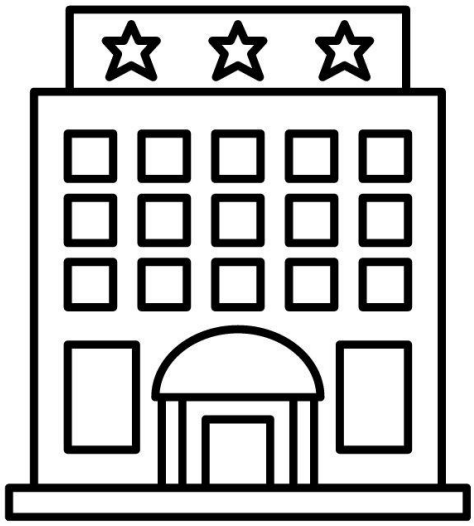
UNIVERSITETI®
METROPOLITAN
TIRANA



HOTEL MANAGEMENT DATABASE

-SQL codes included!!!

MySQL®

NIKOLIAN BEHARI

INTRODUCTION: Hotel Management Database

Abstract

This project focuses on the design and development of a hotel management database system that streamlines hotel operations for improved guest experience. The main goals of the project were to provide a centralized location for storing and managing all information related to hotel operations, to improve the efficiency of daily tasks for hotel staff, and to provide a range of features for guests to manage their stay more effectively.

The system was designed to meet the needs of both hotel management and guests, with an easy-to-use interface that allows hotel staff to view and manage all aspects of the hotel's operations from a single location. The system handles managing room inventory, handling reservations and cancellations, managing staff schedules and payroll, monitoring guest feedback, and generating reports on the hotel's performance. For guests, the system provides features that include making reservations online or over the phone, viewing room availability and rates, checking in and out of the hotel, ordering room service, requesting housekeeping or maintenance services, and providing feedback on their stay.

This project outlines the design and development process of the hotel management database system, including the data modeling, database design, and MySQL queries. The system was implemented using a MySQL database. The system's effectiveness was evaluated based on user feedback and a comparison of operational efficiency before and after system implementation.

Overall, the hotel management database system provides an essential tool for hotels looking to improve their operations and provide a better guest experience. It streamlines operations and allows for more efficient management, resulting in better customer satisfaction and higher revenues. This paper provides a comprehensive guide for hotels interested in implementing a similar system, with insights into the design and development process, as well as the benefits and limitations of the system.

Brief description of the project

The hotel management database is a software system that is designed to help hotels manage their operations more efficiently. It provides a centralized location for storing and managing all the information related to hotel operations, including guest reservations, room inventory, customer information, staff schedules, billing, and feedback.

The system is designed to be user-friendly and easy to navigate, with a range of features and tools that help hotel staff manage their daily tasks more efficiently. The database is designed to meet the needs of both hotel management and guests.

For hotel management, the system provides an easy-to-use interface that allows them to view and manage all aspects of the hotel's operations from a single location. This includes managing room inventory, handling reservations and cancellations, managing staff schedules and payroll, monitoring guest feedback, and generating reports on the hotel's performance. The system can also help with accounting and revenue management, by tracking room occupancy, pricing, and payments.

For guests, the hotel management database provides a range of features that help them to manage their stay more effectively. This includes the ability to make reservations online or over the phone, view room availability and rates, check in and out of the hotel, order room service, request housekeeping or maintenance services, and provide feedback on their stay. The system can also help with loyalty programs and guest profiling, by tracking guest preferences and history.

Overall, the hotel management database is an essential tool for any hotel looking to improve their operations and provide a better guest experience. It streamlines operations and allows for more efficient management, resulting in better customer satisfaction and higher revenues.

Main Goals

The main goals of the hotel management database project are to help hotels manage their operations more efficiently and provide a better guest experience. This is accomplished through the creation of a centralized database that stores and manages all information related to hotel operations. By streamlining operations and allowing for more efficient management, the project aims to improve customer satisfaction and increase revenue for hotels.

