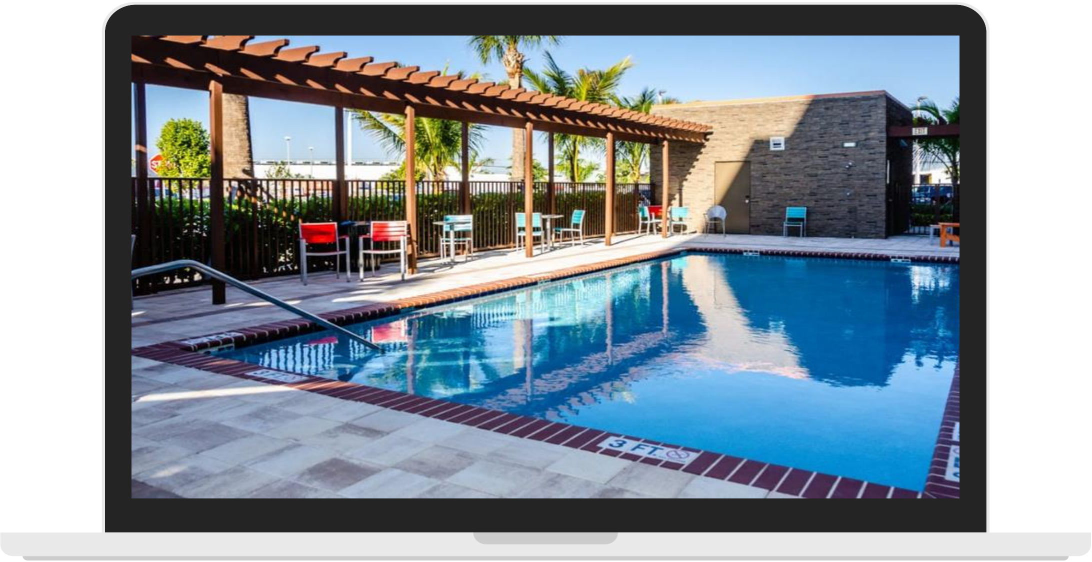
HOSPITALITY – HOTEL BOOKING



Siva Sakthi Balan Sudhakar Sundarambal
sivasakthibalanss@gmail.com

Mentor: Dr. Junaid Qazi

https://github.com/SivaSakthiBalanSS

TABLE OF CONTENTS

1. Introduction
2. Mission Statement
3. Objectives
4. Table List
    - 4.1 Guest
    - 4.2 Room Specification
    - 4.3 Booking Reservation
    - 4.4 Staff
    - 4.5 Service
    - 4.6 Feedback
5. Entity Relationship Diagram
6. Relationship
7. Conclusion
8. Appendix

INTRODUCTION:

The current hospitality environment demands exceptional guest experiences since business competition remains exceptionally high. The process of achieving this objective depends heavily on a trustworthy hotel booking system. The Hotel Booking System focuses on three core components which include personalized service and seamless reservations and efficient hotel management. Through the collection of data-driven insights this system allows hotels to personalize guest encounters by offering particular suggestions which results in transformative hotel visits. Hospitality guests experience simple reservations through the hotel website and third-party partners that enables instant booking confirmation and reservation handling. The system makes hotel operations more efficient through automatic execution of hotel essentials including room inventory control and payment handling and guest care processing. The automated system allows staff members to dedicate their time towards outstanding service delivery. The implementation of our Hotel Booking System results in both improved guest satisfaction and optimized hotel management operations which lead to prolonged customer retention and organizational success.

MISSION STATEMENT:

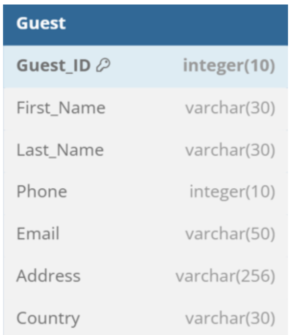
Our mission is to create unforgettable experiences by combining innovative technology with a personal touch and a commitment to sustainability. We strive to make every stay unique by understanding and catering to individual preferences while also giving back to the communities we serve. By investing in our team’s growth and development, we cultivate a culture of excellence, ensuring that every guest feels valued and receives exceptional service.

