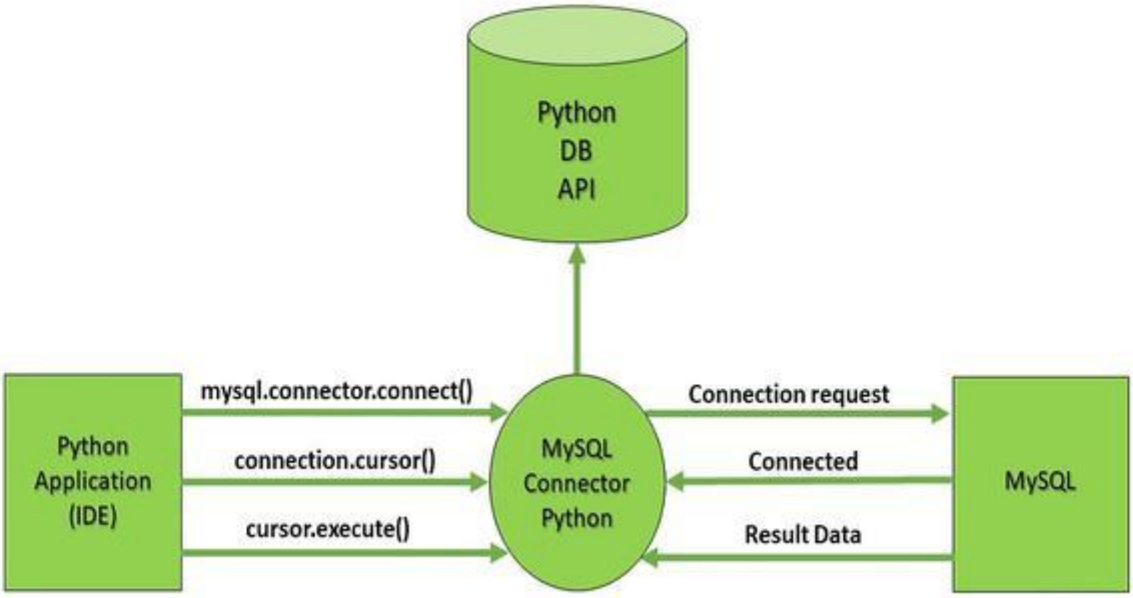
Python Database Connectivity

How to Connect Python with SQL Database?

- Python is a high-level, general-purpose, and very popular programming language.
- Basically, it was designed with an emphasis on code readability, and programmers can express their concepts in fewer lines of code.
- We can also use Python with SQL.
- Here, we will learn how to connect SQL with Python using the 'MySQL Connector Python module.
- The diagram given below illustrates how a connection request is sent to MySQL connector Python, how it gets accepted from the database and how the cursor is executed with result data.



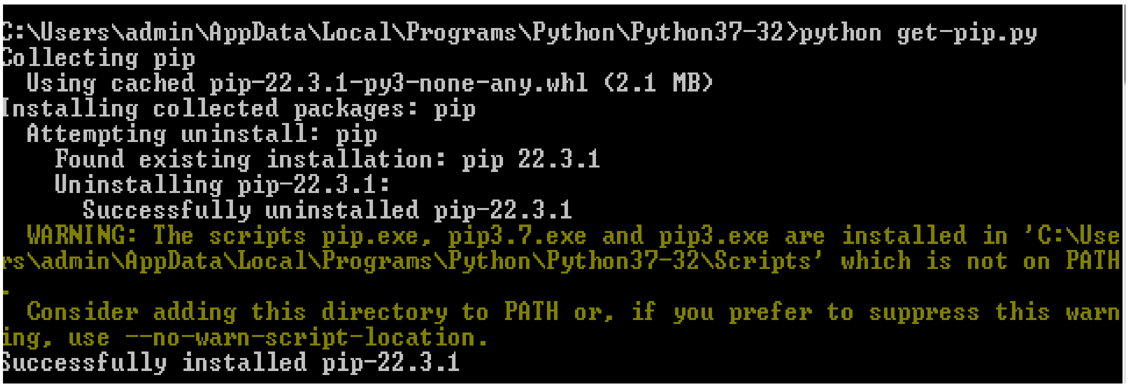
Connecting MySQL with Python

- To create a connection between the MySQL database and Python, the **connect()** method of **mysql.connector** module is used.
- We pass the database details like HostName, username, and the password in the method call, and then the method returns the connection object.
- The following steps are required to connect SQL with Python:

Step 1: Download and Install the free MySQL database.

Step 2: After installing the MySQL database, open your **Command prompt**.

Step 3: Navigate your Command prompt to the location of **PIP**. **PIP** is most likely already installed in your Python environment or download and install **get-pip.py**.



