DDL Commands in SQL

DDL is an abbreviation of Data Definition Language.

The DDL Commands in Structured Query Language are used to create and modify the schema of the database and its objects. The syntax of DDL commands is predefined for describing the data. The commands of Data Definition Language deal with how the data should exist in the database.

Following are the five DDL commands in SQL:

1. CREATE Command
2. DROP Command
3. ALTER Command
4. TRUNCATE Command
5. RENAME Command

## CREATE Command

CREATE is a DDL command used to create databases, tables, triggers and other database objects.

**Syntax to Create a Database:**

CREATE Database Database\_Name;

Example .

Create Database Books;

Syntax to create a new table:

CREATE TABLE table\_name  (

column\_Name1 data\_type ( size of the column ) ,

column\_Name2 data\_type ( size of the column) ,

column\_Name3 data\_type ( size of the column) ,

column\_NameN data\_type ( size of the column )

) ;

Example:

CREATE TABLE Student   (

Roll\_No. Int ,

First\_Name Varchar (20) ,

Last\_Name Varchar (20) ,

Age Int ,

Marks Int ,

) ;

## DROP Command

DROP is a DDL command used to delete/remove the database objects from the SQL database. We can easily remove the entire table, view, or index from the database using this DDL command.

### Examples of DROP Command in SQL

**Example 1: This example describes how to remove a database from the SQL database.**

**Syntax to remove a database:**

DROP DATABASE Database\_Name;

Example:

DROP DATABASE Books;

## ALTER Command

ALTER is a DDL command which changes or modifies the existing structure of the database, and it also changes the schema of database objects.

We can also add and drop constraints of the table using the ALTER command.

### Examples of ALTER Command in SQL

**Example 1: This example shows how to add a new field to the existing table.**

**Syntax to add a newfield in the table:**

ALTER TABLE Student ADD Father's\_Name Varchar(60);

**Example 2: This example describes how to remove the existing column from the table.**

**Syntax to remove a column from the table:**

1. ALTER TABLE name\_of\_table DROP Column\_Name\_1 , column\_Name\_2 , ….., column\_Name\_N;

Suppose, you want to remove the Age and Marks column from the existing Student table. To do this, you have to write the following DDL command:

1. ALTER TABLE Student DROP Age, Marks;

**Example 3: This example describes how to modify the existing column of the existing table.**

**Syntax to modify the column of the table:**

1. ALTER TABLE table\_name MODIFY ( column\_name column\_datatype(size));

## TRUNCATE Command

TRUNCATE is another DDL command which deletes or removes all the records from the table.

This command also removes the space allocated for storing the table records.

**Syntax of TRUNCATE command**

1. TRUNCATE TABLE Table\_Name;

### Example

Suppose, you want to delete the record of the Student table.

TRUNCATE TABLE Student;

SELECT \* FROM Student;

## RENAME Command

RENAME is a DDL command which is used to change the name of the database table.

### Example

RENAME TABLE Student TO Student\_Details ;

DML Commands in SQL

DML is an abbreviation of **Data Manipulation Language**.

The DML commands in Structured Query Language change the data present in the SQL database. We can easily access, store, modify, update and delete the existing records from the database using DML commands.

**Following are the four main DML commands in SQL:**

1. SELECT Command
2. INSERT Command
3. UPDATE Command
4. DELETE Command

## SELECT DML Command

SELECT is the most important data manipulation command in Structured Query Language. The SELECT command shows the records of the specified table. It also shows the particular record of a particular column by using the WHERE clause.

**Example 1: This example shows all the values of every column from the table.**

SELECT \* FROM Student;

**Example 2:** To displays all the values of **Emp\_Salary** and **Emp\_Id** column of **Employee** table:

SELECT Emp\_Id, Emp\_Salary FROM Employee;

**Example 3:** If you want to access all the records of those students whose marks is 80 from the above table, then you have to write the following DML command in SQL:

SELECT \* FROM Student WHERE Stu\_Marks = 80;

## INSERT DML Command

INSERT is another most important data manipulation command in Structured Query Language, which allows users to insert data in database tables.

**Example 1: This example describes how to insert the record in the database table.**

INSERT INTO Student (Stu\_id, Stu\_Name, Stu\_Marks, Stu\_Age) VALUES (104, Anmol, 89, 19);

## UPDATE DML Command

UPDATE is another most important data manipulation command in Structured Query Language, which allows users to update or modify the existing data in database tables.

**Example 1: This example describes how to update the value of a single field.**

UPDATE Product SET Product\_Price = 80 WHERE Product\_Id = 'P102' ;

**Example 2: This example describes how to update the value of multiple fields of the database table.**

UPDATE Student SET Stu\_Marks = 80, Stu\_Age = 21 WHERE Stu\_Id = 103 AND Stu\_Id = 202;

## DELETE DML Command

DELETE is a DML command which allows SQL users to remove single or multiple existing records from the database tables.

