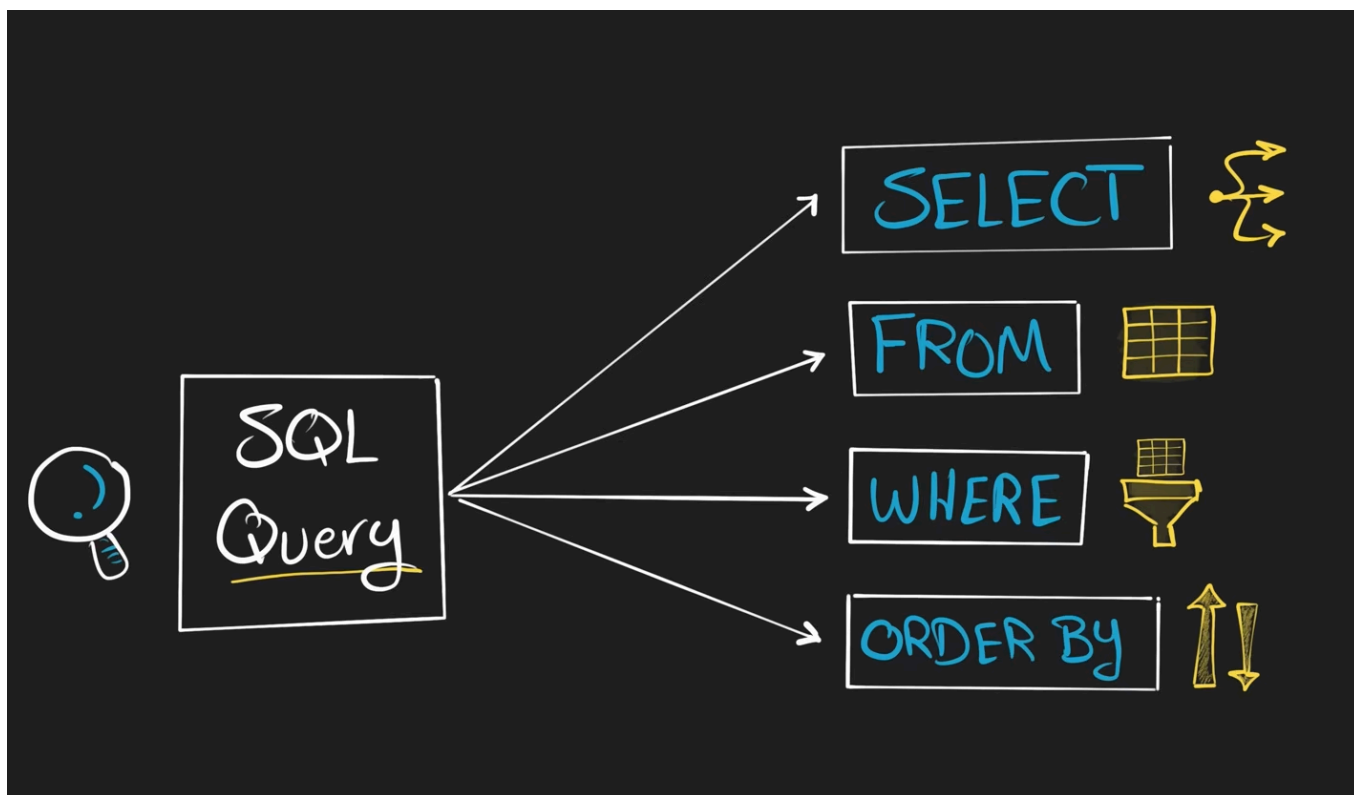


## 7. ORDER BY Clause - Sort the Data



**ORDER BY** Clause is used to sort the Data . Based on the requirement we should be able to decide in which way to sort a data

- **ASC** - Ascending , From lowest to highest
- **DESC** - Descending , From Highest to lowest

### NOTE:

For using ORDER BY , we need to specify 2 things .,

- By which column as reference we are gonna sort it
- Then in which way we want to sort (ASC or DESC)
- If you dont specify anything as Default SQL will use ASC - Ascending

### SYNTAX:

```
SELECT *  
FROM table_name  
ORDER BY ref_column sorting_way
```

### QUERY:

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

#### Best Practice:

- Always try to specify the type of sort (ASC or DESC) than leaving it blank

#### How SQL works here ?

- First , it will read the **FROM** statment and get the required table
- Then it will read **ORDER BY** , read the reference column by which it should sort the data and based on the sorting type it will sort the data based on the reference column
- Then it will read **SELECT** , gives the final result of the query

The diagram illustrates the execution of an SQL query. It starts with a 'Database' icon and a table with 5 rows. The table has columns: id, name, Country, and Score. The data is as follows:

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

Step 1: **SELECT \*** (indicated by a blue arrow pointing to the table).

Step 2: **FROM Table** (indicated by a yellow arrow pointing to the table).

Step 3: **ORDER BY Score DESC** (indicated by a yellow arrow pointing to the 'Score' column). The scores are circled in red in the resulting table, and a large red arrow points downwards from 'Highest' to 'Lowest', indicating descending order.

The resulting sorted table is:

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

A small circular inset shows a person speaking into a microphone.

## NESTED ORDER BY

In SQL , we can sort the the data by multiple columns

For example lets say we want to sort the data using a column (lets say , Country) , now again we want to sort the sorted data based on another column (Lets say , score) . In this type of scenarios we will be using **Nested Order By**

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

If you see here the data is ordered by country but the score are not ordered . This type of inconsistent sorting happens when there are repetition of values in a column

## SYNTAX

```
SELECT *
FROM table_name
ORDER BY
ref_col1 sort_type
ref_col2 sort_type
```

## QUERY

```
SELECT *
FROM customers
ORDER BY
country ASC,
score DESC
```

Here, SQL will first sort the data by the given 1st condition in ORDER BY ie , it will sort the data based on country first , then it will again sort the sorted data by score.

Practice Statements:

-- Retrieve all customers and sort the results by the highest score first

SELECT

\*

FROM customers

ORDER BY score DESC

```
-- Retrieve all customers and sort the results by the highest score first
SELECT
*
FROM customers
ORDER BY score DESC
```

100 %

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

-- Retrieve all customers and sort the results by the lowest score first but filter out anyone with 0 score

SELECT

\*

FROM customers

WHERE score != 0

ORDER BY score ASC

```
-- Retrieve all customers and sort the results by the lowest score first but filter out anyone with 0 score
SELECT
*
FROM customers
WHERE score != 0
ORDER BY score ASC
```

100 %

Results Messages

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

-- Retrieve all customers and sort the results by country and then by the highest score

USE MyDatabase

SELECT

\*

FROM customers

ORDER BY

country ASC,

score DESC

```
-- Retrieve all customers and sort the results by country and then by the highest score
```

```
USE MyDatabase
```

```
SELECT
```

```
*
```

```
FROM customers
```

```
ORDER BY
```

```
country ASC,
```

```
score DESC
```

100 %

Results Messages

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