

Implementation of a comprehensive Hotel Management System (HMS) utilizing a Database Management System (DBMS)

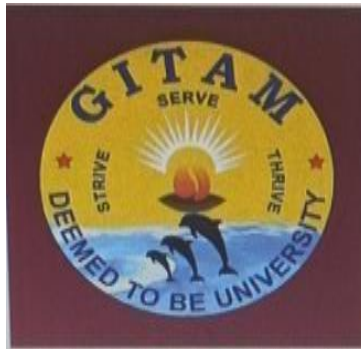
**A Case study Report submitted in partial fulfilment of the requirement
for awarding marks as part of the Course Continuous Evaluation**

Submitted by

TEAM-8-HOTEL MANAGEMENT SYSTEM

Under the supervision of

Course Faculty Name: Sanapala Venkata Lakshmi



DEPARTMENT OF COMPUTER SCIENCE &

ENGINEERING GITAM

(Deemed to be University)

VISAKHAPATNAM

October 2023

**DEPARTMENT OF COMPUTER SCIENCE AND
ENGINEERING**

GITAM SCHOOL OF TECHNOLOGY

GITAM

**(Deemed to be
University)**



I now declare that the case study report entitled **“Implementation of a comprehensive Hotel Management System (HMS) utilizing a Database Management System (DBMS)”** is an original work done in the Department of Computer Science and Engineering, GITAM School of Technology, GITAM (Deemed to be University), submitted in partial fulfilment of the requirements for awarding marks as part of the Course Continuous Evaluation. The work has not been copied or shared with any others.

Date:

17/10/2023

Satla Sree Varshith	VU21CSEN0100481
Bhuvan Lohit Dev Nakka	VU21CSEN0100463
Ratna Varshith Kolachala	VU21CSEN0100496

TABLE OF CONTENTS

<u>S.NO</u>	<u>Contents</u>	<u>Page numbers</u>
<u>1</u>	<u>Abstract of the Project</u>	<u>4</u>
<u>2</u>	<u>Introduction about the use case & Requirement Analysis</u>	<u>5-6</u>
<u>3</u>	<u>Module Description</u>	<u>7-10</u>
<u>4</u>	<u>Relationships</u>	<u>11-12</u>
<u>5</u>	<u>ER Diagram and Lucid chart diagram</u>	<u>13-14</u>
<u>6</u>	<u>Logical Database Definition and Creating Tables</u>	<u>15-22</u>
<u>7</u>	<u>Inserting Data/Records into Tables</u>	<u>23-41</u>
<u>8</u>	<u>Framing Conditional Queries</u>	<u>42-59</u>
<u>9</u>	<u>Conclusion</u>	<u>60</u>

ABSTRACT OF THE PROJECT

This case study focuses on the design and implementation of a comprehensive Hotel Management System (HMS) utilizing a Database Management System (DBMS). The primary goal is to develop a robust and efficient database structure that supports various functionalities essential for efficient hotel operations.

The system encompasses guest reservations, room management, billing, staff administration, and reporting. The database design involves entity-relationship modelling, schema design, normalization, and query optimization to enhance performance and data integrity.

This abstract provides a high-level overview of the case study, emphasizing the objectives, key functionalities, and the importance of the Hotel Management System in the context of database management.

Introduction about the Use Case

The primary goal is to develop a robust and efficient database structure that supports various functionalities essential for efficient hotel operations.

The system encompasses guest reservations, room management, billing, staff administration, and reporting.

The database design involves entity-relationship modeling, schema design, normalization, and query optimization to enhance performance and data integrity.

The system also includes features for managing inventory, housekeeping, and guest feedback.

REQUIREMENT ANALYSIS

Software Requirements:

(i) Operating System:

The Hotel Management System should be compatible with common operating systems, including Windows, Linux, and macOS.

(ii) Database Management System (DBMS):

The system requires a DBMS to manage the database. Commonly used DBMSs include:

Oracle Database

MySQL

Microsoft SQL Server

SQLite

(iii) Documentation Tools:

- Tools for creating technical documentation and user manuals, such as Microsoft Word.

Hardware Requirements:

(i) Server Hardware:

The system may require a server to host the database and, if it's a web-based system, the web application. Hardware requirements for the server include:

Processor: A multi-core processor for handling concurrent requests.

Memory (RAM): Sufficient RAM to support database operations and application logic.

Storage: Adequate storage capacity for the database and system files.

Network Interface: Network connectivity to serve requests.

(ii) Client Devices:

End-user devices (e.g., computers, mobile devices) to access the system. There are no strict hardware requirements for clients, as most modern devices can access web-based systems or install client applications as needed.

(iii) Networking:

A reliable network infrastructure to connect clients to the server and enable data exchange.

(iv) Backup and Redundancy:

Backup systems or redundancy measures for data protection and system availability.

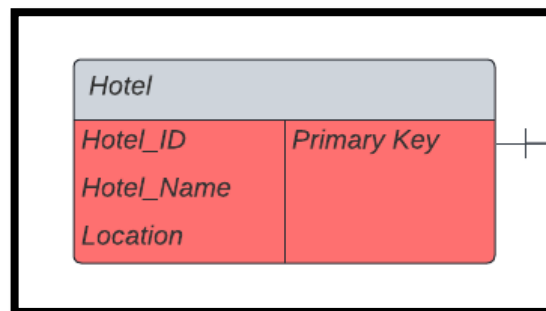
(v) Security Measures:

Security devices or practices, such as firewalls, intrusion detection systems, and security protocols.

Module Description

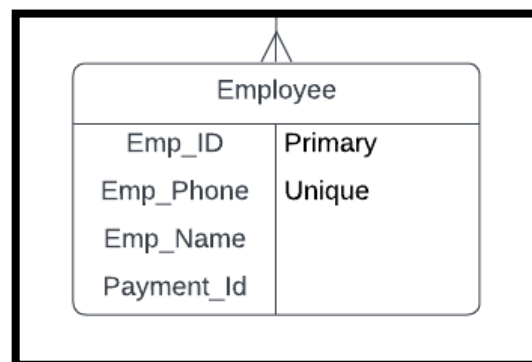
1)Hotel Table:

- Requirement:
- Each hotel should have a unique **Hotel_ID**.
- It should have a name (**Hotel_Name**).
- It should be located at a specific location.



2)Employee Table:

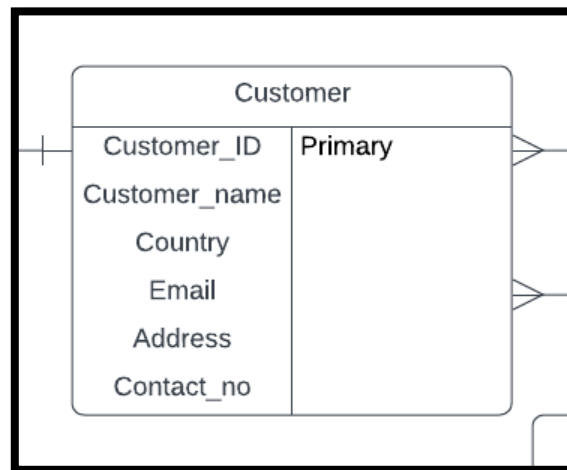
- Requirement:
- Each employee should have a unique **Emp_ID**.
- They should have a phone number (**Emp_Phone**), name (**Emp_Name**), and a **Payment_Id**.



3)Customer Table:

Requirement:

- Each customer should have a unique Customer_ID.
- They should have a name, country, a unique email, address, and a unique contact number.



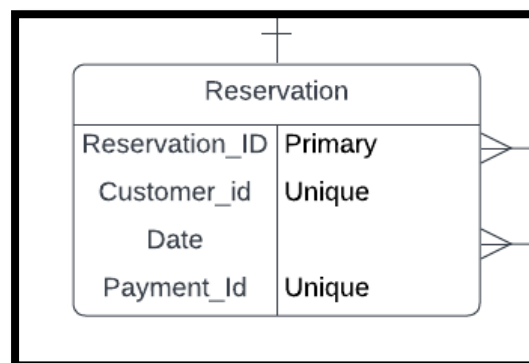
4)Reservation Table:

Requirement:

Each reservation should have a unique **Reservation_ID**.

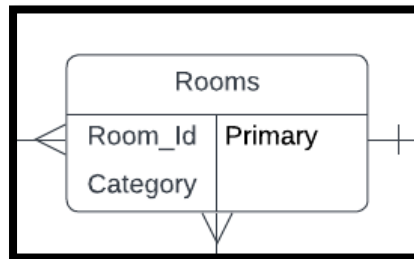
It should be associated with a **customer (Customer_id)**.

It should have a date and a unique **Payment_Id**.



