CHITKARA
UNIVERSITY

DBMS PROJECT (ST-1 Evaluation)
AUGUST-2025

Project Title: Hotel Booking System

Submitted By: Aneet Kaur (2410990838)
Armaanpreet Kaur (2410990855)
Arpita Sharma (2410990859)
Daman Preet Kaur (2410990115)

Group No: G-11

Batch: 2024-28

Submitted To: Sachin Garg Sir

Hotel Booking System

1. Introduction

Hotels are essential in hospitality industry, offering travelers accommodation, food, and related services. Traditionally, bookings were managed manually through registers or spreadsheets, which often led to errors, delays, and overbooking. This approach became inefficient as customer demands, and hotel operations grew.

With the rise of tourism and online services, guests now expect instant booking confirmations, accurate room availability, and quick check-in and check-out processes. Manual methods cannot keep up with these expectations, especially for hotels operating at larger scales or across multiple branches.

A **Hotel Booking System** provides a digital solution by centralizing reservations, customer records, room availability, staff management, and payments. It reduces errors, improves efficiency, and ensures smoother experience for both hotel staff and guests while optimizing resources.

2. Problem Statement

Hotels deal with multiple interconnected entities such as **Customers**, **Reservations, Rooms, Hotel_branch, Staff** and **Payments**. In the absence of a well-designed booking system, the following challenges arise:

- **Manual errors** in booking and cancellations.
- **Double-booking** of rooms due to lack of real-time updates.
- Difficulty in **tracking customer history** and preferences.
- **Inefficient management** of payments, check-ins, and check-outs.
- Lack of **centralized access** for multiple branches or online bookings.
- Limited ability to **generate reports** for occupancy, revenue, and performance.

A well-structured Hotel Booking System eliminates these challenges by offering real-time, centralized, and secure management.

3. Scope of the Project

The scope of this project is to design a relational database that will:

- Store all details of customers, rooms, reservations, staff, payments and hotel branch.
- Manage check-in, check-out, and room availability in real time.
- Integrate with payment systems to process cash, card, or online transactions securely.
- Maintain staff details and roles for better assignment of responsibilities.
- Enable multi-branch support for hotel chains to manage multiple locations.
- Handle different room types and pricing structures.

-

4. Objectives

The primary objectives of the Hotel Booking System are:

1. **Centralized Data Management**
   The system will maintain all hotel-related records such as customers, rooms, reservations, staff, and payments in a single integrated database. This ensures consistency, accuracy, and easier access to information across all departments.
2. **Efficient Reservation Handling**
   The booking process will be automated to handle reservations, cancellations, and room availability updates in real time. This helps prevent double-booking and improves customer satisfaction with instant confirmations.
3. **Customer Relationship Management**
   The system will track customer details, booking history, and preferences. This information can be used to build loyalty programs and provide personalized services for repeat guests.
4. **Secure Payments**
   The system will provide safe and transparent billing by recording all payment details accurately. It will support multiple payment modes while ensuring financial data security and integrity.
5. **Operational Efficiency**
   By automating repetitive tasks such as check-ins, check-outs, and reservation management, the system reduces manual workload for staff. This minimizes errors and allows employees to focus on customer service.
6. **Scalability**
   The system will be designed to support the addition of new branches, rooms, and services. This ensures the hotel can expand operations without major changes to the database structure.

5. Significance of the Project

The Hotel Booking System will significantly improve:

- Real-time tracking of room availability ensures optimal use of hotel resources, preventing both overbooking and underutilization.
- All transactions are recorded securely, ensuring accurate billing and protecting sensitive customer data through role-based access.
- Improve customer satisfaction with quick and accurate service.
- Ensure data security and accountability.
- Save time and costs by automating repetitive tasks.

Core Functional Requirements (Entities and Relationships):

1. **Customers**
   - Each customer has a unique ID, name, contact details, and address.

-
    - Customers can make multiple reservations over time.
    - Customers can have specific preferences stored.

2. **Hotel Branch**
    - Each branch has a unique ID, name, location, category and contact details.
    - A branch contains multiple rooms of different types.
    - A branch employs staff members for operations and management.

3. **Rooms**
    - Each room has a unique room ID, type, price, and availability status.
    - Rooms belong to a specific hotel branch.
    - Rooms can be reserved multiple times by different customers over time.

4. **Reservations**
    - Each reservation has a unique reservation ID, customer ID, room ID, checkin/check-out dates, and status.
    - A reservation is linked to one customer and one room.
    - Each reservation generates a corresponding payment record.

5. **Staff**
    - Each staff member has a unique ID, name, role (manager, receptionist, housekeeping), and contact details.
    - Staff are assigned to a specific hotel branch.
    - Staff may handle multiple reservations depending on their role.

6. **Payments**
    - Each payment has a unique pay ID, reservation ID, amount, pay mode, and pay date.
    - Payments are linked directly to reservations made by customers.
    - Each payment ensures that the booking is confirmed and recorded securely.

Relationships:

1. **Customer – Reservation**
    - Many-to-Many relationship.
    - Multiple customers can make many reservations, and many reservations can belong to multiple customers.
2. **Reservation – Room**
    - Many-to-one relationship.
    - Many reservations can be made for the same room (on different dates), but each reservation is linked to one specific room.

-

3. **Hotel Branch – Room**
    - One-to-Many relationship.
    - A hotel branch can have many rooms, but each room belongs to one hotel branch.

4. **Hotel Branch – Staff**
    - One-to-Many relationship.
    - A hotel branch can employ many staff members, but each staff member is assigned to one hotel branch.

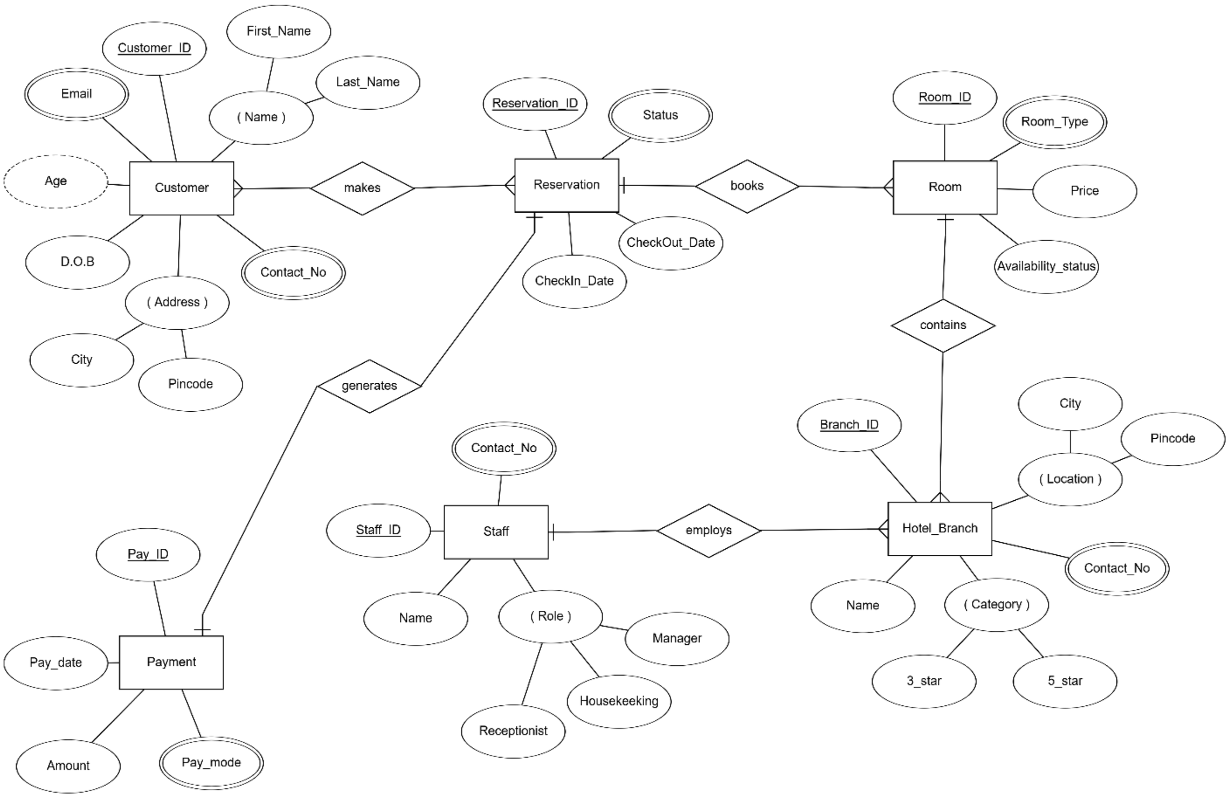
5. **Reservation – Payment**
    - One-to-One relationship.
    - Each reservation generates one payment, and each payment is linked to one reservation.

