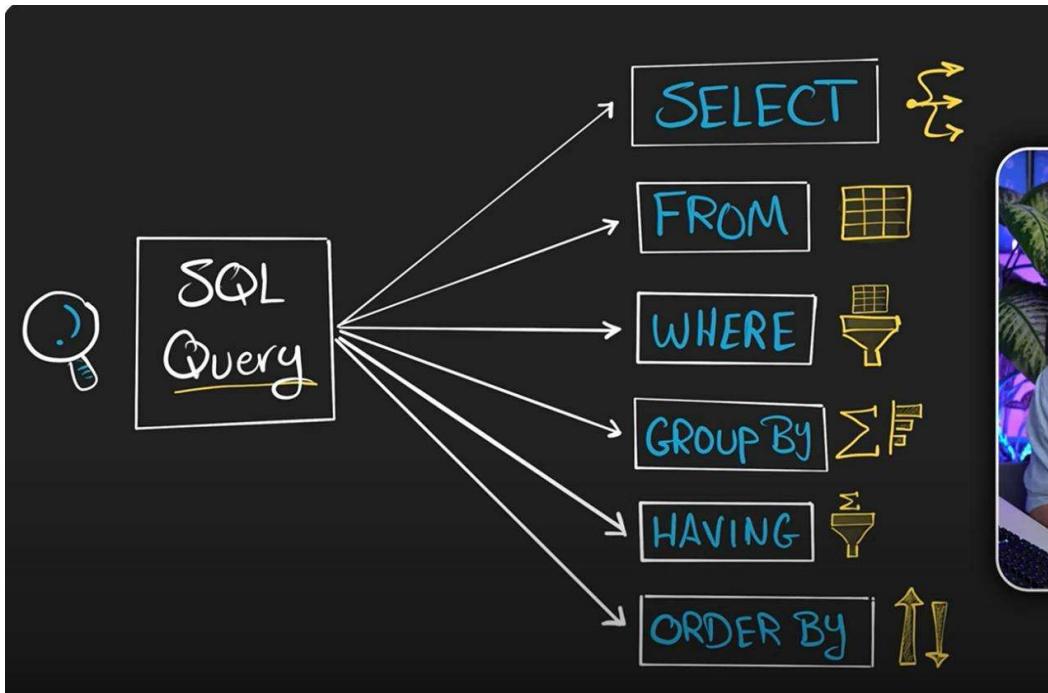
```

Results

	country	total_score	total_customers
1	Germany	850	2
2	UK	750	1
3	USA	900	2



Group by works with aggregate function without mentioning them into group by clause



A circular inset photo of a man with glasses and a microphone is located in the top-left corner of the slide.

	id	name	Country	Score
1	Maria	Germany	350	
2	John	USA	900	
3	Georg	UK	750	
4	Martin	Germany	500	
5	Peter	USA	0	

Having
Filters Data **After** Aggregation
 Can be **used** only with **Group By**

SELECT

Country,

SUM(score)

FROM Table

GROUP BY Country

HAVING SUM(score) > 800

SELECT
Country,
SUM(score)
FROM Table
GROUP BY Country
HAVING Condition



	id	name	Country	Score
1	Maria		Germany	350
2	John		USA	900
3	Georg		UK	750
4	Martin		Germany	500
5	Peter		USA	0

	id	name	Country	Score
1	Maria		Germany	350
2	John		USA	900
3	Georg		UK	750
4	Martin		Germany	500
5	Peter		USA	0

Germany	850
USA	900
UK	750

SELECT
Country,
SUM(score)

- ① FROM Table
- ② Group By Country
- ③ HAVING SUM(score) > 800

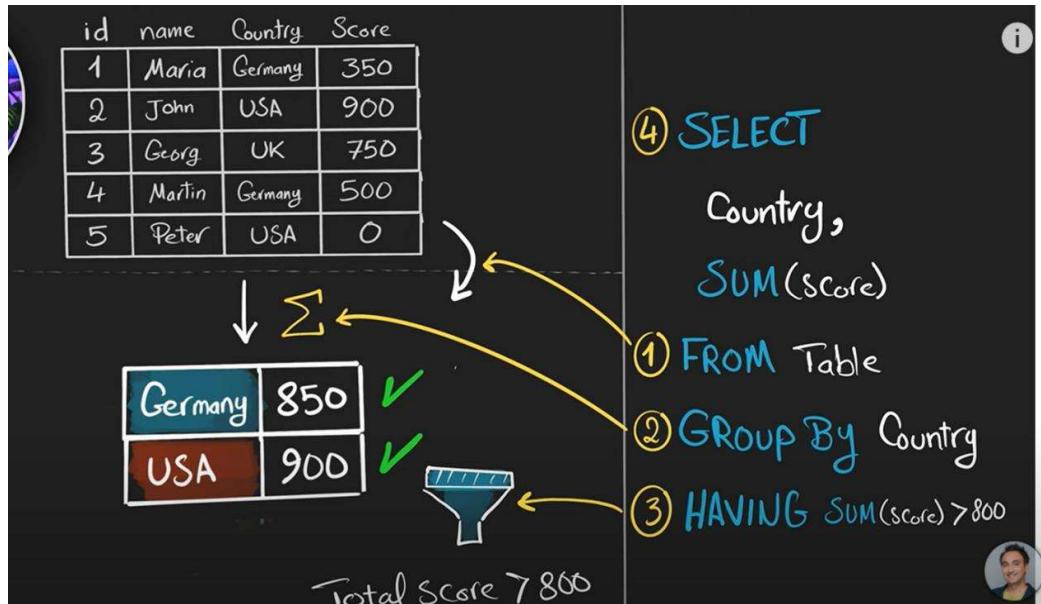
	id	name	Country	Score
1	Maria		Germany	350
2	John		USA	900
3	Georg		UK	750
4	Martin		Germany	500
5	Peter		USA	0

Germany	850
USA	900
UK	750

Total Score > 800

SELECT
Country,
SUM(score)

- ① FROM Table
- ② Group By Country
- ③ HAVING SUM(score) > 800



SELECT

Country,

SUM(score)

FROM Table

WHERE Score > 400

GROUP BY Country

HAVING SUM(score) > 800





	id	name	Country	Score
1	Maria	Germany	350	
2	John	USA	900	
3	Georg	UK	750	
4	Martin	Germany	500	
5	Peter	USA	0	

SELECT

Country,

SUM(score)

FROM Table

WHERE Score > 400

GROUP BY Country

HAVING SUM(score) > 800



Filter
Your Data

① Before
Aggregation

② After
Aggregation



	id	name	Country	Score
1	Maria	Germany	350	
2	John	USA	900	
3	Georg	UK	750	
4	Martin	Germany	500	
5	Peter	USA	0	

	id	name	Country	Score
1	Maria	Germany	350	X
2	John	USA	900	✓
3	Georg	UK	750	✓
4	Martin	Germany	500	✓
5	Peter	USA	0	X

Before Aggregation



SELECT

Country,

SUM(score)

FROM Table

WHERE Score > 400

GROUP BY Country

HAVING SUM(score) > 800



	id	name	Country	Score
1	Maria	Germany	350	
2	John	USA	900	
3	Georg	UK	750	
4	Martin	Germany	500	
5	Peter	USA	0	

	id	name	Country	Score
1	Maria	Germany	350	X
2	John	USA	900	✓
3	Georg	UK	750	✓
4	Martin	Germany	500	✓
5	Peter	USA	0	X

SELECT

Country,
SUM(score)

- ① FROM Table
- ② WHERE Score > 400
- ③ GROUP By Country
- ④ HAVING SUM(score) > 800



	id	name	Country	Score
1	Maria	Germany	350	
2	John	USA	900	
3	Georg	UK	750	
4	Martin	Germany	500	
5	Peter	USA	0	

	SUM(score)	
USA	900	✓
UK	750	X
Germany	500	X

SELECT

Country,
SUM(score)

- ① FROM Table
- ② WHERE Score > 400
- ③ GROUP By Country
- ④ HAVING SUM(score) > 800



	id	name	Country	Score
1	Maria	Germany	350	
2	John	USA	900	
3	Georg	UK	750	
4	Martin	Germany	500	
5	Peter	USA	0	

Final Result

USA	900
-----	-----

SELECT

Country,
SUM(score)

- ① FROM Table
- ② WHERE Score > 400
- ③ GROUP By Country
- ④ HAVING SUM(score) > 800

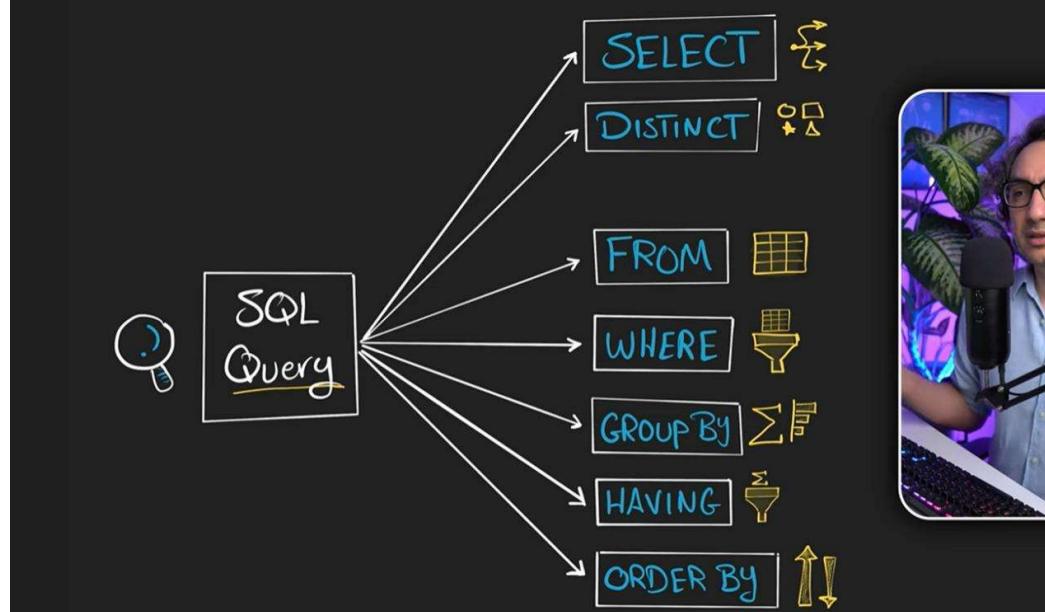
Having execute after aggregation - works on aggregation functions after group by

Where execute before aggregation

