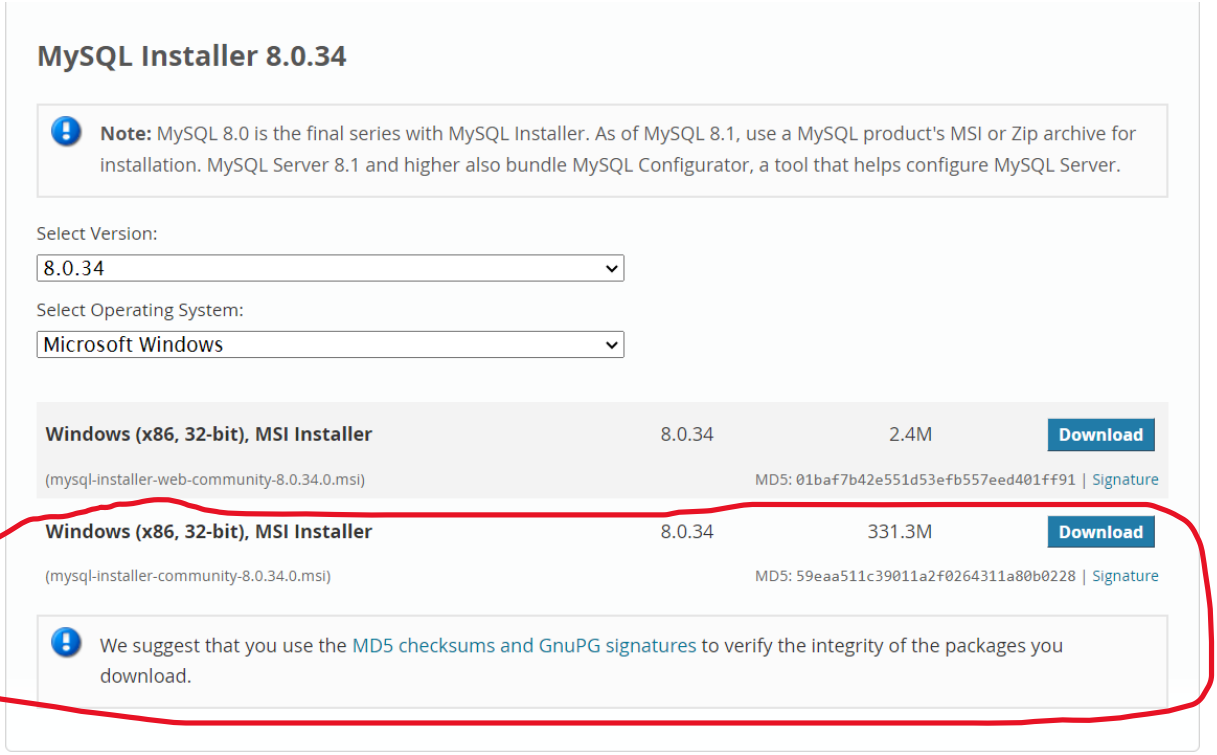
How to install MySQL Database

Navigate the official URL of the MySQL as follows:

https://dev.mysql.com/downloads/installer/



MySQL client

MySQL client is a common name for tools that are designed to connect to MySQL Server. Client programs are used to send commands or queries to the server and allow managing data in the databases stored on the server.

MySQL command-line client

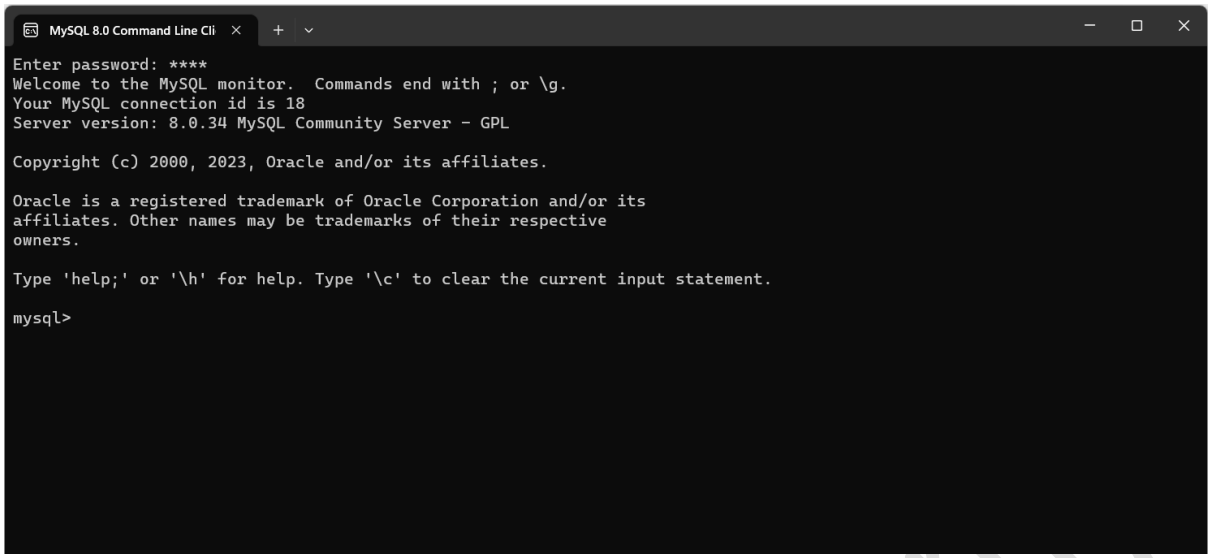
It is the tool that allows sending commands to MySQL Server from the command line.