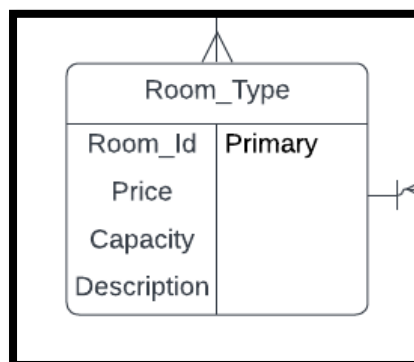
5)Rooms Table:

- Requirement:
- Each room should have a unique Room_Id.
- It should belong to a specific category.
-



6)RoomType Table:

- Requirement:
- Each room type should have a unique **Room_Id**.
- It should have a price, capacity, and a description.
-

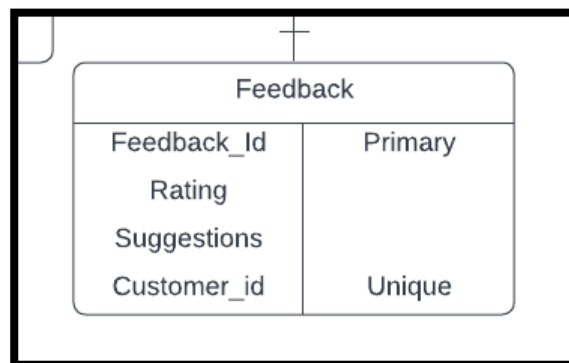


7)Feedback Table:

Requirement:

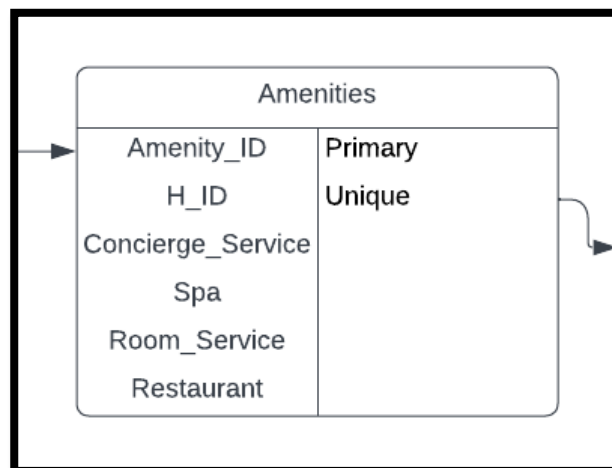
Each feedback should have a unique Feedback_Id.

It should include a rating, suggestions, and be associated with a customer (Customer_id).



8)Amenities Table:

- Requirements:
- It should have an **Amenity_ID** and should have a unique **H_ID**
- It should have services like **Concierge_Service** , **Spa** , **Room_Service** and **Restaurant**



RELATIONSHIPS

Concierge Service - Amenities:

Concierge Service may have a relationship with Amenities if the "Concierge_Service" attribute is true. This relationship may involve services provided by the concierge.

Amenities - Spa:

Amenities may have a relationship with Spa if the "Spa" attribute is true. This relationship may involve specific spa services available.

Amenities - Room_Service:

Amenities may have a relationship with Room_Service if the "Room_Service" attribute is true. This relationship may involve details of room service options.

Amenities - Restaurant:

Amenities may have a relationship with Restaurant if the "Restaurant" attribute is true. This relationship may involve information about the restaurant(s) available at the hotel.

Room_Service - Employee:

Room_Service is related to Employee via the emp_id foreign key. This relationship allows tracking the employees responsible for room service.

Spa - Amenities:

Spa has a relationship with Amenities via the Service_no. This relationship allows amenities to specify spa services.

Rooms - Hotel:

Rooms are related to Hotel. This relationship represents ownership, indicating that rooms are part of a specific hotel.

Restaurant - Hotel:

Restaurant is related to Hotel. This relationship represents ownership, indicating that the restaurant is part of a specific hotel.

Room_Type - Rooms:

Room_Type is related to Rooms. This relationship allows categorizing rooms into different types with varying prices and capacities.

Employee - Payment:

Employee is related to Payment via the Payment_Id attribute. This relationship connects employee information to payment details.

Reservation - Customer:

Reservation is related to Customer. This relationship associates reservations with the customer who made them.

Reservation - Payment:

Reservation is related to Payment via the Payment_Id attribute. This relationship links reservations to their corresponding payments.

Payment - Customer:

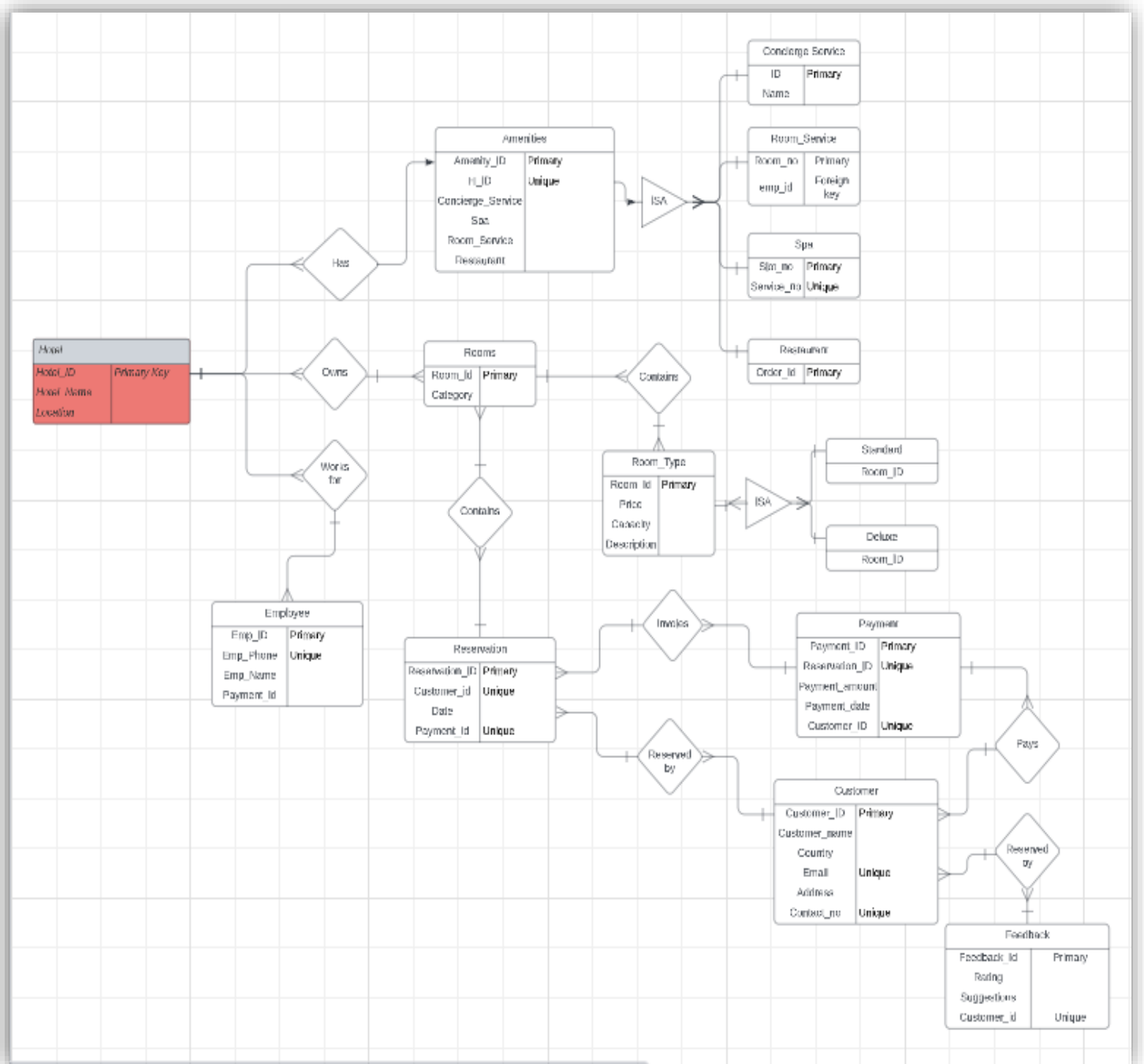
Payment is related to Customer via the Customer_ID attribute. This relationship connects payments to the customers who made them.