```
/* Find the average score for each country  
considering only customers with a score not equal to 0  
and return only those countries with an average score greater than 430 */  
  
SELECT  
    country,  
    AVG(score) AS avg_score  
FROM customers  
WHERE score != 0  
GROUP BY country  
HAVING AVG(score) > 430
```

Results Messages

	country	avg_score
1	UK	750
2	USA	900





Database

	id	name	Country	Score
1	Maria	Germany	350	
2	John	USA	900	
3	Georg	UK	750	
4	Martin	Germany	500	
5	Peter	USA	0	

Distinct

Removes Duplicates (Repeated Values)

each Value appears only Once

↑ ↴

SELECT DISTINCT

Col

FROM Table

Database

	id	name	Country	Score
1	Maria	Germany	350	
2	John	USA	900	
3	Georg	UK	750	
4	Martin	Germany	500	
5	Peter	USA	0	

① **FROM Table**

② **SELECT DISTINCT Country**

	name	Country	Score
1	Maria	Germany	350
2	John	USA	900
3	Georg	UK	750
4	Martin	Germany	500
5	Peter	USA	0

Database

	id	name	Country	Score
1	Maria	Germany	350	
2	John	USA	900	
3	Georg	UK	750	
4	Martin	Germany	500	
5	Peter	USA	0	

① **FROM Table**

② **SELECT DISTINCT Country**

③ **③**

	Country
1	Germany
2	USA
3	UK
4	Germany
5	USA

```
-- Return Unique list of all countries
```

```
SELECT DISTINCT country  
FROM customers
```

	country
1	Germany
2	UK
3	USA

Object Explorer SQLQuery6.sql - D... 8QBU\Youtube (51)*

```
-- Return Unique list of all countries  
SELECT DISTINCT country  
FROM customers
```

Bad Habit with DISTINCT

Don't use DISTINCT unless it's necessary; it can slow down your query

Results Messages

	country
1	Germany
2	UK
3	USA

43:47 / 54:49 • DISTINCT >

```
SQLQuery6.sql - D...8QBU\Youtube (51))*  ✎ X  
-- Return Unique li  
  
SELECT | id  
FROM customers
```

Results Messages

	id
1	1
2	2
3	3
4	4
5	5

43:56 / 54:49 • DISTINCT

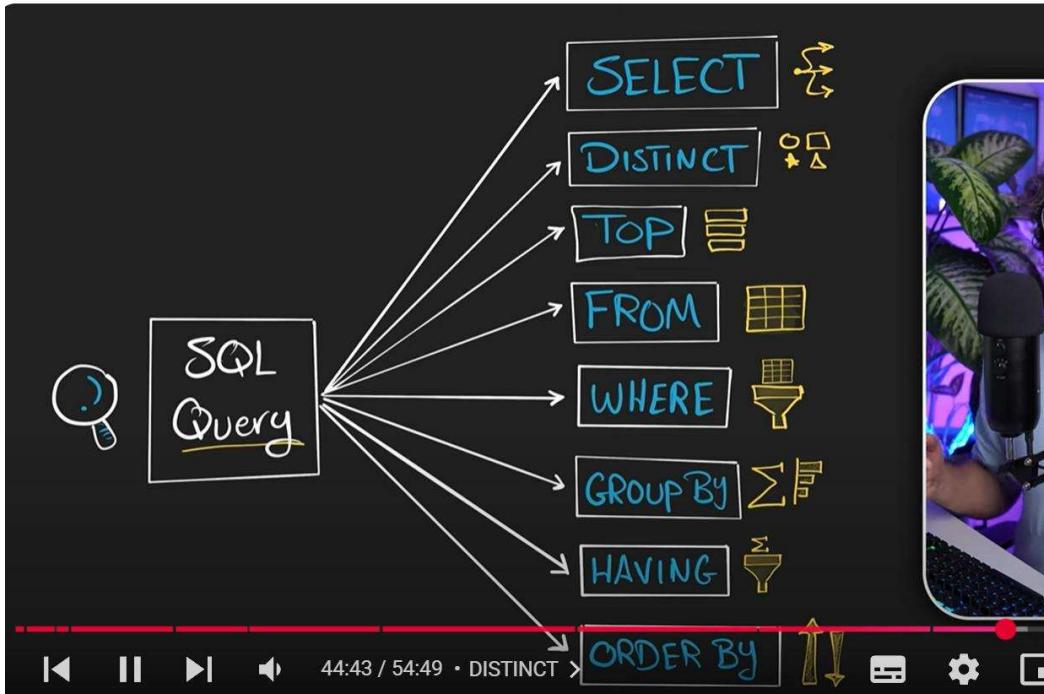
SQLQuery6.sql - D:\8QBU\Youtube (51)*