MySQL Server and MySQL Client:

MySQL Server holds data, and MySQL Client is used for accessing and manipulating it.

MySQL Command Line Client:

In order to access MySQL Command Line Client, Go to Windows → Search “MySQL Command Line Client” → Type Password then It displays the below window as follows:



Manage MySQL database from the command line:

Step 1:

Create a user from the command line:

CREATE USER 'username' IDENTIFIED BY 'password';

replace the username and password placeholders with a username and password of your choice.

Step 2:

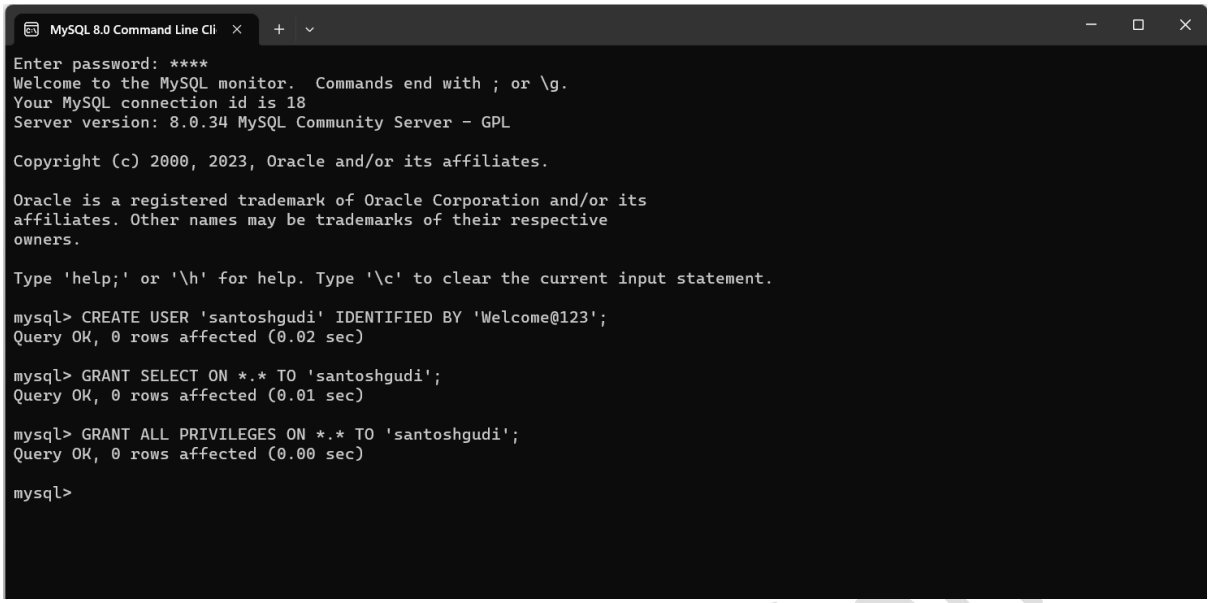
Once After Creating the User, We need to grant SELECT privileges to this user. For this, run the MySQL query:

GRANT SELECT ON *.* TO 'username';

Step 3:

In case we want to grant a user all permissions on all databases, run the following command:

GRANT ALL PRIVILEGES ON *.* TO 'username';



How to display all available users?

mysql> SELECT user FROM mysql.user;

```
mysql> SELECT user FROM mysql.user;
+------------------+
| user             |
+------------------+
| santoshgudi      |
| mysql.infoschema |
| mysql.session    |
| mysql.sys        |
| root             |
+------------------+
5 rows in set (0.02 sec)
```

