# Project Group: Group 11

# Project Title: Parking Lot Management System

**Project carried out by**
Regis Yizerwe
Deepali Attavar
Gurjas Chawla
Nissy Shirley Guduri
Shruthi Kannapiran
Supriya Mohan

**BUAN 6320 - Database Foundations for Business Analytics**

**Fall 2024**

# Table of contents

# Introduction

The **Parking Lot Management System** is a technology-driven solution designed to streamline parking operations, reduce manual intervention, and enhance user system addresses common challenges such as inefficient space utilization, operational bottlenecks, and poor customer experiences. Its key features include real-time availability updates, online reservations, categorized space tracking (e.g., standard, compact, handicapped), and automated payment processing, ensuring a hassle-free experience for both drivers and parking lot operators.

The system also offers data analytics to help operators optimize space utilization and identify peak hours, leading to better decision-making. With its scalable design and open-source architecture, the system is adaptable to various parking lot layouts and sizes, making it suitable for malls, supermarkets, and urban areas. It supports the transition toward smart parking solutions, reduces congestion, and minimizes the environmental impact by cutting down vehicle idling time. Overall, this project aims to modernize parking management, improve urban mobility, and create a seamless parking experience for all stakeholders.

# Meeting Minutes

- **Meeting date: September 24, 2024: Microsoft Teams**

This meeting began with team introductions. The primary focus was developing the project charter, which outlined the scope and objectives of the parking management system.

- **Meeting date: September 30, 2024: In-person**

This was mostly a brainstorming session to design the database structure. The team came up with an initial set of tables required for the system, laying the foundation for the database schema.

- **Meeting date: October 7, 2024: In-person**

1. Came up with additional tables to enhance functionality and solve some many-to-many relationships
2. Set primary and foreign key relationships
3. Implemented necessary constraints
4. Distributed the work among team members

- **Meeting Date: October 21, 2024: Microsoft Teams**

1. Reviewed current progress
2. Discussed some stored programs for our system and allocated responsibilities for stored procedures development (divided work for functions, procedures, and triggers)

- **Meeting Date: October 28, 2024: In-person**

1. Data repopulation of database tables
2. Integration of stored procedures into the main script
3. Verification of data integrity

- **Meeting Date: November 18, 2024: In-person**

1. Planned the presentation structure and content
2. Complete presentation materials
3. Brainstorm creative ideas for the project's YouTube video presentation.

# Project Charter

- **Project Title:** Parking Lot Management System
- **Prepared By:**
    - Regis Yizerwe
    - Deepali Attavar
    - Gurjas Chawla
    - Nissy Shirley Guduri
    - Shruthi Kannapiran
    - Supriya Mohan
- **Date**: 09/27/2024
- **Project Objective:**
    - To optimize Parking Efficiency
    - To Automate Parking Management
    - To Enhance User Experience
    - Enable Data-Driven Decisions
    - Ensure Scalability and Flexibility
- **Assumptions:**
    - Constant demand for parking
    - Reliable real-time technology access
    - User familiarity with technology
    - Standardized parking lot layouts
    - Willingness to automate
- **Scope:**
    - Parking space management
    - Parking space listing
    - Availability and occupancy tracking
    - Parking reservation system
    - Checkout and payment processing

# Project writeup

The Parking Lot Management System is designed to modernize parking operations by leveraging technology to address inefficiencies and improve the overall user experience. This system offers a comprehensive solution for parking lot operators to manage spaces effectively and for drivers to find parking spots with ease. Key features include real-time monitoring of parking availability, automated reservation capabilities, categorized tracking of space types (e.g., standard, compact, handicapped), and accurate billing through automated payment systems. These functionalities reduce the need for manual oversight, lowering operational costs while enhancing convenience for customers.

