Introduction to Data Management Processing
Project Report

Krithika Iyer
Rahul Chettri
Rohan Shriniwas Devasthale

Introduction

A hotel reservation and management system is a database design solution to streamline the process for reservations of hotels for room bookings, activity bookings and adding guest information, adding and viewing reviews and storing employee information. The primary objectives of such a system are to improve operational efficiency, enhance customer service, and provide accurate and timely management of hotel reservation and related data. The key components and objective of the system are described below :

- Relational Database that store information about rooms, customer, hotels, activities, payment and employee.
- Includes UI that allows user to login,view and update past reservation, book rooms and activities, review the hotel, view reviews of the hotels and logout the system.
- Includes roles (customer and admin) that allows to switch between application. Admin role allows to view employees, analytics of booked rooms, update salary of employees and complete the payment of the customers.
- System allows to efficiently manage room reservations, including checking availability, making new reservations, modifying existing bookings, and handling cancellations.
- System allows adding and viewing reviews for the hotel and room.
- System supports accurate billing and payment processes by tracking room rates, additional charges including activities, and payment details.
- System includes reporting tools to generate insights into hotel ratings and activity bookings.
- System is designed to accommodate scaling and growth of the hotel, addition of new rooms and services.

Data Collection

For this project we are populating the data ourselves and not using any dataset. As a result pre-processing or cleaning of data is not required for our application.

Database Design

Entity-Relationship Diagram and Tables

The below diagram shows the final Entity-Relationship diagram for the hotel reservation system.

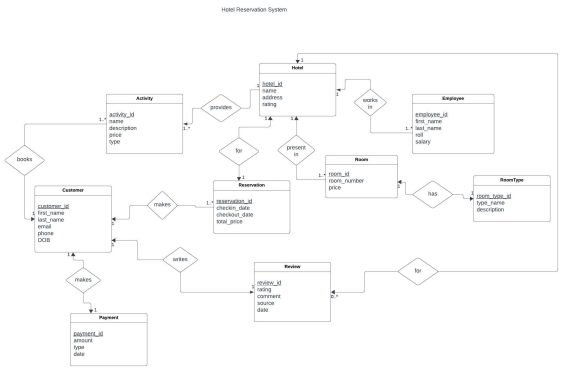

Figure 1: Entity-Relationship Diagram

The entities and the their relationships are captured by the following tables which we used in our project:

- customer(customerId, firstName, lastName, email, phone, DOB)
- hotel(hotelId, name, address, rating)
- room(roomId, roomNumber, roomTypeId, hotelId, price)
- roomType(roomTypeId, typeName, description)
- reservation(reservationId, hotelId, customerId, checkinDate, checkoutDate, guestCount, totalPrice)
- reservationDetails(reservationId, roomId, guestCount)
- activity(activityId, name, description, price, type)
- activityBooking(bookingId, activity_id, customer_id, date)
- employee(employeeId, name, role, DOB, salary, hotelId)
- payment(paymentId, customerId, amount, type, date)
- review(reviewId, customerId, hotelId, rating, comment, source)

- roomAvailability(availabilityId, roomId, hotelId, roomTypeId, isAvailable, date, cleaningStatus)
- auth(username, password)

Relationships with Entities

Here are some of the key relationships that the entites have with each other:

- Customer-Reservation: One customer can have multiple reservations
- Customer-ActivityBooking: One customer can book multiple activities
- Hotel-Room: One hotel has many rooms
- Customer-Reviews: One customer can leave multiple reviews for multiple hotels
- Room-RoomType; Each room belong to a specific room type
- Hotel-Activities: A hotel can support multiple activities
- Hotel-Employee: A hotel can have many employees

Key Design Decisions

- **Normalization to Achieve 3NF:** Tables are designed and decomposed to ensure they are in the Third Normal Form (3NF), minimizing data redundancy and potential anomalies. For example, the Reservation and ReservationDetails are created separately to avoid repetition of information. Further our database design is such that there is no partial dependency. For example, the Customer table has all the info related to customer like name, date of birth, contact details with customerId as primary key.
- **Primary Keys and Foreign Keys:** Each table includes a primary key for unique identification of records. Further, the foreign keys establish relationships between tables. For example CustomerId in Reservation table relates to Customer table
- **Flexibility and Scalability:** The database is designed to be flexible and scalable. For example: separate RoomType table allows easy addition or modification of room types without altering room-specific details.
- **Data Integrity and Consistency:** Constraints like primary keys, foreign keys unique and not null are applied to maintain data consistency and integrity along with enforcing referential integrity. For example, CustomerID in the Payment table references the Customer table to ensure payment records correspond to existing customers. The email id of customer in the customer table has not null constraint.
- **Usability and Readability:** All the Tables are named descriptively to enhance readability and understanding of their contents.
- **Entity Attributes and Relationships:** We have chosen attributes to reflect essential information for each entity (e.g., customer name, hotel address, room type).

Application Description

HotelReservation is a comprehensive hotel reservation management system, designed to streamline the process of managing bookings, customer details, and analytics for hotel businesses. Built with a MySQL database backend and an intuitive Python-based graphical user interface (GUI), this application provides a user-friendly experience for both hotel administrators and staff.

Key Features:

- **User Authentication:** Secure login system for employees, ensuring data security and role-based access control.
- **Customer Management:**
  - Features for registration, updating, and deletion of customer details.
  - Efficient handling of customer information
- **Reservations:**
  - Capability to book rooms for specific dates.
  - Options to cancel or modify existing reservations, enhancing customer convenience.
  - Real-time room availability checks based on dates and room types.
- **Payment Processing:**
  - System for recording and updating payments made by customers.
  - Seamless integration with reservation and customer management modules.
- **Review System:**
  - Platform for customers to provide feedback on their stay.
- **Employee Management:**
  - Management of employee details, roles, and schedules.
  - Streamlined process for assigning tasks and managing work hours.
- **Room Management:**
  - Detailed room management system to monitor room status, availability, and types.
  - Integration with the reservation system for efficient room allocation.

Technical Highlights:

- Robust MySQL backend for reliable data storage and management.
- User-friendly Python-based GUI for easy navigation and operation.
- Customizable Views for tailored data presentation and analysis.

Use Cases:

- Hotel administrators can oversee operations, manage employee schedules, and analyze customer feedback.

- Front desk staff can efficiently handle reservations, check-ins, check-outs, and customer queries.
- Housekeeping and maintenance staff can stay updated with room statuses and schedules.

Conclusion and Future Directions

The HotelManagement project was a valuable experience in integrating MySQL and Python to manage hotel operations effectively. Key learnings included the importance of real-time data processing for room bookings and customer management. Future enhancements would focus on adding inventory management for the hotel and developing a website for increased accessibility for the customers. For future students, my advice is to prioritize understanding user needs, maintain clean code, and embrace new technologies. This project highlighted the dynamic nature of data science and the importance of practical application in software development. For the source code you can check the github repository: https://github.com/rd4398/hotel-reservation-system