SELECT Statement

- It is used to select data from a database.

Example: **SELECT CustomerName, City FROM Customers;**

Syntax:

```
SELECT column1, column2, ...
FROM table_name;
```

SELECT DISTINCT

- It is used to return only distinct (different) values.

Example: **SELECT DISTINCT Country FROM Customers;**

Syntax:

```
SELECT DISTINCT column1, column2, ...
FROM table_name;
```

WHERE Clause

- The WHERE clause is used to filter records.
- It is used to extract only those records that fulfill a specified condition.

Example: **SELECT * FROM Customers**
WHERE Country='Mexico';

Note: The WHERE clause is not only used in SELECT statements, it is also used in UPDATE, DELETE, etc.!

The following operators can be used in the `WHERE` clause:

Operator	Description
=	Equal
>	Greater than
<	Less than
>=	Greater than or equal
<=	Less than or equal
<>	Not equal. **Note:** In some versions of SQL this operator may be written as !=
BETWEEN	Between a certain range
LIKE	Search for a pattern
IN	To specify multiple possible values for a column

ORDER BY

It is used to sort the result-set in ascending or descending order.

```
Example: SELECT * FROM Products
         ORDER BY Price;
```

DESC

- The ORDER BY keyword sorts the records in ascending order by default.
- To sort the records in descending order, use the DESC keyword.

```
Example:  SELECT * FROM Products
          ORDER BY Price DESC;
```

AND Operator

- The WHERE clause can contain one or many AND operators.
- It is used to filter records based on more than one condition, like if you want to return all customers from Spain that starts with the letter 'G':

```
Example: SELECT *FROM Customers
      WHERE Country = 'Spain' AND CustomerName LIKE 'G%';
```

AND vs OR

- The AND operator displays a record if *all* the conditions are TRUE.
- The OR operator displays a record if *any* of the conditions are TRUE.

INSERT INTO

It is used to insert new records in a table.

UPDATE

It is used to modify the existing records in a table.

DELETE

It is used to delete existing records in a table.

COUNT( )

It returns the number of rows that matches a specified criterion.

Example:
```
SELECT COUNT(*)
FROM Products;
```

LIKE

It is used in a WHERE clause to search for a specified pattern in a column.

IN

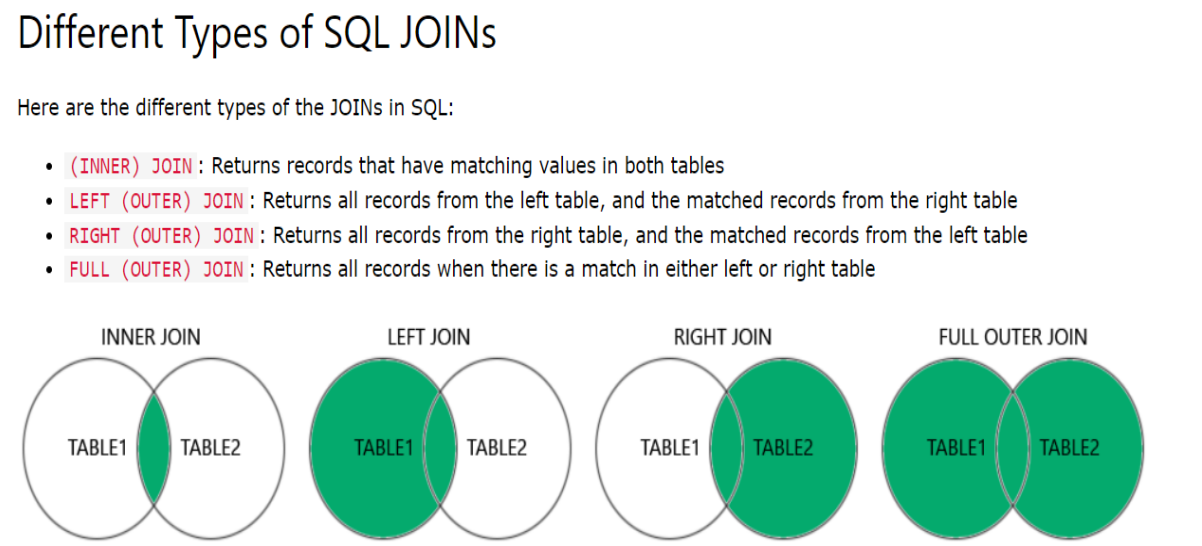
- It allows you to specify multiple values in a WHERE clause.
- It is a shorthand for multiple OR conditions.

