Practical 02 Reading Material

Summary of Introduction to MySQL

- MySQL is an open source relational database management system (RDBMS) based on Structured Query Language (SQL).
- MySQL Community Edition (GPL)
  - **MySQL Community Server** (GPL)
- Terminology
  - Database, Table, Column, Row, Primary key, Foreign Key, Composite key, Candidate Key, Index, Redundancy
- Install MySQL on Ubuntu 16.04.2
  - **sudo apt-get update**
  - **sudo apt-get install mysql-server**

SQL Data Types

1. Integer Types (Exact Value) - INTEGER, INT, SMALLINT, TINYINT, MEDIUMINT, BIGINT

Type	**Storage (Bytes)**	**Minimum Value Signed**	**Minimum Value Unsigned**	**Maximum Value Signed**	**Maximum Value Unsigned**
TINYINT	1	**-128**	**0**	**127**	**255**
SMALLINT	2	**-32768**	**0**	**32767**	**65535**
MEDIUMINT	3	**-8388608**	**0**	**8388607**	**16777215**
INT	4	**-2147483648**	**0**	**2147483647**	**4294967295**
BIGINT	8	**-2^{63}**	**0**	**$2^{63}-1$**	**$2^{64}-1$**

2. Fixed-Point Types (Exact Value) - DECIMAL, NUMERIC

The DECIMAL and NUMERIC types store exact numeric data values. These types are used when it is important to preserve exact precision, for example with monetary data. In MySQL, NUMERIC is implemented as DECIMAL, so the following remarks about DECIMAL apply equally to NUMERIC.

```
salary DECIMAL(5,2)
```

Floating-Point Types (Approximate Value) - FLOAT, DOUBLE

FLOAT(***M***,***D***)

DOUBLE(***M***,***D***)

3. Date and Time Data Types

The date and time data types for representing temporal values are DATE, TIME, DATETIME, TIMESTAMP, and YEAR.

```
CREATE TABLE t1 (t TIME(3), dt DATETIME(6), ts TIMESTAMP(0));
```

- DATE
  A date. The supported range is '1000-01-01' to '9999-12-31'. MySQL displays DATE values in '***YYYY-MM-DD***' format, but permits assignment of values to DATE columns using either strings or numbers.
- DATETIME[(***fsp***)]
  A date and time combination. The supported range is '1000-01-01 00:00:00.000000' to '9999-12-31 23:59:59.999999'. MySQL displays DATETIME values in '***YYYY-MM-DD hh:mm:ss***[.***fraction***]' format, but permits assignment of values to DATETIME columns using either strings or numbers.
  An optional ***fsp*** value in the range from 0 to 6 may be given to specify fractional seconds precision. A value of 0 signifies that there is no fractional part. If omitted, the default precision is 0.
  Automatic initialization and updating to the current date and time for DATETIME columns can be specified using DEFAULT and ON UPDATE column definition clauses, as described in Section 11.2.5, "Automatic Initialization and Updating for TIMESTAMP and DATETIME".
- TIMESTAMP[(***fsp***)]
  A timestamp. The range is '1970-01-01 00:00:01.000000' UTC to '2038-01-19 03:14:07.999999' UTC. TIMESTAMP values are stored as the number of seconds since the epoch ('1970-01-01 00:00:00' UTC). A TIMESTAMP cannot represent the value '1970-01-01 00:00:00' because that is equivalent to 0 seconds from the epoch and the value 0 is reserved for representing '0000-00-00 00:00:00', the "zero" TIMESTAMP value.
  An optional ***fsp*** value in the range from 0 to 6 may be given to specify fractional seconds precision. A value of 0 signifies that there is no fractional part. If omitted, the default precision is 0.
  The way the server handles TIMESTAMP definitions depends on the value of the explicit_defaults_for_timestamp system variable (see Section 5.1.8, "Server System Variables").
  If explicit_defaults_for_timestamp is enabled, there is no automatic assignment of the DEFAULT CURRENT_TIMESTAMP or ON UPDATE CURRENT_TIMESTAMP attributes to any TIMESTAMP column. They must be included explicitly in the column definition. Also, any TIMESTAMP not explicitly declared as NOT NULL permits NULL values.

If explicit_defaults_for_timestamp is disabled, the server handles TIMESTAMP as follows:
Unless specified otherwise, the first TIMESTAMP column in a table is defined to be automatically set to the date and time of the most recent modification if not explicitly assigned a value. This makes TIMESTAMP useful for recording the timestamp of an INSERT or UPDATE operation. You can also set any TIMESTAMP column to the current date and time by assigning it a NULL value, unless it has been defined with the NULL attribute to permit NULL values.
Automatic initialization and updating to the current date and time can be specified using DEFAULT CURRENT_TIMESTAMP and ON UPDATE CURRENT_TIMESTAMP column definition clauses. By default, the first TIMESTAMP column has these properties, as previously noted. However, any TIMESTAMP column in a table can be defined to have these properties.

- TIME[(***fsp***)]
  A time. The range is '-838:59:59.000000' to '838:59:59.000000'. MySQL displays TIME values in '***hh:mm:ss***[.***fraction***]' format, but permits assignment of values to TIME columns using either strings or numbers.
  An optional ***fsp*** value in the range from 0 to 6 may be given to specify fractional seconds precision. A value of 0 signifies that there is no fractional part. If omitted, the default precision is 0.
- YEAR[(4)]
  A year in 4-digit format. MySQL displays YEAR values in ***YYYY*** format, but permits assignment of values to YEAR columns using either strings or numbers. Values display as 1901 to 2155, or 0000.
  For additional information about YEAR display format and interpretation of input values, see Section 11.2.4, “The YEAR Type”.

4. String Data Types

The string data types are CHAR, VARCHAR, BINARY, VARBINARY, BLOB, TEXT, ENUM, and SET.

- **CHAR(M)** − A fixed-length string between 1 and 255 characters in length (for example CHAR(5)), right-padded with spaces to the specified length when stored. Defining a length is not required, but the default is 1.
- **VARCHAR(M)** − A variable-length string between 1 and 255 characters in length. For example, VARCHAR(25). You must define a length when creating a VARCHAR field.
- **BLOB or TEXT** − A field with a maximum length of 65535 characters. BLOBs are "Binary Large Objects" and are used to store large amounts of binary data, such as images or other types of files. Fields defined as TEXT also hold large amounts of data. The difference between the two is that the sorts and comparisons on the stored data are **case sensitive** on BLOBs and are **not case sensitive** in TEXT fields. You do not specify a length with BLOB or TEXT.
- **TINYBLOB or TINYTEXT** − A BLOB or TEXT column with a maximum length of 255 characters. You do not specify a length with TINYBLOB or TINYTEXT.

- **MEDIUMBLOB or MEDIUMTEXT** − A BLOB or TEXT column with a maximum length of 16777215 characters. You do not specify a length with MEDIUMBLOB or MEDIUMTEXT.
- **LONGBLOB or LONGTEXT** − A BLOB or TEXT column with a maximum length of 4294967295 characters. You do not specify a length with LONGBLOB or LONGTEXT.
- **ENUM** − An enumeration, which is a fancy term for list. When defining an ENUM, you are creating a list of items from which the value must be selected (or it can be NULL). For example, if you wanted your field to contain "A" or "B" or "C", you would define your ENUM as ENUM ('A', 'B', 'C') and only those values (or NULL) could ever populate that field.

- Connect to MySQL on Ubuntu
  - **sudo mysql –u root –p**
- Check Version
  - **SELECT VERSION();**
- List Databases
  - **SHOW DATABASES;**
- Create a Database
  - **CREATE DATABASE db_name;**
- Select a Database
  - **USE db_name;**
- Create New Tables
  - **CREATE TABLE table_name**

```
(
        Column-definitions,
        primaryKey-definition
);
```

- View structure of a table
  - **DESC table_name;**
- List Tables
  - **SHOW TABLES;**

- View Data in a table

  SELECT * FROM table_name;

  SELECT column1, column2

  FROM table_name;

MySQL Connection

You can establish the MySQL database using the **mysql** binary at the command prompt.

Example

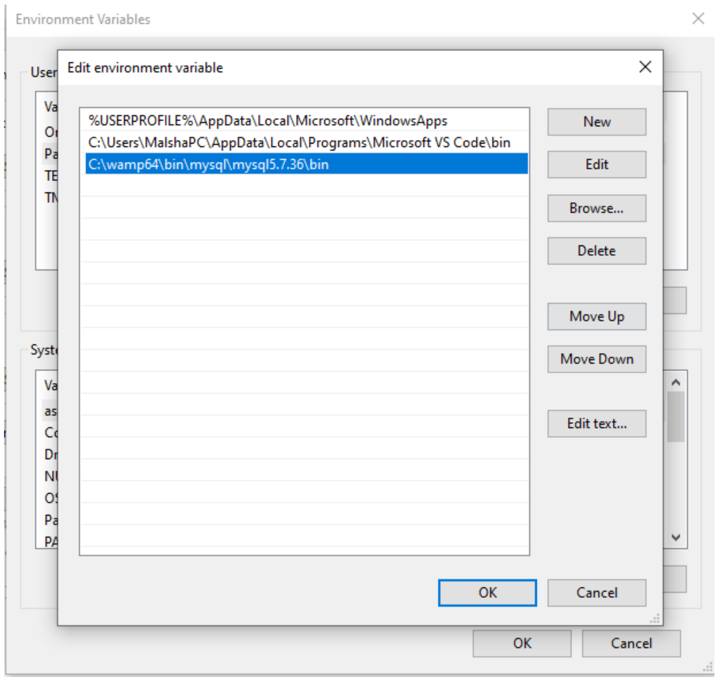
Here is a simple example to connect to the MySQL server from the command prompt −

```
[root@host]# mysql -u root -p
Enter password:******
```

This will give you the mysql> command prompt where you will be able to execute any SQL command. Following is the result of above command

The various methods to use MySQL on CMD on Windows.

1. Install and Run WAMP Server.
2. Add the environment path variable as below.

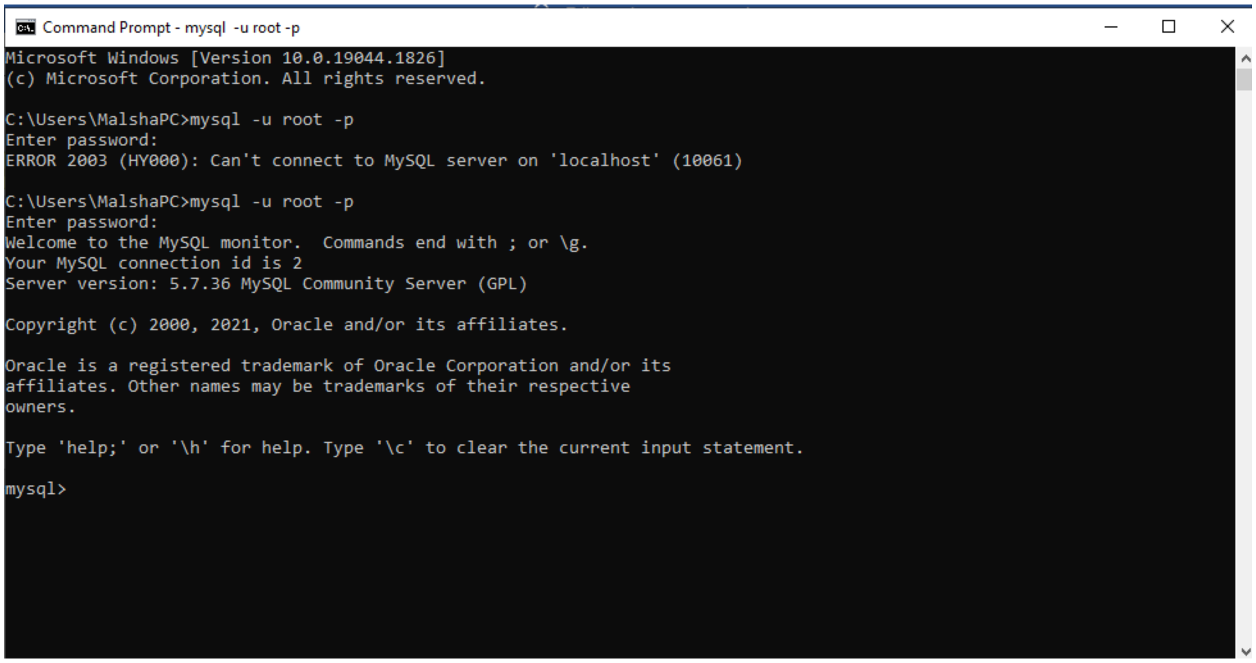


```
%USERPROFILE%\AppData\Local\Microsoft\WindowsApps
C:\Users\MalshaPC\AppData\Local\Programs\Microsoft VS Code\bin
C:\wamp64\bin\mysql\mysql5.7.36\bin
```

3. Go to command prompt and give following command.

   mysql -u root -p

4. MySQL will be connected and can be used as below.

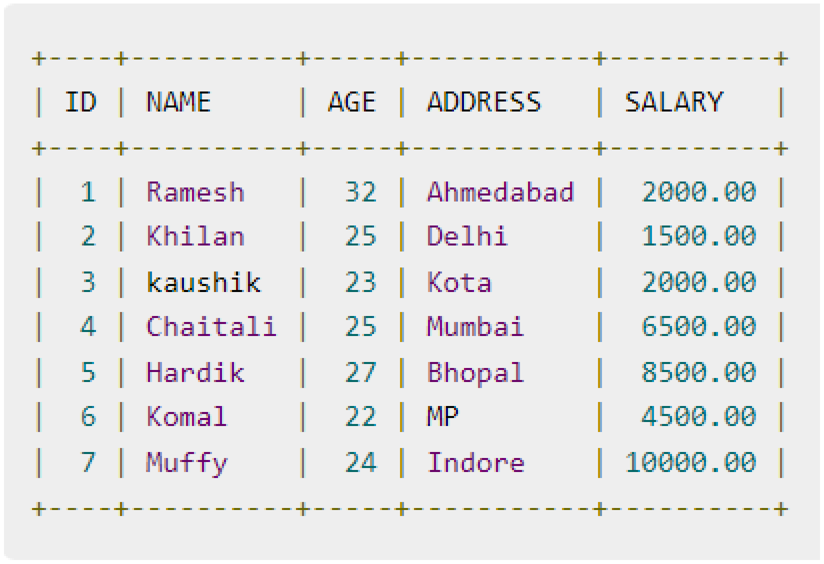


```
Microsoft Windows [Version 10.0.19044.1826]
(c) Microsoft Corporation. All rights reserved.

C:\Users\MalshaPC>mysql -u root -p
Enter password:
ERROR 2003 (HY000): Can't connect to MySQL server on 'localhost' (10061)

C:\Users\MalshaPC>mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2
Server version: 5.7.36 MySQL Community Server (GPL)

Copyright (c) 2000, 2021, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

RDBMS Concepts

1. Table

The data in an RDBMS is stored in database objects which are called as **tables**. This table is basically a collection of related data entries and it consists of numerous columns and rows.



