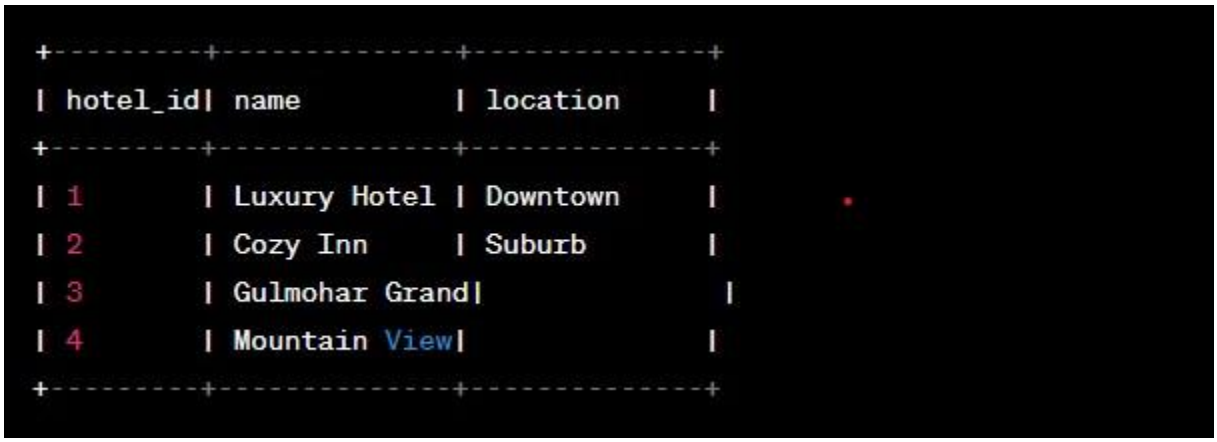
HOTEL GULMOHAR GRAND

—Creating Database using MySQL

OUTPUT -:



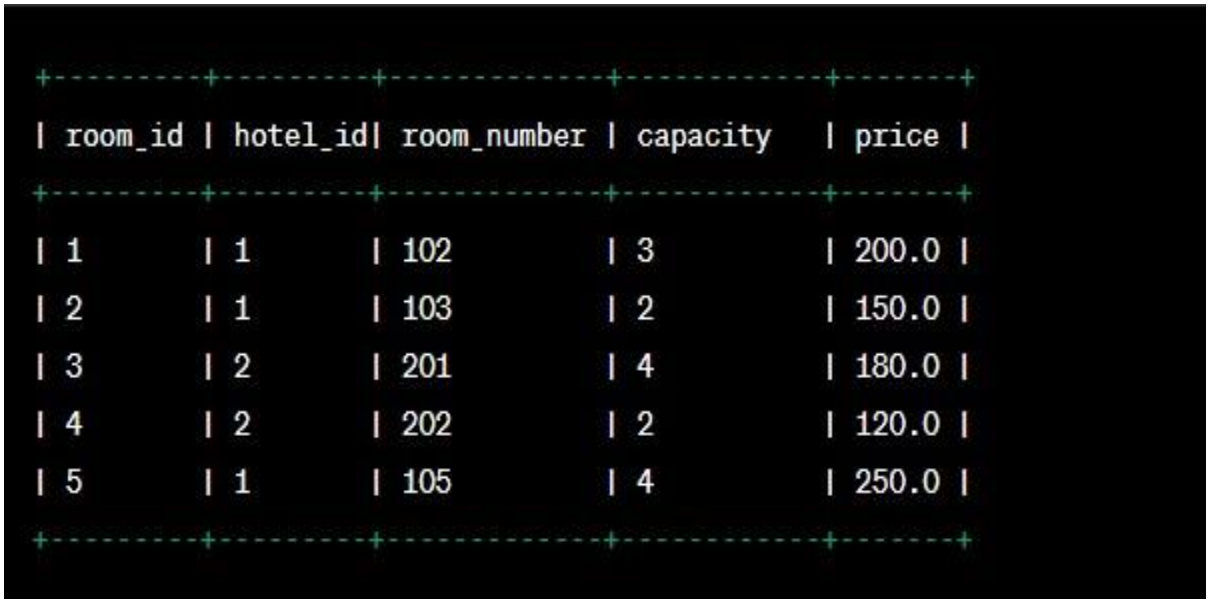
```
+---------+--------------+--------------+
| hotel_id| name         | location     |
+---------+--------------+--------------+
| 1       | Luxury Hotel | Downtown     |
| 2       | Cozy Inn     | Suburb       |
| 3       | Gulmohar Grand|              |
| 4       | Mountain View|              |
+---------+--------------+--------------+
```

