SQL – Structured Query Language

What is RDBMS?

RDBMS stands for Relational Database Management System. RDBMS is the basis for SQL, and for all modern database systems like MS SQL Server, IBM DB2, Oracle, MySQL, and Microsoft Access.

A Relational database management system (RDBMS) is a database management system (DBMS) that is based on the relational model as introduced by E. F. Codd.

What is a table?

The data in an RDBMS is stored in database objects which are called as tables. This table is basically a collection of related data entries and it consists of numerous columns and rows.

Remember, a table is the most common and simplest form of data storage in a relational database. The following is an example of a CUSTOMERS table:

Customer Id	Customer Name	Customer Contact No	Customer City
10001	Sam	99494999999	Berlin
10020	Adam	99994894849	Prague

What is a field?

Every table is broken up into smaller entities called fields. The fields in the CUSTOMERS table consist of ID, NAME, AGE, ADDRESS and SALARY.

A field is a column in a table that is designed to maintain specific information about every record in the table.

What is a Record or a Row?

A record is also called as a row of data is each individual entry that exists in a table. For example, there are 7 records in the above CUSTOMERS table. Following is a single row of data or record in the CUSTOMERS table:

What is a column?

column is a vertical entity in a table that contains all information associated with a specific field in a table.

What is a NULL value?

A NULL value in a table is a value in a field that appears to be blank, which means a field with a NULL value is a field with no value. It is very important to understand that a NULL value is different than a zero value or a field that contains spaces. A field with a NULL value is the one that has been left blank during a record creation.

What are SQL Constraints ?

Constraints are the rules enforced on data columns on a table. These are used to limit the type of data that can go into a table. This ensures the accuracy and reliability of the data in the database. Constraints

can either be column level or table level. Column level constraints are applied only to one column whereas, table level constraints are applied to the entire table.

Following are some of the most commonly used constraints available in SQL:

- **NOT NULL** Constraint: Ensures that a column cannot have a NULL value.
- **DEFAULT** Constraint: Provides a default value for a column when none is specified.
- **UNIQUE** Constraint: Ensures that all the values in a column are different.
- **PRIMARY Key**: Uniquely identifies each row/record in a database table.
- **FOREIGN Key**: Uniquely identifies a row/record in any another database table.
- **CHECK Constraint**: The CHECK constraint ensures that all values in a column satisfy certain conditions.
- **INDEX**: Used to create and retrieve data from the database very quickly

SQL Syntax :
SQL is followed by a unique set of rules and guidelines called Syntax. This tutorial gives you a quick start with SQL by listing all the basic SQL Syntax. All the SQL statements start with any of the keywords like SELECT, INSERT, UPDATE, DELETE, ALTER, DROP, CREATE, USE, SHOW and all the statements end with a semicolon (;). The most important point to be noted here is that SQL is case insensitive, which means SELECT and select have same meaning in SQL statements. Whereas, MySQL makes difference in table names. So, if you are working with MySQL, then you need to give table names as they exist in the database.

Various Syntax in SQL
All the examples given in this tutorial have been tested with a MySQL server.

SQL SELECT Statement
SELECT column1, column2....columnN FROM table_name;

SQL DISTINCT Clause
SELECT DISTINCT column1, column2....columnN FROM table_name;

SQL WHERE Clause
SELECT column1, column2....columnN FROM table_name WHERE CONDITION;

SQL AND/OR Clause
SELECT column1, column2....columnN FROM table_name WHERE CONDITION1 {AND|OR} CONDITION2;

SQL IN Clause
SELECT column1, column2....columnN FROM table_name WHERE column_name IN (val-1, val-2,...val-N);

SQL BETWEEN Clause
SELECT column1, ...columnN FROM table_name WHERE column_name BETWEEN value1 AND value2;

SQL LIKE Clause
SELECT column1, column2....columnN FROM table_name WHERE column_name LIKE { PATTERN };

SQL ORDER BY Clause
SELECT column1, column2....columnN FROM table_name WHERE CONDITION ORDER BY column_name {ASC|DESC};
SQL GROUP BY Clause
SELECT SUM(column_name) FROM table_name WHERE CONDITION GROUP BY column_name;

SQL COUNT Clause
SELECT COUNT(column_name) FROM table_name WHERE CONDITION;

SQL HAVING Clause
SELECT SUM(column_name) FROM table_name WHERE CONDITION GROUP BY column_name HAVING (arithematic function condition);

SQL CREATE TABLE Statement
CREATE TABLE table_name(column1 datatype, column2 datatype, column3 datatype, ... columnN datatype, PRIMARY KEY(one or more columns));

SQL DROP TABLE Statement
DROP TABLE table_name; SQL CREATE INDEX Statement CREATE UNIQUE INDEX index_name ON table_name (column1, column2,...columnN);

SQL DROP INDEX Statement
ALTER TABLE table_name DROP INDEX index_name;

SQL DESC Statement
DESC table_name;

SQL TRUNCATE TABLE Statement
TRUNCATE TABLE table_name;

SQL ALTER TABLE Statement
ALTER TABLE table_name {ADD|DROP|MODIFY} column_name {data_ype};

SQL ALTER TABLE Statement (Rename)
ALTER TABLE table_name RENAME TO new_table_name;

SQL INSERT INTO Statement
INSERT INTO table_name(column1, column2....columnN) VALUES (value1, value2....valueN);

SQL UPDATE Statement

UPDATE table_name SET column1 = value1, column2 = value2....columnN=valueN
[WHERE CONDITION];

SQL DELETE Statement

DELETE FROM table_name
WHERE {CONDITION};

SQL CREATE DATABASE Statement

CREATE DATABASE database_name;

SQL DROP DATABASE Statement

DROP DATABASE database_name;

SQL USE Statement

USE database_name;

SQL COMMIT Statement

COMMIT;

SQL ROLLBACK Statement

ROLLBACK;