| | |
|---|---|
| Database Connection | • Launch MySQL Workbench<br>• Connect to MySQL Server<br>• Creating a new Database<br>• Data Types<br>• CAST or CONVERT<br>• Keys in SQL<br>• Constraints |

MySQL is a relational database management system based on the Structured Query Language, which is the popular language for accessing and managing the records in the database. MySQL is open-source and free software under the GNU license. It is supported by **Oracle Company**.


**What is MySQL?**

MySQL is currently the most popular database management system software used for managing the relational database. It is open-source database software, which is supported by Oracle Company. It is fast, scalable, and easy to use database management system in comparison with Microsoft SQL Server and Oracle Database. It is commonly used in conjunction with PHP scripts for creating powerful and dynamic server-side or web-based enterprise applications.

It is developed, marketed, and supported by **MySQL AB, a Swedish company**, and written in C programming language and C++ programming language. The official pronunciation of MySQL is not the My Sequel; it is ***My Ess Que Ell***. *However, you can pronounce it in your way*. Many small and big companies use MySQL. MySQL supports many

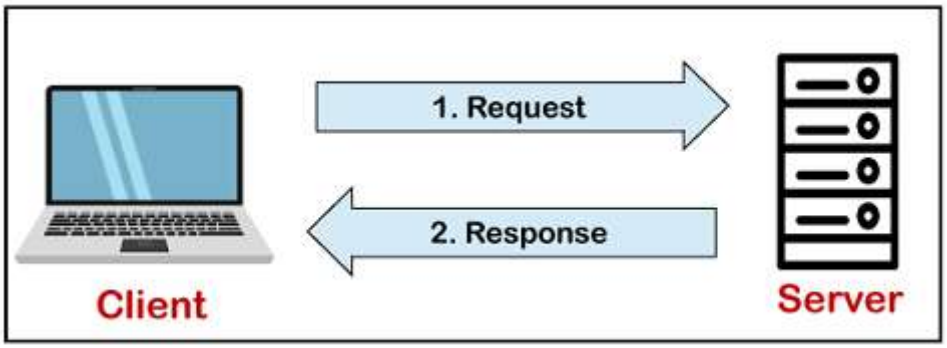
Operating Systems like Windows, Linux, MacOS, etc. with C, C++, and Java languages.

MySQL is a Relational Database Management System (RDBMS) software that provides many things, which are as follows:

- It allows us to implement database operations on tables, rows, columns, and indexes.
- It defines the database relationship in the form of tables (collection of rows and columns), also known as relations.
- It provides the Referential Integrity between rows or columns of various tables.
- It allows us to updates the table indexes automatically.
- It uses many SQL queries and combines useful information from multiple tables for the end-users.

**How MySQL Works?**

MySQL follows the working of Client-Server Architecture. This model is designed for the end-users called clients to access the resources from a central computer known as a server using network services. Here, the clients make requests through a graphical user interface (GUI), and the server will give the desired output as soon as the instructions are matched. The process of MySQL environment is the same as the client-server model.




The core of the MySQL database is the MySQL Server. This server is available as a separate program and responsible for handling all the

database instructions, statements, or commands. The working of MySQL database with MySQL Server are as follows:

1. MySQL creates a database that allows you to build many tables to store and manipulate data and defining the relationship between each table.
2. Clients make requests through the GUI screen or command prompt by using specific SQL expressions on MySQL.
3. Finally, the server application will respond with the requested expressions and produce the desired result on the client-side.

A client can use any MySQL GUI. But, it is making sure that your GUI should be lighter and user-friendly to make your data management activities faster and easier. Some of the most widely used MySQL GUIs are MySQL Workbench, SequelPro, DBVisualizer, and the Navicat DB Admin Tool. Some GUIs are commercial, while some are free with limited functionality, and some are only compatible with MacOS. Thus, you can choose the GUI according to your needs.

**Reasons for popularity**

MySQL is becoming so popular because of these following reasons:

- MySQL is an open-source database, so you don't have to pay a single penny to use it.
- MySQL is a very powerful program that can handle a large set of functionality of the most expensive and powerful database packages.
- MySQL is customizable because it is an open-source database, and the open-source GPL license facilitates programmers to modify the SQL software according to their own specific environment.

## 1. Data Types in MySQL

MySQL supports a wide range of data types for handling various kinds of data. These data types can be broadly categorized as:

- **Numeric Types**: Used for storing numeric values.
    - INT, SMALLINT, TINYINT, MEDIUMINT, BIGINT
    - FLOAT, DOUBLE, DECIMAL
- **Date and Time Types**: Used for storing dates and times.
    - DATE, DATETIME, TIMESTAMP, TIME, YEAR
- **String Types**: Used for storing string data.
    - CHAR, VARCHAR, BINARY, VARBINARY, BLOB, TEXT, ENUM, SET

**2. CAST or CONVERT**

These functions are used to change the data type of a value in MySQL.