```
+----+----------+-----+-----------+----------+
| ID | NAME     | AGE | ADDRESS   | SALARY   |
+----+----------+-----+-----------+----------+
|  1 | Ramesh   |  32 | Ahmedabad |  2000.00 |
|  2 | Khilan   |  25 | Delhi     |  1500.00 |
|  3 | kaushik  |  23 | Kota      |  2000.00 |
|  4 | Chaitali |  25 | Mumbai    |  6500.00 |
|  5 | Hardik   |  27 | Bhopal    |  8500.00 |
|  6 | Komal    |  22 | MP        |  4500.00 |
|  7 | Muffy    |  24 | Indore    | 10000.00 |
+----+----------+-----+-----------+----------+
```

2. Field

Every table is broken up into smaller entities called fields. The fields in the CUSTOMERS table consist of ID, NAME, AGE, ADDRESS and SALARY.

3. **Record or a Row**

A record is also called as a row of data is each individual entry that exists in a table. For example, there are 7 records in the above CUSTOMERS table.

A record is a horizontal entity in a table.

```
+----+----------+-----+-----------+----------+
|  1 | Ramesh   |  32 | Ahmedabad |  2000.00 |
+----+----------+-----+-----------+----------+
```

4. Column

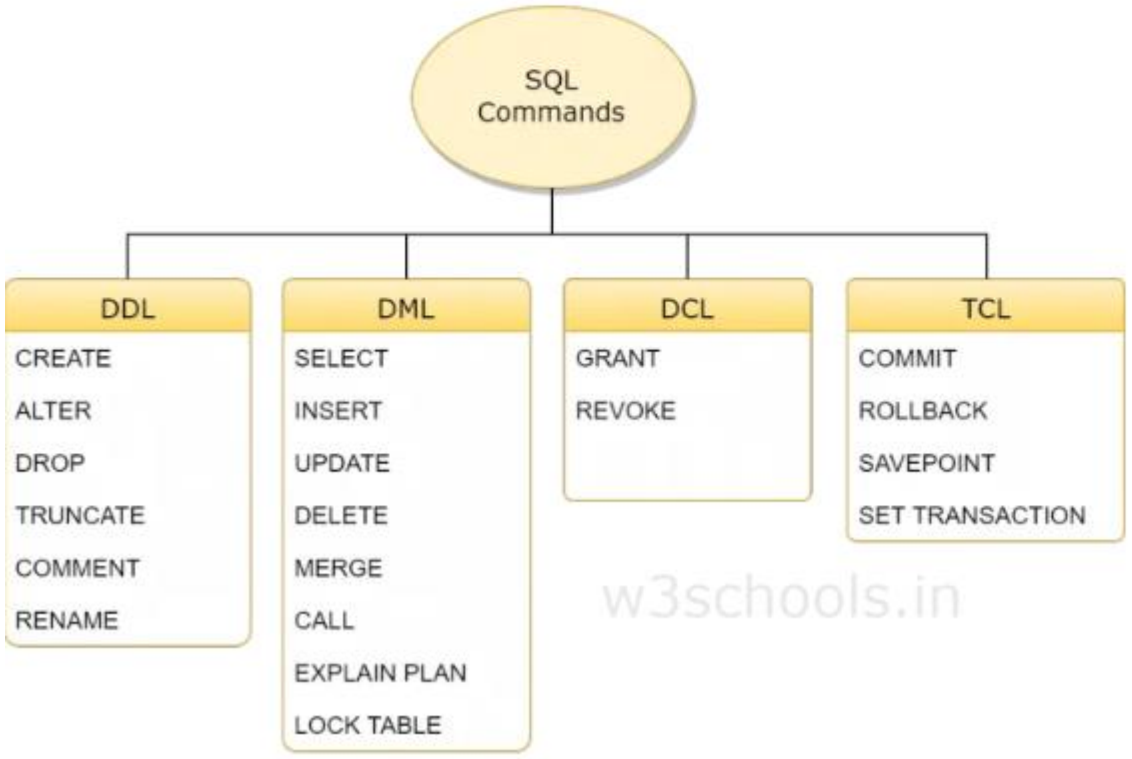
A column is a vertical entity in a table that contains all information associated with a specific field in a table.

