DBMS MINI PROJECT

TOPIC : Vehicle Parking Management System

NAME : VANSHIKA SINGH

Cno: UCE2023665

BATCH-C3

SRS REPORT:

1.INTRODUCTION

Purpose:

The purpose of the Parking Management System is to streamline parking operations by automating slot allocation and tracking. It simplifies customer registration, vehicle tracking, and fee calculations. The system promotes efficient space utilization and offers membership-based benefits to enhance customer satisfaction. It provides better management and scalability for different parking setups. This ensures a modernized and user-friendly parking experience for both customers and operators.

Scope:

The Parking Management System will allow for:

- Manage customers and their memberships.
- Track parking lots, parking slots, and vehicle parking sessions.
- Provide dynamic pricing and discounts for members.
- Automate slot allocation using triggers and stored procedures.
- Offer a user-friendly interface for customer management, vehicle tracking, and fee calculations.

Definitions:

- **Customer**: A person who registers to use the parking services.
- **Vehicle**: A registered car linked to a customer.
- **Parking Slot**: A spot in a parking lot, either vacant or occupied.
- **Parking Session**: A record of a vehicle's parking time, including entry and exit times.
- **Membership Plan**: Subscription plans offering benefits like discounts for parking.

2. SYSTEM OVERVIEW

The Parking Management System is composed of:

1. **Membership Plans**: Define customer benefits such as discounts and additional perks.
2. **Customers**: Tracks customer details, membership status, and registration.
3. **Vehicles**: Links customer vehicles to parking lots and logs entry time.

4. **Parking Lots and Slots**: Manages parking spaces dynamically.
5. **Parking Sessions**: Logs parking activities and calculates fees.

3. **Functional Requirements**

3.1 **Customer Management**

- Add new customers to the system.
- Customers can be classified as "member" or "non-member."
- The system maintains details such as name, phone number, email, and membership status.

3.2 **Vehicle Management**

- Register vehicles for customers.
- Each vehicle is associated with a license plate and a customer ID.

3.3 **Parking Lot Management**

- Track parking lots and their available spots.
- A parking lot has a location and a total number of spots.
- When a parking lot is added, the system automatically creates vacant parking slots.

3.4 **Parking Slot Management**

- Parking slots can either be vacant or occupied.
- Each parking slot is linked to a specific parking lot.

3.5 **Parking Session Management**

- Start a parking session for a vehicle when a parking slot is available.
- Track entry time and exit time for each session.
- End a parking session, update the exit time, and calculate the parking fee.

3.6 **Membership Plan Management**

- There are multiple membership plans: Basic, Silver, Gold, and Platinum.
- Each plan offers discounts on hourly parking rates.
- Membership status can affect parking fee calculation.

3.7 **Fee Calculation**

- Calculate parking fees based on the duration of parking and apply membership discounts.

4.**NON-FUNCTIONAL REQUIREMENTS:**

4.1 **Performance**

- The system should handle at least 100 concurrent parking sessions without significant

degradation in performance.

4.2 **Security**

- The system should ensure data integrity, especially for financial transactions related to parking fees.
- Sensitive information like customer details and payment history should be stored securely.

4.3 **Reliability**

- The system should ensure consistent operation without crashes or data loss.

4.4 **Usability**

- The user interface should be simple and intuitive for the users, allowing them to easily register customers, vehicles, and manage parking sessions.

5. External Interfaces

5.1 **Database Interface**
The system interacts with a MySQL database that contains tables such as:

- Customers
- Vehicles
- Parking Lots
- Parking Slots
- Membership Plans
- Parking Sessions

5.2 **User Interface**
A console-based application is used to interact with the system. Users will enter data via prompts, and the system will output relevant messages about parking status and session details.

6. System Architecture

The system architecture is built using:

- **Database**: MySQL for storing data.
- **Backend**: Java with JDBC for connecting to the MySQL database and handling user input/output.
- **Procedures/Triggers**: Stored procedures and triggers for calculating fees and managing slot availability.

7. Constraints

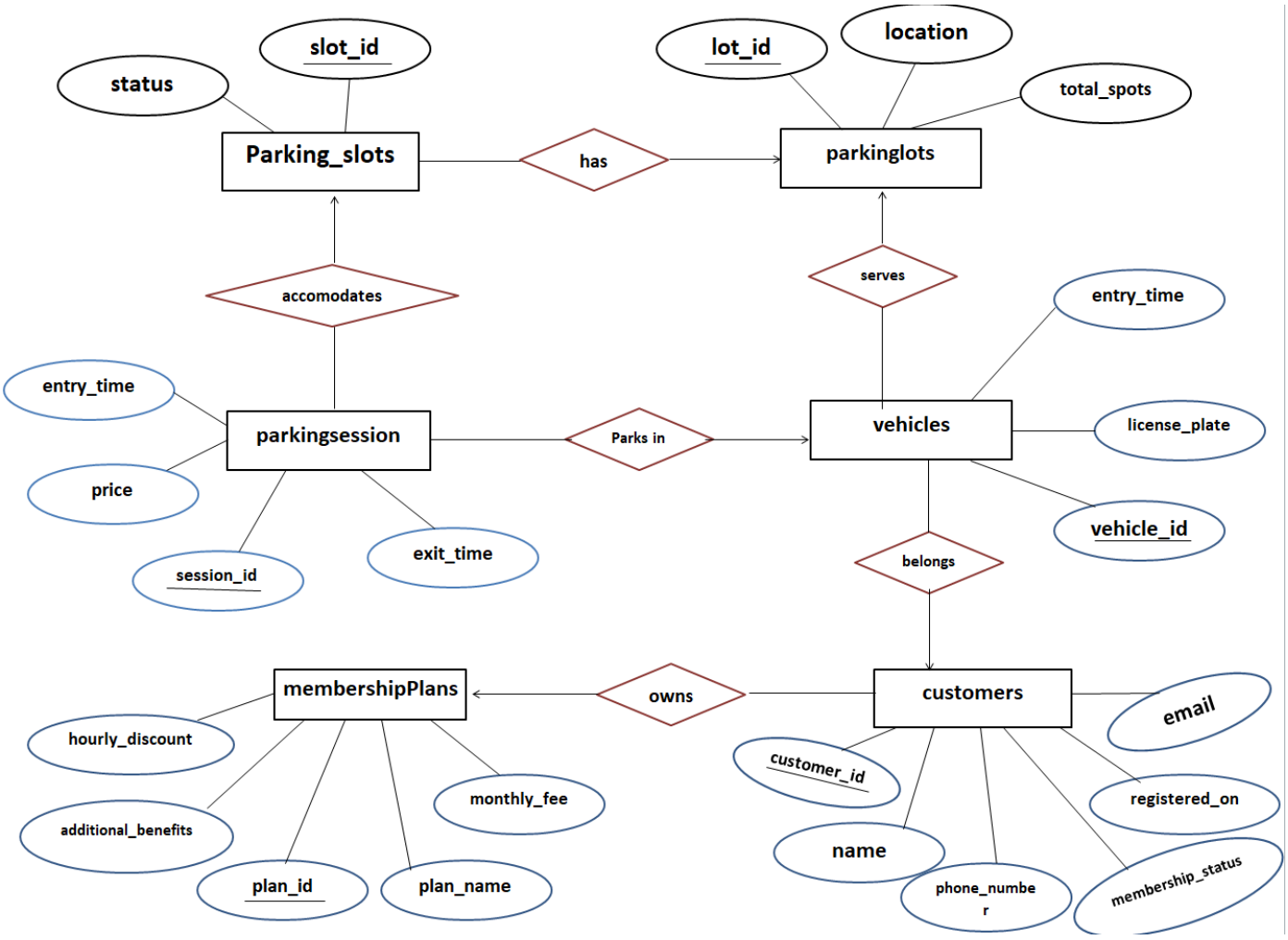
- The system will be built using Java, MySQL, and JDBC.
- It should be compatible with MySQL 8.0 or higher.