Business Description

The hotel is a four-star establishment located outside Tirana, Albania, on the height of Mount Dajti within the National Park of Dajti Mount. The hotel boasts a stunning natural setting with breathtaking views of Tirana and numerous local attractions. The hotel provides comfortable and luxurious accommodations for its guests, ranging from standard rooms to suites with amenities like air conditioning, free Wi-Fi, and room service. The hotel also offers a variety of dining options, including a restaurant with a terrace and panoramic views, a bar, and a café. Guests can also enjoy a range of services such as a spa, fitness center, swimming pool, and outdoor activities like hiking, horseback riding, and cycling.

The hotel management team is dedicated to ensuring that every guest has a memorable and enjoyable experience during their stay. The team oversees the day-to-day operations of the hotel, including managing reservations, supervising staff, maintaining facilities, and providing excellent customer service. They are committed to providing guests with the highest quality of service and ensuring that their needs are met in a timely and efficient manner.

Organization/Department Scope:

The scope of this project is within the Hotel Operations department, specifically the areas of customer reservations, room management, payment processing, service requests, staff management, feedback management, and reporting. This department is responsible for the day-to-day operations of the hotel and ensuring that customers have an enjoyable stay. The proposed database will support and enhance these operations by providing a centralized system for storing and managing information related to these areas. This will enable staff to efficiently manage customer reservations, monitor room availability and occupancy, process payments, respond to service requests, and manage staff schedules. The database will also provide valuable insights to hotel management by generating various reports on hotel performance, allowing them to make data-driven decisions to improve services and increase revenue.

Project Scope

The project is expected to be completed within 1(one) month and is focused on creating a database system that will support the operations of the hotel. The database will be implemented within the hotel's IT infrastructure, and all relevant data will be migrated to the new system.

The database will support the entire hotel, including all of its rooms, staff members, and customers. The hotel management team will use the database to streamline operations, manage customer reservations, and process payments. The front desk staff will use the system to check-in and check-out customers and assign rooms. Housekeeping staff will use the database to keep track of room status and service requests.

Processes/Operations Supported by the Proposed Database:

- Customer reservations: The database will store customer information and track their reservations, including check-in and check-out dates, room types, and rates.
- Room management: The database will store information about each room, including room type, rate, and status (vacant, occupied, or under maintenance).
- Payment processing: The database will store payment information and process payments for reservations and other hotel services.
- Service requests: The database will store information about customer service requests, including the type of service, date of request, and associated costs.
- Staff management: The database will store staff information, including contact details, positions, and work schedules.
- Feedback management: The database will store customer feedback, including ratings, comments, and suggestions, and use it to improve the hotel's services.
- Reporting: The database will generate various reports to assist hotel management in monitoring hotel performance, such as revenue generated, room occupancy rates, and service requests.

List of information requirements

List of all customers along with their contact information (name, address, phone number, email)

List of all reservations with their corresponding check-in and check-out dates and the associated customer information

Total revenue generated by the hotel within a specified time period

List of all rooms along with their corresponding room numbers, types, and rates

List of all service requests along with the service type, date, cost, and the associated customer information