OBJECTIVES:

- Create Meaningful Guest Experiences: Use smart data insights to personalize interactions, making every guest feel valued and encouraging long-term loyalty.
- Simplify Communication: Integrate real-time messaging tools so guests can easily connect with staff for quick assistance and a seamless stay.
- Optimize Daily Operations: Automate routine tasks like billing and housekeeping requests, reducing errors and allowing staff to focus on exceptional service.
- Protect Guest Privacy: Implement strong security measures to safeguard personal data and ensure compliance with privacy regulations.
- Encourage Direct Bookings: Offer exclusive perks or discounts for guests who book directly through the hotel's website, reducing reliance on third-party platforms.
- Go Green with Sustainability: Adopt eco-friendly practices such as energy-efficient systems and paperless check-ins to minimize environmental impact.
- Maximize Revenue Potential: Use dynamic pricing tools to adjust room rates based on demand, ensuring competitive pricing and high occupancy.
- Reach a Global Audience: Connect with worldwide booking networks to attract travellers from different markets and expand the hotel's reach.

TABLES:

GUEST TABLE:



Guest	
Guest_ID	integer(10)
First_Name	varchar(30)
Last_Name	varchar(30)
Phone	integer(10)
Email	varchar(50)
Address	varchar(256)
Country	varchar(30)

Overview:

This table is designed to store essential guest details, including personal information, location, and a unique identifier for each guest. It plays a crucial role in hotel management and booking systems, ensuring efficient data organization and retrieval.

Fields & Their Purpose:

- **Guest_ID**

Type: Integer (10)

Purpose: Acts as the unique identifier for each guest, serving as the primary key.

- **First_Name**

Type: VARCHAR (30)

Purpose: Stores the guest's first name, with a maximum length of 30 characters.

- **Last_Name**

Type: VARCHAR (30)

Purpose: Stores the guest's last name, limited to 30 characters.

- **Phone**

Type: Integer (10)

Purpose: Holds the guest's phone number, restricted to a maximum of 10 digits.

- **Email**

Type: VARCHAR (50)

Purpose: Stores the guest’s email address, allowing up to 50 characters.

- **Address**

Type: VARCHAR (256)

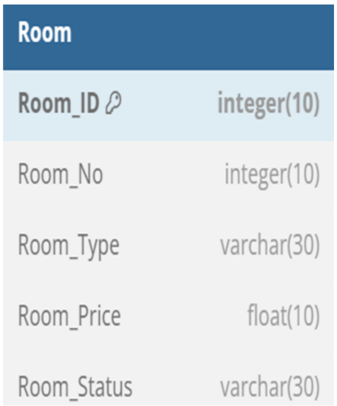
Purpose: Contains the full address of the guest, supporting up to 256 characters.

- **Country**

Type: VARCHAR (30)

Purpose: Records the guest’s country of residence, with a 30-character limit.

ROOM TABLE:



Room	
Room_ID 🔑	integer(10)
Room_No	integer(10)
Room_Type	varchar(30)
Room_Price	float(10)
Room_Status	varchar(30)

Overview:

The Room table is designed to efficiently manage room details within a hotel or property management system. It helps in tracking room availability, pricing, and categorization to streamline operations and enhance guest experiences.

Fields & Their Purpose:

- **Room_ID**

Purpose: Serves as the primary key, uniquely identifying each room in the system.

Ensures: Prevents duplication of room records for accurate management.

- **Room_No**

Purpose: Represents the actual room number assigned within the facility.

Benefit: Allows easy mapping of database records to physical rooms.

- **Room_Type**

Purpose: Categorizes rooms based on type, such as "Single," "Double," or "Suite."

Benefit: Helps guests and staff filter and select rooms based on preferences and needs.

- **Room_Price**

Purpose: Stores the cost of booking the room.

Data Type: Float (supports decimal values like $100.50).

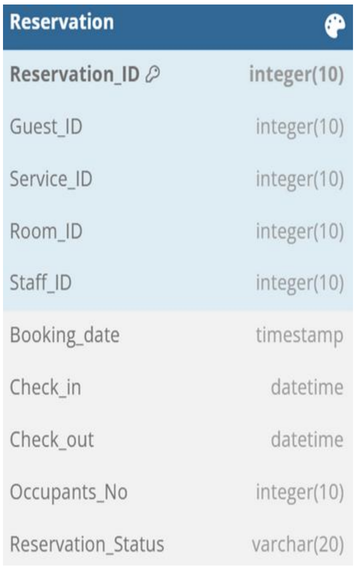
Benefit: Ensures accurate pricing calculations for guests and management.

