

Search...

[Databases](#) [SQL](#) [MySQL](#) [PostgreSQL](#) [PL/SQL](#) [MongoDB](#) [SQL Cheat Sheet](#)[Sign In](#)

Database Languages in DBMS

Last Updated : 29 Jan, 2025

Databases are essential for efficiently **storing, managing, and retrieving** large volumes of data. They utilize both **software** and **hardware** components. The **software** provides an interface that enables users or applications to interact with the database, while the **hardware** consists of servers and storage systems responsible for physically storing and organizing the data, either in memory or on physical devices like hard drives.

In this article, we will explore the **types of database languages** used in a **Database Management System (DBMS)**. These languages are crucial for performing operations such as **data creation, manipulation, retrieval, and management**. They provide the tools necessary to define, control, and manipulate the data effectively.

What is a Database?

A **database** is a structured collection of data stored **electronically**. It allows users to easily **access, manage, and update** data. For example, a phone contact list on our smartphone is a small database. It keeps track of names, phone numbers, and addresses, allowing us to search and access this information quickly when needed. Databases are widely used in businesses, websites, and applications to store and manage large volumes of data efficiently.

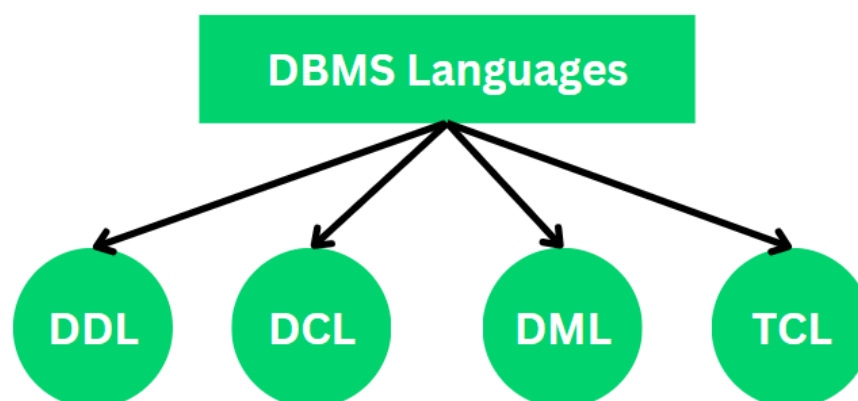
A **Database Management System (DBMS)** is the software that interacts with the database, providing an interface to users and applications. It ensures the efficient storage and retrieval of data, maintains data integrity, and provides security features like access control and authentication.

Types of Database Languages in DBMS

Database languages are specialized languages used to interact with a database. They allow users to perform different tasks such as **defining**, **controlling**, and **manipulating** the data. There are several types of database languages in DBMS, categorized into the following four main types:

1. DDL (Data Definition Language)
2. DCL (Data Control Language)
3. DML (Data Manipulation Language)
4. TCL (Transaction Control Language)

Each category serves a different purpose within the database management process. Let's break down each one.



Type of Database Language

1. DDL (Data Definition Language)

The DDL stands for Data Definition Language, Which is used to define the database's internal structure and Pattern of the Database. It is used to define and **modify** the structure of the database itself, including the **tables**, **views**, **indexes**, and other schema-related objects. It deals with the creation and modification of database schema, but it doesn't deal with the data itself.

Following are the five DDL commands in SQL:

- **CREATE**: Used to create database objects like tables, indexes, or views.

- **ALTER:** Used to modify the structure of an existing database object, such as adding a new column to a table.
- **DROP:** Used to delete database objects.
- **TRUNCATE:** Used to remove all rows from a table, without affecting the structure.
- **RENAME:** Used to change the name of a database object.

Now we will explain each command with examples for better understanding of the concepts.

CREATE Command

The CREATE is a DDL command used to create **databases**, **tables**, [triggers](#) and other database objects.

Syntax

```
CREATE TABLE Students (  
    column1 INT,  
    column2 VARCHAR(50),  
    column3 INT  
);
```

Example:

```
mysql> CREATE TABLE Students (  
->     StudentID INT PRIMARY KEY,  
->     Name VARCHAR(50),  
->     Age INT  
-> );  
Query OK, 0 rows affected (0.05 sec)  
  
mysql> desc students;  
+-----+-----+-----+-----+-----+-----+  
| Field      | Type          | Null | Key | Default | Extra |  
+-----+-----+-----+-----+-----+-----+  
| StudentID  | int           | NO   | PRI | NULL    |       |  
| Name       | varchar(50)   | YES  |     | NULL    |       |  
| Age        | int           | YES  |     | NULL    |       |  
+-----+-----+-----+-----+-----+-----+  
3 rows in set (0.01 sec)
```

create table

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.

