

Vidyavardhini's College of Engineering and Technology Department of Artificial Intelligence & Data Science

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()

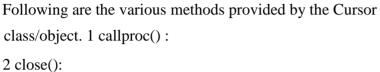


Vidyavardhini's College of Engineering and Technology Department of Artificial Intelligence & Data Science

Execute the Query

Use the execute() method of the cursor object to execute the
query Cursorobj.execute(SQL statement)

Methods



- 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



Vidyavardhini's College of Engineering and Technology

Department of Artificial Intelligence & Data Science

PROGRAM

1. To create a database

```
import sqlite3
def create connection(db file):
  """ create a database connection to the SQLite database specified by the db_file """
  conn = None
  try:
    conn = sqlite3.connect(db file)
    print("Connected successfully!")
    return conn
  except sqlite3.Error as e:
    print(e)
def create_table(conn):
  """ create a table if it does not exist """
  try:
    cursor = conn.cursor()
    cursor.execute("'CREATE TABLE IF NOT EXISTS example table (
                 id INTEGER PRIMARY KEY,
                 name TEXT NOT NULL,
                 age INTEGER
    print("Table created successfully!")
  except sqlite3.Error as e:
    print(e)
def insert_data(conn, name, age):
  """ insert data into the table """
  try:
    cursor = conn.cursor()
    cursor.execute("INSERT INTO example_table (name, age) VALUES (?, ?)", (name, age))
    conn.commit()
    print("Data inserted successfully!")
  except sqlite3.Error as e:
    print(e)
def delete_data(conn, id):
  """ delete data from the table """
    cursor = conn.cursor()
    cursor.execute("DELETE FROM example_table WHERE id = ?", (id,))
    conn.commit()
    print("Data deleted successfully!")
  except sqlite3.Error as e:
    print(c)
```



Vidyavardhini's College of Engineering and Technology

Department of Artificial Intelligence & Data Science

```
def update data(conn, id, new name, new age):
  """ update data in the table """
  try:
    cursor = conn.cursor()
    cursor.execute("UPDATE example_table SET name = ?, age = ? WHERE id = ?", (new_name,
new_age, id))
    conn.commit()
    print("Data updated successfully!")
  except sqlite3.Error as e:
    print(e)
def main():
  database = "example.db"
  conn = create connection(database)
    create_table(conn)
    # Inserting data
    insert data(conn, "Alice", 30)
    insert_data(conn, "Bob", 25)
    # Deleting data
    delete_data(conn, 1) # Assuming '1' is the id of the record to be deleted
    # Updating data
    update_data(conn, 2, "Bob", 26) # Assuming '2' is the id of the record to be updated
    conn.close()
    print("Connection closed successfully!")
if __name__ == '__main__':
  main()
       OUTPUT
PS C:\Users\Lenovo\Downloads\Python Prgs> python -u "c:\Users\Lenovo\Downloads\Python Prgs\database.py"
Connected Successfully
PS C:\Users\Lenovo\Downloads\Python Prgs>
```

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.