**Syntax of DELETE Command**

DELETE FROM Table\_Name WHERE condition;

**Example 1: This example describes how to delete a single record from the table.**

DELETE FROM Product WHERE Product\_Id = 'P202' ;

**Example 2: This example describes how to delete the multiple records or rows from the database table.**

DELETE FROM Student WHERE Stu\_Marks > 70 ;

## DCL Commands in SQL

Two types of DCL commands can be used by the user in SQL. These commands are useful, especially when several users access the database. It enables the administrator to manage access control. The two types of DCL commands are as follows:

* GRANT
* REVOKE

## GRANT Command

GRANT, as the name itself suggests, provides. This command allows the administrator to provide particular privileges or permissions over a database object, such as a table, view, or procedure. It can provide user access to perform certain database or component operations.

In simple language, the GRANT command allows the user to implement other SQL commands on the database or its objects. The primary function of the GRANT command in SQL is to provide administrators the ability to ensure the security and integrity of the data is maintained in the database.

GRANT SELECT ON student TO Aman;

This command will allow Aman to implement the SELECT queries on the student table. This will enable the user to read or retrieve information from the student table.

## REVOKE Command

As the name suggests, revoke is to take away. The REVOKE command enables the database administrator to remove the previously provided privileges or permissions from a user over a database or database object, such as a table, view, or procedure. The REVOKE commands prevent the user from accessing or performing a specific operation on an element in the database.

In simple language, the REVOKE command terminates the ability of the user to perform the mentioned SQL command in the REVOKE query on the database or its component. The primary reason for implementing the REVOKE query in the database is to ensure the data's security and integrity.

1. REVOKE SELECT ON student FROM Aman;

This will stop the user Aman from implementing the SELECT query on the student table. The user may be able to implement other queries in the database

# TCL Commands in SQL

* In SQL, TCL stands for **Transaction control language**.
* A single unit of work in a database is formed after the consecutive execution of commands is known as a transaction.
* There are certain commands present in SQL known as TCL commands that help the user manage the transactions that take place in a database.
* **COMMIT. ROLLBACK** and **SAVEPOINT** are the most commonly used TCL commands in SQL.

Now let us take a deeper dive into the TCL commands of SQL with the help of examples. All the queries in the examples will be written using the MySQL database.

### 1. COMMIT

COMMIT command in SQL is used to save all the transaction-related changes permanently to the disk. Whenever DDL commands such as INSERT, UPDATE and DELETE are used, the changes made by these commands are permanent only after closing the current session. So before closing the session, one can easily roll back the changes made by the DDL commands. Hence, if we want the changes to be saved permanently to the disk without closing the session, we will use the commit command.

**Syntax:**

COMMIT;

**Example:**

mysql> COMMIT;

### SAVEPOINT

We can divide the database operations into parts. For example, we can consider all the insert related queries that we will execute consecutively as one part of the transaction and the delete command as the other part of the transaction. Using the SAVEPOINT command in SQL, we can save these different parts of the same transaction using different names. For example, we can save all the insert related queries with the savepoint named INS. To save all the insert related queries in one savepoint, we have to execute the SAVEPOINT query followed by the savepoint name after finishing the insert command execution.

**Syntax:**

SAVEPOINT savepoint\_name;

### 3. ROLLBACK

While carrying a transaction, we must create savepoints to save different parts of the transaction. According to the user's changing requirements, he/she can roll back the transaction to different savepoints. Consider a scenario: We have initiated a transaction followed by the table creation and record insertion into the table. After inserting records, we have created a savepoint INS. Then we executed a delete query, but later we thought that mistakenly we had removed the useful record. Therefore in such situations, we have an option of rolling back our transaction. In this case, we have to roll back our transaction using the ROLLBACK command to the savepoint INS, which we have created before executing the DELETE query.

ADVERTISEMENT

**Syntax:**

ROLLBACK TO savepoint\_name;

# SQL WHERE

A WHERE clause in SQL is a data manipulation language statement.

WHERE clauses are not mandatory clauses of SQL DML statements. But it can be used to limit the number of rows affected by a SQL DML statement or returned by a query.

WHERE clause is used in SELECT, UPDATE, DELETE statement etc.

SELECT column1, column 2, ... column n  FROM    table\_name  WHERE [conditions]

SQL AND

* The SQL AND condition is used in SQL query to create two or more conditions to be met.
* It is used in SQL SELECT, INSERT, UPDATE and DELETE

### SQL "AND" example with "SELECT" statement

mysql> SELECT \*FROM emp WHERE Department = "IT" AND Location = "Chennai";

# SQL OR

The SQL **OR** condition is used in SQL query to create a SQL statement where records are returned when any one condition met. It can be used in a **SELECT** statement, **INSERT** statement, **UPDATE** statement or **DELETE** statement.