- The system must provide accurate billing and parking fee calculations based on the parking duration.

8. Use Case

- **Add a Customer**:
  The system will prompt for the customer's name, email, phone number, and membership type. After validating the input, the customer will be registered.add
- **Start a Parking Session**:
  A user selects a vehicle and a parking lot. The system checks for available slots in the selected lot and starts a session by marking the slot as occupied.
- **Calculate Parking Fee**:
  Once a session ends, the system calculates the fee by determining the duration and applying the applicable discount for the customer’s membership plan.

ER-Diagram:



Tables from ER-D:-

PARKINGLOTS:-

lot_id	location	total_spots

PARKING_SLOTS:-

lot_id	status	slot_id

VEHICLES:-

vehicle_id	lisence_plate	entry_time	customer_id	lot_id

PARKINGSESSIONS:-

session_id	vehicle_id	slot_id	exit_time	entry_time	price

MEMBERSHIPPLANS:-

plan_id	hourly_discount	additional_benefits	plan_name	monthly_fee

CUSTOMERS:-

customer_id	name	phone_number	membership_status	registered_on	membership_id	email

NORMALIZTION UPTO 3NF:

tables in er diagram are already in 3NF :

1. First Normal Form (1NF)

1NF requires that each column contains only atomic (indivisible) values, each record is unique, and each field contains a single type of data.

All tables in the system meet this requirement:

- **Each attribute holds a single value.**
- **Records are unique due to primary keys.**

2. Second Normal Form (2NF)

2. Second Normal Form (2NF)

2NF requires:

- **The table is in 1NF.**
- **All non-key attributes are fully dependent on the primary key (no partial dependencies).**

Each table meets 2NF:

- **Customers: All fields (name, email, etc.) depend on customer_id. The membership_id correctly links to MembershipPlans.**
- **MembershipPlans: All fields depend on plan_id, with no partial dependencies.**
- **ParkingLots: location and total_spots depend fully on lot_id.**
- **ParkingSlots: status depends on slot_id, and lot_id links to ParkingLots.**
- **Vehicles: All fields (license_plate, entry_time, lot_id) depend on vehicle_id.**
- **ParkingSessions: All attributes (vehicle_id, slot_id, exit_time, price) depend on session_id.**

3. Third Normal Form (3NF)

3NF requires:

- **The table is in 2NF.**
- **All non-key attributes are directly dependent on the primary key, with no transitive dependencies.**

Each table is in 3NF:

- **Customers: All non-key attributes depend directly on customer_id. membership_id is a foreign key.**
- **MembershipPlans: All attributes depend directly on plan_id, with no transitive dependencies.**
- **ParkingLots: location and total_spots depend directly on lot_id, no transitive dependencies.**
- **ParkingSlots: status depends on slot_id, and lot_id links to ParkingLots.**
- **Vehicles: license_plate, entry_time, and lot_id depend directly on vehicle_id.**
- **ParkingSessions: All attributes depend directly on session_id, no transitive dependencies.**

PROGRAM FLOW:

```
mysql> CREATE TABLE ParkingLots (
    ->     lot_id INT PRIMARY KEY,
    ->     location VARCHAR(100) NOT NULL,
    ->     total_spots INT NOT NULL
    -> );
Query OK, 0 rows affected (0.07 sec)

mysql> CREATE TABLE ParkingSlots (
    ->     slot_id INT PRIMARY KEY AUTO_INCREMENT,
    ->     lot_id INT NOT NULL,
    ->     status ENUM('vacant', 'occupied') DEFAULT 'vacant',
    ->     FOREIGN KEY (lot_id) REFERENCES ParkingLots(lot_id)
    -> );
Query OK, 0 rows affected (0.05 sec)

mysql> CREATE TABLE MembershipPlans (
    ->     plan_id INT PRIMARY KEY AUTO_INCREMENT,
    ->     plan_name VARCHAR(50) NOT NULL,
    ->     monthly_fee DECIMAL(10, 2) NOT NULL,
    ->     hourly_discount DECIMAL(5, 2) NOT NULL COMMENT 'Discount percentage on hourly rate',
    ->     additional_benefits TEXT
    -> );
Query OK, 0 rows affected (0.05 sec)

mysql> CREATE TABLE Customers (
    ->     customer_id INT PRIMARY KEY AUTO_INCREMENT,
    ->     name VARCHAR(100) NOT NULL,
    ->     phone_number VARCHAR(15) UNIQUE,
    ->     email VARCHAR(100) UNIQUE,
    ->     membership_status ENUM('member', 'non-member') DEFAULT 'non-member',
    ->     membership_id INT DEFAULT NULL,
    ->     registered_on TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    ->     FOREIGN KEY (membership_id) REFERENCES MembershipPlans(plan_id)
    -> );
Query OK, 0 rows affected (0.09 sec)
```

```
mysql> CREATE TABLE Vehicles (
    ->     vehicle_id INT PRIMARY KEY AUTO_INCREMENT,
    ->     license_plate VARCHAR(15) NOT NULL UNIQUE,
    ->     customer_id INT NOT NULL,
    ->     entry_time TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    ->     lot_id INT NOT NULL,
    ->     FOREIGN KEY (customer_id) REFERENCES Customers(customer_id),
    ->     FOREIGN KEY (lot_id) REFERENCES ParkingLots(lot_id)
    -> );
Query OK, 0 rows affected (0.05 sec)

mysql> CREATE TABLE ParkingSessions (
    ->     session_id INT PRIMARY KEY AUTO_INCREMENT,
    ->     vehicle_id INT NOT NULL,
    ->     slot_id INT NOT NULL,
    ->     entry_time TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    ->     exit_time TIMESTAMP NULL,
    ->     price DECIMAL(10, 2) DEFAULT NULL,
    ->     FOREIGN KEY (vehicle_id) REFERENCES Vehicles(vehicle_id),
    ->     FOREIGN KEY (slot_id) REFERENCES ParkingSlots(slot_id)
    -> );
Query OK, 0 rows affected (0.06 sec)

mysql> show tables;
+------------------+
| Tables_in_dbms_mp |
+------------------+
| customers         |
| membershipplans   |
| parkinglots       |
| parkingsessions   |
| parkingslots      |
| vehicles          |
+------------------+
6 rows in set (0.07 sec)
```

```
mysql> describe customers;
+-------------------+-----------------------------+------+-----+-------------------+-------------------+
| Field             | Type                        | Null | Key | Default           | Extra             |
+-------------------+-----------------------------+------+-----+-------------------+-------------------+
| customer_id       | int                         | NO   | PRI | NULL
   | auto_increment    |
| name              | varchar(100)                | NO   |     | NULL
   |                   |
| phone_number      | varchar(15)                 | YES  | UNI | NULL
   |                   |
| email             | varchar(100)                | YES  | UNI | NULL
   |                   |
| membership_status | enum('member','non-member') | YES  |     | non-member        |                   |
| membership_id     | int                         | YES  | MUL | NULL
   |                   |
| registered_on     | timestamp                   | YES  |     | CURRENT_TIMESTAMP | DEFAULT_GENERATED |
+-------------------+-----------------------------+------+-----+-------------------+-------------------+
7 rows in set (0.04 sec)

mysql> describe membershipplans;
+--------------------+--------------+------+-----+---------+----------------+
| Field              | Type         | Null | Key | Default | Extra          |
+--------------------+--------------+------+-----+---------+----------------+
| plan_id            | int          | NO   | PRI | NULL    | auto_increment |
| plan_name          | varchar(50)  | NO   |     | NULL    |                |
| monthly_fee        | decimal(10,2)| NO   |     | NULL    |                |
| hourly_discount    | decimal(5,2) | NO   |     | NULL    |                |
| additional_benefits| text         | YES  |     | NULL    |                |
+--------------------+--------------+------+-----+---------+----------------+
5 rows in set (0.00 sec)
```

```
mysql> describe vehicles;
+---------------+-------------+------+-----+-------------------+-------------------+
| Field         | Type        | Null | Key | Default           | Extra
   |
+---------------+-------------+------+-----+-------------------+-------------------+
| vehicle_id    | int         | NO   | PRI | NULL              | auto_increment    |
| license_plate | varchar(15) | NO   | UNI | NULL              |
   |
| customer_id   | int         | NO   | MUL | NULL              |
   |
| entry_time    | timestamp   | YES  |     | CURRENT_TIMESTAMP | DEFAULT_GENERATED |
| lot_id        | int         | NO   | MUL | NULL              |
   |
+---------------+-------------+------+-----+-------------------+-------------------+
5 rows in set (0.01 sec)

mysql> describe parkinglots;
+-------------+--------------+------+-----+---------+-------+
| Field       | Type         | Null | Key | Default | Extra |
+-------------+--------------+------+-----+---------+-------+
| lot_id      | int          | NO   | PRI | NULL    |       |
| location    | varchar(100) | NO   |     | NULL    |       |
| total_spots | int          | NO   |     | NULL    |       |
+-------------+--------------+------+-----+---------+-------+
3 rows in set (0.00 sec)

mysql> describe parkingslots;
+---------+---------------------------+------+-----+---------+----------------+
| Field   | Type                      | Null | Key | Default | Extra          |
+---------+---------------------------+------+-----+---------+----------------+
| slot_id | int                       | NO   | PRI | NULL    | auto_increment |
| lot_id  | int                       | NO   | MUL | NULL    |                |
| status  | enum('vacant','occupied') | YES  |     | vacant  |                |
+---------+---------------------------+------+-----+---------+----------------+
3 rows in set (0.00 sec)
```

```
mysql> describe parkingsessions;
+------------+---------------+------+-----+-------------------+-------------------+
| Field      | Type          | Null | Key | Default           | Extra
  |
+------------+---------------+------+-----+-------------------+-------------------+
| session_id | int           | NO   | PRI | NULL              | auto_increment    |
| vehicle_id | int           | NO   | MUL | NULL              |
  |
| slot_id    | int           | NO   | MUL | NULL              |
  |
| entry_time | timestamp     | YES  |     | CURRENT_TIMESTAMP | DEFAULT_GENERATED |
| exit_time  | timestamp     | YES  |     | NULL              |
  |
| price      | decimal(10,2) | YES  |     | NULL              |
  |
+------------+---------------+------+-----+-------------------+-------------------+
6 rows in set (0.00 sec)
```

```
mysql> DELIMITER //
mysql> CREATE TRIGGER after_parkinglot_insert
    -> AFTER INSERT ON parkinglots
    -> FOR EACH ROW
    -> BEGIN
    ->     DECLARE i INT DEFAULT 1;
    ->     WHILE i <= NEW.total_spots DO
    ->         INSERT INTO parkingslots (lot_id, status) VALUES (NEW.lot_id, 'vacant');
    ->         SET i = i + 1;
    ->     END WHILE;
    -> END//
Query OK, 0 rows affected (0.03 sec)
```

```
mysql> delimiter ;
mysql> INSERT INTO parkinglots (lot_id, location, total_spots)
    -> VALUES
    ->     (1, 'Downtown', 4),
    ->     (2, 'City Center', 5),
    ->     (3, 'Green Park', 4),
    ->     (4, 'Mall Area', 5),
    ->     (5, 'Tech Valley', 4),
    ->     (6, 'Seaside', 5);
Query OK, 6 rows affected (0.03 sec)
Records: 6  Duplicates: 0  Warnings: 0

mysql> select * from parkinglots;
+--------+-------------+-------------+
| lot_id | location    | total_spots |
+--------+-------------+-------------+
|      1 | Downtown    |           4 |
|      2 | City Center |           5 |
|      3 | Green Park  |           4 |
|      4 | Mall Area   |           5 |
|      5 | Tech Valley |           4 |
|      6 | Seaside     |           5 |
+--------+-------------+-------------+
6 rows in set (0.00 sec)
```

```
mysql> select * from parkingslots;
+---------+--------+--------+
| slot_id | lot_id | status |
+---------+--------+--------+
|       1 |      1 | vacant |
|       2 |      1 | vacant |
|       3 |      1 | vacant |
|       4 |      1 | vacant |
|       5 |      2 | vacant |
|       6 |      2 | vacant |
|       7 |      2 | vacant |
|       8 |      2 | vacant |
|       9 |      2 | vacant |
|      10 |      3 | vacant |
|      11 |      3 | vacant |
|      12 |      3 | vacant |
|      13 |      3 | vacant |
|      14 |      4 | vacant |
|      15 |      4 | vacant |
|      16 |      4 | vacant |
|      17 |      4 | vacant |
|      18 |      4 | vacant |
|      19 |      5 | vacant |
|      20 |      5 | vacant |
|      21 |      5 | vacant |
|      22 |      5 | vacant |
|      23 |      6 | vacant |
|      24 |      6 | vacant |
|      25 |      6 | vacant |
|      26 |      6 | vacant |
|      27 |      6 | vacant |
+---------+--------+--------+
27 rows in set (0.00 sec)
```

```
mysql> INSERT INTO MembershipPlans (plan_name, monthly_fee, hourly_discount, additional_benefits)
    -> VALUES
    ->     ('Basic Plan', 20.00, 5.00, '5% discount on hourly rates, no additional benefits'),
    ->     ('Silver Plan', 50.00, 10.00, '10% discount on hourly rates, priority parking allocation'),
    ->     ('Gold Plan', 100.00, 15.00, '15% discount on hourly rates, access to premium parking lots'),
    ->     ('Platinum Plan', 200.00, 20.00, '20% discount on hourly rates, reserved slots, valet service');
Query OK, 4 rows affected (0.01 sec)
Records: 4  Duplicates: 0  Warnings: 0

mysql> select * from MembershipPlans;
+---------+--------------+-------------+-----------------+-----------------------------------------------------------------+
| plan_id | plan_name    | monthly_fee | hourly_discount | additional_benefits                                             |
+---------+--------------+-------------+-----------------+-----------------------------------------------------------------+
|       1 | Basic Plan   |       20.00 |            5.00 | 5% discount on hourly rates, no additional benefits             |
|       2 | Silver Plan  |       50.00 |           10.00 | 10% discount on hourly rates, priority parking allocation       |