How to display the Current User?

mysql> SELECT user();

```
mysql> SELECT user();
+----------------+
| user()         |
+----------------+
| root@localhost |
+----------------+
1 row in set (0.00 sec)
```

How to switch from one user to another user on MySQL?

mysql> system mysql -u santoshgudi -p;

Enter password: **********

mysql> SELECT user();

```
mysql> SELECT user();
+-----------------------+
| user()                |
+-----------------------+
| santoshgudi@localhost |
+-----------------------+
1 row in set (0.00 sec)
```

Create a database from the command line:

Step 1:

To create a database, use the following command. Replace the placeholder with the required name of the database.

CREATE DATABASE databasename;

Step 2:

To start working with the newly created database, execute the query:

USE databasename;

Execution:

mysql> CREATE DATABASE devdb;

Query OK, 1 row affected (0.01 sec)

mysql> USE devdb;

Database changed

mysql>

How to delete a MySQL database from the command line?

To delete a database, run the following simple command. Remember that you won't be able to revert the deletion, so perform the operation with caution.

DROP DATABASE dbname;

How to delete a MySQL user account?

To delete a user in MySQL, execute a query:

DROP USER 'username';

Introduction to MySQL:

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.

Features of MySQL:

- MySQL is a relational database management system
- MySQL is open-source
- MySQL is free
- MySQL is ideal for both small and large applications
- MySQL is very fast, reliable, scalable, and easy to use

- MySQL is cross-platform
- MySQL is compliant with the ANSI SQL standard
- MySQL was first released in 1995
- MySQL is developed, distributed, and supported by Oracle Corporation
- MySQL is named after co-founder Monty Widenius's daughter: My

MySQL Built In Functions:

MySQL has many built-in functions. those are classified as

1. String Functions

2. Numeric Functions

3. Date Functions

4. Advanced Functions

1. String Functions:

LOWER/LCASE: It converts the given string in to lower-case.

```
mysql> SELECT lower('PROGRAM');
+------------------+
| lower('PROGRAM') |
+------------------+
| program          |
+------------------+
1 row in set (0.01 sec)
```

UPPER/UCASE: It converts the given string in to upper-case.

```
mysql> SELECT upper('welcome');
+------------------+
| upper('welcome') |
+------------------+
| WELCOME          |
+------------------+
1 row in set (0.00 sec)
```

LENGTH/CHAR_LENGTH/CHARACTER_LENGTH:
It provides number of characters available in the given string.

```
mysql> SELECT length('welcome');
+------------------+
| length('welcome') |
+------------------+
|                7 |
+------------------+
1 row in set (0.00 sec)
```

CONCAT: It combines two or more Strings.

```
mysql> SELECT concat('Sindhanur','Raichur');
+-------------------------------+
| concat('Sindhanur','Raichur') |
+-------------------------------+
| SindhanurRaichur              |
+-------------------------------+
1 row in set (0.00 sec)
```

CANCAT_WS: It adds two or more strings together with a separator

```
mysql> SELECT CONCAT_WS(',','Sindhanur','Raichur');
+--------------------------------------+
| CONCAT_WS(',','Sindhanur','Raichur') |
+--------------------------------------+
| Sindhanur,Raichur                    |
+--------------------------------------+
1 row in set (0.00 sec)
```

LEFT: It extracts number of characters from left side of a given string.

```
mysql> SELECT LEFT('Programming',7);
+-----------------------+
| LEFT('Programming',7) |
+-----------------------+
| Program               |
+-----------------------+
1 row in set (0.00 sec)
```

RIGHT: It extracts number of characters from right side of a given string.

```
mysql> SELECT right('Programming',8);
+-------------------------+
| right('Programming',8) |
+-------------------------+
| gramming                |
+-------------------------+
1 row in set (0.00 sec)
```

LPAD: It left-pads a string with another string, to a certain length

```
mysql> SELECT lpad('GenTech',10,'#');
+------------------------+
| lpad('GenTech',10,'#') |
+------------------------+
| ###GenTech             |
+------------------------+
1 row in set (0.00 sec)
```

RPAD: It right-pads a string with another string, to a certain length

```
mysql> SELECT rpad('GenTech',10,'#');
+------------------------+
| rpad('GenTech',10,'#') |
+------------------------+
| GenTech###             |
+------------------------+
1 row in set (0.00 sec)
```