List of all staff members along with their contact information and positions

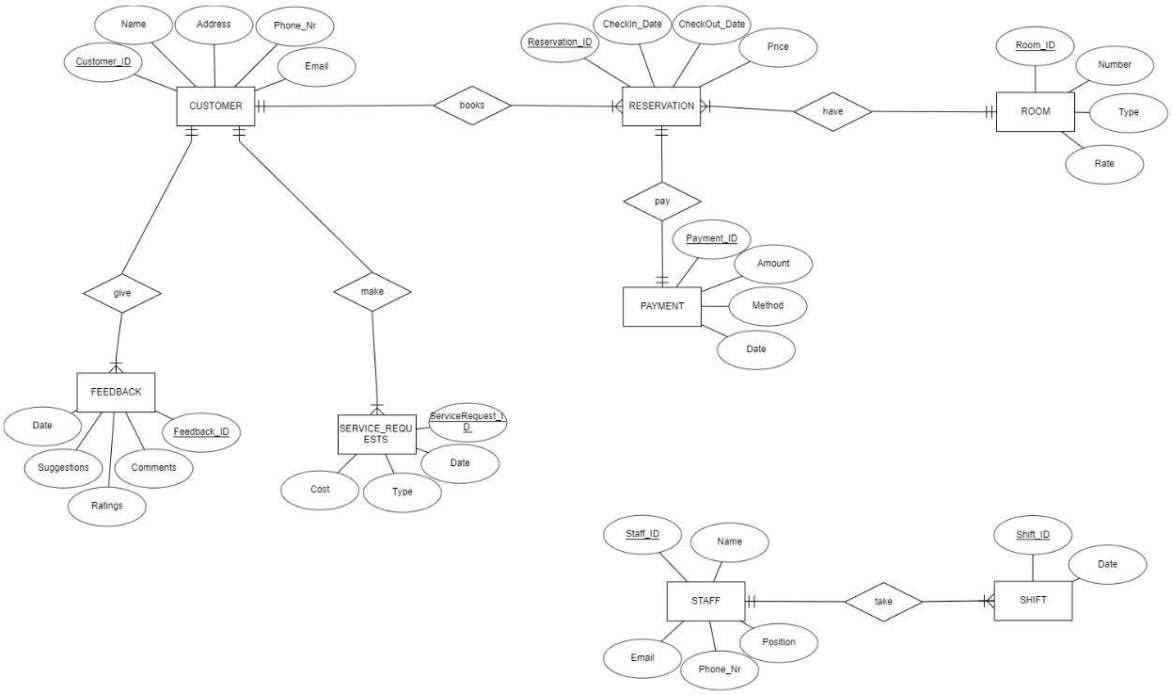
Average rating of the hotel based on all customer feedback

List of all reservations made by a specific customer, along with the associated room information, payment details, and feedback (if available)

List of all shifts worked by a specific staff member along with the date and the associated staff and room information

Total revenue generated by the hotel from a specific room type within a specified time period.

Entity Relationship Diagram



The Customer entity represents a customer who **can make zero or many reservations.** The entity has attributes like `CustomerID`(Primary Key), `Name`, `Address`, `PhoneNumber`,` Email`, and `ReservationID ` (Foreign key referencing Reservation table).

The Reservation entity represents a reservation made by a customer. The entity has attributes like `ReservationID`(PK), `CustomerID`, `RoomID`, `CheckInDate`,` CheckOutDate`, `Price`, `PaymentID`, and `FeedbackID`. **A reservation can have one room, one payment, and one feedback. A customer can make zero or many reservations.**

The Room entity represents a room in the hotel. The entity has attributes like `RoomID`(PK), `RoomNumber`, `RoomType`, `RoomRate`, and `ReservationID`. **A room can be part of zero or many reservations.**

The Staff entity represents the hotel staff. The entity has attributes like `StaffID`, `Name`, `Position`, `PhoneNumber`, and `Email`. **A staff member can work zero or many shifts.**

The Service Requests entity represents a request for a service by a customer. The entity has attributes like `ServiceRequestID`, `RequestDate`, `ServiceType`,` ServiceCost`, and `CustomerID`. **A customer can make zero or many service requests.**

The Payment entity represents a payment made for a reservation. The entity has attributes like `PaymentID`, `ReservationID`, `PaymentAmount`, `PaymentMethod`, and `PaymentDate`. **A payment is made for one reservation.**

The Feedback entity represents feedback provided by a customer for a reservation. The entity has attributes like `FeedbackID`, `ReservationID`, `CustomerID`, `Comments`, `Ratings`, `Suggestions`, and `FeedbackDate`. **Feedback is made for one reservation.**

