Hotel Reservation Database Design

OU Lefeng, LAW Hoi, ZHU Jun, ZHAO Yuhao, LIN Lixin

· Department of Artificial Intelligence and Business Analytics, Lingnan University, Tuen Mun district of New Territories, Hong Kong

Abstract

Booking.com, a global online accommodation booking platform, offers user-friendly and easy-to-use booking services. It connects millions of accommodation providers and travellers around the world, offering a wide range of options from budget hostels to luxury hotels. The aim of this report is to design a database system similar to Booking.com's business requirements. The goal is to create a database architecture that can handle highly concurrent accesses while maintaining data consistency and security. The report will cover requirements analysis, conceptual design, logical design and physical design of the database.

Keyword: Diagram, ER, database model, SQL, Source code

1 Introduction

1.1 Background and Motivation

With the development of the global economy and the improvement of people's living standards, the tourism industry is experiencing an unprecedented boom. According to statistics, the number of international tourists is increasing year by year, which has led to a surge in demand for tourism-related services. Founded in 1996 in the Netherlands, Booking.com has grown to become one of the most popular travel websites in the world. The platform provides users with access to millions of listings, complemented by user reviews, travel guides, and price comparisons. Its user-friendly interface, abundant accommodation options, and real-time price comparisons greatly improve the booking efficiency and satisfaction of users, making it the go-to resource for travelers looking for convenient and reliable booking options.

1.2 Reason and Significance

In this context, it is particularly important to design an efficient hotel reservation database system. Booking.com was chosen as the subject of this study because of its significant impact on the travel industry and its innovative approach to online booking. The platform's wide selection and user-centric features make it an ideal case for researching travel booking apps. In addition, as people increasingly rely on digital solutions for travel planning, the relevance of studying such platforms in today's travel environment is even more prominent. The system is capable of handling a large number of concurrent accesses, ensuring data accuracy and consistency while meeting the needs of different user roles. Users want to be able to quickly find accommodations that match their preferences, and platform staff need to effectively manage user data and order processing.

1.3 Target users

Leisure travelers: Individuals and families looking for vacation accommodation.

Business travelers: Professionals looking for convenient accommodations on business trips.

Travelers on a budget: People looking for budget accommodations, such as hostels and budget hotels.

1.4 Function

The hotel booking database system is designed to provide an efficient, secure, and user-friendly platform for users to quickly find and book all types of accommodation. The system supports user registration and management, allowing users to search by destination and preferences, view detailed hotel information, and complete secure bookings and payments. Users can rate and review hotels after check-in, facilitating decision-making by other users. At the same time, the system enhances the user's booking experience through promotion and offer management. Role-based access control ensures permission management for different user roles. The system also provides powerful data query and visual analysis functions to help administrators optimize their business. Through strict data integrity and transaction management, the system ensures security and consistency in high-concurrency situations, providing a comprehensive solution for users and managers.

2 Requirements Analysis

2.1 Business Process

User registration: the user creates an account and provides the necessary personal information.

Search for Accommodation: User searches for available accommodation based on destination, dates and preferences.

Booking Process: User selects accommodation, completes booking and makes payment.

User Rating: User can rate the accommodation after the stay.

2.2 Key business entities

Users: including user ID, login information, email, address, etc.

Hotel: including price, facilities, user rating, availability, etc.

Room Type: including room type ID, area, capacity, total number, etc.

Order: including order number, user ID, creation and payment time, check-in and check-out date, price, quantity, etc.

Evaluation: including evaluation content, rating, evaluation time, evaluation user ID, etc.

2.3 Data type

Personally Identifiable Information: name, address, email, phone number, passport information, etc.

Accommodation information: type, price, facilities, location, number of rooms, user reviews, etc.

Booking details: booking number, user ID, accommodation ID, check-in and check-out dates, price,etc.

2.4 Date Integrity and Consistenncy Requirements

Booking status consistency: booking status must be updated synchronously with payment status.

Data accuracy: user information and accommodation information must be accurate andtimely updated.

Data Integrity: Ensure the data integrity of all booking and payment operations to prevent data loss.

2.5 Needs of Different User Roles

Customers: need fast, reliable search and booking processes and an intuitive user interface.

Platform staff: need to access the backend system to manage user data, handle booking issues and monitor business performance.

2.6 Permission and Access Control Requirements

Role-Based Access Control: Design the RBAC (Role-Based Access Control) system to ensure that users with different roles can only access the corresponding data.

3 Diagram Design

3.1 Design Concepts

The idea behind the design of this ER diagram is to create a comprehensive data model for managing the various entities in a hotel reservation system and the relationships between them. It allows the system to store and manage user information, hotel information, room information, order information, promotions, user reviews, and hotel pictures. It also maintains the scalability and maintainability of the system to adapt to changing business needs. With this design, the system can efficiently organise and retrieve data to support a variety of business operations such as booking rooms, managing user rights, processing orders and promotions, and collecting user feedback. The concept of data flow is emphasised, providing a visual representation of the flow of data through the system through arrows and labelling on the relational lines. This design facilitates the understanding of the sources and processing of data.

3.2 Relationship modelling and Attributes

Users:

Attributes: Users_id,Users_name. Passwords, Mailbox, PhoneNum, Payment_info, Addr;

Relationships: There is a many-to-one relationship between Users and Permission, indicating that a user can have multiple permissions.

Permission:

Attributes: Permission_id, Permission_desc;

Relationships: The Permissions entity defines the different permissions in the system, each with a description.

Hotel:

Attributes: Hotel_id, Hotel_name, Hotel_star, Hotel_Addr, Facilities;

Relationships: There is a many-to-one relationship between hotels and cities (City), indicating that a city can have more than one hotel. There is a one-to-many relationship between hotels and pictures (Pictures), indicating that a hotel can have more than one picture displayed.

Room:

Attributes: Total_count, Room_id, Room_name, Bedtype, Capacity, Price, Area, Descriptions;

Relationships: There is a many-to-one relationship between rooms and hotels, indicating that a hotel can have more than one room.

Orders:

Attributes: Orders_id, create_day, Payment_day, Progress;

Relationships: There is a one-to-many relationship between an order and a user, indicating that a user can create multiple orders. There is a one-to-one relationship between an order and a room, indicating that only one room can be booked for an order.

Promotion:

Attributes: Discount_ID, Descriptions, Start_time, End_time, DisPercent;

Relationships: There is a one-to-many relationship between promotions and orders, indicating that a promotion can be applied to multiple orders.

Review:

Attributes: Review_id, Rating, Comment_Content;

Relationships: There is a one-to-many relationship between a comment and a user, indicating that a user can post more than one comment. There is a many-to-one relationship between reviews and hotels, indicating that a hotel can receive multiple reviews.

City:

Attributes: City_id, City_name;

Relationships: Easy to pinpoint exactly which city the hotel is in

Pictures:

Attributes: Image_id, Image_addr, Image_desc;

Relationships: Through the picture this entity can facilitate the user to view the hotel facilities, environment and etc.

OrderDetail:

Attributes: Counts, Checkindate, Checkoutdate;

Relationships: By creating the weak entity of order information it is possible to better store information that the user wants to know more about the order.

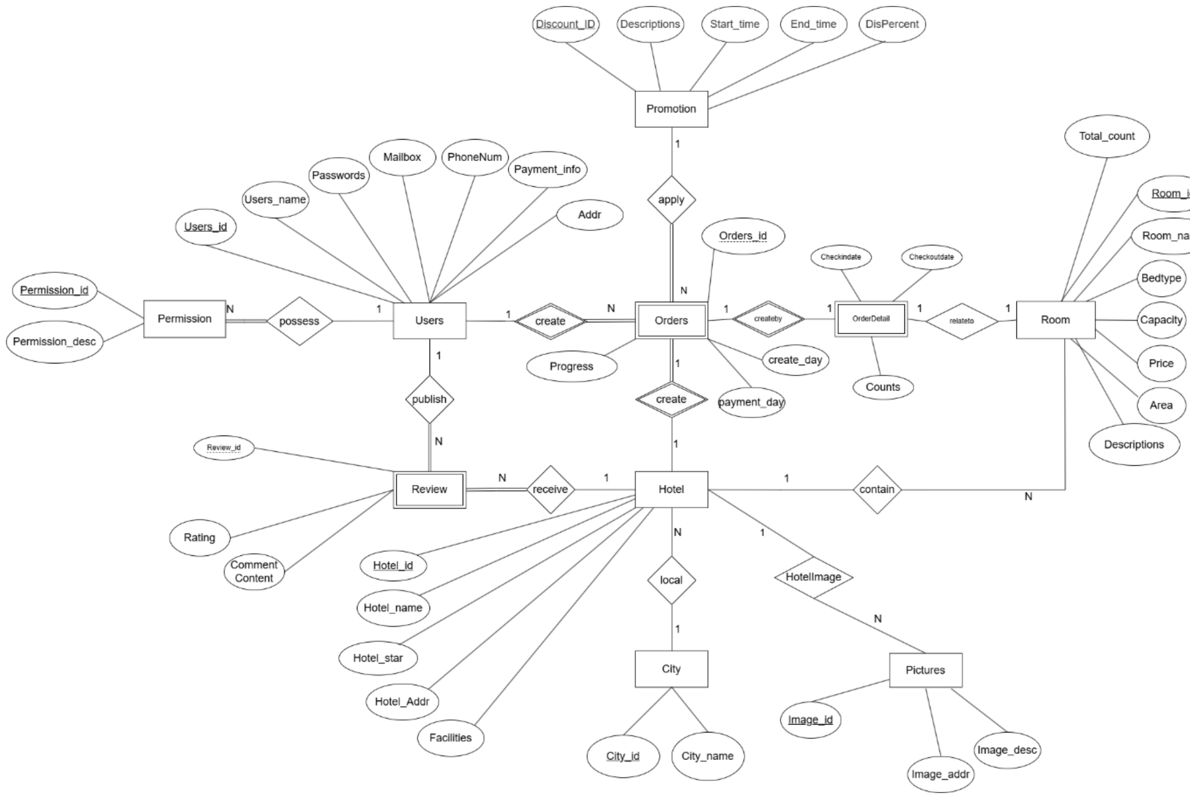


Figure 1 Hotel ER Diagram

4 Database Design

4.1 Database Design Concept

It aims to build a hotel reservation system that is fully functional, user-friendly, data consistent and easy to maintain and extend. It achieves data consistency, integrity and flexibility through careful organisation and definition of relationships between entities and weak entities. The system not only supports territory management and user search, but also enhances sales flexibility through automated order processing and promotions. A user-involved commenting system adds transparency, while rich photo displays enhance user experience. The modular design facilitates the addition of new features while keeping the system scalable and maintainable to adapt to changing business needs. Focusing on the relationships between entities and the structure of the data, it shows how the data is organised through the association of primary and foreign keys. This design helps to understand how data is stored and retrieved.

4.2 Primary and Foreign Keys

Permission: Permission_id is the primary key that uniquely identifies the different permissions.
Users: Users_id is the primary key that uniquely identifies the user. Permission_id is a foreign key, associated with the Permission table, indicating the user's permission level.
Promotion: Discount_ID is the primary key that uniquely identifies the promotion.
Hotel: Hotel_id is the primary key that uniquely identifies the hotel. City_id is a foreign key associated with the City table, indicating the city where the hotel is located

Room: Room_id is the primary key that uniquely identifies the room. Hotel_id is a foreign key that is associated with the Hotel table and indicates the hotel to which the room belongs.
Orders: Orders_id is the primary key that uniquely identifies the order. Users_id and Hotel_id are foreign keys associated with the Users and Hotel tables, representing the user and hotel of the order, respectively.
OrderDetail: Orders_id and Room_id are foreign keys associated with the Orders and Room tables, respectively, representing the room information contained in the order.
Review: Review_id is the primary key that uniquely identifies the review. Users_id and Hotel_id are foreign keys associated with the Users and Hotel tables, representing the user and hotel of the review, respectively.
Pictures: Image_id is the primary key that uniquely identifies the image. Hotel_id is a foreign key associated with the Hotel table, indicating the hotel to which the image belongs.
City: City_id is the primary key that uniquely identifies the city.

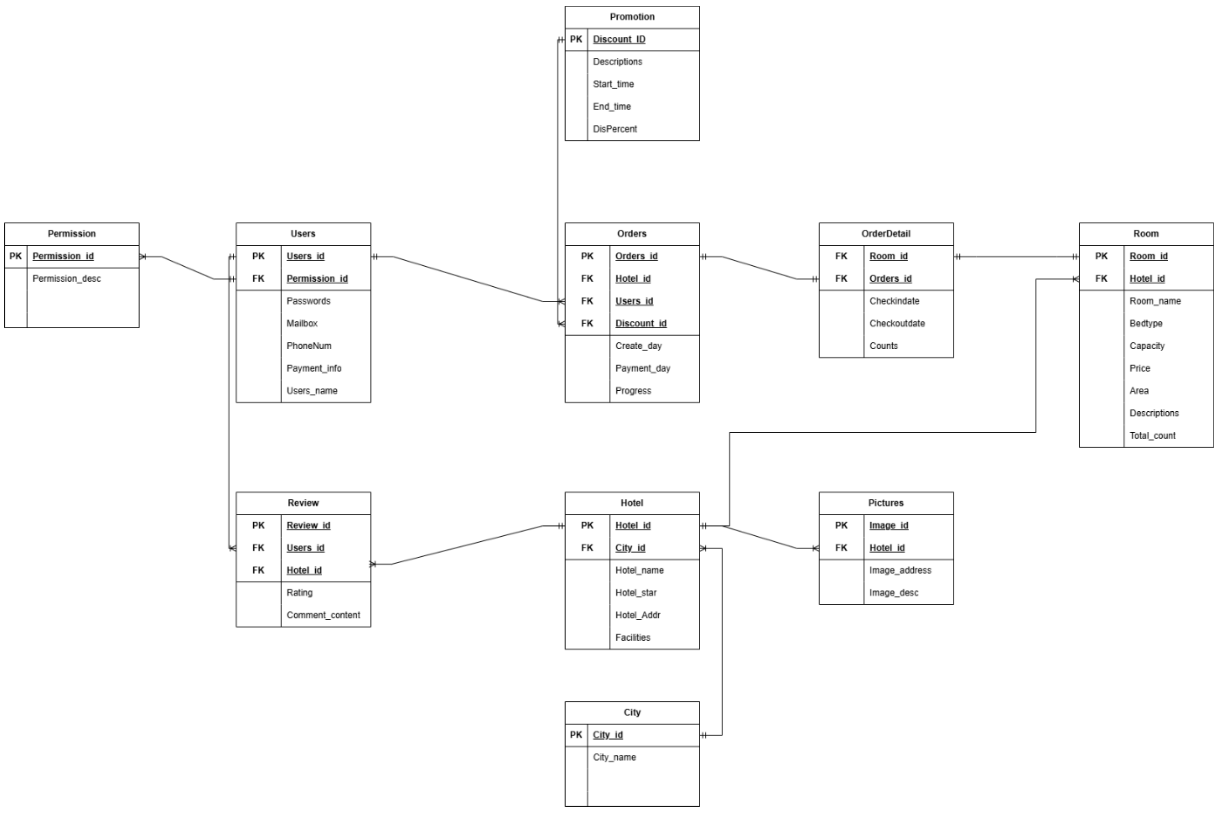


Figure 2 Hotel Database Design Diagram

4.3 About FDs

Users table:
Users_id → Passwords, Mailbox, PhoneNum, Payment_info, Users_name
Permission table:
Permission_id → Permission_desc
Promotion table:
Discount_ID → Descriptions, Start_time, End_time, DisPercent
Orders table:
Orders_id → Hotel_id, Users_id, Discount_id, Create_day, Payment_day, Progress

OrderDetail table:
(Orders_id, Room_id) → Checkindate, Checkoutdate, Counts
Room table:
Room_id → Hotel_id, Room_name, Bedtype, Capacity, Price, Area, Descriptions, Total_count
Hotel table:
Hotel_id → City_id, Hotel_name, Hotel_star, Hotel_Addr, Facilities
City table:
City_id → City_name
Review table:
Review_id → Users_id, Hotel_id, Rating, Comment_content
Pictures table:
Image_id → Hotel_id, Image_address, Image_desc

4.4 Reduces Data Redundancy

The design of each table follows the 3NF principle and there is no dependency of a non-primary attribute on a part of the primary key or a dependency of a non-primary attribute on other non-primary attributes. Such a design helps to reduce data redundancy and improve data integrity and consistency.

5 Database Implementation

5.1 About DBMS

Obviously, using the DBMS to manage data is an appropriate choice, especially if we need to support large data volumes and high data complexity in the future
The following points are the reasons for us to choose DBMS as our database design.

(1) Data Integrity and Consistency

DBMS provides strong data integrity constraints (e.g., primary keys, foreign keys, uniqueness constraints, etc.) to ensure data accuracy and consistency. With these constraints, data redundancy and conflicts can be avoided, ensuring that every record in the database is valid at the time of insertion, update and deletion.
For example, in a hotel reservation system without a DBMS, different users or the database administrator may accidentally insert duplicate data or inconsistent data, and the DBMS system can prevent these problems through constraints and transaction mechanisms.

(2) Transaction management and data recovery

DBMS supports ACID (Atomicity, Consistency, Isolation, and Durability) transaction attributes to ensure the consistency and reliability of data when accessed concurrently by multiple users. If an error or crash occurs while processing data, the DBMS is able to restore the data to its previous state through a rollback operation, ensuring that the data is not corrupted by incomplete or incorrect operations.
For example, when a hotel reservation system failure occurs, we are worried that data will be lost due to a system failure, so this is a point that the DBMS is able to automatically restore to the last

consistent state, ensuring data persistence and availability. Thus its function will let us not worry too much about this problem.

(3) Efficient data querying and management

DBMS provides a powerful query language (SQL) to efficiently handle complex data queries, aggregations and analyses. With indexes, views, stored procedures, and other features, the DBMS can quickly retrieve the information you need from large amounts of data and improve query performance.

For a hotel reservation application, bearing a large amount of data is inevitable in the future, so one of the reasons to use a DBMS is because its SQL can quickly query and extract thousands of data, especially when the system has multiple data tables

5.2 Data Definition Language (DDL) Implementation

According to the design of ER diagram and database diagram above, the next step is to use the DDL to create database tables. The following are the specific design details for each database table(e.g. Primary, Foreign Key, Attributes.)

(1) Table City

City	
Primary Key	City_id
	City_name

```
CREATE TABLE City (

City_id INT NOT NULL,

City_name VARCHAR(50) NOT NULL,

PRIMARY KEY (City_id),

);
```

(2) Table Hotel

Hotel	
Primary Key	Hotel_id
Foreign Key	City_id
	Hotel_name
	Hotel_star
	Hotel_Addr
	Facilities

```
CREATE TABLE Hotel (

 Hotel_id int NOT NULL ,

 Hotel_name VARCHAR(100) NOT NULL,

 Hotel_star int NOT NULL,

 Hotel_Addr varchar(100) NOT NULL,

 Facilities varchar(50),

 City_id int not null,

 PRIMARY KEY (hotel_id),

 FOREIGN KEY (City_id) REFERENCES City(City_id),

 );
```

(3) Table Pictures

Pictures	
Primary Key	Image_id
Foreign Key	Hotel_id
	Image_address
	Image_desc

```
CREATE TABLE Pictures(

  Image_id int not null,

  Hotel_id int not null,

  Image_address varchar(50) not null,

  Image_desc varchar(255),

  PRIMARY KEY (Image_id),
  FOREIGN KEY (Hotel_id) REFERENCES Hotel(Hotel_id),

  );
```

(4) Table Room

Room	
Primary Key	Room_id
Foreign Key	Hotel_id
	Room_name
	Bedtype
	Capacity
	Price
	Area
	Descriptions
	Total_count

```
CREATE TABLE Room (

 Room_id int NOT NULL,

 Hotel_id int not null,

 Room_name VARCHAR(50) NOT NULL,

 Bedtype VARCHAR(50) not NULL,

 Capacity int NOT NULL,

 Price int not null,

 Area decimal(10,2)  NOT NULL,

 Descriptions varchar(255),

 Total_count int not null,

 PRIMARY KEY  (Room_id),

 FOREIGN KEY (Hotel_id) REFERENCES Hotel(Hotel_id),
```

);

(5) Table Permission

Permission	
Primary Key	Permission_id
	Permission_desc

```
Create Table Permission (

   Permission_id int not null,

   Permission_desc varchar(100) not null,

   Primary key (Permission_id),);
```

(6) Table Users

Users	
Primary Key	Users_id
Foreign Key	Permission_id
	Users_name
	Passwords
	Mailbox
	PhoneNum
	Payment_info
	Addr

```
CREATE TABLE Users (

   Users_id int not null,

   Users_name varchar(50) not null,

   Passwords varchar(50) not null,

   Mailbox varchar(50) not null,

   PhoneNum bigint not null,

   Payment_info bigint not null,
   Permission_id int not null,

   Addr varchar(255) not null,

   primary key (Users_id),

   FOREIGN KEY (Permission_id) REFERENCES Permission(Permission_id),

   );
```

(7) Table Promotion

Promotion	
Primary Key	Discount_ID
	Descriptions
	Start_time
	End_time
	DisPercent

```
CREATE TABLE Promotion (

 Discount_ID int not null,

 Descriptions varchar(255) not null,

 Start_time datetime not null,

 End_time datetime not null,

 DisPercent decimal(10,2) not null,

 primary key (Discount_ID),

);
```

(8) Table Orders

Orders	
Primary Key	Orders_id
Foreign Key	Hotel_id
	Users_id
	Discount_id

	Create_day
	Payment_day
	Progress

```
CREATE TABLE Orders (

 Orders_id INT NOT NULL ,

 Create_day datetime NOT NULL,

 Payment_day datetime NOT NULL,

 Hotel_id int not null,

 Users_id int not null,

 Progress varchar(50) not null,

 Discount_id int,

 PRIMARY KEY (Orders_id),

 FOREIGN KEY (Hotel_id) REFERENCES Hotel(Hotel_id),

 FOREIGN KEY (Users_id) REFERENCES Users(Users_id),

 FOREIGN KEY (Discount_id) REFERENCES Promotion(Discount_id),);
```

(9) Table OrderDetail

OrderDetail	
Foreign Key	Room_id
	Orders_id
	Checkindate
	Checkoutdate
	Counts

```
CREATE TABLE OrderDetail(

  Checkindate datetime not null,

  Checkoutdate datetime not null,
  Orders_id int not null,

  Room_id int not null,

  Counts int not null,

  FOREIGN KEY (Room_id) REFERENCES Room(Room_id),

  FOREIGN KEY (Orders_id) REFERENCES Orders(Orders_id),);
```

(10) Table Review

Review	
Primary Key	Review_id
Foreign Key	Users_id
	Hotel_id
	Rating
	Comment_content

```
CREATE TABLE Review(

    Review_id int not null,

    Rating int not null,

    Comment_content varchar(255) not null,

    Users_id int not null,

    Hotel_id int not null,

    primary key (review_id),

    FOREIGN KEY (Users_id) REFERENCES Users(Users_id),

    FOREIGN KEY (Hotel_id) REFERENCES Hotel(Hotel_id),

    );
```

5.3 Data Manipulation Language (DML) Implementation

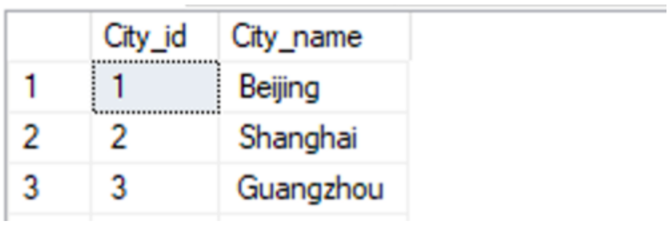
After completing the construction of the database tables, the next step is to use the SQL insert statement [INSERT INTO table_name (column1, column2, column3, ...) VALUES (value1, value2, value3, ...);] to insert data into each table separately to construct the complete database.

The following are some of the SQL insert statements and some of the actual data for each data table.

(1) City Data

```
INSERT INTO City (City_id,City_name) VALUES(1,'Beijing');

INSERT INTO City (City_id,City_name) VALUES(2,'Shanghai');

INSERT INTO City (City_id,City_name) VALUES(3,'Guangzhou');
```

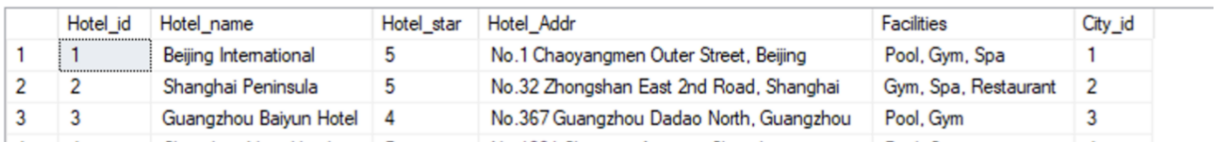
5.4



	City_id	City_name
1	1	Beijing
2	2	Shanghai
3	3	Guangzhou

(2) Hotel Data

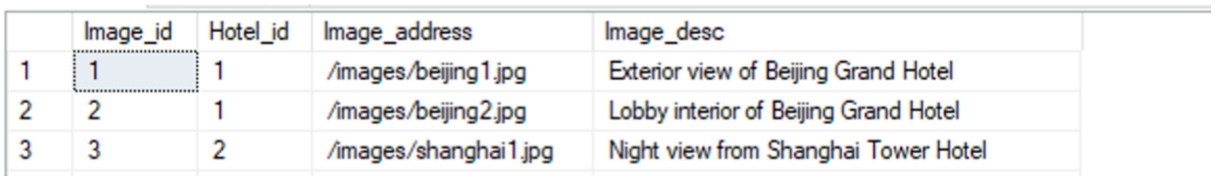
```
INSERT INTO Hotel ( Hotel_id , Hotel_name , Hotel_star , Hotel_Addr , Facilities , City_id) VALUES
( 1 , 'Beijing International' , 5 , 'No.1 Chaoyangmen Outer Street, Beijing' , 'Pool, Gym,
 Spa' , 1 );

INSERT INTO Hotel ( Hotel_id , Hotel_name , Hotel_star , Hotel_Addr , Facilities , City_id) VALUES
( 2 , 'Shanghai Peninsula' , 5 , 'No.32 Zhongshan East 2nd Road, Shanghai' , 'Gym, Spa, Re
staurant' , 2 );

INSERT INTO Hotel ( Hotel_id , Hotel_name , Hotel_star , Hotel_Addr , Facilities , City_id) VALUES
( 3 , 'Guangzhou Baiyun Hotel' , 4 , 'No.367 Guangzhou Dadao North, Guangzhou' , 'Pool, Gy
m' , 3 );
```



	Hotel_id	Hotel_name	Hotel_star	Hotel_Addr	Facilities	City_id
1	1	Beijing International	5	No.1 Chaoyangmen Outer Street, Beijing	Pool, Gym, Spa	1
2	2	Shanghai Peninsula	5	No.32 Zhongshan East 2nd Road, Shanghai	Gym, Spa, Restaurant	2
3	3	Guangzhou Baiyun Hotel	4	No.367 Guangzhou Dadao North, Guangzhou	Pool, Gym	3

(3) Pictures Data

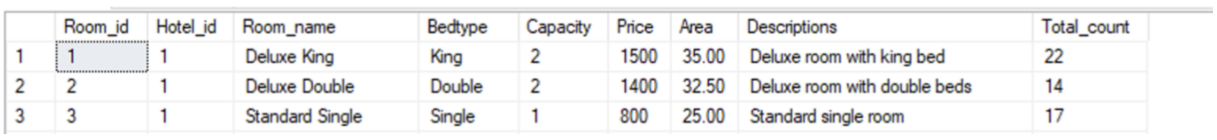
```
INSERT INTO Pictures (Image_id, Hotel_id, Image_address, Image_desc) VALUES (1,1,'/images/beijing1.jpg','Exterior view of Beijing Grand Hotel');

INSERT INTO Pictures (Image_id, Hotel_id, Image_address, Image_desc) VALUES (2,1,'/images/beijing2.jpg','Lobby interior of Beijing Grand Hotel');

INSERT INTO Pictures (Image_id, Hotel_id, Image_address, Image_desc) VALUES (3,2,'/images/shanghai1.jpg','Night view from Shanghai Tower Hotel');
```



	Image_id	Hotel_id	Image_address	Image_desc
1	1	1	/images/beijing1.jpg	Exterior view of Beijing Grand Hotel
2	2	1	/images/beijing2.jpg	Lobby interior of Beijing Grand Hotel
3	3	2	/images/shanghai1.jpg	Night view from Shanghai Tower Hotel

(4) Room Data

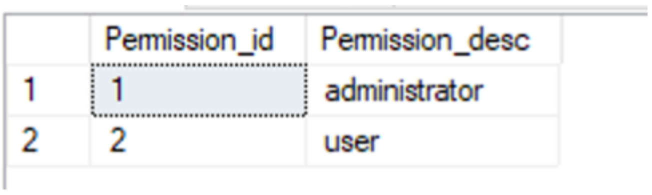
```
INSERT INTO Room (Room_id,Hotel_id,Room_name,Bedtype,Capacity,Price,Area,Descriptions,Total_count) VALUES (1,1,'Deluxe King','King',2,1500,35,'Deluxe room with king bed',22);

INSERT INTO Room (Room_id,Hotel_id,Room_name,Bedtype,Capacity,Price,Area,Descriptions,Total_count) VALUES (2,1,'Deluxe Double','Double',2,1400,32.5,'Deluxe room with double beds',14);

INSERT INTO Room (Room_id,Hotel_id,Room_name,Bedtype,Capacity,Price,Area,Descriptions,Total_count) VALUES (3,1,'Standard Single','Single',1,800,25,'Standard single room',17);
```



	Room_id	Hotel_id	Room_name	Bedtype	Capacity	Price	Area	Descriptions	Total_count
1	1	1	Deluxe King	King	2	1500	35.00	Deluxe room with king bed	22
2	2	1	Deluxe Double	Double	2	1400	32.50	Deluxe room with double beds	14
3	3	1	Standard Single	Single	1	800	25.00	Standard single room	17

(5) Permission Data

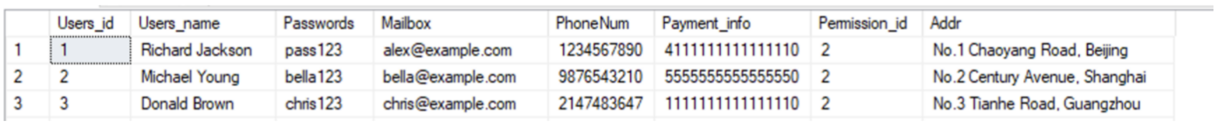
```
INSERT INTO Permission (Permission_id, Permission_desc) VALUES (1,'administrator');

INSERT INTO Permission (Permission_id, Permission_desc) VALUES (2,'user');
```



	Permission_id	Permission_desc
1	1	administrator
2	2	user

(6) Users Data

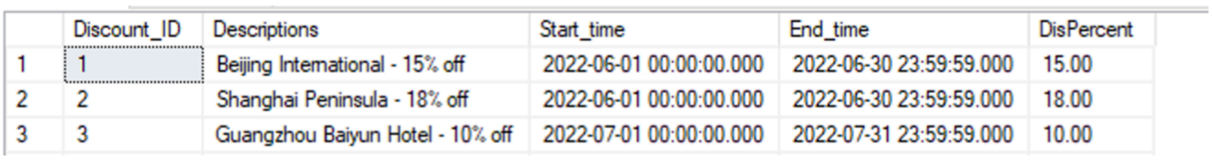
```sql
INSERT INTO Users (Users_id,Users_name,Passwords,Mailbox,PhoneNum,Payment_info,Permission_id,Addr) VALUES (1,'Richard Jackson','pass123','alex@example.com',1234567890,4111111111111110,2,'No.1 Chaoyang Road, Beijing');

INSERT INTO Users (Users_id,Users_name,Passwords,Mailbox,PhoneNum,Payment_info,Permission_id,Addr) VALUES (2,'Michael Young','bella123','bella@example.com',9876543210,5555555555555550,2,'No.2 Century Avenue, Shanghai');

INSERT INTO Users (Users_id,Users_name,Passwords,Mailbox,PhoneNum,Payment_info,Permission_id,Addr) VALUES (3,'Donald Brown','chris123','chris@example.com',2147483647,1111111111111110,2,'No.3 Tianhe Road, Guangzhou');
```



	Users_id	Users_name	Passwords	Mailbox	PhoneNum	Payment_info	Permission_id	Addr
1	1	Richard Jackson	pass123	alex@example.com	1234567890	4111111111111110	2	No.1 Chaoyang Road, Beijing
2	2	Michael Young	bella123	bella@example.com	9876543210	5555555555555550	2	No.2 Century Avenue, Shanghai
3	3	Donald Brown	chris123	chris@example.com	2147483647	1111111111111110	2	No.3 Tianhe Road, Guangzhou

(7) Promotion Data

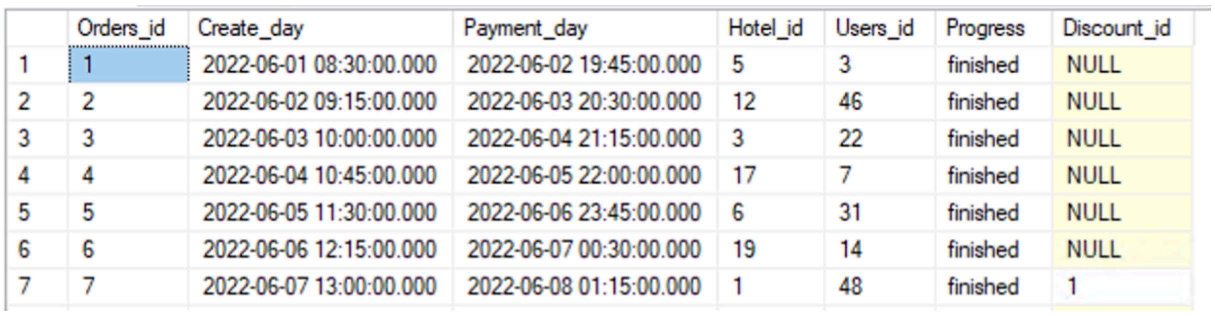
```sql
INSERT INTO Promotion (Discount_ID,Descriptions,Start_time,End_time,Dispercent) VALUES (1,'Beijing International - 15% off','2022-06-01 00:00:00','2022-06-30 23:59:59',15);

INSERT INTO Promotion (Discount_ID,Descriptions,Start_time,End_time,Dispercent) VALUES (2,'Shanghai Peninsula - 18% off','2022-06-01 00:00:00','2022-06-30 23:59:59',18);

INSERT INTO Promotion (Discount_ID,Descriptions,Start_time,End_time,Dispercent) VALUES (3,'Guangzhou Baiyun Hotel - 10% off','2022-07-01 00:00:00','2022-07-31 23:59:59',10);
```



	Discount_ID	Descriptions	Start_time	End_time	DisPercent
1	1	Beijing International - 15% off	2022-06-01 00:00:00.000	2022-06-30 23:59:59.000	15.00
2	2	Shanghai Peninsula - 18% off	2022-06-01 00:00:00.000	2022-06-30 23:59:59.000	18.00
3	3	Guangzhou Baiyun Hotel - 10% off	2022-07-01 00:00:00.000	2022-07-31 23:59:59.000	10.00

(8) Orders Data

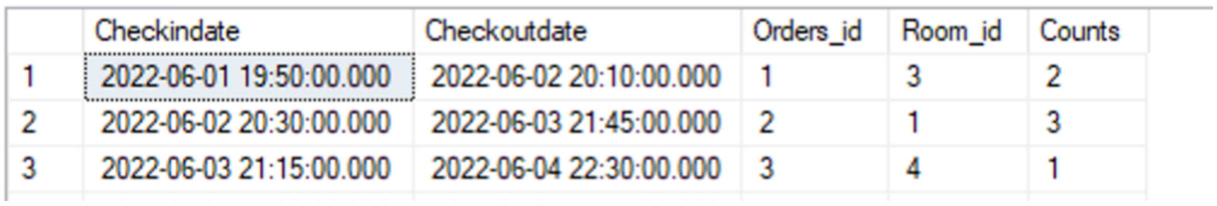
```
INSERT INTO Orders (Orders_id,Create_day,Payment_day,Hotel_id,Users_id,Progress,Discount_id)
 VALUES(1,'2022-06-01 08:30:00','2022-06-02 19:45:00',5,3,'finished',null);

INSERT INTO Orders (Orders_id,Create_day,Payment_day,Hotel_id,Users_id,Progress,Discount_id)
 VALUES(2,'2022-06-02 09:15:00','2022-06-03 20:30:00',12,46,'finished',null);

INSERT INTO Orders (Orders_id,Create_day,Payment_day,Hotel_id,Users_id,Progress,Discount_id)
 VALUES(3,'2022-06-03 10:00:00','2022-06-04 21:15:00',3,22,'finished',null);

INSERT INTO Orders (Orders_id,Create_day,Payment_day,Hotel_id,Users_id,Progress,Discount_id)
 VALUES(4,'2022-06-04 10:45:00','2022-06-05 22:00:00',17,7,'finished',null);

INSERT INTO Orders (Orders_id,Create_day,Payment_day,Hotel_id,Users_id,Progress,Discount_id)
 VALUES(5,'2022-06-05 11:30:00','2022-06-06 23:45:00',6,31,'finished',null);

INSERT INTO Orders (Orders_id,Create_day,Payment_day,Hotel_id,Users_id,Progress,Discount_id)
 VALUES(6,'2022-06-06 12:15:00','2022-06-07 00:30:00',19,14,'finished',null);

INSERT INTO Orders (Orders_id,Create_day,Payment_day,Hotel_id,Users_id,Progress,Discount_id)
 VALUES(7,'2022-06-07 13:00:00','2022-06-08 01:15:00',1,48,'finished',1);
```



	Orders_id	Create_day	Payment_day	Hotel_id	Users_id	Progress	Discount_id
1	1	2022-06-01 08:30:00.000	2022-06-02 19:45:00.000	5	3	finished	NULL
2	2	2022-06-02 09:15:00.000	2022-06-03 20:30:00.000	12	46	finished	NULL
3	3	2022-06-03 10:00:00.000	2022-06-04 21:15:00.000	3	22	finished	NULL
4	4	2022-06-04 10:45:00.000	2022-06-05 22:00:00.000	17	7	finished	NULL
5	5	2022-06-05 11:30:00.000	2022-06-06 23:45:00.000	6	31	finished	NULL
6	6	2022-06-06 12:15:00.000	2022-06-07 00:30:00.000	19	14	finished	NULL
7	7	2022-06-07 13:00:00.000	2022-06-08 01:15:00.000	1	48	finished	1

(9) OrderDetail Data

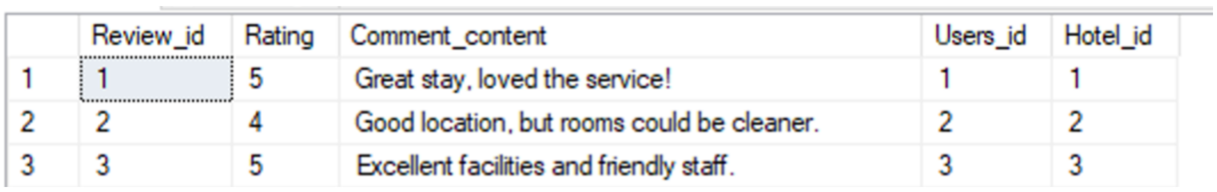
```
INSERT INTO OrderDetail (Checkindate,Checkoutdate,Orders_id,Room_id,Counts) VALUES('2022-
06-01 19:50:00','2022-06-02 20:10:00',1,3,2);

INSERT INTO OrderDetail (Checkindate,Checkoutdate,Orders_id,Room_id,Counts) VALUES('2022-
06-02 20:30:00','2022-06-03 21:45:00',2,1,3);

INSERT INTO OrderDetail (Checkindate,Checkoutdate,Orders_id,Room_id,Counts) VALUES('2022-
06-03 21:15:00','2022-06-04 22:30:00',3,4,1);
```



	Checkindate	Checkoutdate	Orders_id	Room_id	Counts
1	2022-06-01 19:50:00.000	2022-06-02 20:10:00.000	1	3	2
2	2022-06-02 20:30:00.000	2022-06-03 21:45:00.000	2	1	3
3	2022-06-03 21:15:00.000	2022-06-04 22:30:00.000	3	4	1

(10)Review Data

```
INSERT INTO Review( Review_id,Rating,Comment_content,Users_id,Hotel_id) VALUES(1,5,'Great
 stay, loved the service!',1,1);

INSERT INTO Review( Review_id,Rating,Comment_content,Users_id,Hotel_id) VALUES(2,4,'Good
location, but rooms could be cleaner.',2,2);

INSERT INTO Review( Review_id,Rating,Comment_content,Users_id,Hotel_id) VALUES(3,5,'Excel
lent facilities and friendly staff.',3,3);
```



	Review_id	Rating	Comment_content	Users_id	Hotel_id
1	1	5	Great stay, loved the service!	1	1
2	2	4	Good location, but rooms could be cleaner.	2	2
3	3	5	Excellent facilities and friendly staff.	3	3

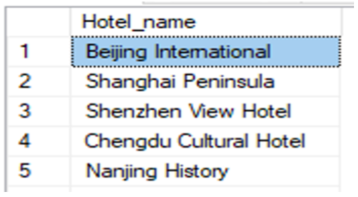
5.5 Data Query Implementation

In order to be able to verify the feasibility of our database design, we take the two roles of hotel administrator and user as the perspective, and try to query the desired data information from the perspective of these two roles.

Query 1 Assume you are a customer and want to enquire about all 5-star rating hotels.

```
select Hotel_name

from Hotel

where Hotel_star = 5
```

Query Result (Partial results)

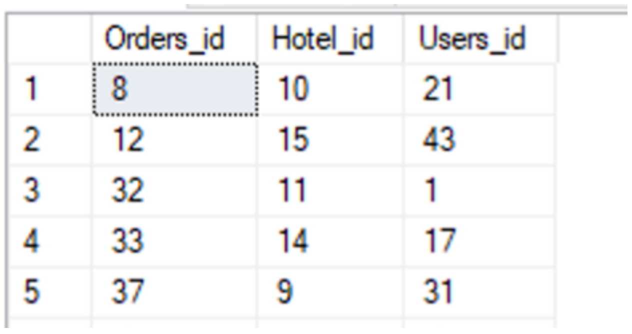


	Hotel_name
1	Beijing International
2	Shanghai Peninsula
3	Shenzhen View Hotel
4	Chengdu Cultural Hotel
5	Nanjing History

Query 2 Assume you are a hotel administrator and want to enquire about those cancelled orders.

```
select Orders_id,

Hotel_id,

Users_id

from orders

where Progress = 'canceled'
```

Query Result (Partial results)

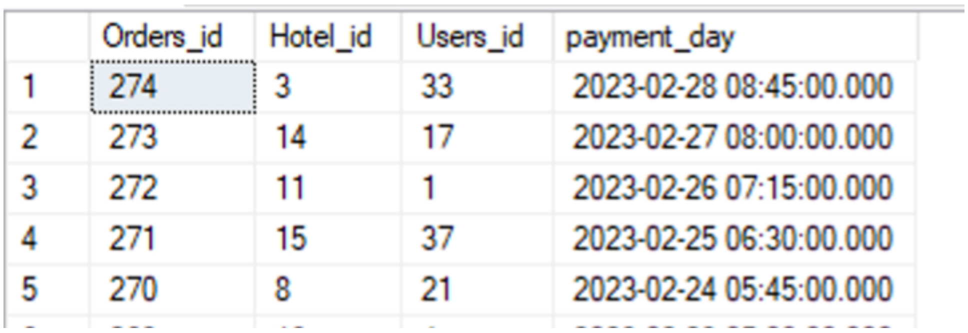


	Orders_id	Hotel_id	Users_id
1	8	10	21
2	12	15	43
3	32	11	1
4	33	14	17
5	37	9	31

Query 3 Assume you are a hotel administrator and want to enquire about the orders in February 2023 and sort them in descending order of payment time.

```
select Orders_id,

Hotel_id,

Users_id,

payment_day

from orders

where payment_day >= '2023-02-01 00:00:00'

AND payment_day <= '2023-03-01 00:00:00'

order by payment_day desc
```

Query Result (Partial results)

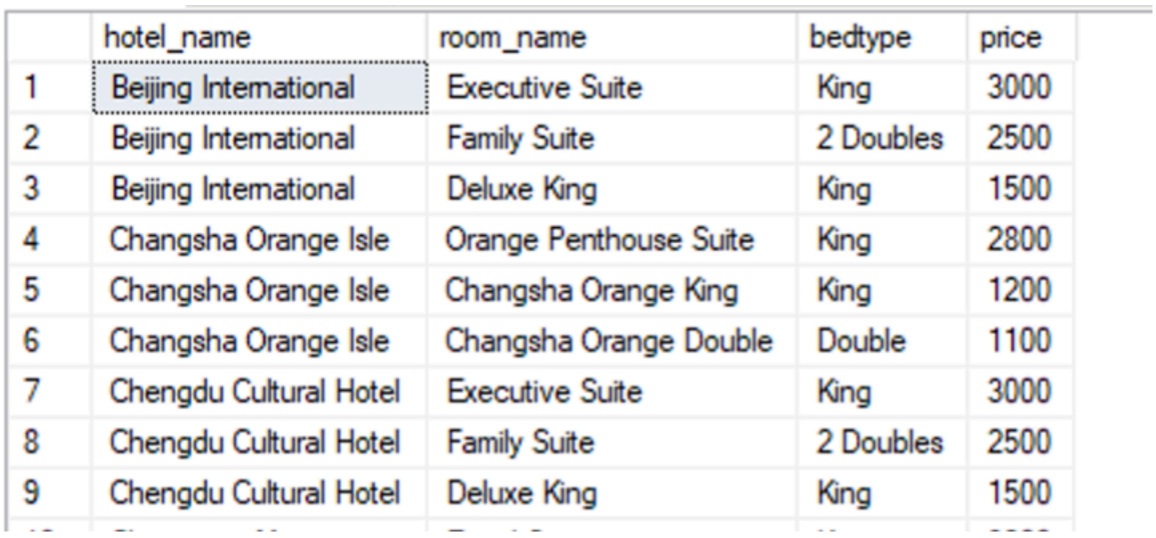


	Orders_id	Hotel_id	Users_id	payment_day
1	274	3	33	2023-02-28 08:45:00.000
2	273	14	17	2023-02-27 08:00:00.000
3	272	11	1	2023-02-26 07:15:00.000
4	271	15	37	2023-02-25 06:30:00.000
5	270	8	21	2023-02-24 05:45:00.000

Query 4 Assume you are a customer and want to enquire information about the top three most expensive rooms in each hotel.

```
SELECT hotel_name,

room_name,

bedtype,

price

From(

select h.hotel_name,

r.room_name,

r.bedtype,

r.price,

RANK()OVER(partition by h.hotel_name order by r.price desc) posn

from Hotel h join Room r

on h.hotel_id = r.hotel_id

) as rk

where rk.posn <= 3
```

Query Result (Partial results)

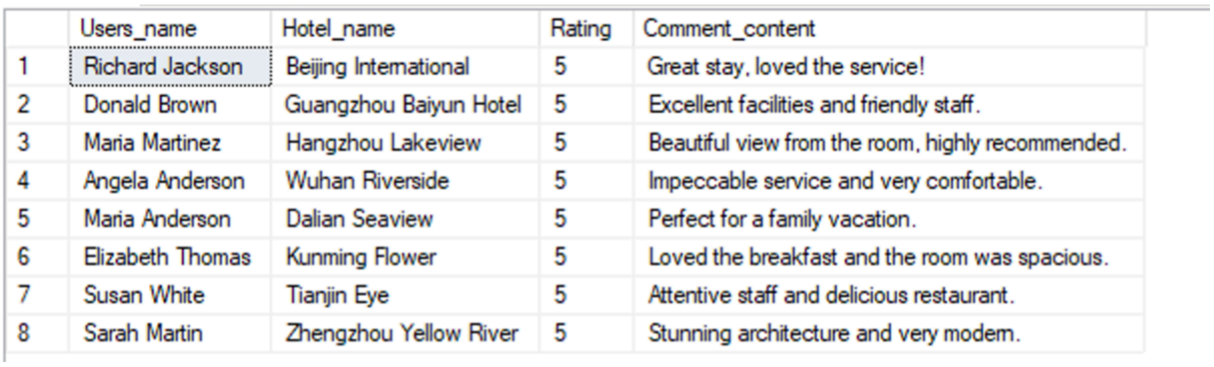


	hotel_name	room_name	bedtype	price
1	Beijing International	Executive Suite	King	3000
2	Beijing International	Family Suite	2 Doubles	2500
3	Beijing International	Deluxe King	King	1500
4	Changsha Orange Isle	Orange Penthouse Suite	King	2800
5	Changsha Orange Isle	Changsha Orange King	King	1200
6	Changsha Orange Isle	Changsha Orange Double	Double	1100
7	Chengdu Cultural Hotel	Executive Suite	King	3000
8	Chengdu Cultural Hotel	Family Suite	2 Doubles	2500
9	Chengdu Cultural Hotel	Deluxe King	King	1500

Query 5 Assume you are a hotel administrator and want to enquire about the users who have given the hotel a 5-star rating and their content of comments.

```
SELECT u.Users_name,

h.Hotel_name,

r.Rating,

r.Comment_content

FROM Review r

join Hotel h on h.hotel_id = r.hotel_id

join Users u on u.Users_id = r.Users_id

where r.Rating = 5
```

Query Result (Partial results)



