# SQL:

## DATA QUERY LANGUAGE

### USE DATABASE:

```
USE MyDataBase;
```

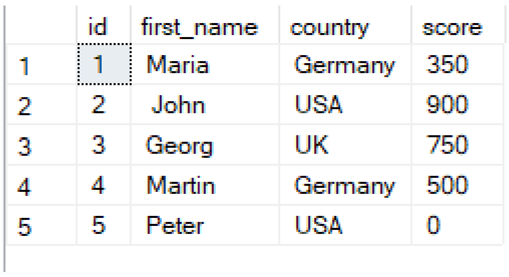
### COMMENTS:

```
--This is a comment
/* This
is
a
comment*/
```

### SELECT:

QUERY 1: RETRIEVE ALL THE CUSTOMERS DATA

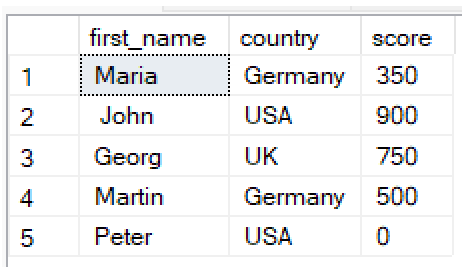
```
SELECT * FROM customers;
```



| | 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 |

QUERY 2: RETRIVE EACH CUSTOMER NAME, COUNTRY AND SCORE

```
SELECT
       first_name,
       country,
       score
FROM customers
```

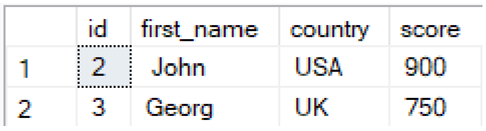


| | first_name | country | score |
|---|---|---|---|
| 1 | Maria | Germany | 350 |
| 2 | John | USA | 900 |
| 3 | Georg | UK | 750 |
| 4 | Martin | Germany | 500 |
| 5 | Peter | USA | 0 |

**WHERE:** Filters your data based on a condition

QUERY 1: RETRIEVE CUSTOMERS DATA WHO SCORED MORE THAN 500.

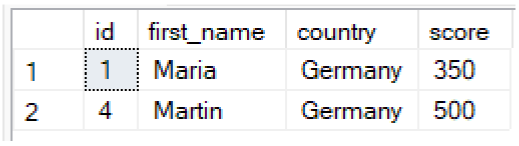
```
SELECT *
FROM customers
WHERE score>500
```



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

<u>QUERY 2: RETRIEVE THE CUSTOMERS FROM GERMANY</u>

```
SELECT *
FROM customers
WHERE country='Germany'
```

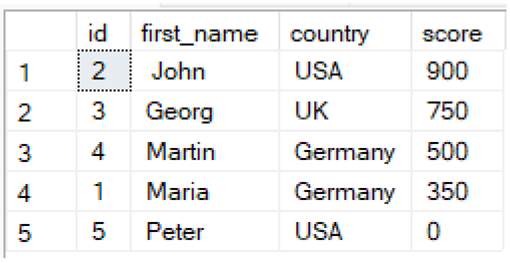


| | id | first_name | country | score |
|---|---|---|---|---|
| 1 | 1 | Maria | Germany | 350 |
| 2 | 4 | Martin | Germany | 500 |

**ORDER BY:**

<u>QUERY 1: RETREIVE ALL CUSTOMERS AND SORT THE RESULTS BY THE HIGHEST SCORE FIRST.</u>

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

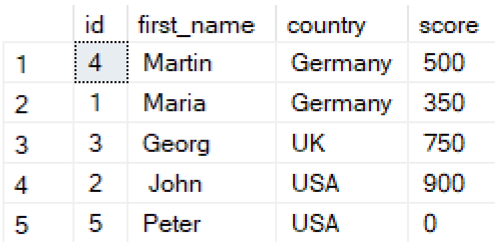


| | id | first_name | country | score |
|---|---|---|---|---|
| 1 | 2 | John | USA | 900 |
| 2 | 3 | Georg | UK | 750 |
| 3 | 4 | Martin | Germany | 500 |
| 4 | 1 | Maria | Germany | 350 |
| 5 | 5 | Peter | USA | 0 |

**Nested ORDER BY:**

<u>QUERY 1: RETRIEVE ALL CUSTOMERS AND SORT BY COUNTRY AND THEN BY THE HIGHEST SCORE</u>

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



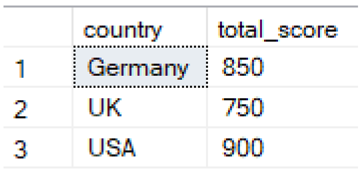
| | id | first_name | country | score |
|---|---|---|---|---|
| 1 | 4 | Martin | Germany | 500 |
| 2 | 1 | Maria | Germany | 350 |
| 3 | 3 | Georg | UK | 750 |
| 4 | 2 | John | USA | 900 |
| 5 | 5 | Peter | USA | 0 |

Column order in ORDER BY is crucial, as sorting is sequential.

**GROUP BY:** Aggregate your data.

QUERY 1: FIND TOTAL OF EACH COUNTRY.

