Experiment No:6

Aim: To demonstrate CRUD(create,read,update,delete)operation on database using python.

Theory:

Python can be used to connect the Database.

MySQL is one of the most popular Databases.
Steps to work with the MySQL using Python.

1. Install MySQL Driver
2. Create a connection Object
3. Create a cursor Object
4. Execute the Query

Install MySQL Driver

1. python -m pip install mysql-connector-python

Create a Connection Object

The mysql.connector provides the **connect()** method used to create a connection between the MySQL database and the Python application. The syntax is given below.

Syntax:

1. Conn_obj= mysql.connector.connect(host = <hostname>, user = <username>, passwd = <password>,database=<database>)

Create a Cursor Object

The connection object is necessary to create because it provides the multiple working environments the same connection to the database. The **cursor()** function is used to create the cursor object. It is import for executing the SQL queries. The syntax is given below.

Syntax:

1. cursorobj= conn.cursor()

Execute the Query

Use the execute() method of the cursor object to execute the

query Cursorobj.execute(SQL statement)

Methods

Following are the various methods provided by the Cursor
class/object. 1 callproc() :

2 close():

3 Info():

4 executemany():

5 execute():

6 fetchall()

7 fetchone()

8 fetchmany()

9 etchwarnings()

Properties

Following are the properties of the Cursor class –

1 column_names

2 description

3 lastrowid

4 rowcount

5 statement

PROGRAM

1. To create a database

```
import mysql.connector

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

mycursor = mydb.cursor()

mycursor.execute("CREATE DATABASE mydatabase")

#If this page is executed with no error, you have successfully created a database.
```

2.**To display Databases**

```
import mysql.connector

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

mycursor = mydb.cursor()

mycursor.execute("SHOW DATABASES")

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

OUTPUT

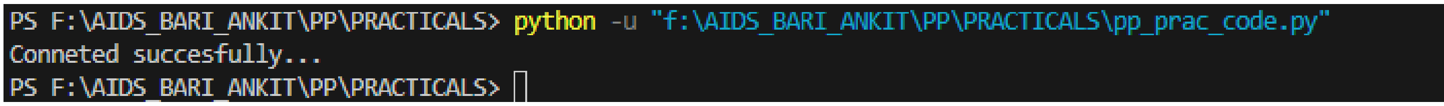
```
('information_scheme',)
('mydatabase',)
('performance_schema',)
('sys',)
```

3.Create table and insert values and update, delete and read the contents.

```
import mysql.connector

mydb = mysql.connector.connect(
 host="localhost",
 user="yourusername",
 password="yourpassword",
 database="mydatabase"
)

mycursor = mydb.cursor()

mycursor.execute("CREATE TABLE customers (id INT AUTO_INCREMENT PRIMARY
KEY, name VARCHAR(255), address VARCHAR(255))")


mycursor = mydb.cursor()

sql = "INSERT INTO customers (name, address) VALUES (%s,
%s)" val = [
  ('Peter', 'Lowstreet 4'),
 ('Amy', 'Apple st 652'),
  ('Hannah', 'Mountain 21'),
 ('Michael', 'Valley 345'),
  ('Sandy', 'Ocean blvd 2'),
 ('Betty', 'Green Grass 1'),
  ('Richard', 'Sky st 331'),
 ('Susan', 'One way 98'),
  ('Vicky', 'Yellow Garden 2'),
 ('Ben', 'Park Lane 38'),
 ('William', 'Central st 954'),
 ('Chuck', 'Main Road 989'),
  ('Viola', 'Sideway 1633')
]

mycursor.executemany(sql, val)

mydb.commit()

print(mycursor.rowcount, "was inserted.")

sql1 = "SELECT * FROM customers WHERE address ='Park Lane

38'" mycursor.execute(sql1)
```

```
myresult = mycursor.fetchall()

for x in myresult:
  print(x)

sql2 = "UPDATE customers SET address = 'Canyon 123' WHERE

address = 'Valley345'" mycursor.execute(sql2)

mydb.commit()

print(mycursor.rowcount, "record(s) affected")


sql3 = " DELETE FROM customers WHERE
address = 'Mountain21'" mycursor.execute(sql3)

mydb.commit()

print(mycursor.rowcount, "record(s) deleted")
```

OUTPUT



```
PS F:\AIDS_BARI_ANKIT\PP\PRACTICALS> python -u "f:\AIDS_BARI_ANKIT\PP\PRACTICALS\pp_prac_code.py"
Conneted succesfully...
PS F:\AIDS_BARI_ANKIT\PP\PRACTICALS> 
```

Conclusion:

The experiment successfully showcased the fundamental CRUD operations - create, read, update, and delete - on a database using Python. Through systematic execution and analysis, it was evident that Python's intuitive syntax and powerful libraries offer efficient means to interact with databases, enabling seamless manipulation of data for various applications.