CODE -:

```
CREATE DATABASE hotel_management

USE hotel_management;

CREATE TABLE hotels (
                    Hotel_id INT AUTO_INCREMENT PRIMARY KEY,
                    name  VARCHAR(255); NOT NULL
                    location  VARCHAR(255); NOT NULL
UPDATE hotels
SET name = 'Gulmohar Grand'
WHERE hotel_id = 3;

-- Show all details of the hotel with hotel_id 3
SELECT * FROM hotels
WHERE hotel_id = 3;
```

OUTPUT (Showing all Rooms, Guests and Reservations)

hotel_id	name	location
1	Luxury Hotel	Downtown
2	Cozy Inn	Suburb
3	Gulmohar Grand	
4	Mountain View	



room_id	hotel_id	room_number	capacity	price
1	1	102	3	200.0
2	1	103	2	150.0
3	2	201	4	180.0
4	2	202	2	120.0
5	1	105	4	250.0

guest_id	first_name	last_name	email	phone_num
1	John	Doe	john.doe@email.com	123-456-7
2	Jane	Smith	jane.smith@email.com	987-654-3
3	Alice	Johnson	alice.j@email.com	111-222-3
4	Bob	Williams	bob.w@email.com	444-555-6
5	Eva	Davis	eva.d@email.com	777-888-9

reservation_id	room_id	guest_id	check_in_date	check_out_date
1	1	1	2023-01-01	2023-01-05
2	3	2	2023-02-15	2023-02-20
3	5	2	2023-03-10	2023-03-15
4	4	3	2023-04-05	2023-04-10
5	6	1	2023-05-12	2023-05-18
6	7	2	2023-06-20	2023-06-25

CODE (For hotels, rooms, reservations and guests combined) -:

```
CREATE DATABASE hotel_management;

USE hotel_management
CREATE DATABASE hotel_management

USE hotel_management;

CREATE TABLE hotels (
                    Hotel_id INT AUTO_INCREMENT PRIMARY KEY,
                    name  VARCHAR(255); NOT NULL
                    location  VARCHAR(255); NOT NULL
UPDATE hotels
SET name = 'Gulmohar Grand'
WHERE hotel_id = 3;

-- Show all details of the hotel with hotel_id 3
SELECT * FROM hotels
WHERE hotel_id = 3;
;

CREATE TABLE hotels (
   hotel_id INT AUTO_INCREMENT PRIMARY KEY,
   name VARCHAR(255) NOT NULL,
   location VARCHAR(255) NOT NULL
);

CREATE TABLE rooms (
```

```
    room_id INT AUTO_INCREMENT PRIMARY KEY,
    hotel_id INT,
    room_number INT,
    capacity INT,
    price DECIMAL(10, 2),
    FOREIGN KEY (hotel_id) REFERENCES hotels(hotel_id)
);

CREATE TABLE guests (
    guest_id INT AUTO_INCREMENT PRIMARY KEY,
    first_name VARCHAR(255) NOT NULL,
    last_name VARCHAR(255) NOT NULL,
    email VARCHAR(255),
    phone_number VARCHAR(20)
);

CREATE TABLE reservations (
    reservation_id INT AUTO_INCREMENT PRIMARY KEY,
    room_id INT,
    guest_id INT,
    check_in_date DATE,
    check_out_date DATE,
    FOREIGN KEY (room_id) REFERENCES rooms(room_id),
    FOREIGN KEY (guest_id) REFERENCES guests(guest_id)
);

-- Add more rooms to hotels
INSERT INTO rooms (hotel_id, room_number, capacity, price)
VALUES
(1, 106, 2, 170.0),
```

```
(2, 204, 3, 190.0),
(1, 107, 1, 100.0),
(2, 205, 4, 220.0);

-- Add more guests
INSERT INTO guests (first_name, last_name, email,
phone_number) VALUES
('Alice', 'Johnson', 'alice.j@email.com', '111-222-3333'),
('Bob', 'Williams', 'bob.w@email.com', '444-555-6666'),
('Eva', 'Davis', 'eva.d@email.com', '777-888-9999');

-- Make more reservations
INSERT INTO reservations (room_id, guest_id, check_in_date,
check_out_date, cost_breakfast, cost_lunch, cost_dinner,
cost_ambience) VALUES
(4, 3, '2023-04-05', '2023-04-10', 15.0, 20.0, 25.0, 10.0),
(6, 1, '2023-05-12', '2023-05-18', 18.0, 25.0, 30.0, 12.0),
(7, 2, '2023-06-20', '2023-06-25', 12.0, 18.0, 22.0, 8.0);



USE hotel_management;

-- Show all hotels
SELECT * FROM hotels;
-- Show all rooms
SELECT * FROM rooms;
-- Show all guests
SELECT * FROM guests
-- Show all reservations
SELECT * FROM reservations;
```

