**SQL Query**

**Batch Code:7670**

**Kamal Dukare.**

**DBMS:**

A database is an organized collection of structured information, or data, typically stored electronically in a computer system. A database is usually controlled by a database management system (DBMS).

Structured Query Language (SQL) as we all know is the database language by the use of which we can perform certain operations on the existing database and also we can use this language to create a database. [SQL](https://www.geeksforgeeks.org/structured-query-language/) uses certain commands like Create, Drop, Insert, etc. to carry out the required tasks.

These [SQL](https://www.geeksforgeeks.org/sql-concepts-and-queries/)commands are mainly categorized into four categories as:

1. DDL – Data Definition Language
2. DQl – Data Query Language/ DRL Data Retrieval Language
3. DML – Data Manipulation Language
4. DCL – Data Control Language

Though many resources claim there to be another category of SQL clauses

**TCL(Transaction Control Language)**. So we will see in detail about TCL as well. 

**DDL (Data Definition Language):**

[DDL](https://www.geeksforgeeks.org/features-of-structured-query-language-sql/) or Data Definition Language actually consists of the SQL commands that can be used to define the database schema. It simply deals with descriptions of the database schema and is used to create and modify the structure of database object in the database. DDL is a set of SQL commands used to create, modify, and delete database structures but not data. These commands are normally not used by a general user, who should be accessing the database via an application.

List of DDL commands:

* [**CREATE**](https://www.geeksforgeeks.org/sql-create/): This command is used to create the database or its objects (like table, index, function, views, store procedure, and triggers).
* [**DROP**](https://www.geeksforgeeks.org/sql-drop-truncate/): This command is used to delete objects from the database.
* [**ALTER**](https://www.geeksforgeeks.org/sql-alter-add-drop-modify/)**:**This is used to alter the structure of the database.
* [**TRUNCATE**](https://www.geeksforgeeks.org/sql-drop-truncate/)**:**This is used to remove all records from a table, including all spaces allocated for the records are removed.
* [**COMMENT**](https://www.geeksforgeeks.org/sql-comments/): This is used to add comments to the data dictionary.
* [**RENAME**](https://www.geeksforgeeks.org/sql-alter-rename/)**:**This is used to rename an object existing in the database.

**DQL (Data Query Language): Data Retrieval language.**

**DQL**statements are used for performing queries on the data within schema objects. The purpose of the DQL Command is to get some schema relation based on the query passed to it. We can define DQL as follows it is a component of SQL statement that allows getting data from the database and imposing order upon it. It includes the SELECT statement. This command allows getting the data out of the database to perform operations with it. When a SELECT is fired against a table or tables the result is compiled into a further temporary table, which is displayed or perhaps received by the program i.e. a front-end.

List of DQL:

* [**SELECT**](https://www.geeksforgeeks.org/sql-select-clause/)**:**It is used to retrieve data from the database.

**DML(Data Manipulation Language):**

The SQL commands that deals with the manipulation of data present in the database belong to DML or Data Manipulation Language and this includes most of the SQL statements. It is the component of the SQL statement that controls access to data and to the database. Basically, DCL statements are grouped with DML statements.

List of DML commands:

* [**INSERT**](https://www.geeksforgeeks.org/sql-insert-statement/) : It is used to insert data into a table.
* [**UPDATE**](https://www.geeksforgeeks.org/sql-update-statement/)**:** It is used to update existing data within a table.
* [**DELETE**](https://www.geeksforgeeks.org/sql-delete-statement/) : It is used to delete records from a database table.
* [**LOCK:**](https://www.geeksforgeeks.org/sql-lock-table/) Table control concurrency.
* **CALL:**Call a PL/SQL or JAVA subprogram.
* **EXPLAIN PLAN:** It describes the access path to data.

**DCL (Data Control Language):**

DCL includes commands such as GRANT and REVOKE which mainly deal with the rights, permissions, and other controls of the database system.

List of  DCL commands:

* [**GRANT:**](https://www.geeksforgeeks.org/mysql-grant-revoke-privileges/)This commandgives users access privileges to the database.
* [**REVOKE:**](https://www.geeksforgeeks.org/difference-between-grant-and-revoke/)This command withdraws the user’s access privileges given by using the GRANT command.

Though many resources claim there to be another category of SQL clauses TCL – Transaction Control Language. So we will see in detail about TCL as well. TCL commands deal with the [transaction within the database](https://www.geeksforgeeks.org/sql-transactions/).

List of TCL commands: 

* [**COMMIT**](https://www.geeksforgeeks.org/sql-transactions/)**:**Commits a Transaction.
* [**ROLLBACK**](https://www.geeksforgeeks.org/sql-transactions/)**:**Rollbacks a transaction in case of any error occurs.
* [**SAVEPOINT**](https://www.geeksforgeeks.org/sql-transactions/)**:** Sets a save point within a transaction.
* [**SET TRANSACTION:**](https://www.geeksforgeeks.org/sql-transactions/)Specify characteristics for the transaction.

