


Working with MySQL and Python

Installing MySQL Workbench

- Download MYSQL application. Go to page : <https://dev.mysql.com/downloads/installer/>
- Click on the link highlighted below.

MySQL Community Downloads

MySQL Installer

- General Availability (GA) Releases
- Archives
- 


MySQL Installer 8.0.19

Select Operating System:

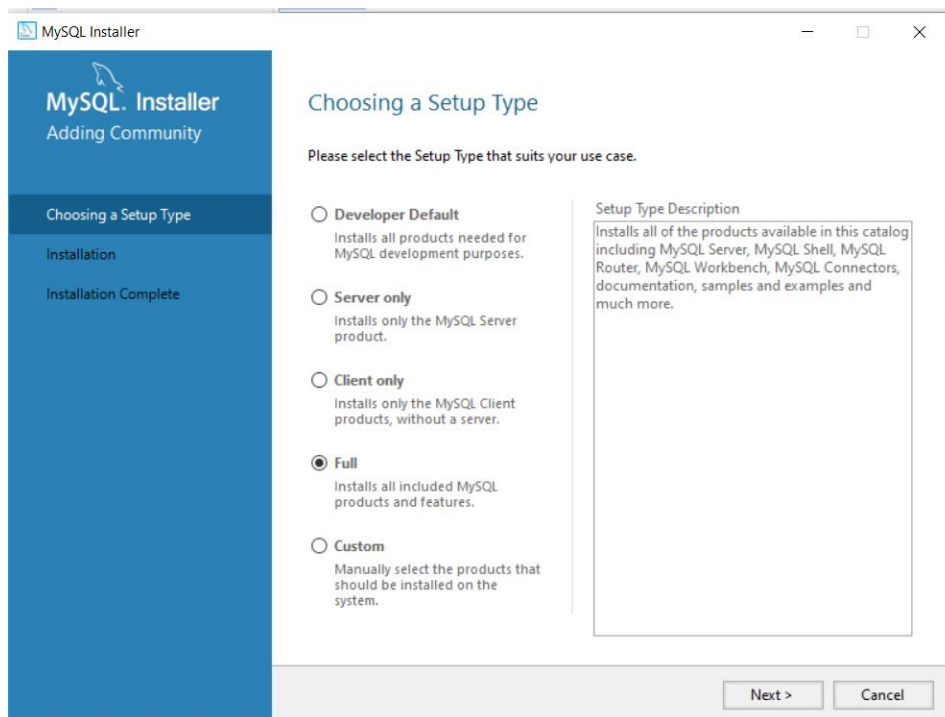
Microsoft Windows

Looking for previous GA versions?

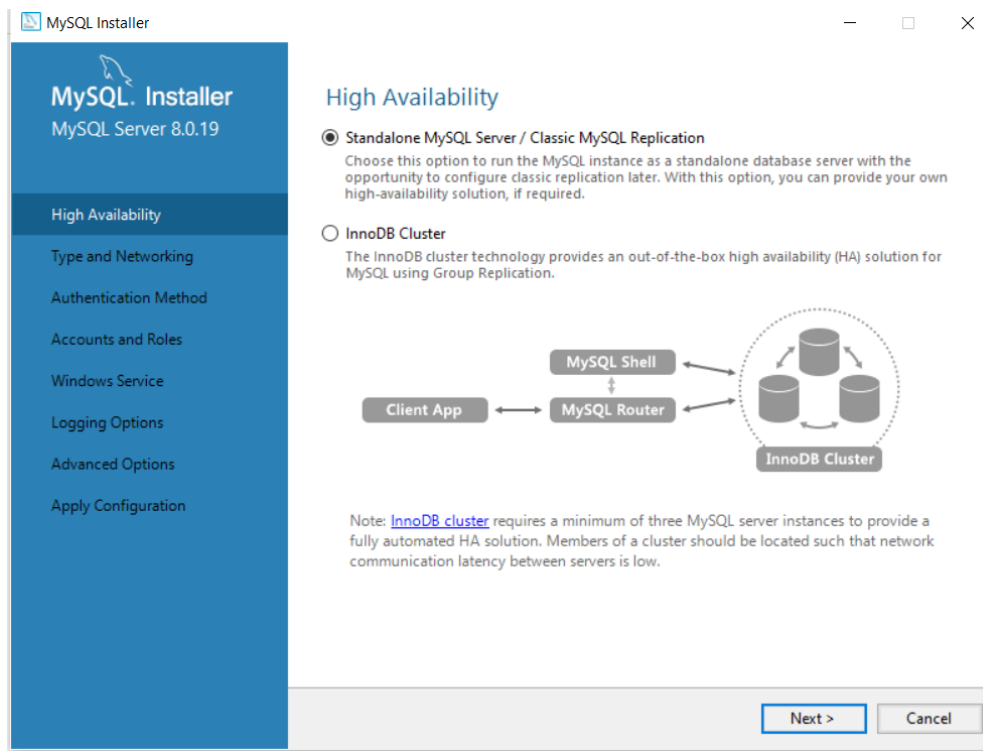
Windows (x86, 32-bit), MSI Installer (mysql-installer-web-community-8.0.19.0.msi)	8.0.19	18.6M	Download
Windows (x86, 32-bit), MSI Installer (mysql-installer-community-8.0.19.0.msi)	8.0.19	398.9M	Download

 We suggest that you use the MD5 checksums and GnuPG signatures to verify the integrity of the packages you download.

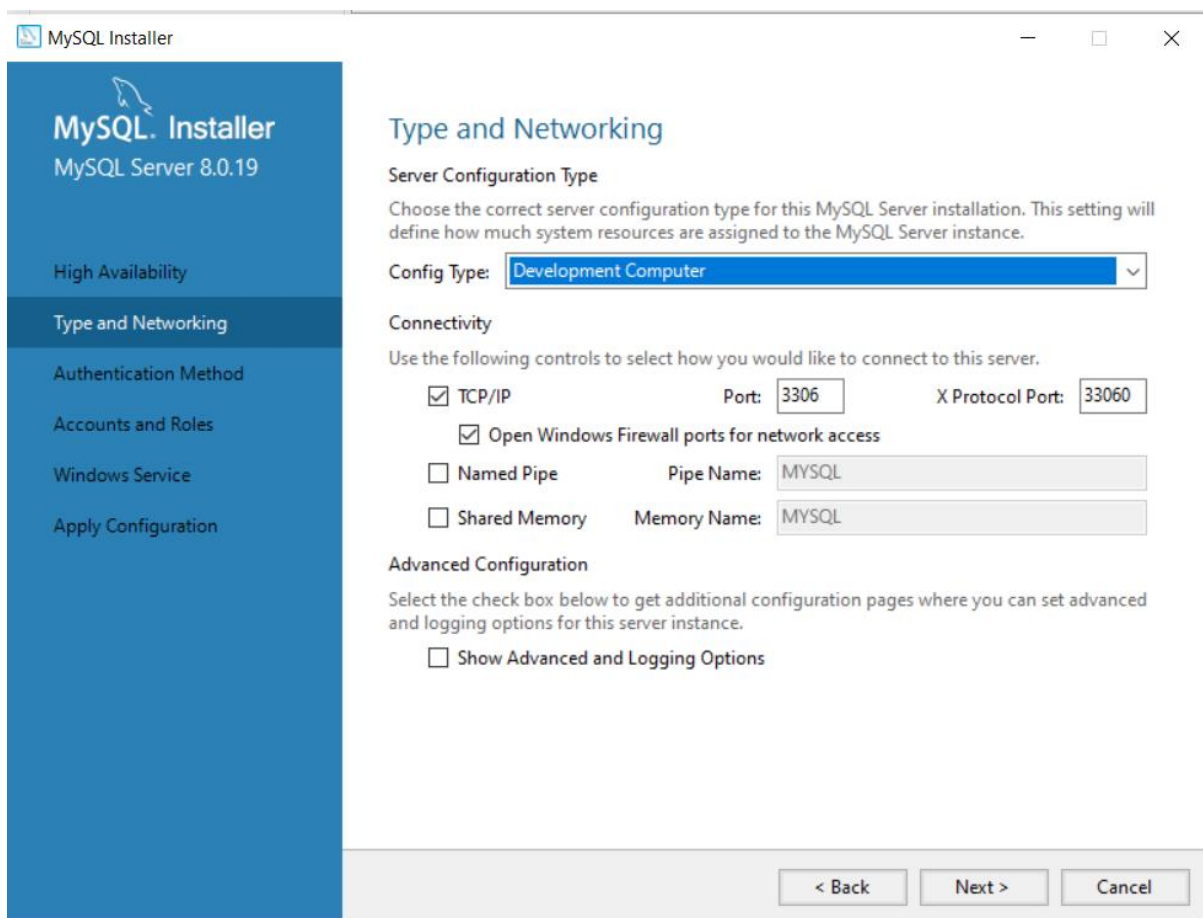
- Once downloaded, install the application. Follow the instructions and continue with installation.

