DRAFT ANALYSIS REPORT

Tate Natland, Leonard Gu, & Livia Morales

Database Design & Implementation

23 October 2025

1. Business Overview

Last Resort Hotels has expanded quickly in recent years, growing from a single chain into an international network of luxury resorts. After acquiring Club Med and other high-end properties, the company now manages large, complex facilities with multiple buildings, wings, and thousands of rooms. Each hotel includes a mix of sleeping rooms, suites, and meeting spaces; some that can even serve dual purposes depending on the event. As these locations added new features like restaurants, pools, and conference centers, the task of managing reservations, billing, maintenance, and guest services became way too complicated for manual systems.

With so many moving parts (i.e. different room types, event spaces, services, and billing setups) the company needs a solid database system to keep everything organized. A database would let them track guest check-ins, event reservations, room statuses, and service charges in real time. It would also help staff provide better customer service by giving quick access to guest info, billing details, and facility availability. The system would turn a messy network of spreadsheets and manual updates into a centralized system where management has a bird's eye view of business logistics and can make more informed decisions.

2. Problem Statement

As with every large and complex business, there are difficulties in constructing an ideal database that can account for every single detail in the business while balancing rapidity in obtaining needed information. The main issues include the “lack of centralized information,” that is, the business description contains repeating or sometimes contradictory information (e.g. smoking status room dependent vs. floor dependent) from different sources around the business. In addition, the hotel’s system of managing rooms, billing information, and events requires a rather complex system of tables to replicate the structure of management as best as possible. There are some cases in which “reasonable assumptions” were needed to be made in order to simplify the database to ensure not only a comprehensive and concise database, but a fast one. However, these assumptions do not take away any important/core information from the description of management.

3. Objectives

The objective of the database for Last Resort Hotels is to create a centralized and normalized structure that accurately represents the company's complex hospitality operations. The database will be especially useful in allowing management and staff to efficiently store, organize and retrieve information related to hotels, buildings, wings, floors and rooms, ensuring that all physical facilities and their relationships are clearly defined. Furthermore, the database will support the need to monitor room availability, room types, bed configuration, amenities, renovation statuses, and smoking policies while maintaining accurate connection between rooms and potential adjacencies.

In addition to room management, the database will handle reservations, guests, and hosts across all hotels for both individual stays and large group events. The database will record a responsible billing party for each booking, allow for composite billing across multiple rooms, and track guest preferences such as bed type or room proximity. Each reservation will link to a responsible party and associated stay or event, ensuring that every use of rooms and facilities is properly documented. Events can be assigned specific hotel staff, and staff-event relationships are stored in the database to ensure that management can monitor who is assigned to each task, event, or department.

Furthermore, the database will record all services, charges, and transitions provided to guests organized by charge type to support detailed billing management. Each charged service, such as room stays, meeting room rentals, dining, and event-related fees, will be associated with the correct billing party. This system also captures task assignments for hotel staff ensuring that operational responsibilities are tracked.

4. Entity and Relationship Summary

* All assumptions are bolded

For starters, we have a “hotel” entity as the first sentence claims Last Resort Hotels owns several hotels worldwide, where the brand of the hotel correlates with the parents company (Last Resort Hotels) or its acquired assets (i.e. Club Med chain). Within each hotel is many buildings, and within each building is many wings, and within each wing is many floors which contain many rooms. This sequence explains our one to many relationships from hotel to building to wing to floor to room. A room can contain multiple beds, accounted for in the room_bed table; and each bed has a bed type (i.e. king, queen, etc.). Room amenities (i.e. toilet, TV) are linked via room_amenity, since amenities can vary across rooms. Our room type table differentiates between sleeping rooms, suites, and meeting rooms. All rooms are linked to status_type to indicate renovation, availability, or under cleaning. **All rooms on a floor are designated as smoking/non-smoking based on floor smoking status regardless of room type (suite, sleeping, meeting)**, so we can include the smokingPolicy attribute in only the floor table. Also, **room numbering is only unique within a wing,** thus room is linked to wing and not directly to building or hotel, since room numbering can become extremely repetitive across resorts. Wings have properties like nearPool, nearParking, handicapAccessible to reflect the idea wings differ by proximity to such amenities. **Meeting rooms all have the ability to serve eating usage (serve food).** Room adjacencies are reflected in the room_adjacency table which displays the roomId and adjacentRoomId that link to the corresponding roomId’s in the room table. **We assume room adjacency obeys the stated cardinalities.** A sleeping room may be adjacent to at most one meeting room, while a meeting room may have one or two adjacent sleeping rooms; sleeping rooms can also be adjacent to another sleeping room via a private access door.