|       3 | Gold Plan    |      100.00 |           15.00 | 15% discount on hourly rates, access to premium parking lots    |
|       4 | Platinum Plan |     200.00 |           20.00 | 20% discount on hourly rates, reserved slots, valet service     |
+---------+--------------+-------------+-----------------+-----------------------------------------------------------------+
4 rows in set (0.00 sec)
```

```
mysql> INSERT INTO Customers (name, phone_number, email, membership_status, membership_id, registered_on)
    -> VALUES
    ->     ('Alice Johnson', '9876543210', 'alice.johnson@example.com', 'member', 2, '2024-01-05 10:30:00'),
    ->     ('Bob Smith', '9876543211', 'bob.smith@example.com', 'member', 3, '2024-02-10 11:00:00'),
    ->     ('Carol Lee', '9876543212', 'carol.lee@example.com', 'non-member', NULL, '2024-03-15 09:45:00'),
    ->     ('David Brown', '9876543213', 'david.brown@example.com', 'member', 1, '2024-04-20 15:20:00'),
    ->     ('Eve Green', '9876543214', 'eve.green@example.com', 'non-member', NULL, '2024-05-25 14:10:00'),
    ->     ('Frank White', '9876543215', 'frank.white@example.com', 'member', 4, '2024-06-30 08:55:00');
Query OK, 6 rows affected (0.01 sec)
Records: 6  Duplicates: 0  Warnings: 0

mysql> select * from Customers;
+-------------+---------------+--------------+--------------------------+-------------------+---------------+---------------------+
| customer_id | name          | phone_number | email                    | membership_status | membership_id | registered_on       |
+-------------+---------------+--------------+--------------------------+-------------------+---------------+---------------------+
|           1 | Alice Johnson | 9876543210   | alice.johnson@example.com | member            |             2 | 2024-01-05 10:30:00 |
|           2 | Bob Smith     | 9876543211   | bob.smith@example.com     | member            |             3 | 2024-02-10 11:00:00 |
|           3 | Carol Lee     | 9876543212   | carol.lee@example.com     | non-member        |          NULL | 2024-03-15 09:45:00 |
|           4 | David Brown   | 9876543213   | david.brown@example.com   | member            |             1 | 2024-04-20 15:20:00 |
|           5 | Eve Green     | 9876543214   | eve.green@example.com     | non-member        |          NULL | 2024-05-25 14:10:00 |
|           6 | Frank White   | 9876543215   | frank.white@example.com   | member            |             4 | 2024-06-30 08:55:00 |
+-------------+---------------+--------------+--------------------------+-------------------+---------------+---------------------+
6 rows in set (0.00 sec)
```

```
mysql> INSERT INTO Vehicles (license_plate, customer_id, lot_id)
    -> VALUES
    ->      ('ABC123', 1, 1),
    ->     ('XYZ789', 1, 1),
    ->     ('LMN456', 2, 2),
    ->     ('PQR678', 2, 2),
    ->     ('DEF234', 3, 3),
    ->     ('GHI567', 4, 4),
    ->     ('JKL890', 5, 5),
    ->     ('MNO123', 6, 6);
Query OK, 8 rows affected (0.01 sec)
Records: 8  Duplicates: 0  Warnings: 0

mysql> select * from Vehicles;
+------------+---------------+-------------+---------------------+--------+
| vehicle_id | license_plate | customer_id | entry_time          | lot_id |
+------------+---------------+-------------+---------------------+--------+
|          1 | ABC123        |           1 | 2024-11-17 13:38:44 |      1 |
|          2 | XYZ789        |           1 | 2024-11-17 13:38:44 |      1 |
|          3 | LMN456        |           2 | 2024-11-17 13:38:44 |      2 |
|          4 | PQR678        |           2 | 2024-11-17 13:38:44 |      2 |
|          5 | DEF234        |           3 | 2024-11-17 13:38:44 |      3 |
|          6 | GHI567        |           4 | 2024-11-17 13:38:44 |      4 |
|          7 | JKL890        |           5 | 2024-11-17 13:38:44 |      5 |
|          8 | MNO123        |           6 | 2024-11-17 13:38:44 |      6 |
+------------+---------------+-------------+---------------------+--------+
8 rows in set (0.00 sec)
```

```
mysql> DELIMITER //
mysql> CREATE PROCEDURE StartParkingSession(IN vehicle_id INT, IN desired_lot INT)
    -> BEGIN
    ->     DECLARE vacant_slot INT;
    ->      SELECT slot_id INTO vacant_slot
    ->     FROM ParkingSlots
    ->     WHERE lot_id = desired_lot AND status = 'vacant'
    ->     LIMIT 1;
    ->     IF vacant_slot IS NOT NULL THEN
    ->         -- Insert into ParkingSessions
    ->         INSERT INTO ParkingSessions (vehicle_id, slot_id)
    ->         VALUES (vehicle_id, vacant_slot);
    ->         UPDATE ParkingSlots
    ->         SET status = 'occupied'
    ->         WHERE slot_id = vacant_slot;
    ->
    ->         SELECT CONCAT('Parking session started for Vehicle ID ', vehicle_id, ' in Slot ID ', vacant_slot) AS message;
    ->     ELSE
    ->         SELECT 'No vacant slots available in the requested lot.' AS message;
    ->     END IF;
    -> END //
Query OK, 0 rows affected (0.04 sec)
```

```
mysql> DELIMITER //
mysql> CREATE PROCEDURE CalculateParkingFee(IN p_session_id INT)
    -> BEGIN
    ->     DECLARE p_vehicle_id INT;
    ->     DECLARE p_customer_id INT;
    ->     DECLARE p_membership_id INT;
    ->     DECLARE p_entry_time TIMESTAMP;
    ->     DECLARE p_exit_time TIMESTAMP;
    ->     DECLARE p_duration_hours DECIMAL(10, 2);
    ->     DECLARE p_total_price DECIMAL(10, 2);
    ->     DECLARE p_discount_rate DECIMAL(10, 2) DEFAULT 0.00;
    ->     DECLARE hourly_rate DECIMAL(10, 2) DEFAULT 100.00;
    ->     DECLARE p_message VARCHAR(255);
    ->     SELECT ps.vehicle_id, ps.exit_time, v.customer_id
    ->     INTO p_vehicle_id, p_exit_time, p_customer_id
    ->     FROM ParkingSessions ps
    ->     JOIN Vehicles v ON ps.vehicle_id = v.vehicle_id
    ->     WHERE ps.session_id = p_session_id
    ->     LIMIT 1;
    ->     SELECT entry_time
    ->     INTO p_entry_time
    ->     FROM Vehicles
    ->     WHERE vehicle_id = p_vehicle_id
    ->     LIMIT 1;
    ->     SELECT membership_id
    ->     INTO p_membership_id
    ->     FROM Customers
    ->     WHERE customer_id = p_customer_id
    ->     LIMIT 1;
    ->     IF p_membership_id IS NOT NULL THEN
    ->         SELECT hourly_discount
    ->         INTO p_discount_rate
    ->         FROM MembershipPlans
    ->         WHERE plan_id = p_membership_id
    ->         LIMIT 1;
    ->     END IF;
    ->     SET p_duration_hours = TIMESTAMPDIFF(MINUTE, p_entry_time, p_exit_time) / 60;
    ->     SET p_total_price = (p_duration_hours * hourly_rate) * (1 - (p_discount_rate / 100));
    ->     UPDATE ParkingSessions
    ->     SET price = p_total_price
    ->     WHERE session_id = p_session_id;
    ->     SET p_message = CONCAT('Total parking fee for session ', p_session_id,
    ->                            ' is $', FORMAT(p_total_price, 2),
    ->                            '. Discount applied: ', FORMAT(p_discount_rate, 2), '%');
    ->     SELECT p_message AS fee_message;
    -> END //
Query OK, 0 rows affected (0.01 sec)
```

```
mysql> DELIMITER //
mysql> CREATE TRIGGER AfterParkingSessionEnd
    -> AFTER UPDATE ON ParkingSessions
    -> FOR EACH ROW
    -> BEGIN
    ->     IF NEW.exit_time IS NOT NULL THEN
    ->         UPDATE ParkingSlots
    ->         SET status = 'vacant'
    ->         WHERE slot_id = NEW.slot_id;
    ->     END IF;
    -> END //
Query OK, 0 rows affected (0.03 sec)
```

```
mysql> call StartParkingSession(1,1);
+-----------------------------------------------------+
| message                                             |
+-----------------------------------------------------+
| Parking session started for Vehicle ID 1 in Slot ID 1 |
+-----------------------------------------------------+
1 row in set (0.04 sec)

Query OK, 0 rows affected (0.05 sec)

mysql> call StartParkingSession(2,1);
+-----------------------------------------------------+
| message                                             |
+-----------------------------------------------------+
| Parking session started for Vehicle ID 2 in Slot ID 2 |
+-----------------------------------------------------+
1 row in set (0.03 sec)

Query OK, 0 rows affected (0.04 sec)

mysql> call StartParkingSession(7,5);
+-----------------------------------------------------+
| message                                             |
+-----------------------------------------------------+
| Parking session started for Vehicle ID 7 in Slot ID 19 |
+-----------------------------------------------------+
1 row in set (0.03 sec)

Query OK, 0 rows affected (0.04 sec)

mysql> call StartParkingSession(4,2);
+-----------------------------------------------------+
| message                                             |
+-----------------------------------------------------+
| Parking session started for Vehicle ID 4 in Slot ID 5 |
+-----------------------------------------------------+
1 row in set (0.03 sec)

Query OK, 0 rows affected (0.04 sec)
```

```
mysql> select * from parkingslots;
+---------+--------+----------+
| slot_id | lot_id | status   |
+---------+--------+----------+
|       1 |      1 | occupied |
|       2 |      1 | occupied |
|       3 |      1 | vacant   |
|       4 |      1 | vacant   |
|       5 |      2 | occupied |
|       6 |      2 | vacant   |
|       7 |      2 | vacant   |
|       8 |      2 | vacant   |
|       9 |      2 | vacant   |
|      10 |      3 | vacant   |
|      11 |      3 | vacant   |
|      12 |      3 | vacant   |
|      13 |      3 | vacant   |
|      14 |      4 | vacant   |
|      15 |      4 | vacant   |
|      16 |      4 | vacant   |
|      17 |      4 | vacant   |
|      18 |      4 | vacant   |
|      19 |      5 | occupied |
|      20 |      5 | vacant   |
|      21 |      5 | vacant   |
|      22 |      5 | vacant   |
|      23 |      6 | vacant   |
|      24 |      6 | vacant   |
|      25 |      6 | vacant   |
|      26 |      6 | vacant   |
|      27 |      6 | vacant   |
+---------+--------+----------+
27 rows in set (0.00 sec)
```

```
mysql> select * from ParkingSessions;
+------------+------------+---------+---------------------+-----------+-------+
| session_id | vehicle_id | slot_id | entry_time          | exit_time | price |
+------------+------------+---------+---------------------+-----------+-------+
|          1 |          1 |       1 | 2024-11-17 13:47:27 | NULL      |  NULL |
|          2 |          2 |       2 | 2024-11-17 13:47:37 | NULL      |  NULL |
|          3 |          7 |      19 | 2024-11-17 13:47:48 | NULL      |  NULL |
|          4 |          4 |       5 | 2024-11-17 13:48:00 | NULL      |  NULL |
+------------+------------+---------+---------------------+-----------+-------+
4 rows in set (0.00 sec)
```

```
mysql> update parkingsessions set exit_time=CURRENT_TIMESTAMP where session_id=1;
Query OK, 1 row affected (0.03 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