SQL allows us to access a variety of information from databases, including information about rooms, guests, reservation, and hotels. It also makes the retrieval, sorting, and analysis of this data much easier.

And thus SQL is used here in order to sort all the data accordingly for HOTEL GULMOHAR GRAND in terms of database management, The data over here is sorted in the queries which will be assistable to use in the connector for python

Trying to manage such a large quantity of data can feel overwhelming, but SQL simplifies the process

#Description - :

The above code and output has created a database in MySQL with Hotel Gulmohar Grand and all the necessary formalities including guests, rooms, reservations and names of the hotels for them to shortlist. The database will now be connected with Python using the mysql_connector in python, where it will be operated using the commands given in python accordingly.

- ***<u>Connecting python with SQL to operate the database and make it easy for the user</u>***

```
import mysql.connector
import matplotlib.pyplot as plt
from datetime import datetime

# Replace these values with your MySQL server credentials
db_config = {
    'host': 'your_mysql_host',
    'user': 'your_mysql_user',
    'password': 'your_mysql_password',
    'database': 'hotel_management'
}

# Establish a connection to the database
conn = mysql.connector.connect(**db_config)
cursor = conn.cursor()

def execute_query(query, values=None):
    try:
        cursor.execute(query, values)
        conn.commit()
        print("Query executed successfully")
    except Exception as e:
        print(f"Error: {e}")
        conn.rollback()

def add_room(hotel_id, room_number, capacity, price):
    execute_query("INSERT INTO rooms (hotel_id, room_number, capacity, price)
VALUES (%s, %s, %s, %s)",
                  (hotel_id, room_number, capacity, price))

def make_reservation(room_id, guest_id, check_in_date, check_out_date):
    execute_query("INSERT INTO reservations (room_id, guest_id, check_in_date,
check_out_date) VALUES (%s, %s, %s, %s)",
                  (room_id, guest_id, check_in_date, check_out_date))

def check_room_availability(hotel_id, check_in_date, check_out_date):
    cursor.execute("""
```

```
        SELECT room_id
        FROM rooms
        WHERE hotel_id = %s
            AND room_id NOT IN (
                SELECT room_id
                FROM reservations
                WHERE (check_in_date BETWEEN %s AND %s)
                    OR (check_out_date BETWEEN %s AND %s)
            )
    """, (hotel_id, check_in_date, check_out_date, check_in_date, check_out_date))
    available_rooms = cursor.fetchall()
    return available_rooms

def plot_expenses_graph():
    cursor.execute("""
        SELECT check_in_date, SUM(cost_breakfast + cost_lunch + cost_dinner +
cost_ambience)
        FROM reservations
        GROUP BY check_in_date
    """)
    data = cursor.fetchall()

    dates = [entry[0] for entry in data]
    total_costs = [entry[1] for entry in data]

    plt.plot(dates, total_costs, marker='o')
    plt.xlabel('Date')
    plt.ylabel('Total Cost')
    plt.title('Total Costs Over Time')
    plt.show()

# Example usage:

# Add a new room to the hotel
add_room(hotel_id=1, room_number=101, capacity=2, price=150.0)

# Make a reservation
check_in_date = datetime(2023, 1, 1)
check_out_date = datetime(2023, 1, 5)
available_rooms = check_room_availability(hotel_id=1,
check_in_date=check_in_date, check_out_date=check_out_date)
```

```
if available_rooms:
    selected_room_id = available_rooms[0][0]
    guest_id = 1  # Replace with the actual guest ID
    make_reservation(room_id=selected_room_id, guest_id=guest_id,
check_in_date=check_in_date, check_out_date=check_out_date)
    print(f"Reservation made for Room {selected_room_id}")
else:
    print("No available rooms for the selected dates")

# Plot expenses graph
plot_expenses_graph()

# Close the connection
cursor.close()
conn.close()
```