LTRIM: It removes blank spaces from left side of a given string.

```
mysql> SELECT ltrim('   Welcome');
+--------------------+
| ltrim('   Welcome') |
+--------------------+
| Welcome            |
+--------------------+
1 row in set (0.00 sec)

mysql> SELECT length(ltrim('   Welcome'));
+----------------------------+
| length(ltrim('   Welcome')) |
+----------------------------+
|                          7 |
+----------------------------+
1 row in set (0.00 sec)
```

RTRIM: It removes blank spaces from right side of a given string.

```
mysql> SELECT rtrim('GenTech   ');
+--------------------+
| rtrim('GenTech   ') |
+--------------------+
| GenTech            |
+--------------------+
1 row in set (0.00 sec)

mysql> SELECT length(rtrim('GenTech   '));
+----------------------------+
| length(rtrim('GenTech   ')) |
+----------------------------+
|                          7 |
+----------------------------+
1 row in set (0.00 sec)
```

TRIM: It removes blank spaces from both the sides of a given string.

```
mysql> SELECT trim('    GenTech    ');
+-------------------------+
| trim('    GenTech    ') |
+-------------------------+
| GenTech                 |
+-------------------------+
1 row in set (0.00 sec)

mysql> SELECT length(trim('    GenTech    '));
+---------------------------------+
| length(trim('    GenTech    ')) |
+---------------------------------+
|                               7 |
+---------------------------------+
1 row in set (0.00 sec)
```

REVERSE: It reverses the given string.

```
mysql> SELECT reverse('Welcome');
+--------------------+
| reverse('Welcome') |
+--------------------+
| emocleW            |
+--------------------+
1 row in set (0.00 sec)
```

REPLACE: It replaces all occurrences of a substring within a string, with a new substring.

```
mysql> SELECT replace('It is a palace','is','was');
+---------------------------------------+
| replace('It is a palace','is','was') |
+---------------------------------------+
| It was a palace                       |
+---------------------------------------+
1 row in set (0.00 sec)
```

REPEAT: It repeats a string as many times as specified

```
mysql> SELECT repeat('GenTech',5);
+--------------------------------------+
| repeat('GenTech',5)                  |
+--------------------------------------+
| GenTechGenTechGenTechGenTechGenTech |
+--------------------------------------+
1 row in set (0.00 sec)
```

FIELD: It returns the index position of a value in a list of values.

```
mysql> SELECT field('GenTech','Orange','White','GenTech');
+---------------------------------------------+
| field('GenTech','Orange','White','GenTech') |
+---------------------------------------------+
|                                           3 |
+---------------------------------------------+
1 row in set (0.00 sec)
```

FIND_IN_SET: It returns the position of a string within a list of strings.

```
mysql> SELECT find_in_set('GenTech','Orange,Green,White,GenTech');
+-----------------------------------------------------+
| find_in_set('GenTech','Orange,Green,White,GenTech') |
+-----------------------------------------------------+
|                                                   4 |
+-----------------------------------------------------+
1 row in set (0.00 sec)
```

INSTR: It returns the position of the first occurrence of a string in another string.

```
mysql> SELECT instr('GenTech','Tech');
+-------------------------+
| instr('GenTech','Tech') |
+-------------------------+
|                       4 |
+-------------------------+
1 row in set (0.00 sec)
```

LOCATE: It returns the position of the first occurrence of a substring in a string

```
mysql> SELECT locate('Academy','GenTech Academy');
+-------------------------------------+
| locate('Academy','GenTech Academy') |
+-------------------------------------+
|                                   9 |
+-------------------------------------+
1 row in set (0.00 sec)
```

MID: in the below example, It extract a substring from a string (start at position 4, extract 4 characters):

```
mysql> SELECT mid('Programming',4,4);
+-----------------------+
| mid('Programming',4,4) |
+-----------------------+
| gram                  |
+-----------------------+
1 row in set (0.00 sec)
```

POSITION: It returns the position of the first occurrence of a substring in a string.

```
mysql> SELECT position('Tech' IN 'GenTech Academy');
+---------------------------------------+
| position('Tech' IN 'GenTech Academy') |
+---------------------------------------+
|                                     4 |
+---------------------------------------+
1 row in set (0.00 sec)
```

STRCMP: It compares two strings, if they are equal it provides 0 otherwise 1 or -1

```
mysql> SELECT STRCMP('GenTech','GenTech');
+-----------------------------+
| STRCMP('GenTech','GenTech') |
+-----------------------------+
|                           0 |
+-----------------------------+
1 row in set (0.00 sec)

mysql> SELECT strcmp('GenTech','Gen');
+-------------------------+
| strcmp('GenTech','Gen') |
+-------------------------+
|                       1 |
+-------------------------+
1 row in set (0.00 sec)
```