**DDL COMMANDS:**

**1.Create Table:**

**Syntax:**

CREATE TABLE table\_name (  
   column1 datatype,  
   column2 datatype,  
   column3 datatype,  
  ....  
);

**Example:**

CREATE TABLE Persons (  
    PersonID int,  
    LastName varchar(255),  
    FirstName varchar(255),  
    Address varchar(255),  
    City varchar(255)  
);

**2.Drop table:**

**Syntax:**

DROP TABLE table\_name;

**Example:**

DROP TABLE PERSONS;

**3.ALTER TABLE**

* **ADD Column:**

**Syntax:**

ALTER TABLE table\_name

ADD column\_name datatype;

**Example:**

ALTER TABLE Persons

ADD Email varchar(255);

**DROP COLUMN:**

**Syntax:**

ALTER TABLE table\_name

DROP COLUMN column\_name;

**Example:**

ALTER TABLE Customers

DROP COLUMN Email;

**4.TRUNCATE TABLE:**

**Syntax:**

TRUNCATE [TABLE] table\_name;

**Example:**

TRUNCATE TABLE Persons;

**5.RENAME TABLE:**

**Syntax:**

1.RENAME tableName TO newTableName;

2.ALTER tableName RENAME TO newTableName;

**Example:**

1.RENAME Persons TO Customers;

2.ALTER Persons RENAME TO Customers;

**DQL Command:**

**Select table:**

**Syntax:**

1.SELECT \* FROM *table\_name*;

2.SELECT column1, column2, ...

FROM table\_name;

**Example:**

1.Select \* from Persons;

2.SELECT Address, City, ...

FROM Persons;

* **Select with where clause**:

**Syntax:**

SELECT \* FROM Name\_of\_Table WHERE [condition];

**Example:**

SELECT \* FROM Customers

WHERE Country='Mexico';

* **SELECT DISTINCT:**

**Syntax:**

SELECT DISTINCT column\_name FROM table\_name;

**Examples:**

SELECT DISTINCT fname FROM Persons;

* **GROUP BY:**

**Syntax:**

SELECT column\_name(s)

FROM table\_name

WHERE condition

GROUP BY column\_name(s)

ORDER BY column\_name(s);

**Example:**

SELECT COUNT(CustomerID), Country

FROM Customers

GROUP BY Country;

* **ORDER BY:**

**Syntax:**

SELECT column1, column2, ...

FROM table\_name

ORDER BY column1, column2, ... ASC|DESC;

**Example:**

SELECT \* FROM Customers

ORDER BY Country;

* **LIKE:**

**Syntax:**

SELECT column1, column2, ...

FROM table\_name

WHERE columnN LIKE pattern;

**Example:**

SELECT \* FROM Customers

WHERE CustomerName LIKE 'a%';

* **TOP Clause:**

**Syntax:**

SELECT TOP number|percent column\_name(s)

FROM table\_name

WHERE condition;

**Example:**

1.SELECT TOP 3 \* FROM Customers;

2.SELECT TOP 3 \* FROM Customers

WHERE Country='Germany';

* **LIMIT Clause:**

**Syntax:**

SELECT column\_name(s)

FROM table\_name

WHERE condition

LIMIT number;

**Example:**

1.SELECT \* FROM Customers

LIMIT 3;

2.SELECT \* FROM Customers

WHERE Country='Germany'

LIMIT 3;

* **MIN() and MAX() Functions:**

**MIN() Syntax:**

SELECT MIN(column\_name)

FROM table\_name

WHERE condition;

**Example:**

SELECT MIN(Price) AS SmallestPrice

FROM Products;

**MAX() Syntax:**

SELECT MAX(column\_name)

FROM table\_name

WHERE condition;

**Example:**

SELECT MAX(Price) AS LargestPrice

FROM Products;

* **COUNT(), AVG() and SUM() Functions:**

**COUNT() Syntax:**

SELECT COUNT(column\_name)

FROM table\_name

WHERE condition;

**Example:**

SELECT COUNT(ProductID)

FROM Products;

**AVG() Syntax:**

SELECT AVG(column\_name)

FROM table\_name

WHERE condition;

**Example:**

SELECT AVG(Price)

FROM Products;

**SUM() Syntax:**

SELECT SUM(column\_name)

FROM table name

WHERE condition;

**Example:**

SELECT SUM(Quantity)

FROM OrderDetails;

**BETWEEN:**

**Syntax:**

SELECT column\_name(s)

FROM table\_name

WHERE column\_name BETWEEN value1 AND value2;

**Syntax:**

SELECT \* FROM Products

WHERE Price BETWEEN 10 AND 20;

**JOIN:**

A join clause is used to combine rows from two or more tables, based on a related column between them.

**1. INNER Join:**

The INNER JOIN keyword selects records that have matching values in both tables.