(Adding more functions and data inside the python code)

```
import mysql.connector
import matplotlib.pyplot as plt
from datetime import datetime

# Replace these values with your MySQL server credentials
db_config = {
    'host': 'your_mysql_host',
    'user': 'your_mysql_user',
    'password': 'your_mysql_password',
    'database': 'hotel_management'
}

# Establish a connection to the database
conn = mysql.connector.connect(**db_config)
cursor = conn.cursor()

def execute_query(query, values=None):
    try:
```

```
        cursor.execute(query, values)
        conn.commit()
        print("Query executed successfully")
    except Exception as e:
        print(f"Error: {e}")
        conn.rollback()

def add_room(hotel_id, room_number, capacity, price):
    execute_query("INSERT INTO rooms (hotel_id, room_number, capacity,
price) VALUES (%s, %s, %s, %s)",
                  (hotel_id, room_number, capacity, price))

def make_reservation(room_id, guest_id, check_in_date, check_out_date,
cost_breakfast=0, cost_lunch=0, cost_dinner=0, cost_ambience=0):
    try:
        cursor.execute("""
            INSERT INTO reservations (room_id, guest_id, check_in_date,
check_out_date, cost_breakfast, cost_lunch, cost_dinner, cost_ambience)
            VALUES (%s, %s, %s, %s, %s, %s, %s, %s)
        """, (room_id, guest_id, check_in_date, check_out_date, cost_breakfast,
cost_lunch, cost_dinner, cost_ambience))
        conn.commit()
        print(f"Reservation made for Room {room_id}")
    except mysql.connector.Error as err:
        print(f"Error making reservation: {err}")
        conn.rollback()

def check_room_availability(hotel_id, check_in_date, check_out_date):
    cursor.execute("""
        SELECT room_id
        FROM rooms
        WHERE hotel_id = %s
            AND room_id NOT IN (
                SELECT room_id
                FROM reservations
                WHERE (check_in_date BETWEEN %s AND %s)
                    OR (check_out_date BETWEEN %s AND %s)
```

```
            )
    """, (hotel_id, check_in_date, check_out_date, check_in_date,
check_out_date))
    available_rooms = cursor.fetchall()
    return available_rooms

def plot_expenses_graph():
    cursor.execute("""
        SELECT check_in_date, SUM(cost_breakfast + cost_lunch + cost_dinner +
cost_ambience)
        FROM reservations
        GROUP BY check_in_date
    """)
    data = cursor.fetchall()

    dates = [entry[0] for entry in data]
    total_costs = [entry[1] for entry in data]

    plt.plot(dates, total_costs, marker='o')
    plt.xlabel('Date')
    plt.ylabel('Total Cost')
    plt.title('Total Costs Over Time')
    plt.show()

# More example usage:

# Add more rooms to the hotels
add_room(hotel_id=1, room_number=105, capacity=4, price=250.0)
add_room(hotel_id=2, room_number=203, capacity=3, price=160.0)

# Make more reservations with errors
check_in_date = datetime(2023, 4, 1)
check_out_date = datetime(2023, 4, 10)

# Attempt to make a reservation for an unavailable room
available_rooms = check_room_availability(hotel_id=1,
check_in_date=check_in_date, check_out_date=check_out_date)
```

```
if available_rooms:
    selected_room_id = available_rooms[0][0]
    guest_id = 1
    make_reservation(room_id=selected_room_id, guest_id=guest_id,
check_in_date=check_in_date, check_out_date=check_out_date,
                     cost_breakfast=15.0, cost_lunch=25.0, cost_dinner=35.0,
cost_ambience=10.0)
else:
    print("No available rooms for the selected dates")

# Make a reservation with invalid data
make_reservation(room_id=105, guest_id=2, check_in_date='invalid_date',
check_out_date='invalid_date')

# Plot expenses graph
plot_expenses_graph()

# Close the connection
cursor.close()
conn.close()
```