A party is defined as the individual who is responsible for the payment of the reservation. The attributes of the party table include firstName, lastName, phone and email. The billing party can also be linked to a larger organization. If the responsible party is an organization, we store an individual contact record; sleeping-room charges may be split among guests in any proportion, but one responsible party remains on file. This applies to both individual stays and group reservations (events, meetings, etc.). Regardless of reservation type, there is one individual who is assigned to the task of paying for said reservation. Advanced deposits are sometimes required depending on various qualifications of the guest responsible for paying (partyId). These qualifications are represented in the customer_qualification table which provides a score and notes for the corresponding partyId. We assume that **every billing party is unique to the reservation they make. Outside (non-residential) guests must make a reservation before using a charged amenity or service**, such as dining, beverage, or other facilities, so the system can create a billing record linked to the responsible party. Additionally, we assume that **each reservation must include at least one charged service**, such as a room booking, event, or meal, to ensure a valid billing record exists in the system.

The stay table is only for sleeping room reservations and reflects a guest's actual occupancy period. In this way, a guest can switch rooms and still receive the composite bill. Events are hosted with one or more rooms and are associated with the reservation. Staff_event and task tables capture which staff are assigned to prepare or maintain rooms or events. **A reservation is recorded as either an event or a stay.** Both the event and the stay table link to the room table to record the room corresponding to the given reservation (either stay or event). **No staff are required to be at an event** in the case that an event may have outsourced staff, reflected in the optionality between the staff and staff_event table and the event and staff_event table. The staff_event table records the staffId and eventId that uniquely identify the assignment of a given staff member to a given event. **Events can occupy multiple rooms simultaneously**, and each event has one designated host, who may also be a billed party or guest. Lastly, **every meeting room is assumed to have the capability to host food and beverage services** as we track the room occupancy and guest count at every event.

Not all staff need to have a task assigned to them, reflected in the relationship between the staff table and the task table. For example, a staff member could provide a service not recorded as a task (such as repair or maintenance), or a staff member could be newly hired and have no tasks yet. Additionally, **staff members can be associated with multiple events or rooms depending on their role**. For instance, housekeeping and maintenance staff may be assigned to specific rooms, while event staff are linked to particular functions or meetings through the event_staff table. We store staff department/role (e.g., housekeeping, maintenance, event ops) to support assignment to rooms vs. events without requiring every staffer to hold an active task at all times.

During the usage of facilities and their associated reservations, there is information that is kept track regarding said usage and permission of facilities. Each guest is given a plastic card with a PIN. The card allows the guest to gain access to their sleeping room and meeting rooms. It is assumed that **the permissions of the card are already encoded/stored in a separate backend that is able to distinguish permissions for each type of user (guest, staff, admin, etc.).** Guests also have the option to allow the hotel to track their location in the hotel. Since this poses a security issue and since the guests have the discretion to give the hotel this permission, it is assumed that **all customers do not want their location to be tracked around the hotel.** Thus, the need to track every guest's location is not included in the ERD.

Guests also have the ability to leave messages, which is accounted for in the message table with attributes of messageId, FK guestId (to associate with the specific guest) and the message contents. **Qualifications are evaluated per party.** If the party is an organization, the organization is primary; if an individual is linked to an org, the system uses the more conservative (i.e., stricter) of the two.

5. Assumptions and Constraints

1. All rooms on a floor are designated as smoking/non-smoking based on floor smoking status regardless of room type (suite, sleeping, meeting)
2. Room numbering is only unique within a wing
3. Meeting rooms all have the ability to serve eating usage (serve food)
4. We assume room adjacency obeys the stated cardinalities
5. Billing party is unique to the reservation they make
6. Outside (non-residential) guests must make a reservation before using a charged amenity or service
7. The stay table is only for sleeping room reservations and reflects a guest's actual occupancy period
8. A reservation is recorded as either an event or a stay
9. No staff are required to be at an event
10. Staff members can be associated with multiple events or rooms depending on their role
11. Events can occupy multiple rooms simultaneously
12. Every meeting room is assumed to have the capability to host food and beverage services
13. Not all staff need to have a task assigned to them
14. Staff members can be associated with multiple events or rooms depending on their role
15. The permissions of the card are already encoded/stored in a separate backend that is able to distinguish permissions for each type of user (guest, staff, admin, etc.)
16. All customers do not want their location to be tracked around the hotel
17. Qualifications are evaluated per party

6. Team Assignments

We considered splitting up the description and working on it separately but given the complicated nature, we decided to approach the ERD together. We held several 2 hour (ish) meetings of active collaboration where we ran through the description line by line from top to bottom. When approaching the analysis, we all choose some pre-written assumptions we made when creating the ERD and describe them fully (linked them to the ERD and relationships). Overall, this collaborative approach ensured that our ERD was consistent, accurate, and grounded in the requirements of the case.

7. Next Steps

Our next steps will be focused on turning out ERD into a working database and preparing for our next milestone submission. First, we’ll finalize the schema in MySQL, making sure all tables, primary keys, and foreign keys properly reflect the relationships outlined in our ERD. Once that’s set, we’ll populate the database with realistic sample data: at least 75 customers, 150 reservations, and enough records for rooms, events, and staff to run meaningful queries.

After the database is populated, we’ll begin writing SQL queries to generate management-level insights, such as occupancy rates, top-spending guests, and revenue trends over time. We’ll also start outlining the basic web interface structure in PyCharm, focusing on how data from our queries will display on a webpage. By the next milestone, we plan to have the database built, data inserted, and a few test queries running successfully.