```
mysql> call CalculateParkingFee(1);
+-----------------------------------------------------------------------+
| fee_message                                                           |
+-----------------------------------------------------------------------+
| Total parking fee for session 1 is $24.30. Discount applied: 10.00% |
+-----------------------------------------------------------------------+
1 row in set (0.01 sec)

Query OK, 0 rows affected (0.01 sec)

mysql> select * from parkingsessions;
+------------+------------+---------+---------------------+---------------------+-------+
| session_id | vehicle_id | slot_id | entry_time          | exit_time           | price |
+------------+------------+---------+---------------------+---------------------+-------+
|          1 |          1 |       1 | 2024-11-17 13:47:27 | 2024-11-17 14:00:22 | 24.30 |
|          2 |          2 |       2 | 2024-11-17 13:47:37 | NULL                |  NULL |
|          3 |          7 |      19 | 2024-11-17 13:47:48 | NULL                |  NULL |
|          4 |          4 |       5 | 2024-11-17 13:48:00 | 2024-11-17 14:33:53 | 69.70 |
|          5 |          9 |      20 | 2024-11-17 14:32:49 | 2024-11-17 15:15:45 | 72.00 |
|          6 |         10 |      14 | 2024-11-17 15:14:32 | NULL                |  NULL |
+------------+------------+---------+---------------------+---------------------+-------+
6 rows in set (0.00 sec)
```

```
mysql> select * from parkingslots;
+---------+--------+----------+
| slot_id | lot_id | status   |
+---------+--------+----------+
|       1 |      1 | vacant   |
|       2 |      1 | occupied |
|       3 |      1 | vacant   |
|       4 |      1 | vacant   |
|       5 |      2 | occupied |
|       6 |      2 | vacant   |
|       7 |      2 | vacant   |
|       8 |      2 | vacant   |
|       9 |      2 | vacant   |
|      10 |      3 | vacant   |
|      11 |      3 | vacant   |
|      12 |      3 | vacant   |
|      13 |      3 | vacant   |
|      14 |      4 | vacant   |
|      15 |      4 | vacant   |
|      16 |      4 | vacant   |
|      17 |      4 | vacant   |
|      18 |      4 | vacant   |
|      19 |      5 | occupied |
|      20 |      5 | vacant   |
|      21 |      5 | vacant   |
|      22 |      5 | vacant   |
|      23 |      6 | vacant   |
|      24 |      6 | vacant   |
|      25 |      6 | vacant   |
|      26 |      6 | vacant   |
|      27 |      6 | vacant   |
+---------+--------+----------+
27 rows in set (0.00 sec)
```

- **TRIGGER : if new lot is added to parkinglot table then new rows will be inserted into parkingslots respectively.**

```
DELIMITER //
CREATE TRIGGER after_parkinglot_insert
AFTER INSERT ON parkinglots
FOR EACH ROW
BEGIN
  DECLARE i INT DEFAULT 1;
  WHILE i <= NEW.total_spots DO
```

```
        INSERT INTO parkingslots (lot_id, status) VALUES (NEW.lot_id, 'vacant');
        SET i = i + 1;
    END WHILE;
END//
```

- **PROCEDURE**: Start the parking session for a specific vehicle

```
DELIMITER //
CREATE PROCEDURE StartParkingSession(IN vehicle_id INT, IN desired_lot INT)
BEGIN
    DECLARE vacant_slot INT;
     SELECT slot_id INTO vacant_slot
    FROM ParkingSlots
    WHERE lot_id = desired_lot AND status = 'vacant'
    LIMIT 1;
    IF vacant_slot IS NOT NULL THEN
        -- Insert into ParkingSessions
        INSERT INTO ParkingSessions (vehicle_id, slot_id)
        VALUES (vehicle_id, vacant_slot);
        UPDATE ParkingSlots
        SET status = 'occupied'
        WHERE slot_id = vacant_slot;

        SELECT CONCAT('Parking session started for Vehicle ID ', vehicle_id, ' in Slot ID ',
vacant_slot) AS message;
    ELSE
        SELECT 'No vacant slots available in the requested lot.' AS message;
    END IF;
END //
```

- **TRIGGER:update the status to vacant once the vehicle exits the parkingsession**

```
DELIMITER //
CREATE TRIGGER AfterParkingSessionEnd
AFTER UPDATE ON ParkingSessions
FOR EACH ROW
BEGIN
    IF NEW.exit_time IS NOT NULL THEN
        UPDATE ParkingSlots
```

```
    SET status = 'vacant'
    WHERE slot_id = NEW.slot_id;
  END IF;
END //
```

- **PROCEDURE: Calculate the parking fee**

```
DELIMITER //
CREATE PROCEDURE CalculateParkingFee(IN p_session_id INT)
BEGIN
  DECLARE p_vehicle_id INT;
  DECLARE p_customer_id INT;
  DECLARE p_membership_id INT;
  DECLARE p_entry_time TIMESTAMP;
  DECLARE p_exit_time TIMESTAMP;
  DECLARE p_duration_hours DECIMAL(10, 2);
  DECLARE p_total_price DECIMAL(10, 2);
  DECLARE p_discount_rate DECIMAL(10, 2) DEFAULT 0.00;
  DECLARE hourly_rate DECIMAL(10, 2) DEFAULT 100.00;
  DECLARE p_message VARCHAR(255);
  SELECT ps.vehicle_id, ps.exit_time, v.customer_id
  INTO p_vehicle_id, p_exit_time, p_customer_id
  FROM ParkingSessions ps
  JOIN Vehicles v ON ps.vehicle_id = v.vehicle_id
  WHERE ps.session_id = p_session_id
  LIMIT 1;
  SELECT entry_time
  INTO p_entry_time
  FROM Vehicles
  WHERE vehicle_id = p_vehicle_id
  LIMIT 1;
  SELECT membership_id
  INTO p_membership_id
  FROM Customers
  WHERE customer_id = p_customer_id
  LIMIT 1;
  IF p_membership_id IS NOT NULL THEN
```

