Filtering Data with `WHERE` clause (Core SQL Skill)

Theory

The **WHERE** clause is used to **filter rows** in a table based on one or more conditions.

Only rows that satisfy the **condition** specified in the **WHERE** clause are turned in the query result.

- **Use case in data analysis:** Analysts often need subsets of data to compute metrics, generate reports, or visualize trends
- **Data type awareness:** Filtering depends on the column type (numeric, string, date, etc.)

The logical execution order relevant to this lesson is:

1. **FROM** - Identify rows
2. **WHERE** - Filter rows
3. **SELECT** - Choose columns

The **WHERE** clause is essential for:

- Restricting datasets
- Performing analytical queries
- Answering business questions

Syntax

```
SELECT column1, column2
FROM table
WHERE condition;
```

- **condition** must evaluate to **TRUE**
- Rows evaluating to **FALSE** or **UNKNOWN** are excluded

Data Types in Filtering

Filtering depends on column data types.

1. **Numerical Data Types**

Used with comparison operators.

```
WHERE age > 30
```

2. **Text (String) Columns**

Must be enclosed in single quotes.

```
WHERE country = 'USA'
```

Comparison Operators

Operator	Meaning	Example
=	Equal to	WHERE age = 25
!= / <>	Not equal	WHERE country != 'UK'
>	Greater than	WHERE age > 30
<	Less than	WHERE age < 30
>=	Greater than or equal	WHERE age >= 18
<=	Less than or equal	WHERE age <= 40

Example Dataset

Table: customers

customer_id	name	country	age

1	Alice	USA	25
2	Bob	UK	40
3	Carol	USA	32
4	Dan	Canada	18
5	Eva	Canada	29

Example 1: Filtering by String Column

Requirement: Retrieve all customers from Canada.

```
SELECT *
FROM customers
WHERE country = 'Canada';
```

Result:

customer_id	name	country	age
4	Dan	Canada	18
5	Eva	Canada	29

Example 2: Filtering by Numeric Column

Requirement: Retrieve customers older than 30.

```
SELECT name, age
FROM customers
WHERE age > 30;
```

Result:

name	age
Bob	40
Carol	32

Example 3: Exact Match Filtering

Requirement: Retrieve customers who are exactly 18 years old.

```
SELECT *
FROM customers
WHERE age = 18;
```

Result:

customer_id	name	country	age
4	Dan	Canada	18

Example 4: Excluding Rows

Requirement: Retrieve customers who are **not** from the UK.

```
SELECT name, country
FROM customers
WHERE country != "UK";
```

Result:

name	country
Alice	USA
Carol	USA
Dan	Canada
Eva	Canada

Best Practices

- Avoid **`SELECT *`** in production; specify only necessary columns
- Verify column types before writing conditions
- Use proper comparison operators

- Start filtering with simple conditions before combining multiple filters

Practice Exercises

Using the **customers** table, write SQL queries for the following requirements:

▼ Retrieve the **names of all customers from the USA**

```
SELECT name, country
FROM customers
WHERE country = 'USA';
```

▼ Retrieve the **name and age** of customers **older than 25**

```
SELECT name, age
FROM customers
WHERE age > 25;
```

▼ Retrieve **all columns** for customers **younger than 20**

```
SELECT *
FROM customers
WHERE age < 20;
```

▼ Retrieve the **names of customers whose age is exactly 29**

```
SELECT name
FROM customers
WHERE age = 29;
```

Instructions:

- Write clean, formatted SQL
- Pay attention to data types
- Avoid unnecessary columns (**SELECT ***) unless requested