BETWEEN

- It 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.

JOIN

A JOIN clause is used to combine rows from two or more tables, based on a related column between them.



INNER JOIN

It selects records that have matching values in both tables.

LEFT JOIN

- It returns all records from the left table , and the matching records from the right table.
- The result is 0 records from the right side, if there is no match.

RIGHT JOIN

- It returns all records from the right table and the matching records from the left table.
- The result is 0 records from the left side, if there is no match.

FULL OUTER JOIN

It returns all records when there is a match in left or right table records.

Tip: FULL OUTER JOIN and FULL JOIN are the same.

Self Join

It is a regular join, but the table is joined with itself.

UNION

It is used to combine the result-set of two or more SELECT statements.

Every SELECT statement within UNION must have the same number of columns.

The columns must also have similar data types.

The columns in every SELECT statement must also be in the same order

GROUP BY

- It groups rows that have the same values into summary rows, like "find the number of customers in each country".

- It is often used with aggregate functions(COUNT(), MAX(), MIN(), SUM(), AVG()) to group the result-set by one or more columns.

HAVING Clause

It was added to SQL because the WHERE keyword cannot be used with aggregate functions.

EXISTS Operator

- It is used to test for the existence of any record in a subquery.

- It returns TRUE if the subquery returns one or more records.

SELECT INTO

It copies data from one table into a new table.

INSERT INTO SELECT

- It copies data from one table and inserts it into another table.

- It requires that the data types in source and target tables match.

Note: The existing records in the target table are unaffected.

CREATE DATABASE

It is used to create a new SQL database.

Syntax: **CREATE DATABASE *databasename*;**

DROP DATABASE

It is used to drop an existing SQL database.

Syntax: **DROP DATABASE *databasename*;**

DROP TABLE Statement

It is used to drop an existing table in a database.

Syntax: **DROP TABLE *table_name*;**

Note: Be careful before dropping a table. Deleting a table will result in loss of complete information stored in the table!

ALTER TABLE Statement

- It is used to add, delete, or modify columns in an existing table.
- It is also used to add and drop various constraints on an existing table.

NOT NULL Constraint

- By default, a column can hold NULL values.
- It enforces a column to NOT accept NULL values.
- This enforces a field to always contain a value, which means that you cannot insert a new record, or update a record without adding a value to this field.

UNIQUE Constraint

- It ensures that all values in a column are different.

- Both the UNIQUE and PRIMARY KEY constraints provide a guarantee for uniqueness for a column or set of columns.

- A PRIMARY KEY constraint automatically has a UNIQUE constraint.

However, you can have many UNIQUE constraints per table, but only one PRIMARY KEY constraint per table.

PRIMARY KEY Constraint

- It uniquely identifies each record in a table.

- It must contain UNIQUE values, and cannot contain NULL values.

- A table can have only ONE primary key; and in the table, this primary key can consist of single or multiple columns (fields).

FOREIGN KEY Constraint

- It is used to prevent actions that would destroy links between tables.

- A FOREIGN KEY is a field (or collection of fields) in one table, that refers to the PRIMARY KEY in another table.

- The table with the foreign key is called the child table, and the table with the primary key is called the referenced or parent table.

CHECK Constraint

- It is used to limit the value range that can be placed in a column.

- If you define a CHECK constraint on a column it will allow only certain values for this column.

- If you define a CHECK constraint on a table it can limit the values in certain columns based on values in other columns in the row.

DEFAULT Constraint

- It is used to set a default value for a column.

- The default value will be added to all new records, if no other value is specified.

CREATE INDEX

- It is used to create indexes in tables.
- Indexes are used to retrieve data from the database more quickly than otherwise.
- The users cannot see the indexes, they are just used to speed up searches/queries.

Note: Updating a table with indexes takes more time than updating a table without (because the indexes also need an update). So, only create indexes on columns that will be frequently searched against.

Views

CREATE VIEW Statement

- In SQL, a view is a virtual table based on the result-set of an SQL statement.
- A view contains rows and columns, just like a real table. The fields in a view are fields from one or more real tables in the database.
- You can add SQL statements and functions to a view and present the data as if the data were coming from one single table.
- A view is created with the CREATE VIEW statement.

Syntax:

```
CREATE VIEW view_name AS
SELECT column1, column2, ...
FROM table_name
WHERE condition;
```

Note: A view always shows up-to-date data! The database engine recreates the view, every time a user queries it.