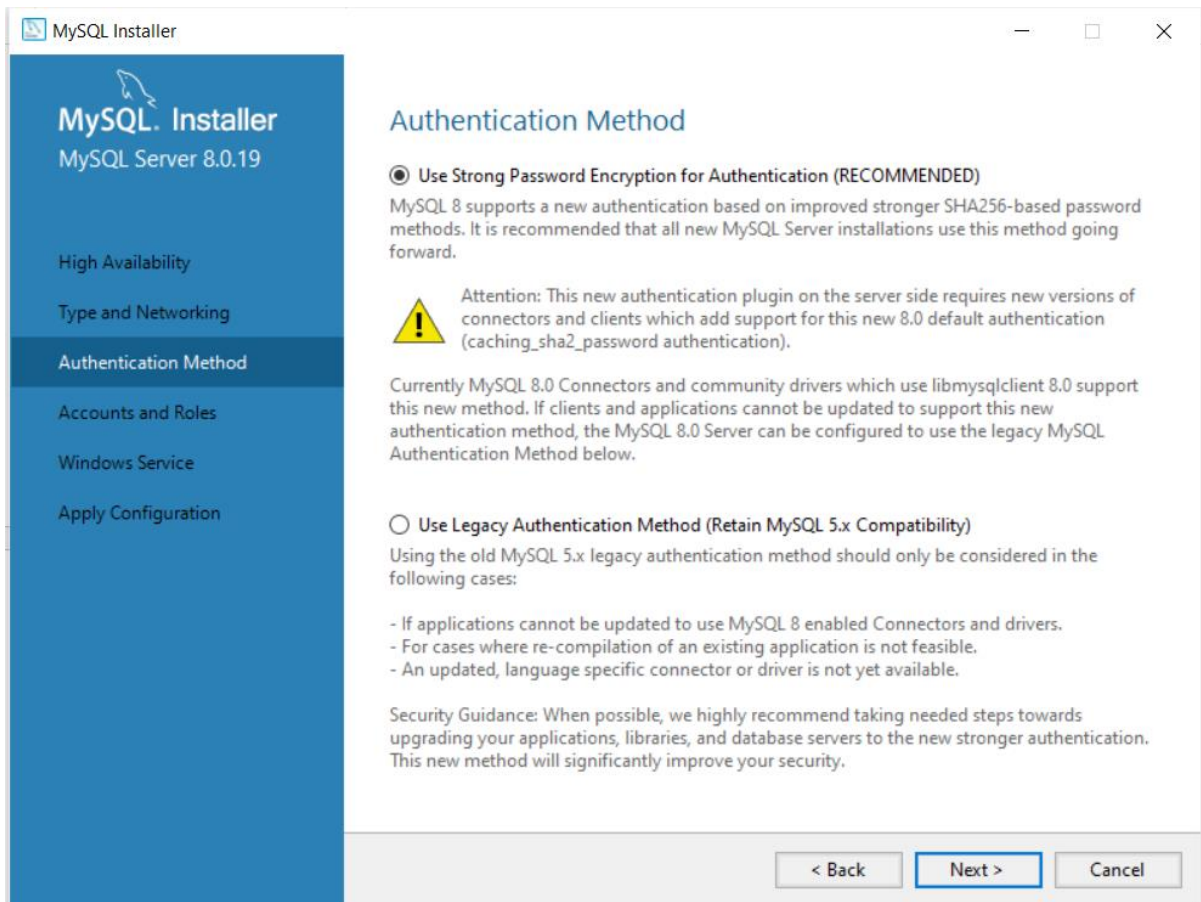


- Select Standalone MySQL server option

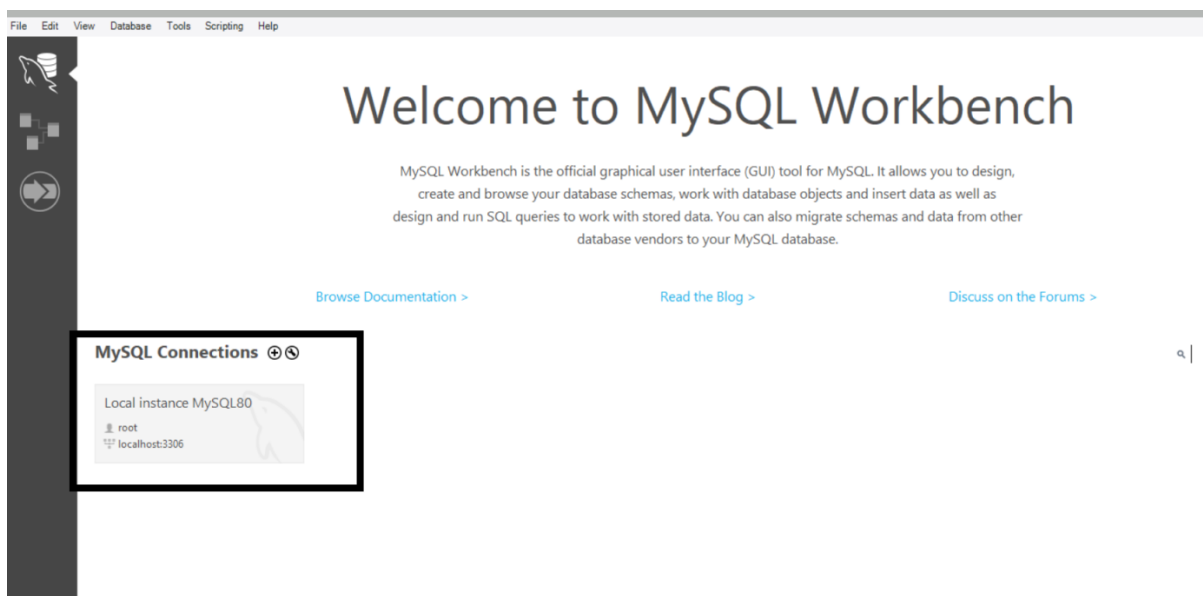


- Keep the default values for “Type and Networking” section





- Finish the installation by pressing next.
- Once the installation is complete, open the MYSQL workbench and click on the local instance connection. The window will look something like this:



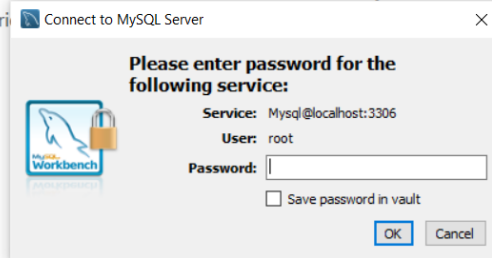
- Provide the password that you setup during installation:

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.

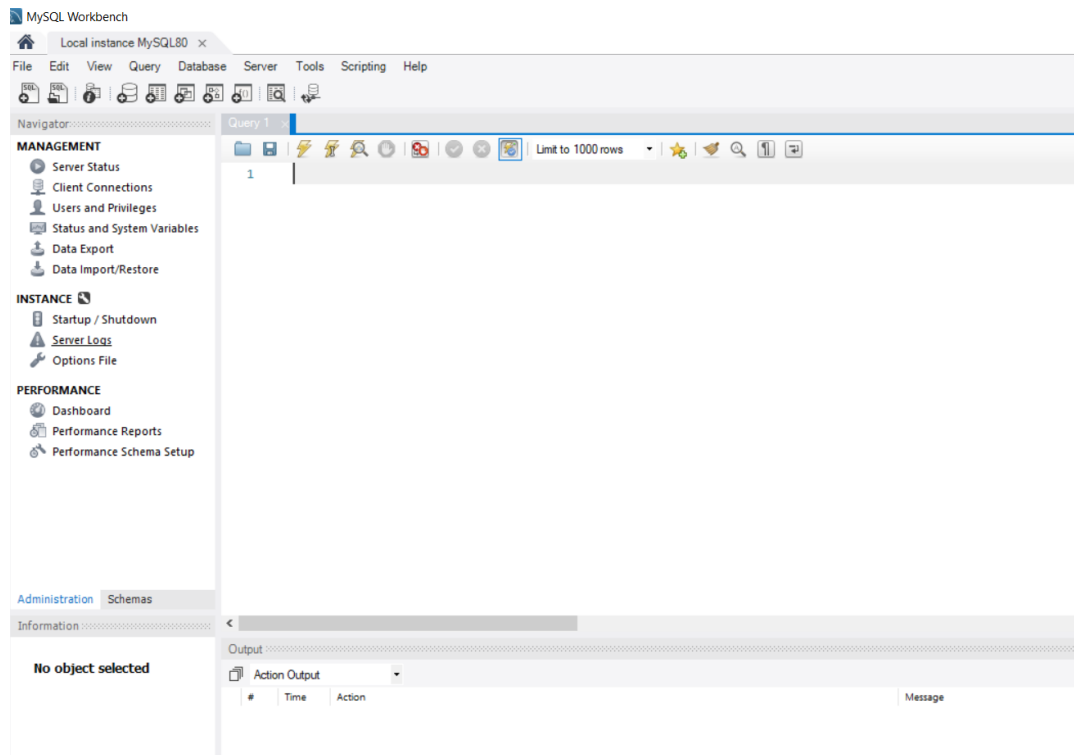
[Browse Documentation >](#)

[Discuss on the Forums >](#)



Remember the user name mentioned above as we will need it to connect python to Mysql.