- **Room_Status**

Purpose: Indicates the current condition of the room, such as "Available," "Occupied," or "Under Maintenance."

Benefit: Helps staff efficiently manage room assignments and maintenance schedules.

RESERVATION TABLE:



Reservation	
Reservation_ID	**integer(10)**
Guest_ID	integer(10)
Service_ID	integer(10)
Room_ID	integer(10)
Staff_ID	integer(10)
Booking_date	timestamp
Check_in	datetime
Check_out	datetime
Occupants_No	integer(10)
Reservation_Status	varchar(20)

Overview:

The Reservation table is essential for managing guest bookings, ensuring seamless room allocation, tracking additional services, and maintaining organized records for hotel operations.

Fields & Their Purpose:

- **Reservation_ID**

Purpose: Serves as the primary key, uniquely identifying each reservation.

Ensures: Prevents duplicate entries and maintains accurate booking records.

- **Guest_ID**

Purpose: Links to the Guest table, representing the guest making the reservation.

Benefit: Helps track all bookings associated with a specific guest.

- **Service_ID**

Purpose: Links to a Services table (if available), recording any extra services like spa treatments, room service, or transportation.

Benefit: Allows for easy tracking of additional amenities used during the stay.

- **Room_ID**

Purpose: References the Room table to specify the reserved room.

Ensures: Guarantees that each reservation is linked to a valid room.

- **Staff_ID**

Purpose: References a Staff table (if present), identifying the staff member responsible for managing the reservation.

Benefit: Helps assign and track staff responsibilities for booking handling.

- **Booking_Date**

Purpose: Captures the exact date and time when the reservation was created.

Benefit: Useful for tracking booking trends and generating reports.

- **Check_In**

Purpose: Stores the check-in date and time for the reservation.

Benefit: Ensures proper planning for room availability and guest arrivals.

- **Check_Out**

Purpose: Records the check-out date and time.

Benefit: Helps calculate the duration of stay and associated charges.

- **Occupants_No**

Purpose: Stores the number of guests included in the reservation.

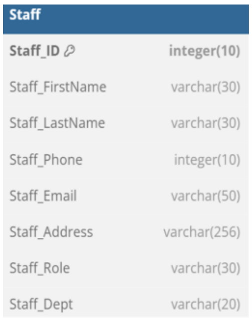
Ensures: That room occupancy limits are not exceeded.

- **Reservation_Status**

Purpose: Indicates the current status of the reservation (e.g., Confirmed, Pending, Cancelled, Completed).

Benefit: Helps track the reservation lifecycle and manage availability efficiently.

STAFF TABLE:



Staff	
Staff_ID 🔑	**integer(10)**
Staff_FirstName	varchar(30)
Staff_LastName	varchar(30)
Staff_Phone	integer(10)
Staff_Email	varchar(50)
Staff_Address	varchar(256)
Staff_Role	varchar(30)
Staff_Dept	varchar(20)

Overview:

The Staff table is designed to store and manage essential details about employees within an organization. It helps in tracking staff members, their roles, and their contact information, ensuring smooth workforce management.

Fields & Their Purpose:

- **Staff_ID (Integer, 10)**

Purpose: Serves as a unique identifier for each staff member.

Primary Key: Ensures that every employee has a distinct ID.

Auto-increment: Typically set to auto-generate a sequential ID for new employees.

- **Staff_FirstName (VARCHAR, 30)**

Purpose: Stores the first name of the staff member.

Example: "John," "Alice," "David."

- **Staff_LastName (VARCHAR, 30)**

Purpose: Stores the last name (surname) of the staff member.

Example: "Doe," "Smith," "Johnson."

- **Staff_Phone (Integer, 10)**

Purpose: Stores the contact phone number of the staff member.

Example: 9876543210, 1234567890.

- **Staff_Email (VARCHAR, 50)**

Purpose: Stores the staff member’s email address, used for communication, login credentials, and notifications.

Example: "john.doe@example.com," "alice.smith@company.com."

- **Staff_Address (VARCHAR, 256)**

Purpose: Stores the staff member’s full address, useful for payroll processing and official documentation.

Example: "123 Main Street, New York, NY," "45B Baker Street, London."

- **Staff_Role (VARCHAR, 30)**

Purpose: Specifies the job title or role of the staff member.

Example: "Manager," "Receptionist," "IT Technician."