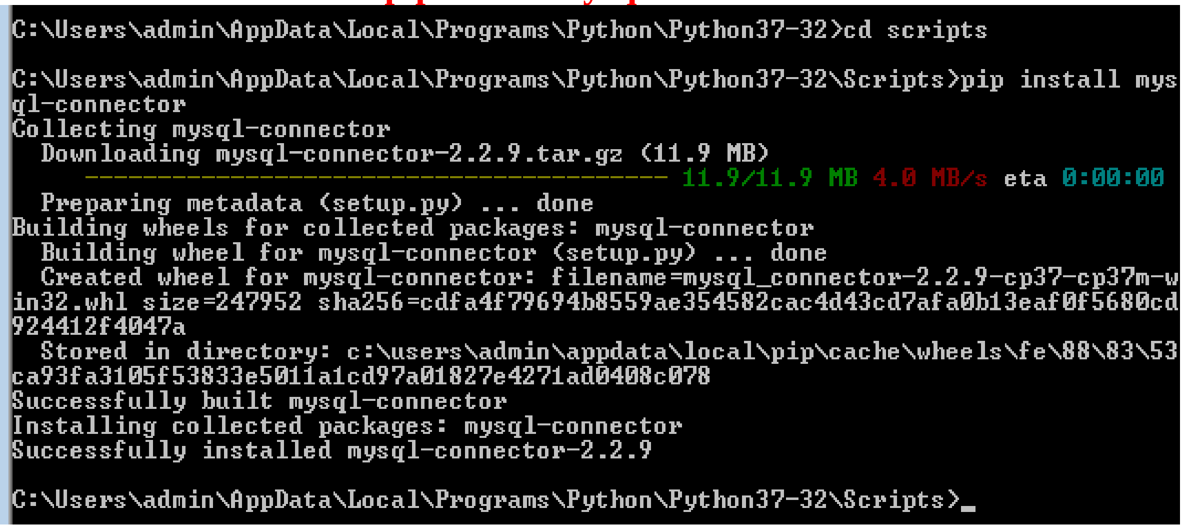
```
C:\Users\admin\AppData\Local\Programs\Python\Python37-32>python get-pip.py
Collecting pip
  Using cached pip-22.3.1-py3-none-any.whl (2.1 MB)
Installing collected packages: pip
  Attempting uninstall: pip
    Found existing installation: pip 22.3.1
    Uninstalling pip-22.3.1:
      Successfully uninstalled pip-22.3.1
  WARNING: The scripts pip.exe, pip3.7.exe and pip3.exe are installed in 'C:\Use
rs\admin\AppData\Local\Programs\Python\Python37-32\Scripts' which is not on PATH
.
  Consider adding this directory to PATH or, if you prefer to suppress this warn
ing, use --no-warn-script-location.
Successfully installed pip-22.3.1
```

Step 4: Navigate your command line to the location of **PIP** to **install mysql-connector**. Now run the commands given below to download and install **"MySQL Connector"**.

pip install mysql-connector-python

or

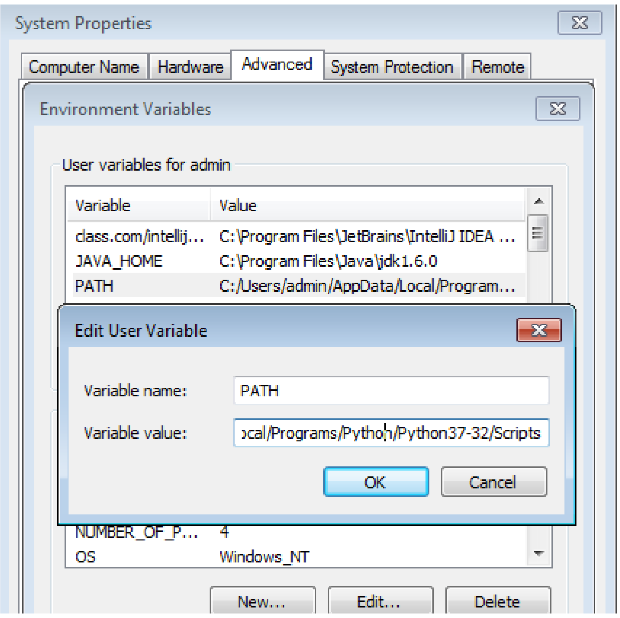
pip install mysql-connector



```
C:\Users\admin\AppData\Local\Programs\Python\Python37-32>cd scripts

C:\Users\admin\AppData\Local\Programs\Python\Python37-32\Scripts>pip install mys
ql-connector
Collecting mysql-connector
  Downloading mysql-connector-2.2.9.tar.gz (11.9 MB)
     ---------------------------------------- 11.9/11.9 MB 4.0 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Building wheels for collected packages: mysql-connector
  Building wheel for mysql-connector (setup.py) ... done
  Created wheel for mysql-connector: filename=mysql_connector-2.2.9-cp37-cp37m-w
in32.whl size=247952 sha256=cdfa4f79694b8559ae354582cac4d43cd7afa0b13eaf0f5680cd
924412f4047a
  Stored in directory: c:\users\admin\appdata\local\pip\cache\wheels\fe\88\83\53
ca93fa3105f53833e5011a1cd97a01827e4271ad0408c078
Successfully built mysql-connector
Installing collected packages: mysql-connector
Successfully installed mysql-connector-2.2.9

C:\Users\admin\AppData\Local\Programs\Python\Python37-32\Scripts>_
```

❖ Here, **mysql.connector** statement will help you to communicate with the MySQL database.