```
-- Return Unique list of all countries
SELECT DISTINCT id
FROM customers
```

Results Messages

	id
1	1
2	2
3	3
4	4
5	5

Distinct makes the query too slow so do not use when not required



Top is used to limit your data

Database

	<u>id</u>	<u>name</u>	<u>Country</u>	<u>Score</u>
	1	Maria	Germany	350
	2	John	USA	900
	3	Georg	UK	750
	4	Martin	Germany	500
	5	Peter	USA	0

TOP (Limit)



Restrict the Number of Rows Returned

SELECT TOP 3



FROM Table

	id	name	Country	Score
1	Maria	Germany	350	
2	John	USA	900	
3	Georg	UK	750	
4	Martin	Germany	500	
5	Peter	USA	0	

Database



Row 1 →	id	name	Country	Score
Row 2 →	1	Maria	Germany	350
Row 3 →	2	John	USA	900
Row 4 →	3	Georg	UK	750
Row 5 →	4	Martin	Germany	500
Row 6 →	5	Peter	USA	0

Database



Row 1 →	id	name	Country	Score
Row 2 →	1	Maria	Germany	350
Row 3 →	2	John	USA	900
Row 4 →	3	Georg	UK	750

③
② SELECT TOP 3
*
① FROM Table



③
② SELECT TOP 3
*
① FROM Table



```
-- Retrieve the Top 3 Customers with the Highest Scores
```

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

Results Messages

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

