## **What is SQL?**

* SQL is a standard database language used to access and manipulate data in databases. Overall SQL is a query language that communicates with databases.
* SQL stands for Structured Query Language. SQL was developed by IBM Computer Scientists in the 1970s.
* By executing queries SQL can create, update, delete, and retrieve data in databases like MySQL, Oracle, PostgreSQL, etc.

## **What is a Database?**

* A database is the organized collection of structured data which is usually controlled by a database management system (DBMS).

## RDBMS

* RDBMS stands for Relational Database Management System.
* RDBMS is the basis for SQL, and for all modern database systems such as MS SQL Server, IBM DB2, Oracle, MySQL, and Microsoft Access.
* The data in RDBMS is stored in database objects called tables. A table is a collection of related data entries and it consists of columns and rows.

## **More About SQL?**

* SQL is case insensitive. But it is a recommended practice to use keywords (like SELECT, UPDATE, CREATE, etc.) in capital letters and use user-defined things (like table name, column name, etc.) in small letters.
* We can write comments in SQL using “–” (double hyphen) at the beginning of any line.

## **Important Terminologies**

These are some important terminologies that are used in terms of relation.

* **Attribute:**Attributes are the properties that define a relation. e.g.; **ROLL\_NO**, **NAME** etc.
* **Tuple:**Each row in the relation is known as tuple. The above relation contains 4 tuples, one of which is shown as:

| **1** | **RAM** | **DELHI** | **9455123451** | **18** |
| --- | --- | --- | --- | --- |

* **Degree:**The number of attributes in the relation is known as degree of the relation. The **STUDENT** relation defined above has degree 5.
* **Cardinality:**The number of tuples in a relation is known as cardinality. The **STUDENT**relation defined above has cardinality 4.
* **Column:**Column represents the set of values for a particular attribute. The column **ROLL\_NO** is extracted from relation STUDENT.

| **ROLL\_NO** |
| --- |
| 1 |
| 2 |
| 3 |
| 4 |

## **How Queries can be Categorized in Relational Database?**

The queries to deal with relational database can be categories as:

* **Data Definition Language:**It is used to define the structure of the database. e.g; CREATE TABLE, ADD COLUMN, DROP COLUMN and so on.
* **Data Manipulation Language:**It is used to manipulate data in the relations. e.g.; [INSERT](https://www.geeksforgeeks.org/sql-insert-statement/), [DELETE](https://www.geeksforgeeks.org/sql-delete-statement/), [UPDATE](https://www.geeksforgeeks.org/sql-update-statement/) and so on.
* **Data Query Language:**It is used to extract the data from the relations. e.g.; SELECT So first we will consider the Data Query Language. A generic query to retrieve from a relational database is:

1. **SELECT** [**DISTINCT**] Attribute\_List **FROM** R1,R2….RM
2. [**WHERE** condition]
3. [**GROUP BY** (Attributes)[**HAVING** condition]]
4. [**ORDER BY**(Attributes)[**DESC**]];

Part of the query represented by statement 1 is compulsory if you want to retrieve from a relational database. The statements written inside [] are optional. We will look at the possible query combination on relation shown in Table 1.

Semicolon after SQL Statements?

* Some database systems require a semicolon at the end of each SQL statement.
* Semicolon is the standard way to separate each SQL statement in database systems that allow more than one SQL statement to be executed in the same call to the server.
* In this tutorial, we will use semicolon at the end of each SQL statement.

### **SQL (Structured Query Language)** is a domain-specific programming language used for managing and manipulating relational databases. SQL consists of various elements, including commands, clauses, and built-in functions, which are essential for interacting with databases. Here's an overview of each of these elements:

### **1. SQL Commands:**

SQL commands are instructions that you use to perform specific tasks on a database. Some common SQL commands include:

- \*\*SELECT\*\*: Retrieves data from one or more tables.

- \*\*INSERT\*\*: Adds new rows of data to a table.

- \*\*UPDATE\*\*: Modifies existing data in a table.

- \*\*DELETE\*\*: Removes rows from a table.

- \*\*CREATE\*\*: Creates a new database object, such as a table, view, or index.

- \*\*ALTER\*\*: Modifies the structure of an existing database object.

- \*\*DROP\*\*: Deletes a database object.

- \*\*GRANT\*\*: Provides privileges to users or roles.

- \*\*REVOKE\*\*: Removes privileges from users or roles.

### **2. SQL Clauses:**

SQL commands are often accompanied by various clauses to specify conditions, constraints, and other details. Some common SQL clauses include:

- \*\*WHERE\*\*: Filters rows based on specified conditions in a SELECT statement.

- \*\*ORDER BY\*\*: Sorts the result set in ascending or descending order.

- \*\*GROUP BY\*\*: Groups rows with the same values in one or more columns.

- \*\*HAVING\*\*: Filters groups in a GROUP BY query based on specified conditions.