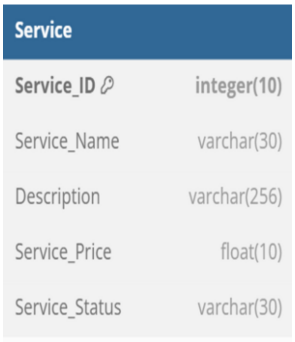
Benefit: Helps define job responsibilities and access levels.

- **Staff_Dept (VARCHAR, 20)**

Purpose: Indicates the department where the staff member works.

Example: "HR," "Finance," "Housekeeping."

SERVICE TABLE:



Service	
Service_ID	integer(10)
Service_Name	varchar(30)
Description	varchar(256)
Service_Price	float(10)
Service_Status	varchar(30)

Overview:

The Service table is designed to keep track of various services offered by a company, including details like service name, description, pricing, and availability. This ensures efficient service management and helps customers understand the options available.

Fields & Their Purpose:

- **Service_ID (Integer, 10)**

Purpose: Serves as a unique identifier for each service.

Primary Key: Ensures that no two services share the same ID.

Auto-increment: Typically set to automatically generate a sequential ID for new services.

- **Service_Name (VARCHAR, 30)**

Purpose: Stores the name of the service.

Example: "Haircut," "Car Wash," "Software Installation."

- **Description (VARCHAR, 256)**

Purpose: Provides a brief overview of the service to help customers understand what it includes.

Example:

"Professional haircut and styling service."

"Complete interior and exterior car wash."

- **Service_Price (Float, 10)**

Purpose: Stores the cost of the service, allowing for decimal values.

Example:

29.99 (for a haircut)

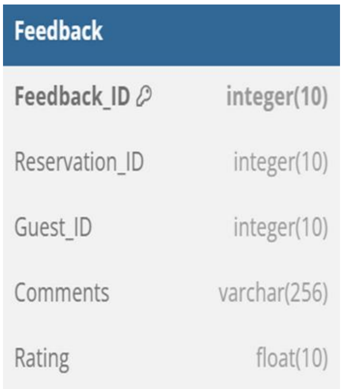
49.50 (for a car wash)

- **Service_Status (VARCHAR, 30)**

Purpose: Indicates whether the service is currently available or not.

Example: "Available," "Temporarily Unavailable."

FEEDBACK TABLE:



Feedback	
Feedback_ID 🔑	integer(10)
Reservation_ID	integer(10)
Guest_ID	integer(10)
Comments	varchar(256)
Rating	float(10)

Overview:

The Feedback table is designed to collect guest reviews, ratings, and comments related to their reservations. This information helps businesses measure customer satisfaction, identify areas for improvement, and enhance overall service quality.

Fields & Their Purpose:

- **Feedback_ID (Integer, 10)**

Purpose: Serves as a unique identifier for each feedback entry.

Primary Key: Ensures that no two feedback records have the same ID.

Auto-increment: Typically set to generate a sequential ID for new feedback submissions.

- **Reservation_ID (Integer, 10)**

Purpose: Links the feedback to a specific reservation, ensuring reviews are associated with an actual stay.

Foreign Key: Likely references a Reservation table to track which booking the feedback belongs to.

- **Guest_ID (Integer, 10)**

Purpose: Identifies the guest who provided the feedback.

Foreign Key: Likely references a Guest table for tracking guest reviews over time.

- **Comments (VARCHAR, 256)**

Purpose: Stores written feedback from guests, capturing their thoughts on their stay.

Example:

"Great service! The room was clean and comfortable."

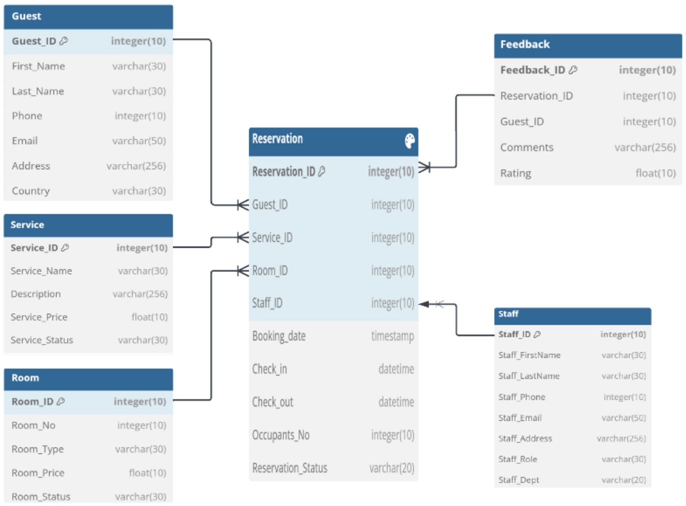
"The food was excellent, but the waiting time was too long."