```
-- Retrieve the Lowest 2 Customers based on the score
```

```
SELECT TOP 2 *
FROM customers
ORDER BY score ASC
```

Results Messages

		id	first_name	country	score
1		5	Peter	USA	0
2		1	Maria	Germany	350

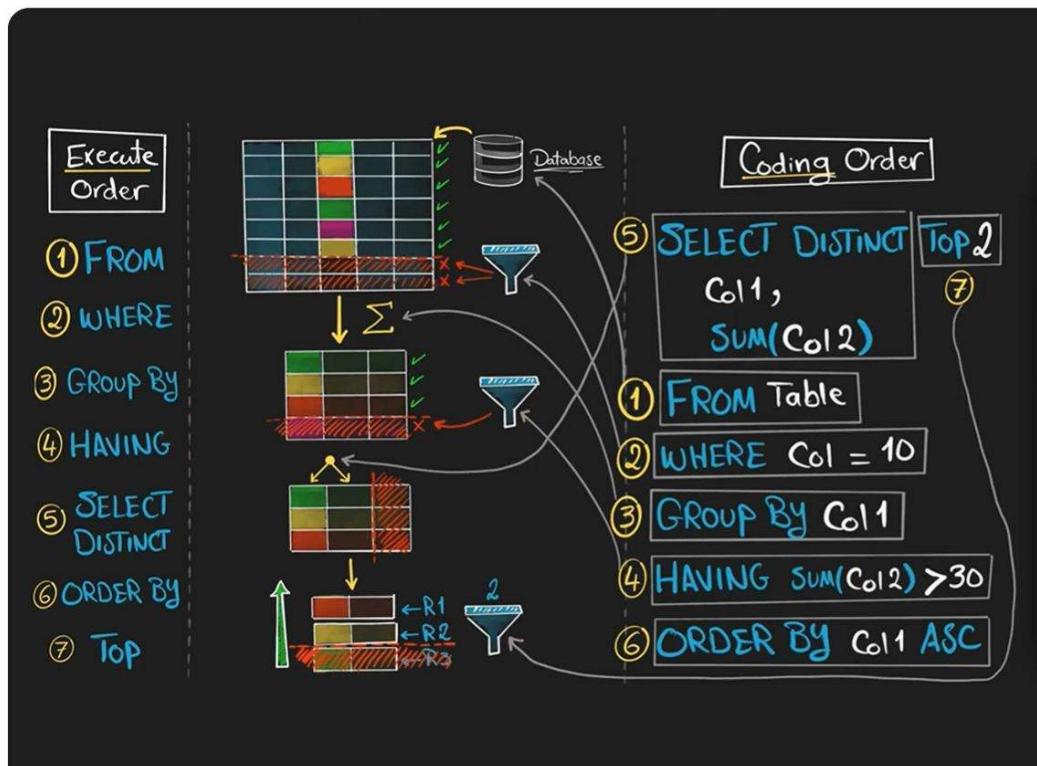
-- Get the Two Most Recent Orders

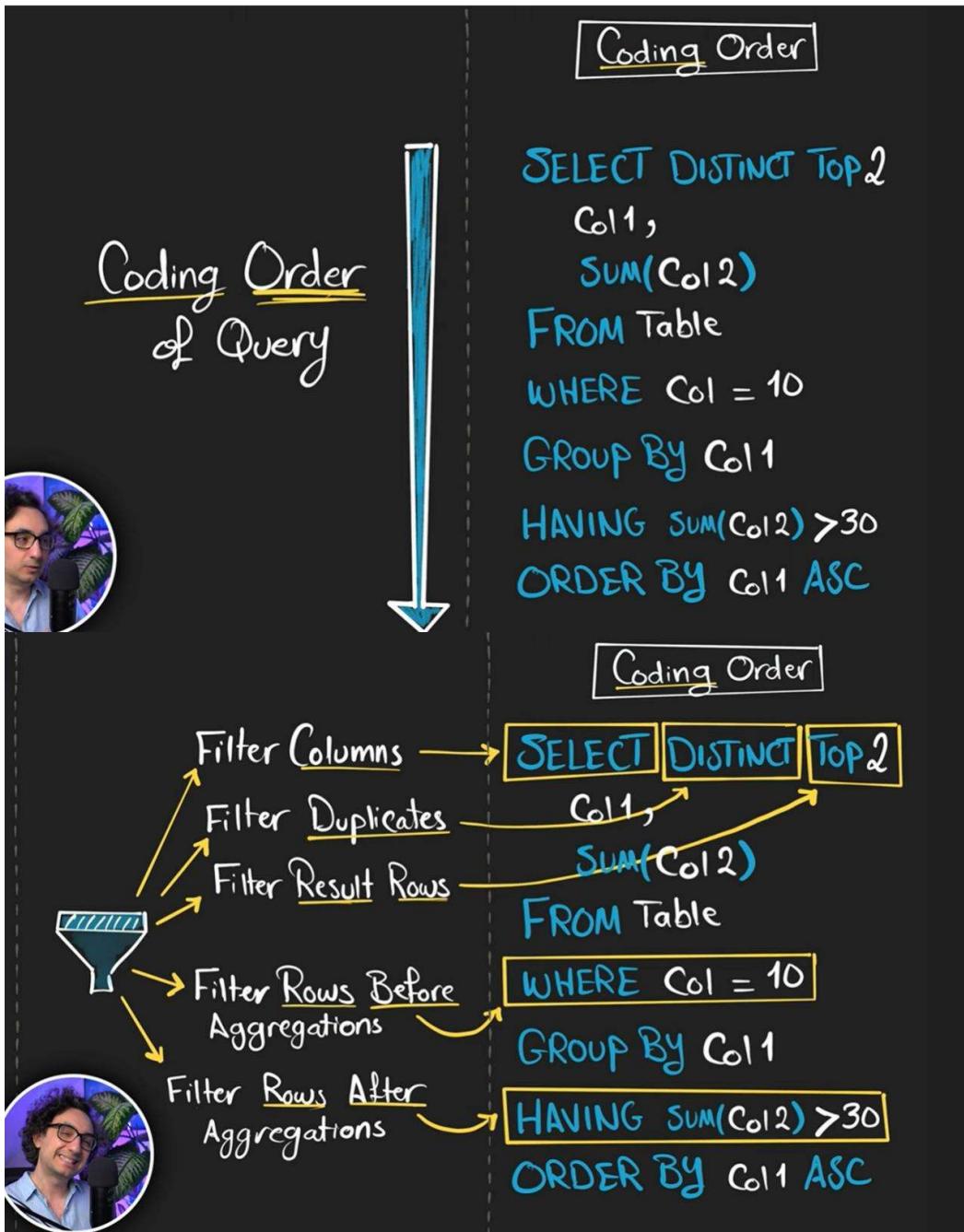
```
SELECT TOP 2 *