**Let's see the syntax for the OR condition:**

SELECT columns FROM tables WHERE condition 1 OR condition 2;

Example:

mysql> SELECT \*FROM emp WHERE Department = "IT" OR Location = "Chennai";

SQL SELECT AS

* SQL 'AS' is used to assign a new name temporarily to a table column or even a table.
* It makes an easy presentation of query results and allows the developer to label results more accurately without permanently renaming table columns or even the table itself.

SELECT day\_of\_order AS 'Date', Customer As 'Client', Product, Quantity FROM orders;

# ORDER BY Clause

* Whenever we want to sort the records based on the columns stored in the tables of the SQL database, then we consider using the ORDER BY clause in SQL.
* The ORDER BY clause in SQL will help us to sort the records based on the specific column of a table. This means that all the values stored in the column on which we are applying ORDER BY clause will be sorted, and the corresponding column values will be displayed in the sequence in which we have obtained the values in the earlier step.
* Using the ORDER BY clause, we can sort the records in ascending or descending order as per our requirement. The records will be sorted in ascending order whenever the ASC keyword is used with ORDER by clause. DESC keyword will sort the records in descending order.

### Syntax to sort the records in ascending order:

SELECT ColumnName1,...,ColumnNameN FROM TableName  ORDER BY ColumnName ASC;

### Syntax to sort the records in descending order:

SELECT ColumnName1,...,ColumnNameN FROM TableName  ORDER BY ColumnNameDESC;

### Syntax to sort the records in ascending order without using ASC keyword:

SELECT ColumnName1,...,ColumnNameN FROM TableName  ORDER BY ColumnName;

### Example 1:

mysql> SELECT \*FROM customers ORDER BY Name ASC;

### Example 2:

mysql> SELECT \*FROM customers ORDER BY Address

### Example 3:

mysql> SELECT \*FROM customers ORDER BY Salary DESC;

# HAVING Clause in SQL

The HAVING clause places the condition in the groups defined by the GROUP BY clause in the SELECT statement.

This SQL clause is implemented after the 'GROUP BY' clause in the 'SELECT' statement.

This clause is used in SQL because we cannot use the WHERE clause with the SQL aggregate functions. Both WHERE and HAVING clauses are used for filtering the records in SQL queries.

### Difference between HAVING and WHERE Clause

The difference between the WHERE and HAVING clauses in the database is the most important question asked during an IT interview.

The following table shows the comparisons between these two clauses, but the main difference is that the [WHERE clause](https://www.javatpoint.com/sql-where) uses condition for filtering records before any groupings are made, while HAVING clause uses condition for filtering values from a group.

|  |  |
| --- | --- |
| HAVING | WHERE |
| 1. The HAVING clause is used in database systems to fetch the data/values from the groups according to the given condition. | 1. The WHERE clause is used in database systems to fetch the data/values from the tables according to the given condition. |
| 2. The HAVING clause is always executed with the GROUP BY clause. | 2. The WHERE clause can be executed without the GROUP BY clause. |
| 3. The HAVING clause can include SQL aggregate functions in a query or statement. | 3. We cannot use the SQL aggregate function with WHERE clause in statements. |
| 4. We can only use SELECT statement with HAVING clause for filtering the records. | 4. Whereas, we can easily use WHERE clause with UPDATE, DELETE, and SELECT statements. |
| 5. The HAVING clause is used in SQL queries after the GROUP BY clause. | 5. The WHERE clause is always used before the GROUP BY clause in SQL queries. |
| 6. We can implements this SQL clause in column operations. | 6. We can implements this SQL clause in row operations. |
| 7. It is a post-filter. | 7. It is a pre-filter. |
| 8. It is used to filter groups. | 8. It is used to filter the single record of the table. |

### Syntax of HAVING clause in SQL

1. SELECT column\_Name1, column\_Name2, ....., column\_NameN aggregate\_function\_name(column\_Name) FROM table\_name GROUP BY column\_Name1 HAVING condition;

Example

1. SELECT SUM(Emp\_Salary), Emp\_City FROM Employee GROUP BY Emp\_City;
2. SELECT SUM(Emp\_Salary), Emp\_City FROM Employee GROUP BY Emp\_City HAVING SUM(Emp\_Salary)>5000;
3. SELECT COUNT(Roll\_No), Age FROM Student\_details GROUP BY Age ;
4. SELECT COUNT(Roll\_No), Age FROM Student\_details GROUP BY Age HAVING COUNT(Roll\_No) >= 2 ;

# COUNT

The **SQL COUNT()** is a function that returns the number of records of the table in the output.

This function is used with the SQL SELECT statement.

SELECT COUNT (Bike\_Color) AS TotalBikeColor FROM Bikes ;

# SUM

It is also known as SQL SUM() function. It is used in a SQL query to return summed value of an expression.

1. SELECT SUM (salary) AS "Total Salary"  FROM employees  WHERE salary > 20000;

1. SELECT department, SUM (sales) AS "Total Sales"  FROM order\_details  GROUP BY department;