Feedback - Customer:

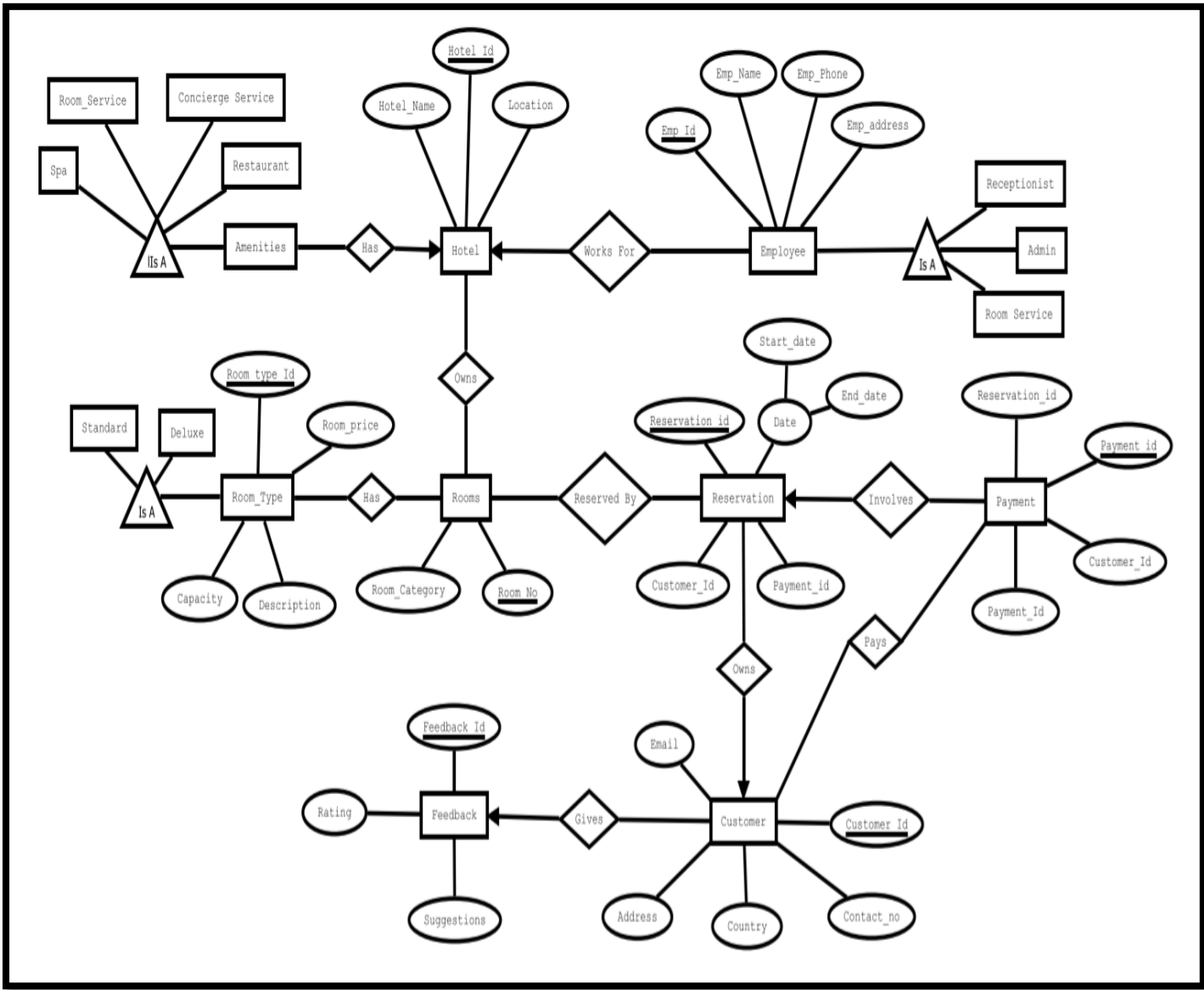
Feedback is related to Customer via the Customer_id attribute. This relationship links feedback to the customer who provided it.

CONCEPTUAL DATABASE DESIGN

LUCID CHART ER DIAGRAM



DIA TOOL ER DIAGRAM



LOGICAL DATABASE DESIGN

The Tables that are created according to the ER Diagram are

CUSTOMER

```
CREATE TABLE Customer (  
  customer_id INT PRIMARY KEY,  
  customer_name VARCHAR2(40),  
  Country VARCHAR(255),  
  email VARCHAR(255),  
  Address VARCHAR(255),  
  Contact_no VARCHAR(20)  
);
```

HOTEL

```
CREATE TABLE Hotel (  
  H_ID INT PRIMARY KEY,  
  H_NAME VARCHAR(255),  
  LOCATION VARCHAR(255)  
);
```

Name	Null?	Type
H_ID	NOT NULL	NUMBER (38)
H_NAME		VARCHAR2 (255)
LOCATION		VARCHAR2 (255)

AMENITIES

```
CREATE TABLE Amenities (  
  Amenity_ID INT PRIMARY KEY,  
  H_ID INT,  
  Concierge_Service varchar2(20),  
  Spa varchar2(20),  
  Room_Service varchar2(20),  
  Restaurant varchar2(20)  
);
```

Name	Null?	Type
AMENITY_ID	NOT NULL	NUMBER (38)
H_ID		NUMBER (38)
CONCIERGE_SERVICE		VARCHAR2 (20)
SPA		VARCHAR2 (20)
ROOM_SERVICE		VARCHAR2 (20)
RESTAURANT		VARCHAR2 (20)

LINKING HOTEL AND AMENITIES

CREATE TABLE Hotel_Amenities (

H_ID INT,

Amenity_ID INT,

FOREIGN KEY (H_ID) REFERENCES Hotel(H_ID),

FOREIGN KEY (Amenity_ID) REFERENCES Amenities(Amenity_ID)

);

Name	Null?	Type
-----	-----	-----
H_ID		NUMBER (38)
AMENITY_ID		NUMBER (38)

EMPLOYEE

CREATE TABLE Employee (

emp_id INT PRIMARY KEY,

emp_name VARCHAR(255),

emp_address VARCHAR(255),

emp_phone VARCHAR(20),

emp_salary number

);

Name	Null?	Type
-----	-----	-----
EMP_ID	NOT NULL	NUMBER (38)
EMP_NAME		VARCHAR2 (255)
EMP_ADDRESS		VARCHAR2 (255)
EMP_PHONE		VARCHAR2 (20)
EMP_SALARY		NUMBER

LINKING HOTEL TO EMPLOYEE

CREATE TABLE Hotel_Employee (

H_ID INT,

emp_id INT,

FOREIGN KEY (H_ID) REFERENCES Hotel(H_ID),

FOREIGN KEY (emp_id) REFERENCES Employee(emp_id)

);

Name	Null?	Type
-----	-----	-----
H_ID		NUMBER (38)
EMP_ID		NUMBER (38)

RECEPTIONIST EMPLOYEE

```
CREATE TABLE Receptionist_Employee (  
    emp_id INT,  
    FOREIGN KEY (emp_id) REFERENCES Employee(emp_id)  
);
```

Name	Null?	Type
EMP_ID		NUMBER(38)

ADMIN EMPLOYEE

```
CREATE TABLE Admin_Employee (  
    emp_id INT,  
    FOREIGN KEY (emp_id) REFERENCES Employee(emp_id)  
);
```

Name	Null?	Type
EMP_ID		NUMBER(38)

ROOMSERVICE EMPLOYEE

```
CREATE TABLE RoomService_Employee (  
    emp_id INT,  
    FOREIGN KEY (emp_id) REFERENCES Employee(emp_id)  
);
```

Name	Null?	Type
EMP_ID		NUMBER(38)

ROOM_TYPE

```
CREATE TABLE Room_Type (  
    room_type_id INT PRIMARY KEY,  
    room_price number,  
    description varchar2(50),  
    occupancy_capacity INT  
);
```

Name	Null?	Type
ROOM_TYPE_ID	NOT NULL	NUMBER(38)
ROOM_PRICE		NUMBER
DESCRIPTION		VARCHAR2(50)
OCCUPANCY_CAPACITY		NUMBER(38)

ROOMS

```
CREATE TABLE Rooms (  
    room_no INT PRIMARY KEY,  
    room_category VARCHAR(255),  
    H_ID INT,  
    room_type_id INT  
);
```

Name	Null?	Type
ROOM_NO	NOT NULL	NUMBER(38)
ROOM_CATEGORY		VARCHAR2(255)
H_ID		NUMBER(38)
ROOM_TYPE_ID		NUMBER(38)

LINKING HOTEL TO ROOMS

```
CREATE TABLE Hotel_Rooms (  
    H_ID INT,  
    room_no INT,  
    FOREIGN KEY (H_ID) REFERENCES Hotel(H_ID),  
    FOREIGN KEY (room_no) REFERENCES Rooms(room_no)  
);
```

Name	Null?	Type
H_ID		NUMBER(38)
ROOM_NO		NUMBER(38)

LINKING ROOMTYPE TO ROOMS

```
CREATE TABLE RoomType_Rooms (  
    room_no INT,  
    room_type_id INT,  
    FOREIGN KEY (room_no) REFERENCES Rooms(room_no),  
    FOREIGN KEY (room_type_id) REFERENCES Room_Type(room_type_id)  
);
```