FROM orders
ORDER BY order_date DESC
```

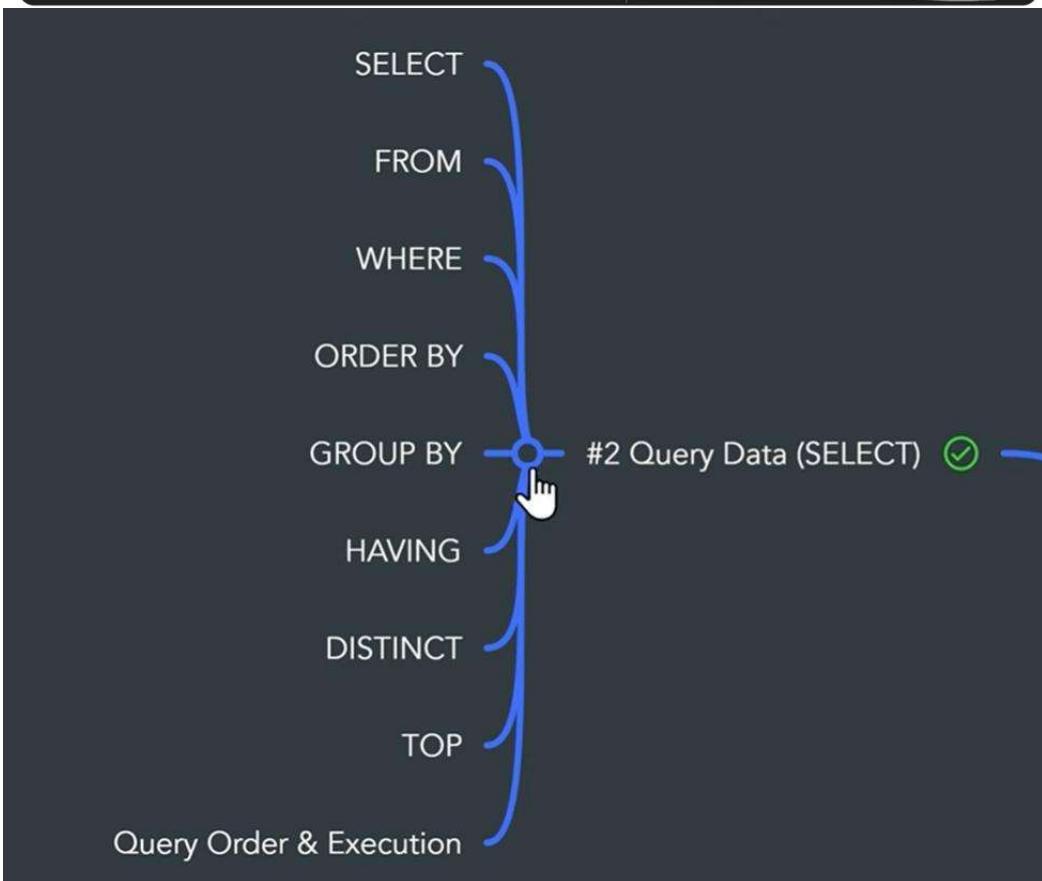
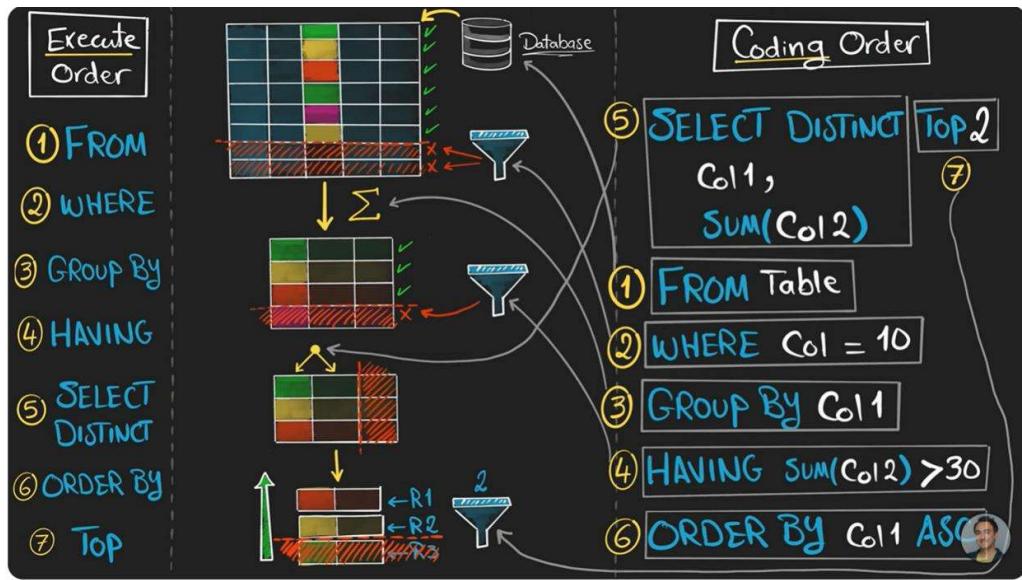


Results Messages

	order_id	customer_id	order_date	sales
1	1004	6	2021-08-31	10
2	1003	3	2021-06-18	20



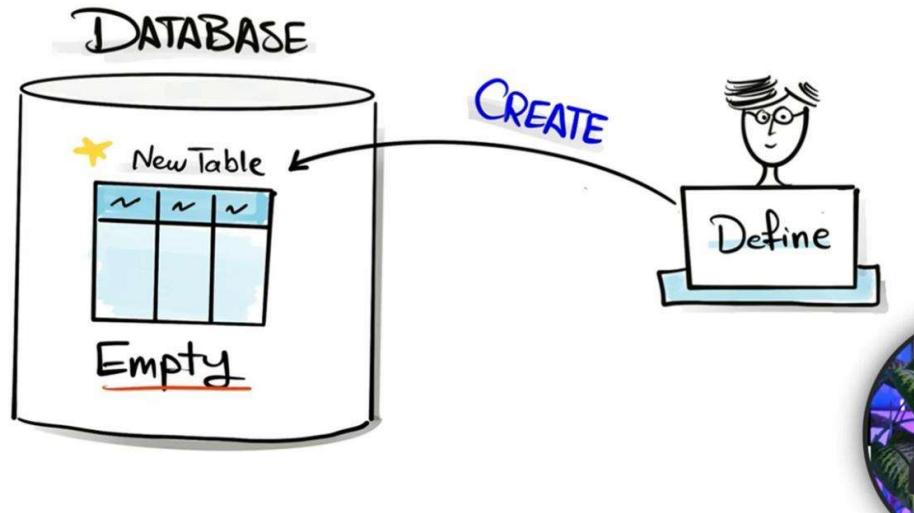




Complete This

DDL Commands

Define Structure of Your Data