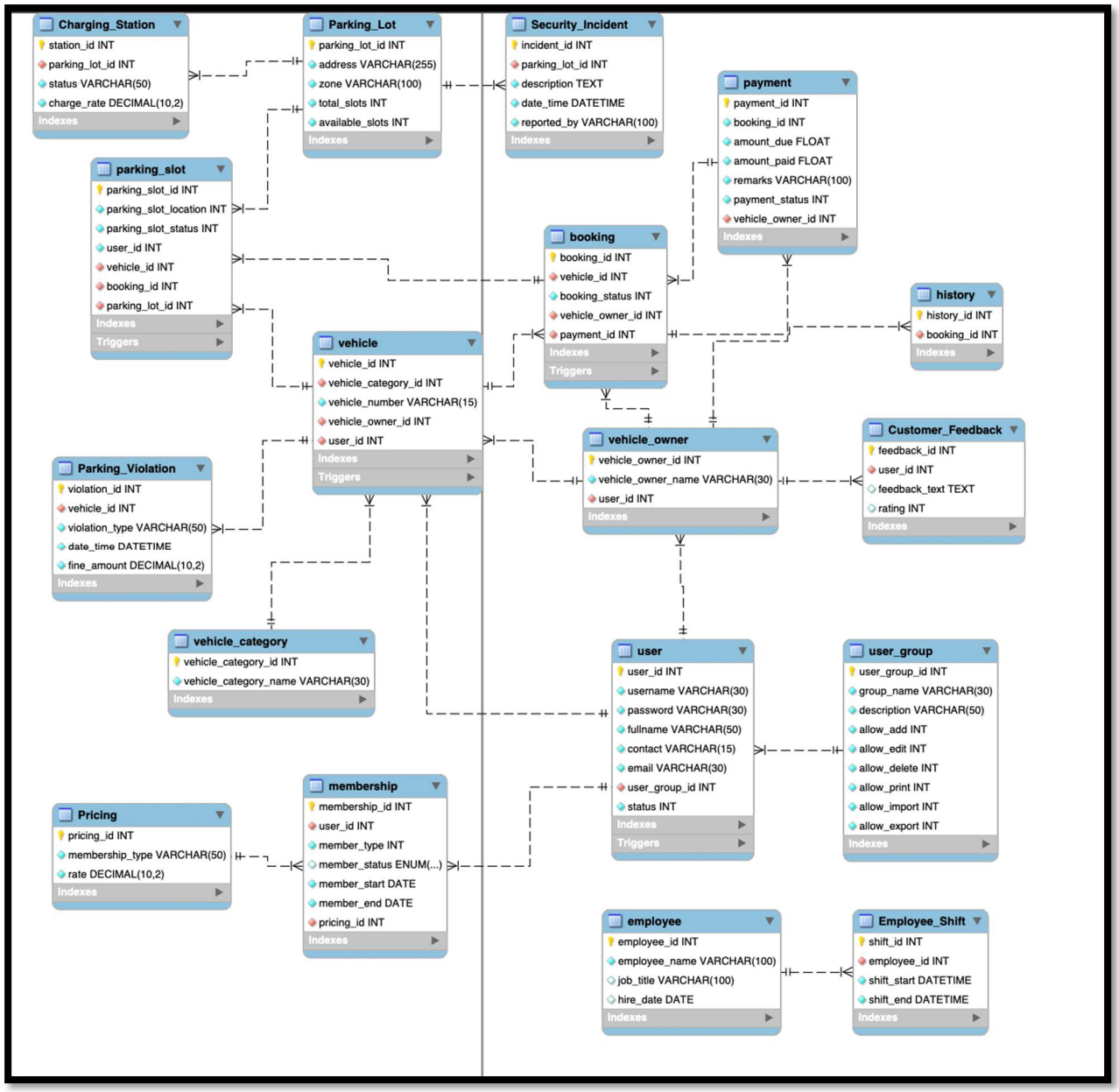
In addition to streamlining parking operations, the system provides data analytics tools to help operators make informed decisions about space utilization and predict peak usage times. Its open-source design allows for scalability and customization, making it suitable for diverse environments, from small private lots to large urban parking facilities. Integration with existing infrastructure and payment gateways ensures a smooth transition for operators adopting this system. The project includes several key milestones to ensure timely and efficient delivery of its components:

- **Project Initialization and Requirement Gathering (09/25/2024):** Completing the scoping phase, gathering requirements, and defining objectives.
- **Completion of Requirement Analysis (09/30/2024):** Finalizing the scope and preparing a detailed feature set document.
- **ER Diagram Design (10/02/2024):** Designing the entity-relationship diagram to represent the database structure.
- **System Development Phase 1 (10/07/2024):** Developing basic modules for parking space management and listing.
- **System Development Phase 2 (10/21/2024):** Implementing availability tracking, reservation, and checkout/payment functionalities.
- **Testing and Quality Assurance (10/28/2024):** Conducting rigorous testing to ensure system reliability and user satisfaction.
- **System Deployment and User Training (11/18/2024):** Deploying the final system, providing training sessions, and delivering user manuals.