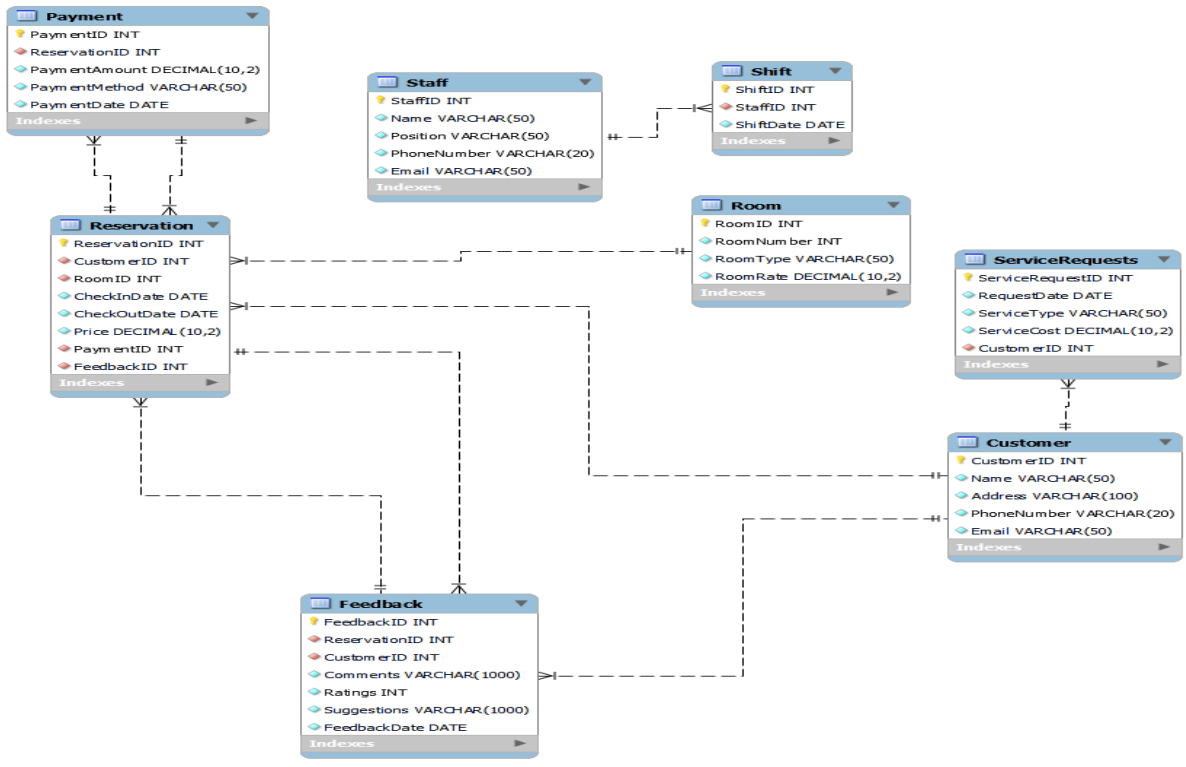
The Shift entity represents a shift worked by a staff member. The entity has attributes like `ShiftID`, `StaffID`, and `ShiftDate`. **A staff member can work zero or many shifts.**

Relationships Example:

A Customer can make zero or many reservations (1:M). This relationship is mandatory for the Reservation entity, as a Reservation cannot exist without a corresponding Customer.

A Reservation is made by one Customer (1:1). This relationship is mandatory for the Reservation entity, as every reservation must have a corresponding customer.

We started with designing the structure of the database. We drew the ERD diagram on ERDPlus, noting down all the tables required. We designed each table with columns and attributes respectively and trying to make an idea about the relationships between tables. We tried to keep the tables in a form where we can reduce the data redundancy and tried to make it accessible in an easy and simple way. The Relational Schema diagram created is as below (generated with MySQL Workbench):



Database Implementation:

All SQLs are available on the 'table_insert.sql' file

Creating the database and tables:

```
CREATE database hotelsystem1;
Use hotelsystem1;

CREATE TABLE Customer (
    CustomerID INT PRIMARY KEY,
    Name VARCHAR(50) NOT NULL,
    Address VARCHAR(100) NOT NULL,
    PhoneNumber VARCHAR(20) NOT NULL,
    Email VARCHAR(50) NOT NULL
);

CREATE TABLE Reservation (
    ReservationID INT PRIMARY KEY,
    CustomerID INT NOT NULL,
    RoomID INT NOT NULL,
    CheckInDate DATE NOT NULL,
    CheckOutDate DATE NOT NULL,
    Price DECIMAL(10,2) NOT NULL,
    PaymentID INT NOT NULL,
    FeedbackID INT NOT NULL
);

CREATE TABLE Room (
    RoomID INT PRIMARY KEY,
    RoomNumber INT NOT NULL,
    RoomType VARCHAR(50) NOT NULL,
    RoomRate DECIMAL(10,2) NOT NULL
);

CREATE TABLE Staff (
    StaffID INT PRIMARY KEY,
    Name VARCHAR(50) NOT NULL,
    Position VARCHAR(50) NOT NULL,
    PhoneNumber VARCHAR(20) NOT NULL,
    Email VARCHAR(50) NOT NULL
);

CREATE TABLE ServiceRequests (
    ServiceRequestID INT PRIMARY KEY,
    RequestDate DATE NOT NULL,
    ServiceType VARCHAR(50) NOT NULL,
    ServiceCost DECIMAL(10,2) NOT NULL,
    CustomerID INT NOT NULL
);

CREATE TABLE Payment (
    PaymentID INT PRIMARY KEY,
    ReservationID INT NOT NULL,
    PaymentAmount DECIMAL(10,2) NOT NULL,
    PaymentMethod VARCHAR(50) NOT NULL,
    PaymentDate DATE NOT NULL
);
```

```sql
CREATE TABLE Feedback (
    FeedbackID INT PRIMARY KEY,
    ReservationID INT NOT NULL,
    CustomerID INT NOT NULL,
    Comments VARCHAR(1000) NOT NULL,
    Ratings INT NOT NULL,
    Suggestions VARCHAR(1000) NOT NULL,
    FeedbackDate DATE NOT NULL
);

CREATE TABLE Shift (
    ShiftID INT PRIMARY KEY,
    StaffID INT NOT NULL,
    ShiftDate DATE NOT NULL
);
ALTER TABLE Reservation
ADD CONSTRAINT FK_Customer_Reservation
FOREIGN KEY (CustomerID)
REFERENCES Customer(CustomerID);

ALTER TABLE Reservation
ADD CONSTRAINT FK_Room_Reservation
FOREIGN KEY (RoomID)
REFERENCES Room(RoomID);

ALTER TABLE Reservation
ADD CONSTRAINT FK_Payment_Reservation
FOREIGN KEY (PaymentID)
REFERENCES Payment(PaymentID);

ALTER TABLE Reservation
ADD CONSTRAINT FK_Feedback_Reservation
FOREIGN KEY (FeedbackID)
REFERENCES Feedback(FeedbackID);

ALTER TABLE ServiceRequests
ADD CONSTRAINT FK_Customer_ServiceRequests
FOREIGN KEY (CustomerID)
REFERENCES Customer(CustomerID);

ALTER TABLE Payment
ADD CONSTRAINT FK_Reservation_Payment
FOREIGN KEY (ReservationID)
REFERENCES Reservation(ReservationID);

ALTER TABLE Feedback
ADD CONSTRAINT FK_Reservation_Feedback
FOREIGN KEY (ReservationID)
REFERENCES Reservation(ReservationID);

ALTER TABLE Feedback
ADD CONSTRAINT FK_Customer_Feedback
FOREIGN KEY (CustomerID)
REFERENCES Customer(CustomerID);

ALTER TABLE Shift
ADD CONSTRAINT FK_Staff_Shift
FOREIGN KEY (StaffID)
REFERENCES Staff(StaffID);
```