ER Diagram to Relational Model (Schema)

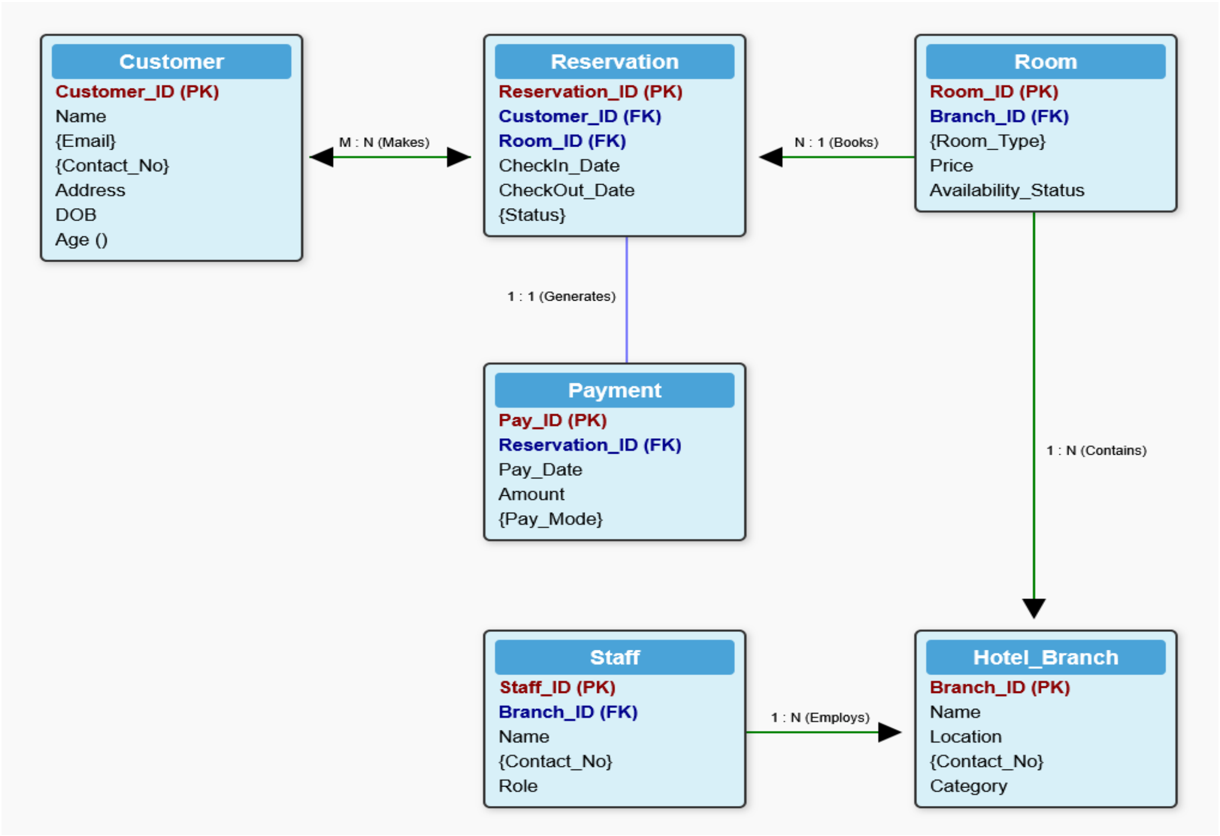
Entity	Attributes
Customer	Customer_ID (PK), First_Name, Last_Name, {Contact_No}, {Email}, City, Pincode, DOB, Age()
Room	Room_ID (PK), {Room_Type}, Price, Branch_ID(FK), Availability_Status
Hotel_Branch	Branch_ID (PK), Name, City, Pincode, {Contact_No}, 3_star, 5_star
Reservation	Reservation_ID (PK), Customer_ID (FK), Room_ID (FK), CheckIn_Date, CheckOut_Date, {Status}
Payment	Pay_ID (PK), Reservation_ID (FK), Amount, {Pay_Mode}, Pay_Date
Staff	Staff_ID (PK), Name, Receptionist, Manager, Housekeeping, {Contact_No}, Branch_ID (FK)

•

ER Diagram



ER Model



Relational Algebra (RA) Queries

1. Show all payments made by cash.
Ans: σ Pay_Mode = 'Cash'(Payment)

2. List details of all available rooms.
Ans: σ Availability_Status = 'Available'(Room)

3. Show payments greater than 5000.
Ans: σ Amount > 5000(Payment)

4. Show names and emails of all customers.
Ans: π Name, Email(Customer)

5. Get names of staff working as Receptionists.
Ans: π Name(σ Role = 'Receptionist'(Staff))

6. Show all hotel branch names.
Ans: π Name(Hotel_Branch)

7. List contact numbers of customers and staff.
Ans: π Contact_No(Customer) ∪ π Contact_No(Staff)

8. Show all names of customers and staff.
Ans: π Name(Customer) ∪ π Name(Staff)

9. Show rooms that were never reserved.
Ans: π Room_ID(Room) − π Room_ID(Reservation)

10. Show reservations that do not have payments.
Ans: π Reservation_ID(Reservation) − π Reservation_ID(Payment)

11. Show branches that have both staff and rooms.
Ans: π Branch_ID(Staff) ∩ π Branch_ID(Room)

12. Show rooms that are both “Suite" type and “Available.”
Ans: π Room_ID(σ Room_Type='Suite(Room)) ∩
π Room_ID(σ Availability_Status='Available'(Room))

13. Show all reservation–room pairs where Room_ID matches.
Ans: σ Reservation.Room_ID = Room.Room_ID (Reservation × Room)

14. Show all customer–reservation pairs where Customer_ID matches.
Ans: σ Customer.Customer_ID = Reservation.Customer_ID (Customer × Reservation)

15. Rename Customer as Guest and then join with Reservation.
Ans: ρ Guest(Customer)(σ Guest.Customer_ID = Reservation.Customer_ID (Guest × Reservation))

16. Find names of customers who have made reservations.
Ans: π Name(Customer ⋈ Reservation)

17. Assign all available rooms to relation X.
Ans: X ← σ Availability_Status='Available'(Room)

18. Assign customers from Delhi to relation Y.
Ans: Y ← σ Address='Delhi'(Customer)

19. Find customers who booked all types of rooms.
Ans: π Customer_ID, Room_Type(Reservation ⋈ Room) ÷ π Room_Type(Room)

20. Show staff IDs working in Branch_ID = 1.
Ans: σ Branch_ID = 1(Staff)

21. Show customers who never made a reservation.
Ans: π Customer_ID(Customer) − π Customer_ID(Reservation)

22. Show all reservation–payment pairs where Reservation_ID matches.
Ans: σ Reservation.Reservation_ID = Payment.Reservation_ID (Reservation × Payment)

23. Rename Staff as Employee and list their names.
Ans: π Name(ρ Employee(Staff))

24. Find customers who made reservations in all branches.
Ans: π Customer_ID, Branch_ID(Reservation ⋈ Room) ÷ π Branch_ID(Hotel_Branch)

25. Show all reservations with check-out date before 2025-12-31.
Ans: σ CheckOut_Date < 2025-12-31(Reservation)