CASE STUDY

HOSPITALITY- DATABASE MANAGEMENT

Prepared by Ray Varghese

Table of Contents

1. Introduction

The Hospitality Industry is a versatile and dynamic sector that caters to providing best customer service to guests. This Industry focuses more on excellence and striving for customer service by exceeding the customer or guest expectations. This Industry has been striving and spanning across the globe with five-star hotel chains to budget friendly resorts which bring value for money. It's vital in the hospitality industry to have a robust and seamless booking or reservation system. Our hotel booking system is designed to enhance the overall guest experience, maintain efficient hotel management of staff and room infrastructure, offering personalized customer service and tailoring services adhering to customer preferences.

The database for a hotel is designed to serve the purpose of day-to-day transactions, streamlining their operations, and creating data-driven decisions. The database we have designed caters to provide memorable customer experiences by deploying an inclusive system where guests, staff, room inventory and services are interlinked to the reservation or booking status which in turn generates a payment module and a feedback module to generate invoices and also feedback portal to record guest experience ratings.

2. Mission Statement

Our mission is to deliver unforgettable, personalized experiences through innovative technology and sustainable practices. We aim to create lasting impressions-memorable moments by tailoring every stay to individual preferences and supporting local communities. By empowering our team with growth opportunities, we foster a culture of excellence and ensure exceptional service for every stay.

3. Business Objectives

- Improve Guest Engagement and streamline booking process: Use data-driven insights to create personalized guest interactions, enhancing loyalty and satisfaction. Integrate messaging systems to facilitate smooth communication between guests and staff for real-time assistance. Offer incentives or discounts for direct bookings through the hotel's own website to reduce dependency on third-party platforms.

- Boost Operational Efficiency: Automate repetitive tasks like billing and housekeeping requests to improve workflow and reduce human error and enhance data security by implementing robust security measures to safeguard the guest data by being compliant with privacy regulations.

- Improve team collaboration and development: Streamline the team or staff efforts to enhance operational efficiency and introduce staff development programs based on the feedback from clients. Integrate eco-friendly initiatives

such as energy-saving features and paperless check-ins to reduce the hotel's environmental impact so as to attain sustainable practices.

4. Entities and Tables identified related to the hospitality industry

The key aspects in the hospitality industry governs around the following entities

Guests- The customer who books the room expecting a remarkable experience

Rooms- This includes different room types, amenities and features

Staff- The team that supports the flow of services and customer experiences

Service- Various services like dining, food and beverages, preferences and customized experiences

Reservation or booking- The transaction, where all the other entities are linked together to form a booking or reservation.

Payment- Payment or Invoice which forms part of the booking process to involve the revenue and cashflow of the organization.

Feedback- The feedback from satisfied or dissatisfied clients which are graded as ratings and feedback left based on the services rendered.

5. TABLES and FIELDS

5.1 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)

Explanation: This table records guest details, including personal information (name, phone, email), location (address, country), and a unique identifier (Guest_ID). It likely supports systems like hotel management or event booking, with data types optimized for efficient storage.

- Guest_ID: The data type is integer values from 0-9 and serves as the primary key, uniquely identifying each guest. Key Icon: Indicates that this column is the primary key.
- First_Name: The data type is variable character of data length 30 characters and it stores the first name of the guest.
- Last_Name: the data type is variable character of data length 30 characters and it stores the last name of the guest.
- Phone: It stores the guest's phone number and has an integer data type with length 10. Limit: Holds numbers with a maximum of 10 digits (assumes integers only).
- Email: It stores the guest's email address with data type variable character with length of 50 characters
- Address: It stores the guest's full postal address with data type variable character of data length 256 characters.
- Country: It stores the country of the residence of the guest with a data type variable character of data length 30 characters.

5.2 ROOM TABLE:

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

Explanation: The "Room" table is designed to manage detailed information about the rooms in a system, such as for a hotel or property management platform.

- Room_ID: It acts as the primary key and avoids duplication of room records. It uniquely identifies each room in the database.
- Room_No: It represents the actual room number assigned within the facility.Allows for easy mapping of database entries to real-world rooms.
- Room_Type: Categorizes the room based on type, such as "Single", "Double", "Suite", or other descriptors. Helps in filtering and querying rooms based on guest requirements.

- Room_Price: It stores the cost of booking the room. Using a float data type supports pricing with decimal values, such as $100.50.
- No of Occupants: It stores the maximum number of occupants that can be accommodated in the room type. Data type is integer and it can have a data length of 10.
- Room_Status: This indicates the current state of the room (e.g., "Available", "Occupied", "Under Maintenance"). Useful for operational purposes, such as assigning rooms to guests or marking them for service.

5.3 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
Invoice_Amount	float(30)
Reservation_Status	varchar(20)

Explanation: Booking or transaction table that is assigned when reservation is confirmed.