SUBSTR / SUBSTRING : It extracts a substring from a string (starting at any position)

```
mysql> SELECT substr('GenTech Academy',9,7);
+-------------------------------+
| substr('GenTech Academy',9,7) |
+-------------------------------+
| Academy                       |
+-------------------------------+
1 row in set (0.00 sec)

mysql> SELECT substring('GenTech Academy',9,7);
+---------------------------------+
| substring('GenTech Academy',9,7) |
+---------------------------------+
| Academy                         |
+---------------------------------+
1 row in set (0.00 sec)
```

2. Numeric Functions

ABS: It returns the absolute value of a number.

```
mysql> SELECT abs(-75);
+----------+
| abs(-75) |
+----------+
|       75 |
+----------+
1 row in set (0.01 sec)
```

POWER / POW: It returns the value of a number raised to the power of another number.

```
mysql> SELECT power(2,5);
+------------+
| power(2,5) |
+------------+
|         32 |
+------------+
1 row in set (0.00 sec)

mysql> SELECT pow(2,5);
+----------+
| pow(2,5) |
+----------+
|       32 |
+----------+
1 row in set (0.00 sec)
```

MOD: It returns the remainder of a division.

```
mysql> SELECT mod(9,2);
+----------+
| mod(9,2) |
+----------+
|        1 |
+----------+
1 row in set (0.00 sec)
```

TRUNCATE: It truncates a number to the specified number of decimal places.

```
mysql> SELECT truncate(12.56789,2);
+----------------------+
| truncate(12.56789,2) |
+----------------------+
|                12.56 |
+----------------------+
1 row in set (0.01 sec)
```

SQRT: It returns the square root of a number.

```
mysql> SELECT sqrt(25);
+-----------+
| sqrt(25) |
+-----------+
|         5 |
+-----------+
1 row in set (0.00 sec)
```

ROUND: It rounds a number to a specified number of decimal places.

```
mysql> SELECT round(12.56789,2);
+--------------------+
| round(12.56789,2) |
+--------------------+
|              12.57 |
+--------------------+
1 row in set (0.00 sec)
```

RAND: It returns a random number.

```
mysql> SELECT rand();
+----------------------+
| rand()               |
+----------------------+
| 0.10785897831233124 |
+----------------------+
1 row in set (0.00 sec)
```

GREATEST: It returns the greatest value from the given list.

```
mysql> SELECT greatest(4,77,10,34,25,50);
+----------------------------+
| greatest(4,77,10,34,25,50) |
+----------------------------+
|                         77 |
+----------------------------+
1 row in set (0.01 sec)
```

LEAST: It returns the smallest value from the given list.

```
mysql> SELECT least(4,77,10,34,25,50);
+-------------------------+
| least(4,77,10,34,25,50) |
+-------------------------+
|                       4 |
+-------------------------+
1 row in set (0.00 sec)
```

CEIL / CEILING: It returns the smallest integer value that is >= to a number.

```
mysql> SELECT ceil(12.15);
+-------------+
| ceil(12.15) |
+-------------+
|          13 |
+-------------+
1 row in set (0.00 sec)

mysql> SELECT ceiling(12.01);
+----------------+
| ceiling(12.01) |
+----------------+
|             13 |
+----------------+
1 row in set (0.00 sec)
```

FLOOR: It returns the largest integer value that is <= to a number.

```
mysql> SELECT floor(12.99);
+--------------+
| floor(12.99) |
+--------------+
|           12 |
+--------------+
1 row in set (0.00 sec)
```

3. Date Functions:

CURDATE / CURRENT_DATE: It returns the current system date.

```
mysql> SELECT curdate();
+------------+
| curdate()  |
+------------+
| 2023-07-30 |
+------------+
1 row in set (0.00 sec)

mysql> SELECT current_date();
+----------------+
| current_date() |
+----------------+
| 2023-07-30     |
+----------------+
1 row in set (0.00 sec)
```

CURTIME / CURRENT_TIME: It returns the current system time.

```
mysql> SELECT curtime();
+-----------+
| curtime() |
+-----------+
| 21:20:42  |
+-----------+
1 row in set (0.00 sec)

mysql> SELECT current_time();
+----------------+
| current_time() |
+----------------+
| 21:20:52       |
+----------------+
1 row in set (0.00 sec)
```

CURRENT_TIMESTAMP: It returns the current system date and time.

```
mysql> SELECT current_timestamp();
+---------------------+
| current_timestamp() |
+---------------------+
| 2023-07-30 21:23:44 |
+---------------------+
1 row in set (0.00 sec)
```

