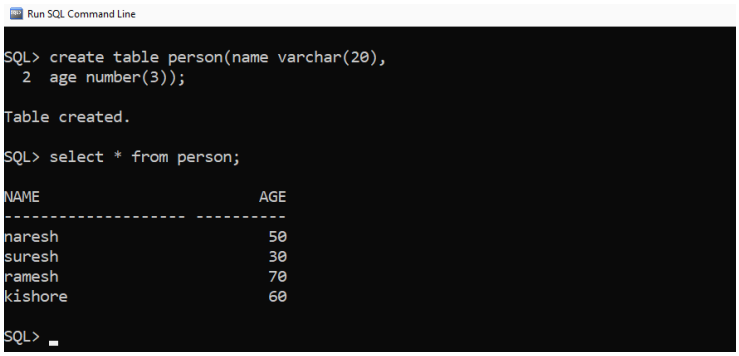
cursor.executemany(operation, seq_of_params)

This method prepares a database operation (query or command) and executes it against all parameter sequences or mappings found in the sequence seq_of_params


```
import cx_Oracle

data=[('naresh',50),
      ('suresh',30),
      ('ramesh',70),
      ('kishore',60)]

cn=cx_Oracle.connect(dsn="localhost:1521/XE",user="system",passw
ord="manager")
c=cn.cursor()
c.executemany("insert into person values(:1,:2)",data)
cn.commit()
cn.close()
```

Output



cursor.executescript(sql_script).
This routine executes multiple SQL statements at once provided in the form of script.

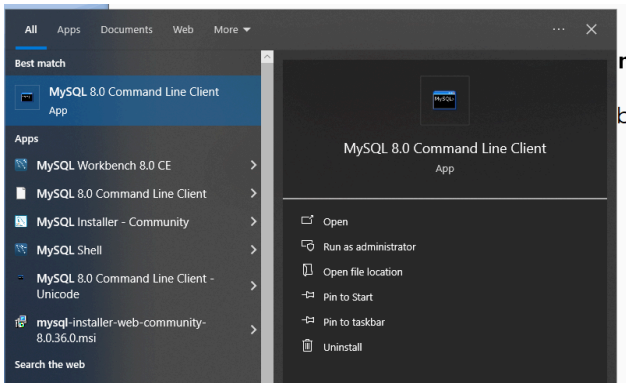
```
import cx_Oracle

cn=cx_Oracle.connect(dsn="localhost:1521/XE",user="system",passw
ord="manager")
c=cn.cursor()
c.executescript('''insert into person values('kiran',50);
update person set age=40 where name='suresh';
delete from person where name='kishore';'')

cn.commit()
cn.close()
```

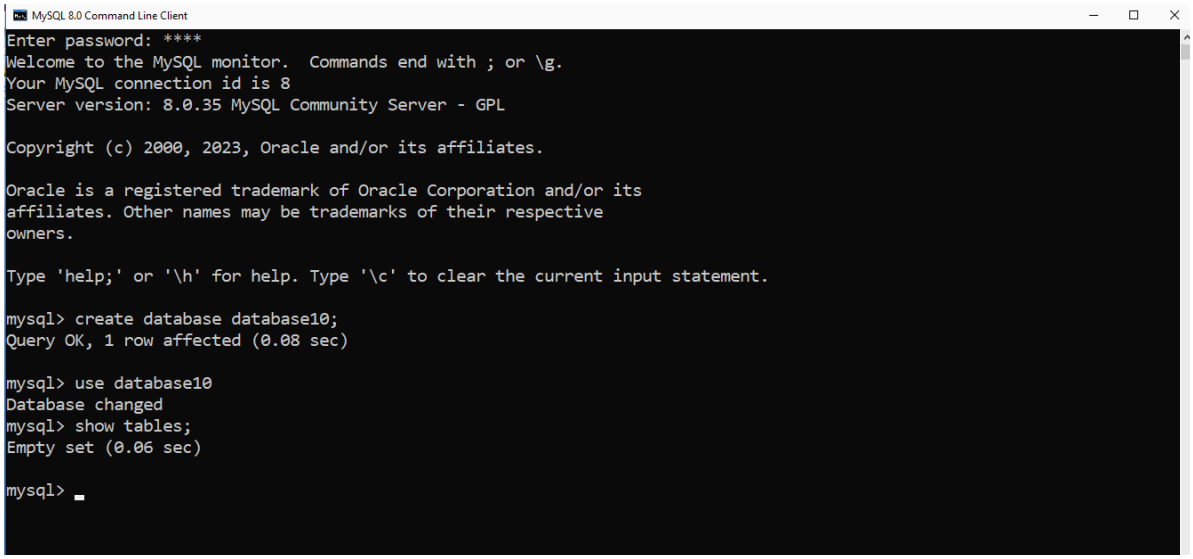
Output
AttributeError: 'cx_Oracle.Cursor' object has no attribute 'executescript'