INSERT INTO values

```
-- Populating Customer table
INSERT INTO Customer (CustomerID, Name, Address, PhoneNumber, Email)
VALUES (1, 'John Doe', '123 Main St, Anytown, USA', '+1-555-555-1234',
'johndoe@email.com'),
       (2, 'Jane Smith', '456 Park Ave, Anytown, USA', '+1-555-555-5678',
'janesmith@email.com'),
       (3, 'Bob Johnson', '789 Elm St, Anytown, USA', '+1-555-555-9012',
'bobjohnson@email.com');

-- Populating Room table
INSERT INTO Room (RoomID, RoomNumber, RoomType, RoomRate)
VALUES (1, 101, 'Standard', 100.00),
       (2, 102, 'Standard', 100.00),
       (3, 201, 'Deluxe', 150.00),
       (4, 202, 'Deluxe', 150.00),
       (5, 301, 'Suite', 200.00),
       (6, 302, 'Suite', 200.00);

-- Populating Staff table
INSERT INTO Staff (StaffID, Name, Position, PhoneNumber, Email)
VALUES (1, 'Sarah Johnson', 'Manager', '+1-555-555-1111',
'sarahjohnson@email.com'),
       (2, 'Bob Smith', 'Receptionist', '+1-555-555-2222',
'bobsmith@email.com');

-- Populating ServiceRequests table
INSERT INTO ServiceRequests (ServiceRequestID, RequestDate, ServiceType,
ServiceCost, CustomerID)
VALUES (1, '2023-02-15', 'Room cleaning', 50.00, 1),
       (2, '2023-02-16', 'Laundry', 25.00, 2),
       (3, '2023-02-17', 'Room service', 30.00, 3);


/* We need to disable the foreign key contstrain
for the current session, otherwise it will give us
 error 1452  when we run the rest of code */
SET FOREIGN_KEY_CHECKS=0;

-- Populating Reservation table
INSERT INTO Reservation (ReservationID, CustomerID, RoomID, CheckInDate,
CheckOutDate, Price, PaymentID, FeedbackID)
VALUES
(1, 1, 1, '2023-02-20', '2023-02-25', 500.00, 1, 1),
(2, 2, 2, '2023-03-10', '2023-03-12', 200.00, 2, 2),
(3, 3, 3, '2023-04-05', '2023-04-09', 800.00, 3, 3),
(4, 4, 4, '2023-05-01', '2023-05-05', 700.00, 4, 4),
(5, 5, 5, '2023-06-15', '2023-06-20', 550.00, 5, 5),
(6, 6, 6, '2023-07-02', '2023-07-07', 600.00, 6, 6);


-- Populating Payment table
INSERT INTO Payment (PaymentID, ReservationID, PaymentAmount,
PaymentMethod, PaymentDate)
VALUES (1, 0, 100.00, 'Credit card', '2023-02-15'),
       (2, 0, 150.00, 'Debit card', '2023-02-16'),
       (3, 0, 200.00, 'Cash', '2023-02-17');
```

```sql
-- Populating Feedback table
INSERT INTO Feedback (FeedbackID, ReservationID, CustomerID, Comments,
Ratings, Suggestions, FeedbackDate)
VALUES (1, 1, 1, 'Great experience overall!', 5, 'None', '2023-02-15'),
       (2, 2, 2, 'Could be better', 3, 'Improve room cleaning', '2023-02-
16'),
       (3, 3, 3, 'Excellent service!', 5, 'None', '2023-02-17');


-- Populating Shift table
INSERT INTO Shift (ShiftID, StaffID, ShiftDate)
VALUES (1, 1, '2022-01-01'),
(2, 2, '2022-01-02'),
(3, 3, '2022-01-03');
```