DAY: It returns the day of the month for a given date.

```
mysql> SELECT day('2023-06-27');
+-------------------+
| day('2023-06-27') |
+-------------------+
|                27 |
+-------------------+
1 row in set (0.00 sec)
```

MONTH: It returns the month part for a given date.

```
mysql> SELECT month('2023-06-27');
+----------------------+
| month('2023-06-27') |
+----------------------+
|                    6 |
+----------------------+
1 row in set (0.00 sec)
```

YEAR: It returns the year part for a given date.

```
mysql> SELECT year('2023-06-27');
+--------------------+
| year('2023-06-27') |
+--------------------+
|               2023 |
+--------------------+
1 row in set (0.00 sec)
```

HOUR: It return the hour part of a given datetime.

```
mysql> SELECT hour('2023-06-27 10:24:55');
+-----------------------------+
| hour('2023-06-27 10:24:55') |
+-----------------------------+
|                          10 |
+-----------------------------+
1 row in set (0.00 sec)
```

MINUTE: It returns the minute part of a given datetime.

```
mysql> SELECT minute('2023-06-27 10:24:55');
+-------------------------------+
| minute('2023-06-27 10:24:55') |
+-------------------------------+
|                            24 |
+-------------------------------+
1 row in set (0.00 sec)
```

SECOND: It returns the second part of a given datetime.

```
mysql> SELECT second('2023-06-27 10:24:55');
+-------------------------------+
| second('2023-06-27 10:24:55') |
+-------------------------------+
|                            55 |
+-------------------------------+
1 row in set (0.00 sec)
```

NOW: It returns the current date and time.

```
mysql> SELECT now();
+---------------------+
| now()               |
+---------------------+
| 2023-07-30 23:00:15 |
+---------------------+
1 row in set (0.00 sec)
```

SYSDATE: It returns the current date and time.

```
mysql> SELECT sysdate();
+---------------------+
| sysdate()           |
+---------------------+
| 2023-07-30 23:02:38 |
+---------------------+
1 row in set (0.00 sec)
```

WEEK: It returns the week number for a given date.

```
mysql> SELECT week('2023-02-26');
+--------------------+
| week('2023-02-26') |
+--------------------+
|                  9 |
+--------------------+
1 row in set (0.00 sec)
```

WEEKDAY: It returns the weekday number for a given date.

```
mysql> SELECT weekday('2023-02-26');
+-----------------------+
| weekday('2023-02-26') |
+-----------------------+
|                     6 |
+-----------------------+
1 row in set (0.00 sec)
```

WEEKOFYEAR: It returns the week number for a given date.

```
mysql> SELECT weekofyear('2023-02-26');
+--------------------------+
| weekofyear('2023-02-26') |
+--------------------------+
|                        8 |
+--------------------------+
1 row in set (0.00 sec)
```

YEARWEEK: It returns the year and week number for a given date.

```
mysql> SELECT yearweek('2023-02-26');
+------------------------+
| yearweek('2023-02-26') |
+------------------------+
|                 202309 |
+------------------------+
1 row in set (0.00 sec)
```

TIMESTAMP: It returns a datetime value based on a date or datetime value.

```
mysql> SELECT timestamp('2023-02-26');
+--------------------------+
| timestamp('2023-02-26') |
+--------------------------+
| 2023-02-26 00:00:00     |
+--------------------------+
1 row in set (0.00 sec)

mysql> SELECT timestamp('2023-02-26','06:25:45');
+-------------------------------------+
| timestamp('2023-02-26','06:25:45') |
+-------------------------------------+
| 2023-02-26 06:25:45                 |
+-------------------------------------+
1 row in set (0.00 sec)
```

TIMEDIFF: It returns the difference between two time expressions.

```
mysql> SELECT timediff('20:30:45','10:25:40');
+---------------------------------+
| timediff('20:30:45','10:25:40') |
+---------------------------------+
| 10:05:05                        |
+---------------------------------+
1 row in set (0.00 sec)
```

TIME_TO_SEC: It converts a time value into seconds.

```
mysql> SELECT time_to_sec('02:10:25');
+-------------------------+
| time_to_sec('02:10:25') |
+-------------------------+
|                    7825 |
+-------------------------+
1 row in set (0.00 sec)
```