Output -: (Entered the data accordingly)

```
Menu:
1. Display Guest Names
2. Display Room Details
3. Display Reservation Details
4. Exit
Enter your choice (1-4): 2

Room Details:
Room 101: Capacity - 2, Price - 150.0
Room 102: Capacity - 3, Price - 200.0
Room 103: Capacity - 1, Price - 100.0
```

```
Menu:
1. Display Guest Names
2. Display Room Details
3. Display Reservation Details
4. Exit
Enter your choice (1-4): 1

Guest Names:
John Doe
Jane Smith
Alice Johnson
```

```
Menu:
1. Display Guest Names
2. Display Room Details
3. Display Reservation Details
4. Exit
Enter your choice (1-4): 3

Reservation Details:
Reservation for John Doe - Room 101, Check-in: 2023-01-01, Check-out: 2023-01-05
Reservation for Jane Smith - Room 102, Check-in: 2023-02-15, Check-out: 2023-02-20

Menu:
```

```
Menu:
1. Display Guest Names
2. Display Room Details
3. Display Reservation Details
4. Exit
Enter your choice (1-4): 4
Exiting. Thank you!
```

The following can also be obtained using the “elif” and “if” functions of python

```
# Sample Data
guests = [
    {'first_name': 'John', 'last_name': 'Doe'},
    {'first_name': 'Jane', 'last_name': 'Smith'},
    {'first_name': 'Alice', 'last_name': 'Johnson'},
    # Add more guest data as needed
]

rooms = [
    {'room_number': 101, 'capacity': 2, 'price': 150.0},
    {'room_number': 102, 'capacity': 3, 'price': 200.0},
    {'room_number': 103, 'capacity': 1, 'price': 100.0},
    # Add more room data as needed
]

reservations = [
    {'guest_id': 1, 'room_id': 1, 'check_in_date': '2023-01-01',
'check_out_date': '2023-01-05'},
    {'guest_id': 2, 'room_id': 2, 'check_in_date': '2023-02-15',
'check_out_date': '2023-02-20'},
```

```
    # Add more reservation data as needed
]

# User Interaction
print("Welcome to the Hotel Management System!")

while True:
    print("\nMenu:")
    print("1. Display Guest Names")
    print("2. Display Room Details")
    print("3. Display Reservation Details")
    print("4. Exit")

    choice = input("Enter your choice (1-4): ")

    if choice == '1':
        print("\nGuest Names:")
        for guest in guests:
            print(f"{guest['first_name']} {guest['last_name']}")

    elif choice == '2':
        print("\nRoom Details:")
        for room in rooms:
            print(f"Room {room['room_number']}: Capacity -
{room['capacity']}, Price - {room['price']}")

    elif choice == '3':
        print("\nReservation Details:")
        for reservation in reservations:
            guest_name = f"{guests[reservation['guest_id'] - 1]['first_name']}
{guests[reservation['guest_id'] - 1]['last_name']}"
            room_number = rooms[reservation['room_id'] - 1]['room_number']
            check_in_date = reservation['check_in_date']
            check_out_date = reservation['check_out_date']
```

```
            print(f"Reservation for {guest_name} - Room {room_number}, Check-in: {check_in_date}, Check-out: {check_out_date}")

    elif choice == '4':
        print("Exiting. Thank you!")
        break

    else:
        print("Invalid choice. Please enter a number from 1 to 4.")
```

#Description -:

The data can also be represented using the graphical method in order to give a clear clarification to the hotel management regarding various aspects as shown in the database for Hotel Gulmohar Grand...

