Creating General Query Action

Connecting to SQLite Database

- To use SQLite, we must import **sqlite3**
  import sqlite3
- Then create a connection using connect() method and pass the name of the database you want to access if there is a file with that name, it will open that file. Otherwise, Python will create a file with the given name.

  sqliteConnection = sqlite3.connect('gfg.db')
- After this, a cursor object is called to be capable to send commands to the SQL.

  cursor = sqliteConnection.cursor()

```
# Python code to demonstrate table creation and

# insertions with SQL

# importing module

import sqlite3



# connecting to the database

connection = sqlite3.connect("gfg.db")
```

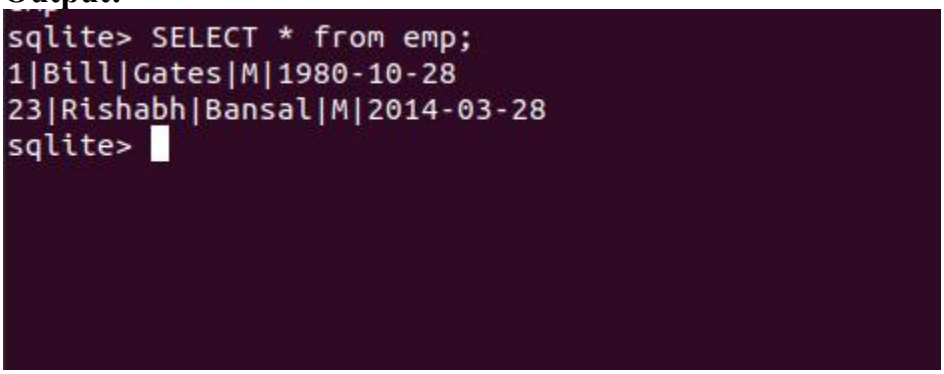
```
# cursor

crsr = connection.cursor()



# SQL command to insert the data in the table

sql_command = """INSERT INTO emp VALUES (23,
"Rishabh",\

"Bansal", "M", "2014-03-28");"""

crsr.execute(sql_command)



# another SQL command to insert the data in the table

sql_command = """INSERT INTO emp VALUES (1, "Bill",
"Gates",\

"M", "1980-10-28");"""

crsr.execute(sql_command)



# To save the changes in the files. Never skip this.

# If we skip this, nothing will be saved in the database.
```

connection.commit()

close the connection

connection.close()

Output:



Example 2: Inserting data input by the user

- Python3

importing module

=

gender = ['M', 'F', 'M', 'M', 'F']

Enter their joining data respectively

date = ['2019-08-24', '2020-01-01', '2018-05-14', '2015-02-02', '2018-05-14']

for i in range(5):

close the conne This is the q-mark style:

crsr.execute('INSERT INTO emp VALUES ({pk[i]}, "{f_name[i]}", "{l_name[i]}", "{gender[i]}", "{date[i]}")')

To save the changes in the files. Never skip this.

If we skip this, nothing will be saved in the database.

connection. #commit()

ction

connection.close()

Output:

Fetching Data

In this section, we have discussed how to create a table and how to add new rows in the database. Fetching the data from records is simple as inserting them. The execute method uses the SQL command of getting all the data from the table using “Select * from table_name” and all the table data can be fetched in an object in the form of a list of lists.

Example: Reading Data from sqlite3 table using Python

- Python

```
# importing the module

import sqlite3



# connect with the myTable database

connection = sqlite3.connect("gfg.db")



# cursor object

crsr = connection.cursor()



# execute the command to fetch all the data from the table emp

crsr.execute("SELECT * FROM emp")
```

```
# store all the fetched data in the ans variable

ans = crsr.fetchall()



# Since we have already selected all the data entries

# using the "SELECT *" SQL command and stored them in

# the ans variable, all we need to do now is to print

# out the ans variable

for i in ans:

    print(i)
```

Output:

```
(1, 'Bill', 'Gates', 'M', '1980-10-28')
(2, 'Nikhil', 'Aggarwal', 'M', '2019-08-24')
(3, 'Nisha', 'Rawat', 'F', '2020-01-01')
(4, 'Abhinav', 'Tomar', 'M', '2018-05-14')
(5, 'Raju', 'Kumar', 'M', '2015-02-02')
(6, 'Anshul', 'Aggarwal', 'F', '2018-05-14')
(23, 'Rishabh', 'Bansal', 'M', '2014-03-28')
```

Note: It should be noted that the database file that will be created will be in the same folder as that of the python file. If we wish to change the path of the file, change the path while opening the file.

Updating Data

For updating the data in the SQLite3 table we will use the UPDATE statement. We can update single columns as well as multiple columns using the UPDATE statement as per our requirement.

```
UPDATE table_name SET column1 = value1, column2 = value2,…
```

```
WHERE condition;
```

In the above syntax, the SET statement is used to set new values to the particular column, and the WHERE clause is used to select the rows for which the columns are needed to be updated.

Example: Updating SQLite3 table using Python

- Python3

```
# Import module

import sqlite3



# Connecting

('gfg.db')

 to sqlite

conn = sqlite3.connect

# Creating a cursor object using

# the cursor() method

cursor = conn.cursor()



# Updating
```

```
cursor.execute('"UPDATE emp SET lname = "Jyoti" WHERE
fname="Rishabh";"')
```

```
# Commit your changes in the database

conn.commit()

 # Closing the connection

conn.close()
```

Output: