# SQL Tutorial

SQL Tutorial

SQL Syntax

SQL Data Types

SQL Operators

# SQL Database

SQL CREATE Database

SQL DROP Database

SQL RENAME Database

SQL SELECT Database

# SQL Table

What is Table

SQL CREATE TABLE

SQL DROP TABLE

SQL DELETE TABLE

SQL RENAME TABLE

SQL TRUNCATE TABLE

SQL COPY TABLE

SQL TEMP TABLE

SQL ALTER TABLE

**SQL Select**

SELECT Statement

SQL SELECT UNIQUE

SQL SELECT DISTINCT

SQL SELECT COUNT

| ← prev | next → |

# SQL Syntax

When you want to do some operations on the data in the database, then you must have to write the query in the predefined syntax of SQL.

The syntax of the structured query language is a unique set of rules and guidelines, which is not case-sensitive. Its Syntax is defined and maintained by the ISO and ANSI standards.

Following are some most important points about the SQL syntax which are to remember:

Free Learning Platform For Better Future

Home        Python        Java        JavaScript        HTML        SQL        PHP        C#

- You can write the keywords of SQL in both uppercase and lowercase, but writing the SQL keywords in uppercase improves the readability of the SQL query.

- SQL statements or syntax are dependent on text lines. We can place a single SQL statement on one or multiple text lines.

- You can perform most of the action in a database with SQL statements.

- SQL syntax depends on relational algebra and tuple relational calculus.

## SQL Statement

SQL statements tell the database what operation you want to perform on the structured data and what information you would like to access from the database.

• • •

The statements of SQL are very simple and easy to use and understand. They are like plain English but with a particular syntax.

**Simple Example of SQL statement:**

> **SELECT** "column_name" **FROM** "table_name";

Each SQL statement begins with any of the SQL keywords and ends with the semicolon (;). The semicolon is used in the SQL for separating the multiple Sql statements which are going to execute in the same call.
In this SQL tutorial, we will use the semicolon (;) at the end of each SQL query or statement.

• • •

## Most Important SQL Commands and Statements

1. Select Statement
2. Update Statement
3. Delete Statement
4. Create Table Statement
5. Alter Table Statement
6. Drop Table Statement
7. Create Database Statement
8. Drop Database Statement
9. Insert Into Statement
10. Truncate Table Statement
11. Describe Statement
12. Distinct Clause
13. Commit Statement
14. Rollback Statement
15. Create Index Statement
16. Drop Index Statement
17. Use Statement

**Let's discuss each statement in short one by one with syntax and one example:**

## 1. SELECT Statement

This SQL statement reads the data from the SQL database and shows it as the output to the database user.

**Syntax of SELECT Statement:**

```
SELECT column_name1, column_name2, ...., column_nameN
    [ FROM

table_name ]
    [ WHERE condition ]
    [ ORDER BY order_column_name1 [ ASC | DESC ], .... ];
```

**Example of SELECT Statement:**

```
SELECT Emp_ID, First_Name, Last_Name, Salary, City
```

**FROM** Employee_details
**WHERE** Salary = 100000
**ORDER BY** Last_Name

This example shows the **Emp_ID, First_Name, Last_Name, Salary, and City** of those employees from the **Employee_details** table whose **Salary** is **100000**. The output shows all the specified details according to the ascending alphabetical order of **Last_Name**.

## 3. UPDATE Statement

This SQL statement changes or modifies the stored data in the SQL database.

**Syntax of UPDATE Statement:**

**UPDATE** table_name
**SET** column_name1 = new_value_1, column_name2 = new_value_2, ...., column_nameN = new_value_N
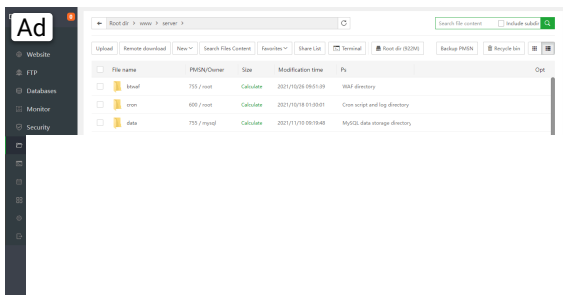[ **WHERE**  CONDITION ];

**Example of UPDATE Statement:**

**UPDATE** Employee_details
**SET** Salary = 100000
**WHERE** Emp_ID = 10;

This example changes the **Salary** of those employees of the **Employee_detail**s table whose **Emp_ID** is **10** in the table.

## 3. DELETE Statement

This SQL statement deletes the stored data from the SQL database.

**Syntax of DELETE Statement:**

**DELETE FROM** table_name
[ **WHERE** CONDITION ];

**Example of DELETE Statement:**

**DELETE FROM** Employee_details
**WHERE** First_Name = 'Sumit';

This example deletes the record of those employees from the **Employee_details** table whose **First_Name** is **Sumit** in the table.

# 4. CREATE TABLE Statement

This SQL statement creates the new table in the SQL database.

• • •

**Syntax of CREATE TABLE Statement:**

**CREATE TABLE** table_name
(
column_name1 data_type [column1 **constraint**(s)],
column_name2 data_type [column2 **constraint**(s)],
.....
.....,
column_nameN data_type [columnN **constraint**(s)],
**PRIMARY KEY**(one or more col)
);

**Example of CREATE TABLE Statement:**

```
CREATE TABLE Employee_details(
    Emp_Id NUMBER(4) NOT NULL,
    First_name VARCHAR(30),
    Last_name VARCHAR(30),
    Salary Money,
    City VARCHAR(30),
    PRIMARY KEY (Emp_Id)
);
```

This example creates the table **Employee_details** with five columns or fields in the SQL database. The fields in the table are **Emp_Id, First_Name, Last_Name, Salary,** and **City**. The **Emp_Id** column in the table acts as a **primary key**, which means that the Emp_Id column cannot contain duplicate values and null values.

## 5. ALTER TABLE Statement

This SQL statement adds, deletes, and modifies the columns of the table in the SQL database.

**Syntax of ALTER TABLE Statement:**

```
ALTER TABLE table_name ADD column_name datatype[(size)];
```

The above SQL alter statement adds the column with its datatype in the existing database table.

```
ALTER TABLE table_name MODIFY column_name column_datatype[(size)];
```

The above 'SQL alter statement' renames the old column name to the new column name of the existing database table.

> **ALTER TABLE** table_name **DROP COLUMN** column_name;

The above SQL alter statement deletes the column of the existing database table.

**Example of ALTER TABLE Statement:**

> **ALTER TABLE** Employee_details
> **ADD** Designation **VARCHAR**(18);

This example adds the new field whose name is **Designation** with size **18** in the **Employee_details** table of the SQL database.

## 6. DROP TABLE Statement

This SQL statement deletes or removes the table and the structure, views, permissions, and triggers associated with that table.

**Syntax of DROP TABLE Statement:**

> **DROP TABLE** [ IF EXISTS ]
> table_name1, table_name2, ......, table_nameN;

The above syntax of the drop statement deletes specified tables completely if they exist in the database.

**Example of DROP TABLE Statement:**

> **DROP TABLE** Employee_details;

This example drops the **Employee_details** table if it exists in the SQL database. This removes the complete information if available in the table.

## 7. CREATE DATABASE Statement

This SQL statement creates the new database in the database management system.

**Syntax of CREATE DATABASE Statement:**

> **CREATE DATABASE** database_name;

**Example of CREATE DATABASE Statement:**

> **CREATE DATABASE** Company;

The above example creates the company database in the system.

## 8. DROP DATABASE Statement

This SQL statement deletes the existing database with all the data tables and views from the database management system.

✖

**Syntax of DROP DATABAS**

> **DROP DATABASE** datab

**Example of DROP DATABA**

> **DROP DATABASE** Comp

The above example deletes the company database from the system.

## 9. INSERT INTO Statement

This SQL statement inserts the data or records in the existing table of the SQL database. This statement can easily insert single and multiple records in a single query statement.

**Syntax of insert a single record:**

> **INSERT INTO** table_name
> (
> column_name1,
> column_name2, ....,
> column_nameN
> )
> **VALUES**
> (value_1,
> value_2, .....,
> value_N
> );

**Example of insert a single record:**

```
INSERT INTO Employee_details
(
Emp_ID,
First_name,
Last_name,
Salary,
City
)
VALUES
(101,
Akhil,
Sharma,
40000,
Bangalore
);
```

This example inserts **101** in the first column, **Akhil** in the second column, **Sharma** in the third column, **40000** in the fourth column, and **Bangalore** in the last column of the table **Employee_details**.