```
        SELECT hourly_discount
        INTO p_discount_rate
        FROM MembershipPlans
        WHERE plan_id = p_membership_id
        LIMIT 1;
    END IF;
    SET p_duration_hours = TIMESTAMPDIFF(MINUTE, p_entry_time, p_exit_time) / 60;
    SET p_total_price = (p_duration_hours * hourly_rate) * (1 - (p_discount_rate / 100));
    UPDATE ParkingSessions
    SET price = p_total_price
    WHERE session_id = p_session_id;
    SET p_message = CONCAT('Total parking fee for session ', p_session_id,
                           ' is $', FORMAT(p_total_price, 2),
                           '. Discount applied: ', FORMAT(p_discount_rate, 2), '%');
    SELECT p_message AS fee_message;
END //
```

{between,like}

```
mysql> SELECT * FROM Customers WHERE registered_on
    -> BETWEEN '2024-01-01' AND '2024-12-31';
+-------------+--------------+--------------+---------------------------+-----------------+---------------+---------------------+
| customer_id | name         | phone_number | email                     | membership_status | membership_id | registered_on       |
+-------------+--------------+--------------+---------------------------+-----------------+---------------+---------------------+
|           1 | Alice Johnson | 9876543210   | alice.johnson@example.com | member          |             2 | 2024-01-05 10:30:00 |
|           2 | Bob Smith     | 9876543211   | bob.smith@example.com     | member          |             3 | 2024-02-10 11:00:00 |
|           3 | Carol Lee     | 9876543212   | carol.lee@example.com     | non-member      |          NULL | 2024-03-15 09:45:00 |
|           4 | David Brown   | 9876543213   | david.brown@example.com   | member          |             1 | 2024-04-20 15:20:00 |
|           5 | Eve Green     | 9876543214   | eve.green@example.com     | non-member      |          NULL | 2024-05-25 14:10:00 |
|           6 | Frank White   | 9876543215   | frank.white@example.com   | member          |             4 | 2024-06-30 08:55:00 |
+-------------+--------------+--------------+---------------------------+-----------------+---------------+---------------------+
6 rows in set (0.00 sec)
```

```
mysql> SELECT email FROM Customers
    ->  WHERE email LIKE "%white%";
+---------------------------+
| email                     |
+---------------------------+
| frank.white@example.com   |
+---------------------------+
1 row in set (0.00 sec)
```

```
mysql> SELECT * FROM Vehicles WHERE license_plate LIKE 'AB%';
+------------+---------------+-------------+---------------------+--------+
| vehicle_id | license_plate | customer_id | entry_time          | lot_id |
+------------+---------------+-------------+---------------------+--------+
|          1 | ABC123        |           1 | 2024-11-17 17:49:23 |      1 |
+------------+---------------+-------------+---------------------+--------+
1 row in set (0.01 sec)
```

{order by}

```
mysql> SELECT * FROM Customers ORDER BY registered_on DESC;
+-------------+--------------+--------------+-------------------------+-----------------+---------------+---------------------+
| customer_id | name         | phone_number | email                   | membership_status | membership_id | registered_on       |
+-------------+--------------+--------------+-------------------------+-----------------+---------------+---------------------+
|           6 | Frank White  | 9876543215   | frank.white@example.com | member          |             4 | 2024-06-30 08:55:00 |
|           5 | Eve Green    | 9876543214   | eve.green@example.com   | non-member      |          NULL | 2024-05-25 14:10:00 |
|           4 | David Brown  | 9876543213   | david.brown@example.com | member          |             1 | 2024-04-20 15:20:00 |
|           3 | Carol Lee    | 9876543212   | carol.lee@example.com   | non-member      |          NULL | 2024-03-15 09:45:00 |
|           2 | Bob Smith    | 9876543211   | bob.smith@example.com   | member          |             3 | 2024-02-10 11:00:00 |
|           1 | Alice Johnson| 9876543210   | alice.johnson@example.com | member        |             2 | 2024-01-05 10:30:00 |
+-------------+--------------+--------------+-------------------------+-----------------+---------------+---------------------+
6 rows in set (0.00 sec)
```

```
mysql> SELECT *FROM MembershipPlans ORDER BY hourly_discount DESC;
+---------+--------------+-------------+-----------------+-----------------------------------------------------------------+
| plan_id | plan_name    | monthly_fee | hourly_discount | additional_benefits                                             |
+---------+--------------+-------------+-----------------+-----------------------------------------------------------------+
|       4 | Platinum Plan|      200.00 |           20.00 | 20% discount on hourly rates, reserved slots, valet service     |
|       3 | Gold Plan    |      100.00 |           15.00 | 15% discount on hourly rates, access to premium parking lots    |
|       2 | Silver Plan  |       50.00 |           10.00 | 10% discount on hourly rates, priority parking allocation       |
|       1 | Basic Plan   |       20.00 |            5.00 | 5% discount on hourly rates, no additional benefits             |
+---------+--------------+-------------+-----------------+-----------------------------------------------------------------+
4 rows in set (0.00 sec)
```

{GROUP BY, COUNT}

```
mysql> SELECT lot_id, COUNT(*) AS total_vehicles FROM Vehicles GROUP BY lot_id;
+--------+----------------+
| lot_id | total_vehicles |
+--------+----------------+
|      1 |              2 |
|      2 |              2 |
|      3 |              1 |
|      4 |              1 |
|      5 |              1 |
|      6 |              1 |
+--------+----------------+
6 rows in set (0.00 sec)
```

```
mysql> SELECT lot_id, status, COUNT(*) AS slot_count FROM ParkingSlots GROUP BY lot_id, status;
+--------+--------+------------+
| lot_id | status | slot_count |
+--------+--------+------------+
|     10 | vacant |          4 |
|     20 | vacant |          5 |
|     30 | vacant |          4 |
|     40 | vacant |          5 |
|     50 | vacant |          4 |
|     60 | vacant |          5 |
+--------+--------+------------+
6 rows in set (0.01 sec)
```

{ cartesian product}

```
mysql> SELECT c.name AS customer_name, m.plan_name AS membership_plan
    -> FROM Customers c, MembershipPlans m;
+---------------+-----------------+
| customer_name | membership_plan |
+---------------+-----------------+
| Alice Johnson | Platinum Plan   |
| Alice Johnson | Gold Plan       |
| Alice Johnson | Silver Plan     |
| Alice Johnson | Basic Plan      |
| Bob Smith     | Platinum Plan   |
| Bob Smith     | Gold Plan       |
| Bob Smith     | Silver Plan     |
| Bob Smith     | Basic Plan      |
| Carol Lee     | Platinum Plan   |
| Carol Lee     | Gold Plan       |
| Carol Lee     | Silver Plan     |
| Carol Lee     | Basic Plan      |
| David Brown   | Platinum Plan   |
| David Brown   | Gold Plan       |
| David Brown   | Silver Plan     |
| David Brown   | Basic Plan      |
| Eve Green     | Platinum Plan   |
| Eve Green     | Gold Plan       |
| Eve Green     | Silver Plan     |
| Eve Green     | Basic Plan      |
| Frank White   | Platinum Plan   |
| Frank White   | Gold Plan       |
| Frank White   | Silver Plan     |
| Frank White   | Basic Plan      |
+---------------+-----------------+
24 rows in set (0.00 sec)
```

{inner join}

```
mysql> SELECT c.name AS customer_name, m.plan_name AS membership_plan
    -> FROM Customers c
    -> INNER JOIN MembershipPlans m ON c.membership_id = m.plan_id;
+---------------+-----------------+
| customer_name | membership_plan |
+---------------+-----------------+
| David Brown   | Basic Plan      |
| Alice Johnson | Silver Plan     |
| Bob Smith     | Gold Plan       |
| Frank White   | Platinum Plan   |
+---------------+-----------------+
4 rows in set (0.00 sec)
```