```
+-----------+
| ADDRESS   |
+-----------+
| Ahmedabad |
| Delhi     |
| Kota      |
| Mumbai    |
| Bhopal    |
| MP        |
| Indore    |
+----+------+
```

5. NULL value

A NULL value in a table is a value in a field that appears to be blank, which means a field with a NULL value is a field with no value.

SQL Commands



1. DDL

DDL is short name of **Data Definition Language,** which deals with database schemas and descriptions, of how the data should reside in the database.

- CREATE - to create a database and its objects like (table, index, views, store procedure, function, and triggers)
- ALTER - alters the structure of the existing database
- DROP - delete objects from the database
- TRUNCATE - remove all records from a table, including all spaces allocated for the records are removed
- COMMENT - add comments to the data dictionary
- RENAME - rename an object

2. DML

DML is short name of **Data Manipulation Language** which deals with data manipulation and includes most common SQL statements such SELECT, INSERT, UPDATE, DELETE, etc., and it is used to store, modify, retrieve, delete and update data in a database.

- SELECT - retrieve data from a database
- INSERT - insert data into a table
- UPDATE - updates existing data within a table
- DELETE - Delete all records from a database table

- MERGE - UPSERT operation (insert or update)
- CALL - call a PL/SQL or Java subprogram
- EXPLAIN PLAN - interpretation of the data access path
- LOCK TABLE - concurrency Control

3. DCL

DCL is short name of **Data Control Language** which includes commands such as GRANT and mostly concerned with rights, permissions and other controls of the database system.

- GRANT - allow users access privileges to the database
- REVOKE - withdraw users access privileges given by using the GRANT command

4. TCL

TCL is short name of Transaction Control Language which deals with a transaction within a database.

- COMMIT - commits a Transaction
- ROLLBACK - rollback a transaction in case of any error occurs
- SAVEPOINT - to rollback the transaction making points within groups
- SET TRANSACTION - specify characteristics of the transaction

RDBMS Packages

MySQL

MySQL is an open source SQL database, which is developed by a Swedish company – MySQL AB. MySQL is pronounced as "my ess-que-ell," in contrast with SQL, pronounced "sequel."

MS SQL Server

MS SQL Server is a Relational Database Management System developed by Microsoft Inc. Its primary query languages are −

- T-SQL
- ANSI SQL

ORACLE

It is a very large multi-user based database management system. Oracle is a relational database management system developed by 'Oracle Corporation'.

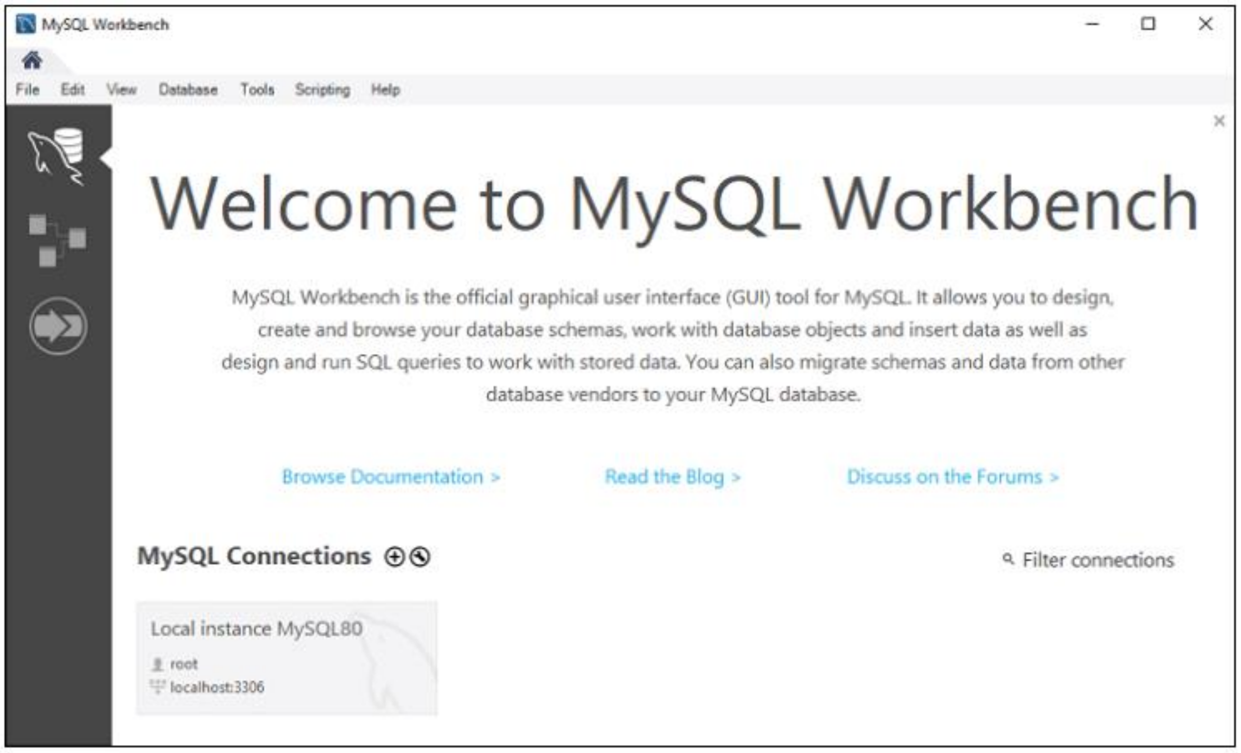
Oracle works to efficiently manage its resources, a database of information among the multiple clients requesting and sending data in the network.

MS ACCESS

This is one of the most popular Microsoft products. Microsoft Access is an entry-level database management software. MS Access database is not only inexpensive but also a powerful database for small-scale projects.

MySQL Workbench

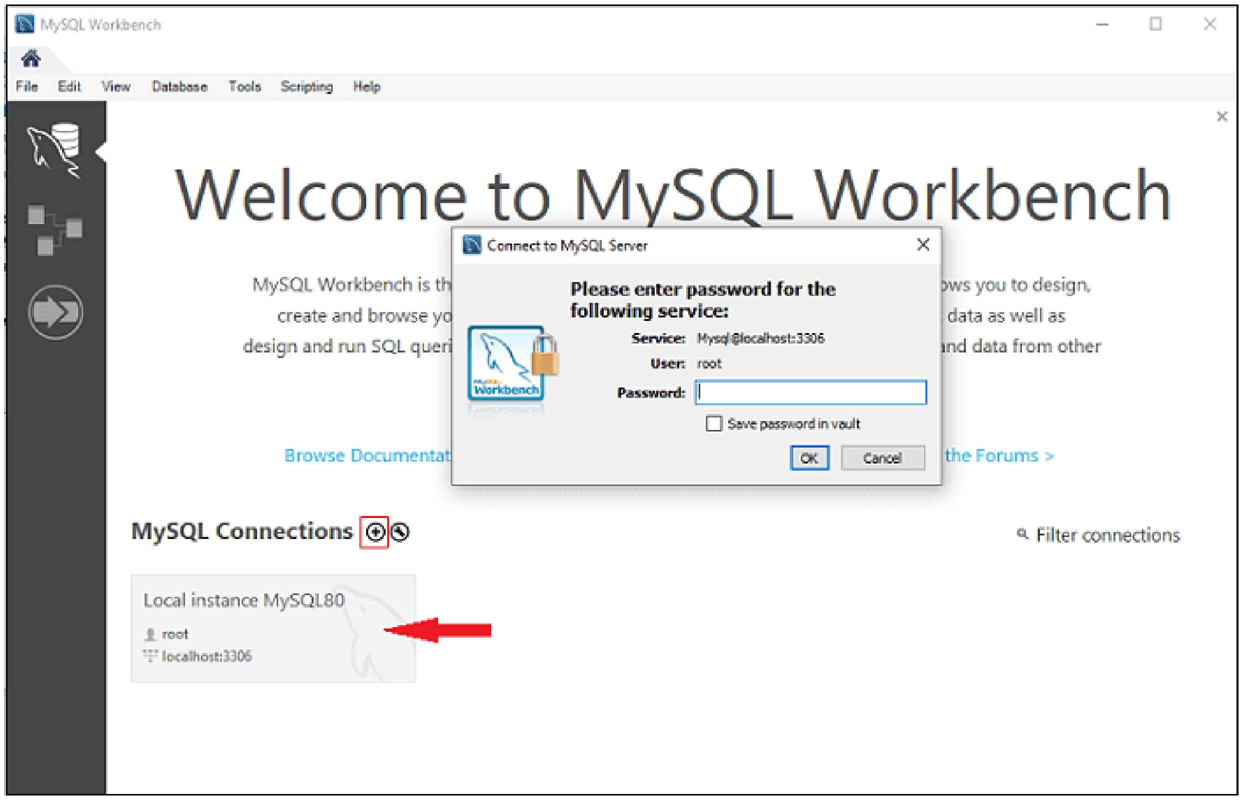
Download and Install MySQL Workbench.



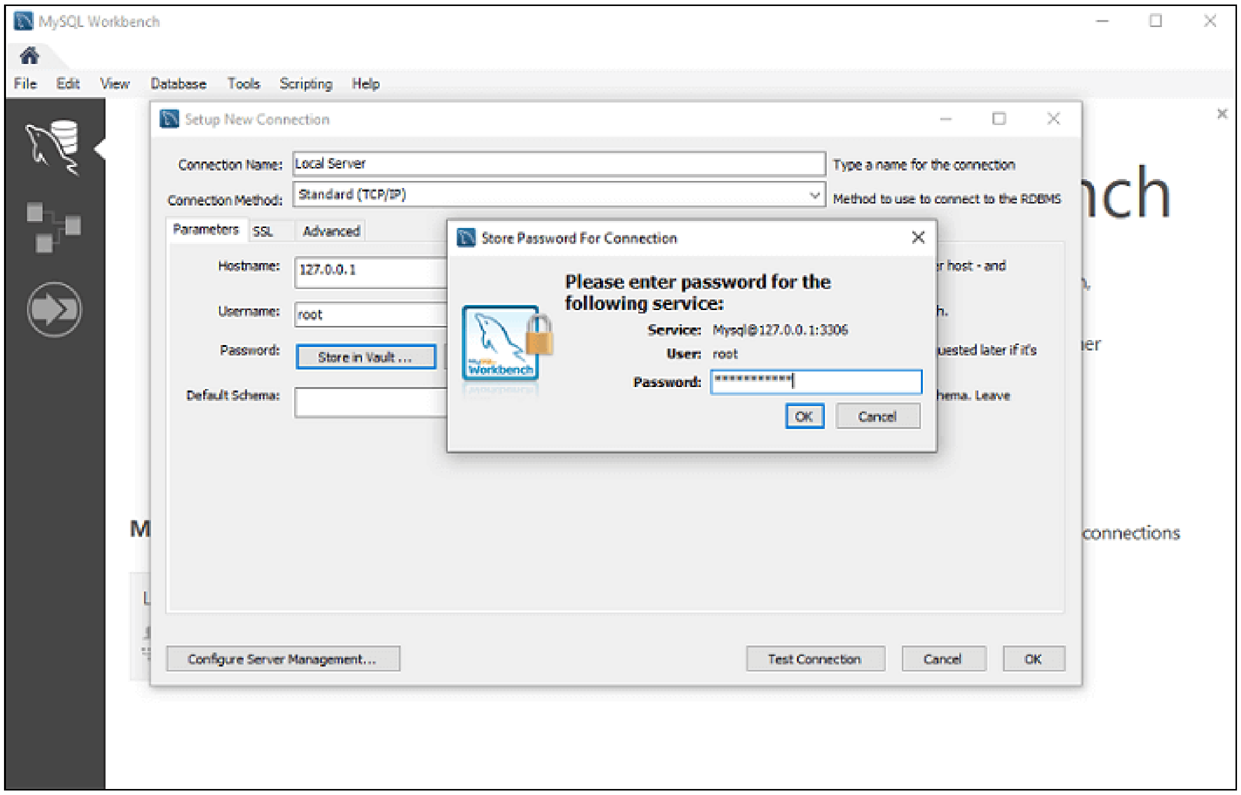
Welcome to MySQL Workbench

MySQL Workbench is the official graphical user interface (GUI) tool for MySQL. It allows you to design, create and browse your database schemas, work with database objects and insert data as well as design and run SQL queries to work with stored data. You can also migrate schemas and data from other database vendors to your MySQL database.