Name	Null?	Type
ROOM_NO		NUMBER(38)
ROOM_TYPE_ID		NUMBER(38)

RESERVATION

```
CREATE TABLE Reservation (  
    reservation_id INT PRIMARY KEY,  
    st_date DATE,  
    end_date DATE,  
    customer_id INT,  
    payment_id INT,  
    description varchar2(20),  
    occupancy_capacity INT  
);
```

Name	Null?	Type
RESERVATION_ID	NOT NULL	NUMBER(38)
ST_DATE		DATE
END_DATE		DATE
CUSTOMER_ID		NUMBER(38)
PAYMENT_ID		NUMBER(38)
DESCRIPTION		VARCHAR2(20)
OCCUPANCY_CAPACITY		NUMBER(38)

PAYMENT

```
CREATE TABLE Payment (  
  payment_id INT PRIMARY KEY,  
  reservation_id INT,  
  payment_amount number,  
  payment_date DATE,  
  customer_id INT  
);
```

Name	Null?	Type
PAYMENT_ID	NOT NULL	NUMBER(38)
RESERVATION_ID		NUMBER(38)
PAYMENT_AMOUNT		NUMBER
PAYMENT_DATE		DATE
CUSTOMER_ID		NUMBER(38)

LINKING CUSTOMER TO RESERVATION

```
CREATE TABLE Customer_Reservation (  
  customer_id INT,  
  reservation_id INT,  
  FOREIGN KEY (customer_id) REFERENCES Customer(customer_id),  
  FOREIGN KEY (reservation_id) REFERENCES Reservation(reservation_id)  
);
```

Name	Null?	Type
CUSTOMER_ID		NUMBER(38)
RESERVATION_ID		NUMBER(38)

LINKING PAYMENT TO RESERVATION

```
CREATE TABLE Payment_Reservation (  
    payment_id INT,  
    reservation_id INT,  
    FOREIGN KEY (payment_id) REFERENCES Payment(payment_id),  
    FOREIGN KEY (reservation_id) REFERENCES Reservation(reservation_id)  
);
```

Name	Null?	Type
PAYMENT_ID		NUMBER(38)
RESERVATION_ID		NUMBER(38)

FEEDBACK

```
CREATE TABLE Feedback (  
    Feedback_id INT PRIMARY KEY,  
    Suggestions varchar2(20),  
    Rating INT,  
    customer_id INT  
);
```

Name	Null?	Type
FEEDBACK_ID	NOT NULL	NUMBER(38)
SUGGESTIONS		VARCHAR2(20)
RATING		NUMBER(38)
CUSTOMER_ID		NUMBER(38)

LINKING CUSTOMER TO FEEDBACK

```
CREATE TABLE Customer_Feedback (  
    customer_id INT,  
    Feedback_id INT,  
    FOREIGN KEY (customer_id) REFERENCES Customer(customer_id),  
    FOREIGN KEY (Feedback_id) REFERENCES Feedback(Feedback_id)  
);
```

Name	Null?	Type
CUSTOMER_ID		NUMBER(38)
FEEDBACK_ID		NUMBER(38)

Inserting records into each tables:

1. Customer table

INSERT INTO Customer values(1, 'Ravi Kumar', 'India', 'ravi.kumar@gmail.com', '1234 elm st, Delhi, India', '123-456-7890');

INSERT INTO Customer values(2, 'Neha Sharma', 'India', 'neha.sharma@gmail.com', '5678 maple ave, Mumbai, India', '987-654-3210');

INSERT INTO Customer values(3, 'Amit Jain', 'India', 'amit.jain@gmail.com', '4567 oxford st, Bangalore, India', '345-678-9012');

INSERT INTO Customer values(4, 'Priya Verma', 'India', 'priya.verma@gmail.com', '789 sydney rd, Chennai, India', '111-222-3333');

INSERT INTO Customer values(5, 'Sandeep Yadav', 'India', 'sandeep.yadav@gmail.com', '456 berlin strasse, Kolkata, India', '555-444-3333');

INSERT INTO Customer values(6, 'Ananya Mishra', 'India', 'ananya.mishra@gmail.com', '789 paris avenue, Hyderabad, India', '777-888-9999');

INSERT INTO Customer values(7, 'Akash Gupta', 'India', 'akash.gupta@gmail.com', '456 madrid blvd, Pune, India', '123-987-5678');

INSERT INTO Customer values(8, 'Shivani Sharma', 'India', 'shivani.sharma@gmail.com', '789 rome street, Jaipur, India', '890-345-6789');

INSERT INTO Customer values(9, 'Arjun Das', 'India', 'arjun.das@gmail.com', '456 tokyo ave, Chandigarh, India', '111-222-3333');

INSERT INTO Customer values(10, 'Deepa Mahajan', 'India', 'deepa.mahajan@gmail.com', '789 bangalore st, Ahmedabad, India', '555-444-3333');

CUSTOMER_ID	CUSTOMER_NAME	COUNTRY	EMAIL	ADDRESS	CONTACT_NO
1	Ravi Kumar	India	ravi.kumar@gmail.com	1234 elm st, Delhi, India	123-456-7890
2	Neha Sharma	India	neha.sharma@gmail.com	5678 maple ave, Mumbai, India	987-654-3210
3	Amit Jain	India	amit.jain@gmail.com	4567 oxford st, Bangalore, India	345-678-9012
4	Priya Verma	India	priya.verma@gmail.com	789 sydney rd, Chennai, India	111-222-3333
5	Sandeep Yadav	India	sandeep.yadav@gmail.com	456 berlin strasse, Kolkata, India	555-444-3333
6	Ananya Mishra	India	ananya.mishra@gmail.com	789 paris avenue, Hyderabad, India	777-888-9999
7	Akash Gupta	India	akash.gupta@gmail.com	456 madrid blvd, Pune, India	123-987-5678
8	Shivani Sharma	India	shivani.sharma@gmail.com	789 rome street, Jaipur, India	890-345-6789
9	Arjun Das	India	arjun.das@gmail.com	456 tokyo ave, Chandigarh, India	111-222-3333
10	Deepa Mahajan	India	deepa.mahajan@gmail.com	789 bangalore st, Ahmedabad, India	555-444-3333



2. Hotel table

INSERT INTO Hotel VALUES (1, 'Taj hotel', 'mumbai, maharashtra');

H_ID	H_NAME	LOCATION
1	Taj hotel	mumbai, maharashtra

3. Amenities table

```
INSERT INTO Amenities VALUES (1, 1, 'Yes', 'Yes', 'Yes', 'Yes');
INSERT INTO Amenities VALUES (2, 1, 'No', 'Yes', 'Yes', 'No');
INSERT INTO Amenities VALUES (3, 1, 'Yes', 'No', 'Yes', 'Yes');
INSERT INTO Amenities VALUES (4, 1, 'Yes', 'Yes', 'No', 'Yes');
INSERT INTO Amenities VALUES (5, 1, 'No', 'No', 'Yes', 'Yes');
INSERT INTO Amenities VALUES (6, 1, 'Yes', 'Yes', 'Yes', 'Yes');
INSERT INTO Amenities VALUES (7, 1, 'No', 'Yes', 'No', 'Yes');
INSERT INTO Amenities VALUES (8, 1, 'Yes', 'Yes', 'Yes', 'No');
INSERT INTO Amenities VALUES (9, 1, 'No', 'Yes', 'No', 'Yes');
INSERT INTO Amenities VALUES (10, 1, 'Yes', 'No', 'Yes', 'Yes');
```

AMENITY_ID	H_ID	CONCIERGE_SERVICE	SPA	ROOM_SERVICE	RESTAURANT
1	1	Yes	Yes	Yes	Yes
2	1	No	Yes	Yes	No
3	1	Yes	No	Yes	Yes
4	1	Yes	Yes	No	Yes
5	1	No	No	Yes	Yes
6	1	Yes	Yes	Yes	Yes
7	1	No	Yes	No	Yes
8	1	Yes	Yes	Yes	No
9	1	No	Yes	No	Yes
10	1	Yes	No	Yes	Yes

4. Hotel_Amenities table

```
INSERT INTO Hotel_Amenities VALUES (1, 1);  
INSERT INTO Hotel_Amenities VALUES (1, 2);  
INSERT INTO Hotel_Amenities VALUES (1, 3);  
INSERT INTO Hotel_Amenities VALUES (1, 4);  
INSERT INTO Hotel_Amenities VALUES (1, 5);  
INSERT INTO Hotel_Amenities VALUES (1, 6);  
INSERT INTO Hotel_Amenities VALUES (1, 7);  
INSERT INTO Hotel_Amenities VALUES (1, 8);  
INSERT INTO Hotel_Amenities VALUES (1, 9);  
INSERT INTO Hotel_Amenities VALUES (1, 10);
```