Syntax

ALTER TABLE Students ADD column_name;

Example:

```
mysql> ALTER TABLE Students
-> ADD Weight DECIMAL(5,2);
Query OK, 0 rows affected (0.06 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> desc students;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| StudentID | int           | NO   | PRI | NULL    |       |
| Name      | varchar(50)   | YES  |     | NULL    |       |
| Age       | int           | YES  |     | NULL    |       |
| Weight    | decimal(5,2) | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)
```

Alter command

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.

Syntax

DROP Table Table_name;

Example:

```
mysql> drop table students;
Query OK, 0 rows affected (0.05 sec)
```

drop

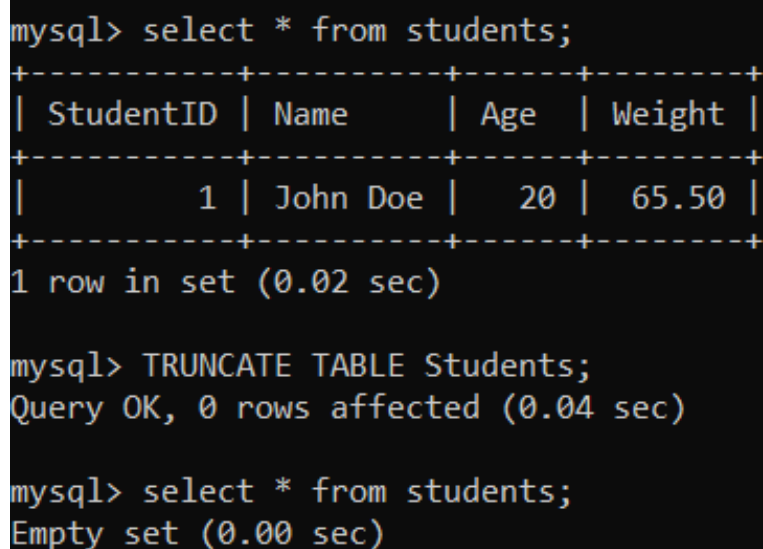
Truncate Command

The TRUNCATE command is used to **delete all the records** from a table without removing the structure. Unlike the **DELETE** command, which can remove specific rows and can be rolled back, [TRUNCATE](#) is a more efficient way to delete all rows in a table without logging individual row deletions.

Syntax

```
TRUNCATE TABLE table_name;
```

Example:



```
mysql> select * from students;
+-----+-----+-----+-----+
| StudentID | Name      | Age  | Weight |
+-----+-----+-----+-----+
|          1 | John Doe | 20   | 65.50  |
+-----+-----+-----+-----+
1 row in set (0.02 sec)

mysql> TRUNCATE TABLE Students;
Query OK, 0 rows affected (0.04 sec)

mysql> select * from students;
Empty set (0.00 sec)
```

truncate

Rename Command

The **RENAME** command in a Database Management System (DBMS) is used to change the name of a database **object**, such as a **table**, **column**, or index. This command is helpful when we want to give a more meaningful or appropriate name to an object without needing to recreate it.

Syntax:

```
ALTER TABLE Old_Table_Name RENAME TO New_Table_Name;
```

Example:

```
mysql> ALTER TABLE Students  
-> RENAME TO ClassMembers;  
Query OK, 0 rows affected (0.02 sec)
```

Rename

DCL (Data Control Language)

DCL stands for [Data Control Language](#). It is used to control the access permissions of users to the database. DCL commands help grant or revoke privileges to users, determining who can perform actions like reading or modifying data. DCL commands are **transactional**, meaning they can be rolled back if necessary.

The two main DCL commands are:

- **Grant:** Gives user access to the database
- **Revoke:** Removes access or permissions from the user

Now we will explain these commands with proper examples for better understanding

Grant Command

The **GRANT** command in a Database Management System (DBMS) is used to provide specific **permissions** or **privileges** to users or roles. This command allows administrators to control access to **database objects**, ensuring that only authorized users can perform certain actions, such as selecting, inserting, updating, or deleting data.