- Once the connection is established, you will see a window like this:



Connecting MySQL With Python

Let's use python to connect with this database.

- First we need to install "mysql-connector-python" package to establish a connection with MySQL.

"pip install mysql-connector-python"

```
(pyMySQL) D:\Project_\PythonToMySQL>pip install mysql-connector-python
Collecting mysql-connector-python
  Using cached mysql_connector_python-8.0.19-cp36-cp36m-win_amd64.whl (4
Requirement already satisfied: protobuf==3.6.1 in c:\programdata\anacond
Requirement already satisfied: dnspython==1.16.0 in c:\programdata\anacond
Requirement already satisfied: six>=1.9 in c:\programdata\anaconda3\envs
Requirement already satisfied: setuptools in c:\programdata\anaconda3\en
Installing collected packages: mysql-connector-python
Successfully installed mysql-connector-python-8.0.19
```

- Once the package is installed, we can go ahead with establishing the connection.

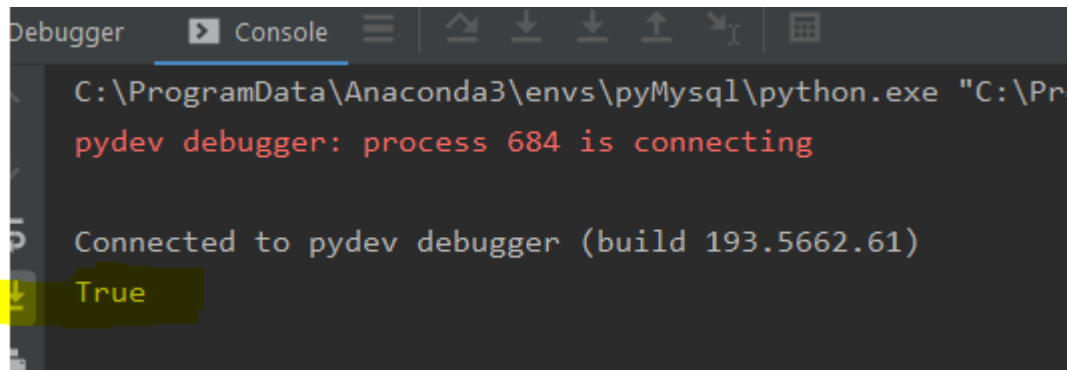
```
import mysql.connector as connection

try:
    mydb = connection.connect(host="localhost", user="root", passwd="password", use_pure=True)
    # check if the connection is established
    print(mydb.is_connected())
    mydb.close()
except Exception as e:
    print(str(e))
```

- Enter the details as shown above, your mysql server is running locally so **host** is "localhost", enter the **username** and **password** as was setup during installation.

Note: "use_pure" argument forces mysqlConnector to use pure python connection instead of C extensions which leads to SSL error.

- To check if the connection is established, we can use **print (mydb.is_connected)**. It will return **TRUE** if the connection is established else **FALSE**.



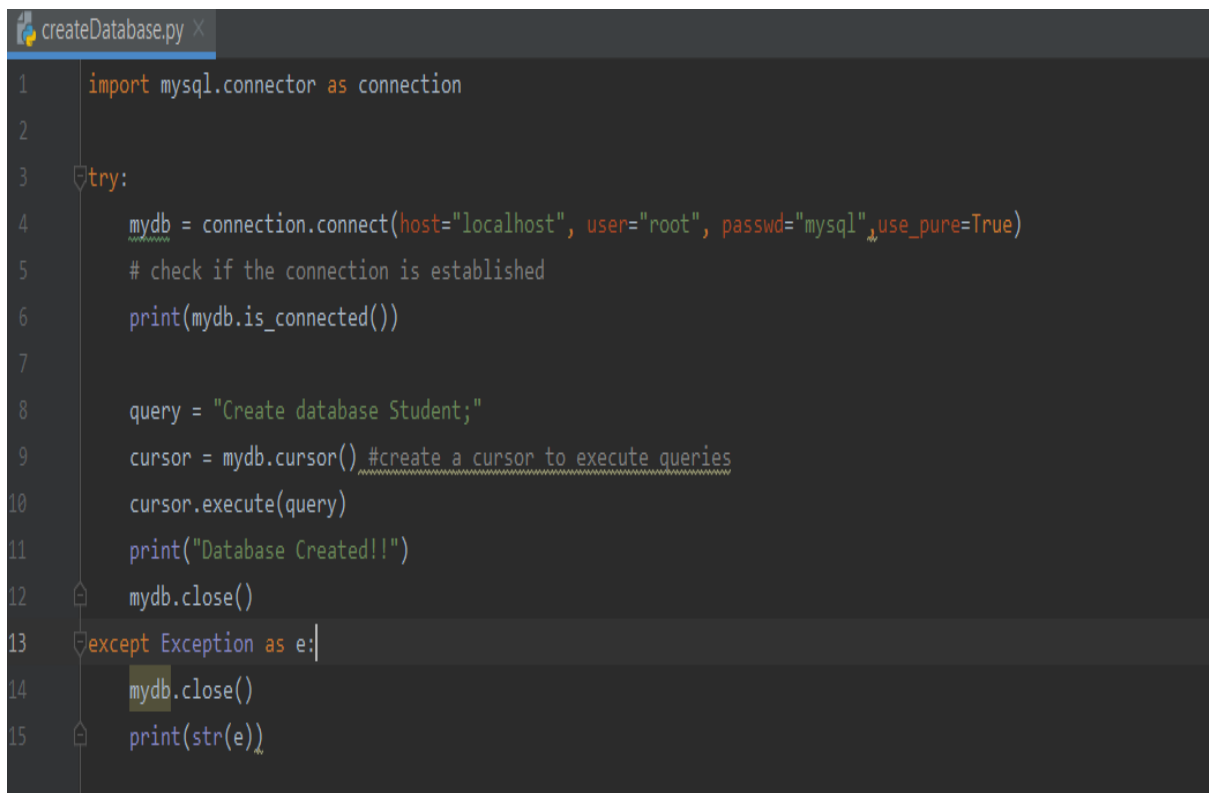
The screenshot shows a PyDev console window with the following text:

```
C:\ProgramData\Anaconda3\envs\pyMysql\python.exe "C:\Pr
pydev debugger: process 684 is connecting

Connected to pydev debugger (build 193.5662.61)
True
```

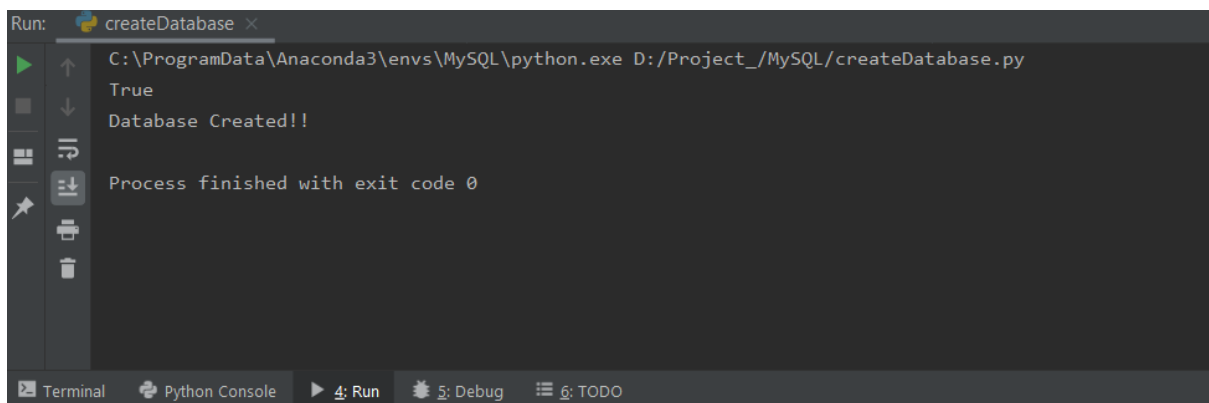
The word "True" is highlighted with a yellow background.

- Our connection is established now.
- Let's start with creating a database with name Student.