```
/* Create a new table called persons  
with columns: id, person_name, birth_date, and phone */  
  
CREATE TABLE persons (  
    id INT NOT NULL  
)
```

Column Name Data Type Constraint

```
/* Create a new table called persons
with columns: id, person_name, birth_date, and phone */

CREATE TABLE persons (
    id INT NOT NULL,
    person_name VARCHAR(50) NOT NULL,
    birth_date DATE,
    phone VARCHAR(15) NOT NULL,
    CONSTRAINT pk_persons PRIMARY KEY (id)
)

192 % < Messages
Commands completed successfully.

Completion time: 2025-02-16T14:08:55.4156439+01:00
```



DDL commands never return data it's changing the structure of your database

BY using direct click of creating table code.

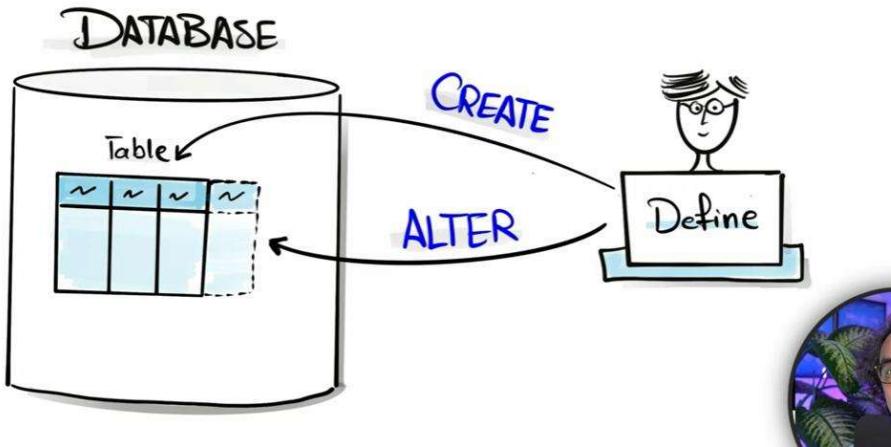
```
USE [MyDatabase]
GO