- \*\*JOIN\*\*: Combines rows from two or more tables based on related columns.

- \*\*ON\*\*: Specifies the join conditions in a JOIN clause.

- \*\*LIMIT\*\* (or \*\*TOP\*\* in some database systems): Limits the number of rows returned in the result set.

- \*\*DISTINCT\*\*: Returns only distinct values in a result set, eliminating duplicates.

- \*\*AS\*\*: Renames columns or tables with aliases.

### 3. SQL Built-In Functions:

SQL provides a wide range of built-in functions that perform specific operations on data. These functions can be used in SELECT statements, WHERE clauses, and other parts of SQL queries. Common categories of SQL built-in functions include:

- \*\*Aggregate Functions\*\*: Perform calculations on groups of rows (e.g., SUM, AVG, COUNT, MAX, MIN).

- \*\*Scalar Functions\*\*: Operate on individual values (e.g., CONCAT, UPPER, LOWER, DATE functions).

- \*\*Date and Time Functions\*\*: Manipulate date and time values (e.g., DATEADD, DATEDIFF, GETDATE).

- \*\*String Functions\*\*: Perform operations on text data (e.g., SUBSTRING, REPLACE, LENGTH).

- \*\*Mathematical Functions\*\*: Perform mathematical operations (e.g., ROUND, ABS, SQRT).

- \*\*Conversion Functions\*\*: Convert data types (e.g., CAST, CONVERT).

SQL's combination of commands, clauses, and functions provides a powerful and flexible way to interact with and manipulate data stored in relational databases. The specific syntax and available functions may vary slightly between different database management systems (DBMS), such as MySQL, PostgreSQL, SQL Server, and Oracle, but the core principles of SQL remain consistent.

## **1. CREATE DATABASE :**

### The CREATE DATABASE statement is used to create a new SQL database.

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

* Example : CREATE DATABASE testDB;

## **2. CREATE Table:**

#### The CREATE TABLE statement is used to create a new table in a database.

#### Syntax : **CREATE TABLE *table\_name* ( *column1 datatype*,**

#### *column2 datatype*,

#### *column3 datatype*

#### );

#### The column parameters specify the names of the columns of the table.

#### The datatype parameter specifies the type of data the column can hold (e.g. varchar, integer, date, etc.).

#### Example :

#### CREATE TABLE Persons (

#### PersonID int,

#### LastName varchar(255),

#### FirstName varchar(255),

#### Address varchar(255),

#### City varchar(255)

#### );

### **3. SELECT :**

#### The SELECT statement is used to select data from a database.

#### Syntax:

SELECT *column1*, *column2, ...*

FROM *table\_name*;

#### Example : Created table :

#### CREATE TABLE Customers (

#### CustomerID INT PRIMARY KEY,

#### CustomerName VARCHAR(255),

#### ContactName VARCHAR(255),

#### Address VARCHAR(255),

#### City VARCHAR(255),

#### PostalCode VARCHAR(10),

#### Country VARCHAR(255)

#### );

#### This SQL query will create a table named "Customers" with the specified columns and data types. You can adjust the data types and field lengths (VARCHAR lengths in this example) based on your specific database requirements.

#### Inserted values :

#### -- Insert data into the Customers table

#### INSERT INTO Customers (CustomerID, CustomerName, ContactName, Address, City, PostalCode, Country)

#### VALUES

#### (1, 'Alfreds Futterkiste', 'Maria Anders', 'Obere Str. 57', 'Berlin', '12209', 'Germany'),

#### (2, 'Ana Trujillo Emparedados y helados', 'Ana Trujillo', 'Avda. de la Constitución 2222', 'México D.F.', '05021', 'Mexico'),

#### (3, 'Antonio Moreno Taquería', 'Antonio Moreno', 'Mataderos 2312', 'México D.F.', '05023', 'Mexico'),

#### (4, 'Around the Horn', 'Thomas Hardy', '120 Hanover Sq.', 'London', 'WA1 1DP', 'UK'),

#### (5, 'Berglunds snabbköp', 'Christina Berglund', 'Berguvsvägen 8', 'Luleå', 'S-958 22', 'Sweden');

#### **Select query :**

#### Select \* from Customers

### **4. . SELECT DISTINCT Statement :**

#### The SQL SELECT DISTINCT Statement

#### The SELECT DISTINCT statement is used to return only distinct (different) values.

#### Inside a table, a column often contains many duplicate values; and sometimes you only want to list the different (distinct) values.

#### Syntax :

#### SELECT DISTINCT column1, column2, FROM table\_name;

* Example :

SELECT DISTINCT Country FROM Customers;

* The following SQL statement lists the number of different (distinct) customer countries:
* Example :

SELECT COUNT(DISTINCT Country) FROM Customers;

#### Distincts for multiple columns :

#### Example : select distinct postalcode , country from customer;

Refer to URL : https://www.w3schools.com/sql/default.asp