❖ Set **PATH** in System Properties - **Environment Variable**



Variable	Value
class.com/intellij...	C:\Program Files\JetBrains\IntelliJ IDEA ...
JAVA_HOME	C:\Program Files\Java\jdk1.6.0
PATH	C:/Users/admin/AppData/Local/Program...

Edit User Variable

Variable name: PATH

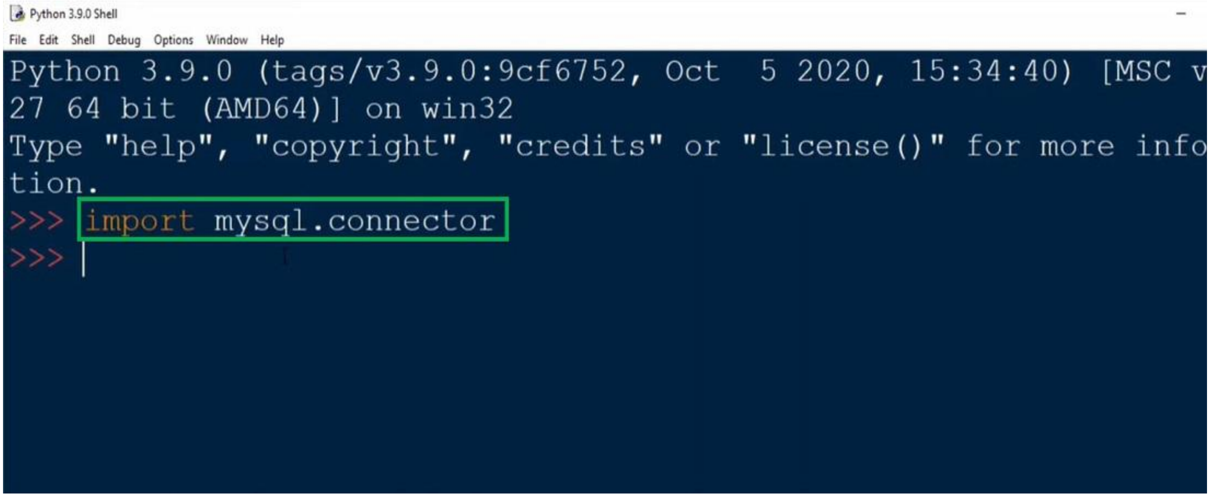
Variable value: ocal/Programs/Python/Python37-32/Scripts

NUMBER_OF_P...	4
OS	Windows_NT

Step 5: Test MySQL Connector

❖ To check if the installation was successful, or if you already installed **“MySQL Connector”**, go to your IDE and run the given below code:

import mysql.connector



```
Python 3.9.0 (tags/v3.9.0:9cf6752, Oct  5 2020, 15:34:40) [MSC v
27 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more info
tion.
>>> import mysql.connector
>>> |
```

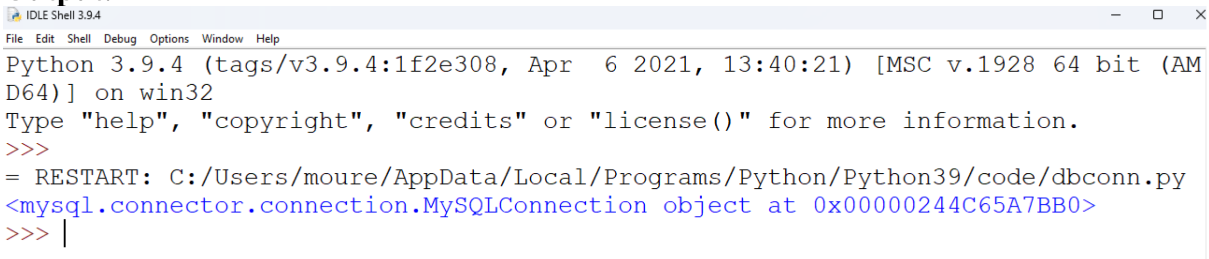
❖ If the above code gets executed with no errors, **“MySQL Connector”** is ready to be used.

Step 6: Create Connection

❖ Now to connect SQL with Python, run the code given below in your IDE.

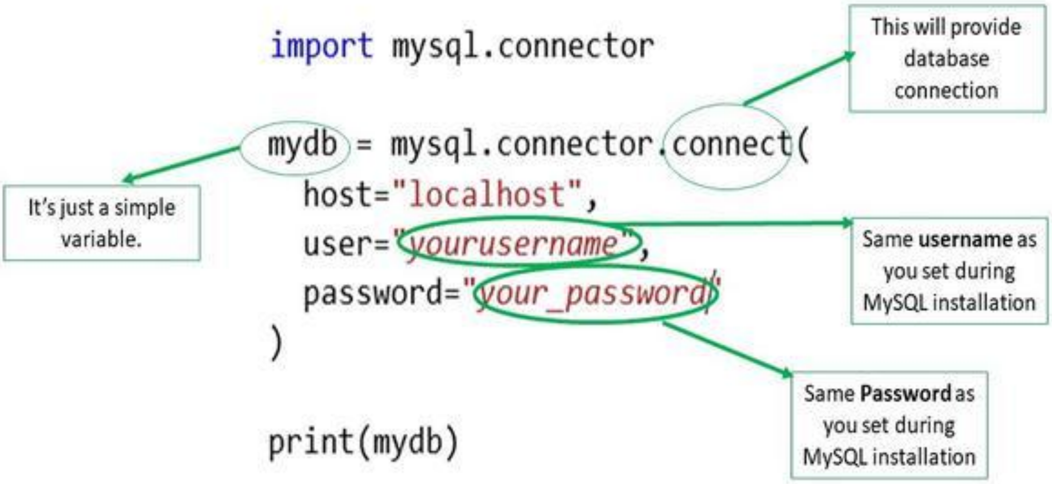
```
# Importing module
import mysql.connector

# Creating connection object
mydb = mysql.connector.connect(
        host="localhost",
        user="root",
        password=" "
)

# Printing the connection object
print(mydb)
```