The screenshot shows a Python script named `createDatabase.py` with the following code:

```
1 import mysql.connector as connection
2
3 try:
4     mydb = connection.connect(host="localhost", user="root", passwd="mysql", use_pure=True)
5     # check if the connection is established
6     print(mydb.is_connected())
7
8     query = "Create database Student;"
9     cursor = mydb.cursor() #create a cursor to execute queries
10    cursor.execute(query)
11    print("Database Created!!")
12    mydb.close()
13 except Exception as e:
14     mydb.close()
15     print(str(e))
```



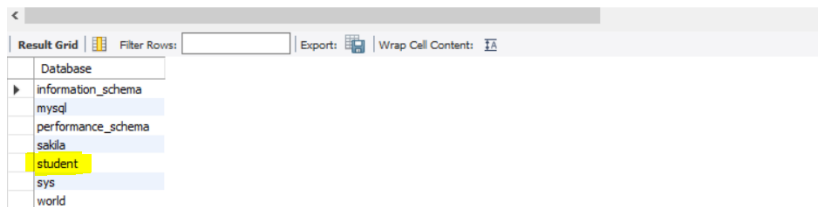
The screenshot shows the PyDev Run console with the following output:

```
Run: createDatabase
C:\ProgramData\Anaconda3\envs\MySQL\python.exe D:/Project_/MySQL/createDatabase.py
True
Database Created!!

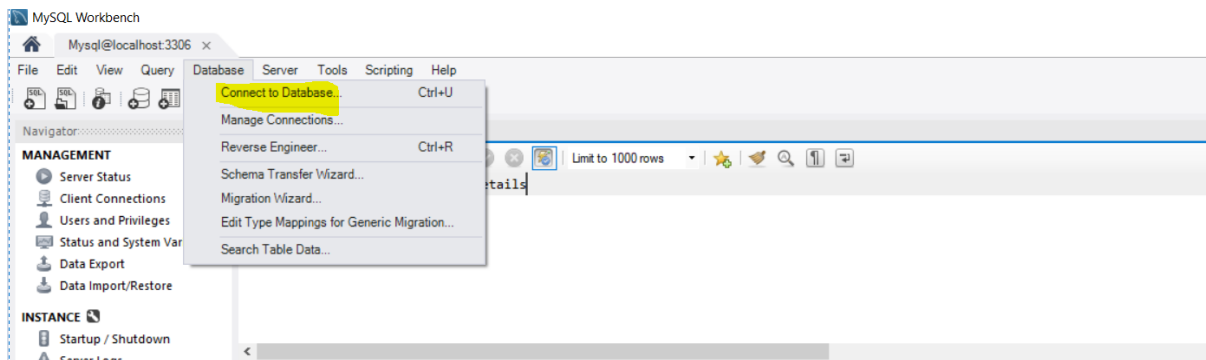
Process finished with exit code 0
```

The bottom of the window shows tabs for Terminal, Python Console, Run, Debug, and TODO.

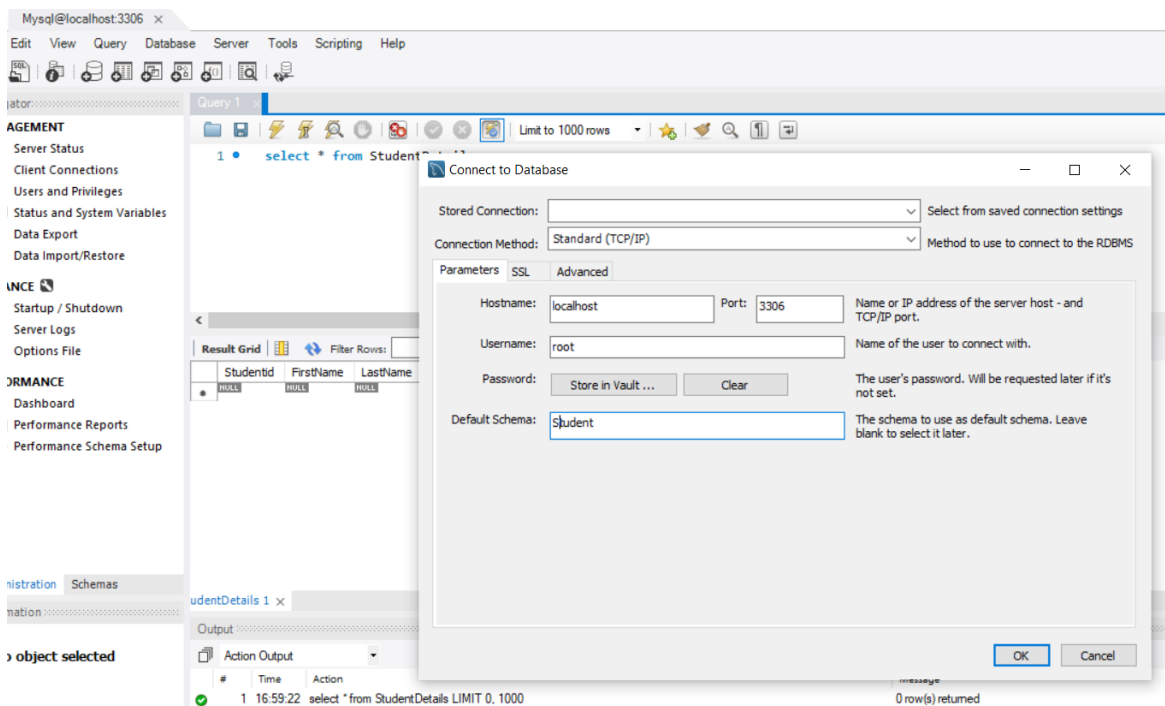
- Let's check if database is created in our MySQL Workbench.



- Great! Database is created.
- Let's start with creating tables. Let's first connect to the created database in our workbench so that we can view the tables, once we create them from python.



- Give the database name:



- Now we are connected to the created database. Let's start with creating tables.

```

createDatabase.py x createTableInDB.py x
import mysql.connector as connection

try:
    mydb = connection.connect(host="localhost", database='Student', user="root", passwd="mysql", use_pure=True)
    # check if the connection is established
    print(mydb.is_connected())

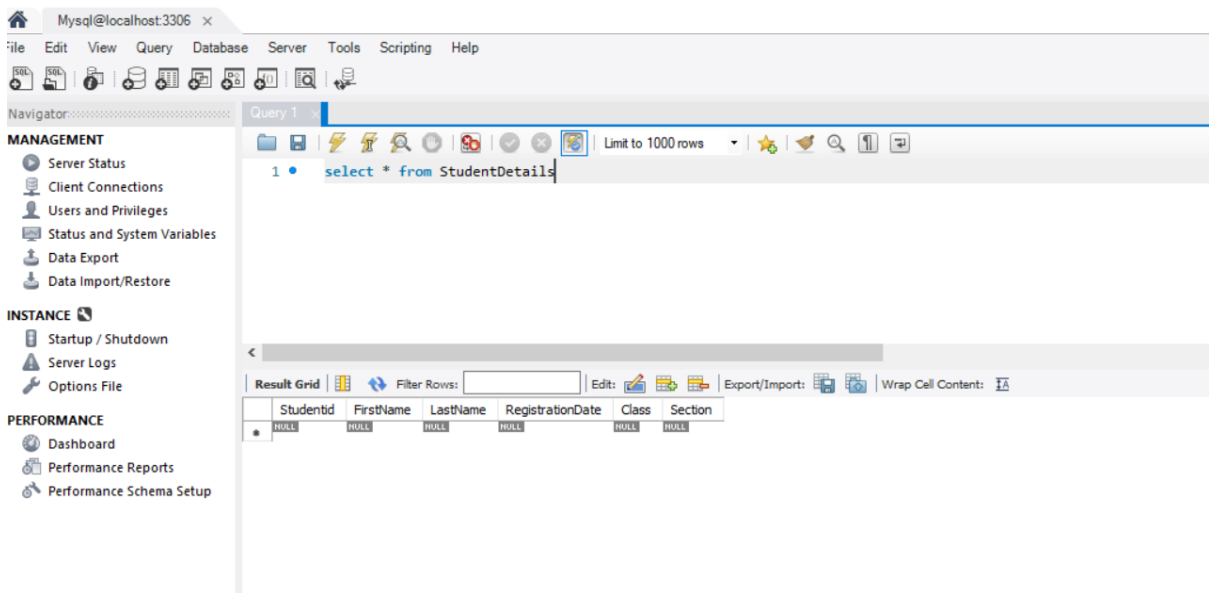
    query = "CREATE TABLE StudentDetails (Studentid INT(10) AUTO_INCREMENT PRIMARY KEY, FirstName VARCHAR(60), \
        LastName VARCHAR(60), RegistrationDate DATE, Class Varchar(20), Section Varchar(10))"

    cursor = mydb.cursor() #create a cursor to execute queries
    cursor.execute(query)
    print("Table Created!!")
    mydb.close()
except Exception as e:
    mydb.close()
    print(str(e))

try

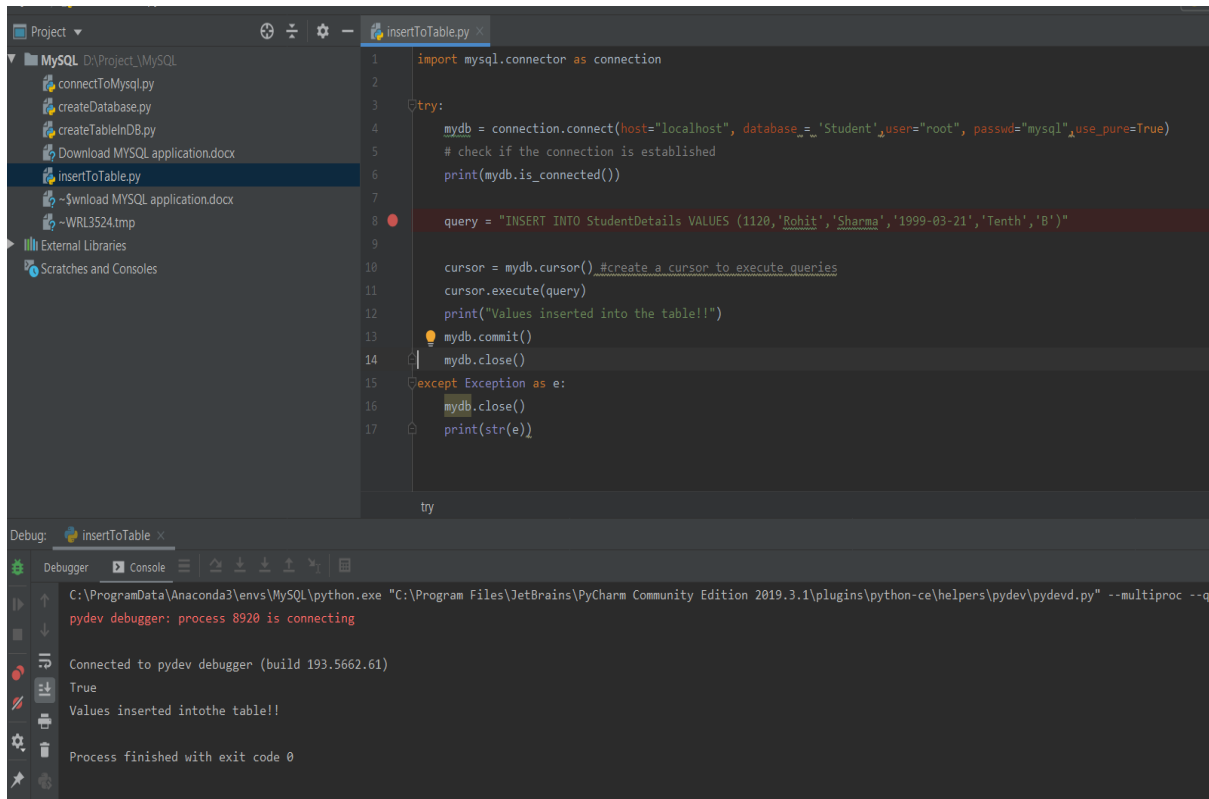
```