H_ID	AMENITY_ID
1	1
1	2
1	3
1	4
1	5
1	6
1	7
1	8
1	9
1	10

5. Employee table

```
INSERT INTO Employee VALUES (1, 'John Doe', '123 Main St, Some City, USA', '555-123-4567', 50000);
```

```
INSERT INTO Employee VALUES (2, 'Jane Smith', '456 Elm Ave, Another City, Canada', '555-987-6543', 48000);
```

```
INSERT INTO Employee VALUES (3, 'Rahul Sharma', '789 Park Rd, Delhi, India', '123-456-7890', 55000);
```

```
INSERT INTO Employee VALUES (4, 'Sneha Gupta', '567 River View, Mumbai, India', '987-654-3210', 49000);
```

```
INSERT INTO Employee VALUES (5, 'Amit Patel', '890 Green Lane, Bangalore, India', '111-222-3333', 51000);
```

```
INSERT INTO Employee VALUES (6, 'Nisha Verma', '456 Hilltop, Chennai, India', '555-444-3333', 47000);
```

```
INSERT INTO Employee VALUES (7, 'Raj Singh', '123 Sunset Blvd, Hyderabad, India', '777-888-9999', 52000);
```

```
INSERT INTO Employee VALUES (8, 'Aruna Kumari', '789 Valley Rd, Pune, India', '123-987-5678', 48000);
```

```
INSERT INTO Employee VALUES (9, 'Vinod Sharma', '456 Garden Ave, Jaipur, India', '890-345-6789', 53000);
```

```
INSERT INTO Employee VALUES (10, 'Sarika Jain', '789 Lakeview, Ahmedabad, India', '555-444-3333', 49000);
```

EMP_ID	EMP_NAME	EMP_ADDRESS	EMP_PHONE	EMP_SALARY
1	John Doe	123 Main St, Some City, USA	555-123-4567	50000
2	Jane Smith	456 Elm Ave, Another City, Canada	555-987-6543	48000
3	Rahul Sharma	789 Park Rd, Delhi, India	123-456-7890	55000
4	Sneha Gupta	567 River View, Mumbai, India	987-654-3210	49000
5	Amit Patel	890 Green Lane, Bangalore, India	111-222-3333	51000
6	Nisha Verma	456 Hilltop, Chennai, India	555-444-3333	47000
7	Raj Singh	123 Sunset Blvd, Hyderabad, India	777-888-9999	52000
8	Aruna Kumari	789 Valley Rd, Pune, India	123-987-5678	48000
9	Vinod Sharma	456 Garden Ave, Jaipur, India	890-345-6789	53000
10	Sarika Jain	789 Lakeview, Ahmedabad, India	555-444-3333	49000

6. Hotel_Employee table

```
INSERT INTO Hotel_Employee VALUES (1, 1);  
INSERT INTO Hotel_Employee VALUES (1, 2);  
INSERT INTO Hotel_Employee VALUES (1, 3);  
INSERT INTO Hotel_Employee VALUES (1, 4);  
INSERT INTO Hotel_Employee VALUES (1, 5);  
INSERT INTO Hotel_Employee VALUES (1, 6);  
INSERT INTO Hotel_Employee VALUES (1, 7);  
INSERT INTO Hotel_Employee VALUES (1, 8);  
INSERT INTO Hotel_Employee VALUES (1, 9);  
INSERT INTO Hotel_Employee VALUES (1, 10);
```

H_ID	EMP_ID
1	1
1	2
1	3
1	4
1	5
1	6
1	7
1	8
1	9
1	10

7. Receptionist_Employee table

```
INSERT INTO Receptionist_Employee VALUES (1);  
INSERT INTO Receptionist_Employee VALUES (2);  
INSERT INTO Receptionist_Employee VALUES (3);  
INSERT INTO Receptionist_Employee VALUES (4);  
INSERT INTO Receptionist_Employee VALUES (5);  
INSERT INTO Receptionist_Employee VALUES (6);  
INSERT INTO Receptionist_Employee VALUES (7);  
INSERT INTO Receptionist_Employee VALUES (8);  
INSERT INTO Receptionist_Employee VALUES (9);  
INSERT INTO Receptionist_Employee VALUES (10);
```

EMP_ID
1
2
3
4
5
6
7
8
9
10

8. Admin_Employee table:

```
INSERT INTO Admin_Employee VALUES (1);  
INSERT INTO Admin_Employee VALUES (2);  
INSERT INTO Admin_Employee VALUES (3);  
INSERT INTO Admin_Employee VALUES (4);  
INSERT INTO Admin_Employee VALUES (5);  
INSERT INTO Admin_Employee VALUES (6);  
INSERT INTO Admin_Employee VALUES (7);  
INSERT INTO Admin_Employee VALUES (8);  
INSERT INTO Admin_Employee VALUES (9);  
INSERT INTO Admin_Employee VALUES (10);
```

EMP_ID
1
2
3
4
5
6
7
8
9
10

9. RoomService_Employee table

```
INSERT INTO RoomService_Employee VALUES (1);  
INSERT INTO RoomService_Employee VALUES (2);  
INSERT INTO RoomService_Employee VALUES (3);  
INSERT INTO RoomService_Employee VALUES (4);  
INSERT INTO RoomService_Employee VALUES (5);  
INSERT INTO RoomService_Employee VALUES (6);  
INSERT INTO RoomService_Employee VALUES (7);  
INSERT INTO RoomService_Employee VALUES (8);  
INSERT INTO RoomService_Employee VALUES (9);  
INSERT INTO RoomService_Employee VALUES (10);
```

EMP_ID
1
2
3
4
5
6
7
8
9
10

10. Rooms table

```
INSERT INTO Rooms VALUES (101, 'Standard', 1, 1);
INSERT INTO Rooms VALUES (102, 'Standard', 1, 1);
INSERT INTO Rooms VALUES (103, 'Deluxe', 1, 2);
INSERT INTO Rooms VALUES (104, 'Deluxe', 1, 2);
INSERT INTO Rooms VALUES (105, 'Suite', 1, 3);
INSERT INTO Rooms VALUES (106, 'Suite', 1, 3);
INSERT INTO Rooms VALUES (107, 'Standard', 1, 4);
INSERT INTO Rooms VALUES (108, 'Standard', 1, 4);
INSERT INTO Rooms VALUES (109, 'Deluxe', 1, 5);
INSERT INTO Rooms VALUES (110, 'Deluxe', 1, 5);
```

ROOM_NO	ROOM_CATEGORY	H_ID	ROOM_TYPE_ID
101	Standard	1	1
102	Standard	1	1
103	Deluxe	1	2
104	Deluxe	1	2
105	Suite	1	3
106	Suite	1	3
107	Standard	1	4
108	Standard	1	4
109	Deluxe	1	5
110	Deluxe	1	5

11. Room_Type table

```
INSERT INTO Room_Type VALUES (1, 100, 'Standard Room', 2);
```

```
INSERT INTO Room_Type VALUES (2, 120, 'Standard Room with View', 2);
```

```
INSERT INTO Room_Type VALUES (3, 150, 'Deluxe Room', 3);
```

```
INSERT INTO Room_Type VALUES (4, 180, 'Suite', 4);
```

```
INSERT INTO Room_Type VALUES (5, 200, 'Executive Suite', 4);
```

```
INSERT INTO Room_Type VALUES (6, 90, 'Economy Room', 1);
```

```
INSERT INTO Room_Type VALUES (7, 140, 'Deluxe Room with View', 3);
```

```
INSERT INTO Room_Type VALUES (8, 220, 'Presidential Suite', 6);
```

```
INSERT INTO Room_Type VALUES (9, 110, 'Superior Room', 2);
```

```
INSERT INTO Room_Type VALUES (10, 130, 'Family Suite', 5);
```

ROOM_TYPE_ID	ROOM_PRICE	DESCRIPTION	OCCUPANCY_CAPACITY
1	100	Standard Room	2
2	120	Standard Room with View	2
3	150	Deluxe Room	3
4	180	Suite	4
5	200	Executive Suite	4
6	90	Economy Room	1
7	140	Deluxe Room with View	3
8	220	Presidential Suite	6
9	110	Superior Room	2
10	130	Family Suite	5

12. Hotel_Rooms table

```
INSERT INTO Hotel_Rooms VALUES (1, 101);  
INSERT INTO Hotel_Rooms VALUES (1, 102);  
INSERT INTO Hotel_Rooms VALUES (1, 103);  
INSERT INTO Hotel_Rooms VALUES (1, 104);  
INSERT INTO Hotel_Rooms VALUES (1, 105);  
INSERT INTO Hotel_Rooms VALUES (1, 106);  
INSERT INTO Hotel_Rooms VALUES (1, 107);  
INSERT INTO Hotel_Rooms VALUES (1, 108);  
INSERT INTO Hotel_Rooms VALUES (1, 109);  
INSERT INTO Hotel_Rooms VALUES (1, 110);
```