Output:



```
Python 3.9.4 (tags/v3.9.4:1f2e308, Apr  6 2021, 13:40:21) [MSC v.1928 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/moure/AppData/Local/Programs/Python/Python39/code/dbconn.py
<mysql.connector.connection.MySQLConnection object at 0x00000244C65A7BB0>
>>> |
```

❖ **Here, in the above code:**



```
import mysql.connector

mydb = mysql.connector.connect(
  host="localhost",
  user="yourusername",
  password="your_password"
)

print(mydb)
```

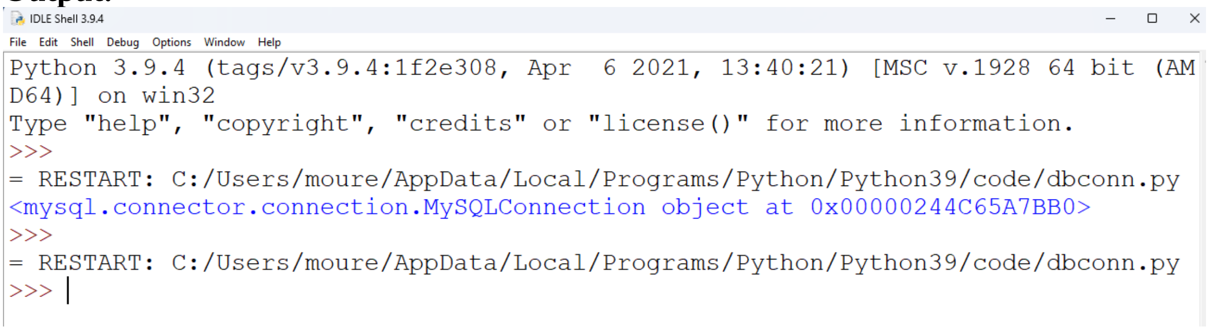
This will provide database connection
It's just a simple variable.
Same username as you set during MySQL installation
Same Password as you set during MySQL installation

Creating MySQL Database

❖ To create a database, we will use CREATE DATABASE database_name statement and we will execute this statement by creating an instance of the **'cursor'** class.

```
import mysql.connector

mydb = mysql.connector.connect(
        host="localhost",
        user="root",
        password=""
)

# Creating an instance of 'cursor' class
# which is used to execute the 'SQL' statements in 'Python'
cursor = mydb.cursor()

# Creating a database with a name
# 'kongu' execute() method is used to compile a SQL statement
# below statement is used to create the 'kongu' database
cursor.execute("CREATE DATABASE kongu")
```

Output:



```
Python 3.9.4 (tags/v3.9.4:1f2e308, Apr  6 2021, 13:40:21) [MSC v.1928 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/moure/AppData/Local/Programs/Python/Python39/code/dbconn.py
<mysql.connector.connection.MySQLConnection object at 0x00000244C65A7BB0>
>>>
= RESTART: C:/Users/moure/AppData/Local/Programs/Python/Python39/code/dbconn.py
>>> |
```

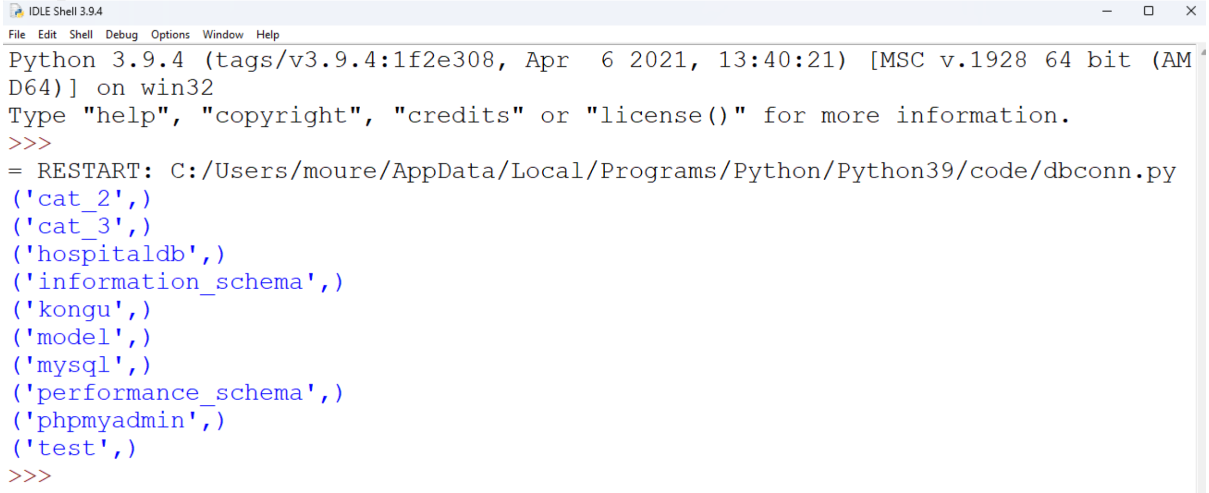
❖ If the database with the name ‘kongu’ already exists then you will get an error, otherwise no error.

