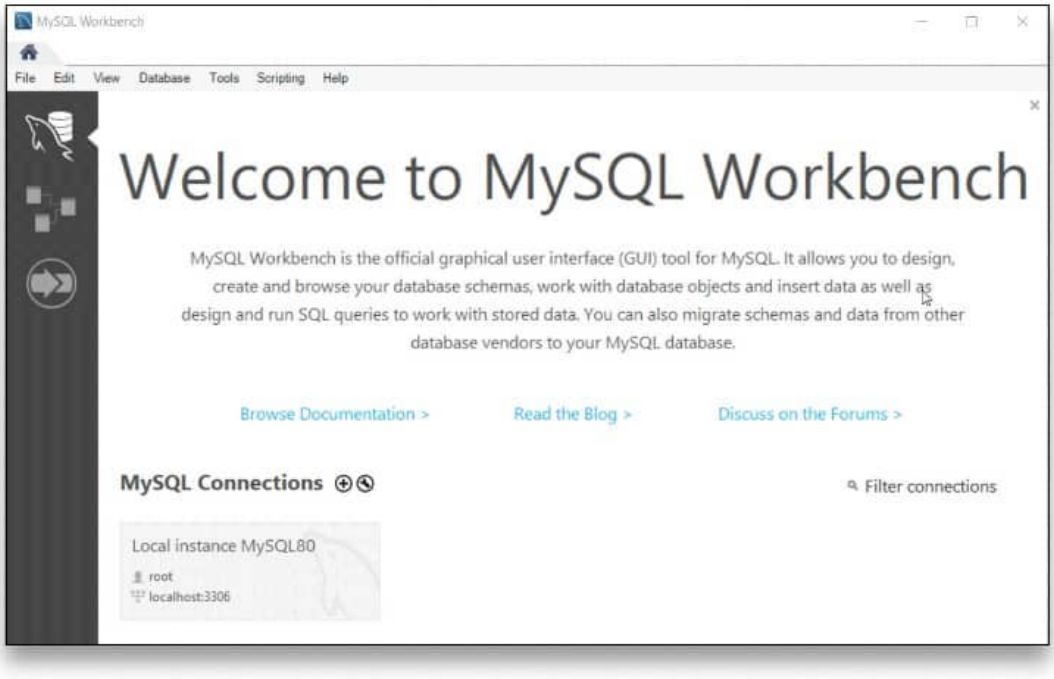
01

MySQL with Python

SAVE IT

Install MySQL Workbench.

MySQL Workbench is a unified visual tool for database architects, developers, and DBAs. MySQL Workbench provides data modeling, SQL development, and comprehensive administration tools for server configuration, user administration, backup, and much more. MySQL Workbench is available on Windows, Linux and Mac OS X.



1. Installing MySQL

Next, you have to install mysql.connector for Python. We need mysql.connector to connect Python Script to the MySQL database.

run in cmd:

- pip install mysql.connector-python

2. Create Database.

```
######################### creating databse ###################################
db = mysql.connect(
    host = "localhost",
    user = "root",
    passwd = "dbms"
)

cursor = db.cursor()

## creating a databse called 'pysql'
cursor.execute("CREATE DATABASE pysql")
```

2.1. Create Database without code.

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| pytest             |
| sakila             |
| sys                |
| world              |
+--------------------+
7 rows in set (0.00 sec)

mysql> create database pysql;
Query OK, 1 row affected (0.02 sec)

mysql> use pysql;
Database changed
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| pysql              |
| pytest             |
| sakila             |
| sys                |
| world              |
+--------------------+
8 rows in set (0.00 sec)

mysql>
```

- open mysql command line client.
- after enter password follow these command.
  - show databases; (for check database availability).
  - create database pysql; ('pysql' is database name).
  - use pysql;
  - show database;

2.2. Creating Tables.

Creating tables in the database to store the information. Before creating tables, we have to select a database first.

Run the following code, to select datacamp database which we have created a minute before.

```
########################## connect with MySQL ################################
try:
    db = mysql.connector.connect(user='root', password='root', host='localhost', database="pysql")
    print('Connet Successfully')
    cursor = db.cursor()
except mysql.connector.Error as err:
    print("Somthig is wrong")

########################## creating table in database##########################
cursor.execute("CREATE TABLE IF NOT EXISTS users (name VARCHAR(255), user_name VARCHAR(255))")
cursor.execute("SHOW TABLES")

tables = cursor.fetchall()
for table in tables:
    print('You are using table ', table)
```

Output

```
PS F:\PYTHON PROGRAMMING\Py_MySQL> python main.py
Connet Successfully
You are using table  ('users',)
```

3. Inserting Data

```
######################## Insert data into users table ########################
## defining the Query
query = "INSERT INTO users (name, user_name) VALUES (%s, %s)"
## storing values in a variable

values =  (("JavaScript", "js"), ("Python", "Py"))

## executing the query with values
for value in values:
    cursor.execute(query, value)

## to make final output we have to run the 'commit()' method of the database object
db.commit()

print(cursor.rowcount, "record inserted")
```

3. Select Data

```
# ###################### select data from users table ######################

query = "SELECT * FROM users"
## getting records from the table
cursor.execute(query)

## fetching all records from the 'cursor' object
records = cursor.fetchall()
print()
print('Select data is : ')
## Showing the data
for record in records:
    print(record)
```

Output

```
Select data is :
('JavaScript', 'js')
('Python', 'Py')
```

4. Select Data using where.

```
# ###################### select data from users table using Where ######################

## defining the Query
query = "SELECT * FROM users WHERE user_name = 'py'"

## getting records from the table
cursor.execute(query)

## fetching all records from the 'cursor' object
records = cursor.fetchall()

print()
print("Using whare condition: ")
## Showing the data
for record in records:
    print(record)
```

Output

```
Select data is :
('JavaScript', 'js')
('Python', 'Py')

Using whare condition:
('Python', 'Py')
```

5. Delete Data

```
# ######################## delete data from users table ######################

# defining the Query
query = "DELETE FROM users WHERE user_name = 'py'"

## executing the query
cursor.execute(query)

## final step to tell the database that we have changed the table data
db.commit()

query = "SELECT * FROM users"

cursor.execute(query)

records = cursor.fetchall()

print()
print("After delete some data : ")
for record in records:
    print(record)
```

Output

```
Select data is :
('JavaScript', 'js')
('Python', 'Py')

After delete some data :
('JavaScript', 'js')
```

6. Update Data

```python
# ######################## update data into users table ########################

## defining the Query
query = "UPDATE users SET name = '123' WHERE user_name = 'js'"

## executing the query
cursor.execute(query)

## final step to tell the database that we have changed the table data
db.commit()

query = "SELECT * FROM users"

cursor.execute(query)

records = cursor.fetchall()

print()
print("After update  data : ")
for record in records:
    print(record)
```

Output

```
Select data is :
('JavaScript', 'js')

After update  data :
('123', 'js')
```

Hope you have found this post helpful

PLEASE LIKE AND COMMENT
DON'T FORGET TO SHARE IT

SHARE IT
with someone who need this

SAVE IT FOR LATER