SQL CMDs - DISTINCT, LIKE, IN, BETWEEN, UPPER & LOWER.

1. DISTINCT

   To get unique data `(no duplicate data)` from whole table.

   ```
   SELECT DISTINCT * FROM table_name;
   ```

   To get Unique data from a column.

   ```
   SELECT DISTINCT column_name FROM table_name;
   ```

   If you want only 100 rows data from the column

   ```
   SELECT DISTINCT column_name FROM table_name limit 100;
   ```

2. LIKE

   To search for a `specified pattern` in a column.

   ```
   SELECT * FROM table_name WHERE column_name LIKE pattern;

   -- WHERE is used for filtering, or for selecting on which the operator
   -- will operate
   -- LIKE is a operator.
   -- Pattern like 'aditya%' , 'milk%'
   ```

   There are two wildcards often used in conjunction with the `LIKE` operator:

- The percent sign (%) represents zero, one, or multiple characters
- The underscore sign (_) represents one, single character

The percent sign and the underscore can also be used in combinations!

You can also combine any number of conditions using `AND` or `OR` operators.

Here are some examples showing different `LIKE` operators with '%' and '_' wildcards:

LIKE Operator	Description
WHERE CustomerName LIKE 'a%'	Finds any values that start with "a"
WHERE CustomerName LIKE '%a'	Finds any values that end with "a"
WHERE CustomerName LIKE '%or%'	Finds any values that have "or" in any position
WHERE CustomerName LIKE '_r%'	Finds any values that have "r" in the second position
WHERE CustomerName LIKE 'a_%'	Finds any values that start with "a" and are at least 2 characters in length
WHERE CustomerName LIKE 'a__%'	Finds any values that start with "a" and are at least 3 characters in length
WHERE ContactName LIKE 'a%o'	Finds any values that start with "a" and ends with "o"

Example:

- The following SQL statement selects all customers with a ContactName that starts with "a" and ends with "o":

```
SELECT * FROM Customers
WHERE ContactName LIKE 'a%o';
```

- The following SQL statement selects all customers with a CustomerName that starts with "a" and are at least 3 characters in length:

```
SELECT * FROM Customers
WHERE CustomerName LIKE 'a__%';
```

- If we got to same word in data but there first letter are different one is in upper case and one in lower case then your `like` will not work.

```
-- OPTIMIZED WAY --
SELECT UPPER(Column_name) FROM table_name
WHERE Column_name LIKE '%ADI';
```

3. IN

   The `IN` operator allows you to specify multiple values in a `WHERE` clause.

   The `IN` operator is a shorthand for multiple `OR` conditions.

```
SELECT column_name(s) FROM table_name WHERE column_name IN (value1, value2, ...);

or

SELECT column_name(s)
FROM table_name
WHERE column_name IN (SELECT STATEMENT);
```

   Example:

   - The following SQL statement selects all customers that are located in "Germany", "France" or "UK":

```
SELECT * FROM Customers
WHERE Country IN ('Germany', 'France', 'UK');
```

   - The following SQL statement selects all customers that are `NOT` located in "Germany", "France" or "UK":

```
SELECT * FROM Customers
WHERE Country NOT IN ('Germany', 'France', 'UK');
```

4. AND & OR & NOT

The `AND` and `OR` operators are used to filter records based on more than one condition:

- The `AND` operator displays a record if all the conditions separated by `AND` are TRUE.
- The `OR` operator displays a record if any of the conditions separated by `OR` is TRUE.

```
---------------- AND --------------------
SELECT column_name(s)
FROM table_name
WHERE condition1 AND condition2 AND condition3 ...;

----------------- OR --------------------
SELECT column_name(s)
FROM table_name
WHERE condition1 OR condition2 OR condition3 ...;

----------------- NOT --------------------
SELECT column_name(s)
FROM table_name
WHERE NOT condition;
```

Example:

- The following SQL statement selects all fields from "Customers" where country is "Germany" AND city is "Berlin":

```
SELECT * FROM Customers
WHERE Country='Germany' AND City='Berlin';
```

- The following SQL statement selects all fields from "Customers" where city is "Berlin" OR "München":

```
SELECT * FROM Customers
WHERE City='Berlin' OR City='München';
```

- The following SQL statement selects all fields from "Customers" where country is NOT "Germany":

```
SELECT * FROM Customers
WHERE NOT Country='Germany';
```

- The following SQL statement selects all fields from "Customers" where country is "Germany" AND city must be "Berlin" OR "München" (use parenthesis to form complex expressions):

```
SELECT * FROM Customers
WHERE Country='Germany' AND (City='Berlin' OR City='München');
```

5. BETWEEN - use where range is given. like price.

   The `BETWEEN` operator selects values within a given range. The values can be numbers, text, or dates.

   The `BETWEEN` operator is inclusive: begin and end values are included.

```
SELECT column_name(s)
FROM table_name
WHERE column_name BETWEEN value1 AND value2;
```

   Examplea:

   - The following SQL statement selects all products with a price between 10 and 20:

```
SELECT * FROM Products
WHERE Price BETWEEN 10 AND 20;
```

   - The following SQL statement selects all products with a price between 10 and 20. In addition; do not show products with a CategoryID of 1,2, or 3:

```
SELECT * FROM Products
WHERE Price BETWEEN 10 AND 20
AND CategoryID NOT IN (1,2,3);
```

- The following SQL statement selects all orders with an OrderDate between '01-July-1996' and '31-July-1996':

```
SELECT * FROM Orders
WHERE OrderDate BETWEEN '1996-07-01' AND '1996-07-31';
```

6. UPPER & LOWER

   - The `UPPER()` function converts a string to upper-case.

```
SELECT UPPER(CustomerName) AS UppercaseCustomerName
FROM Customers;
```

   - The `LOWER()` function converts a string to lower-case.

```
SELECT LOWER(CustomerName) AS LowercaseCustomerName
FROM Customers;
```