Working with a MySQL database using Python typically involves the following steps:

1. **Setting up MySQL Database**
2. **Connecting to MySQL from Python**
3. **Creating Tables and Inserting Data**
4. **Retrieving and Manipulating Data**
5. **Closing the Connection**

Here's a simple guide to help you understand the basics of interacting with MySQL using Python.

1. Setting Up MySQL Database

Before you can interact with MySQL through Python, ensure that:

- MySQL is installed on your system.
- You have created a database in MySQL for storing your data.
- You have a MySQL user account with proper privileges.

You can use MySQL commands to create a database and table:

```
Sql Command to create database:-

CREATE DATABASE mydatabase;

Sql Command to create table:-

USE mydatabase;

CREATE TABLE notes (
    id INT AUTO_INCREMENT PRIMARY KEY,
    title VARCHAR(255),
    content TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

2. Installing MySQL Connector for Python

To interact with a MySQL database in Python, you'll need to install the MySQL connector library. You can install it using pip:

```
pip install mysql-connector-python
```

3. Connecting to MySQL from Python

You can use the `mysql.connector` module to establish a connection to your MySQL database:

```
import mysql.connector

# Establishing the connection
db_connection = mysql.connector.connect(
    host="localhost",     # Your MySQL host, typically 'localhost' or an IP
address
    user="yourusername",  # Your MySQL username
    password="yourpassword",  # Your MySQL password
    database="mydatabase"  # The database you want to work with
)

# Create a cursor object using the connection
cursor = db_connection.cursor()
```

4. Creating Tables and Inserting Data

Once connected, you can create tables, insert data, and perform various database operations.

Creating a Table

```
cursor.execute("""
CREATE TABLE IF NOT EXISTS notes (
    id INT AUTO_INCREMENT PRIMARY KEY,
    title VARCHAR(255),
    content TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
)
""")
```

Inserting Data into the Table

To insert notes into the `notes` table:

```

insert_query = "INSERT INTO notes (title, content) VALUES (%s, %s)"
```

```
note_data = ("My First Note", "This is the content of my first note.")
cursor.execute(insert_query, note_data)

# Commit the changes
db_connection.commit()

print(f"Inserted {cursor.rowcount} row(s) into the 'notes' table.")
```

5. Retrieving and Manipulating Data

You can retrieve data from the MySQL database using SQL queries.

Fetching All Notes

```
cursor.execute("SELECT * FROM notes")
notes = cursor.fetchall()

for note in notes:
    print(note)
```

Fetching a Single Note by ID

```
note_id = 1
cursor.execute("SELECT * FROM notes WHERE id = %s", (note_id,))
note = cursor.fetchone()
print(note)
```

Updating a Note

```

update_query = "UPDATE notes SET content = %s WHERE id = %s"
new_content = "This is the updated content of my first note."
cursor.execute(update_query, (new_content, 1))

# Commit the changes
db_connection.commit()

print(f"Updated {cursor.rowcount} row(s).")
```

Deleting a Note

```
delete_query = "DELETE FROM notes WHERE id = %s"
cursor.execute(delete_query, (1,))

# Commit the changes
db_connection.commit()

print(f"Deleted {cursor.rowcount} row(s).")
```

6. Closing the Connection

Once you are done with the database operations, close the cursor and the connection:

```

cursor.close()
db_connection.close()
```

Complete Example: Managing Notes

Here is a full example demonstrating all the steps (create table, insert, fetch, update, delete):

```
import mysql.connector

# Establishing the connection
db_connection = mysql.connector.connect(
    host="localhost",
    user="yourusername",
    password="yourpassword",
    database="mydatabase"
)

# Create a cursor object
cursor = db_connection.cursor()

# Create the table (if it doesn't already exist)
cursor.execute("""
CREATE TABLE IF NOT EXISTS notes (
    id INT AUTO_INCREMENT PRIMARY KEY,
    title VARCHAR(255),
```

```
    content TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
)
""")

# Insert a new note
insert_query = "INSERT INTO notes (title, content) VALUES (%s, %s)"
note_data = ("Note 1", "Content of note 1.")
cursor.execute(insert_query, note_data)
db_connection.commit()

# Fetch and display all notes
cursor.execute("SELECT * FROM notes")
notes = cursor.fetchall()
for note in notes:
    print(note)

# Update a note
update_query = "UPDATE notes SET content = %s WHERE id = %s"
cursor.execute(update_query, ("Updated content of note 1", 1))
db_connection.commit()

# Fetch and display the updated note
cursor.execute("SELECT * FROM notes WHERE id = 1")
note = cursor.fetchone()
print(note)

# Delete a note
cursor.execute("DELETE FROM notes WHERE id = 1")
db_connection.commit()

# Close the cursor and connection
cursor.close()
db_connection.close()
```