H_ID	ROOM_NO
1	101
1	102
1	103
1	104
1	105
1	106
1	107
1	108
1	109
1	110

13. RoomType_Rooms table

```
INSERT INTO RoomType_Rooms VALUES (101, 1);  
INSERT INTO RoomType_Rooms VALUES (102, 2);  
INSERT INTO RoomType_Rooms VALUES (103, 3);  
INSERT INTO RoomType_Rooms VALUES (104, 4);  
INSERT INTO RoomType_Rooms VALUES (105, 5);  
INSERT INTO RoomType_Rooms VALUES (106, 6);  
INSERT INTO RoomType_Rooms VALUES (107, 7);  
INSERT INTO RoomType_Rooms VALUES (108, 8);  
INSERT INTO RoomType_Rooms VALUES (109, 9);  
INSERT INTO RoomType_Rooms VALUES (110, 10);
```

ROOM_NO	ROOM_TYPE_ID
101	1
102	2
103	3
104	4
105	5
106	6
107	7
108	8
109	9
110	10

14. Reservation table

INSERT INTO Reservation VALUES (1, '20-Oct-2023', '22-Oct-2023', 1, 1, 'Standard reservation', 2);

INSERT INTO Reservation VALUES (2, '22-Oct-2023', '25-Oct-2023', 2, 2, 'Deluxe reservation', 2);

INSERT INTO Reservation VALUES (3, '24-Oct-2023', '27-Oct-2023', 3, 3, 'Suite reservation', 4);

INSERT INTO Reservation VALUES (4, '26-Oct-2023', '30-Oct-2023', 4, 4, 'Standard reservation', 2);

INSERT INTO Reservation VALUES (5, '28-Oct-2023', '31-Oct-2023', 5, 5, 'Deluxe reservation', 2);

INSERT INTO Reservation VALUES (6, '30-Oct-2023', '03-Nov-2023', 6, 6, 'Suite reservation', 4);

INSERT INTO Reservation VALUES (7, '01-Nov-2023', '05-Nov-2023', 7, 7, 'Standard reservation', 2);

INSERT INTO Reservation VALUES (8, '03-Nov-2023', '08-Nov-2023', 8, 8, 'Deluxe reservation', 2);

INSERT INTO Reservation VALUES (9, '05-Nov-2023', '10-Nov-2023', 9, 9, 'Suite reservation', 4);

INSERT INTO Reservation VALUES (10, '07-Nov-2023', '12-Nov-2023', 10, 10, 'Standard reservation', 2);

RESERVATION_ID	ST_DATE	END_DATE	CUSTOMER_ID	PAYMENT_ID	DESCRIPTION	OCCUPANCY_CAPACITY
1	20-OCT-23	22-OCT-23	1	1	Standard reservation	2
2	22-OCT-23	25-OCT-23	2	2	Deluxe reservation	2
3	24-OCT-23	27-OCT-23	3	3	Suite reservation	4
4	26-OCT-23	30-OCT-23	4	4	Standard reservation	2
5	28-OCT-23	31-OCT-23	5	5	Deluxe reservation	2
6	30-OCT-23	03-NOV-23	6	6	Suite reservation	4
7	01-NOV-23	05-NOV-23	7	7	Standard reservation	2
8	03-NOV-23	08-NOV-23	8	8	Deluxe reservation	2
9	05-NOV-23	10-NOV-23	9	9	Suite reservation	4
10	07-NOV-23	12-NOV-23	10	10	Standard reservation	2

15. Payment table

```
INSERT INTO Payment VALUES (1, 1, 500.00, '21-OCT-2023', 1);
INSERT INTO Payment VALUES (2, 2, 600.00, '23-OCT-2023', 2);
INSERT INTO Payment VALUES (3, 3, 700.00, '25-OCT-2023', 3);
INSERT INTO Payment VALUES (4, 4, 800.00, '27-OCT-2023', 4);
INSERT INTO Payment VALUES (5, 5, 900.00, '29-OCT-2023', 5);
INSERT INTO Payment VALUES (6, 6, 1000.00, '31-OCT-2023', 6);
INSERT INTO Payment VALUES (7, 7, 1100.00, '02-NOV-2023', 7);
INSERT INTO Payment VALUES (8, 8, 1200.00, '04-NOV-2023', 8);
INSERT INTO Payment VALUES (9, 9, 1300.00, '06-NOV-2023', 9);
INSERT INTO Payment VALUES (10, 10, 1400.00, '08-NOV-2023', 10);
```

PAYMENT_ID	RESERVATION_ID	PAYMENT_AMOUNT	PAYMENT_DATE	CUSTOMER_ID
1	1	500	21-OCT-23	1
2	2	600	23-OCT-23	2
3	3	700	25-OCT-23	3
4	4	800	27-OCT-23	4
5	5	900	29-OCT-23	5
6	6	1000	31-OCT-23	6
7	7	1100	02-NOV-23	7
8	8	1200	04-NOV-23	8
9	9	1300	06-NOV-23	9
10	10	1400	08-NOV-23	10

16. Customer_Reservation table

```
INSERT INTO Customer_Reservation VALUES (1, 1);  
INSERT INTO Customer_Reservation VALUES (2, 2);  
INSERT INTO Customer_Reservation VALUES (3, 3);  
INSERT INTO Customer_Reservation VALUES (4, 4);  
INSERT INTO Customer_Reservation VALUES (5, 5);  
INSERT INTO Customer_Reservation VALUES (6, 6);  
INSERT INTO Customer_Reservation VALUES (7, 7);  
INSERT INTO Customer_Reservation VALUES (8, 8);  
INSERT INTO Customer_Reservation VALUES (9, 9);  
INSERT INTO Customer_Reservation VALUES (10, 10);
```

CUSTOMER_ID	RESERVATION_ID
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	10

17. Payment_Reservation table

```
INSERT INTO Payment_Reservation VALUES (1, 1);  
INSERT INTO Payment_Reservation VALUES (2, 2);  
INSERT INTO Payment_Reservation VALUES (3, 3);  
INSERT INTO Payment_Reservation VALUES (4, 4);  
INSERT INTO Payment_Reservation VALUES (5, 5);  
INSERT INTO Payment_Reservation VALUES (6, 6);  
INSERT INTO Payment_Reservation VALUES (7, 7);  
INSERT INTO Payment_Reservation VALUES (8, 8);  
INSERT INTO Payment_Reservation VALUES (9, 9);  
INSERT INTO Payment_Reservation VALUES (10, 10);
```

PAYMENT_ID	RESERVATION_ID
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	10

18. Feedback table

```
INSERT INTO Feedback VALUES (1, 'Good service', 4, 1);
INSERT INTO Feedback VALUES (2, 'Amazing experience', 5, 2);
INSERT INTO Feedback VALUES (3, 'Satisfactory stay', 3, 3);
INSERT INTO Feedback VALUES (4, 'Excellent service', 5, 4);
INSERT INTO Feedback VALUES (5, 'Poor experience', 1, 5);
INSERT INTO Feedback VALUES (6, 'Average service', 2, 6);
INSERT INTO Feedback VALUES (7, 'Wonderful stay', 5, 7);
INSERT INTO Feedback VALUES (8, 'Great hospitality', 5, 8);
INSERT INTO Feedback VALUES (9, 'Fantastic experience', 5, 9);
INSERT INTO Feedback VALUES (10, 'Exceptional service', 5,
10);
```

FEEDBACK_ID	SUGGESTIONS	RATING	CUSTOMER_ID
1	Good service	4	1
2	Amazing experience	5	2
3	Satisfactory stay	3	3
4	Excellent service	5	4
5	Poor experience	1	5
6	Average service	2	6
7	Wonderful stay	5	7
8	Great hospitality	5	8
9	Fantastic experience	5	9
10	Exceptional service	5	10

19. Customer_Feedback table

```
INSERT INTO Customer_Feedback VALUES (1, 1);  
INSERT INTO Customer_Feedback VALUES (2, 2);  
INSERT INTO Customer_Feedback VALUES (3, 3);  
INSERT INTO Customer_Feedback VALUES (4, 4);  
INSERT INTO Customer_Feedback VALUES (5, 5);  
INSERT INTO Customer_Feedback VALUES (6, 6);  
INSERT INTO Customer_Feedback VALUES (7, 7);  
INSERT INTO Customer_Feedback VALUES (8, 8);  
INSERT INTO Customer_Feedback VALUES (9, 9);  
INSERT INTO Customer_Feedback VALUES (10,10);
```

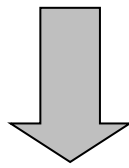
CUSTOMER_ID	FEEDBACK_ID
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	10

SQL QUERIES for Hotel Management System

(a)

1) Retrieve the names and email addresses of all customers.

```
SELECT customer_id, customer_name, email FROM Customer;
```



CUSTOMER_ID	CUSTOMER_NAME	EMAIL
1	Ravi Kumar	ravi.kumar@gmail.com
2	Neha Sharma	neha.sharma@gmail.com
3	Amit Jain	amit.jain@gmail.com
4	Priya Verma	priya.verma@gmail.com
5	Sandeep Yadav	sandeep.yadav@gmail.com
6	Ananya Mishra	ananya.mishra@gmail.com
7	Akash Gupta	akash.gupta@gmail.com
8	Shivani Sharma	shivani.sharma@gmail.com
9	Arjun Das	arjun.das@gmail.com
10	Deepa Mahajan	deepa.mahajan@gmail.com

(2) List the amenities for the hotel.