- **CAST**: Converts a value from one data type to another.

  ```
  SELECT CAST('2023-07-09' AS DATETIME);
  ```

- **CONVERT**: Similar to CAST, but the syntax is different.

  ```
  SELECT CONVERT('2023-07-09', DATETIME);
  ```

  Both functions are functionally equivalent, but CAST is part of the SQL standard, making it more portable.

  Portability in the context of SQL and databases refers to the ability of SQL code or database schema to be easily transferred and used across different database management systems (DBMS) without requiring significant modification. This is important in environments where the same codebase might need to run on multiple types of databases, or where there might be a need to switch from one DBMS to another.

**Non-Portable SQL (Using CONVERT)**:

```
SELECT CONVERT(DATE, '2024-07-10', 101)
```

This statement uses the SQL Server-specific CONVERT function, which includes a style parameter. This is not ANSI-compliant and would likely need to be rewritten to run on a different DBMS.

**Common Style Numbers for Date and Time Formats**

Here are some other common style numbers used with the CONVERT function in SQL Server:

- **100**: mon dd yyyy hh:miAM (or PM) (e.g., Jul 10 2024 12:00AM)
- **101**: mm/dd/yyyy (e.g., 07/10/2024)
- **103**: dd/mm/yyyy (e.g., 10/07/2024)
- **104**: dd.mm.yyyy (e.g., 10.07.2024)
- **105**: dd-mm-yyyy (e.g., 10-07-2024)
- **106**: dd mon yyyy (e.g., 10 Jul 2024)
- **110**: mm-dd-yyyy (e.g., 07-10-2024)
- **111**: yyyy/mm/dd (e.g., 2024/07/10)
- **112**: yyyymmdd (e.g., 20240710)

**3. Keys in SQL**

Keys are constraints that define which columns in a table can be used to uniquely identify rows in the table. There are several types of keys:

- **Primary Key**: Uniquely identifies each row in a table. A table can have only one primary key, which can consist of one or multiple columns (composite key).

```
CREATE TABLE students (
    student_id INT PRIMARY KEY,
    name VARCHAR(100)
);
```

- **Foreign Key**: A key used to link two tables together. This is a column (or columns) in one table that refers to the primary key in another table.

```
CREATE TABLE enrollments (
    enrollment_id INT PRIMARY KEY,
    student_id INT,
    course_id INT,
    FOREIGN KEY (student_id) REFERENCES students(student_id)
);
```

- **Unique Key**: Ensures that all values in a column are unique.

```
CREATE TABLE users (
    user_id INT PRIMARY KEY,
    email VARCHAR(100) UNIQUE
);
```

- **Index Key**: Used to speed up searches and queries.

```
CREATE TABLE products (
    product_id INT PRIMARY KEY,
    name VARCHAR(100),
    INDEX (name)
);
```

**4. Constraints**

Constraints are rules enforced on data columns to ensure data integrity. Here are some common constraints:

- **NOT NULL**: Ensures that a column cannot have a NULL value.

```
CREATE TABLE employees (
    employee_id INT PRIMARY KEY,
    name VARCHAR(100) NOT NULL
);
```

- **UNIQUE**: Ensures that all values in a column are different.

```
CREATE TABLE employees (
    employee_id INT PRIMARY KEY,
    email VARCHAR(100) UNIQUE
);
```

- **PRIMARY KEY**: A combination of NOT NULL and UNIQUE. Uniquely identifies each row in a table.

```
CREATE TABLE employees (
    employee_id INT PRIMARY KEY,
    name VARCHAR(100) NOT NULL
);
```

- **FOREIGN KEY**: Ensures the referential integrity of the data in one table to match values in another table.

```
CREATE TABLE orders (
    order_id INT PRIMARY KEY,
    customer_id INT,
    FOREIGN KEY (customer_id) REFERENCES
customers(customer_id)
);
```

- **CHECK**: Ensures that all values in a column satisfy a specific condition.

```
CREATE TABLE employees (
    employee_id INT PRIMARY KEY,
    age INT,
    CHECK (age >= 18)
);
```

- **DEFAULT**: Sets a default value for a column if no value is specified.

```
CREATE TABLE employees (
    employee_id INT PRIMARY KEY,
    status VARCHAR(20) DEFAULT 'active'
);
```

These components form the basis of creating and managing robust, efficient, and reliable databases in MySQL.

- MySQL is quicker than other databases, so it can work well even with the large data set.
- MySQL supports many operating systems with many languages like PHP, PERL, C, C++, JAVA, etc.
- MySQL uses a standard form of the well-known SQL data language.
- MySQL is very friendly with PHP, the most popular language for web development.
- MySQL supports large databases, up to 50 million rows or more in a table. The default file size limit for a table is 4GB, but you can increase this (if your operating system can handle it) to a theoretical limit of 8 million terabytes (TB).