Analysis Report – Last Resort Hotel Database System

1. Business Overview

Since Last Resort Hotels (LRH) has rapidly expanded worldwide, it is necessary to get all information centralized for management and further potential analysis. Currently, much of the information could be divided into 8 sections: rooms, buildings, room availability, reservations, customers, events, and billing. The centralized database system needs to allow staff to access accurate, real-time information, improve customer service, and enable management to make data-driven decisions.

We separated the room availability and billing information from room and customer tables because room availability and billing information would change frequently while employees are updating real-time information, while basic information of room and customer would not be changing so frequently. On the webpage, employees can check the details of each reservation, event, and so on, while they are able to add new data entries into the database.

To support potential data analysis for operation improvement, the data system would have more tables than needed, since some information would be separated into tables for more efficient queries. For example, while we have a centralized table for all reservations, we also have separated tables for sleeping room and meeting room reservation.

2. Challenges

The foremost challenge is the wide range of data scope. For example, the international footprint of the hotel requires careful track of data about the address, complex structure, which depend on local convention. Therefore, it is hard to apply some naming conventions. For instance, in some countries, the length and the composition of postal codes is not a pure number series.

When it comes to dynamic management of the database, it is important to make sure the pipeline of inserting data, retrieving data, and analyzing data strictly adheres to the established rules.
Hard coding the data into the tables is not efficient when data grows larger. For example, in our system, the pricing of the rooms is determined by the extra charges and a base price that depends on the room type. This simplification is aimed to improve the efficiency while sacrificing some realistic assumptions (e.g. pricing can change with the floor, holiday).

Given the grand institution of this hotel, it is hard to make a large reasonable dataset to demonstrate the relationship between all tables in the database. For instance, 20 rooms and 20 customers could lead to hundreds of reservations, which may lead to even more assignments and payment entries. Therefore, we only focus on part of the dataset and well-develop it.

Meanwhile, since this system tracks data from various sources, we assume some data is not coming from employees keying in information with this system. For example, the data of reservation could come from various channels, like Airbnb or hotel official websites. However, we keep some access for employees to enter information if needed. There's an entry for customer and event, for example.

3. Objectives

The goals of the LRH database are to:

- Track all room types, buildings, floors, and facilities across the hotel complex.
- Manage reservations, check-ins, and check-outs for guests and event hosts.
- Record customer information, including preferences, membership types, and historical data.
- Store and monitor maintenance activities and room availability in real time.
- Record all services and transactions for accurate billing and reporting.
- Support events, conferences, and multi-room bookings with linked customer and payment records.
- Enable management queries and reports for occupancy, revenue, and customer activity trends.

4. Entity and Relationship Summary

The database model is based on a fully normalized Entity-Relationship Diagram (ERD) that captures the hotel's operational complexity.

- **Complex to Floor:** This section incorporates basic hierarchical tables establishing the physical hierarchy of the hotel's spaces: complex → building → wing → floor.

    - Although every wing has a unique id in the database, the name appearing in the webpage app is only uniquely designated within a building (e.g. many buildings will have wings called “North” and “South”). The wing name is used to designate rooms.
    - Every room has alphanumeric designations. For example, a room number of N1101 means that this room is the first room on the 11th floor in the North wing.

- **Room Types and Features:** This section defines the characteristics of each room, including smoking status, capacity, and rental rates.
    - Since the database could grow enormously with a hotel with business worldwide, we assume that the difference of bed (e.g. queen, king), room function (e.g. sleeping, meeting), and rental rate in different rooms are dependent on room type, rather than every single room.
    - This part has many bridging tables. For example, since many of the rooms can serve multiple functions, we created a bridging table to form the many-to-many relation between room type and room function.
    - The room number follows a pattern. The first letter means the wing to which the room belongs. Since the name of wings is only uniquely attributed to one building, it is common for the name of the wings from different buildings to overlap.

- **Customer and Customer Type:** Store information about guests, hosts, and organizations with links to reservations and billing. Since the billing party can be different from the party who uses the facility, we have attributes in both reservation and billing table to explicitly indicate the party. However, to be assigned to any reservation or billing record, the customer needs to be registered first.

