DATABASE MANAGEMENT SYSTEM - CSA0593

ASSIGNMENT 2

T.ARUN KUMAR

192372325

QUESTION:

Develop a database for managing parking spaces, vehicles, and payments.

- Model tables for parking spaces, vehicles, and payments.

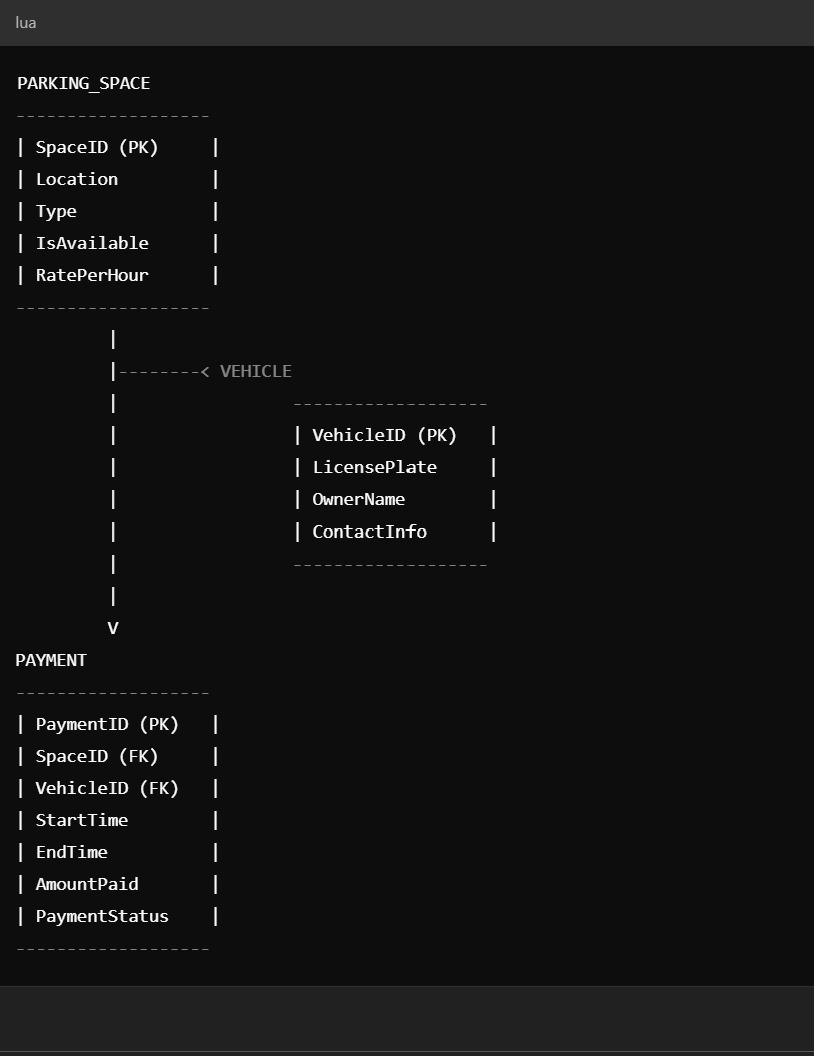
- Write stored procedures for reserving and paying for parking.

- Implement triggers to update parking space availability and payment status.

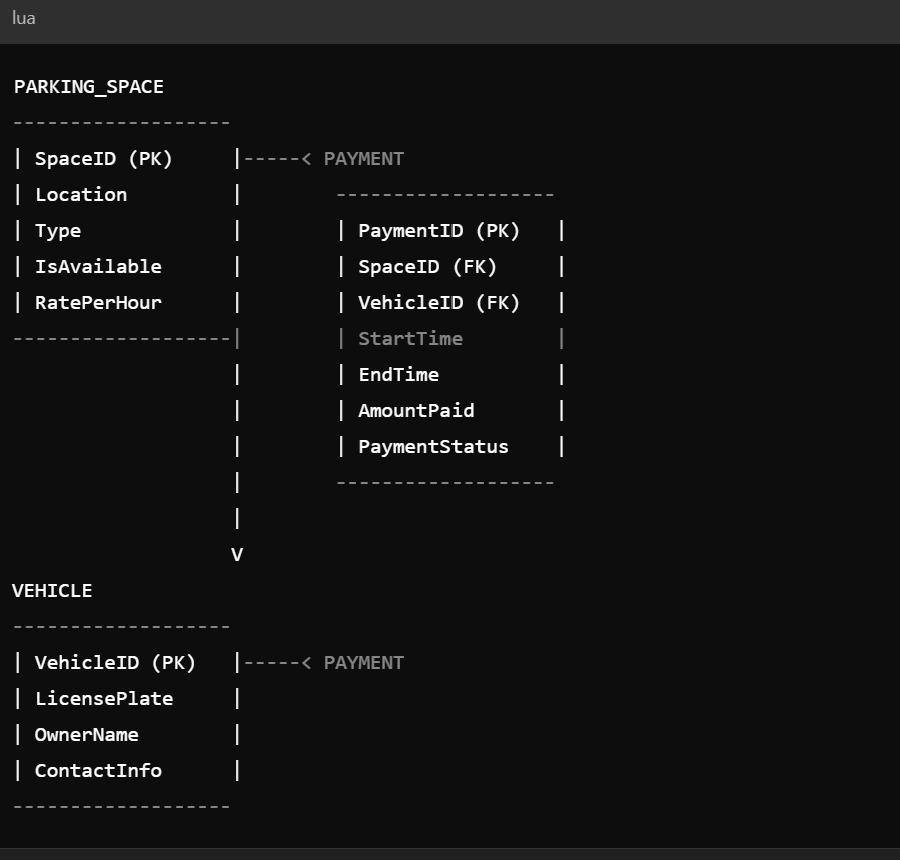
- Write SQL queries to analyze parking occupancy and revenue