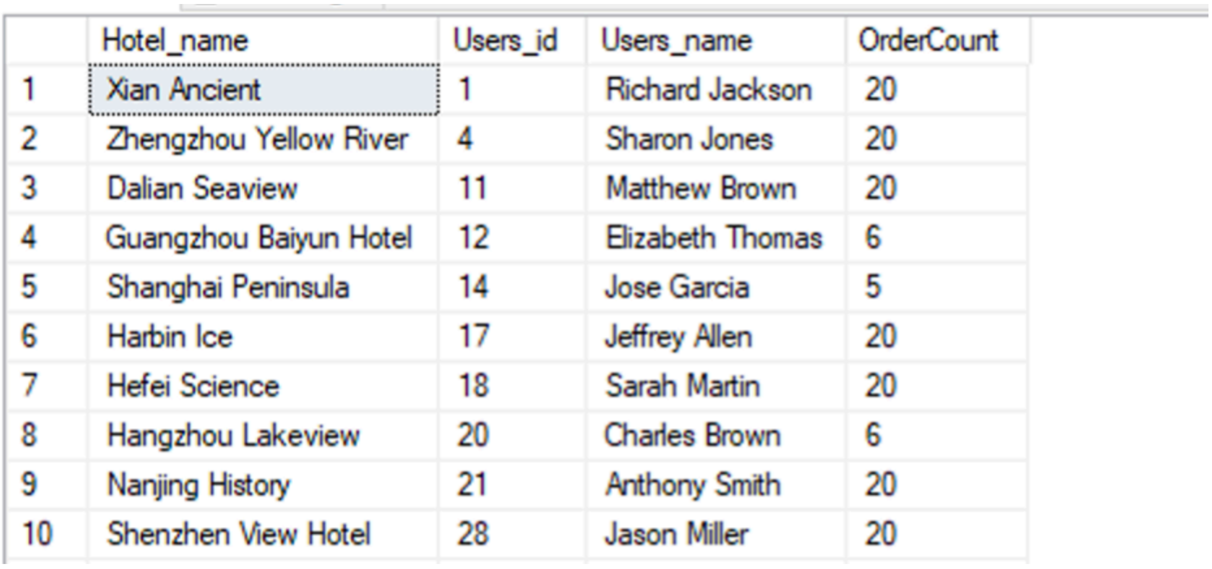
	Users_name	Hotel_name	Rating	Comment_content
1	Richard Jackson	Beijing International	5	Great stay, loved the service!
2	Donald Brown	Guangzhou Baiyun Hotel	5	Excellent facilities and friendly staff.
3	Maria Martinez	Hangzhou Lakeview	5	Beautiful view from the room, highly recommended.
4	Angela Anderson	Wuhan Riverside	5	Impeccable service and very comfortable.
5	Maria Anderson	Dalian Seaview	5	Perfect for a family vacation.
6	Elizabeth Thomas	Kunming Flower	5	Loved the breakfast and the room was spacious.
7	Susan White	Tianjin Eye	5	Attentive staff and delicious restaurant.
8	Sarah Martin	Zhengzhou Yellow River	5	Stunning architecture and very modern.

Query 6 Assume you are a hotel administrator and want to enquire about the most loyal customers in your hotel.

```
SELECT

  h.Hotel_name,   u.Users_id,   u.Users_name,   uo.OrderCount
FROM

  Hotel h

JOIN

  (SELECT

     o.Hotel_id,   o.Users_id,
     COUNT(*) AS OrderCount

   FROM

     Orders o

   GROUP BY

     o.Hotel_id, o.Users_id

  HAVING COUNT(*) > 1

  ) uo ON h.Hotel_id = uo.Hotel_id

JOIN

  Users u ON u.Users_id = uo.Users_id

LEFT JOIN

  (SELECT

     Hotel_id,

     MAX(OrderCount) AS MaxOrders

   FROM
```

```
        (SELECT
            Hotel_id,    Users_id,
            COUNT(*) AS OrderCount
         FROM
            Orders
         GROUP BY
            Hotel_id, Users_id
         HAVING COUNT(*) > 1
        ) AS UserOrderCounts
    GROUP BY
        Hotel_id
) mo ON uo.Hotel_id = mo.Hotel_id AND uo.OrderCount = mo.MaxOrders;
```

Query Result (Partial results)



	Hotel_name	Users_id	Users_name	OrderCount
1	Xian Ancient	1	Richard Jackson	20
2	Zhengzhou Yellow River	4	Sharon Jones	20
3	Dalian Seaview	11	Matthew Brown	20
4	Guangzhou Baiyun Hotel	12	Elizabeth Thomas	6
5	Shanghai Peninsula	14	Jose Garcia	5
6	Harbin Ice	17	Jeffrey Allen	20
7	Hefei Science	18	Sarah Martin	20
8	Hangzhou Lakeview	20	Charles Brown	6
9	Nanjing History	21	Anthony Smith	20
10	Shenzhen View Hotel	28	Jason Miller	20

So, based on all of the above about the database design, starting from the conceptual model ER diagram design to the DBMS data table structure design, and then to the insertion of data and query results verification, we basically completed the database design about the hotel reservation.

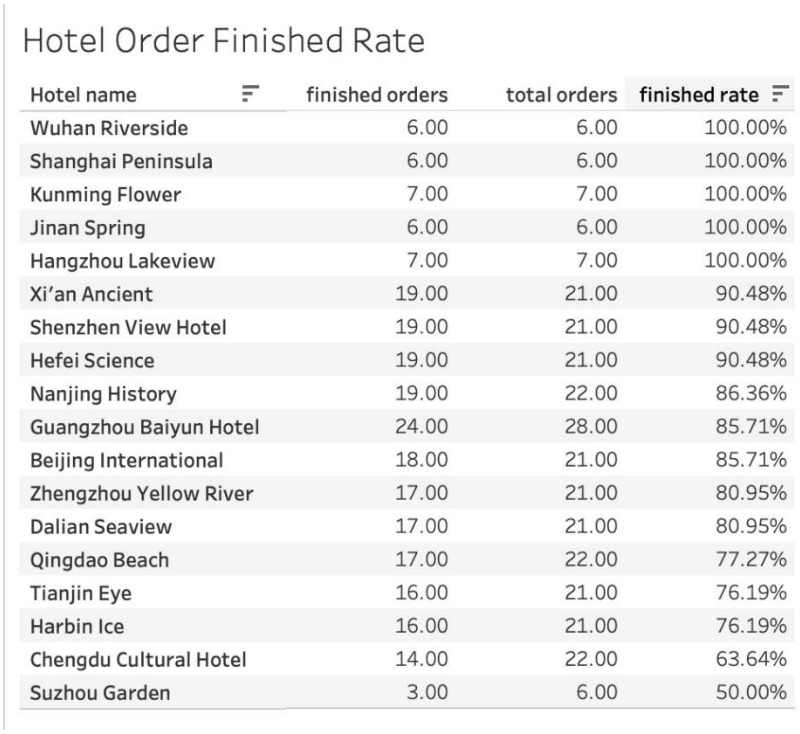
6 Database Visualization

6.1 Tools for data visualization: Tableau

6.2 Data visualization and analysis

（1） Hotel Order Finished Rate

This table shows the order completion status of each hotel, specifically the ratio of completed orders to total orders. It helps identify which hotels are performing well in terms of order completion and which may need to improve their services or processes. For hotel management, this data can be used to assess customer satisfaction and service quality. A high completion rate may indicate that customers are satisfied with the hotel's services, while a low completion rate might suggest issues that require further investigation and resolution.