- **Rating (Float, 10)**

Purpose: Stores a numerical rating provided by the guest to quantify their experience.

Example: 4.5 (out of 5).

ENTITY RELATIONSHIP DIAGRAM:



Guest	
Guest_ID	integer(10)
First_Name	varchar(30)
Last_Name	varchar(30)
Phone	integer(10)
Email	varchar(50)
Address	varchar(256)
Country	varchar(30)

Service	
Service_ID	integer(10)
Service_Name	varchar(30)
Description	varchar(256)
Service_Price	float(10)
Service_Status	varchar(30)

Room	
Room_ID	integer(10)
Room_No	integer(10)
Room_Type	varchar(30)
Room_Price	float(10)
Room_Status	varchar(30)

Reservation	
Reservation_ID	integer(10)
Guest_ID	integer(10)
Service_ID	integer(10)
Room_ID	integer(10)
Staff_ID	integer(10)
Booking_date	timestamp
Check_in	datetime
Check_out	datetime
Occupants_No	integer(10)
Reservation_Status	varchar(20)

Feedback	
Feedback_ID	integer(10)
Reservation_ID	integer(10)
Guest_ID	integer(10)
Comments	varchar(256)
Rating	float(10)

Staff	
Staff_ID	integer(10)
Staff_FirstName	varchar(30)
Staff_LastName	varchar(30)
Staff_Phone	integer(10)
Staff_Email	varchar(50)
Staff_Address	varchar(256)
Staff_Role	varchar(30)
Staff_Dept	varchar(20)

Overview:

This Entity-Relationship Diagram (ERD) outlines the structure of a hotel reservation management system, showcasing how various entities—guests, rooms, staff, payments, services, and feedback—interconnect to ensure smooth booking and management operations.

Key Components & Their Roles:

Guest Table:

- Stores guest details, including Guest_ID, name, phone number, email, address, and country.
- Each guest can make multiple reservations, which are recorded in the Reservation Table.

Reservation Table:

- Links each guest to their booked room(s) and services, assigning a Reservation_ID to each booking.

- Captures details like booking date, check-in/check-out dates, number of occupants, and reservation status.
- Each reservation is managed by a staff member (linked via Staff_ID).

Room Table:

- Stores room-related data, including Room_ID, Room_No, Room_Type, Room_Price, and Room_Status.
- Helps manage room availability and assignments efficiently.

Service Table:

- Lists additional services offered to guests, such as spa treatments, dining, and laundry services.
- Each service is uniquely identified by a Service_ID and includes Name, Description, Price, and Status.

Feedback Table:

- Captures guest reviews and ratings for completed reservations.
- Each feedback entry includes Feedback_ID, Reservation_ID, Guest_ID, Comments, and Rating.

Staff Table:

- Maintains employee records, tracking details such as Staff_ID, First Name, Last Name, Phone, Email, Address, Role, and Department.
- Helps in assigning roles and managing responsibilities.

RELATIONSHIPS BETWEEN ENTITIES:

Guests & Reservations (One-to-Many)

- A single guest can have multiple reservations over time, but each reservation is linked to only one guest.
- Foreign Key: Guest_ID in the Reservation table references Guest_ID in the Guest table.

Rooms & Reservations (One-to-Many)

- A room can be booked multiple times by different guests on different dates, but each reservation is tied to a single room at a time.
- Foreign Key: Room_ID in the Reservation table references Room_ID in the Room table.

Reservations & Feedback (One-to-One)

- Each reservation has a single feedback entry, linking guest reviews to specific stays.
- Foreign Key: Reservation_ID in the Feedback table references Reservation_ID in the Reservation table.

Guests & Feedback (One-to-Many)

- A guest can provide multiple feedback entries for different reservations, but each feedback entry is tied to only one guest.
- Foreign Key: Guest_ID in the Feedback table references Guest_ID in the Guest table.

