Airbnb Final Phase Abstract

Project Overview

This project focuses on developing a database management system for an Airbnb-like platform, enabling hosts to list properties, guests to make bookings, and admins to oversee activities and transactions. The database is designed to support comprehensive functionalities including user management, booking, reviews, payment processing, and maintenance, all while ensuring data integrity and streamlined operations.

Key Functionalities

- User Management
    i. **Admin:** Admins have the ability to create, update, and manage user roles, approve or reject listings, monitor bookings, handle disputes, and oversee payment transactions. They can also generate reports on platform activity and perform maintenance.
    ii. **Host:** Hosts can create and manage property listings, set prices, availability, and house rules, respond to guest bookings, and communicate with guests through a messaging system. Hosts also manage guest reviews, receive payments, and update profile and verification details.
    iii. **Guest:** Guests can browse listings, filter based on preferences, request or confirm bookings, make payments, communicate with hosts, and leave reviews and ratings.
    iv. **Messaging:** A built-in messaging system enables hosts and guests to communicate directly, allowing them to clarify booking details and provide real-time responses.

- Listings and Bookings
    i. Hosts can create property listings, update details such as title, description, and price, set availability, and add amenities.
    ii. The booking system allows guests to select properties, confirm available dates, and complete bookings through the platform's payment system, which automatically updates the availability of the property.

- Review and Rating System
    i. Guests can rate and review their experiences with hosts and properties, providing feedback that builds credibility and transparency on the platform.

- Payment and Transaction Management
    i. The payment processing system supports seamless transactions, allowing users to complete bookings, request refunds, and track the status of payments. Additionally, discount codes managed through the `Coupon` table can be applied to bookings.

- Admin Controls

i. Admins ensure the platform's integrity by approving or rejecting listings, handling disputes, monitoring booking and payment statuses, and generating activity reports to assess platform performance.

- Maintenance Management
  i. Maintenance functionality allows hosts and admins to schedule maintenance for specific listings, assign a status, and notify guests, ensuring properties remain in optimal condition.

Metadata and Database Structure

- Database Structure
  i. Number of Tables: 21 tables form the foundation of this database, with key tables including `**User**`, `**Host**`, `**Guest**`, `**Admin**`, `**Listing**`, `**Booking**`, `**Review**`, `**Payment**`, `**Message**`, `**Amenity**`, `**Coupon**`, and `**TransactionHistory**`.
  ii. Data Relationships: Foreign keys establish strong relationships across the database to enforce data integrity. For example:
  iii. User and Role Tables (Host, Guest, Admin): Each user is categorized with a specific role that determines access permissions.
  iv. Listing, Booking, and Payment Tables: Listings link to hosts, and each booking is associated with a listing, guest, and corresponding payment details.
  v. Review and Rating: Relationships are maintained between users and listings for reviews, allowing guests to review properties and hosts after their stay.
  vi. ER Diagram: An Entity-Relationship Diagram visualizes all entities and their relationships, aiding in understanding data flow and ensuring proper structure.

- Database Size and Entries
  i. Entries: Each table is populated with at least 20 records to support extensive testing and relationship validation across the system.
  ii. Data Volume: The database holds significant volumes of user profiles, listing data, booking transactions, payment details, and communication records, structured for efficient querying and optimized indexing for large-scale operations.

- SQL Documentation and Queries
  i. Table Definitions: SQL statements define each table with specific data types, constraints, and primary/foreign key relationships to ensure data accuracy and security.
  ii. Sample Queries: Queries are tailored to specific use cases, such as:
    a) Retrieve User Roles: A SQL query retrieves all user details, including their assigned role (Admin, Host, or Guest).
    b) Show Listings by City: A query fetches listing details, including host information and city details.

c) Display Booking and Payment Status: A detailed query lists each booking along with payment information.