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



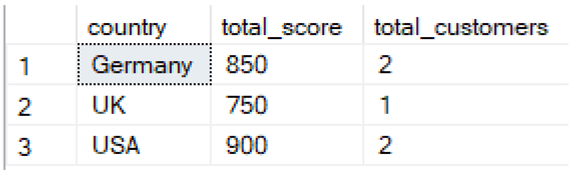
| | country | total_score |
|---|---|---|
| 1 | Germany | 850 |
| 2 | UK | 750 |
| 3 | USA | 900 |

**AS(Alias):** shorthand name(label) assigned to a column or table in a query.

**GROUP BY Rule:** All the columns in the SELECT must be either aggregated or included in the GROUP BY.

QUERY 2: FIND THE TOTAL SCORE AND TOAL CUSTOMERS IN EACH COUNTRY.

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

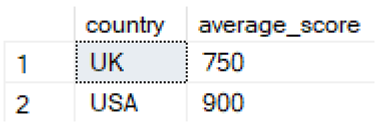


| | country | total_score | total_customers |
|---|---|---|---|
| 1 | Germany | 850 | 2 |
| 2 | UK | 750 | 1 |
| 3 | USA | 900 | 2 |

**HAVING:** Used to filter the data only after aggregating which means after group by.

QUERY 1: 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 average_score
FROM customers
WHERE score != 0
GROUP BY country
HAVING AVG(score)>430
```



| | country | average_score |
|---|---|---|
| 1 | UK | 750 |
| 2 | USA | 900 |

**DISTINCT:** Removes duplicates (Repeated values)

QUERY 1: RETURN UNIQUE LIST OF ALL COUNTRIES

```
SELECT DISTINCT
country
FROM customers
```

| | country |
|---|---|
| 1 | Germany |
| 2 | UK |
| 3 | USA |

Don't use DISTINCT unless it is necessary, because it slows down the query.

**TOP(Limit):**

QUERY 1: RETREIVE ONLY FIRST 3 CUSTOMERS

```
SELECT TOP 3 *
FROM customers
```

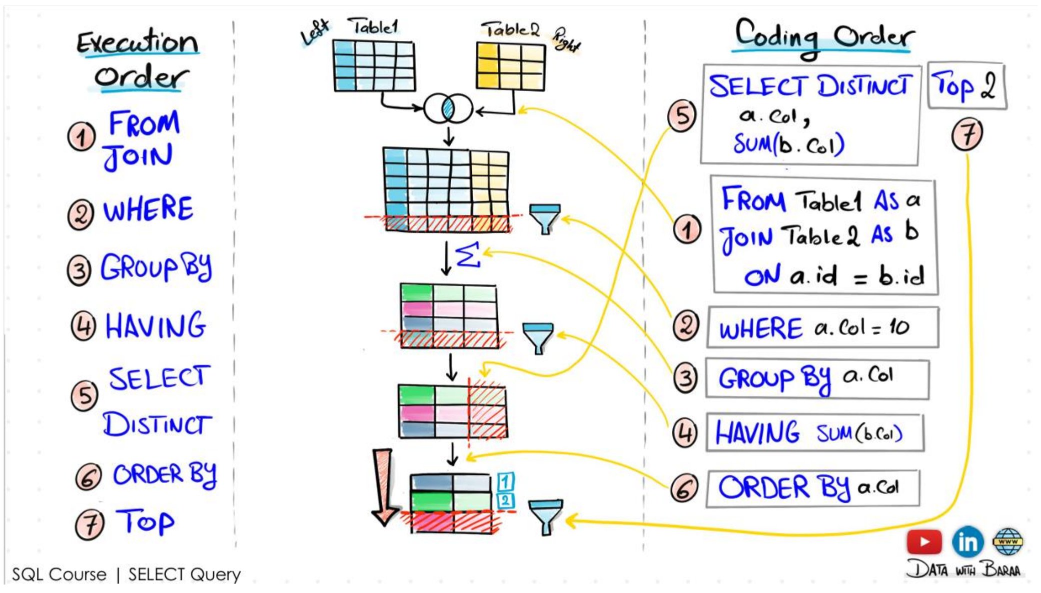
| | id | first_name | country | score |
|---|---|---|---|---|
| 1 | 1 | Maria | Germany | 350 |
| 2 | 2 | John | USA | 900 |
| 3 | 3 | Georg | UK | 750 |

QUERY 2: RETREIVE TOP 3 CUSTOMERS

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

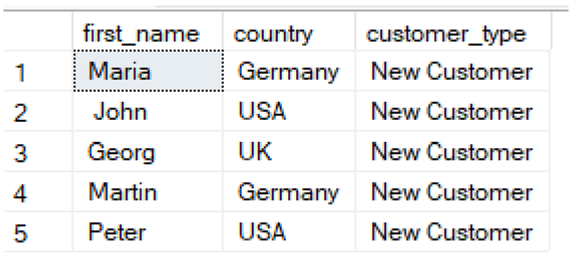
| | id | first_name | country | score |
|---|---|---|---|---|
| 1 | 2 | John | USA | 900 |
| 2 | 3 | Georg | UK | 750 |
| 3 | 4 | Martin | Germany | 500 |

**EXECUTION ORDER V/S CODING ORDER**



**STATIC VALUES:**

```
SELECT
first_name,
country,
'New Customer' AS customer_type
FROM customers
```



| | first_name | country | customer_type |
|---|---|---|---|
| 1 | Maria | Germany | New Customer |
| 2 | John | USA | New Customer |
| 3 | Georg | UK | New Customer |
| 4 | Martin | Germany | New Customer |
| 5 | Peter | USA | New Customer |