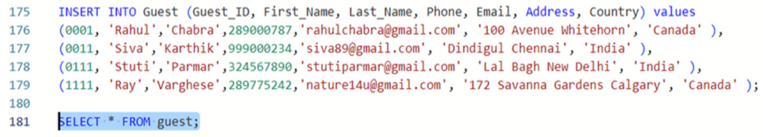
CONCLUSION:

The Hotel Booking System improves efficiency, security, and guest satisfaction by streamlining reservations, payments, and service management. By automating tasks and personalizing guest experiences, it helps hotels attract more customers while maintaining high operational standards. Additionally, its support for eco-friendly initiatives makes it a sustainable and competitive solution in the hospitality industry.

Appendix:

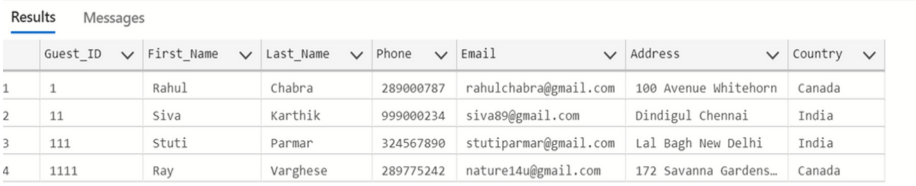
Screenshots of a Databases and SQL Queries:

Guest Table:



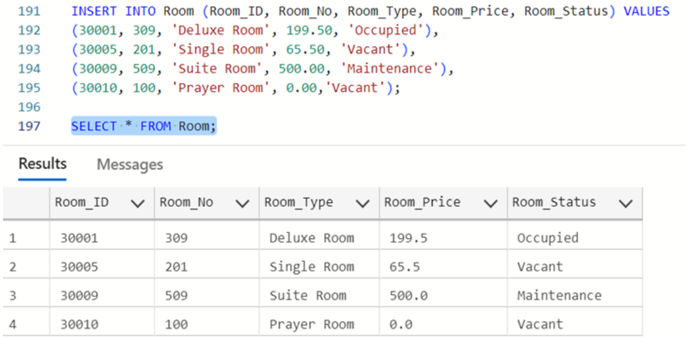
```sql
INSERT INTO Guest (Guest_ID, First_Name, Last_Name, Phone, Email, Address, Country) values
(0001, 'Rahul','Chabra',289000787,'rahulchabra@gmail.com', '100 Avenue Whitehorn', 'Canada' ),
(0011, 'Siva','Karthik',999000234,'siva89@gmail.com', 'Dindigul Chennai', 'India' ),
(0111, 'Stuti','Parmar',324567890,'stutiparmar@gmail.com', 'Lal Bagh New Delhi', 'India' ),
(1111, 'Ray','Varghese',289775242,'nature14u@gmail.com', '172 Savanna Gardens Calgary', 'Canada' );

SELECT * FROM guest;
```



	Guest_ID	First_Name	Last_Name	Phone	Email	Address	Country
1	1	Rahul	Chabra	289000787	rahulchabra@gmail.com	100 Avenue Whitehorn	Canada
2	11	Siva	Karthik	999000234	siva89@gmail.com	Dindigul Chennai	India
3	111	Stuti	Parmar	324567890	stutiparmar@gmail.com	Lal Bagh New Delhi	India
4	1111	Ray	Varghese	289775242	nature14u@gmail.com	172 Savanna Gardens…	Canada

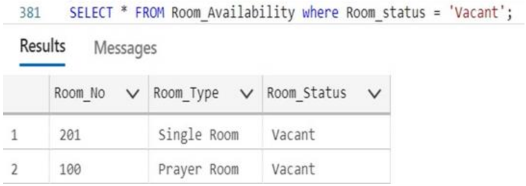
Room Table:



```sql
INSERT INTO Room (Room_ID, Room_No, Room_Type, Room_Price, Room_Status) VALUES
(30001, 309, 'Deluxe Room', 199.50, 'Occupied'),
(30005, 201, 'Single Room', 65.50, 'Vacant'),
(30009, 509, 'Suite Room', 500.00, 'Maintenance'),
(30010, 100, 'Prayer Room', 0.00,'Vacant');

SELECT * FROM Room;
```