**Syntax of inserting a multiple records in a single query:**

```
INSERT INTO table_name
( column_name1, column_name2, ...., column_nameN)
VALUES (value_1, value_2, ....., value_N), (value_1, value_2, ....., value_N),....;
```

**Example of inserting multiple records in a single query:**

```
INSERT INTO Employee_details
```

( Emp_ID, First_name, Last_name, Salary, City )
**VALUES** (101, Amit, Gupta, 50000, Mumbai), (101,  John, Aggarwal, 45000, Calcutta), (101, Sidhu, Arora, 55000, Mumbai);

This example inserts the records of three employees in the **Employee_details** table in the single query statement.

## 10. TRUNCATE TABLE Statement

This SQL statement deletes all the stored records from the table of the SQL database.

**Syntax of TRUNCATE TABLE Statement:**

**TRUNCATE TABLE** table_name;

**Example of TRUNCATE TABLE Statement:**

**TRUNCATE TABLE** Employee_details;

This example deletes the record of all employees from the Employee_details table of the database.

## 11. DESCRIBE Statement

This SQL statement tells something about the specified table or view in the query.

**Syntax of DESCRIBE Statement:**

DESCRIBE table_name | view_name;

**Example of DESCRIBE Statement:**

DESCRIBE Employee_details;

This example explains the structure and other details about the **Employee_details** table.

## 12. DISTINCT Clause

This SQL statement shows the distinct values from the specified columns of the database table. This statement is used with the **SELECT** keyword.

**Syntax of DISTINCT Clause:**

**SELECT DISTINCT** column_name1, column_name2, ...
**FROM** table_name;

**Example of DISTINCT Clause:**

**SELECT DISTINCT** City, Salary
**FROM** Employee_details;

This example shows the distinct values of the **City** and **Salary** column from the **Employee_details** table.

## 13. COMMIT Statement

This SQL statement saves the changes permanently, which are done in the transaction of the SQL database.

**Syntax of COMMIT Statement:**

**COMMIT**

**Example of COMMIT Statement:**

**DELETE FROM** Employee_details
**WHERE** salary = 30000;
**COMMIT**;

This example deletes the records of those employees whose **Salary** is **30000** and then saves the changes permanently in the database.

## 14. ROLLBACK Statement

This SQL statement undo the transactions and operations which are not yet saved to the SQL database.

**Syntax of ROLLBACK Statement:**

**ROLLBACK**

**Example of ROLLBACK Statement:**

**DELETE FROM** Employee_details
**WHERE** City = Mumbai;
**ROLLBACK**;

This example deletes the records of those employees whose **City** is **Mumbai** and then undo the changes in the database.

## 15. CREATE INDEX Statement

This SQL statement creates the new index in the SQL database table.

**Syntax of CREATE INDEX Statement:**

**CREATE INDEX** index_name
**ON** table_name ( column_name1, column_name2, ..., column_nameN );

**Example of CREATE INDEX Statement:**

**CREATE INDEX** idx_First_Name
**ON** employee_details (First_Name);

This example creates an index **idx_First_Name** on the **First_Name** column of the **Employee_details** table.

## 16. DROP INDEX Statement

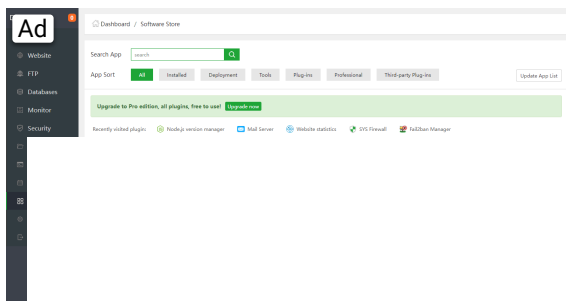This SQL statement deletes the existing index of the SQL database table.

**Syntax of DROP INDEX Statement:**

**DROP INDEX** index_name;

**Example of DROP INDEX Statement:**

**DROP INDEX** idx_First_Name;

This example deletes the index **idx_First_Name** from the SQL database.

# 17. USE Statement

This SQL statement selects the existing SQL database. Before performing the operations on the database table, you have to select the database from the multiple existing databases.

**Syntax of USE Statement:**

USE database_name;

**Example of USE DATABASE Statement:**

USE Company;

This example uses the company database.

Next Topic          SQL Data Types

← prev                                                                                            next →