Experiment No. 13
Program to demonstrate CRUD (create, read, update and delete) operations on database (SQLite/ MySQL) using python
Date of Performance:
Date of Submission:

Experiment No. 13

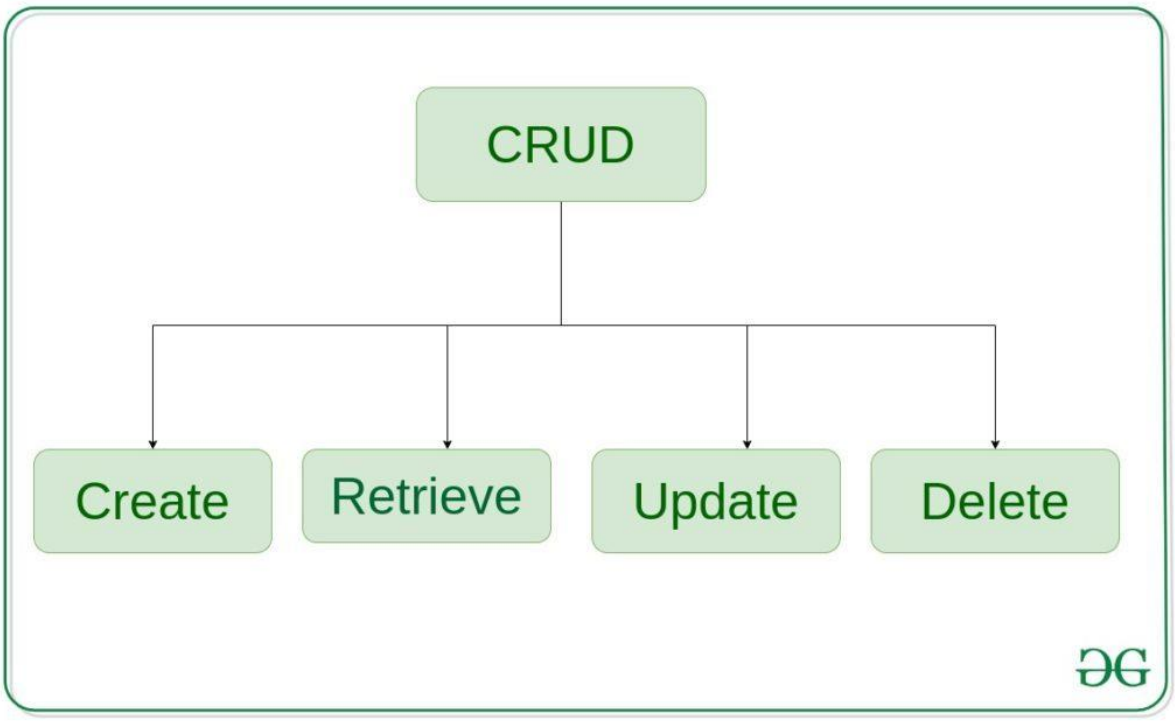
Title: Program to demonstrate CRUD (create, read, update and delete) operations on database (SQLite/ MySQL) using python

Aim: To study and implement CRUD (create, read, update and delete) operations on database (SQLite/ MySQL) using python

Objective: To introduce database connectivity with python

Theory:

In general CRUD means performing Create, Retrieve, Update and Delete operations on a table in a database. Let's discuss what actually CRUD means,



Create – create or add new entries in a table in the database.
Retrieve – read, retrieve, search, or view existing entries as a list(List View) or retrieve a particular entry in detail (Detail View)
Update – update or edit existing entries in a table in the database
Delete – delete, deactivate, or remove existing entries in a table in the database

CODE:

```
import mysql.connector

conn = mysql.connector.connect(
    host="localhost",
    user="root",
    password="root",
    database="sbl"
)
cursor = conn.cursor()

def create_record(name, age):
    sql = "INSERT INTO records (name, age) VALUES (%s, %s)"
    val = (name, age)
    cursor.execute(sql, val)
    conn.commit()
    print("Record created successfully")

def read_records():
    cursor.execute("SELECT * FROM records")
    records = cursor.fetchall()
    for record in records:
        print(record)

def update_record(id, new_name, new_age):
    sql = "UPDATE records SET name = %s, age = %s WHERE id = %s"
    val = (new_name, new_age, id)
    cursor.execute(sql, val)
    conn.commit()
    print("\nRecord updated successfully")

def delete_record(id):
    sql = "DELETE FROM records WHERE id = %s"
    val = (id,)
    cursor.execute(sql, val)
    conn.commit()
    print("\nRecord deleted successfully")

create_record("John", 30)
create_record("Alice", 25)
print("\nInitial records:")
read_records()

update_record(1, "John Doe", 35)
print("\nAfter update:")
read_records()
```

```
delete_record(2)
print("\nAfter deletion:")
read_records()

cursor.close()
conn.close()
```

OUTPUT:

```
============== RESTART: C:/Vedanti_Degree/SEM_4/SBL_PYTHON/pr13.py =============
Record created successfully
Record created successfully

Initial records:
(1, 'John', 30)
(2, 'Alice', 25)

Record updated successfully

After update:
(1, 'John Doe', 35)
(2, 'Alice', 25)

Record deleted successfully

After deletion:
(1, 'John Doe', 35)
```

CONCLUSION:

In conclusion, we've explored the fundamental operations of creating, reading, updating, and deleting data from a database, providing a practical understanding of how to interact with persistent storage in software applications. Moreover, utilizing Python for database operations has showcased the flexibility and simplicity of the language for handling data-centric tasks.