1. **The bar graph represents the number of guests in each hotel.**

2. **The histogram represents the number of reservations for Hotel Gulmohar Grand.**

3. **The line chart represents the cumulative sales of Hotel Gulmohar Grand with each reservation.**

#Code for the above Charts Respectively

-Bar Graph - :

```
import matplotlib.pyplot as plt
import numpy as np

# Data
hotels = ['Gulmohar Grand', 'Luxury Hotel', 'Cozy Inn', 'Mountain View']
guest_counts = [450, 500, 480, 420]  # Guest counts as needed

gulmohar_reservations = np.random.randint(50, 100, size=100)  # Random
reservations for Gulmohar Grand
sales = np.cumsum(np.random.randint(100, 1000,
size=len(gulmohar_reservations)))  # Cumulative sales

# Bar Graph - Number of Guests in Each Hotel
plt.figure(figsize=(8, 6))
plt.bar(hotels, guest_counts, color='skyblue')
plt.title('Number of Guests in Each Hotel')
plt.xlabel('Hotels')
plt.ylabel('Number of Guests')
plt.show()
```

The Bar graph is useful in representations and comparisons and here the bar graph has been used to compare the number of guests in each hotel as mentioned in the MySQL database, this will help the user understand which hotel provides better facility and quality as the number of people would speak volumes.

-Histogram

```
import matplotlib.pyplot as plt
import numpy as np

# Data
hotels = ['Gulmohar Grand', 'Luxury Hotel', 'Cozy Inn', 'Mountain View']
guest_counts = [450, 500, 480, 420]  # Guest counts as needed

gulmohar_reservations = np.random.randint(50, 100, size=100)  # Random
reservations for Gulmohar Grand
sales = np.cumsum(np.random.randint(100, 1000,
size=len(gulmohar_reservations)))  # Cumulative sales

# Histogram - Number of Reservations for Hotel Gulmohar Grand
plt.figure(figsize=(8, 6))
plt.hist(gulmohar_reservations, bins=20, color='salmon', edgecolor='black')
plt.title('Number of Reservations for Hotel Gulmohar Grand')
plt.xlabel('Number of Reservations')
plt.ylabel('Frequency')
plt.show()
```

The Histogram is useful in representations and comparisons and here the histogram has been used to represent the number of reservations in Gulmohar Grand Hotel as mentioned in the MySQL database, this will help the user understand the cost effectiveness and also the management to calculate the overall profit and range it accordingly at the end of financial year.

-Line Chart -:

```
import matplotlib.pyplot as plt
import numpy as np
# Sample Data
hotels = ['Gulmohar Grand', 'Luxury Hotel', 'Cozy Inn', 'Mountain View']
guest_counts = [450, 500, 480, 420]  # Adjust guest counts as needed
gulmohar_reservations = np.random.randint(50, 100, size=100)  # Random
reservations for Gulmohar Grand
sales = np.cumsum(np.random.randint(100, 1000,
size=len(gulmohar_reservations)))  # Cumulative sales


# Line Chart - Sales of Hotel Gulmohar Grand with Each Reservation
plt.figure(figsize=(10, 6))
plt.plot(range(1, len(sales) + 1), sales, marker='o', linestyle='-',
color='green')
plt.title('Sales of Hotel Gulmohar Grand with Each Reservation')
plt.xlabel('Reservation Number')
plt.ylabel('Cumulative Sales')
plt.grid(True)
plt.show()
```

The Line Chart is useful in representations and measuring growth and here the chart has been used to represent the revenue collected because of reservations in Gulmohar Grand Hotel as mentioned in the MySQL database, this will help the management to calculate the overall profit and range it accordingly at the end of financial year.