1. In the above screen, you need to make a connection. To do this, double click the box designated by the **red arrow**. Here, you will get the popup screen that asks to enter the password created earlier during the installation. After entering the password, you are able to connect with the Server.


MySQL Workbench
File Edit View Database Tools Scripting Help
Welcome to MySQL Workbench
MySQL Workbench is th ... ows you to design,
create and browse yo ... data as well as
design and run SQL queri ... nd data from other
Browse Documentat
the Forums >
Connect to MySQL Server
Please enter password for the following service:
Service: Mysql@localhost:3306
User: root
Password:
Save password in vault
OK
Cancel
MySQL Connections
Filter connections
Local instance MySQL80
root
localhost:3306

2. If you do not have a connection, you can create a new connection. To make a connection, click the plus (+) icon or go to the menu bar -> Database -> Connect to Database, the following screen appears. Now, you need to fill all the details. Here, you have to make sure that the entered password should be the same as you have created earlier.

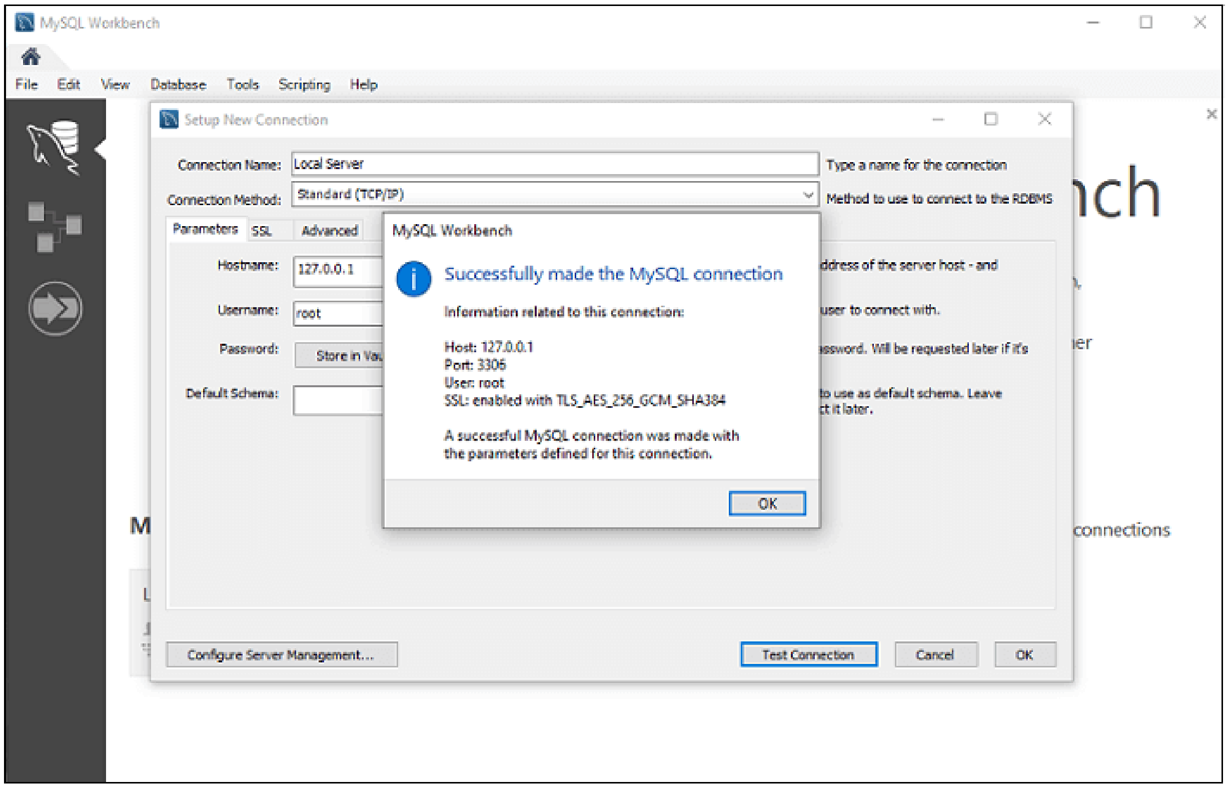

MySQL Workbench
File Edit View Database Tools Scripting Help
Setup New Connection
Store Password For Connection

Connection Name: Local Server — Type a name for the connection
Connection Method: Standard (TCP/IP) — Method to use to connect to the RDBMS
Parameters | SSL | Advanced
Hostname: 127.0.0.1
Username: root
Password: Store in Vault ...
Default Schema:

Please enter password for the following service:
Service: Mysql@127.0.0.1:3306
User: root
Password: ************

OK
Cancel
Configure Server Management...
Test Connection
Cancel
OK

3. After entering all the details, click on the **Test Connection** to test the database connectivity. If the connection is successful, you will get the following screen. Now, click on OK->OK button to finish the setup.



Connection Name: Local Server — Type a name for the connection
Connection Method: Standard (TCP/IP) — Method to use to connect to the RDBMS
Hostname: 127.0.0.1
Username: root
Password:
Default Schema:

MySQL Workbench

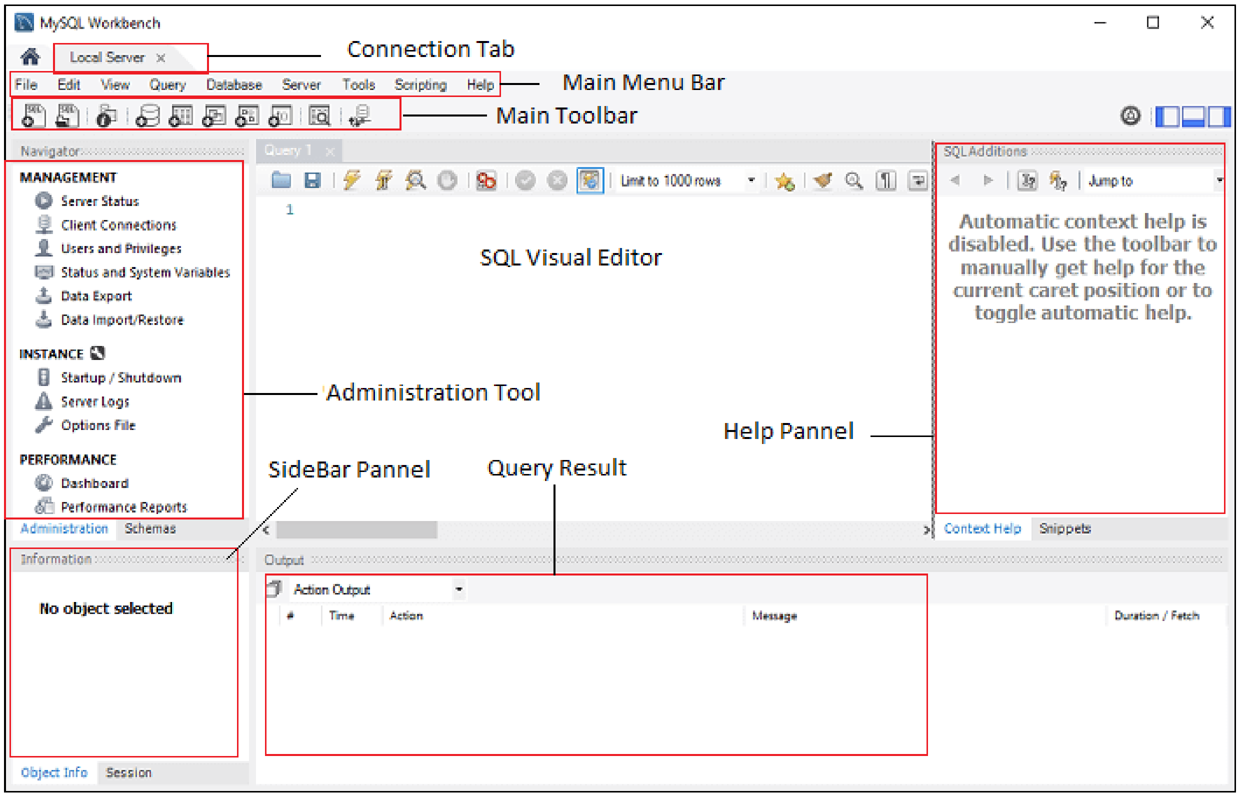
Successfully made the MySQL connection

Information related to this connection:

Host: 127.0.0.1
Port: 3306
User: root
SSL: enabled with TLS_AES_256_GCM_SHA384

A successful MySQL connection was made with the parameters defined for this connection.

4. Once you have finished all the setup, it will open the MySQL Workbench screen. Now, double click on the newly created connection, you will get the following screen where the SQL command can be executed.



Adding/Deleting users

- Use “**mysql**” database
- List tables inside “**mysql**” db
- Check the table structure of “**user**” table
- Add a new **USER**

CREATE USER [IF NOT EXISTS] account_name IDENTIFIED **BY** 'password';

In the above syntax, the **account_name** has two parts one is the **username**, and another is the **hostname**, which is separated by **@** symbol.

username@hostname

CREATE USER 'user_name'@'host' **IDENTIFIED BY** ‘password';

- **Example 01**

CREATE USER ‘ict_admin'@‘localhost' **IDENTIFIED BY** ‘adminpwd';

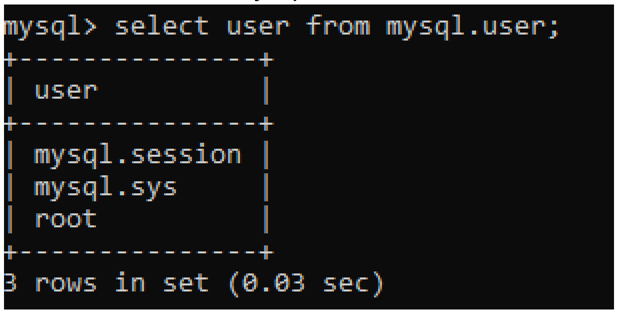
- **Example 02**

CREATE USER ‘ict_guest'@‘localhost’;

1. Go to CMD and open MySQL using CMD.

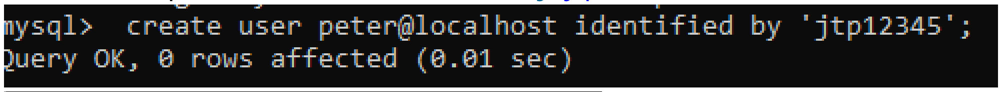
mysql -u root -p

2. Execute the following command to show all users in the current MySQL server.
   select user **from** mysql.user;

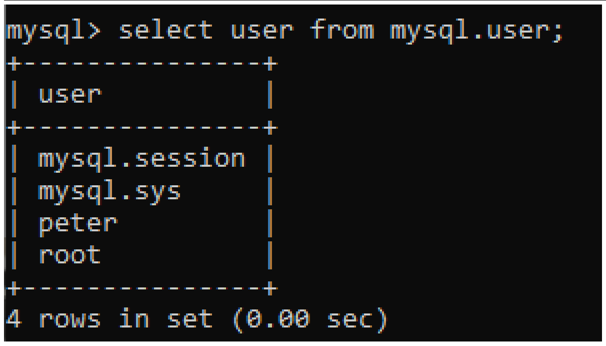