Managerial Queries stated earlier:

```sql
-- # List of all customers along with their contact information (name,
address, phone number, email)
SELECT *
FROM Customer;

-- # List of all reservations with their corresponding check-in and check-
out dates and the
-- # associated customer information
SELECT Reservation.ReservationID, Customer.Name, Customer.PhoneNumber,
Customer.Email, Reservation.CheckInDate, Reservation.CheckOutDate
FROM Reservation
INNER JOIN Customer ON Reservation.CustomerID = Customer.CustomerID;

-- # Total revenue generated by the hotel within a specified time period
SELECT SUM(PaymentAmount) AS TotalRevenue
FROM Payment
INNER JOIN Reservation ON Payment.ReservationID = Reservation.ReservationID
WHERE PaymentDate BETWEEN 'start_date' AND 'end_date';
-- Note: Replace 'start_date' and 'end_date' with the desired time period.

-- # List of all rooms along with their corresponding room numbers, types,
-- and rates
SELECT *
FROM Room;

-- # List of all service requests along with the service type, date, cost,
-- # and the associated customer information
SELECT ServiceRequests.ServiceRequestID, Customer.Name,
Customer.PhoneNumber, Customer.Email, ServiceRequests.RequestDate,
ServiceRequests.ServiceType, ServiceRequests.ServiceCost
FROM ServiceRequests
INNER JOIN Customer ON ServiceRequests.CustomerID = Customer.CustomerID;

-- # List of all staff members along with their contact information and ---
-- positions
SELECT *
FROM Staff;

-- # Average rating of the hotel based on all customer feedback
SELECT AVG(Ratings) AS AvgRating
FROM Feedback;
```

```sql
-- # List of all reservations made by a specific customer, along with the
associated room information,
-- # payment details, and feedback (if available)
SELECT Reservation.ReservationID, Room.RoomNumber, Room.RoomType,
Room.RoomRate, Reservation.CheckInDate, Reservation.CheckOutDate,
Payment.PaymentMethod, Payment.PaymentAmount, Feedback.Comments,
Feedback.Ratings, Feedback.Suggestions
FROM Reservation
INNER JOIN Room ON Reservation.RoomID = Room.RoomID
LEFT JOIN Payment ON Reservation.PaymentID = Payment.PaymentID
LEFT JOIN Feedback ON Reservation.FeedbackID = Feedback.FeedbackID
WHERE Reservation.CustomerID = 'customer_id';
-- Note: Replace 'customer_id' with the desired customer ID.

-- # List of all shifts worked by a specific staff member along with the --
-- date
-- # and the associated staff and room information
SELECT Shift.ShiftID, Staff.Name, Staff.Position, Room.RoomNumber,
Room.RoomType, Room.RoomRate, Shift.ShiftDate
FROM Shift
INNER JOIN Staff ON Shift.StaffID = Staff.StaffID
INNER JOIN Reservation ON Shift.ShiftDate BETWEEN Reservation.CheckInDate
AND Reservation.CheckOutDate
INNER JOIN Room ON Reservation.RoomID = Room.RoomID
WHERE Staff.StaffID = 'staff_id';
-- Note: Replace 'staff_id' with the desired staff ID.

-- #Total revenue generated by the hotel from a specific room type within a
-- specified time period
SELECT SUM(PaymentAmount) AS TotalRevenue
FROM Payment
INNER JOIN Reservation ON Payment.ReservationID = Reservation.ReservationID
INNER JOIN Room ON Reservation.RoomID = Room.RoomID
WHERE Room.RoomType = 'room_type' AND PaymentDate BETWEEN 'start_date' AND
'end_date';
-- Note: Replace 'room_type', 'start_date', and 'end_date' with the desired
-- room type and time period.a
```

Data Warehouse Implementation

A data warehouse is a large, central repository of data that is designed to support business intelligence and data analysis. It typically contains a copy of data from various sources across the organization and is organized in a way that is optimized for querying and analysis. In the case of the hotel management system, we can create a data warehouse by consolidating all the data from the various tables into a single location.

To create the data warehouse, we will use the following steps:

Create a new database to store the data warehouse.

Create the fact table and dimension tables that will be used to organize the data.

Load the data into the data warehouse using ETL (Extract, Transform, Load) processes.

Create indexes and other optimizations to speed up querying and analysis.

All SQLs are available on the 'datawarehouse.sql' file

In conclusion, the hotel management database is an essential tool for hotels looking to improve their operations and provide a better guest experience. It allows for centralized storage and management of all hotel-related information, including guest reservations, room inventory, customer information, staff schedules, billing, and feedback. The system is designed to be user-friendly and easy to navigate, with features that help hotel staff manage their daily tasks more efficiently. For guests, the database provides a range of features that help them to manage their stay more effectively, including online reservations, room availability and rates, check-in and check-out, room service, and feedback. The system can also help with loyalty programs and guest profiling, resulting in better customer satisfaction and higher revenues for the hotel.

Implementing a hotel management database requires a data warehouse design, which involves creating a star schema with fact and dimension tables. The ETL process is also essential to transfer data from the operational systems into the data warehouse, where it can be used for analysis. Effective data analysis requires an understanding of the needs of the hotel and its customers, and the creation of SQL statements and queries to extract meaningful insights from the data.

In summary, the hotel management database is a crucial system that can help hotels manage their operations more efficiently and provide a better guest experience. By using the data warehouse design, ETL process, and data analysis, hotels can leverage the power of their data to improve operations, increase customer satisfaction, and drive revenue growth.