```
mysql> SELECT pl.location AS lot_location, ps.slot_id
    -> FROM ParkingSlots ps
    -> INNER JOIN ParkingLots pl ON ps.lot_id = pl.lot_id
    -> WHERE ps.status = 'vacant';
+--------------+---------+
| lot_location | slot_id |
+--------------+---------+
| Downtown     |       1 |
| Downtown     |       2 |
| Downtown     |       3 |
| Downtown     |       4 |
| City Center  |       5 |
| City Center  |       6 |
| City Center  |       7 |
| City Center  |       8 |
| City Center  |       9 |
| Green Park   |      10 |
| Green Park   |      11 |
| Green Park   |      12 |
| Green Park   |      13 |
| Mall Area    |      14 |
| Mall Area    |      15 |
| Mall Area    |      16 |
| Mall Area    |      17 |
| Mall Area    |      18 |
| Tech Valley  |      19 |
| Tech Valley  |      20 |
| Tech Valley  |      21 |
| Tech Valley  |      22 |
| Seaside      |      23 |
| Seaside      |      24 |
| Seaside      |      25 |
| Seaside      |      26 |
| Seaside      |      27 |
+--------------+---------+
27 rows in set (0.00 sec)
```

{left outer join}

```
mysql> SELECT c.name AS customer_name, COUNT(v.vehicle_id) AS total_vehicles
    -> FROM Customers c
    -> LEFT JOIN Vehicles v ON c.customer_id = v.customer_id
    -> GROUP BY c.customer_id;
+---------------+----------------+
| customer_name | total_vehicles |
+---------------+----------------+
| Alice Johnson |              2 |
| Bob Smith     |              2 |
| Carol Lee     |              1 |
| David Brown   |              1 |
| Eve Green     |              1 |
| Frank White   |              1 |
+---------------+----------------+
6 rows in set (0.00 sec)
```

{right outer join}

```
mysql> SELECT m.plan_name, c.name AS customer_name
    -> FROM MembershipPlans m
    -> RIGHT JOIN Customers c ON c.membership_id = m.plan_id
    -> WHERE m.plan_id IS NOT NULL;
+---------------+---------------+
| plan_name     | customer_name |
+---------------+---------------+
| Basic Plan    | David Brown   |
| Silver Plan   | Alice Johnson |
| Gold Plan     | Bob Smith     |
| Platinum Plan | Frank White   |
+---------------+---------------+
4 rows in set (0.00 sec)
```

```
mysql> SELECT pl.location AS lot_location, COUNT(v.vehicle_id) AS total_vehicles
    -> FROM ParkingLots pl
    -> RIGHT JOIN Vehicles v ON pl.lot_id = v.lot_id
    -> GROUP BY pl.lot_id;
+--------------+----------------+
| lot_location | total_vehicles |
+--------------+----------------+
| Downtown     |              2 |
| City Center  |              2 |
| Green Park   |              1 |
| Mall Area    |              1 |
| Tech Valley  |              1 |
| Seaside      |              1 |
+--------------+----------------+
6 rows in set (0.00 sec)
```

{view}

```
mysql> CREATE VIEW ActiveParkingSessions AS
    -> SELECT ps.session_id, v.license_plate, c.name AS customer_name, ps.entry_time
    -> FROM ParkingSessions ps
    -> ^C
mysql> CREATE VIEW ActiveParkingSessions AS
    -> SELECT ps.session_id, v.license_plate, c.name AS customer_name, ps.entry_time
    -> FROM ParkingSessions ps
    -> INNER JOIN Vehicles v ON ps.vehicle_id = v.vehicle_id
    -> INNER JOIN Customers c ON v.customer_id = c.customer_id
    -> WHERE ps.exit_time IS NULL;
Query OK, 0 rows affected (0.01 sec)

mysql> SELECT * FROM ActiveParkingSessions;
Empty set (0.00 sec)
```

{index}

```
mysql> CREATE INDEX idx_phone_number ON Customers(phone_number);
Query OK, 0 rows affected (0.07 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> CREATE INDEX idx_lot_status ON ParkingSlots(lot_id, status);
Query OK, 0 rows affected (0.05 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

{ IN, NOT IN}

```
mysql> select * from customers
    -> WHERE membership_id IN (1, 2, 3);
+-------------+--------------+--------------+--------------------------+-------------------+--------------+---------------------+
| customer_id | name         | phone_number | email                    | membership_status | membership_id | registered_on       |
+-------------+--------------+--------------+--------------------------+-------------------+--------------+---------------------+
|           4 | David Brown  | 9876543213   | david.brown@example.com  | member            |             1 | 2024-04-20 15:20:00 |
|           1 | Alice Johnson | 9876543210  | alice.johnson@example.com | member           |             2 | 2024-01-05 10:30:00 |
|           2 | Bob Smith    | 9876543211   | bob.smith@example.com    | member            |             3 | 2024-02-10 11:00:00 |
+-------------+--------------+--------------+--------------------------+-------------------+--------------+---------------------+
3 rows in set (0.00 sec)
```

```
mysql> select * from customers
    -> WHERE customer_id IN (
    ->  SELECT customer_id
    ->  from vehicles
    ->  WHERE lot_id = 1);
+-------------+--------------+--------------+--------------------------+-------------------+--------------+---------------------+
| customer_id | name         | phone_number | email                    | membership_status | membership_id | registered_on       |
+-------------+--------------+--------------+--------------------------+-------------------+--------------+---------------------+
|           1 | Alice Johnson | 9876543210  | alice.johnson@example.com | member            |             2 | 2024-01-05 10:30:00 |
+-------------+--------------+--------------+--------------------------+-------------------+--------------+---------------------+
1 row in set (0.01 sec)
```

```
mysql> SELECT * FROM Vehicles WHERE vehicle_id NOT IN
    ->  ( SELECT vehicle_id FROM ParkingSessions WHERE exit_time IS NULL);
+------------+---------------+-------------+---------------------+--------+
| vehicle_id | license_plate | customer_id | entry_time          | lot_id |
+------------+---------------+-------------+---------------------+--------+
|          1 | ABC123        |           1 | 2024-11-17 17:49:23 |      1 |
|          2 | XYZ789        |           1 | 2024-11-17 17:49:23 |      1 |
|          3 | LMN456        |           2 | 2024-11-17 17:49:23 |      2 |
|          4 | PQR678        |           2 | 2024-11-17 17:49:23 |      2 |
|          5 | DEF234        |           3 | 2024-11-17 17:49:23 |      3 |
|          6 | GHI567        |           4 | 2024-11-17 17:49:23 |      4 |
|          7 | JKL890        |           5 | 2024-11-17 17:49:23 |      5 |
|          8 | MNO123        |           6 | 2024-11-17 17:49:23 |      6 |
+------------+---------------+-------------+---------------------+--------+
8 rows in set (0.01 sec)
```

{aggregate functions}

```
mysql> SELECT COUNT(*) AS total_members
    -> from customers
    -> WHERE membership_status = 'member';
+---------------+
| total_members |
+---------------+
|             4 |
+---------------+
1 row in set (0.00 sec)
```

```
mysql> SELECT SUM(monthly_fee) AS total_high_fee
    -> FROM MembershipPlans
    -> WHERE monthly_fee > 50;
+----------------+
| total_high_fee |
+----------------+
|         300.00 |
+----------------+
1 row in set (0.00 sec)
```

```
mysql> SELECT MIN(registered_on) AS first_registered_customer
    -> FROM Customers;
+---------------------------+
| first_registered_customer |
+---------------------------+
| 2024-01-05 10:30:00       |
+---------------------------+
1 row in set (0.00 sec)
```

```
mysql> SELECT AVG(total_spots) AS avg_spots_per_lot
    -> FROM ParkingLots;
+--------------------+
| avg_spots_per_lot |
+--------------------+
|             4.5000 |
+--------------------+
1 row in set (0.00 sec)
```

