9. SQLite3 and PyMySQL (Database Connectors)

❖Introduction to SQLite3 and PyMySQL for database connectivity

Python provides powerful libraries for database connectivity, with SQLite3 and PyMySQL being two popular choices. Below is a detailed comparison and usage guide.

1. SQLite3 (Built-in Python Library) :

SQLite is a serverless, file-based database that doesn't require a separate server process.

- A lightweight, serverless, self-contained SQL database engine.
- Comes built into Python as the sqlite3 module—no installation required.
- Ideal for small to medium applications.

- Suitable for single-user or local file-based databases (e.g., desktop apps, prototypes, testing).

2. PyMySQL (For MySQL/MariaDB) :

PyMySQL is a pure-Python MySQL client that connects to remote/local MySQL databases.

- A Python library used to connect to a MySQL or MariaDB database server.
- Not included with Python—must be installed using pip install pymysql.
- Suitable for client-server applications.
- Common in web development, multi-user platforms, or large-scale systems.

❖Creating and executing SQL queries from Python using these connectors

1. SQLite3 :

- sqlite3 is a built-in Python module used to interact with SQLite databases.
- SQLite is a lightweight, serverless, and file-based database engine.
- It is ideal for small to medium-scale applications or for testing purposes.

Steps to Use SQLite3:

1. Import the sqlite3 module.
2. Connect to a database.
3. Create a cursor object.
4. Execute SQL queries.
5. Commit changes.
6. Close the connection.

2. PyMySQL :

- PyMySQL is a third-party Python library used to connect with MySQL or MariaDB databases.
- It is suitable for **client-server** applications that use a remote database.
- It must be installed using the command: pip install pymysql.

Steps to Use PyMySQL:

1. Import the pymysql module.
2. Establish a connection with the MySQL server.
3. Create a cursor object.
4. Execute SQL queries.
5. Fetch results (for SELECT queries).
6. Commit changes.
7. Close the connection.