

DATABASE MANAGEMENT SYSTEM - CSA0593

ASSIGNMENT 4

K.SAI YASWANTH

192372374

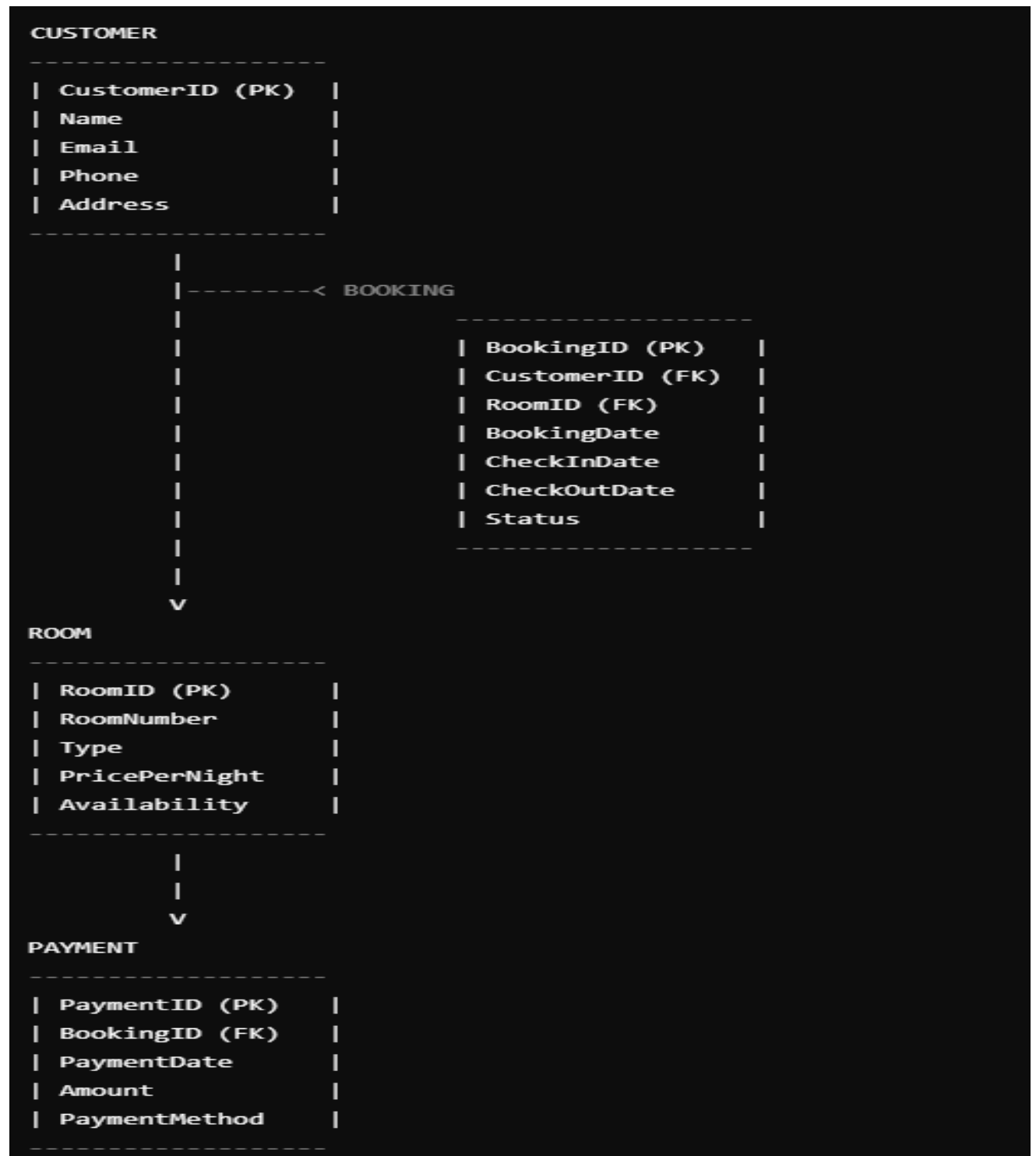
QUESTION:

Design a database schema and write SQL code for managing customers, rooms, bookings, and payments

- Model tables for customers, rooms, bookings, and payments.
- Write stored procedures to book rooms and process customer payments.
- Implement triggers to manage room availability and update booking status.
- Write SQL queries to analyze room occupancy and total revenue from bookings.

ANSWER:

CONCEPTUAL E.R.DIAGRAM:



LOGICAL E.R.DIAGRAM:

```
CUSTOMER
-----
| CustomerID (PK) | -----< BOOKING
| Name           | | -----
| Email          | | BookingID (PK)   |
| Phone          | | CustomerID (FK)  |
| Address        | | RoomID (FK)      |
-----| | BookingDate       |
         | | CheckInDate        |
         | | CheckOutDate     |
         | | Status         |
         | | -----
         |
         v

ROOM
-----
| RoomID (PK)    | -----< BOOKING
| RoomNumber     | | -----
| Type           | | BookingID (FK)   |
| PricePerNight  | | -----
| Availability    |
-----
         |
         |
         v

PAYMENT
-----
| PaymentID (PK) |
| BookingID (FK) |
| PaymentDate    |
| Amount         |
| PaymentMethod  |
```

PHYSICAL E.R.DIAGRAM:

CUSTOMER

```
-----  
| CustomerID (PK)  INT          |  
| Name            VARCHAR(100) NOT NULL |  
| Email           VARCHAR(150) NOT NULL |  
| Phone           VARCHAR(15)          |  
| Address         TEXT                  |  
-----
```

```
      |  
      |-----< BOOKING
```

```
      |  
      |-----  
      | BookingID (PK)  INT          |  
      | CustomerID (FK) INT          |  
      | RoomID (FK)    INT          |  
      | BookingDate    DATE          |  
      | CheckInDate    DATE          |  
      | CheckOutDate   DATE          |  
      | Status          VARCHAR(20) NOT NULL |  
      |-----
```

```
      |  
      V
```

ROOM

```
-----  
| RoomID (PK)      INT          |  
| RoomNumber       VARCHAR(10) NOT NULL |  
| Type             VARCHAR(50)          |  
| PricePerNight    DECIMAL(10,2) NOT NULL |  
| Availability     BOOLEAN              |  
-----
```

```
      |  
      |  
      V
```

PAYMENT

```
-----  
| PaymentID (PK)   INT          |  
| BookingID (FK)   INT          |  
| PaymentDate      DATE          |  
| Amount           DECIMAL(10,2) NOT NULL |  
| PaymentMethod    VARCHAR(50)          |  
-----
```

MYSQL STATEMENTS:

mysql

```
CREATE DATABASE HotelManagement;
```

```
USE HotelManagement;
```

```
CREATE TABLE Customers (  
    CustomerID INT AUTO_INCREMENT PRIMARY KEY,  
    FirstName VARCHAR(50),  
    LastName VARCHAR(50),  
    Email VARCHAR(100),  
    Phone VARCHAR(20)  
);
```

```
CREATE TABLE Rooms (  
    RoomID INT AUTO_INCREMENT PRIMARY KEY,  
    RoomNumber INT,  
    RoomType VARCHAR(50),  
    Rate DECIMAL(10, 2),  
    Status VARCHAR(20)  
);
```

```
CREATE TABLE Bookings (  
    BookingID INT AUTO_INCREMENT PRIMARY KEY,  
    CustomerID INT,  
    RoomID INT,  
    ArrivalDate DATE,  
    DepartureDate DATE,  
    BookingStatus VARCHAR(20),
```

```
FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID),  
FOREIGN KEY (RoomID) REFERENCES Rooms(RoomID)  
);
```

```
CREATE TABLE Payments (  
    PaymentID INT AUTO_INCREMENT PRIMARY KEY,  
    BookingID INT,  
    PaymentDate DATE,  
    PaymentAmount DECIMAL(10, 2),  
    PaymentMethod VARCHAR(50),  
    FOREIGN KEY (BookingID) REFERENCES Bookings(BookingID)  
);
```

Stored Procedures:

mysql

DELIMITER //

```
CREATE PROCEDURE sp_BookRoom(  
    IN customerID INT,  
    IN roomID INT,  
    IN arrivalDate DATE,  
    IN departureDate DATE  
)  
BEGIN  
    INSERT INTO Bookings (CustomerID, RoomID, ArrivalDate, DepartureDate, BookingStatus)  
    VALUES (customerID, roomID, arrivalDate, departureDate, 'Booked');
```

```
UPDATE Rooms
SET Status = 'Occupied'
WHERE RoomID = roomID;
END //
```

```
CREATE PROCEDURE sp_ProcessPayment(
    IN bookingID INT,
    IN paymentDate DATE,
    IN paymentAmount DECIMAL(10, 2),
    IN paymentMethod VARCHAR(50)
)
BEGIN
    INSERT INTO Payments (BookingID, PaymentDate, PaymentAmount, PaymentMethod)
    VALUES (bookingID, paymentDate, paymentAmount, paymentMethod);

    UPDATE Bookings
    SET BookingStatus = 'Paid'
    WHERE BookingID = bookingID;
END //

DELIMITER ;
```

Triggers:

```
mysql
DELIMITER //
```

```
CREATE TRIGGER tr_UpdateRoomAvailability
AFTER INSERT ON Bookings
FOR EACH ROW
BEGIN
    UPDATE Rooms
    SET Status = 'Occupied'
    WHERE RoomID = NEW.RoomID;
END //
```

```
CREATE TRIGGER tr_UpdateBookingStatus
AFTER UPDATE ON Payments
FOR EACH ROW
BEGIN
    UPDATE Bookings
    SET BookingStatus = 'Paid'
    WHERE BookingID = NEW.BookingID;
END //
```

```
DELIMITER;
```

SQL Queries for Analysis:

```
mysql
-- Room Occupancy
SELECT
    RoomNumber,
```



```
RoomType,  
COUNT(*) AS TotalBookings,  
SUM(CASE WHEN BookingStatus = 'Paid' THEN 1 ELSE 0 END) AS PaidBookings  
FROM  
Rooms  
JOIN Bookings ON Rooms.RoomID = Bookings.RoomID  
GROUP BY  
RoomNumber, RoomType;
```

-- Total Revenue from Bookings

```
SELECT  
SUM(PaymentAmount) AS TotalRevenue  
FROM  
Payments;
```

-- Customer Booking History

```
SELECT  
CustomerID,  
FirstName,  
LastName,  
COUNT(*) AS TotalBookings  
FROM  
Customers  
JOIN Bookings ON Customers.CustomerID = Bookings.CustomerID  
GROUP BY  
CustomerID, FirstName, LastName;
```

Conclusion:

This database design provides a comprehensive foundation for managing customers, rooms, bookings, and payments. The stored procedures simplify room booking and payment processing, while the triggers ensure data consistency and accuracy. The SQL queries enable analysis of room occupancy, total revenue, and customer booking history.