❖ So make sure that the new database that you are creating does not have the same name as the database already you created or exists previously.

❖ Now to check the databases that you created, use ***“SHOW DATABASES”*** *– SQL statement* i.e. cursor.execute(“SHOW DATABASES”)

```
# Importing module
import mysql.connector

# Creating connection object
mydb = mysql.connector.connect(
        host="localhost",
        user="root",
        password=""
)

'''Creating an instance of 'cursor' class which is used to
execute the 'SQL' statements in 'Python' '''
cursor = mydb.cursor()

# Show database
cursor.execute("SHOW DATABASES")

for x in cursor:
   print(x)
```

Output:



```
Python 3.9.4 (tags/v3.9.4:1f2e308, Apr  6 2021, 13:40:21) [MSC v.1928 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> 
= RESTART: C:/Users/moure/AppData/Local/Programs/Python/Python39/code/dbconn.py
('cat_2',)
('cat_3',)
('hospitaldb',)
('information_schema',)
('kongu',)
('model',)
('mysql',)
('performance_schema',)
('phpmyadmin',)
('test',)
>>>
```

Creating Tables

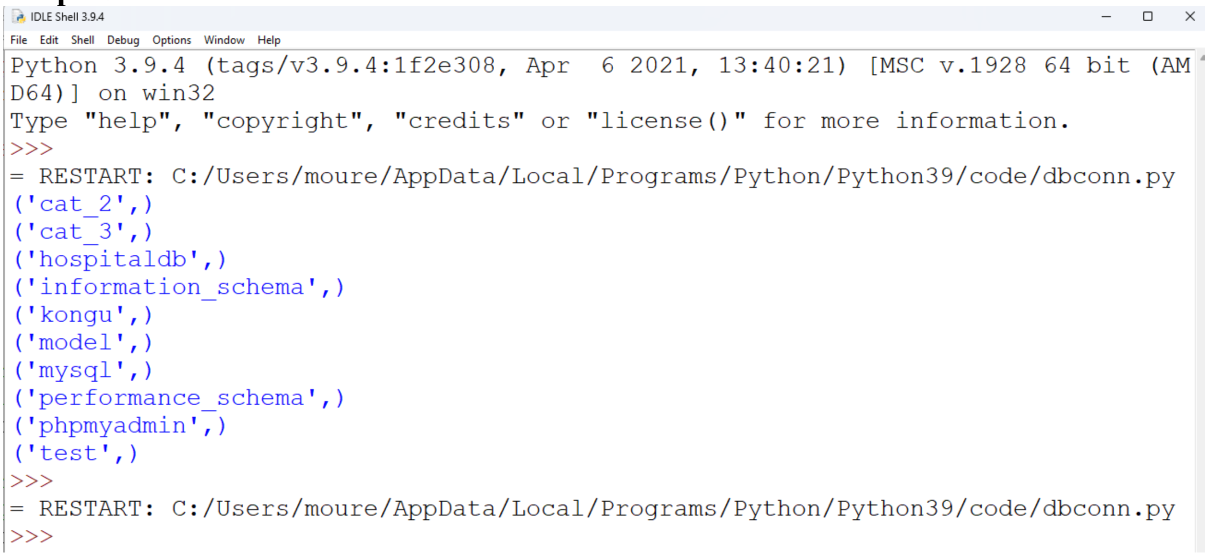
❖ Now to create **tables** in a database, **first,** we have to select a database and for that, we will pass ***database = “NameofDatabase”*** as your fourth parameter in connect() function.

❖ Since we have created a database with the name **‘kongu’** above, so we will use that and create our tables.

❖ We will use ***CREATE TABLE*** *kec (variableName1 datatype, variableName2 datatype)* statement to create our table with the name ‘kec’.

```
# Importing module
import mysql.connector

# Creating connection object
mydb = mysql.connector.connect(
            host="localhost",
            user="root",
            password="",
        database = "kongu"
)

'''Creating an instance of 'cursor' class which is used to
execute the 'SQL' statements in 'Python' '''
cursor = mydb.cursor()

'''Creating a table called 'kec' in the
'kongu' database'''
cursor.execute("CREATE TABLE kec(name VARCHAR(255), user_name
VARCHAR(255))")
```

