**SQL – W3schools**

# Introduction

SQL is a standard language for storing, manipulating and retrieving data in databases.

**What is SQL?**

* SQL stands for Structured Query Language
* SQL lets you access and manipulate databases
* SQL became a standard of the American National Standards Institute (ANSI) in 1986, and of the International Organization for Standardization (ISO) in 1987

**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.

Every table is broken up into smaller entities called fields. The fields in the Customers table consist of CustomerID, CustomerName, ContactName, Address, City, PostalCode and Country. A field is a column in a table that is designed to maintain specific information about every record in the table.

A record, also called a row, is each individual entry that exists in a table. For example, there are 91 records in the above Customers table. A record is a horizontal entity in a table.

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

SQL Syntax

Table customers



The table above contains two records (one for each customer) and seven columns (CustomerID, CustomerName, ContactName, Address, City, PostalCode, and Country).

**SQL Statements**

Most of the actions you need to perform on a database are done with SQL statements.

The following SQL statement selects all the records in the "Customers" table:

SELECT \* FROM Customers;

**Keep in Mind That...**

* SQL keywords are NOT case sensitive: select is the same as SELECT

Semicolon after SQL Statements?

Some database systems require a semicolon at the end of each SQL statement.

# SQL Commands

**Some of The Most Important SQL Commands**

* **SELECT** - extracts data from a database
* **UPDATE** - updates data in a database
* **DELETE** - deletes data from a database
* **INSERT INTO** - inserts new data into a database
* **CREATE DATABASE** - creates a new database
* **ALTER DATABASE** - modifies a database
* **CREATE TABLE** - creates a new table
* **ALTER TABLE** - modifies a table
* **DROP TABLE** - deletes a table
* **CREATE INDEX** - creates an index (search key)
* **DROP INDEX** - deletes an index

## SQL SELECT Statement

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

The data returned is stored in a result table, called the result-set.

**SELECT Syntax**

SELECT column1, column2, ...  
FROM table\_name;

## 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.

**SELECT DISTINCT Syntax**

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

SELECT DISTINCT Country FROM Customers;

Autre exemple :

SELECT COUNT(DISTINCT Country) FROM Customers;

## SQL WHERE Clause

The WHERE clause is used to filter records.

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

**WHERE Syntax**

SELECT column1, column2, ...  
FROM table\_name  
WHERE condition;

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

Example

SELECT \* FROM Customers  
WHERE Country='Mexico';

**Text Fields vs. Numeric Fields**

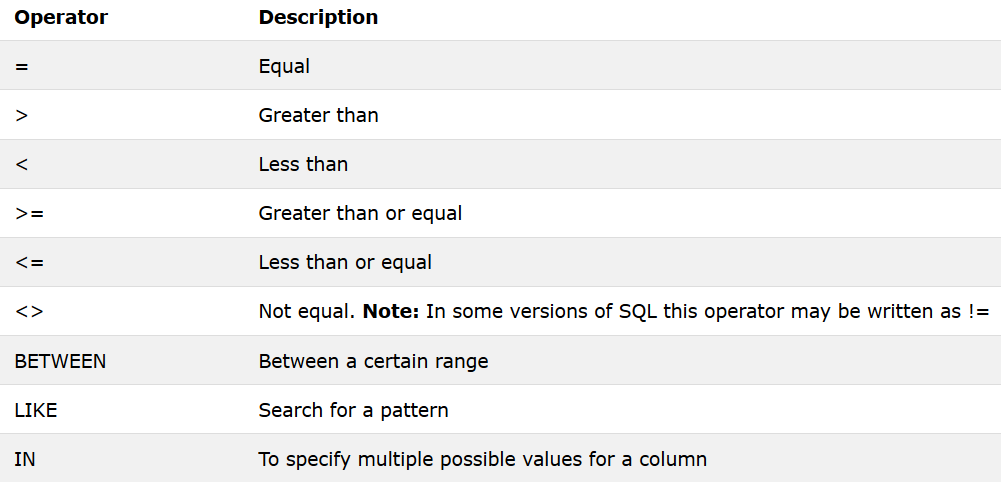
SQL requires single quotes around text values (most database systems will also allow double quotes).