***** Object: Table [dbo].[persons] Script Date: 16/02/2025 14:13:14 *****/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[persons](
    [id] [int] NOT NULL,
    [person_name] [varchar](50) NOT NULL,
    [birth_date] [date] NULL,
    [phone] [varchar](15) NOT NULL,
    CONSTRAINT [pk_persons] PRIMARY KEY CLUSTERED
(
    [id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
GO
```



Define Structure of Your Data



```
-- Add a new column called email to the persons table
```

```
ALTER TABLE persons  
ADD email VARCHAR(50) NOT NULL
```

Messages
Commands completed successfully.



```
-- Add a new column called email to the persons table
```

```
ALTER TABLE persons  
ADD email VARCHAR(50) NOT NULL
```

```
SELECT * FROM persons
```



ADDING COLUMNS

The new columns are appended at the end of table by default

Results Messages
id person_name birth_date phone email



To add column name in middle we have to drop table and create new one that is bad practice may loose our data already existing inside the table.

```
-- Remove the column phone from the persons table

ALTER TABLE persons
DROP COLUMN phone

SELECT * FROM persons
```

Messages

Commands completed successfully.

Completion time: 2025-02-16T14:27:01.4046064+01:00

```
-- Remove the column phone from the persons table

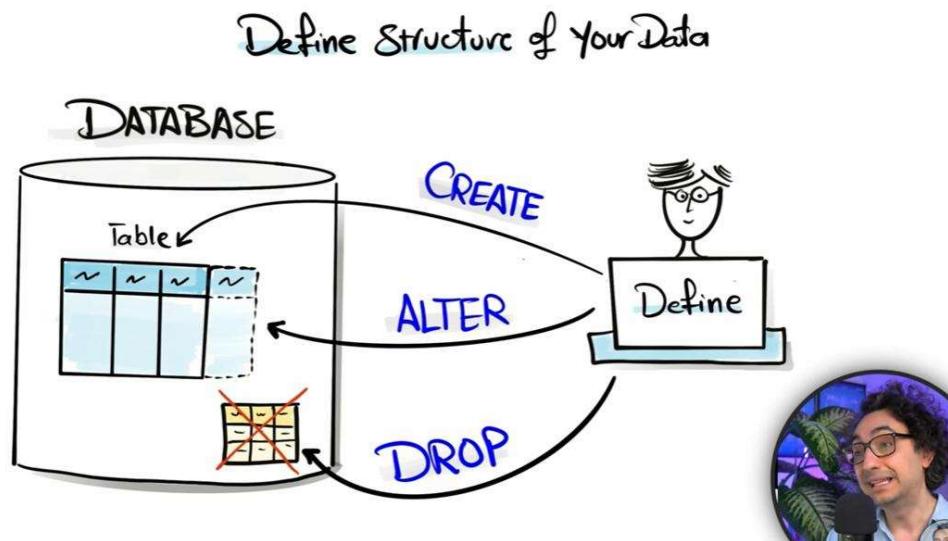
ALTER TABLE persons
DROP COLUMN phone

SELECT * FROM persons
```

Results Messages

id	person_name	birth_date	email
----	-------------	------------	-------

if we delete column we will lose its entire data



Drop remove data and table both from database

```
-- Delete the table persons from the database

DROP TABLE persons

I

Messages
Commands completed successfully.

Completion time: 2025-02-16T14:29:25.3777151

MyDatabase
+ Database Diagrams
+ Tables
  + System Tables
  + FileTables
  + External Tables
  + Graph Tables
  + dbo.customers
  + dbo.orders
+ Views
+ External Resources
+ Synonyms
+ Programmability
+ Query Store
+ Service Broker
+ Storage
+ Security
  SalesDB
+ Security
+ Server Objects
+ Replication
+ Management
+ XEvent Profiler