```
mysql> select user from mysql.user;
+---------------+
| user          |
+---------------+
| mysql.session |
| mysql.sys     |
| root          |
+---------------+
3 rows in set (0.03 sec)
```

3. Create a new user with the following command.
   create user peter@localhost identified **by** 'jtp12345';



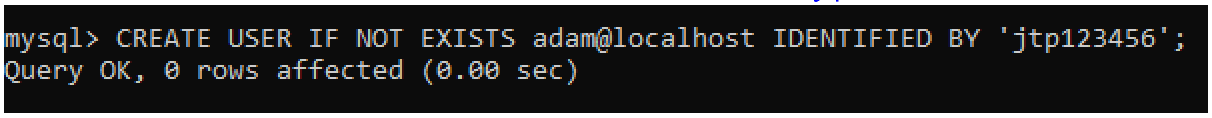
```
mysql>  create user peter@localhost identified by 'jtp12345';
Query OK, 0 rows affected (0.01 sec)
```



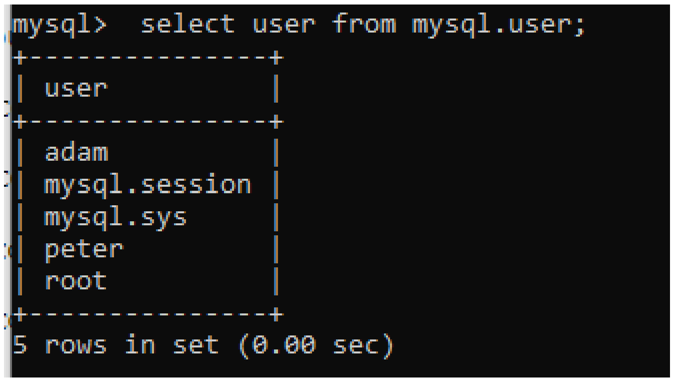
```
mysql> select user from mysql.user;
+---------------+
| user          |
+---------------+
| mysql.session |
| mysql.sys     |
| peter         |
| root          |
+---------------+
4 rows in set (0.00 sec)
```

4. Now, we will use the IF NOT EXISTS clause with the CREATE USER statement.

   CREATE USER IF NOT EXISTS adam@localhost IDENTIFIED **BY** 'jtp123456';