- We need to pass an **additional parameter, database name, while connecting to server**. We have passed "Student" database in which we are going to create the table.
- Let's see if the table is connected in our Mysql.



- Our table is now created in the mentioned database.

- Let's start with inserting values in our table:



```
1 import mysql.connector as connection
2
3 try:
4     mydb = connection.connect(host="localhost", database='Student', user="root", passwd="mysql", use_pure=True)
5     # check if the connection is established
6     print(mydb.is_connected())
7
8     query = "INSERT INTO StudentDetails VALUES (1120,'Rohit','Sharma','1999-03-21','Tenth','B')"
9
10    cursor = mydb.cursor() #create a cursor to execute queries
11    cursor.execute(query)
12    print("Values inserted into the table!!")
13    mydb.commit()
14    mydb.close()
15 except Exception as e:
16     mydb.close()
17     print(str(e))
18
19 try
```

Debugger: insertToTable x

Debugger Console

C:\ProgramData\Anaconda3\envs\MySQL\python.exe "C:\Program Files\JetBrains\PyCharm Community Edition 2019.3.1\plugins\python-ce\helpers\pydev\pydevd.py" --multiproc --q

pydev debugger: process 8920 is connecting

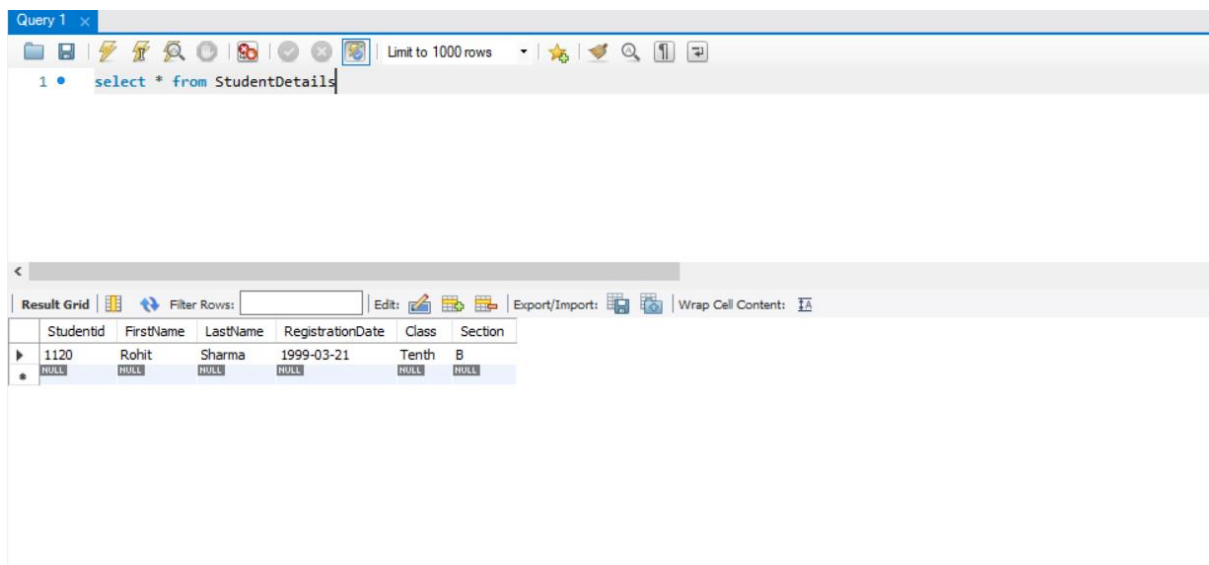
Connected to pydev debugger (build 193.5662.61)

True

Values inserted into the table!!

Process finished with exit code 0

- Let's check if the values are inserted:



Query 1 x

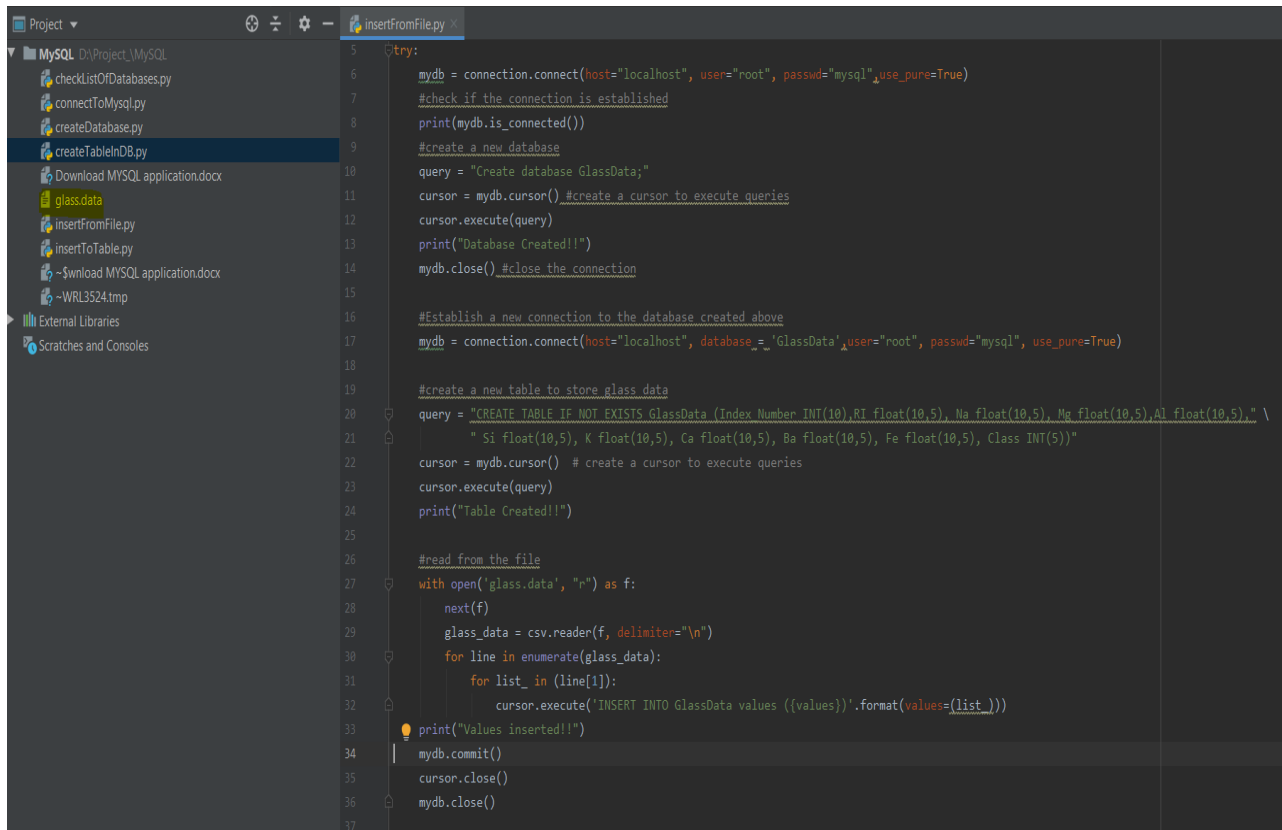
Limit to 1000 rows

1 • select * from StudentDetails

Result Grid

	StudentId	FirstName	LastName	RegistrationDate	Class	Section
▶	1120	Rohit	Sharma	1999-03-21	Tenth	B
*	NULL	NULL	NULL	NULL	NULL	NULL