The project supports the growing demand for smart parking solutions in urban areas, reducing congestion and environmental impact by minimizing the time and fuel wasted in searching for parking. By automating routine processes and offering real-time insights, the Parking Lot Management System aligns with modern trends in automation and data-driven management. This project not only improves operational efficiency and customer satisfaction but also contributes to broader smart city initiatives, making it a forward-thinking solution to a critical urban challenge.

**Impact Statement:**
The system impacts various areas, including integration with existing systems, IT infrastructure requirements, operational workflows, and data security. It offers a streamlined solution for parking management but requires customer education and additional infrastructure to support real-time operations.

# ER Diagram:



# Database Tables

Our database is made of the following tables

1. **Vehicle -** Stores vehicle information (vehicle_id, category, number, owner)
2. **vehicle_category -** Contains different categories of vehicles
3. **vehicle_owner -** Stores information about vehicle owners
4. **parking_slot -** Manages individual parking slots and their status
5. **Booking-** Handles parking space bookings
6. **Payment-** Tracks payment information for bookings
7. **User-** Stores user account information
8. **user_group-** Defines user groups and their permissions
9. **History-** Maintains booking history
10. **Membership-** Manages user memberships and their status
11. **Parking_lot-** Contains information about different parking facilities
12. **Pricing-** Defines pricing for different membership types
13. **Parking_Violation-** Records parking violations and fines
14. **Security_Incident-** Logs security incidents at parking lots
15. **Customer_Feedback-** Stores customer ratings and feedback
16. **Employee-** Contains employee information
17. **Employee_Shift-** Manages employee work shifts.
18. **Charging_Station-** Tracks electric vehicle charging stations

# Complex Queries

## 1. This Query Fetch Detailed Parking Slot and Booking Information

This query retrieves comprehensive details about occupied parking slots, providing a holistic view of parking transactions including parking lot, vehicle, owner, booking, and payment information.

```
SELECT
   parking_slot.parking_slot_id,  parking_slot.parking_slot_location,
   parking_slot.parking_slot_status, Parking_Lot.address AS parking_lot_address,
   Parking_Lot.zone AS parking_zone,
   Parking_Lot.available_slots AS parking_lot_available_slots,
   vehicle.vehicle_number AS vehicle_number,
   vehicle_owner.vehicle_owner_name AS owner_name,
   user.username AS owner_username,
   booking.booking_id, booking.booking_status,
   payment.payment_id, payment.amount_due,
   payment.amount_paid, payment.payment_status,
   CASE
      WHEN membership.member_status = 'active' THEN 'Member'
      ELSE 'Non-Member'
   END AS membership_status
FROM
   parking_slot
JOIN
   booking ON parking_slot.booking_id = booking.booking_id
JOIN
   vehicle ON booking.vehicle_id = vehicle.vehicle_id
JOIN
   vehicle_owner ON vehicle.vehicle_owner_id = vehicle_owner.vehicle_owner_id
JOIN
   user ON vehicle_owner.user_id = user.user_id
```

```
JOIN
    Parking_Lot ON parking_slot.parking_lot_id = Parking_Lot.parking_lot_id
JOIN
    payment ON booking.payment_id = payment.payment_id
LEFT JOIN
    membership ON user.user_id = membership.user_id
WHERE parking_slot.parking_slot_status = 1  -- Only occupied parking slots
ORDER BY parking_slot.parking_slot_location ASC;
```