- Reservation_ID: This serves as the primary key for the table which uniquely identifies each reservation and prevents duplication of reservation entries.
- Guest_ID: A foreign key linking to the "Guest" table. It represents the unique identifier for the guest making the reservation and helps to track reservations made by a specific guest.
- Service_ID: This is a foreign key linking to a "services" table. Refers to any additional services (e.g., spa, room service, or transportation) associated with the reservation. It also allows tracking of extra amenities.
- Room_ID: A foreign key referencing the "Room" table and specifies the room reserved by the guest. It ensures a valid room is associated with the reservation.

- Staff_ID: A foreign key referencing a "Staff" table and identifies the staff member managing or confirming the reservation. It also helps in assigning responsibilities for reservation handling.
- Booking_date: A timestamp field storing the exact date and time when the reservation was created. It is useful for auditing and reporting purposes.
- Check_in: A datetime field indicating the check-in date and time for the reservation. It ensures accurate planning for room availability.
- Check_out: A datetime field representing the check-out date and time. It helps to calculate the duration of stay and associated charges.
- Invoice Amount: It is the amount or price charged for the booking. It is a float data type with 2 decimal places with data length 10.
- Reservation_Status: A varchar(20) field indicating the current status of the reservation. Possible values might include "Confirmed," "Pending," "Cancelled," or "Completed." It is useful for tracking the reservation lifecycle.

5.4 PAYMENT TABLE:

Payment	
Payment_ID	integer(10)
Reservation_ID	integer(10)
Payment_date	timestamp
Payment_Method	varchar(20)
Payment_Status	varchar(20)

Explanation:

This Payment table is part of a relational database and is used to store information about financial transactions related to reservations. Below is a more detailed explanation of its structure, purpose, and functionality.

- Payment_ID: This is the primary key linked as a unique identifier to the reservation table. It has a data type integer with data length 10.

- Reservation_ID: It's a foreign key used in referencing the reservation table and links the payment to the reservation. It uses a datatype integer with length of 10.
- Payment_date (timestamp): It stores the date and time when the payment was made. Uses a timestamp data type for automatic date/time tracking.
- Payment_Method: It specifies the method used for payment (e.g., "Credit Card," "Cash," "PayPal"). It is stored as a varchar(20), meaning it can hold up to 20 characters.
- Payment_Status: This indicates the status of the payment (e.g., "Pending," "Completed," "Failed") and is stored as a varchar(20) to accommodate different status values.

5.5 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)

Explanation: The Staff table represents the structure of an employee database, storing details about staff members working in an organization. On the next page is a detailed explanation of each column:

- Staff_ID : This is a primary key and a unique identifier for each staff member. The Data Type is integer(10), meaning it stores numerical values up to 10 digits. It ensures each staff member has a unique ID. This field can be set as auto-increment, so each new employee gets a sequentially generated ID.
- Staff_FirstName : This field contains the first name of the staff member. The Data Type is varchar(30), meaning it can store up to 30 characters. Example Values: "John", "Alice", "David"

- Staff_LastName: This field stores the last name (surname) of the staff member. It uses a Data Type: varchar(30), meaning it can store up to 30 characters. Example Values: "Doe", "Smith", "Johnson"

- Staff_Phone: This field stores the contact phone number of the staff member. It uses a Data Type: integer(10), meaning it can store a 10-digit phone number. Example Values: 9876543210, 1234567890.

  Possible Issue: Phone numbers should ideally be stored as a VARCHAR(15) to support country codes (e.g., +1-9876543210).

- Staff_Email: This field stores the email address of the staff member. It uses a Data Type: varchar(50), allowing up to 50 characters. It is used for communication, login credentials, and notifications. Example Values: "john.doe@example.com" or "alice.smith@company.com"

- Staff_Address: This field stores the physical address of the staff member. This is useful for employee records, payroll processing and official documentation. It uses a datatype: varchar(256), meaning it can store up to 256 characters. Example Values: "123 Main Street, New York, NY" or "45B Baker Street, London"

- Staff_Role : This field defines the role or designation of the staff member in the organization and helps to define job responsibilities and authority levels. It uses a data type varchar(30), allowing up to 30 characters length. Example Values: "Manager" or "Receptionist" or "IT Technician"

- Staff_Dept : This field defines the department where the staff works and it uses a variable character (varchar) data type with 20 characters of length. Example Values: "HR" or “Maintenance”

5.6 SERVICE 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)

Explanation: The Service table keeps track of different services that a company provides, including their names, descriptions, pricing, and current status.