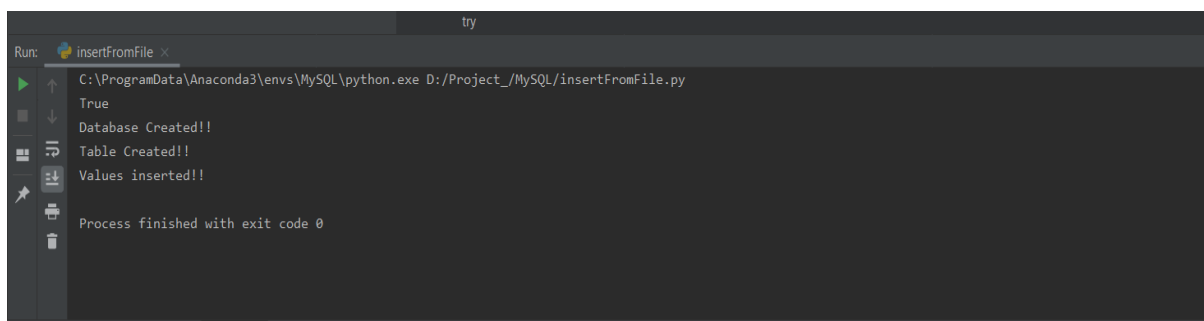
- Let's now **insert values into a new table in a new database from a file**.
- We are loading all the values in the file "glass. Data" into our table.
- We created a new database named "Glass Data" and a new table "Glass Data" in it.
- We are reading each row from the file and inserting into the table.



```

1 try:
2     mydb = connection.connect(host="localhost", user="root", passwd="mysql", use_pure=True)
3     #check if the connection is established
4     print(mydb.is_connected())
5     #create a new database
6     query = "Create database GlassData;"
7     cursor = mydb.cursor() #create a cursor to execute queries
8     cursor.execute(query)
9     print("Database Created!!")
10    mydb.close() #close the connection
11
12    #Establish a new connection to the database created above
13    mydb = connection.connect(host="localhost", database="GlassData", user="root", passwd="mysql", use_pure=True)
14
15    #create a new table to store glass data
16    query = "CREATE TABLE IF NOT EXISTS GlassData (Index Number INT(10), RI float(10,5), Na float(10,5), Mg float(10,5), Al float(10,5), " \
17            " Si float(10,5), K float(10,5), Ca float(10,5), Ba float(10,5), Fe float(10,5), Class INT(5))"
18    cursor = mydb.cursor() # create a cursor to execute queries
19    cursor.execute(query)
20    print("Table Created!!")
21
22    #read from the file
23    with open('glass.data', "r") as f:
24        next(f)
25        glass_data = csv.reader(f, delimiter="\n")
26        for line in enumerate(glass_data):
27            for list in (line[1]):
28                cursor.execute("INSERT INTO GlassData values ({values})".format(values=(list)))
29    print("Values inserted!!")
30    mydb.commit()
31    cursor.close()
32    mydb.close()
33

```



```

Run: insertFromFile x
try
C:\ProgramData\Anaconda3\envs\MySQL\python.exe D:/Project/_MySQL/insertFromFile.py
True
Database Created!!
Table Created!!
Values inserted!!
Process finished with exit code 0

```

- Let's check if the values are inserted in the table.

Query 1

Limit to 1000 rows

1 select * from GlassData.GlassData

Result Grid

Index_Number	RI	Na	Mg	Al	Si	K	Ca	Ba	Fe	Class
1	1.52101	13.64000	4.49000	1.10000	71.78000	0.06000	8.75000	0.00000	0.00000	1
2	1.51761	13.89000	3.60000	1.36000	72.73000	0.48000	7.83000	0.00000	0.00000	1
3	1.51618	13.53000	3.55000	1.54000	72.99000	0.39000	7.78000	0.00000	0.00000	1
4	1.51766	13.21000	3.69000	1.29000	72.61000	0.57000	8.22000	0.00000	0.00000	1
5	1.51742	13.27000	3.62000	1.24000	73.08000	0.55000	8.07000	0.00000	0.00000	1
6	1.51596	12.79000	3.61000	1.62000	72.97000	0.64000	8.07000	0.00000	0.26000	1
7	1.51743	13.30000	3.60000	1.14000	73.09000	0.58000	8.17000	0.00000	0.00000	1
8	1.51756	13.15000	3.61000	1.05000	73.24000	0.57000	8.24000	0.00000	0.00000	1
9	1.51918	14.04000	3.58000	1.37000	72.08000	0.56000	8.30000	0.00000	0.00000	1
10	1.51755	13.00000	3.60000	1.36000	72.99000	0.57000	8.40000	0.00000	0.11000	1
11	1.51571	12.72000	3.46000	1.56000	73.20000	0.67000	8.09000	0.00000	0.24000	1

GlassData 9

Read Only

- The values are inserted.

Let's see some other commands:

1) Selecting from table

```

Project
  MySQL D:\Project\MySQL
  checkListOfDatabases.py
  connectToMySQL.py
  createDatabase.py
  createTableInDB.py
  Download MySQL application.docx
  glass.data
  insertFromFile.py
  insertToTable.py
  selectFromDbIntoDataframe.py
  -download MySQL application.docx
  -WRL3524.tmp
  External Libraries
  Scratches and Consoles

insertFromFile.py
selectFromDbIntoDataframe.py
1 import mysql.connector as connection
2 import pandas as pandas
3
4 try:
5     mydb = connection.connect(host="localhost", database='GlassData', user="root", passwd="mysql", use_pure=True)
6     #check if the connection is established
7     print(mydb.is_connected())
8     query = "Select * from GlassData;"
9     cursor = mydb.cursor() #create a cursor to execute queries
10    cursor.execute(query)
11    for result in cursor.fetchall():
12        print(result)
13    mydb.close() #close the connection
14
15 except Exception as e:
16     #mydb.close()
17     print(str(e))
  
```

Result:

```
Run C:\ProgramData\Anaconda3\envs\MySQL\python.exe D:/Project_/MySQL/selectFromDbIntoDataframe.py
True
(1, 1.52101, 13.64, 4.49, 1.1, 71.78, 0.06, 8.75, 0.0, 0.0, 1)
(2, 1.51761, 13.89, 3.6, 1.36, 72.73, 0.48, 7.83, 0.0, 0.0, 1)
(3, 1.51618, 13.53, 3.55, 1.54, 72.99, 0.39, 7.78, 0.0, 0.0, 1)
(4, 1.51766, 13.21, 3.69, 1.29, 72.61, 0.57, 8.22, 0.0, 0.0, 1)
(5, 1.51742, 13.27, 3.62, 1.24, 73.08, 0.55, 8.07, 0.0, 0.0, 1)
(6, 1.51596, 12.79, 3.61, 1.62, 72.97, 0.64, 8.07, 0.0, 0.26, 1)
(7, 1.51743, 13.3, 3.6, 1.14, 73.09, 0.58, 8.17, 0.0, 0.0, 1)
(8, 1.51756, 13.15, 3.61, 1.05, 73.24, 0.57, 8.24, 0.0, 0.0, 1)
(9, 1.51918, 14.04, 3.58, 1.37, 72.08, 0.56, 8.3, 0.0, 0.0, 1)
(10, 1.51759, 13.0, 3.6, 1.36, 72.99, 0.57, 8.4, 0.0, 0.11, 1)
(11, 1.51571, 12.72, 3.46, 1.56, 73.2, 0.67, 8.09, 0.0, 0.24, 1)
(12, 1.51763, 12.8, 3.66, 1.27, 73.01, 0.6, 8.56, 0.0, 0.0, 1)
(13, 1.51589, 12.88, 3.43, 1.4, 73.28, 0.69, 8.05, 0.0, 0.24, 1)
(14, 1.51748, 12.86, 3.56, 1.27, 73.21, 0.54, 8.38, 0.0, 0.17, 1)
(15, 1.51763, 12.61, 3.59, 1.31, 73.29, 0.58, 8.5, 0.0, 0.0, 1)
(16, 1.51761, 12.81, 3.54, 1.23, 73.24, 0.58, 8.39, 0.0, 0.0, 1)
(17, 1.51784, 12.68, 3.67, 1.16, 73.11, 0.61, 8.7, 0.0, 0.0, 1)
(18, 1.52196, 14.36, 3.85, 0.89, 71.36, 0.15, 9.15, 0.0, 0.0, 1)
(19, 1.51911, 13.9, 3.73, 1.18, 72.12, 0.06, 8.89, 0.0, 0.0, 1)
(20, 1.51735, 13.02, 3.54, 1.69, 72.73, 0.54, 8.44, 0.0, 0.07, 1)
(21, 1.5175, 12.82, 3.55, 1.49, 72.75, 0.54, 8.52, 0.0, 0.19, 1)
(22, 1.51966, 14.77, 3.75, 0.29, 72.02, 0.03, 9.0, 0.0, 0.0, 1)
(23, 1.51736, 12.78, 3.62, 1.29, 72.79, 0.59, 8.7, 0.0, 0.0, 1)
(24, 1.51751, 12.81, 3.57, 1.35, 73.02, 0.62, 8.59, 0.0, 0.0, 1)
(25, 1.5172, 13.38, 3.5, 1.15, 72.85, 0.5, 8.43, 0.0, 0.0, 1)
(26, 1.51764, 12.98, 3.54, 1.21, 73.0, 0.65, 8.53, 0.0, 0.0, 1)
(27, 1.51793, 13.21, 3.48, 1.41, 72.64, 0.59, 8.43, 0.0, 0.0, 1)
```