**Result:**

| parking_slot_id | parking_slot_location | parking_slot_status | parking_lot_address | parking_zone | parking_lot_available_slots | vehicle_number | owner_name | owner_username | booking_id | booking_status | payment_id | amount_due | amount_paid | payment_status | membership_status |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **2** | 102 | 1 | 456 Elm St | Suburbs | 150 | CDE4567 | Olivia Cook | customer13 | 2 | 1 | 30 | 250 | 125 | 0 | Non-Member |
| **2** | 102 | 1 | 456 Elm St | Suburbs | 150 | CDE4567 | Olivia Cook | customer13 | 2 | 1 | 30 | 250 | 125 | 0 | Non-Member |
| **3** | 103 | 1 | 789 Oak Ave | City Center | 250 | FGH5678 | Liam Moore | customer8 | 3 | 1 | 12 | 110 | 110 | 1 | Non-Member |
| **5** | 105 | 1 | 456 Elm St | Suburbs | 150 | FGH7890 | Grace Reed | supervisor8 | 5 | 1 | 28 | 180 | 90 | 0 | Non-Member |
| **7** | 107 | 1 | 404 Cedar St | City Center | 300 | PQR2345 | Mike Ross | customer4 | 7 | 0 | 14 | 250 | 250 | 1 | Member |
| **9** | 109 | 1 | 606 Spruce St | Downtown | 80 | LMN2345 | John Doe | operator2 | 9 | 0 | 16 | 120 | 120 | 1 | Non-Member |
| **9** | 109 | 1 | 606 Spruce St | Downtown | 80 | LMN2345 | John Doe | operator2 | 9 | 0 | 16 | 120 | 120 | 1 | Non-Member |
| **11** | 111 | 1 | 808 Palm Dr | City Center | 350 | GHI1234 | Ella Roberts | supervisor6 | 11 | 0 | 24 | 300 | 150 | 0 | Non-Member |
| **11** | 111 | 1 | 808 Palm Dr | City Center | 350 | GHI1234 | Ella Roberts | supervisor6 | 11 | 0 | 24 | 300 | 150 | 0 | Non-Member |
| **13** | 113 | 1 | 1010 Fir Ave | Downtown | 90 | LMN2345 | John Doe | operator2 | 13 | 1 | 3 | 200 | 200 | 1 | Non-Member |
| **13** | 113 | 1 | 1010 Fir Ave | Downtown | 90 | LMN2345 | John Doe | operator2 | 13 | 1 | 3 | 200 | 200 | 1 | Non-Member |
| **15** | 115 | 1 | 1212 Pine Dr | City Center | 270 | STU4567 | Nina Lopez | operator4 | 15 | 1 | 13 | 160 | 160 | 1 | Member |

## 2. Query to Calculate Revenue and Violations per Parking Lot

This query calculates total violations, revenue, and payment statistics for each parking lot, enabling management to assess parking lot financial performance.

```
SELECT
   Parking_Lot.parking_lot_id,
   Parking_Lot.address AS parking_lot_address,
   Parking_Lot.zone AS parking_zone,
   COUNT(Parking_Violation.violation_id) AS total_violations,
   SUM(payment.amount_paid) AS total_revenue,
   COUNT(payment.payment_id) AS total_payments,
   ROUND(AVG(payment.amount_paid), 2) AS avg_payment_amount
FROM
   Parking_Lot
LEFT JOIN
   parking_slot ON Parking_Lot.parking_lot_id = parking_slot.parking_lot_id
LEFT JOIN
   booking ON parking_slot.booking_id = booking.booking_id
LEFT JOIN
   payment ON booking.payment_id = payment.payment_id
LEFT JOIN
   Parking_Violation ON Parking_Violation.vehicle_id = booking.vehicle_id
GROUP BY
   Parking_Lot.parking_lot_id, Parking_Lot.address, Parking_Lot.zone
ORDER BY
   total_revenue DESC;
```