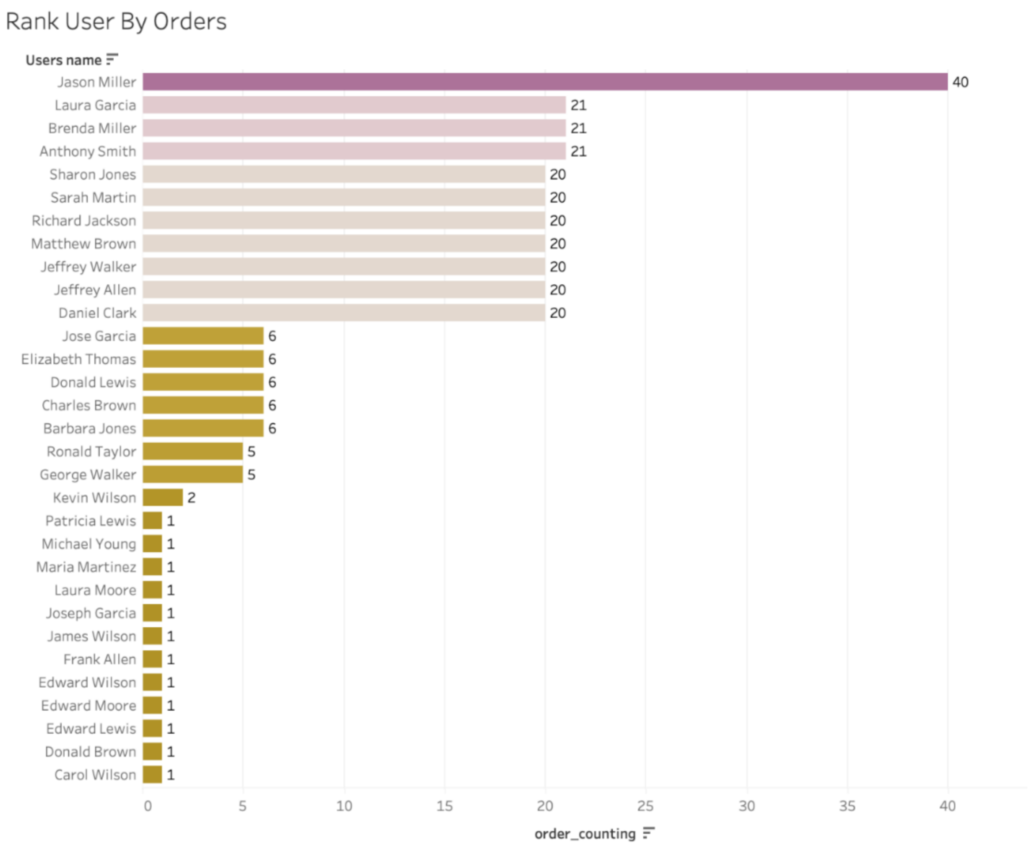


Hotel Order Finished Rate

Hotel name	finished orders	total orders	finished rate
Wuhan Riverside	6.00	6.00	100.00%
Shanghai Peninsula	6.00	6.00	100.00%
Kunming Flower	7.00	7.00	100.00%
Jinan Spring	6.00	6.00	100.00%
Hangzhou Lakeview	7.00	7.00	100.00%
Xi’an Ancient	19.00	21.00	90.48%
Shenzhen View Hotel	19.00	21.00	90.48%
Hefei Science	19.00	21.00	90.48%
Nanjing History	19.00	22.00	86.36%
Guangzhou Baiyun Hotel	24.00	28.00	85.71%
Beijing International	18.00	21.00	85.71%
Zhengzhou Yellow River	17.00	21.00	80.95%
Dalian Seaview	17.00	21.00	80.95%
Qingdao Beach	17.00	22.00	77.27%
Tianjin Eye	16.00	21.00	76.19%
Harbin Ice	16.00	21.00	76.19%
Chengdu Cultural Hotel	14.00	22.00	63.64%
Suzhou Garden	3.00	6.00	50.00%

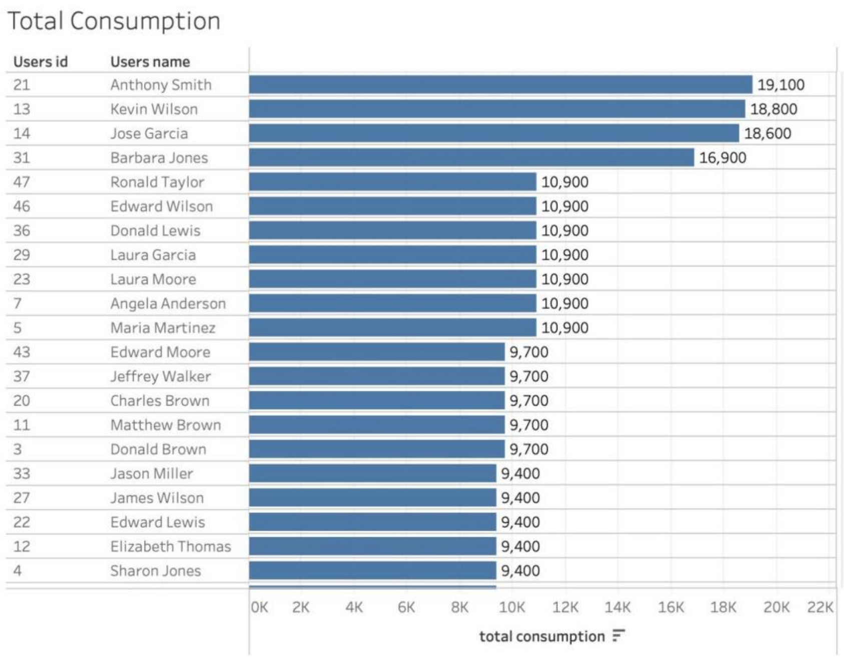
（2） Rank Users By Orders

This table displays the number of orders from different users, making it easy to compare orders among them. For marketing and customer relationship management, understanding which users place the most orders can help identify key or loyal customers. This information can be used to tailor marketing strategies, such as providing personalized services.



(3) Total Consumption

This table lists the total spending of users. Total consumption data is important for understanding the purchasing power and value of customers. It can help identify high-value customers to whom they can provide better services or exclusive offers. Additionally, this data can be used for financial analysis to help businesses understand their revenue sources and the spending patterns of their customer base.



Users id	Users name	total consumption
21	Anthony Smith	19,100
13	Kevin Wilson	18,800
14	Jose Garcia	18,600
31	Barbara Jones	16,900
47	Ronald Taylor	10,900
46	Edward Wilson	10,900
36	Donald Lewis	10,900
29	Laura Garcia	10,900
23	Laura Moore	10,900
7	Angela Anderson	10,900
5	Maria Martinez	10,900
43	Edward Moore	9,700
37	Jeffrey Walker	9,700
20	Charles Brown	9,700
11	Matthew Brown	9,700
3	Donald Brown	9,700
33	Jason Miller	9,400
27	James Wilson	9,400
22	Edward Lewis	9,400
12	Elizabeth Thomas	9,400
4	Sharon Jones	9,400

7 Conclusion

This hotel booking database design comprehensively explores all aspects from requirement analysis to implementation, aiming to meet the growing demand for online booking. Booking.com was chosen for the study due to its influence in the market and innovative user experience. The system design focuses on a user-friendly interface and efficient search and booking functions to ensure that users complete transactions safely and conveniently, and to enhance the sense of engagement through a post-occupancy evaluation mechanism. In the requirement analysis, key business processes and user roles were identified to optimise user experience. Through a complete process from conceptual model (ER diagram) design to database table structure design to data insertion and query validation, we ensure that the system can handle highly concurrent user access while maintaining data consistency and security and the relationships between entities are clearly depicted and data integrity is ensured through the setting of primary and foreign keys. DBMS was chosen as the data management platform, providing the advantages of data integrity, transaction management and efficient querying to effectively support highly concurrent access. SQL queries ensure system availability and data accuracy during data insertion and validation. The scalability and maintainability of the database design allows it to adapt to changes in business requirements, while the modular design and clear data relationships facilitate the addition of new functionality or system upgrades in the future. Data analysis using visualisation tools helps managers understand data in multiple dimensions and supports decision making. In summary, this design lays the foundation for an efficient and reliable online booking experience that will continue to be optimised for market changes in the future.

Future Outlook

Enhanced Data Analysis and Machine Learning: The database can be upgraded to integrate advanced analytics and machine learning to offer insights like personalized accommodation suggestions and market trend predictions, enhancing user satisfaction and operational efficiency.

Data Security and Privacy: Emphasizing data integrity, the database will adopt state-of-the-art encryption and comply with data protection laws to safeguard user information, building trust and ensuring privacy.

Advanced Permission Control and Auditing: The database will implement sophisticated permission controls, possibly including ABAC and context-aware access, and enhance auditing to track data access, providing compliance and security, with real-time monitoring to detect and address security issues effectively.

Reference

[1] Schmidtke, J., & Schmidt, K. (2006). Data management and data base implementation for GMO monitoring. *Journal Für Verbraucherschutz Und Lebensmittelsicherheit*, *1*(S1), 92–94. https://doi.org/10.1007/s00003-006-0096-0

[2] Ordonez, C., Varghese, R., Phan, N., & Macyna, W. (2024). Growing a FLOWER: Building a Diagram Unifying Flow and ER Notation for Data Science. *Proceedings of the 2024 Workshop on Human-In-the-Loop Data Analytics*, 1–8. https://doi.org/10.1145/3665939.3665958

[3] Tao, M., Nawaz, M. Z., Nawaz, S., Butt, A. H., & Ahmad, H. (2018). Users' acceptance of innovative mobile hotel booking trends: UK vs. PRC. *Information Technology & Tourism*, *20*(1–4), 9–36. https://doi.org/10.1007/s40558-018-0123-x

[4] Chen, P. P.-S. (1977). The entity-relationship model: a basis for the enterprise view of data. *Proceedings of the June 13-16, 1977, National Computer Conference*, 77–84. https://doi.org/10.1145/1499402.1499421

[5] Weldon, J.-L. (1975). Review of “An introduction to database systems” by C. J. Date. Addison-Wesley Publishing Co. 1975. SIGMOD Record, 7(3–4), 53–54. https://doi.org/10.1145/984403.984407

[6] Hayden, R. W. (2010). A Review of: “Now You See It: Simple Visualization Techniques for Quantitative Analysis, by S. C. Few,”: Oakland, CA: Analytics Press, 2009, ISBN 0-9706019-8-0, xi + 327 pp., $45 [Review of A Review of: “Now You See It: Simple Visualization Techniques for Quantitative Analysis, by S. C. Few,”: Oakland, CA: Analytics Press, 2009, ISBN 0-9706019-8-0, xi + 327 pp., $45]. Journal of Biopharmaceutical Statistics, 20(3), 701–702. Taylor & Francis Group. https://doi.org/10.1080/10543401003641225

[7] Sun, N., Yang, X., & Liu, Y. (2020). TableQA: a Large-Scale Chinese Text-to-SQL Dataset for Table-Aware SQL Generation. arXiv.Org.

[8] HAERDER, T., & REUTER, A. (1983). Principles of transaction-oriented database recovery. ACM Computing Surveys, 15(4), 287–317. https://doi.org/10.1145/289.291