INTERFACE PYTHON WITH MYSQL

By:
Ms. Swati Chawla
(PGT Computer)

Introduction

- When you design real life applications, you are bound to encounter situations wherein you need to manipulate data stored in a database through an application designed by you.
- In order to connect to a database from within the Python, you need a library that provides connectivity functionality. There are so many libraries , but we will use **mysql connector** library.

Installing MYSQL Connector

- Write the following command:

  pip install mysql-connector

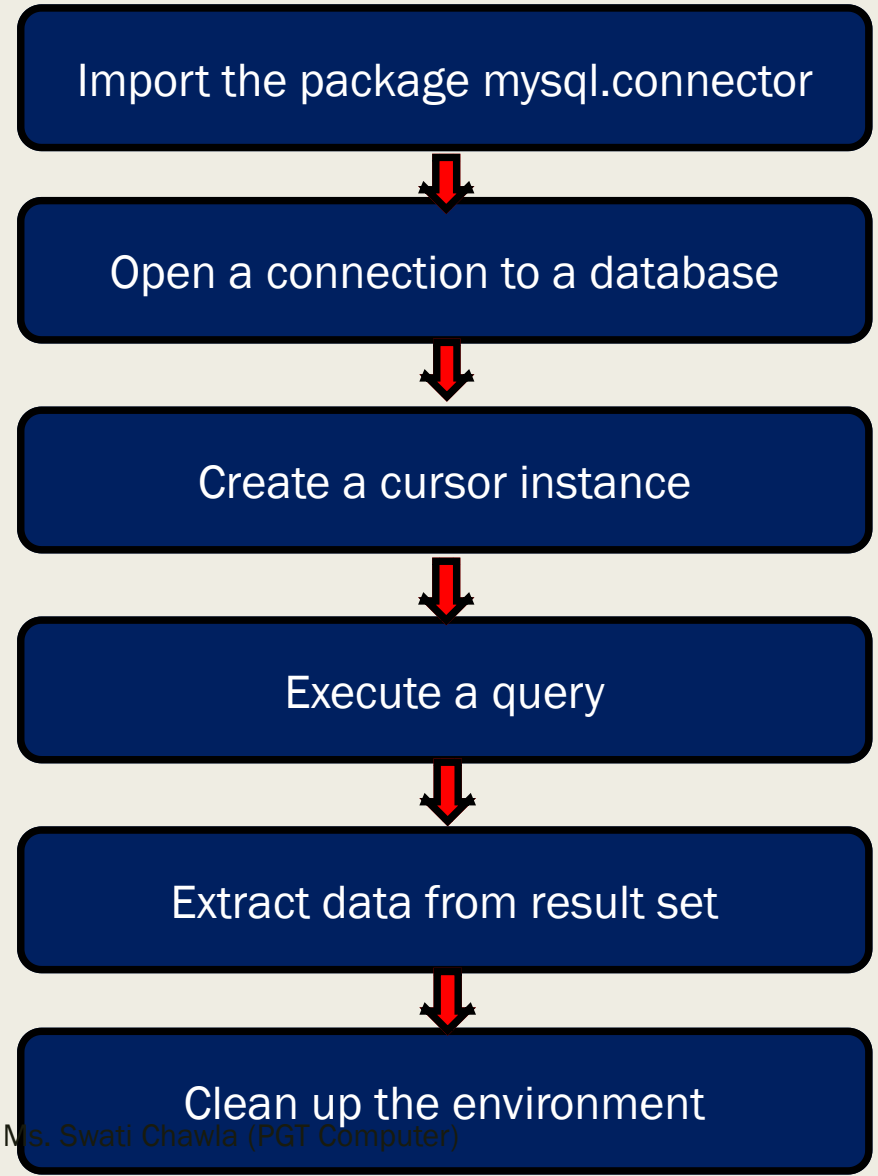
  in the cmd prompt

- After installing , open Python shell and write

  import mysql.connector

- If it will not show any error, it means you have successfully installed it.