Output:



```
Python 3.9.4 (tags/v3.9.4:1f2e308, Apr  6 2021, 13:40:21) [MSC v.1928 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/moure/AppData/Local/Programs/Python/Python39/code/dbconn.py
('cat_2',)
('cat_3',)
('hospitaldb',)
('information_schema',)
('kongu',)
('model',)
('mysql',)
('performance_schema',)
('phpmyadmin',)
('test',)
>>>
= RESTART: C:/Users/moure/AppData/Local/Programs/Python/Python39/code/dbconn.py
>>>
```

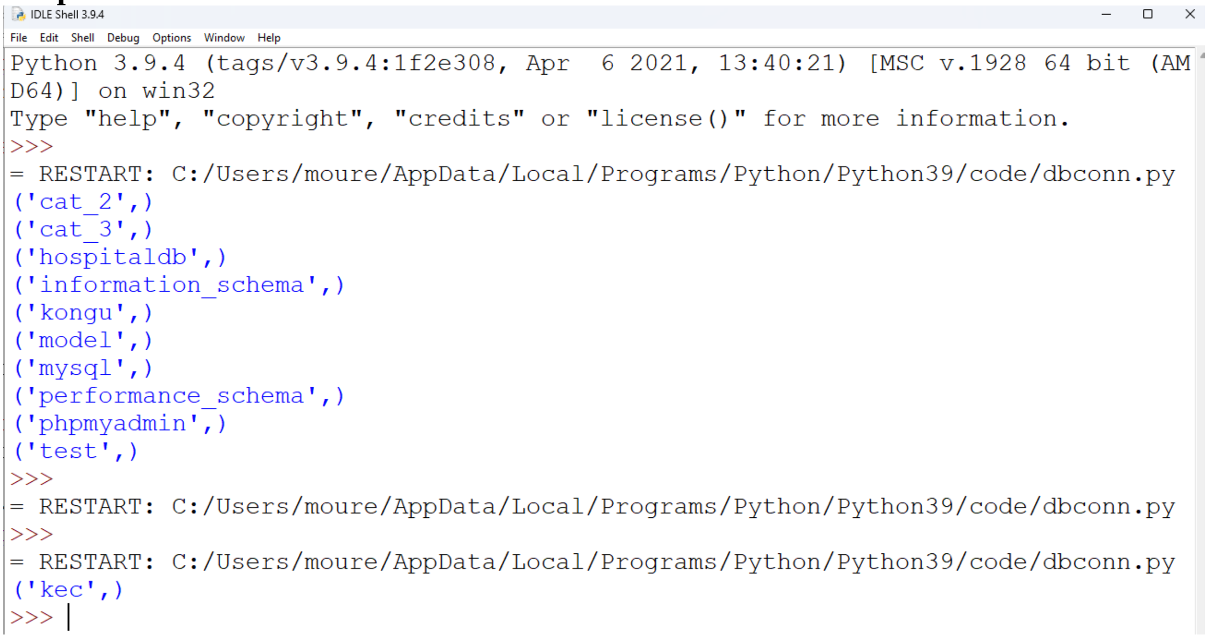
❖ If the table with the name ‘kec’ already exists, you will get an error, otherwise no error.

❖ So make sure that the new table that you are creating does not have the same name as the table already you created or exists previously.

❖ Now to check tables that you created, use ***“SHOW TABLES”*** – *SQL statement* i.e. cursor.execute(“SHOW TABLES”).

```
# Importing module
import mysql.connector

# Creating connection object
mydb = mysql.connector.connect(
            host="localhost",
            user="root",
            password="",
    database = "kongu"
)

cursor = mydb.cursor()

# Show existing tables
cursor.execute("SHOW TABLES")

for x in cursor:
  print(x)
```

Output:

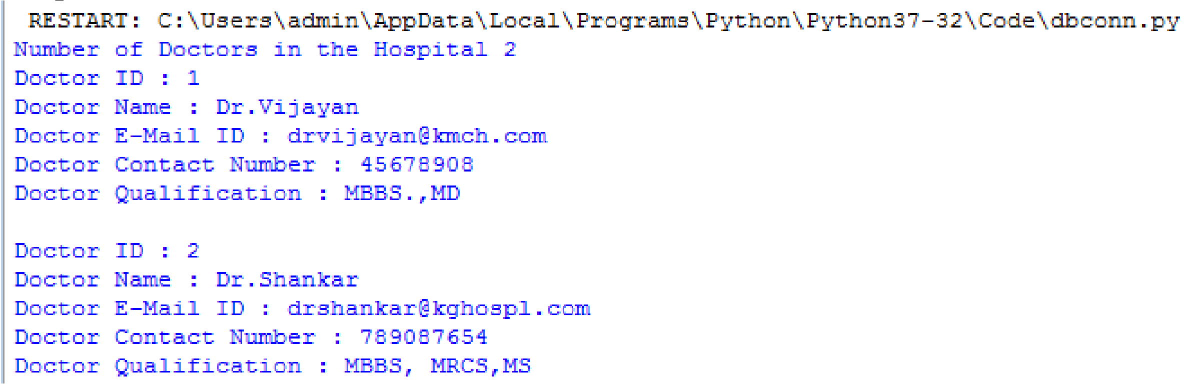


```
Python 3.9.4 (tags/v3.9.4:1f2e308, Apr  6 2021, 13:40:21) [MSC v.1928 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/moure/AppData/Local/Programs/Python/Python39/code/dbconn.py
('cat_2',)
('cat_3',)
('hospitaldb',)
('information_schema',)
('kongu',)
('model',)
('mysql',)
('performance_schema',)
('phpmyadmin',)
('test',)
>>>
= RESTART: C:/Users/moure/AppData/Local/Programs/Python/Python39/code/dbconn.py
>>>
= RESTART: C:/Users/moure/AppData/Local/Programs/Python/Python39/code/dbconn.py
('kec',)
>>> 
```