TIME_FORMAT: This function formats a time by a specified format.

```
mysql> SELECT time_format('19:35:45','%h %i %s');
+-----------------------------------+
| time_format('19:35:45','%h %i %s') |
+-----------------------------------+
| 07 35 45                          |
+-----------------------------------+
1 row in set (0.00 sec)

mysql> SELECT time_format('19:35:45','%h %i %s %p');
+--------------------------------------+
| time_format('19:35:45','%h %i %s %p') |
+--------------------------------------+
| 07 35 45 PM                          |
+--------------------------------------+
1 row in set (0.00 sec)
```

TIME: It extracts the time part from a given time expression.

```
mysql> SELECT time('2023-02-26 17:35:45');
+----------------------------+
| time('2023-02-26 17:35:45') |
+----------------------------+
| 17:35:45                   |
+----------------------------+
1 row in set (0.00 sec)
```

SUBDATE: It subtracts a time/date interval from a date and then returns the date.

```
mysql> SELECT subdate('2023-02-26', interval 2 MONTH);
+-----------------------------------------+
| subdate('2023-02-26', interval 2 MONTH) |
+-----------------------------------------+
| 2022-12-26                              |
+-----------------------------------------+
1 row in set (0.00 sec)
```

STR_TO_DATE: It returns a date based on a given string and a given format.

```
mysql> SELECT str_to_date('June 27 2023','%M %d %Y');
+---------------------------------------+
| str_to_date('June 27 2023','%M %d %Y') |
+---------------------------------------+
| 2023-06-27                            |
+---------------------------------------+
1 row in set (0.01 sec)
```

DAYNAME: It returns the weekday name for a given date.

```
mysql> SELECT dayname('2023-05-14');
+-----------------------+
| dayname('2023-05-14') |
+-----------------------+
| Sunday                |
+-----------------------+
1 row in set (0.00 sec)
```

DATEDIFF: It returns the number of days between two given date values.

```
mysql> SELECT datediff('2023-07-30','2023-07-20');
+-------------------------------------+
| datediff('2023-07-30','2023-07-20') |
+-------------------------------------+
|                                  10 |
+-------------------------------------+
1 row in set (0.00 sec)
```

DATE_ADD: It adds a time/date interval to a date and then returns the date.

```
mysql> SELECT date_add('2023-06-15',interval 5 month);
+-----------------------------------------+
| date_add('2023-06-15',interval 5 month) |
+-----------------------------------------+
| 2023-11-15                              |
+-----------------------------------------+
1 row in set (0.00 sec)
```

DATE_SUB: It subtracts a time/date interval from a date and then returns the date.

```
mysql> SELECT date_sub('2023-08-15',interval 5 day);
+----------------------------------------+
| date_sub('2023-08-15',interval 5 day) |
+----------------------------------------+
| 2023-08-10                             |
+----------------------------------------+
1 row in set (0.00 sec)
```

LAST_DAY: It provides the last day of the month for the given date.

```
mysql> SELECT last_day('2023-08-15');
+-------------------------+
| last_day('2023-08-15') |
+-------------------------+
| 2023-08-31              |
+-------------------------+
1 row in set (0.00 sec)
```

4. Database Functions

DATABASE: It returns the name of the current database.

```
mysql> SELECT database();
+------------+
| database() |
+------------+
| NULL       |
+------------+
1 row in set (0.00 sec)

mysql> CREATE DATABASE SAmpleDB;
Query OK, 1 row affected (0.06 sec)

mysql> SELECT database();
+------------+
| database() |
+------------+
| NULL       |
+------------+
1 row in set (0.00 sec)

mysql> USE SampleDB;
Database changed
mysql> SELECT database();
+------------+
| database() |
+------------+
| sampledb   |
+------------+
1 row in set (0.00 sec)
```

CURRENT_USER: It returns the user name and host name for the MySQL account.

```
mysql> SELECT current_user();
+----------------+
| current_user() |
+----------------+
| root@localhost |
+----------------+
1 row in set (0.00 sec)
```

CONNECTION_ID: It returns the unique connection ID for the current connection.

```
mysql> SELECT connection_id();
+-----------------+
| connection_id() |
+-----------------+
|             199 |
+-----------------+
1 row in set (0.00 sec)
```

SESSION_USER: It returns the current MySQL user name and host name.

```
mysql> SELECT session_user();
+----------------+
| session_user() |
+----------------+
| root@localhost |
+----------------+
1 row in set (0.00 sec)
```

SYSTEM_USER: It returns the current MySQL user name and host name.

```
mysql> SELECT system_user();
+----------------+
| system_user()  |
+----------------+
| root@localhost |
+----------------+
1 row in set (0.00 sec)
```

USER: It returns the current MySQL user name and host name.

```
mysql> SELECT user();
+----------------+
| user()         |
+----------------+
| root@localhost |
+----------------+
1 row in set (0.00 sec)
```

VERSION: It returns the current version of the MySQL database.

```
mysql> SELECT version();
+-----------+
| version() |
+-----------+
| 8.0.34    |
+-----------+
1 row in set (0.00 sec)
```

Datatypes in MySQL:

A Data Type specifies a particular type of data, like integer, floating points, Boolean, etc. The data type of a column defines what value the column can hold: integer, character, money, date and time, binary, and so on. Each column in a database table is required to have a name and a data type.