- Service_ID : This is the primary key and unique identifier for each service and ensures that there is no duplication of services associated with a reservation. It uses an integer datatype with a length of 10. This field is typically set for auto-increment, so each new service gets a sequentially generated ID.
- Service_Name: This field describes the service name and it uses a data type of variable character supporting upto 30 characters. Example Values: "Haircut" or "Car Wash" or "Software Installation"
- Service_Description: This field is a brief description of the service. It serves as a data description to understand what the service includes. It uses a datatype: varchar(256), meaning it can store up to 256 characters. Example Values: "Professional haircut and styling service" or "Complete interior and exterior car wash"
- Service_Price (float(10): This field defines the price of the service. It uses a data type float or decimal with 2 places allowing price to be accurately recorded. Example Values: 29.99 (for a haircut) or 49.50 (for a car wash).
- Service_Status : This field defines the current status of the service and it uses a data type variable character (varchar) and can accommodate a length of 30 characters. Example Values: "Available" or ”Temporarily Unavailable"

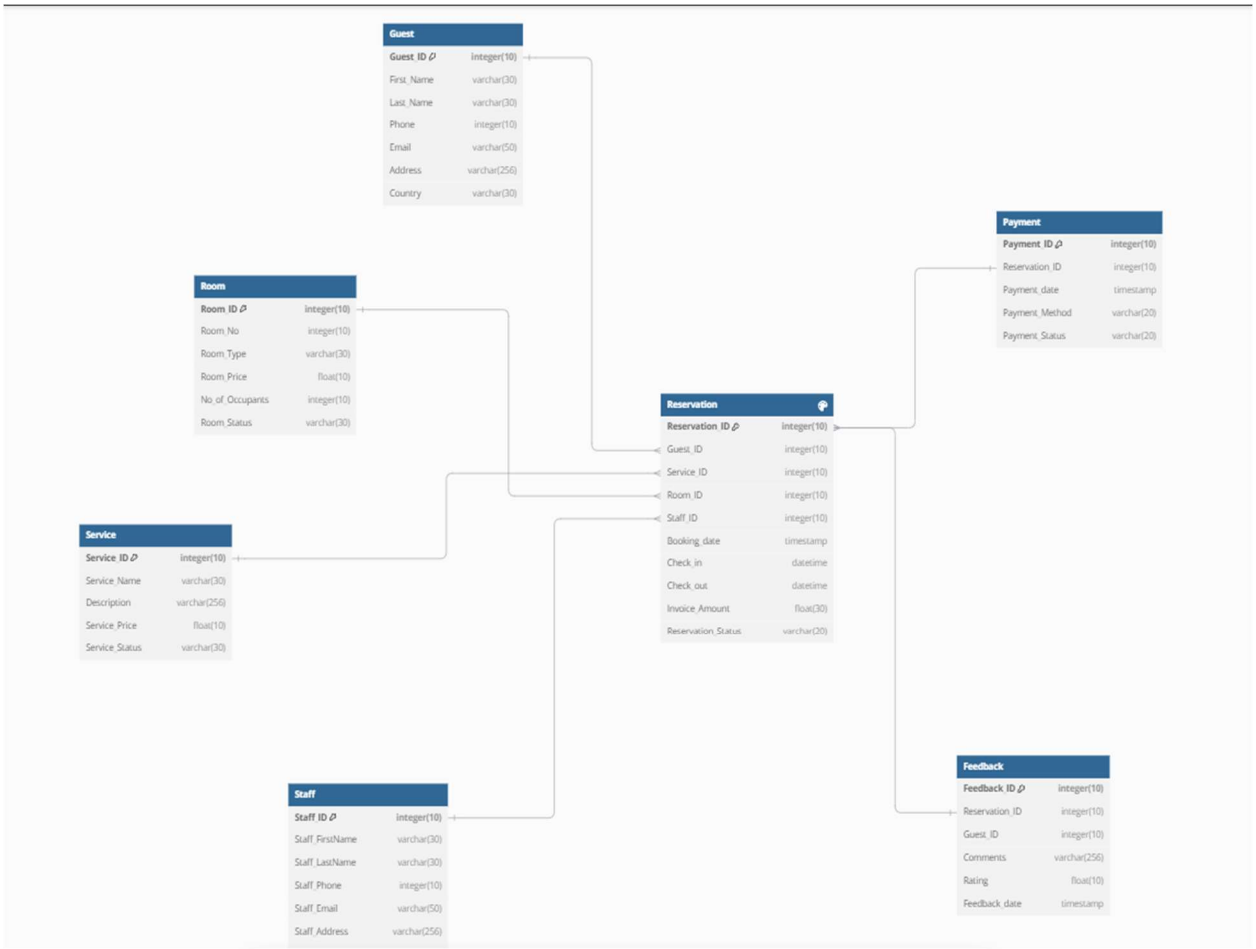
5.7 FEEDBACK TABLE:

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

Explanation: The Feedback table helps collect guest reviews, ratings, and comments related to their reservations. This information is useful for businesses to assess customer satisfaction and improve services.

- Feedback_ID : This field is a unique identifier for each feedback entry for each reservation made by a guest. It uses a datatype integer which stores numerical upto 10 integers of length. It's a primary key and ensures that each feedback entry has a unique ID. It's typically set to auto-increment, meaning each new feedback entry gets a sequentially generated ID.
- Reservation_ID: This is a foreign key that references a reservation table to track which reservation this feedback is related to and it forms a link on the reservation table. The data type used is integer with maximum length of 10 integers.
- Guest_ID: This field is also a reference foreign key that links the guest table with the feedback table and it serves to identify the guest who made the rating. It helps to track the guest who submitted the review or feedback. It uses an integer datatype with 10 integers as maximum length.
- Comments: This field captures and stores the customer feedback in text format. It captures qualitative feedback from guests. It uses a variable character (varchar) data type with 256 characters length. Example Values: "Great service! The room was clean and comfortable." Or "The food was excellent, but the waiting time was too long."
- Rating: This field is a numerical rating provided by the guest. It stores data as float(10) data type or decimal values with length 10. Example Values: 4.5 (out of 5)

6. ENTITY RELATIONSHIP DIAGRAM (ER Diagram)



This Entity-Relationship Diagram (ERD) represents a hotel reservation management system, detailing the relationships between different entities involved in the booking and management process

- The Guest table stores guest details, including their unique Guest_ID, name, phone number, email, address, and country. Each guest can make one or more reservations, which are recorded in the Reservation table. This table links guests to their booked rooms and services, assigning a Reservation_ID to each booking. It includes booking details like booking date, check-in/check-out dates, number of occupants, and reservation status. Each reservation is managed by a staff member (linked through Staff_ID).

- The Room table stores room details, including Room_ID, Room_No, Room_Type, Room_Price, and Room_Status. The Service table records additional services available to guests, such as spa, dining, or laundry, identified by Service_ID, with attributes including Service_Name, Description, Price, and Status.

- Payments for reservations are stored in the Payment table, which records each Payment_ID, the associated Reservation_ID, the Payment Date, Invoice Amount, Payment Method, and Payment Status.
- Customer feedback is stored in the Feedback table, where each Feedback_ID links back to a specific Reservation_ID and Guest_ID. It contains customer comments and ratings, providing insights into guest satisfaction.
- Lastly, the Staff table maintains records of hotel employees, each identified by a Staff_ID. It includes details like first name, last name, phone number, email, address, role, and department, helping manage staff responsibilities.

This database structure ensures efficient hotel operations, tracking guest stays, payments, services, and feedback while maintaining organized staff and room records.

7. ENTITY RELATIONSHIPS

7.1 Guests & Reservations:

The One-to-Many relationship between Guests and Reservations means that:

- A single guest can make multiple reservations over time. For example, a guest may book a room multiple times on different dates.
- Each reservation is linked to only one guest, ensuring that every booking is associated with a specific individual.

Foreign Key Implementation:

- The GuestID in the Reservations table acts as a foreign key referencing the GuestID in the Guests table.
- This enforces referential integrity, meaning a reservation cannot exist without a valid guest record.

7.2 Rooms and Reservations:

The One-to-Many relationship between Rooms and Reservations means that:

- A single room can be booked multiple times by different guests on different dates. For example, Room 101 can be reserved by multiple guests at different times.

- Each reservation is for only one specific room, ensuring that a booking is linked to a particular room at a given time.

Foreign Key Implementation:

- The RoomID in the Reservations table acts as a foreign key referencing the RoomID in the Rooms table.

- This maintains referential integrity, ensuring that a reservation is always associated with an existing room.

7.3 Reservations & Payments

The One-to-Many relationship between Reservations and Payments means that:

- A single reservation can have multiple payments associated with it. For example, a guest may pay in installments or make separate payments for additional services.

- Each payment is linked to only one reservation, ensuring that every transaction corresponds to a specific booking.

Foreign Key Implementation:

- The ReservationID in the Payments table serves as a foreign key, referencing the ReservationID in the Reservations table.

- This ensures referential integrity, meaning every payment must be associated with an existing reservation.

7.4 Reservations & Feedback

The One-to-One relationship between Reservations and Feedback means that:

- Each reservation can have only one feedback entry provided by the guest after their stay.

- Each feedback is linked to a specific reservation, ensuring that reviews and ratings correspond to the correct booking.

Foreign Key Implementation:

- The ReservationID in the Feedback table acts as a foreign key, referencing the ReservationID in the Reservations table.

- This maintains referential integrity, ensuring that feedback exists only for valid reservations.

7.5 Guests & Feedback

The One-to-Many relationship between Guests and Feedback means that:

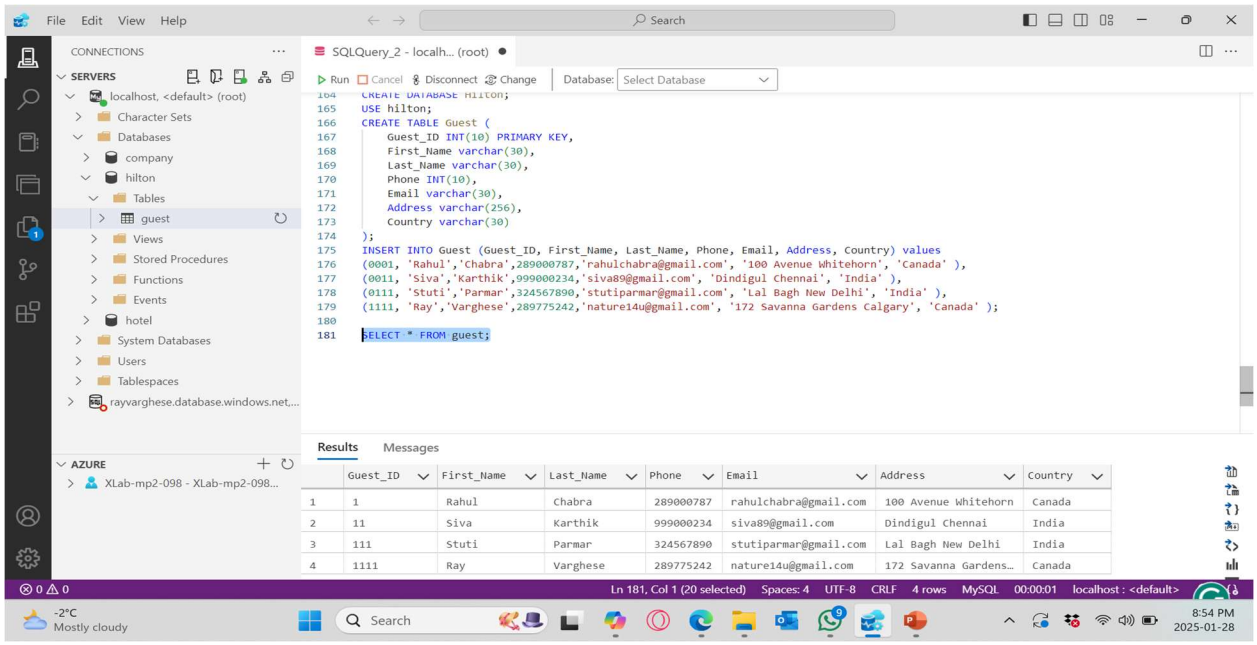
- A single guest can leave multiple feedback entries for different reservations (stays).
- Each feedback entry is linked to only one guest, ensuring that reviews are associated with the correct individual.

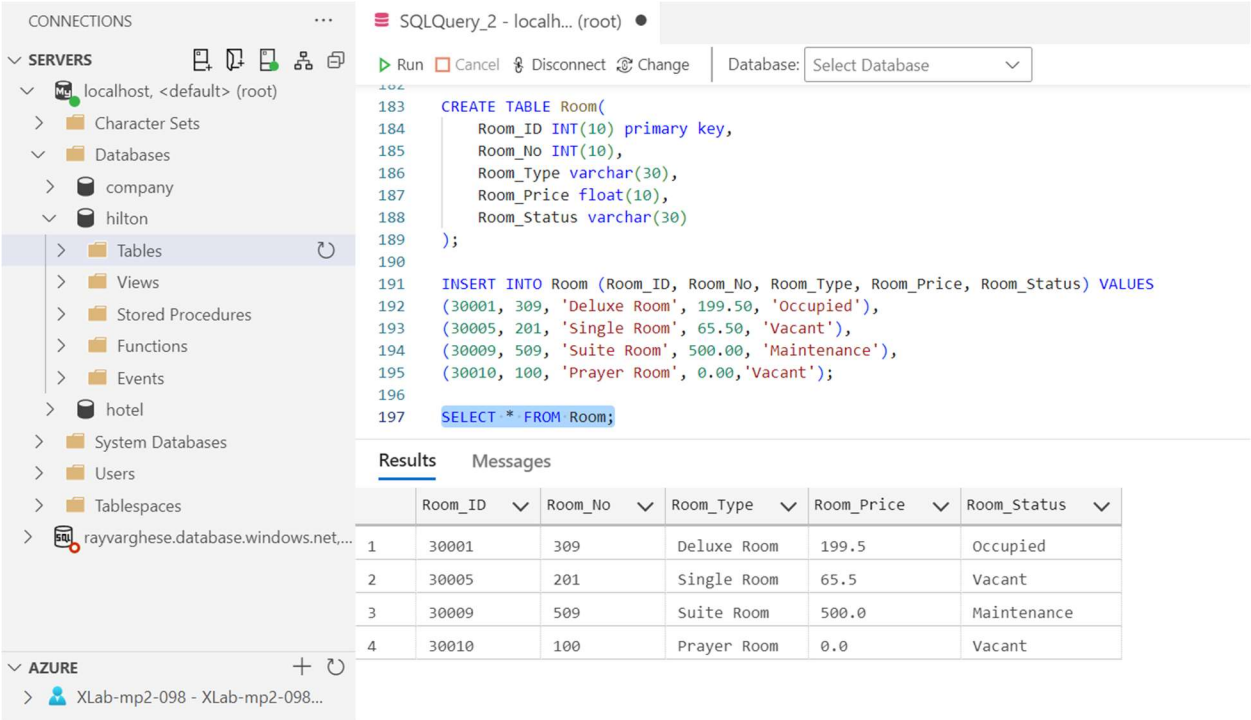
Foreign Key Implementation:

- The GuestID in the Feedback table serves as a foreign key, referencing the GuestID in the Guests table.
- This ensures that every feedback entry is tied to a valid guest, maintaining data integrity and allowing businesses to track guest experiences over multiple visits.

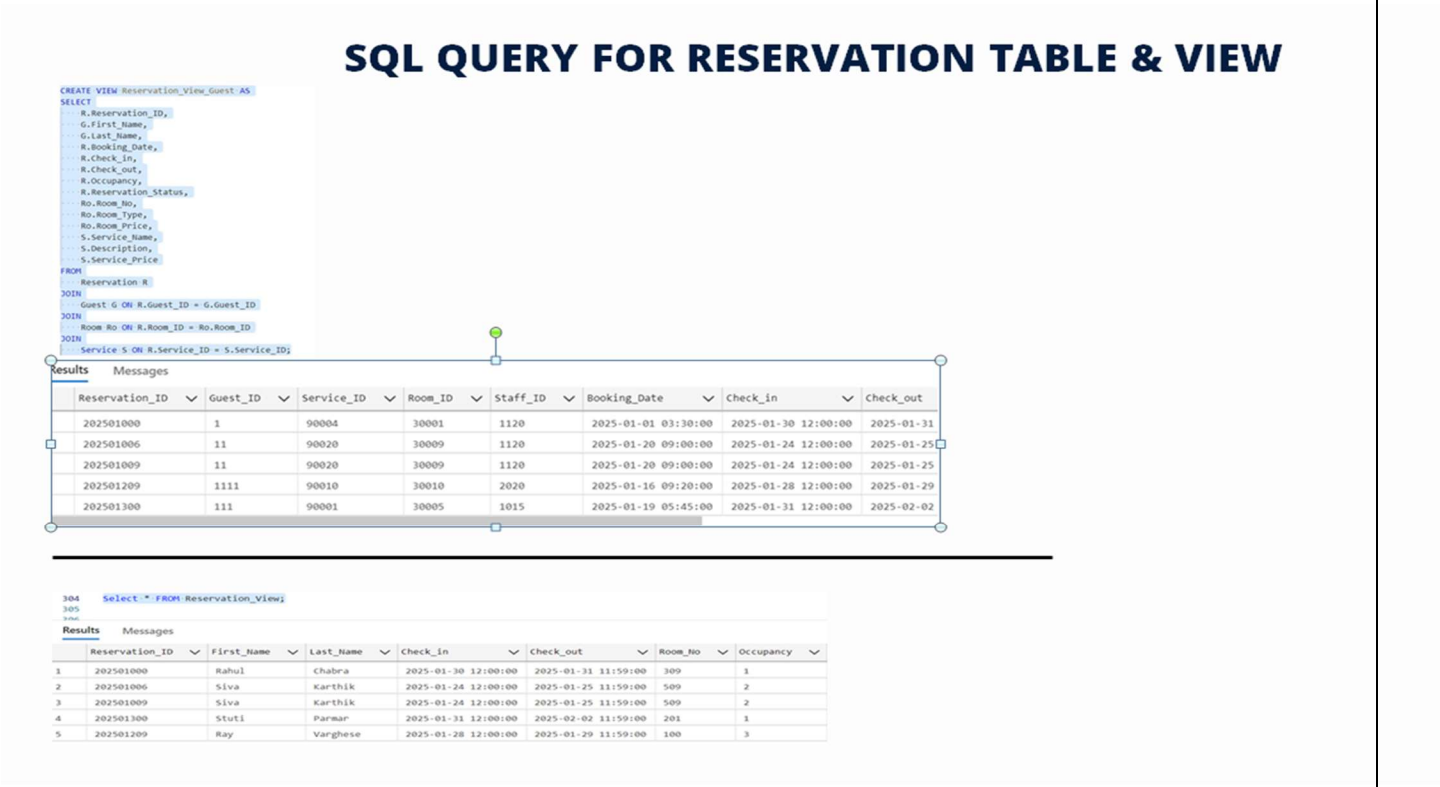
8. SQL Queries for Database design and VIEWS

Created database HILTON and all tables- Guest, Room, Staff, Service, Reservation, Payment and Feedback

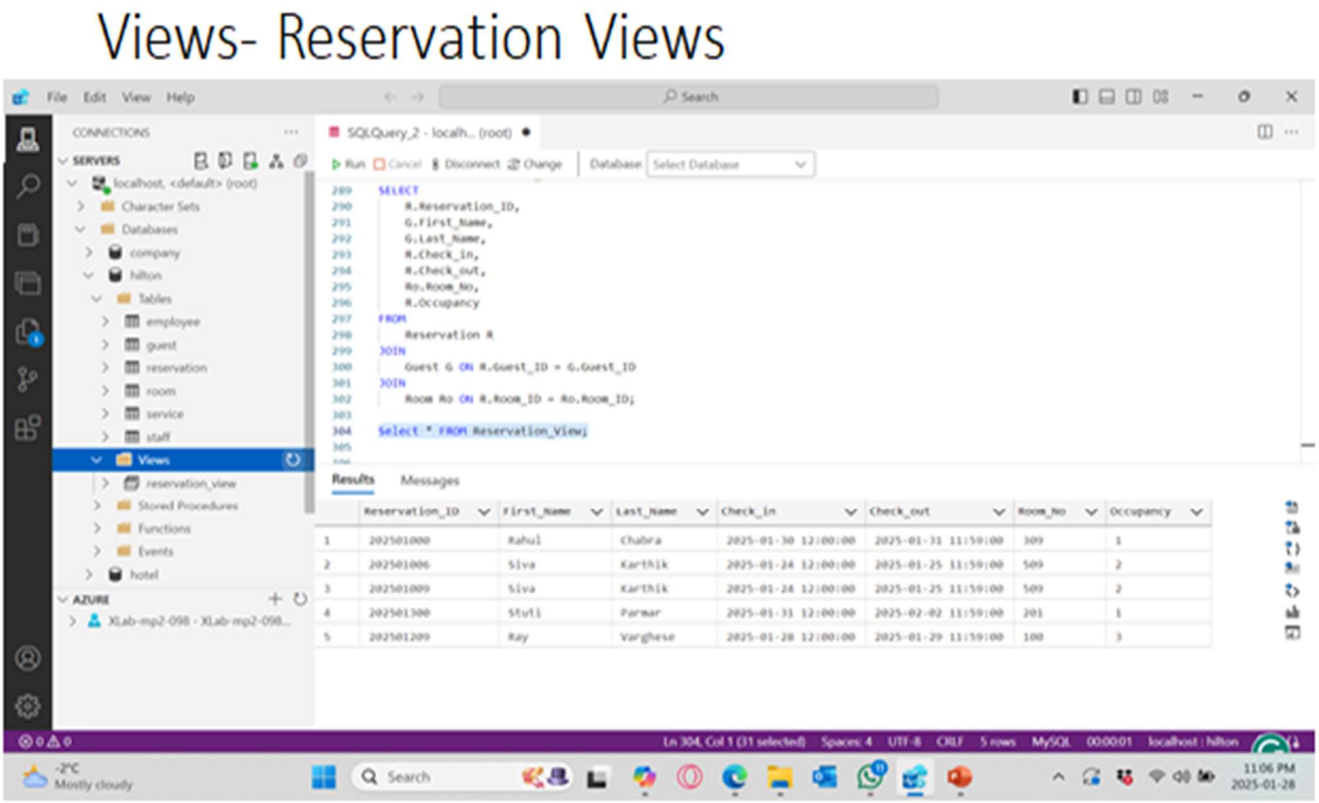


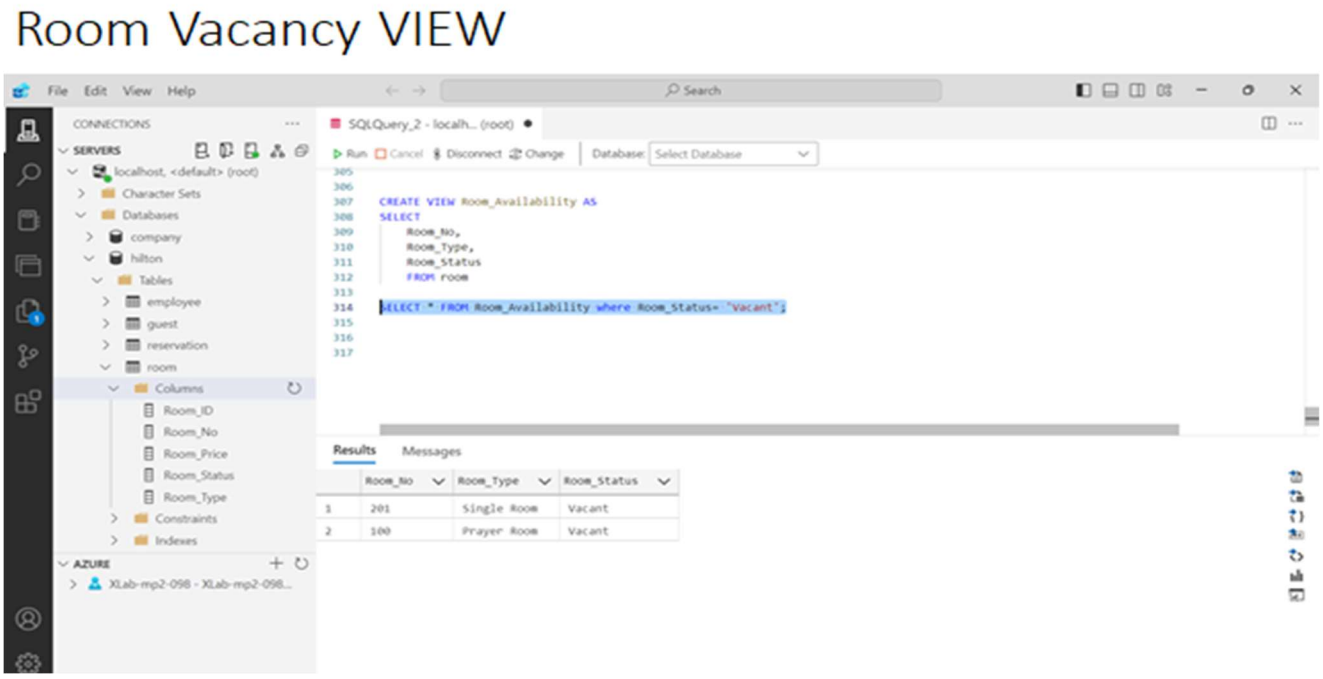


Using Inner Join created Reservation table view



Reservation table view using Join Query





SQL QUERY FOR PAYMENT TABLE & VIEW

```sql
create table Payment(
    Payment_ID INT(10) PRIMARY KEY,
    Reservation_ID int(10),
    Payment_date DATETIME,
    Invoice_Amount float(30),
    Payment_method varchar(20),
    Payment_Status varchar(20),
    FOREIGN key (Reservation_ID) REFERENCES Reservation (Reservation_ID)
);

INSERT INTO Payment (Payment_ID, Reservation_ID, Payment_date, Invoice_Amount, Payment_method, Payment_Status) values
(220000, 202501000, '2025-01-09', 200.00, 'Cash', 'Invoiced'),
(220003, 202501006, '2025-01-21', 100.00, 'Credit', 'Pending'),
(220002, 202501209, '2025-01-30', 66.50, 'Debit', 'Pending'),
(220005, 202501300, '2025-01-08', 99.00, 'Cash','Invoiced')

INSERT INTO Payment(Payment_ID, Payment_method) values (220001, 'Cash'), (220004, 'Debit'), (220004, "Credit"), (220007, 'Debit')

SELECT* FROM Payment where Payment_method is not null;
```

Results Messages

	Payment_ID	Reservation_ID	Payment_date	Invoice_Amount	Payment_method	Payment_Status
1	220000	202501000	2025-01-09 00:00:00	200.0	Cash	Invoiced
2	220002	202501209	2025-01-30 00:00:00	66.5	Debit	Pending