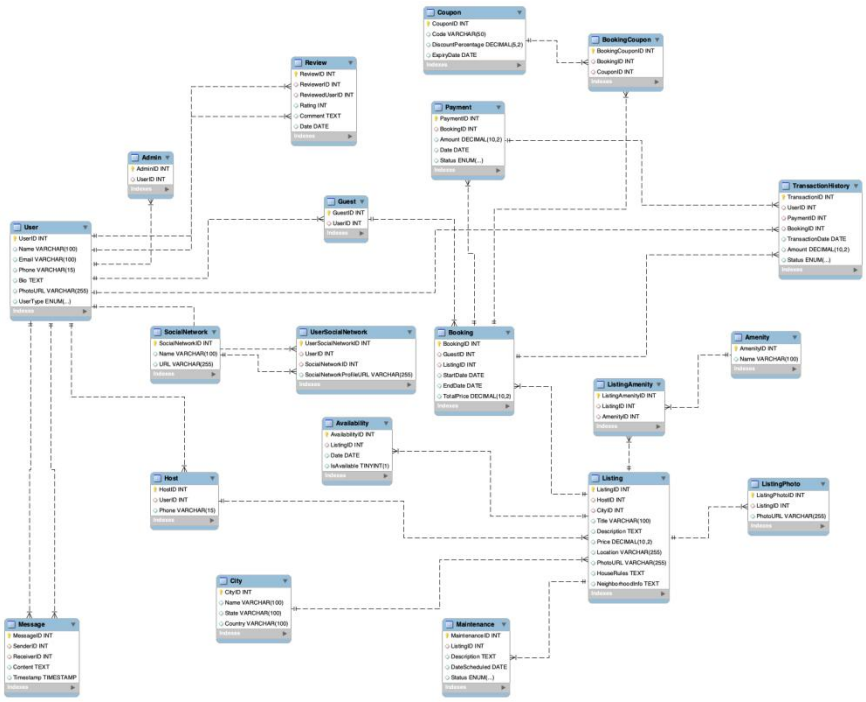
d) View Review Histories: Displays all reviews along with details on the reviewer and reviewed user.

iii. Test Cases: Each table and function has corresponding test cases to demonstrate usage. For instance, the `Booking` table test cases validate successful and failed booking scenarios, while the `Review` table tests ensure correct linking of reviews to listings and guests.

Final Submission Overview

- Files in Submission
  - i. SQL Files: SQL scripts for table creation, data insertion, and test queries covering each database function are included in the final package.
  - ii. Documentation: Each table is documented with a data dictionary explaining columns, data types, and relationships, along with sample queries to illustrate functionality.
  - iii. Presentation: Slides provide an overview of the database structure, key functionalities, ER diagram, and testing processes, showcasing the system's capabilities.
  - iv. Screenshots: Visuals from each phase, including ER diagrams, query results, and table definitions, demonstrate the database's evolution and final state.

- Integration of Feedback
  - i. Final improvements were made based on tutor feedback, including optimized indexing, improved data validation, and refined queries to streamline performance. Particular attention was given to enhancing data retrieval efficiency and ensuring a robust, scalable structure for all operations.



Conclusion

This project successfully implements a comprehensive database management system for an Airbnb-like platform, meeting the complex requirements of user, booking, and transaction management with a robust, scalable structure. Through the development of 21 interconnected tables, each supporting key functionalities such as user roles, listings, bookings, reviews, and payments, this database achieves efficient data organization and integrity.

The design emphasizes scalability, ensuring that the system can handle a large volume of data while supporting real-time communication, efficient querying, and secure transaction processing. The project also integrates tutor feedback, refining indexing, data validation, and query optimization to further enhance performance.

The final system demonstrates reliability and effectiveness in managing an online property rental platform, laying a solid foundation for future expansion or additional functionality. This project not only showcases practical skills in database design and SQL but also exemplifies how structured data management can support and optimize platform operations, delivering value to both hosts and guests in the rental ecosystem.