**Syntax:**

SELECT column\_name(s)

FROM table1

INNER JOIN table2

ON table1.column\_name = table2.column\_name;

**Example:**

SELECT Orders.OrderID, Customers.CustomerName

FROM Orders

INNER JOIN Customers ON Orders.CustomerID = Customers.CustomerID;

**2. Left Join:**

The LEFT JOIN keyword returns all records from the left table (table1), and the matching records from the right table (table2). The result is 0 records from the right side, if there is no match.

**Syntax:**

SELECT column\_name(s)

FROM table1

LEFT JOIN table2

ON table1.column\_name = table2.column\_name;

**Example:**

SELECT Customers.CustomerName, Orders.OrderID

FROM Customers

LEFT JOIN Orders ON Customers.CustomerID = Orders.CustomerID

ORDER BY Customers.CustomerName;

**3.Right Join:**

The RIGHT JOIN keyword returns all records from the right table (table2), and the matching records from the left table (table1). The result is 0 records from the left side, if there is no match.

**syntax:**

SELECT table1.column1, table2.column2.....

FROM table1

RIGHT JOIN table2

ON table1.column\_field = table2.column\_field;

**Example:**

SELECT ID,NAME,AMOUNT,DATE

FROM CUSTOMER

RIGHT JOIN ORDER

ON CUSTOMER.ID = ORDER.CUSTOMER\_ID;

**4.Full Join:**

The FULL OUTER JOIN keyword returns all records when there is a match in left (table1) or right (table2) table records.

Syntax:

SELECT column\_name(s)

FROM table1

FULL OUTER JOIN table2

ON table1.column\_name = table2.column\_name

WHERE condition;

**Example:**

SELECT Customers.CustomerName, Orders.OrderID

FROM Customers

FULL OUTER JOIN Orders ON Customers.CustomerID=Orders.CustomerID

ORDER BY Customers.CustomerName;

### **Alias:**

SQL aliases are used to give a table, or a column in a table, a temporary name.

Aliases are often used to make column names more readable.

An alias only exists for the duration of that query.

An alias is created with the AS keyword.

### **Syntax:**

SELECT column\_name*AS* alias\_name  
FROM table\_name;

**Example:**

SELECT CustomerID AS ID, CustomerName AS Customer  
FROM Customers:

## **IN Operator:**

The IN operator allows you to specify multiple values in a WHERE clause.

**Syntax:**

SELECT column\_name(s)  
FROM table\_name  
WHERE column\_name IN *(*value1*,* value2*, ...);*

**Example:**

SELECT \* FROM Customers  
WHERE Country NOT IN ('Germany', 'France', 'UK');

## **UNION Operator:**

The UNION operator is used to combine the result-set of two or more SELECT statements.

**Syntax:**

SELECT column\_name(s)FROMtable1  
UNION  
SELECT column\_name(s) FROM table2;

**Example:**

SELECT City FROM Customers  
UNION  
SELECT City FROM Suppliers  
ORDER BY City;

# **HAVING Clause:**

The HAVING clause was added to SQL because the WHERE keyword cannot be used with aggregate functions.

**Syntax:**

SELECT column\_name(s)FROM table\_name  
WHERE condition  
GROUP BY column\_name(s)HAVING conditionORDER BY column\_name(s);

**Example:**

SELECT COUNT(CustomerID), Country  
FROM Customers  
GROUP BY Country  
HAVING COUNT(CustomerID) > 5;

**DML Command:**

**INSERT INTO Statement:**

**Syntax:**

INSERT INTO table\_name (column1, column2, column3, ...)

VALUES (value1, value2, value3, ...);

**Example:**

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

VALUES ('Cardinal', 'Tom B. Erichsen', 'Skagen 21', 'Stavanger', '4006', 'Norway');

**UPDATE Table:**

**Syntax:**

UPDATE table name

SET column1 = value1, column2 = value2, ...

WHERE condition;

**Example:**

UPDATE Customers

SET ContactName = 'Alfred Schmidt', City= 'Frankfurt'

WHERE CustomerID = 1;

**DELETE Table:**

**Syntax:**

DELETE FROM table\_name WHERE condition;

**Example:**

DELETE FROM Customers WHERE CustomerName='Alfreds Futterkiste';

**DCL Commands:**

**Grant Command:**

**Syntax:**

GRANT privilege\_name

ON object\_name

TO {user\_name |PUBLIC |role\_name}

[WITH GRANT OPTION];

**Example:**

GRANT SELECT ON employee TO user1

**REVOKE Command:**

**Syntax:**

REVOKE privilege\_name

ON object\_name

FROM {user\_name |PUBLIC |role\_name}

**Example:**

REVOKE SELECT ON employee FROM user1;

**TCL Commands:**

**Commit Command:**

Commit**;**

**ROLLBACK command:**

ROLLBACK TO savepoint\_name;

**SAVEPOINT command:**

SAVEPOINT savepoint\_name;