In MySQL there are three main data types:

1. String Data Types

2. Numeric Data Types

3. Date and time Data Types

String Data Types:

Data Type	Description
CHAR(size)	A Fixed length string (It can contain letters, numbers, and special characters). from 0 to 255. Default is 1
VARCHAR(size)	A Variable length string (can contain letters, numbers, and special characters). from 0 to 65535
BINARY(size)	It is equal to CHAR(), but stores binary byte strings. Default is 1
VARBINARY(size)	It is equal to VARCHAR(), but stores binary byte strings.
TINYBLOB	For BLOBs (Binary Large OBjects). Max length: 255 bytes
TINYTEXT	It holds a string with a maximum length of 255 characters
TEXT(size)	It holds a string with a maximum length of 65,535 bytes
BLOB(size)	For BLOBs (Binary Large OBjects). It holds up to 65,535 bytes of data
MEDIUMTEXT	It holds a string with a maximum length of 16,777,215 characters.
MEDIUMBLOB	For BLOBs (Binary Large OBjects). It holds up to 16,777,215 bytes of data
LONGTEXT	It holds a string with a maximum length of 4,294,967,295 characters
LONGBLOB	For BLOBs (Binary Large OBjects). It holds up to 4,294,967,295 bytes of data
ENUM(val1, val2, val3, ...)	We can list up to 65535 values in an ENUM list. If a value is inserted that is not in the list, a blank value will be inserted. The values are sorted in the order you enter them
SET(val1, val2, val3, ...)	We can list up to 0 to 64 values in a SET list

Numeric Data Types:

Data Type	Description
BIT(size)	It can hold a value from 1 to 64. The default value for size is 1.
TINYINT(size)	A very small integer. Signed range is from -128 to 127. Unsigned range is from 0 to 255.
SMALLINT(size)	A small integer. Signed range is from -32768 to 32767. Unsigned range is from 0 to 65535.
MEDIUMINT(size)	A medium integer. Signed range is from -8388608 to 8388607. Unsigned range is from 0 to 16777215.

BIGINT(size)	A large integer. Signed range is from -9223372036854775808 to 9223372036854775807. Unsigned range is from 0 to 18446744073709551615.
BOOL	0 is considered as false, 1 is considered as true.
BOOLEAN	0 is considered as false, 1 is considered as true.
INT(size)	A medium integer. Signed range is from -2147483648 to 2147483647. Unsigned range is from 0 to 4294967295.
INTEGER(size)	It works as similar to INT(size)
DOUBLE(size, d)	A normal-size floating point number. The total number of digits is specified in size. The number of digits after the decimal point is specified in the d parameter. Example: DOUBLE(5,2)
DECIMAL(size, d)	An exact fixed-point number. The total number of digits is specified in size. The number of digits after the decimal point is specified in the d parameter. The maximum number for size is 65. The maximum number for d is 30. The default value for size is 10. The default value for d is 0. Example: DOUBLE(5,2)
DEC(size, d)	It works as similar to DECIMAL(size,d)

Date and time Data Types:

Data Type	Description
DATE	It supports format as YYYY-MM-DD. The supported range is from '1000-01-01' to '9999-12-31'
DATETIME(fsp)	A date and time combination. Format: YYYY-MM-DD hh:mm:ss. The supported range is from '1000-01-01 00:00:00' to '9999-12-31 23:59:59'.
TIMESTAMP(fsp)	A timestamp. TIMESTAMP values are stored as the number of seconds since the Unix epoch ('1970-01-01 00:00:00' UTC). Format: YYYY-MM-DD hh:mm:ss. The supported range is from '1970-01-01 00:00:01' UTC to '2038-01-09 03:14:07' UTC.
TIME(fsp)	A time. Format: hh:mm:ss. The supported range is from '-838:59:59' to '838:59:59'
YEAR	A year in four-digit format. Values allowed in four-digit format: 1901 to 2155, and 0000.