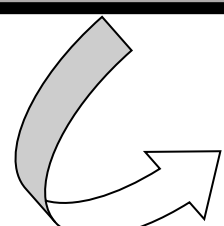
```
SELECT Amenity_ID, Concierge_Service, Spa, Room_Service,  
Restaurant  
FROM Amenities;
```



AMENITY_ID	CONCIERGE_SERVICE	SPA	ROOM_SERVICE	RESTAURANT
1	Yes	Yes	Yes	Yes
2	No	Yes	Yes	No
3	Yes	No	Yes	Yes
4	Yes	Yes	No	Yes
5	No	No	Yes	Yes
6	Yes	Yes	Yes	Yes
7	No	Yes	No	Yes
8	Yes	Yes	Yes	No
9	No	Yes	No	Yes
10	Yes	No	Yes	Yes

(3) Find Customer Reservations: Retrieve all reservations made by a specific customer.

```
SELECT R.reservation_id, R.st_date, R.end_date  
FROM Reservation R  
WHERE R.customer_id = 6;
```



RESERVATION_ID	ST_DATE	END_DATE
6	30-OCT-23	03-NOV-23

(4) Find the customer details who made payments on '29-OCT-2023'.

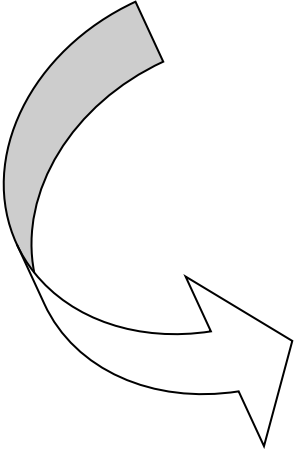
```
SELECT C.customer_id, C.email  
FROM Customer C, Payment P  
WHERE C.customer_id = P.customer_id
```



CUSTOMER_ID	EMAIL
5	sandeep.yadav@gmail.com

(5) Show the room numbers and their categories in the hotel.

```
SELECT HR.room_no, R.room_category  
FROM Hotel_Rooms HR, Rooms R  
WHERE HR.H_ID = 1 AND HR.room_no = R.room_no;
```

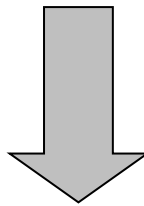


ROOM_NO	ROOM_CATEGORY
101	Standard
102	Standard
103	Deluxe
104	Deluxe
105	Suite
106	Suite
107	Standard
108	Standard
109	Deluxe
110	Deluxe

(b)

(1) Retrieve the names of customers who provided feedback.

```
SELECT C.customer_id, C.email, CF.feedback_id  
FROM Customer C  
JOIN Customer_Feedback CF ON C.customer_id =  
CF.customer_id;
```

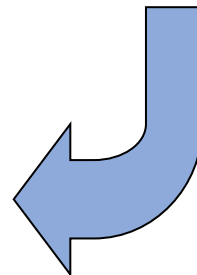


CUSTOMER_ID	EMAIL	FEEDBACK_ID
1	ravi.kumar@gmail.com	1
2	neha.sharma@gmail.com	2
3	amit.jain@gmail.com	3
4	priya.verma@gmail.com	4
5	sandeep.yadav@gmail.com	5
6	ananya.mishra@gmail.com	6
7	akash.gupta@gmail.com	7
8	shivani.sharma@gmail.com	8
9	arjun.das@gmail.com	9
10	deepa.mahajan@gmail.com	10

(2) List all the customers who made reservations for the Suite category.

```
SELECT C.customer_id, C.email
FROM Customer C
JOIN Customer_Reservation CR ON C.customer_id =
CR.customer_id
JOIN Reservation R ON CR.reservation_id = R.reservation_id
JOIN Room_Type RT ON R.occupancy_capacity =
RT.occupancy_capacity
WHERE RT.description = 'Suite';
```

CUSTOMER_ID	EMAIL
3	amit.jain@gmail.com
6	ananya.mishra@gmail.com
9	arjun.das@gmail.com



(3) Calculate the total revenue generated by the hotel from room reservations.

```
SELECT SUM(P.payment_amount) AS total_revenue
FROM Payment P
JOIN Reservation R ON P.reservation_id = R.reservation_id;
```

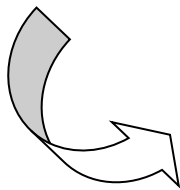


TOTAL_REVENUE
9500

(c)

(1) Find the average rating of feedback received from customers.

```
SELECT AVG(CF.Rating) AS average_rating  
FROM Feedback CF;
```



AVERAGE_RATING
4

(2) Retrieve the highest-paid staff member.

```
SELECT e.emp_id, e.emp_name  
FROM Employee e  
WHERE e.emp_salary = (SELECT MAX(emp_salary) FROM  
Employee);
```

EMP_ID	EMP_NAME
3	Rahul Sharma

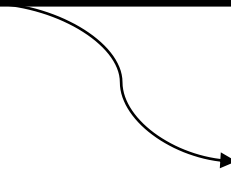
(3) Calculate the total number of customers in the database.

```
SELECT COUNT(customer_id) AS total_customers  
FROM Customer;
```

TOTAL_CUSTOMERS
10

(4) Determine the maximum payment amount made in any reservation.

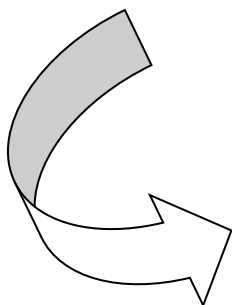
```
SELECT MAX(payment_amount) AS max_payment_amount  
FROM Payment;
```



MAX_PAYMENT_AMOUNT
1400

(5) Retrieve the highest and lowest room prices and display room type:

```
SELECT description, room_price FROM Room_Type WHERE  
room_price = (SELECT MAX(room_price) FROM Room_Type)  
union all  
SELECT description, room_price FROM Room_Type WHERE  
room_price = (SELECT MIN(room_price) FROM Room_Type);
```



DESCRIPTION	ROOM_PRICE
Presidential Suite	220
Economy Room	90

(d)

(1) Create a view that displays rooms with prices higher than 150.

```
CREATE VIEW V1 AS
```

```
SELECT r.room_no, r.room_category, rt.description AS  
RoomType, rt.room_price
```

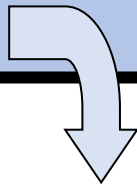
```
FROM Rooms r, Room_Type rt
```

```
WHERE r.room_type_id = rt.room_type_id AND rt.room_price  
> 150;
```



View created.

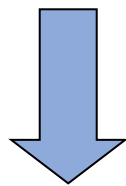
```
SELECT * FROM V1;
```



ROOM_NO	ROOM_CATEGORY	ROOMTYPE	ROOM_PRICE
107	Standard	Suite	180
108	Standard	Suite	180
109	Deluxe	Executive Suite	200
110	Deluxe	Executive Suite	200

(2) Create a view that displays employee names and salaries.

```
CREATE VIEW V2 AS  
SELECT emp_name AS EmployeeName, emp_salary AS Salary  
FROM Employee;  
SELECT * FROM V2;
```



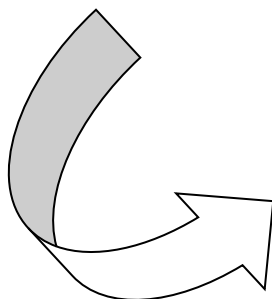
View created.