```
mysql> CREATE USER IF NOT EXISTS adam@localhost IDENTIFIED BY 'jtp123456';
Query OK, 0 rows affected (0.00 sec)
```



```
mysql>  select user from mysql.user;
+---------------+
| user          |
+---------------+
| adam          |
| mysql.session |
| mysql.sys     |
| peter         |
| root          |
+---------------+
5 rows in set (0.00 sec)
```

Grant Privileges to the MySQL New User

GRANT ALL PRIVILEGES ON database.table
TO ‘user'@‘host'
WITH GRANT OPTION;

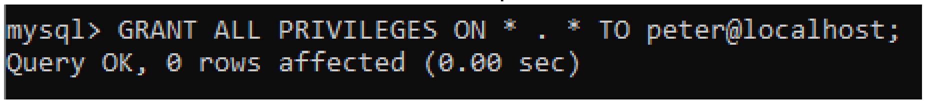
- **Example 01**
  GRANT ALL PRIVILEGES ON *.*
  TO ‘ict_admin '@‘localhost'
  WITH GRANT OPTION;

FLUSH PRIVILEGES;

Some of the most commonly used privileges are given below:

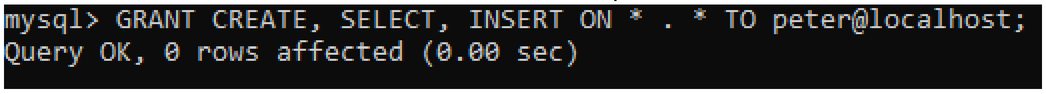
1. **ALL PRIVILEGES:** It permits all privileges to a new user account.
2. **CREATE:** It enables the user account to create databases and tables.
3. **DROP:** It enables the user account to drop databases and tables.
4. **DELETE:** It enables the user account to delete rows from a specific table.
5. **INSERT:** It enables the user account to insert rows into a specific table.
6. **SELECT:** It enables the user account to read a database.
7. **UPDATE:** It enables the user account to update table rows.

If you want to give all privileges to a newly created user, execute the following command.
GRANT ALL **PRIVILEGES ON** * . * **TO** peter@localhost;



```
mysql> GRANT ALL PRIVILEGES ON * . * TO peter@localhost;
Query OK, 0 rows affected (0.00 sec)
```

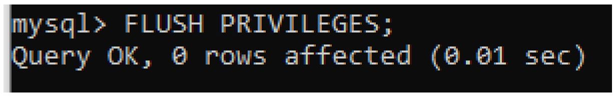
If you want to give specific privileges to a newly created user, execute the following command.
GRANT CREATE, **SELECT**, **INSERT ON** * . * **TO** peter@localhost;



```
mysql> GRANT CREATE, SELECT, INSERT ON * . * TO peter@localhost;
Query OK, 0 rows affected (0.00 sec)
```

Sometimes, you want to **flush** all the privileges of a user account for changes occurs immediately, type the following command.

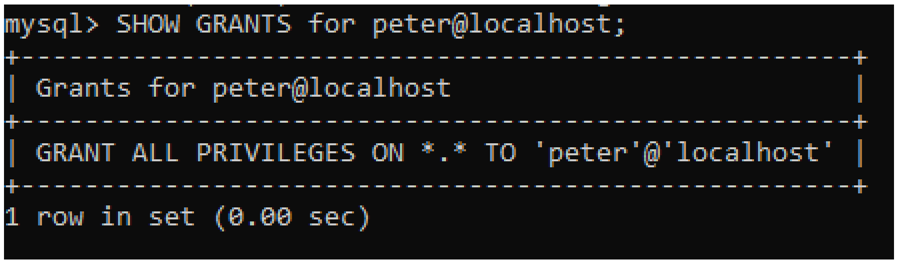
FLUSH **PRIVILEGES**;



```
mysql> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.01 sec)
```

If you want to see the existing privileges for the user, execute the following command.

SHOW GRANTS **for** username;



```
mysql> SHOW GRANTS for peter@localhost;
+---------------------------------------------------+
| Grants for peter@localhost                        |
+---------------------------------------------------+
| GRANT ALL PRIVILEGES ON *.* TO 'peter'@'localhost' |
+---------------------------------------------------+
1 row in set (0.00 sec)
```

Deleting users

DROP USER 'account_name';

Example

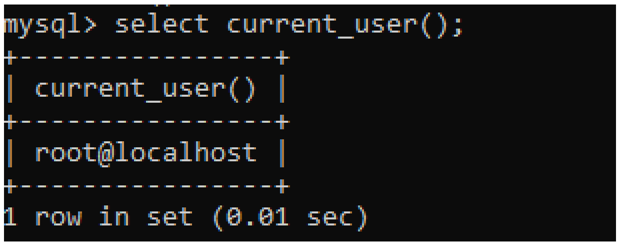
DROP USER ict_guest@localhost;

DROP USER john@localhost, peter@localhost;

Show Users

Select user **from** mysql.user;

Show current user