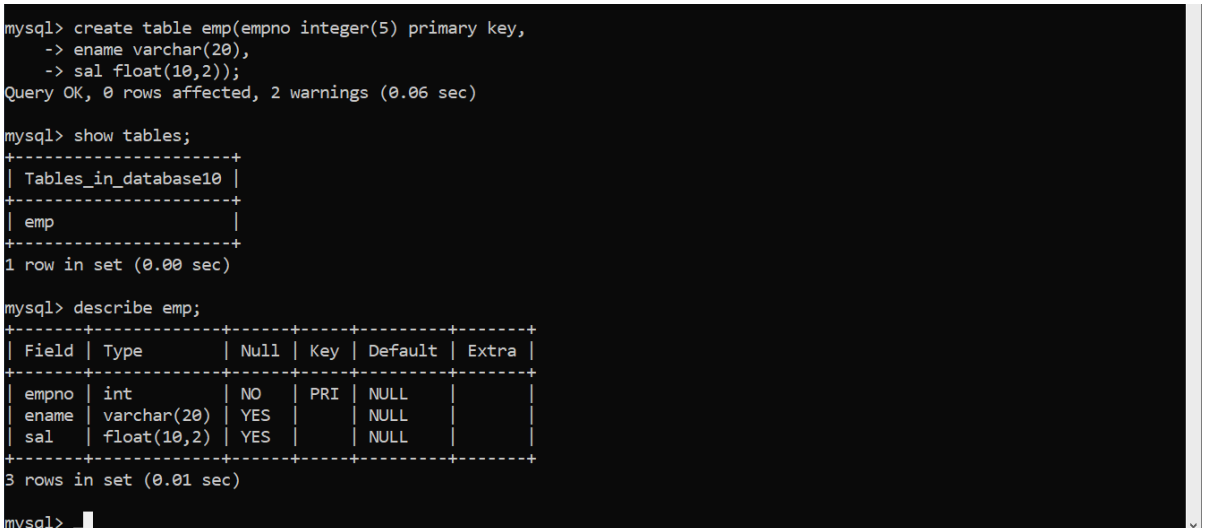
Python and MySQL Database Communication

1. Install Mysql Database Software
2. Install mysql-connector-python library

- Open mysql command line client







```
mysql> create table emp(empno integer(5) primary key,
    -> ename varchar(20),
    -> sal float(10,2));
Query OK, 0 rows affected, 2 warnings (0.06 sec)

mysql> show tables;
+----------------------+
| Tables_in_database10 |
+----------------------+
| emp                  |
+----------------------+
1 row in set (0.00 sec)

mysql> describe emp;
+-------+-------------+------+-----+---------+-------+
| Field | Type        | Null | Key | Default | Extra |
+-------+-------------+------+-----+---------+-------+
| empno | int         | NO   | PRI | NULL    |       |
| ename | varchar(20) | YES  |     | NULL    |       |
| sal   | float(10,2) | YES  |     | NULL    |       |
+-------+-------------+------+-----+---------+-------+
3 rows in set (0.01 sec)

mysql>
```

Basic steps for communicating with database

1. Establish Connection
2. Create Cursor
3. Send SQL statements
4. Read Results
5. Close Connection

Example of establishing connection to mysql database

```
import mysql.connector

cn=mysql.connector.connect(database="database10",user="root",password="root")
print("connection established...")
```

Output

connection established.

Example:

```
import mysql.connector

cn=mysql.connector.connect(database="database10",user="root",password="root")
```

```
print("connection established...")
c=cn.cursor()
while True:
    empno=int(input("EmployeeNo: "))
    ename=input("EmployeeName: ")
    sal=float(input("Salary: "))
    try:
        c.execute("insert into emp
values(%s,%s,%s)",params=[empno,ename,sal])
        k=c.rowcount
        if k==1:
            print("Employee details are inserted...")
            cn.commit()
    except:
        print("Error in inserting data")

    ans=input("Add another employee?")
    if ans=="no":
        break

cn.close()
```

Output

```
connection established...
EmployeeNo: 1
EmployeeName: naresh
Salary: 5000
Employee details are inserted...
Add another employee?yes
EmployeeNo: 2
EmployeeName: suresh
Salary: 9000
Employee details are inserted...
Add another employee?yes
```

EmployeeNo: 3
EmployeeName: kishore
Salary: 6000
Employee details are inserted...
Add another employee?yes
EmployeeNo: 4
EmployeeName: kiran
Salary: 8000
Employee details are inserted...
Add another employee?no

Example:

```
import mysql.connector


cn=mysql.connector.connect(database="database10",user="root",pas
sword="root")
c=cn.cursor()
c.execute("select * from emp")
employees=c.fetchall()
total=0
for emp in employees:
    print(emp)
    total=total+emp[2]

print("Total Salaries ",total)
```

Output

(1, 'naresh', 5000.0)
(2, 'suresh', 9000.0)
(3, 'kishore', 6000.0)
(4, 'kiran', 8000.0)
Total Salaries 28000.0

Python --- SQLLite Database Communicate

SQLite is a C library that provides a lightweight disk-based database that doesn't require a separate server process and allows accessing the database using a nonstandard variant of the SQL query language. Some applications can use SQLite for internal data storage. It's also possible to prototype an application using SQLite and then port the code to a larger database such as PostgreSQL or Oracle.

The sqlite3 module was written by Gerhard Häring. It provides an SQL interface compliant with the DB-API 2.0

Sqlite3 is a default module which comes with python software.

Example:

```
import sqlite3

cn=sqlite3.connect("database1")
print("connection")
c=cn.cursor()

c.execute("create table student(rollno,name,course)")
print("table created")
```

Output

connection
table created

Example:

```
import sqlite3

cn=sqlite3.connect("database1")
print("connection")
```

```
c=cn.cursor()

while True:
    rollno=int(input("Rollno "))
    name=input("Name ")
    course=input("Course")
    c.execute("insert into student values(?,?,?)",[rollno,name,course])
    k=c.rowcount
    if k==1:
        print("student details are inserted")
        cn.commit()
    ans=input("Add another student?")
    if ans=="no":
        break

cn.close()
```

Output

```
connection
Rollno 1
Name naresh
Coursepython
student details are inserted
Add another student?yes
Rollno 2
Name suresh
Coursejava
student details are inserted
Add another student?no
```