3	220003	202501006	2025-01-21 00:00:00	100.0	Credit	Pending
4	220005	202501300	2025-01-08 00:00:00	99.0	Cash	Invoiced

SQL QUERY FOR FEEDBACK VIEW

```sql
CREATE VIEW Feedback_view AS
SELECT
    R.Reservation_ID,
    G.First_Name,
    G.Last_Name,
    R.Check_in,
    R.Check_out,
    Ro.Room_No,
    F.Comments,
    F.Rating
FROM
    Reservation R
JOIN
    Guest G ON R.Guest_ID = G.Guest_ID
JOIN
    Room Ro ON R.Room_ID = Ro.Room_ID
JOIN
    Feedback F ON F.Reservation_ID = R.Reservation_ID;

Select * FROM feedback_view;
```

Results Messages

_ID	First_Name	Last_Name	Check_in	Check_out	Room_No	Comments	Rating
	Siva	Karthik	2025-01-24 12:00:00	2025-01-25 11:59:00	509	Remarkable experience	4.5
	Rahul	Chabra	2025-01-30 12:00:00	2025-01-31 11:59:00	309	Moderate dining	3.5
	Stuti	Parmar	2025-01-31 12:00:00	2025-02-02 11:59:00	201	Bad staff, less food options	3.0
	Ray	Varghese	2025-01-28 12:00:00	2025-01-29 11:59:00	100	Unclean washrooms	2.0

9. CONCLUSION

The Hotel Booking System enhances efficiency, sustainability, and customer satisfaction in the hospitality industry. It streamlines reservations, ensures secure transactions, and personalizes guest experiences. By promoting eco-friendly practices and improving operational efficiency, the system helps hotels attract more customers and stay competitive.