Python Program to interact with database

```
import mysql.connector
conn =
mysql.connector.connect(host="localhost",user="root",password="",database="hospitaldb")
cursor=conn.cursor()
selectquery="select * from doctor"
cursor.execute(selectquery)
records=cursor.fetchall()
print("Number of Doctors in the Hospital",cursor.rowcount)
for row in records:
  print("Doctor ID :", row[0])
  print("Doctor Name :", row[1])
  print("Doctor E-Mail ID :", row[2])
  print("Doctor Contact Number :", row[3])
  print("Doctor Qualification :", row[4])
  print()
cursor.close()
conn.close()
```

Output:



```
 RESTART: C:\Users\admin\AppData\Local\Programs\Python\Python37-32\Code\dbconn.py
Number of Doctors in the Hospital 2
Doctor ID : 1
Doctor Name : Dr.Vijayan
Doctor E-Mail ID : drvijayan@kmch.com
Doctor Contact Number : 45678908
Doctor Qualification : MBBS.,MD

Doctor ID : 2
Doctor Name : Dr.Shankar
Doctor E-Mail ID : drshankar@kghospl.com
Doctor Contact Number : 789087654
Doctor Qualification : MBBS, MRCS,MS
```

Notes:

- **mysql.connector** allows Python programs to access MySQL databases.
- **connect()** method of the MySQL Connector class with the arguments will connect to MySQL and would return a MySQLConnection object if the connection is established successfully.
- **user = "yourusername",** here "yourusername" should be the same username as you set during MySQL installation.
- **password = "your_password"**, here "your_password" should be the same password as you set during MySQL installation.
- **cursor()** is used to **execute** the SQL statements in Python.
- **execute()** method is used to **compile** a SQL statement.

Querying Data from a Database using fetchone() and fetchall()

- The fetchone() and fetchall() are the methods of Python MySQL connector and they are used to display data.
- This connector helps in enabling the Python programs to use MySQL databases.

1. Fetchone(): Fetchone() method is used when there is a need to retrieve only the first row from the table.

- The method only returns the first row from the defined table.

Syntax:

row = cursor.fetchone()

Steps for using fetchone() in Mysql using Python:

- First, import MySQL connector
- Now, create a connection with the MySQL connector using connect() method
- Next, create a cursor object with the cursor() method
- Now create and execute the query using "SELECT *" statement with execute() method to retrieve the data

- Use **fetchone()** method on the result variable.
- print the result

Example:

```
# Importing module
import mysql.connector

# Creating connection object
mydb = mysql.connector.connect(
        host="localhost",
        user="root",
        password="",
        database = "kongu"
)

mycursor = mydb.cursor()

mycursor.execute("SELECT * FROM kec")

result = mycursor.fetchone()

print(result)
```

2. Fetchall():

Fetchall() is a method that fetches all the remaining tuples from the last executed statement from a table (returns a list of tuples).

- ❖ The method only returns the first row from the defined table and If there are no tuples then it returns an empty list in the output.

Syntax:

row = cursor.fetchall()

Steps for using fetchall() in Mysql using Python:

- First. import MySQL connector
- Now, create a connection with the MySQL connector using connect() method
- Next, create a cursor object with the cursor() method
- Now create and execute the query using “SELECT *” statement with execute() method to retrieve the data
- Use fetchall() method on the result variable.
- print the result using for each loop to display all

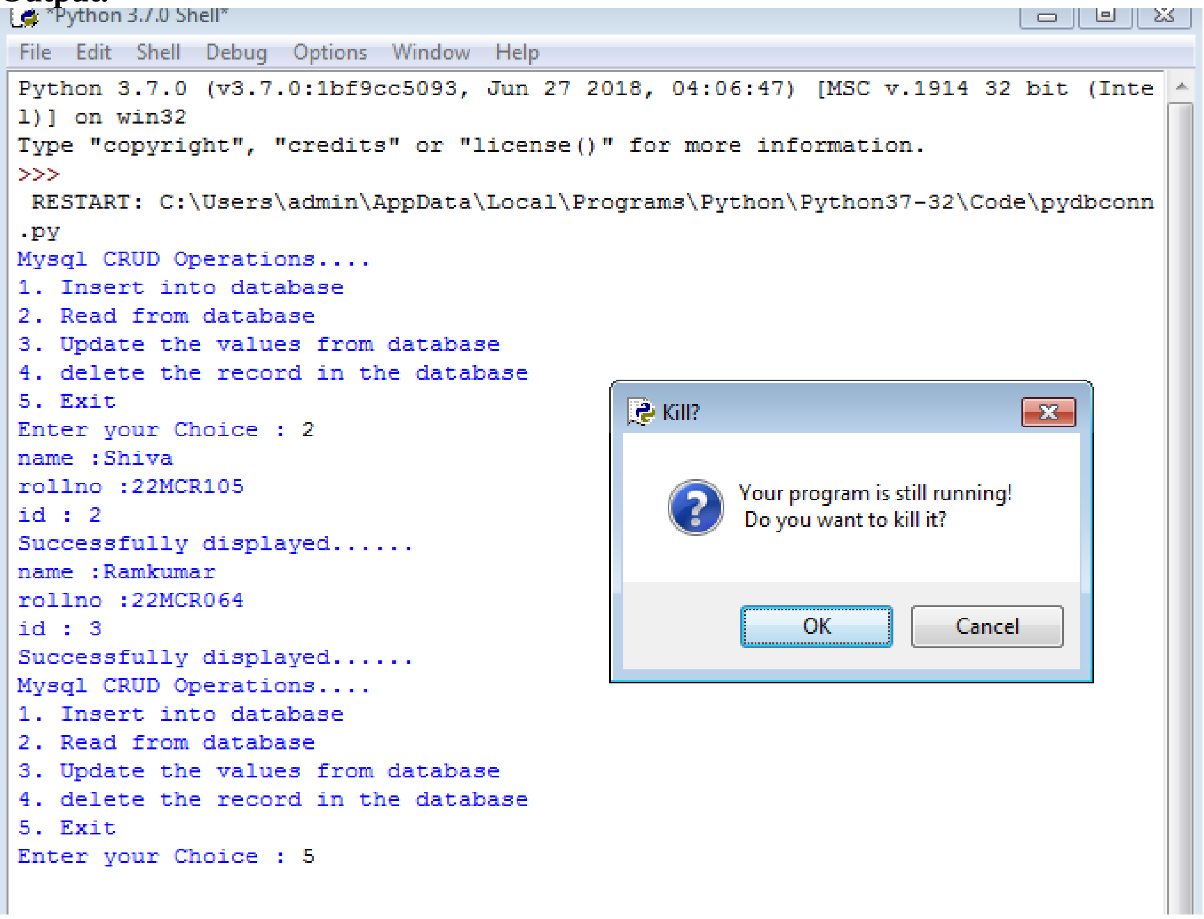
Example:

```
# Importing module
import mysql.connector

# Creating connection object
mydb = mysql.connector.connect(
        host="localhost",
        user="root",
        password="",
        database = "kongu"
)

mycursor = mydb.cursor()

mycursor.execute("SELECT * FROM kec")

# This SQL statement selects all data from the CUSTOMER table.
result = mycursor.fetchall()

# Printing all records or rows from the table. It returns a result set.
for all in result:
        print(all)
```

Python Program to perform CRUD operations on database (MySQL/Oracle)

```
import mysql.connector

mydb = mysql.connector.connect(
    host="localhost",
    user="root",
    password="",
    database="student"
)
mycursor = mydb.cursor()
while True:
    print('Mysql CRUD Operations....')
    print('1. Insert into database')
    print('2. Read from database')
    print('3. Update the values from database')
    print('4. delete the record in the database')
    print('5. Exit')
    choice = int(input('Enter your Choice : '))
    if choice==1:
        val = (input('Enter name : '), input('Enter rollno : '))
        sql = "INSERT INTO table1 (name, rollno) VALUES (%s, %s)"
        mycursor.execute(sql, val)
        mydb.commit()
        print('Inserted succesfully....')
    elif choice==2:
        mycursor.execute("SELECT * FROM table1")
        myresult = mycursor.fetchall()
        for x in myresult:
            print('name :'  + x[0] + '\nrollno :'  + x[1] + '\nid : '  + str(x[2]))
            print('Successfully displayed......')
    elif choice==3:
        try:
            up = (input('Enter new name : '),input('Enter roll no. : '), input('\nEnter id : '))
            sql = "UPDATE table1 SET name = %s, rollno = %s WHERE id = %s"
            mycursor.execute(sql, up)
            mydb.commit()
            print('Updated successfully...')
        except:
            print('Error in update values')
    elif choice==4:
        try:
            delete = input('Enter the table row id to delete : ')
            sql = "DELETE FROM table1 WHERE id = " + delete
            mycursor.execute(sql)
            mydb.commit()
            print(mycursor.rowcount, "record deleted")
        except:
            print('error in delete table records')
    elif choice==5:
        exit()
    else:
        print('Please choose correct option')
print('done')
```

Output:



```
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:06:47) [MSC v.1914 32 bit (Inte
l)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
 RESTART: C:\Users\admin\AppData\Local\Programs\Python\Python37-32\Code\pydbconn
.py
Mysql CRUD Operations....
1. Insert into database
2. Read from database
3. Update the values from database
4. delete the record in the database
5. Exit
Enter your Choice : 2
name :Shiva
rollno :22MCR105
id : 2
Successfully displayed......
name :Ramkumar
rollno :22MCR064
id : 3
Successfully displayed......
Mysql CRUD Operations....
1. Insert into database
2. Read from database
3. Update the values from database
4. delete the record in the database
5. Exit
Enter your Choice : 5
```

Your program is still running!
Do you want to kill it?

****