- **Reservation and Room Assignment:** Record booking details for both sleeping rooms and meeting rooms.
    - We link all the information to reservation and treat reservation as the record for an “order”, which means that this table will be updated frequently. We have startDate and endDate to record the estimated/planned staying periods as the reservation is made, while checkInDate and checkOutDate are left blank until the customer is assigned a room or check out at the front desk.

- **Event:** Represents conferences, meetings, or other gatherings linked to rooms and hosts.

- **Billing, Payment, Deposit, and Transaction:** Track all financial activity, including split payments and charged services.

    - As described in the hotel description, customers can have different identities (billing party, using party), so we created a separate tag and combined the tag and customer ID in every single reservation and billing entries. It seems a bit burdensome, but to divide the responsibility of billing is important to enhance the efficiency of the system and hotel operation.

- **Room types:** include sleeping rooms, suites, and meeting rooms. Each room is defined by configurations, beds, smoking status, and other features like movable walls.
    - This part realized much of our assumption about the room and pricing.

- **Staff and Roles:** Maintain staff information, their assigned roles, and logged activities.

- **Maintenance and Room Status:** Keep real-time updates on repairs, cleaning, and availability.
    - In this part, many bridging tables are implemented to depict the relationship between the rooms while keeping the burden of the system minimized.

Each of these entities connects through foreign key relationships to maintain referential integrity and prevent redundancy. Bridging tables (e.g., customer_payment_relation, room_feature_relation) handle many-to-many relationships, ensuring flexible data modeling.

Relations overview:

- A Customer makes a Reservation for one or more Rooms.
- The Reservation may also include an Event (like a conference).
- The system generates a Billing record for that reservation.
- The Billing contains multiple Transactions, each linked to a Charged Service Type (e.g., room service, restaurant).
- The Customer makes Payments that are recorded in connection with the bill.
- If a Deposit was made in advance, it is linked directly to the bill.

- Staff members update room status, perform cleaning, and record activities related to the reservation.

5. Assumptions and Constraints

To simplify the implementation of the Last Resort Hotel database while maintaining realistic and functional accuracy, the team established a set of working assumptions and design constraints. These assumptions guided the development of the Entity-Relationship Diagram (ERD) and ensured that the system remained both efficient and practical to implement.

First, each room is assumed to belong to one specific wing, floor, and building within a single hotel complex. This hierarchical structure allows the database to maintain clear spatial relationships among facilities and ensures that room identifiers remain unique across the entire system. Second, every event is assigned one primary host who holds full financial responsibility for the booking, even when multiple guests or organizations are involved. This simplifies billing relationships and ensures accountability for event payments.

Additionally, while some rooms may serve multiple functions, such as operating as both sleeping quarters and meeting spaces, the system assumes that no room can be double-booked for overlapping time slots. This constraint prevents scheduling conflicts and maintains accurate availability tracking. The smoking status of each room is stored as a Boolean value, where “True” represents smoking and “False” represents non-smoking, allowing staff and guests to easily filter and assign rooms based on preference or regulation.

To handle complex financial scenarios, such as shared costs between multiple guests, the database includes a dedicated relation table that connects several customers to a single payment or reservation record. This approach supports flexible billing arrangements while maintaining a normalized structure. For maintenance operations, each recorded maintenance activity includes a clearly defined start and end date, ensuring that room availability can be accurately managed and updated in real time.

Finally, all database tables and relationships are designed to conform to Third Normal Form (3NF). This normalization standard eliminates redundant data, maintains referential integrity, and ensures that all non-key attributes depend solely on the primary key. These design principles create a robust, scalable system capable of handling the complex operations of a multi-property hotel network while maintaining clarity, accuracy, and performance.

6. Team Assignments

Our team consists of 4 members. We divided the whole database design into 8 parts (rooms, buildings, room availability, reservations, customers, events, and billing), each group member is responsible for 2 parts. While we are working on our own parts, we will work together for the relationships and connection between parts with shared platforms. For this business analysis report, we co-work on it by using a shared doc. For Sql, we will use Ed Workspace as the shared platform.

Vincent Chen: Responsible for event, billing, and consumer on ERD, SQL database design, and web design**.**

Lei Chen: Responsible for room availability on ERD, SQL database design and web design.

Chenyang Kong: Responsible for rooms and buildings on ERD, Sql database design, and web design.

Abel Ma: Reservation, Staff