Steps for creating Database Connectivity

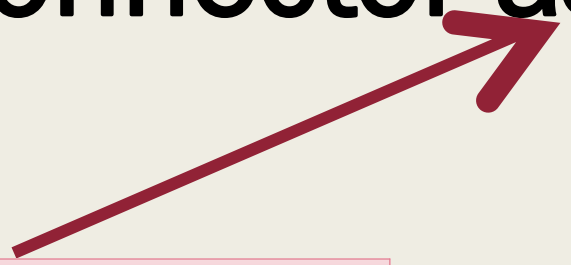


import mysql.connector

import mysql.connector

OR

import mysql.connector as sqltor



Member Aliasing

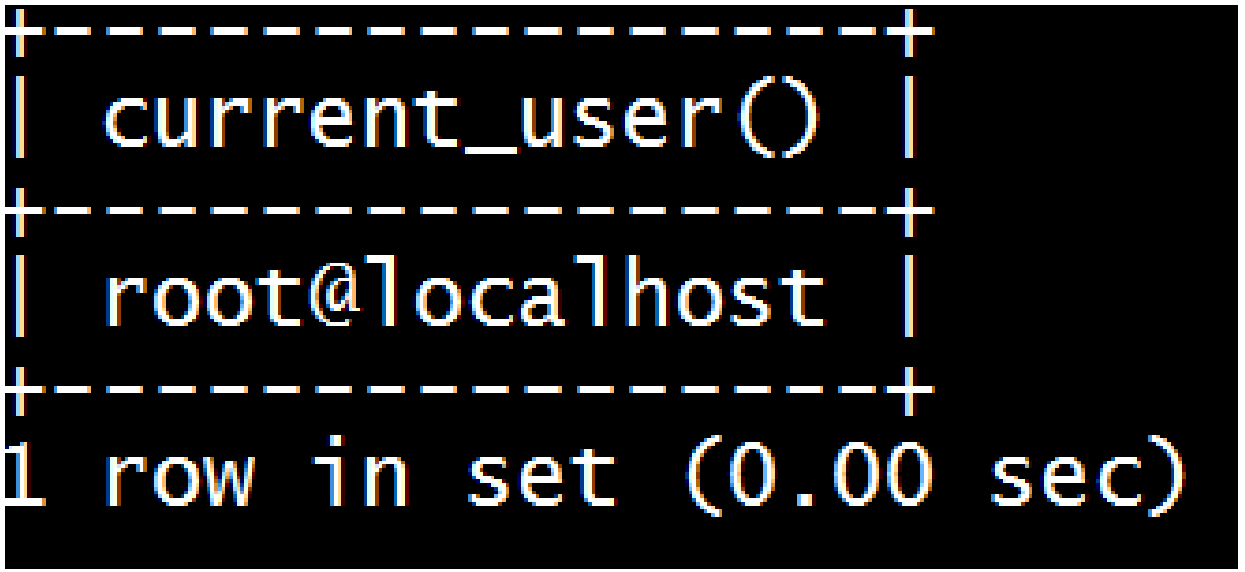
Open a connection to MYSQL Database

```
con=mysql.connector.connect(host=<HostName>,
                            user=<username>,
                            password=<password>,
                            database=<DatabaseName>)

con=mysql.connector.connect(host='localhost',
                            user='root',
                            password='123',
                            database='school')
```

To check the current user and host of a system, type the following command:

mysql>select current_user();



Create a cursor instance

- A database cursor is a special control structure that facilitates the row by row processing of records in the resultset, i.e. set of records retrieved as per query.

CursorObject = ConnectionObject . cursor()

Execute SQL Query

- Once you have created a cursor, you can execute SQL query using execute() function with cursor object.
- Syntax:

CursorObject.execute(<SQL Query string>)

Extract data from Resultset

- **Data=cursor.fetchall()** - Return all the records retrieved as per query in a tuple form.
- **data=cursor.fetchone() –** It will return one record from the resultset as a tuple. First time, it will fetch the first record, next time it will fetch the second record.
- **data=cursor.fetchmany(n) –** This method fetches n records in the form of a tuple.
- **Variable=cursor.rowcount –** This property of a cursor object returns the number of rows retrieved.

Parameterized Query

- **(i) Old Style: String Templates with % formatting**

f%v

Where,

f is a template string

v specifies the values to be formatted using the template. v must be a tuple.

"Select * from student where marks > %s" % (70,)

Parameterized Query

(ii) New Style : String Template with % formatting

```
Str="Select * from student where
   marks > { } and section = '{ }'
   ".format (70,'B')
```

commit( )

- Commit ( ) function is used to save the changes in a database physically.
- After performing : insert , update and delete queries , call the commit ( ) function with the help of an connection object.

Performing insert operation.....

```
import mysql.connector
con=mysql.connector.connect(host='localhost',
                            user='root',
                            password='123',
                            database='Books')
cur=con.cursor()
while True:
    no=int(input("Enter book number:"))
    n=input("Enter book name :")
    p=float(input("Enter price:"))
    query="Insert into compsci values({},'{}',{})".format(no,n,p)
    cur.execute(query)
    con.commit()
    print("Row inserted successfully...")
    ch=input("Do you want to enter more records?(y/n)")
    if ch=='n':
        break
```