

<u>Aim:</u> Understand how to create a SQLite database and perform basic **CRUD** (Create, Read, Update, Delete) operations using Python.

IDE:

SQLite3 can be integrated with Python using sqlite3 module. It provides an SQL interface compliant with the DB-API 2.0 specification described by PEP 249. You do not need to install this module separately because it is shipped by default along with Python version 2.5.x onwards. To use sqlite3 module, you must first create a connection object that represents the database and then optionally you can create a cursor object, which will help you in executing all the SQL statements.

Let's enhance the examples with a more practical use case, focusing on **Student Record Management**. We will simulate managing student_record by storing student data like their enrollment, **name**, subject, and mark in the database, and include additional operations like calculating the average mark.

Install sqlite-database

pip install sqlite-database

Database Setup

We'll set up an SQLite database to manage student record information.

Example

import sqlite3
Connect to database (or create it)
conn = sqlite3.connect('student_record.db')
Create a cursor object using the cursor() method
cursor = conn.cursor()

Create an Student Table

We'll create a student record table to store student details such as Enrollment, name, subject, and Mark.



Marwadi University Faculty of Engineering & Technology Department of Information and Communication Technology

Subject: Programming With Python (01CT1309)

Aim: Understand how to create an SQLite database and perform basic **CRUD** (Create, Read, Update, Delete) operations using Python.

Experiment No: 15

Enrollment No: 92400133037

Example

Date:

Commit the changes conn.commit()

Insert Student Data

Let's insert multiple students into the table.

Example

```
# Insert multiple employee records

student_record = [

(92301733016,'ASHUTOSH KUMAR SINGH','PWP',95),

(92301733017,'HARSH VISHALBHAI TRIVEDI','PWP',85),

(92301733027,'VIRAJ PRAKASHBHAI VAGHASIYA','PWP',90),

(92301733046,'SHIVAM ATULKUMAR BHATT', 'PWP',93),

(92301733058,'DEVENDRASINH DOLATSINH JADEJA','PWP',75)

]

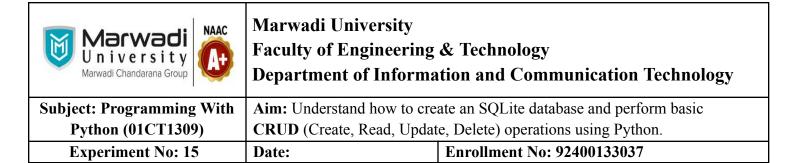
# Using executemany to insert multiple records

cursor.executemany("'INSERT INTO student_record (Enrollment, name, subject,Mark)

VALUES (?, ?, ?,?)'", student_record)

# Commit the changes

conn.commit()
```



Fetch Student Data Let's retrieve and display all student records.

Example

Fetch all student records
cursor.execute('SELECT * FROM student_record')
rows = cursor.fetchall()
Display the results
print("All Student Records:")
for row in rows:
 print(row)

Fetch Data with Specific Criteria

Let's fetch employees who earn more than 90.

Example

Fetch student got more than 90
cursor.execute('SELECT name, subject, Mark FROM student_record WHERE Mark > 90')
high_marks = cursor.fetchall()
print("\nStudents with Marks greater than 90:")
for student in high_marks:
 print(student)

Update Student Information

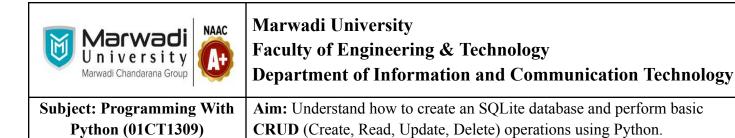
Suppose a student gets a raise in mark. We can update their mark using an UPDATE statement.

Example:

```
# Update MArk for Ashutosh kumar (PWP)

cursor.execute('''UPDATE student_record SET Mark = 98

WHERE name = 'ASHUTOSH KUMAR SINGH' AND subject = 'PWP' ''')
```



Commit the changes

Experiment No: 15

conn.commit()

Verify the update

cursor.execute('SELECT name, MArk FROM student_record WHERE name = "ASHUTOSH KUMAR SINGH"')

Enrollment No: 92400133037

updated_mark = cursor.fetchone()

print(f"\nUpdated Mark for {updated mark[0]}: {updated mark[1]}")

Date:

Delete a Student

Let's remove a student from the database.

Example:

Delete a student record (e.g., DEVENDRASINH DOLATSINH JADEJA)
cursor.execute("'DELETE FROM student_record WHERE name = 'DEVENDRASINH DOLATSINH JADEJA' "')

Commit the changes conn.commit()

Verify the deletion

cursor.execute('SELECT * FROM student_record WHERE name = "DEVENDRASINH DOLATSINH JADEJA"')
deleted name = cursor.fetchone()

if deleted_name is None:

print("\nDEVENDRASINH DOLATSINH JADEJA has been successfully deleted.")

Calculate Average Mark

Let's calculate the average mark of all students.