- Let's try to store all the select values into a Dataframe.

```
MySQL selectIntoDataframe.py
Project
  MySQL D:\Project_\MySQL
    checkListOfDatabases.py
    connectToMySQL.py
    createDatabase.py
    createTableInDB.py
    Download MYSQL application.docx
    glass.data
    insertFromFile.py
    insertToTable.py
    selectFromDb.py
    selectIntoDataframe.py
    ~$wload MYSQL application.docx
    ~WRL3524.tmp
  External Libraries
  Scratches and Consoles

1 import mysql.connector as connection
2 import pandas as pandas
3
4 try:
5
6     mydb = connection.connect(host="localhost", database='GlassData', user="root", passwd="mysql", use_pure=True)
7     # check if the connection is established
8     print(mydb.is_connected())
9     query = "Select * from GlassData;"
10    result_dataframe = pandas.read_sql(query, mydb)
11    print(result_dataframe)
12
13    mydb.close() # close the connection
14
15 except Exception as e:
16     #mydb.close()
17     print(str(e))
```

We can use **pandas.read_sql** to store the values in a dataframe.

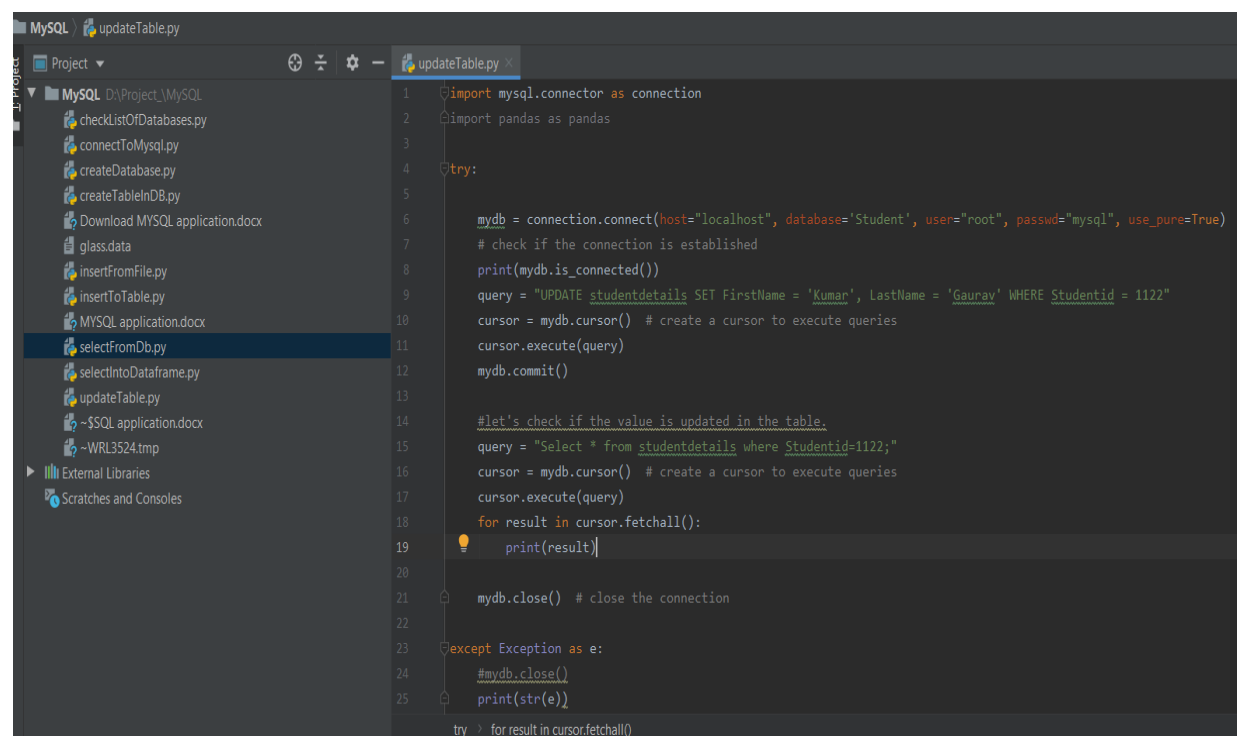
Let's see the result:

```
True
  Index_Number    RI    Na    Mg    Al    ...    K    Ca    Ba    Fe    Class
0             1  1.52101  13.64  4.49  1.10  ...  0.06  8.75  0.00  0.0    1
1             2  1.51761  13.89  3.60  1.36  ...  0.48  7.83  0.00  0.0    1
2             3  1.51618  13.53  3.55  1.54  ...  0.39  7.78  0.00  0.0    1
3             4  1.51766  13.21  3.69  1.29  ...  0.57  8.22  0.00  0.0    1
4             5  1.51742  13.27  3.62  1.24  ...  0.55  8.07  0.00  0.0    1
..          ...    ...    ...    ...    ...    ...    ...    ...    ...    ...
209          210  1.51623  14.14  0.00  2.88  ...  0.08  9.18  1.06  0.0    7
210          211  1.51685  14.92  0.00  1.99  ...  0.00  8.40  1.59  0.0    7
211          212  1.52065  14.36  0.00  2.02  ...  0.00  8.44  1.64  0.0    7
212          213  1.51651  14.38  0.00  1.94  ...  0.00  8.48  1.57  0.0    7
213          214  1.51711  14.23  0.00  2.08  ...  0.00  8.62  1.67  0.0    7

[214 rows x 11 columns]

Process finished with exit code 0
```

2) Update statement



```
MySQL updateTable.py
Project
  MySQL D:\Project_MySQL
    checkListOfDatabases.py
    connectToMySQL.py
    createDatabase.py
    createTableInDB.py
    Download MySQL application.docx
    glass.data
    insertFromFile.py
    insertToTable.py
    MySQL application.docx
    selectFromDb.py
    selectIntoDataframe.py
    updateTable.py
    ~$SQL application.docx
    ~WRL3524.tmp
  External Libraries
  Scratches and Consoles

1 import mysql.connector as connection
2 import pandas as pandas
3
4 try:
5
6     mydb = connection.connect(host="localhost", database='Student', user="root", passwd="mysql", use_pure=True)
7     # check if the connection is established
8     print(mydb.is_connected())
9     query = "UPDATE studentdetails SET FirstName = 'Kumar', LastName = 'Gaurav' WHERE Studentid = 1122"
10    cursor = mydb.cursor() # create a cursor to execute queries
11    cursor.execute(query)
12    mydb.commit()
13
14    #let's check if the value is updated in the table.
15    query = "Select * from studentdetails where Studentid=1122;"
16    cursor = mydb.cursor() # create a cursor to execute queries
17    cursor.execute(query)
18    for result in cursor.fetchall():
19        print(result)
20
21    mydb.close() # close the connection
22
23 except Exception as e:
24     #mydb.close()
25     print(str(e))
26
27 try > for result in cursor.fetchall()
```

```
Run: updateTable x
C:\ProgramData\Anaconda3\envs\MySQL\python.exe D:/Project_/MySQL/updateTable.py
True
(1122, 'Kumar', 'Gaurav', datetime.date(1999, 6, 11), 'Ninth', 'B')

Process finished with exit code 0
```

- Let's check in MySQL workbench:

Query 1 x

Limit to 1000 rows

1 • select * from studentdetails

Result Grid

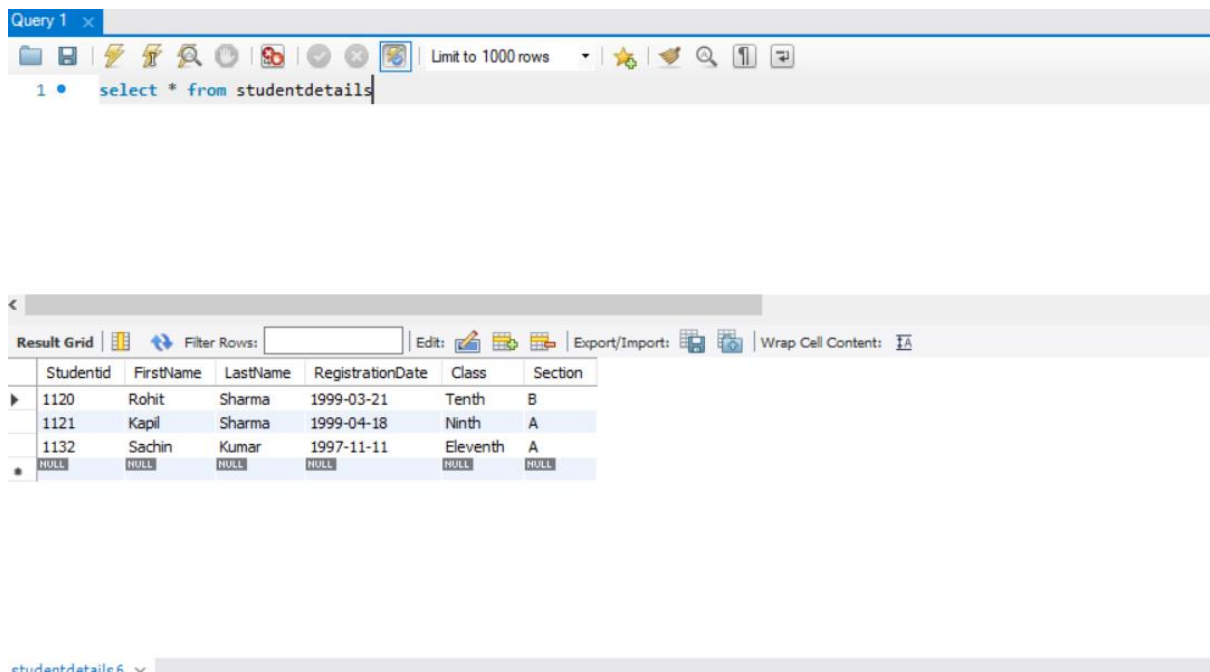
	Studentid	FirstName	LastName	RegistrationDate	Class	Section
▶	1120	Rohit	Sharma	1999-03-21	Tenth	B
	1121	Kapil	Sharma	1999-04-18	Ninth	A
	1122	Kumar	Gaurav	1999-06-11	Ninth	B
	1132	Sachin	Kumar	1997-11-11	Eleventh	A
*	NULL	NULL	NULL	NULL	NULL	NULL

3) Delete statement

```
MySQL deleteFromTable.py
Project
MySQL D:\Project_MySQL
  checkListOfDatabases.py
  connectToMySQL.py
  createDatabase.py
  createTableInDB.py
  deleteFromTable.py
  Download MySQL application.docx
  glass.data
  insertFromFile.py
  insertToTable.py
  MySQL application.docx
  selectFromDb.py
  selectIntoDataframe.py
  updateTable.py
  ~$SQL application.docx
  ~WRL3524.tmp
External Libraries
Scratches and Consoles

1 import mysql.connector as connection
2
3 try:
4
5     mydb = connection.connect(host="localhost", database='Student', user="root", passwd="mysql", use_pure=True)
6     # check if the connection is established
7     print(mydb.is_connected())
8     query = "DELETE FROM studentdetails WHERE Studentid = 1122"
9     cursor = mydb.cursor() # create a cursor to execute queries
10    cursor.execute(query)
11    mydb.commit()
12
13    #let's check if the value is updated in the table.
14    query = "Select * from studentdetails where Studentid=1122;"
15    cursor = mydb.cursor() # create a cursor to execute queries
16    cursor.execute(query)
17    for result in cursor.fetchall():
18        print(result)
19
20    mydb.close() # close the connection
21
22 except Exception as e:
23     #mydb.close()
24     print(str(e))
```

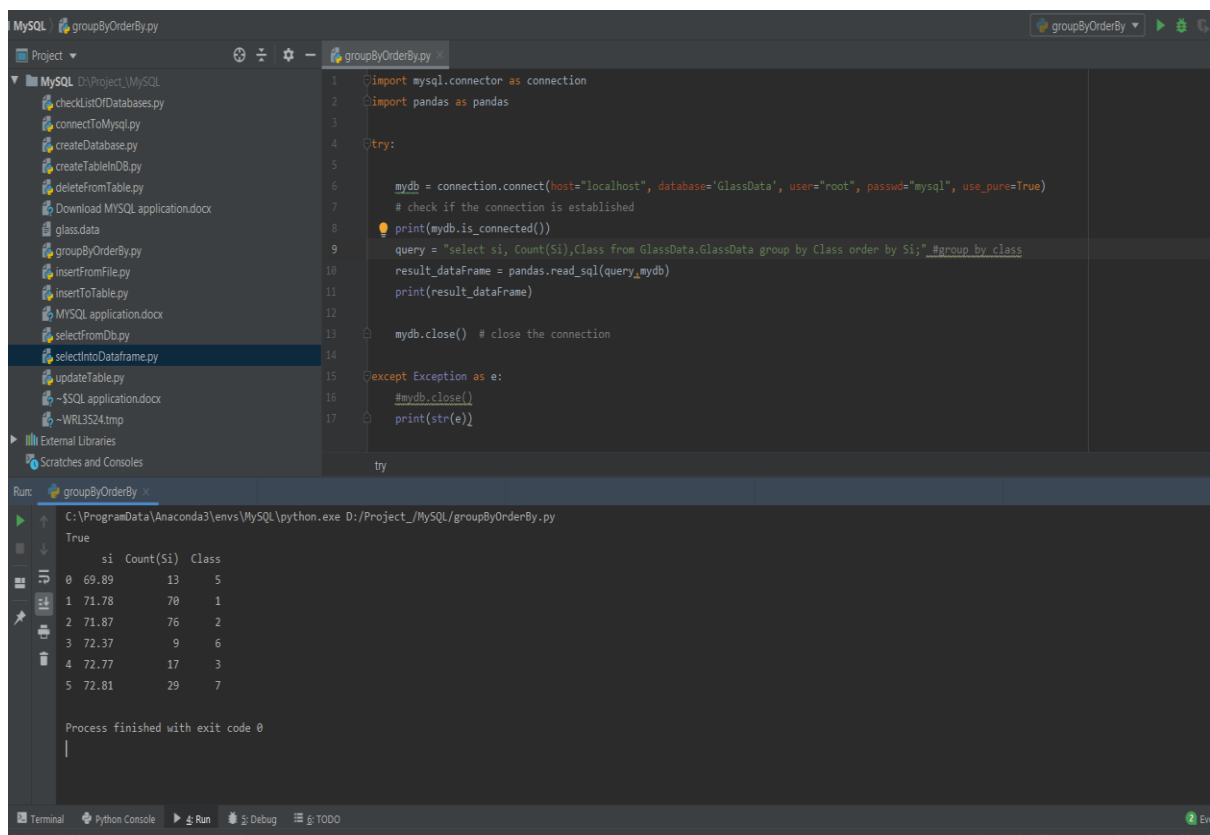
- Let's check in MySQL workbench:



The screenshot shows the MySQL Workbench interface. At the top, the 'Query 1' tab is active, displaying the SQL query: `select * from studentdetails`. Below the query editor, the 'Result Grid' is visible, showing the results of the query. The grid has columns for StudentId, FirstName, LastName, RegistrationDate, Class, and Section. The data is as follows:

StudentId	FirstName	LastName	RegistrationDate	Class	Section
1120	Rohit	Sharma	1999-03-21	Tenth	B
1121	Kapil	Sharma	1999-04-18	Ninth	A
1132	Sachin	Kumar	1997-11-11	Eleventh	A
NULL	NULL	NULL	NULL	NULL	NULL

4) Group by, Order by



The screenshot shows an IDE with a Python script named `groupByOrderBy.py`. The script connects to a MySQL database, executes a group by and order by query, and displays the results in a console window.

```

1 import mysql.connector as connection
2 import pandas as pandas
3
4 try:
5
6     mydb = connection.connect(host="localhost", database='GlassData', user="root", passwd="mysql", use_pure=True)
7     # check if the connection is established
8     print(mydb.is_connected())
9     query = "select si, Count(Si),Class from GlassData.GlassData group by Class order by Si;"
10    result_dataframe = pandas.read_sql(query,mydb)
11    print(result_dataframe)
12
13    mydb.close() # close the connection
14
15 except Exception as e:
16     #mydb.close()
17     print(str(e))
18
19 try
  
```

The console output shows the results of the query:

```

True
  si  Count(Si)  Class
0  69.89      13     5
1  71.78      70     1
2  71.87      76     2
3  72.37       9     6
4  72.77      17     3
5  72.81      29     7
  
```

Process finished with exit code 0