However, numeric fields should not be enclosed in quotes:

SELECT \* FROM Customers WHERE CustomerID=1;

**Operators in The WHERE Clause**

The following operators can be used in the WHERE clause:



SELECT \* FROM Customers WHERE CustomerID > 25 and CustomerID <50;

## SQL AND, OR and NOT Operators

The WHERE clause can be combined with AND, OR, and NOT operators.

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.

The NOT operator displays a record if the condition(s) is NOT TRUE.

AND Syntax

SELECT column1, column2, ...  
FROM table\_name  
WHERE condition1 AND condition2 AND condition3 ...;

Example :

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

NOT Syntax

SELECT column1, column2, ...  
FROM table\_name  
WHERE NOT condition;

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

Example :

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

**Combining AND, OR and NOT**

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

## SQL ORDER BY Keyword

The ORDER BY keyword is used to sort the result-set in ascending or descending order.

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

**ORDER BY Syntax**

SELECT column1, column2, ...  
FROM table\_name  
ORDER BY column1, column2, ... ASC|DESC;

Example :

SELECT \* FROM Customers  
ORDER BY Country;

**ORDER BY DESC Example**

The following SQL statement selects all customers from the "Customers" table, sorted DESCENDING by the "Country" column

Example

SELECT \* FROM Customers  
ORDER BY Country DESC;

**ORDER BY Several Columns Example**

The following SQL statement selects all customers from the "Customers" table, sorted by the "Country" and the "CustomerName" column. This means that it orders by Country, but if some rows have the same Country, it orders them by CustomerName:

Example

SELECT \* FROM Customers  
ORDER BY Country, CustomerName;

**ORDER BY Several Columns Example 2**

The following SQL statement selects all customers from the "Customers" table, sorted ascending by the "Country" and descending by the "CustomerName" column:

SELECT \* FROM Customers  
ORDER BY Country ASC, CustomerName DESC;

# SQL INSERT INTO Statement

The INSERT INTO statement is used to insert new records in a table.

INSERT INTO Syntax

It is possible to write the INSERT INTO statement in two ways.

The first way specifies both the column names and the values to be inserted:

INSERT INTO table\_name (column1, column2, column3, ...)  
VALUES (value1, value2, value3, ...);

If you are adding values for all the columns of the table, you do not need to specify the column names in the SQL query. However, make sure the order of the values is in the same order as the columns in the table. The INSERT INTO syntax would be as follows:

INSERT INTO table\_name  
VALUES (value1, value2, value3, ...);

Example

INSERT INTO Customers (CustomerName, Address, City, PostalCode, Country)  
VALUES ('Cardinal', 'Skagen 21', 'Stavanger', '4006', 'Norway');

Insert Data Only in Specified Columns

It is also possible to only insert data in specific columns.

The following SQL statement will insert a new record, but only insert data in the "CustomerName", "City", and "Country" columns (CustomerID will be updated automatically):

Example

INSERT INTO Customers (CustomerName, City, Country)  
VALUES ('Cardinal', 'Stavanger', 'Norway');

SQL NULL Values

A field with a NULL value is a field with no value.

If a field in a table is optional, it is possible to insert a new record or update a record without adding a value to this field. Then, the field will be saved with a NULL value.

Note: A NULL value is different from a zero value or a field that contains spaces. A field with a NULL value is one that has been left blank during record creation!

How to Test for NULL Values?

It is not possible to test for NULL values with comparison operators, such as =, <, or <>.

We will have to use the IS NULL and IS NOT NULL operators instead.

IS NULL Syntax

SELECT column\_namesFROM table\_name  
WHERE column\_name IS NULL;

1