Example:

Calculate the average Mark
cursor.execute("'SELECT AVG(Mark) FROM student_record"')
avg mark = cursor.fetchone()[0]



print(f"\nThe average mark of students is: \${avg_mark:.2f}")

Close the Database Connection

Always close the connection after completing your operations.

Example

Close the connection conn.close()

Output:



Marwadi University

Faculty of Engineering & Technology

Department of Information and Communication Technology

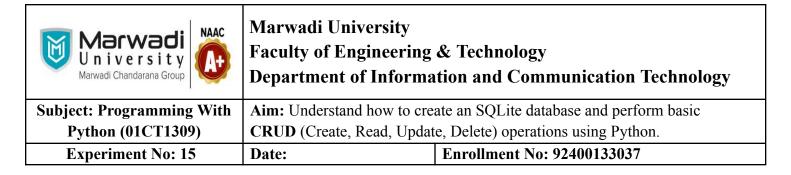
Subject: Programming With Python (01CT1309)

Aim: Understand how to create an SQLite database and perform basic **CRUD** (Create, Read, Update, Delete) operations using Python.

Experiment No: 15 Date: Enrollment No: 92400133037

```
cursor.execute('SELECT name,subject,Mark FROM student_record WHERE Mark>90')
high_marks=cursor.fetchall()
print("\n Students with Marks greater than 90:")
for i in high_marks:
    print(i)
cursor.execute('''UPDATE student_record SET Mark=98
               WHERE name='ASHUTOSH KUMAR SINGH' AND subject='PWP' ''')
conn.commit()
cursor.execute('SELECT name,Mark FROM student_record WHERE name="ASHUTOSH KUMAR SINGH"')
update mark=cursor.fetchone()
print(f"\nUpdate Mark for {update_mark[0]}:{update_mark[1]}")
cursor.execute('''DELETE FROM student_record WHERE name='DEVENDRASINH DOLATSINH JADEJA' ''')
conn.commit()
cursor.execute('SELECT*FROM student record WHERE name="DEVENDRASINH DOLATSINH JADEJA"')
deleted_name=cursor.fetchone()
if(deleted_name is None):
    print("\n DEVENDRASINH DOLATSINH JADEJA has been succefully deleted.")
```

```
cursor.execute('''SELECT AVG(Mark) FROM student_record''')
avg_mark=cursor.fetchone()[0]
print(f"\n The averge mark of student is:${avg_mark:.2f}")
conn.close()
```



```
PS G:\sem-3\python_lab> python _u "g:\sem-3\python_lab\lab15\sql.py"

Database connected successfully!

All Student Records:
(92301733016, 'ASHUTOSH KUMAR SINGH', 'PWP', 95)
(92301733017, 'HARSH VISHALBHAI TRIVEDI', 'PWP', 85)
(92301733027, 'VIRAJ PRAKASHBHAI VAGHASIYA', 'PWP', 90)
(92301733046, 'SHIVAM ATULKUMAR BHATT', 'PWP', 93)
(92301733058, 'DEVENDRASINH DOLATSINH JADEJA', 'PWP', 75)

Students with Marks greater than 90:
('ASHUTOSH KUMAR SINGH', 'PWP', 95)
('SHIVAM ATULKUMAR BHATT', 'PWP', 93)

Update Mark for ASHUTOSH KUMAR SINGH:98

DEVENDRASINH DOLATSINH JADEJA has been succefully deleted.

The averge mark of student is:$91.50
```

Post Lab Exercise:

• Modify the system to allow a student to enroll in multiple subjects at once.



Marwadi University

Faculty of Engineering & Technology

Department of Information and Communication Technology

Subject: Programming With Python (01CT1309)

Aim: Understand how to create an SQLite database and perform basic **CRUD** (Create, Read, Update, Delete) operations using Python.

Experiment No: 15 Date: Enrollment No: 92400133037

```
students = [
       ('ASHUTOSH KUMAR SINGH',),
       ('VIRAJ PRAKASHBHAI VAGHASIYA',),
   cursor.executemany('INSERT INTO student_record(name) VALUES(?)', students)
       (1, 'PWP', 95),
(2, 'DBMS', 90),
       (4, 'OOP', 88),
       (6, 'Maths', 89)
   cursor.executemany('INSERT INTO subjects( Enrollment,Subject, Mark) VALUES(?,?,?)', subjects)
   cursor.execute('''
   JOIN subjects sub ON s.Enrollment = sub.Enrollment
   rows = cursor.fetchall()
   for row in rows:
        print(row)
    conn.commit()
    conn.close()
PS G:\sem-3\python_lab> python -u "g:\sem-3\python_lab\lab15\PostLab.py"
Database connected successfully!
('ASHUTOSH KUMAR SINGH', 'PWP', 95)
('HARSH VISHALBHAI TRIVEDI', 'DBMS', 90)
('VIRAJ PRAKASHBHAI VAGHASIYA', 'PWP', 85)
PS G:\sem-3\python_lab>
```

GITHUB LINK

https://github.com/Heer972005/Python Lab