1. Table Creation Queries

1.1. Users Table

```
CREATE TABLE users (
    id SERIAL PRIMARY KEY,
    username VARCHAR(50) UNIQUE NOT NULL,
    password VARCHAR(255) NOT NULL,
    email VARCHAR(100) UNIQUE NOT NULL,
    phone_number VARCHAR(15),
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

1.2. Hotels Table

```
CREATE TABLE hotels (
    id SERIAL PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    location VARCHAR(255) NOT NULL,
    description TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

1.3. Rooms Table

```
CREATE TABLE rooms (
    id SERIAL PRIMARY KEY,
    hotel_id INT REFERENCES hotels(id) ON DELETE CASCADE,
    room_type VARCHAR(50) NOT NULL,
    availability BOOLEAN DEFAULT TRUE,
    price_per_night DECIMAL(10, 2) NOT NULL,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

1.4. **Bookings Table**

```
CREATE TABLE bookings (
    id SERIAL PRIMARY KEY,
    user_id INT REFERENCES users(id) ON DELETE CASCADE,
    hotel_id INT REFERENCES hotels(id) ON DELETE CASCADE,
    room_id INT REFERENCES rooms(id) ON DELETE CASCADE,
    start_date DATE NOT NULL,
    end_date DATE NOT NULL,
    booking_status VARCHAR(50) DEFAULT 'CONFIRMED',
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

1.5. **Payments Table**

```
CREATE TABLE payments (
    id SERIAL PRIMARY KEY,
    user_id INT REFERENCES users(id) ON DELETE CASCADE,
    booking_id INT REFERENCES bookings(id) ON DELETE CASCADE,
    payment_status VARCHAR(50) DEFAULT 'PENDING',
    amount DECIMAL(10, 2) NOT NULL,
    transaction_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

---

2. Insert Data Queries

2.1. Insert a New User

```
INSERT INTO users (username, password, email, phone_number)
VALUES ('john_doe', 'hashed_password', 'john.doe@example.com', '1234567890');
```

2.2. Insert a New Hotel

```
INSERT INTO hotels (name, location, description)
VALUES ('Sunset Hotel', 'Downtown City', 'A luxury hotel with all amenities.');
```

2.3. Insert a New Room for a Hotel

```
INSERT INTO rooms (hotel_id, room_type, availability, price_per_night)
VALUES (1, 'Deluxe Room', TRUE, 150.00);
```

2.4. Insert a New Booking

```
INSERT INTO bookings (user_id, hotel_id, room_id, start_date, end_date)
VALUES (1, 1, 1, '2024-10-01', '2024-10-05');
```

2.5. Insert a Payment

```
INSERT INTO payments (user_id, booking_id, payment_status, amount)
VALUES (1, 1, 'COMPLETED', 600.00);
```

3. **Select Queries**

3.1. **View Booking Records for a Specific Hotel**

```
SELECT b.id AS booking_id, u.username, u.email, b.start_date, b.end_date,
b.booking_status
FROM bookings b
JOIN users u ON b.user_id = u.id
WHERE b.hotel_id = 1;  -- Replace 1 with the specific hotel_id
```

3.2. **View Booking Records for a Specific User**

```
SELECT b.id AS booking_id, h.name AS hotel_name, b.start_date, b.end_date,
b.booking_status
FROM bookings b
JOIN hotels h ON b.hotel_id = h.id
WHERE b.user_id = 1;  -- Replace 1 with the specific user_id
```

3.3. **View Payment Records for a Specific Hotel**

```
SELECT p.id AS payment_id, u.username, u.email, p.amount, p.payment_status,
p.transaction_date
FROM payments p
JOIN bookings b ON p.booking_id = b.id
JOIN users u ON b.user_id = u.id
WHERE b.hotel_id = 1;  -- Replace 1 with the specific hotel_id
```

3.4. **View Payment Status for a Specific User**

```
SELECT p.id AS payment_id, h.name AS hotel_name, p.amount, p.payment_status,
p.transaction_date
FROM payments p
JOIN bookings b ON p.booking_id = b.id
JOIN hotels h ON b.hotel_id = h.id
WHERE p.user_id = 1;  -- Replace 1 with the specific user_id
```

4. **Update Queries**

4.1. **Update Room Availability**

```
UPDATE rooms
SET availability = FALSE
WHERE id = 1;  -- Replace 1 with the specific room_id
```

4.2. **Update Booking Dates (Reschedule Booking)**

```
UPDATE bookings
SET start_date = '2024-10-10', end_date = '2024-10-15'
WHERE id = 1;  -- Replace 1 with the specific booking_id
```

4.3. **Update User Profile Information**

```
UPDATE users
SET email = 'new.email@example.com', phone_number = '9876543210'
WHERE id = 1;  -- Replace 1 with the specific user_id
```

4.4. **Update Payment Status**

```
UPDATE payments
SET payment_status = 'COMPLETED'
WHERE id = 1;  -- Replace 1 with the specific payment_id
```

---

5. **Delete Queries**

5.1. **Delete a Booking (Cancel Booking)**

```
DELETE FROM bookings
WHERE id = 1;  -- Replace 1 with the specific booking_id
```

5.2. **Delete a User**

```
DELETE FROM users
WHERE id = 1;  -- Replace 1 with the specific user_id
```

Summary of Key Queries:

- **Creating tables**: Define the schema for users, hotels, rooms, bookings, and payments.
- **Inserting data**: Add new users, hotels, rooms, bookings, and payments.
- **Selecting data**: View booking and payment records based on specific criteria.
- **Updating data**: Modify room availability, reschedule bookings, update user info, and change payment status.
- **Deleting data**: Remove bookings (for cancellations) or users (for account deletion).