Syntax

```
GRANT privileges  
ON object  
TO user_or_role [WITH GRANT OPTION];
```

Example:

```
GRANT SELECT, INSERT ON students TO user;
```

Revoke Command

The **REVOKE** command in a Database Management System (DBMS) is used to remove previously granted permissions or privileges from users or roles. This command is essential for **managing access control**, ensuring that users do not have more privileges than necessary to perform their tasks.

Syntax

REVOKE privileges ON object FROM user_or_role;

Example

```
REVOKE ALL PRIVILEGES ON students FROM user;
```

DML (Data Manipulation Language)

The DML ([Data Manipulation Language](#)) is used to manage and manipulate data within a database. With DML, you can perform various operations such as inserting, updating, selecting, and deleting data. These operations allow you to work with the actual content in your database tables.

Here are the key DML commands:

- **SELECT**: Retrieves data from the table based on specific criteria.
- **INSERT**: Adds new rows of data into an existing table.
- **UPDATE**: Modifies existing data in a table.
- **DELETE**: Removes data from a table.
- **MERGE**: Performs an "upsert" operation, which means updating existing records or inserting new ones if they don't exist.
- **CALL**: Executes stored procedures or functions.
- **LOCK TABLE**: Prevents other users from accessing the table while changes are being made.

Now, we will explain these commands with examples for better understanding:

SELECT Command

The [SELECT command](#) in SQL (Structured Query Language) is used to retrieve data from one or more tables in a database. It is the most

commonly used commands in SQL, allowing users to specify which columns and rows of data they want to retrieve and how they want that data organized. The **SELECT** statement can be used in various ways, such as selecting all data from a table, filtering records based on conditions, or sorting the results.

Syntax

*SELECT * FROM Table_Name*

Example:

```
mysql> select * from ClassMembers;
+-----+-----+-----+-----+
| StudentID | Name      | Age  | Weight |
+-----+-----+-----+-----+
|          1 | John Doe | 20   | 65.50  |
+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

Select Command

Insert Command

The **INSERT** command in SQL (Structured Query Language) is used to add new records or rows to a table in a database. It is a key operation in database management, essential for populating tables with new data. This command can insert data into all columns or specific columns of a table.

Syntax

INSERT INTO Table_Name (Column 1, Column 2, Column 3, Column 4) VALUES (Value 1, Value 2, Value 3, Value 4);

Example:


```
mysql> INSERT INTO ClassMembers (StudentID, Name, Age, Weight)
-> VALUES (2, 'Alice Smith', 22, 60.2);
Query OK, 1 row affected (0.03 sec)

mysql> select * from ClassMembers;
+-----+-----+-----+-----+
| StudentID | Name       | Age | Weight |
+-----+-----+-----+-----+
|          1 | John Doe   | 20  | 65.50  |
|          2 | Alice Smith | 22  | 60.20  |
+-----+-----+-----+-----+
2 rows in set (0.00 sec)
```

Insert command

Update Command

The [UPDATE command](#) in SQL (Structured Query Language) is used to modify existing records in a table. This command enables users to change the values of one or more columns for specific rows based on a condition (criteria). It is crucial for maintaining and adjusting data in a database when necessary.

Syntax

UPDATE Table_Name SET Name = 'New_Value' WHERE Name = 'Old_Value';

Example:

```
mysql> UPDATE ClassMembers
-> SET Name = 'Roman'
-> WHERE Name = 'John Doe';
Query OK, 1 row affected (0.03 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from ClassMembers;
+-----+-----+-----+-----+
| StudentID | Name       | Age | Weight |
+-----+-----+-----+-----+
|          1 | Roman      | 20  | 65.50  |
|          2 | Alice Smith | 22  | 60.20  |
+-----+-----+-----+-----+
2 rows in set (0.00 sec)
```

Update

Delete Command

The [DELETE command](#) in SQL (Structured Query Language) is used to remove one or more existing records from a table in a database. It is an essential operation for managing data, enabling users to delete specific rows that meet certain conditions or criteria.

Syntax:

DELETE FROM Table_Name WHERE Column = Value;

Example:

```
mysql> DELETE FROM ClassMembers
      -> WHERE StudentID = 2;
Query OK, 1 row affected (0.03 sec)

mysql> select * from ClassMembers;
+-----+-----+-----+-----+
| StudentID | Name  | Age  | Weight |
+-----+-----+-----+-----+
|          1 | Roman | 20   | 65.50  |
+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

Delete Command

Merge Command

The [Merge command](#) in SQL is used to perform an **upsert** operation, which combines both UPDATE and INSERT actions. It allows you to insert new rows if they do not exist, or update existing rows if they match a certain condition. This command is particularly useful for synchronizing two tables by inserting new records and updating existing ones in a single operation.