{Subqueries}

```
mysql> SELECT *
    -> from customers
    -> WHERE customer_id IN (
    -> select customer_id
    -> from vehicles
    ->  WHERE lot_id = 1);
+-------------+--------------+--------------+-------------------------+-----------------+--------------+---------------------+
| customer_id | name         | phone_number | email                   | membership_status | membership_id | registered_on       |
+-------------+--------------+--------------+-------------------------+-----------------+--------------+---------------------+
|           1 | Alice Johnson | 9876543210   | alice.johnson@example.com | member          |            2 | 2024-01-05 10:30:00 |
+-------------+--------------+--------------+-------------------------+-----------------+--------------+---------------------+
1 row in set (0.00 sec)
```

```
mysql> select *
    -> from Vehicles
    -> WHERE vehicle_id NOT IN (
    -> select vehicle_id
    -> FROM ParkingSessions
    ->  WHERE exit_time IS NULL);
+------------+---------------+-------------+---------------------+--------+
| vehicle_id | license_plate | customer_id | entry_time          | lot_id |
+------------+---------------+-------------+---------------------+--------+
|          1 | ABC123        |           1 | 2024-11-17 17:49:23 |      1 |
|          2 | XYZ789        |           1 | 2024-11-17 17:49:23 |      1 |
|          3 | LMN456        |           2 | 2024-11-17 17:49:23 |      2 |
|          4 | PQR678        |           2 | 2024-11-17 17:49:23 |      2 |
|          5 | DEF234        |           3 | 2024-11-17 17:49:23 |      3 |
|          6 | GHI567        |           4 | 2024-11-17 17:49:23 |      4 |
|          7 | JKL890        |           5 | 2024-11-17 17:49:23 |      5 |
|          8 | MNO123        |           6 | 2024-11-17 17:49:23 |      6 |
+------------+---------------+-------------+---------------------+--------+
8 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM Vehicles WHERE vehicle_id NOT IN
    ->  ( SELECT vehicle_id FROM ParkingSessions WHERE exit_time IS NULL);
+------------+---------------+-------------+---------------------+--------+
| vehicle_id | license_plate | customer_id | entry_time          | lot_id |
+------------+---------------+-------------+---------------------+--------+
|          1 | ABC123        |           1 | 2024-11-17 17:49:23 |      1 |
|          2 | XYZ789        |           1 | 2024-11-17 17:49:23 |      1 |
|          3 | LMN456        |           2 | 2024-11-17 17:49:23 |      2 |
|          4 | PQR678        |           2 | 2024-11-17 17:49:23 |      2 |
|          5 | DEF234        |           3 | 2024-11-17 17:49:23 |      3 |
|          6 | GHI567        |           4 | 2024-11-17 17:49:23 |      4 |
|          7 | JKL890        |           5 | 2024-11-17 17:49:23 |      5 |
|          8 | MNO123        |           6 | 2024-11-17 17:49:23 |      6 |
+------------+---------------+-------------+---------------------+--------+
8 rows in set (0.01 sec)
```

JDBC code:

```
package dbms;

import java.sql.*;

import java.util.Scanner;

public class MiniProject {

   private static final String DB_URL = "jdbc:mysql://localhost:3306/dbms_mp";

   private static final String USER = "root";

   private static final String PASSWORD = "vanshika@20";

   public static void main(String[] args) {

      try {

         Connection conn = DriverManager.getConnection(DB_URL, USER, PASSWORD);

         Scanner scanner = new Scanner(System.in);

         int choice = 0;

         System.out.println("Welcome to the Parking System!");

         do {

            System.out.println("\nChoose an option:");

            System.out.println("1. Add a Customer");

            System.out.println("2. Add a Vehicle");
```

```
            System.out.println("3. Start Parking Session");
            System.out.println("4. End Parking Session");
            System.out.println("5. Exit");
            choice = scanner.nextInt();
            switch (choice) {
            case 1:
                addCustomer(conn, scanner);
                break;
            case 2:
                addVehicle(conn, scanner);
                break;
            case 3:
                startParkingSession(conn, scanner);
                break;
            case 4:
                endParkingSession(conn, scanner);
                break;
            case 5: {
                System.out.println("Exiting... Goodbye!");
                break;
            }
            default:
                System.out.println("Invalid choice. Try again.");
            }
        } while (choice != 5);
    } catch (SQLException e) {
        e.printStackTrace();
```

```
    }
}
private static void addCustomer(Connection conn, Scanner scanner) throws SQLException {
    System.out.println("Enter customer name:");
    String name = scanner.next();
    System.out.println("Enter email:");
    String email = scanner.next();
    System.out.println("Enter phone number:");
    String phone = scanner.next();
    System.out.println("Enter membership ID (1 for Basic, 2 for Premium, etc.):");
    int membershipId = scanner.nextInt();
    String sql = "INSERT INTO customers (name, email, phone_number, membership_id) VALUES (?, ?, ?, ?)";
    try {
        PreparedStatement stmt = conn.prepareStatement(sql);
        stmt.setString(1, name);
        stmt.setString(2, email);
        stmt.setString(3, phone);
        stmt.setInt(4, membershipId);
        stmt.executeUpdate();
        System.out.println("Customer added successfully!");
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
private static void addVehicle(Connection conn, Scanner scanner) throws SQLException {
    System.out.println("Enter customer ID:");
    int customerId = scanner.nextInt();
```

```
    System.out.println("Enter license plate:");
    String licensePlate = scanner.next();
    System.out.println("Enter lot ID:");
    int lotid = scanner.nextInt();

    String sql = "INSERT INTO vehicles (license_plate, customer_id,lot_id) VALUES (?, ?,?)";
    try {
        PreparedStatement stmt = conn.prepareStatement(sql);
        stmt.setString(1, licensePlate);
        stmt.setInt(2, customerId);
        stmt.setInt(3,lotid);
        stmt.executeUpdate();
        System.out.println("Vehicle added successfully!");
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
private static void startParkingSession(Connection conn, Scanner scanner) throws SQLException {
    System.out.println("Enter vehicle ID:");
    int vehicleId = scanner.nextInt();
    System.out.println("Enter desired parking lot ID:");
    int lotId = scanner.nextInt();
    CallableStatement stmt = conn.prepareCall("{CALL StartParkingSession(?, ?)}");
    stmt.setInt(1, vehicleId);
    stmt.setInt(2, lotId);
    try {
        ResultSet rs = stmt.executeQuery();
```

```
            while (rs.next()) {
                System.out.println(rs.getString("message"));
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
    private static void endParkingSession(Connection conn, Scanner scanner) throws SQLException {
        System.out.println("Enter session ID to end parking:");
        int sessionId = scanner.nextInt();
        // Update exit time
        String updateExitTimeSQL = "UPDATE ParkingSessions SET exit_time = CURRENT_TIMESTAMP WHERE session_id = ?";
        try {
            PreparedStatement stmt = conn.prepareStatement(updateExitTimeSQL);
            stmt.setInt(1, sessionId);
            int rowsUpdated = stmt.executeUpdate();
            if (rowsUpdated > 0) {
                System.out.println("Exit time updated for session ID " + sessionId);
            } else {
                System.out.println("Invalid session ID.");
                return;
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
        // Calculate parking fee
```

```
        CallableStatement calcFeeStmt = conn.prepareCall("{CALL CalculateParkingFee(?)}");
        calcFeeStmt.setInt(1, sessionId);
        calcFeeStmt.execute();
        // Retrieve updated price
        String fetchPriceSQL = "SELECT price FROM ParkingSessions WHERE session_id = ?";
        try {
            PreparedStatement fetchPriceStmt = conn.prepareStatement(fetchPriceSQL);
            fetchPriceStmt.setInt(1, sessionId);
            try {
                ResultSet rs = fetchPriceStmt.executeQuery();
                if (rs.next()) {
                    double price = rs.getDouble("price");
                    System.out.println("Total parking fee for session ID " + sessionId + ": $" + price);
                }
            } catch (SQLException e) {
                e.printStackTrace();
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```