ANSWER:

CONCEPTUAL MODEL[E.R DIAGRAM]:



LOGICAL MODEL[ E.R.DIAGRAM]:



PHYSICAL MODEL[E.R.DIAGRAM]:



SQL STATEMENTS :

Here are the SQL statements and conclusion for the topic:

SQL Statements:

CREATE DATABASE ParkingManagement;

USE ParkingManagement;

CREATE TABLE ParkingSpaces (

SpaceID INT AUTO\_INCREMENT PRIMARY KEY,

SpaceNumber INT,

SpaceType VARCHAR(50),

Availability VARCHAR(20)

);

CREATE TABLE Vehicles (

VehicleID INT AUTO\_INCREMENT PRIMARY KEY,

LicensePlate VARCHAR(20),

VehicleType VARCHAR(50)

);

CREATE TABLE Payments (

PaymentID INT AUTO\_INCREMENT PRIMARY KEY,

VehicleID INT,

SpaceID INT,

PaymentDate DATE,

PaymentAmount DECIMAL(10, 2),

PaymentStatus VARCHAR(20),

FOREIGN KEY (VehicleID) REFERENCES Vehicles(VehicleID),

FOREIGN KEY (SpaceID) REFERENCES ParkingSpaces(SpaceID)

);

CREATE TABLE Reservations (

ReservationID INT AUTO\_INCREMENT PRIMARY KEY,

VehicleID INT,

SpaceID INT,

ReservationDate DATE,

StartTime TIME,

EndTime TIME,

FOREIGN KEY (VehicleID) REFERENCES Vehicles(VehicleID),

FOREIGN KEY (SpaceID) REFERENCES ParkingSpaces(SpaceID)

);

Stored Procedures:

mysql

DELIMITER //

CREATE PROCEDURE sp\_ReserveParking(

IN vehicleID INT,

IN spaceID INT,

IN reservationDate DATE,

IN startTime TIME,

IN endTime TIME

)

BEGIN

INSERT INTO Reservations (VehicleID, SpaceID, ReservationDate, StartTime, EndTime)

VALUES (vehicleID, spaceID, reservationDate, startTime, endTime);

UPDATE ParkingSpaces

SET Availability = 'Reserved'

WHERE SpaceID = spaceID;

END //

CREATE PROCEDURE sp\_MakePayment(

IN paymentID INT,

IN vehicleID INT,

IN spaceID INT,

IN paymentDate DATE,

IN paymentAmount DECIMAL(10, 2)

)

BEGIN

INSERT INTO Payments (PaymentID, VehicleID, SpaceID, PaymentDate, PaymentAmount, PaymentStatus)

VALUES (paymentID, vehicleID, spaceID, paymentDate, paymentAmount, 'Paid');

UPDATE ParkingSpaces

SET Availability = 'Available'

WHERE SpaceID = spaceID;

UPDATE Reservations

SET EndTime = NULL

WHERE VehicleID = vehicleID AND SpaceID = spaceID;

END //

DELIMITER;

Triggers:

mysql

DELIMITER //

CREATE TRIGGER tr\_UpdateAvailability

AFTER INSERT ON Reservations

FOR EACH ROW

BEGIN

UPDATE ParkingSpaces

SET Availability = 'Reserved'

WHERE SpaceID = NEW.SpaceID;

END //

CREATE TRIGGER tr\_UpdatePaymentStatus

AFTER UPDATE ON Payments

FOR EACH ROW

BEGIN

UPDATE Reservations

SET EndTime = NULL

WHERE VehicleID = NEW.VehicleID AND SpaceID = NEW.SpaceID;

END //

DELIMITER;

SQL Queries:

mysql

-- Parking Occupancy

SELECT

SpaceNumber,

SpaceType,

Availability,

COUNT(\*) AS TotalReservations

FROM

ParkingSpaces

JOIN Reservations ON ParkingSpaces.SpaceID = Reservations.SpaceID

GROUP BY

SpaceNumber, SpaceType, Availability;

-- Revenue Analysis

SELECT

MONTH(PaymentDate) AS Month,

SUM(PaymentAmount) AS TotalRevenue

FROM

Payments

GROUP BY

MONTH(PaymentDate);

-- Vehicle Parking History

SELECT

VehicleID,

LicensePlate,

COUNT(\*) AS TotalReservations

FROM

Vehicles

JOIN Reservations ON Vehicles.VehicleID = Reservations.VehicleID

GROUP BY

VehicleID, LicensePlate;

Conclusion:

This database design provides a comprehensive foundation for managing parking spaces, vehicles, and payments. The stored procedures simplify parking reservations and payment processing, while the triggers ensure data consistency and accuracy. The SQL queries enable analysis of parking occupancy, revenue, and vehicle parking history. Best Practices:

1. Regularly backup the database.

2. Use secure passwords and authentication.

3. Implement data validation and error handling.

4. Optimize queries for performance.

5. Use indexing for efficient data retrieval.

Future Enhancements:

1. Integrate with parking management software.

2. Implement automated email notifications.

3. Develop a web-based interface for parking reservations and payments.

4. Add support for multiple payment gateways.

5. Integrate with access control systems.