Syntax:

*MERGE INTO target_table AS target
USING source_table AS source
ON (target.id = source.id)
WHEN MATCHED THEN
UPDATE SET target.name = source.name*

```
WHEN NOT MATCHED THEN  
    INSERT (id, name) VALUES (source.id, source.name);
```

Example:

```
MERGE INTO Employees AS target  
USING New_Hires AS source  
ON target.EmployeeID = source.EmployeeID  
WHEN MATCHED THEN  
    UPDATE SET target.Name = source.Name, target.Salary =  
    source.Salary  
WHEN NOT MATCHED THEN  
    INSERT (EmployeeID, Name, Salary)  
    VALUES (source.EmployeeID, source.Name, source.Salary);
```

Table 1. Employees Table (Before Merge):

EmployeeID	Name	Salary
1	John	50000
2	Jane	55000
3	Alice	60000

Employee Table

Table 2: New_Hires Table:

EmployeeID	Name	Salary
2	Jane	58000
3	Alice	65000
4	Bob	45000

New_Hires Table

Output

EmployeeID	Name	Salary
1	John	50000
2	Jane	58000
3	Alice	65000
4	Bob	45000

Employee Table (After Merge)

CALL Command

The [Call command](#) is used to invoke a stored procedure or user-defined function, which is a set of precompiled SQL statements. It allows you to

execute complex operations within a database as a single unit.

Syntax:

CALL user_defined_function(parameter 1, parameter 2);

Example:

```
CALL UpdateEmployeeSalary(101, 55000);
```

LOCK TABLE

The [lock table](#) command is used to lock the table for preventing access from others, ensuring no other operations (like insertions, updates, or deletions) can be performed on the table while it's locked. Useful for transactional operations where consistency is important.

Syntax:

LOCK TABLE your_table IN EXCLUSIVE MODE;

Example:

```
LOCK TABLE ClassMembers IN EXCLUSIVE MODE;
```

TCL (Transaction Control Language)

The TCL full form is [Transaction Control Language](#) commands are used to manage and control transactions in a database, grouping them into logical units. These commands help ensure the integrity of data and consistency during complex operations. Here are the two main commands in this category:

- **Commit:** Saves all the changes made during the current transaction to the database. These are very useful in the banking sector.
- **Rollback:** used to restore the database to its original state from the last commit. This command also plays an important role in Banking Sectors.

Now we will explain these two commands for better understanding with examples

Commit Command

The **COMMIT command** is used to save all changes made during a transaction in the database. This command ensures that modifications made by DML statements (such as INSERT, UPDATE, or DELETE) become permanent in the database.

Syntax

Database Operation

Commit

Example:

```
mysql> INSERT INTO ClassMembers (StudentID, Name, Age, Weight)
      -> VALUES (4, 'Emma Johnson', 23, 50);
Query OK, 1 row affected (0.03 sec)

mysql>
mysql> COMMIT;
Query OK, 0 rows affected (0.00 sec)
```

commit

ROLLBACK Command

The [rollback command](#) is used to restore the database to its state at the last COMMIT, effectively undoing any changes made since that point. It helps ensure data consistency by allowing the reversal of partial or erroneous operations.

Syntax

ROLLBACK;

Example:

```
mysql> START TRANSACTION;
Query OK, 0 rows affected (0.00 sec)

mysql> commit;
Query OK, 0 rows affected (0.00 sec)

mysql> DELETE FROM ClassMembers where StudentID = 4;
Query OK, 1 row affected (0.03 sec)

mysql> rollback;
Query OK, 0 rows affected (0.00 sec)
```

rollback

Conclusion

In conclusion, database languages play a crucial role in managing and manipulating data in a database management system ([DBMS](#)). There are different type of database language like [DDL](#), [DCL](#), [DML](#), and [TCL](#). The DDL language is used for defining the database's internal structure and Pattern of the Database. The DCL commands are used to control the data from the user means They can provide control of the Database, Table, and Data. The DML commands are used to manipulate the Data in the Table like Inserting, updating, and deleting the Data finally the TCL commands are used to save and restore the previous state of the Database.

Comment

More info

Advertise with us

Next Article

Purpose of Database System in
DBMS

Similar Reads