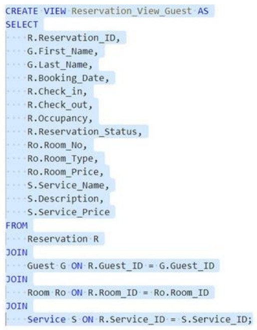
	Room_ID	Room_No	Room_Type	Room_Price	Room_Status
1	30001	309	Deluxe Room	199.5	Occupied
2	30005	201	Single Room	65.5	Vacant
3	30009	509	Suite Room	500.0	Maintenance
4	30010	100	Prayer Room	0.0	Vacant



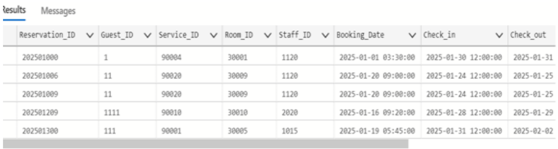
```sql
SELECT * FROM Room_Availability where Room_status = 'Vacant';
```

	Room_No	Room_Type	Room_Status
1	201	Single Room	Vacant
2	100	Prayer Room	Vacant

Reservation Table:

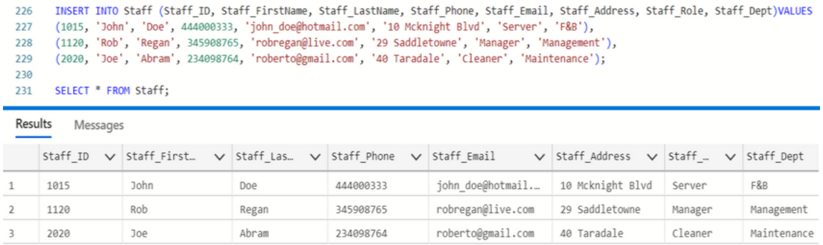


```sql
CREATE VIEW Reservation_View_Guest AS
SELECT
    R.Reservation_ID,
    G.First_Name,
    G.Last_Name,
    R.Booking_Date,
    R.Check_in,
    R.Check_out,
    R.Occupancy,
    R.Reservation_Status,
    Ro.Room_No,
    Ro.Room_Type,
    Ro.Room_Price,
    S.Service_Name,
    S.Description,
    S.Service_Price
FROM
    Reservation R
JOIN
    Guest G ON R.Guest_ID = G.Guest_ID
JOIN
    Room Ro ON R.Room_ID = Ro.Room_ID
JOIN
    Service S ON R.Service_ID = S.Service_ID;
```



Reservation_ID	Guest_ID	Service_ID	Room_ID	Staff_ID	Booking_Date	Check_in	Check_out
202501000	1	90004	30001	1120	2025-01-01 03:30:00	2025-01-30 12:00:00	2025-01-31
202501006	11	90020	30009	1120	2025-01-20 09:00:00	2025-01-24 12:00:00	2025-01-25
202501009	11	90020	30009	1120	2025-01-20 09:00:00	2025-01-24 12:00:00	2025-01-25
202501209	1111	90010	30010	2020	2025-01-16 09:20:00	2025-01-28 12:00:00	2025-01-29
202501300	111	90001	30005	1015	2025-01-19 05:45:00	2025-01-31 12:00:00	2025-02-02

Staff Table:

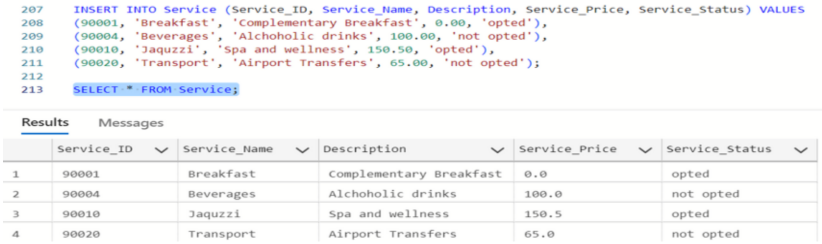


```sql
INSERT INTO Staff (Staff_ID, Staff_FirstName, Staff_LastName, Staff_Phone, Staff_Email, Staff_Address, Staff_Role, Staff_Dept)VALUES
(1015, 'John', 'Doe', 444000333, 'john_doe@hotmail.com', '10 Mcknight Blvd', 'Server', 'F&B'),
(1120, 'Rob', 'Regan', 345908765, 'robregan@live.com', '29 Saddletowne', 'Manager', 'Management'),
(2020, 'Joe', 'Abram', 234098764, 'roberto@gmail.com', '40 Taradale', 'Cleaner', 'Maintenance');

SELECT * FROM Staff;
```