able persons from the database

ons

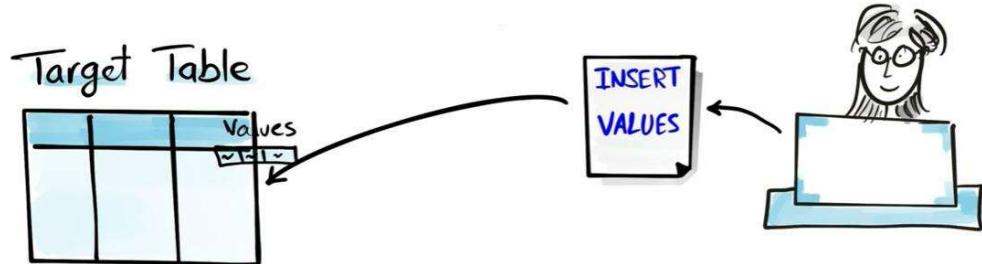
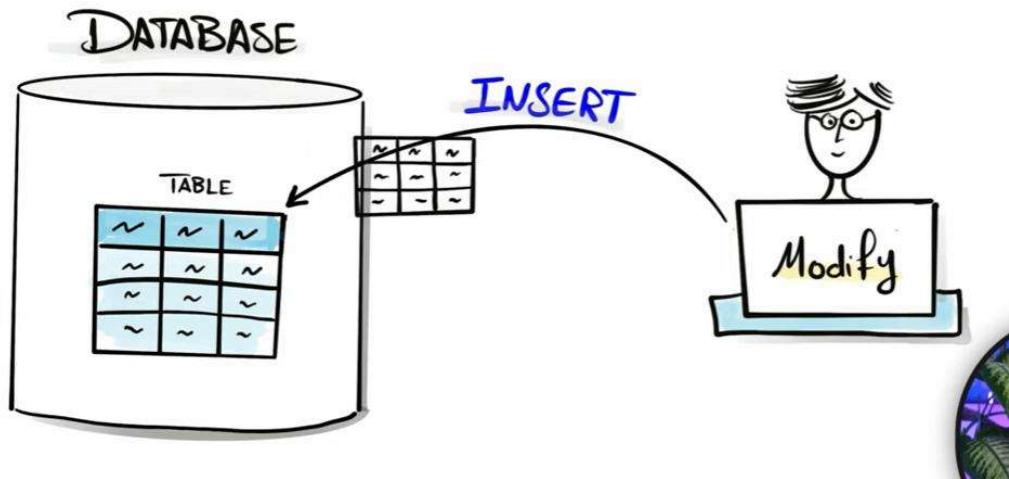
I

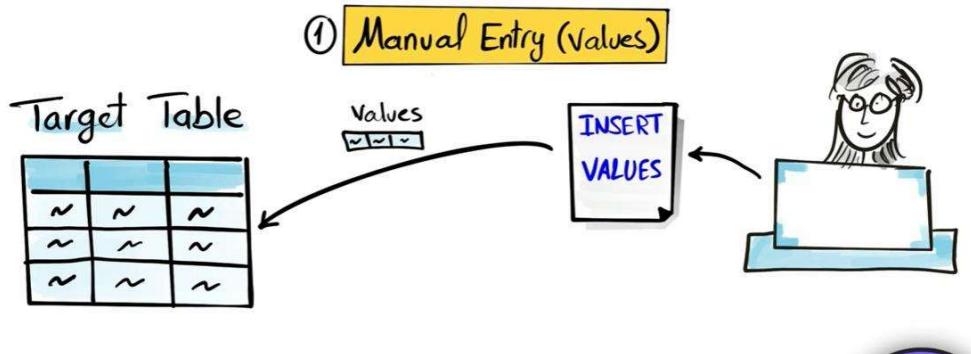
pleted successfully.

ime: 2025-02-16T14:29:25.3777151
```

DML commands:

Modify (Manipulate) Your Data





INSERT Syntax

OPTIONAL: If no columns are specified, SQL expects values for all columns

```
INSERT INTO table_name (column1, column2, column3,...)
VALUES (value1, value2, value3,...)
```

Rule

- Match the number of **columns** and **values**.



INSERT Syntax

OPTIONAL: If no columns are specified, SQL expects values for all columns

```
INSERT INTO table_name (column1, column2, column3,...)
VALUES (value1, value2, value3,...)
,(value1, value2, value3,...) ←----- Multiple Inserts
```

Rule

- Match the number of **columns** and **values**.



```

INSERT INTO customers (id, first_name, country, score)
VALUES
(6, 'Anna', 'USA', NULL),
(NULL, 'Sam', NULL, 100)

```

Messages

```

Msg 515, Level 16, State 2, Line 1
Cannot insert the value NULL into column 'id'
The statement has been terminated.

```

Completion time: 2025-02-16T14:51:03.912631

Primary key error

```

INSERT INTO customers (id, first_name, country, score)
VALUES
(6, 'Anna', 'USA', NULL),
(7, 'Sam', NULL, 100)

```

Messages

```

(2 rows affected)

```

Completion time: 2025-02-16T14:51:24.339695

```

INSERT INTO customers (id, first_name, country, score)
VALUES
(6, 'Anna', 'USA', NULL),
(7, 'Sam', NULL, 100)

```

CAUTION

Columns and values must be in the same order

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
6	6	Anna	USA	NULL
7	7	Sam	NULL	100



Accepted although there is country name in place of name and name in place of country name this is because both are in varchar format

```

INSERT INTO customers (id, first_name, country, score)
VALUES
    (8, 'USA', 'Max', NULL)
SELECT * FROM customers

```

Messages

(1 row affected)

Completion time: 2025-02-16T14:54:39.507605



Due to data type rules and its constraints

```

INSERT INTO customers (id, first_name, country, score)
VALUES
    (8, 'USA', 'Max', NULL)

```

RULES FROM customers

Matching Data Types, Column Count & Constraints

	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
6	6	Anna	USA	NULL
7	7	Sam	NULL	100
8	8	USA	Max	NULL



```

INSERT INTO customers (id, first_name, country, score)
VALUES
    ('Max', 9, 'Max', NULL)

```

RULES FROM customers

Matching Data Types, Column Count & Constraints

Msg 245, Level 16, State 1, Line 1

Conversion failed when converting the varch

Completion time: 2025-02-16T14:55:38.832552