EMPLOYEENAME	SALARY
John Doe	50000
Jane Smith	48000
Rahul Sharma	55000
Sneha Gupta	49000
Amit Patel	51000
Nisha Verma	47000
Raj Singh	52000
Aruna Kumari	48000
Vinod Sharma	53000
Sarika Jain	49000

(e)

(1) Find the total payment amount for reservations at your hotel with a room price greater than 1000/-:

```
SELECT SUM(payment_amount) AS TotalPayment
FROM Payment
WHERE reservation_id IN (
  SELECT rt.reservation_id
  FROM Reservation rt
  WHERE rt.reservation_id IN (
    SELECT pr.reservation_id
    FROM Reservation rt, Payment_Reservation pr, Payment p
    WHERE rt.reservation_id = pr.reservation_id
    AND p.payment_id = pr.payment_id
    AND p.payment_amount > 1000
  )
);
```



TOTALPAYMENT
5000

(2) Retrieve the customers who made reservations at the hotel for 'Suite' rooms:

```
SELECT c.customer_name, c.customer_id
FROM Customer c
WHERE c.customer_id NOT IN (
    SELECT r.customer_id
    FROM Reservation r
    WHERE r.reservation_id NOT IN (SELECT reservation_id
    FROM Reservation WHERE description = 'Suite reservation') );
```

CUSTOMER_NAME	CUSTOMER_ID
Ananya Mishra	6
Amit Jain	3
Arjun Das	9

(3) Find the top 5 highest paid employees in the hotel:

```
SELECT emp_name, emp_salary
FROM Employee e
WHERE 5 >= (
    SELECT COUNT(DISTINCT emp_salary)
    FROM Employee e1
    WHERE e1.emp_salary >= e.emp_salary
)
ORDER BY emp_salary DESC;
```

EMP_NAME	EMP_SALARY
Rahul Sharma	55000
Vinod Sharma	53000
Raj Singh	52000
Amit Patel	51000
John Doe	50000

(4) Retrieve the customers who provided feedback with a rating of 5 at your hotel:

```
SELECT customer_name, customer_id, email
FROM Customer
WHERE customer_id IN (
    SELECT customer_id
    FROM Feedback
    WHERE Rating = 5
);
```

CUSTOMER_NAME	CUSTOMER_ID	EMAIL
Neha Sharma	2	neha.sharma@gmail.com
Priya Verma	4	priya.verma@gmail.com
Akash Gupta	7	akash.gupta@gmail.com
Shivani Sharma	8	shivani.sharma@gmail.com
Arjun Das	9	arjun.das@gmail.com
Deepa Mahajan	10	deepa.mahajan@gmail.com

(5) Find the reservations with the earliest start date:

```
SELECT reservation_id, st_date
FROM Reservation
WHERE st_date = (
    SELECT MIN(st_date)
    FROM Reservation
);
```

RESERVATION_ID	ST_DATE
1	20-OCT-23

(6) Find the total number of reservations made for each room category at the hotel:

```
SELECT description, COUNT(*)  
FROM Reservation  
GROUP BY description;
```

DESCRIPTION	COUNT(*)
Suite reservation	3
Deluxe reservation	3
Standard reservation	4

(f)

(1) Calculate and display the total hotel revenue

```
DECLARE  
total_revenue NUMBER := 0;  
BEGIN  
    SELECT SUM(payment_amount) INTO total_revenue  
    FROM Payment;  
    DBMS_OUTPUT.PUT_LINE('Total Hotel Revenue: ' ||  
total_revenue);  
END;
```

```
Statement processed.  
Total Hotel Revenue: 9500
```

(2) Retrieve the highest and lowest room prices and their descriptions:

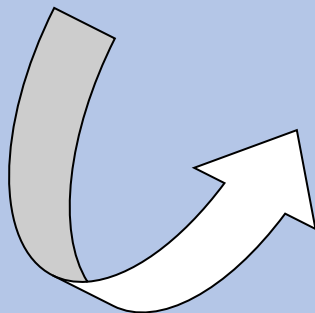
```
DECLARE
  highest_room_description VARCHAR2(50);
  highest_room_price NUMBER;
  lowest_room_description VARCHAR2(50);
  lowest_room_price NUMBER;
BEGIN
  SELECT description, room_price
  INTO highest_room_description, highest_room_price
  FROM Room_Type
  WHERE room_price = (SELECT MAX(room_price) FROM
Room_Type);
  SELECT description, room_price
  INTO lowest_room_description, lowest_room_price
  FROM Room_Type
  WHERE room_price = (SELECT MIN(room_price) FROM
Room_Type);
  DBMS_OUTPUT.PUT_LINE('Highest Room Description: ' ||
highest_room_description || ', Price: ' || highest_room_price);
  DBMS_OUTPUT.PUT_LINE('Lowest Room Description: ' ||
lowest_room_description || ', Price: ' || lowest_room_price);
END;
END;
```



```
Statement processed.
Highest Room Description: Presidential Suite, Price: 220
Lowest Room Description: Economy Room, Price: 90
```

(3) Retrieve Customer Names Who Only Booked "Suite" Reservations:

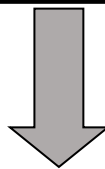
```
DECLARE
    v_customer_name VARCHAR2(255);
BEGIN
    FOR c IN (SELECT c.customer_name
              FROM Customer c
              WHERE NOT EXISTS (
                  SELECT 1
                  FROM Reservation r
                  WHERE r.customer_id = c.customer_id
                  AND r.description = 'Suite reservation'
              ))
    LOOP
        v_customer_name := c.customer_name;
        DBMS_OUTPUT.PUT_LINE('Customer Name: ' ||
v_customer_name);
    END LOOP;
END;
```



```
Statement processed.
Customer Name: Ravi Kumar
Customer Name: Akash Gupta
Customer Name: Neha Sharma
Customer Name: Shivani Sharma
Customer Name: Priya Verma
Customer Name: Sandeep Yadav
Customer Name: Deepa Mahajan
```


(4) Retrieve the Names and Salaries of the Top 3 Highest-Earning Employees:

```
DECLARE  
  TYPE emp_salary_list IS TABLE OF NUMBER;  
  emp_salaries emp_salary_list;  
BEGIN  
  SELECT emp_salary  
  BULK COLLECT INTO emp_salaries  
  FROM Employee  
  ORDER BY emp_salary DESC;  
  FOR i IN 1..3  
  LOOP  
    DBMS_OUTPUT.PUT_LINE('Employee Salary: ' ||  
emp_salaries(i));  
  END LOOP;  
END;
```



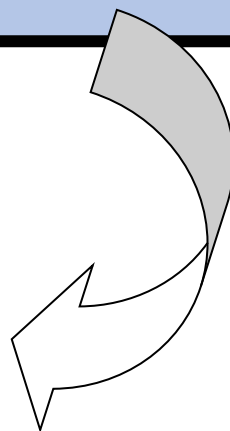
```
Statement processed.  
Employee Salary: 55000  
Employee Salary: 53000  
Employee Salary: 52000
```

(g)

(1) Fetching Employee Names using Cursor:

```
DECLARE  
    employee_name VARCHAR2(50);  
    CURSOR emp_cursor IS  
        SELECT emp_name  
        FROM Employee;  
BEGIN  
    FOR emp_rec IN emp_cursor  
    LOOP  
        employee_name := emp_rec.emp_name;  
        DBMS_OUTPUT.PUT_LINE('Employee Name: ' ||  
employee_name);  
    END LOOP;  
END;
```

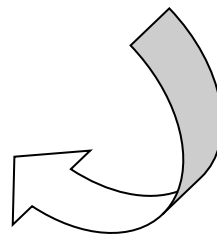
```
Statement processed.  
Employee Name: John Doe  
Employee Name: Jane Smith  
Employee Name: Rahul Sharma  
Employee Name: Sneha Gupta  
Employee Name: Amit Patel  
Employee Name: Nisha Verma  
Employee Name: Raj Singh  
Employee Name: Aruna Kumari  
Employee Name: Vinod Sharma  
Employee Name: Sarika Jain
```



(2) Calculating Total Salary using Cursor:

```
DECLARE
    total_salary NUMBER := 0;
    CURSOR emp_cursor IS
        SELECT emp_salary
        FROM Employee;
BEGIN
    FOR emp_rec IN emp_cursor
    LOOP
        total_salary := total_salary +
emp_rec.emp_salary;
    END LOOP;
    DBMS_OUTPUT.PUT_LINE('Total Salary: ' ||
total_salary);
END;
```

```
Statement processed.
Total Salary: 502000
```



CONCLUSION:

In conclusion, the hotel management system described in the provided case study offers a comprehensive solution for efficiently managing various aspects of hotel operations. The system includes entities such as Concierge Service, Amenities, Room Service, Spa, Hotel, Rooms, Restaurant, Room Type, Employee, Reservation, Payment, Customer, and Feedback, each with specific attributes and relationships designed to meet the unique requirements of the hotel industry.

The data model's relationships ensure that services, reservations, payments, and customer feedback are well-integrated and accessible, allowing for streamlined hotel operations and improved guest experiences. The system's data validation and security measures are crucial for maintaining data integrity, ensuring that only authorized personnel can access sensitive information.

THE END