	Staff_ID	Staff_First...	Staff_Las...	Staff_Phone	Staff_Email	Staff_Address	Staff_...	Staff_Dept
1	1015	John	Doe	444000333	john_doe@hotmail...	10 Mcknight Blvd	Server	F&B
2	1120	Rob	Regan	345908765	robregan@live.com	29 Saddletowne	Manager	Management
3	2020	Joe	Abram	234098764	roberto@gmail.com	40 Taradale	Cleaner	Maintenance

Service Table:

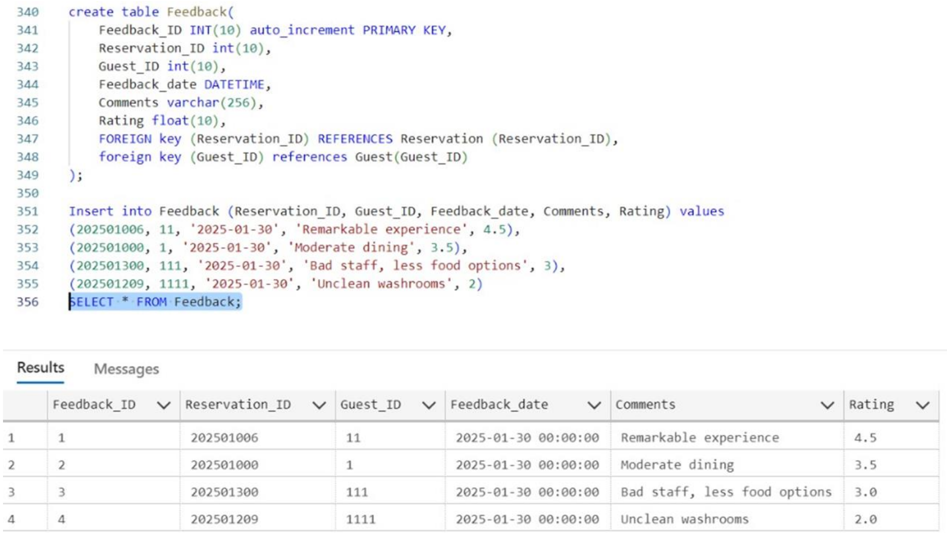


```sql
INSERT INTO Service (Service_ID, Service_Name, Description, Service_Price, Service_Status) VALUES
(90001, 'Breakfast', 'Complementary Breakfast', 0.00, 'opted'),
(90004, 'Beverages', 'Alchoholic drinks', 100.00, 'not opted'),
(90010, 'Jaquzzi', 'Spa and wellness', 150.50, 'opted'),
(90020, 'Transport', 'Airport Transfers', 65.00, 'not opted');

SELECT * FROM Service;
```

	Service_ID	Service_Name	Description	Service_Price	Service_Status
1	90001	Breakfast	Complementary Breakfast	0.0	opted
2	90004	Beverages	Alchoholic drinks	100.0	not opted
3	90010	Jaquzzi	Spa and wellness	150.5	opted
4	90020	Transport	Airport Transfers	65.0	not opted

Feedback Table:



```sql
create table Feedback(
    Feedback_ID INT(10) auto_increment PRIMARY KEY,
    Reservation_ID int(10),
    Guest_ID int(10),
    Feedback_date DATETIME,
    Comments varchar(256),
    Rating float(10),
    FOREIGN key (Reservation_ID) REFERENCES Reservation (Reservation_ID),
    foreign key (Guest_ID) references Guest(Guest_ID)
);

Insert into Feedback (Reservation_ID, Guest_ID, Feedback_date, Comments, Rating) values
(202501006, 11, '2025-01-30', 'Remarkable experience', 4.5),
(202501000, 1, '2025-01-30', 'Moderate dining', 3.5),
(202501300, 111, '2025-01-30', 'Bad staff, less food options', 3),
(202501209, 1111, '2025-01-30', 'Unclean washrooms', 2)
SELECT * FROM Feedback;
```

	Feedback_ID	Reservation_ID	Guest_ID	Feedback_date	Comments	Rating
1	1	202501006	11	2025-01-30 00:00:00	Remarkable experience	4.5
2	2	202501000	1	2025-01-30 00:00:00	Moderate dining	3.5
3	3	202501300	111	2025-01-30 00:00:00	Bad staff, less food options	3.0
4	4	202501209	1111	2025-01-30 00:00:00	Unclean washrooms	2.0