```
mysql> select current_user();
+----------------+
| current_user() |
+----------------+
| root@localhost |
+----------------+
1 row in set (0.01 sec)
```

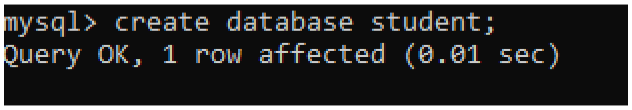
Create Database

MySQL implements a database as a directory that stores all files in the form of a table. It allows us to create a database mainly in **two ways**:

1. MySQL Command Line Client
2. MySQL Workbench

CREATE DATABASE database_name;

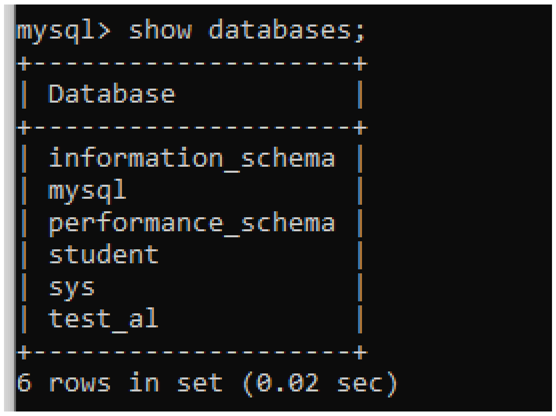
CREATE DATABASE student;



```
mysql> create database student;
Query OK, 1 row affected (0.01 sec)
```

We can check the created database using the following query:

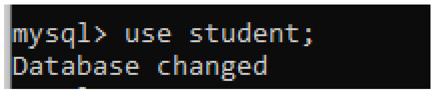
SHOW DATABASES;



```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| student            |
| sys                |
| test_al            |
+--------------------+
6 rows in set (0.02 sec)
```

Finally, we can use the below command to access the database that enables us to create a table and other database objects.

USE student;

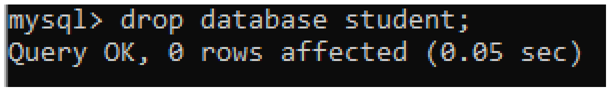


```
mysql> use student;
Database changed
```

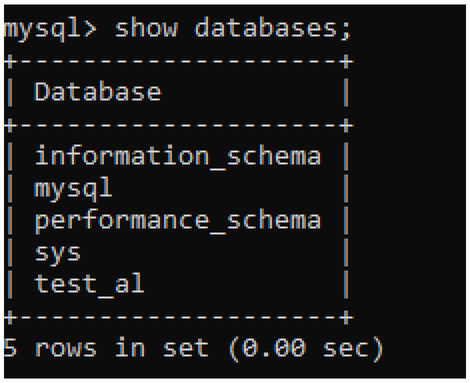
DROP Database

DROP DATABASE [IF EXISTS] database_name;

DROP DATABASE student;



```
mysql> drop database student;
Query OK, 0 rows affected (0.05 sec)
```



```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| test_al            |
+--------------------+
5 rows in set (0.00 sec)
```

Create table

CREATE TABLE table_name(

```
        column_definition1,
        column_definition2,
        ........,
        table_constraints
    );
```

CREATE TABLE student(

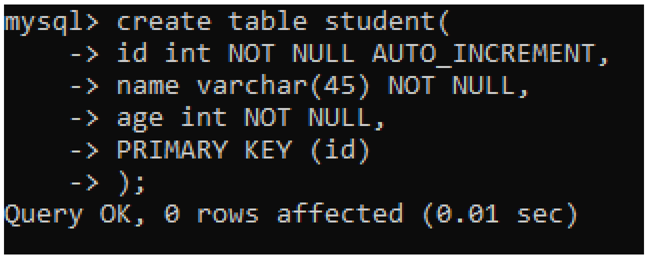
id int NOT NULL AUTO_INCREMENT,

name varchar(45) NOT NULL,

age int NOT NULL,

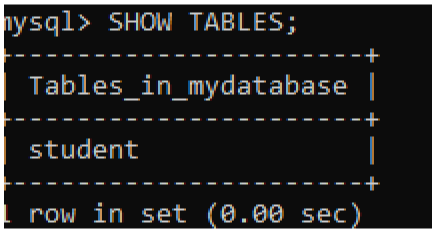
PRIMARY KEY (id)

);



```
mysql> create table student(
    -> id int NOT NULL AUTO_INCREMENT,
    -> name varchar(45) NOT NULL,
    -> age int NOT NULL,
    -> PRIMARY KEY (id)
    -> );
Query OK, 0 rows affected (0.01 sec)
```

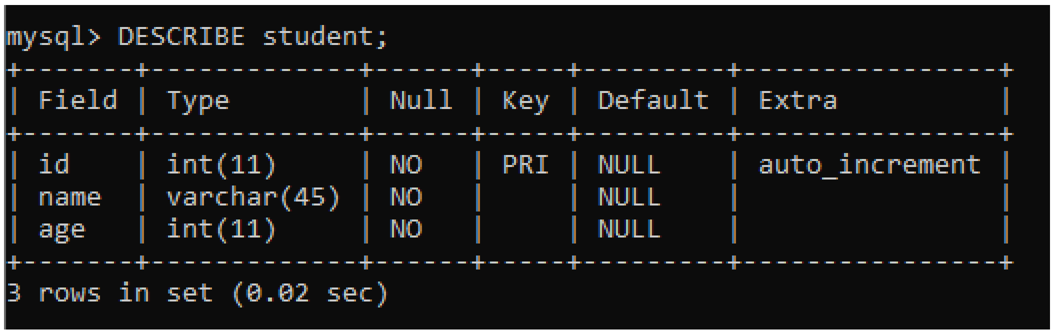
We need to use the following command to see the newly created table:



```
ysql> SHOW TABLES;
+----------------------+
| Tables_in_mydatabase |
+----------------------+
| student              |
+----------------------+
 row in set (0.00 sec)
```

Show the table structure

DESCRIBE student;



```
mysql> DESCRIBE student;
+-------+-------------+------+-----+---------+----------------+
| Field | Type        | Null | Key | Default | Extra          |
+-------+-------------+------+-----+---------+----------------+
| id    | int(11)     | NO   | PRI | NULL    | auto_increment |
| name  | varchar(45) | NO   |     | NULL    |                |
| age   | int(11)     | NO   |     | NULL    |                |
+-------+-------------+------+-----+---------+----------------+
3 rows in set (0.02 sec)
```

➢ ***Practice***

Now create the same database, table using MySQL Workbench.

<u>View Data in Table.</u>

SELECT * FROM table_name;

SELECT column1, column2

FROM table_name;

<u>Insert values into tables</u>

INSERT INTO table_name (field1, field2, ...)

VALUES (value1, value2, ...) ;

<u>Delete table data/Drop table/ Drop database</u>

- **DELETE FROM table_name;**
- **DROP TABLE table_name;**
- **DROP DATABASE db_name;**

Select Statement

The SELECT statement is used to select data from a database.

```
SELECT column1, column2, ...
FROM table_name;
```

```
SELECT field_name1, field_name 2,... field_nameN
FROM table_name1, table_name2...
[WHERE condition]
[GROUP BY field_name(s)]
[HAVING condition]
[ORDER BY field_name(s)]
[OFFSET M ][LIMIT N];
```

```
SELECT Name FROM student;
```

```
SELECT Name, Email, City FROM employee_detail;
```

```
SELECT * FROM employee_detail;
```