## Result:

| parking_lot_id | parking_lot_address | parking_zone | total_violations | total_revenue | total_payments | avg_payment_amount |
|---|---|---|---|---|---|---|
| **1** | 123 Main St | Downtown | 3 | 640 | 4 | 160 |
| **13** | 1010 Fir Ave | Downtown | 3 | 600 | 3 | 200 |
| **8** | 505 Redwood Blvd | Industrial Area | 3 | 450 | 3 | 150 |
| **25** | 2222 Fir Blvd | Downtown | 2 | 440 | 2 | 220 |
| **6** | 303 Birch Rd | Suburbs | 3 | 360 | 3 | 120 |
| **34** | 3131 Palm Blvd | Suburbs | 2 | 340 | 2 | 170 |
| **50** | 4747 Birch Rd | Suburbs | 3 | 300 | 3 | 100 |
| **20** | 1717 Redwood St | Industrial Area | 2 | 300 | 2 | 150 |
| **29** | 2626 Maple Blvd | Downtown | 1 | 270 | 1 | 270 |
| **45** | 4242 Fir Blvd | Downtown | 2 | 260 | 2 | 130 |
| **7** | 404 Cedar St | City Center | 1 | 250 | 1 | 250 |
| **17** | 1414 Maple Ave | Downtown | 0 | 250 | 1 | 250 |
| **23** | 2020 Palm Rd | City Center | 2 | 250 | 2 | 125 |
| **47** | 4444 Pine Ave | City Center | 0 | 240 | 1 | 240 |
| **28** | 2525 Oak Blvd | Industrial Area | 0 | 240 | 1 | 240 |
| **26** | 2323 Chestnut Ave | Suburbs | 1 | 220 | 1 | 220 |
| **10** | 707 Willow Ln | Suburbs | 0 | 220 | 1 | 220 |
| **12** | 909 Sequoia St | Industrial Area | 1 | 220 | 1 | 220 |
| **2** | 456 Elm St | Suburbs | 2 | 215 | 2 | 107.5 |
| **18** | 1515 Birch Blvd | Suburbs | 0 | 200 | 1 | 200 |
| **21** | 1818 Spruce Ave | Downtown | 0 | 200 | 1 | 200 |
| **39** | 3636 Maple Blvd | City Center | 1 | 200 | 1 | 200 |
| **43** | 4040 Redwood Rd | City Center | 1 | 190 | 1 | 190 |
| **42** | 3939 Sequoia Blvd | Suburbs | 1 | 190 | 1 | 190 |
| **16** | 1313 Oak St | Industrial Area | 1 | 180 | 1 | 180 |
| **14** | 1111 Chestnut Rd | Suburbs | 1 | 180 | 1 | 180 |
| **46** | 4343 Chestnut Rd | Suburbs | 0 | 180 | 1 | 180 |
| **44** | 4141 Palm Ave | Industrial Area | 1 | 170 | 1 | 170 |
| **15** | 1212 Pine Dr | City Center | 1 | 160 | 1 | 160 |
| **40** | 3737 Birch Blvd | Industrial Area | 1 | 160 | 1 | 160 |
| **41** | 3838 Cedar Rd | Downtown | 1 | 150 | 1 | 150 |

### 3. Vehicle Owners with the Highest Parking Violations

This query identifies vehicle owners with more than three violations, helping track problematic vehicle owners and potential repeat offenders.

```
SELECT
   vo.vehicle_owner_name,
   vo.user_id,
   COUNT(pv.violation_id) AS total_violations
FROM
   vehicle_owner vo
JOIN
   vehicle v ON vo.vehicle_owner_id = v.vehicle_owner_id
LEFT JOIN
   Parking_Violation pv ON v.vehicle_id = pv.vehicle_id
GROUP BY
   vo.vehicle_owner_name, vo.user_id
HAVING
   total_violations > 3  -- Only show owners with more than 3 violations
ORDER BY
   total_violations DESC;
```

**Result**:

| vehicle_owner_name | user_id | total_violations |
|---|---|---|
| **Jane Smith** | 3 | 4 |
| **Bob Johnson** | 5 | 4 |
| **John Doe** | 6 | 4 |

**4. This query gives the count of Vehicles by Category**

This query counts total vehicles in each vehicle category, providing insights into vehicle type distribution and helping management understand the composition of vehicles using the facilities.

```
SELECT
   vc.vehicle_category_name,
   COUNT(v.vehicle_id) AS total_vehicles
FROM vehicle v
JOIN
   vehicle_category vc ON v.vehicle_category_id = vc.vehicle_category_id
GROUP BY vc.vehicle_category_name
ORDER BY total_vehicles DESC;
```

**Result:**

| vehicle_category_name | total_vehicles |
|---|---|
| **Car** | 5 |
| **Motorcycle** | 4 |
| **Truck** | 4 |
| **Van** | 4 |
| **Convertible** | 4 |
| **Electric Vehicle** | 3 |
| **Bicycle** | 3 |
| **Scooter** | 3 |
| **SUV** | 3 |
| **Pickup Truck** | 3 |
| **Minivan** | 3 |
| **ATV** | 3 |
| **Tractor** | 3 |
| **Golf Cart** | 3 |
| **Bus** | 2 |

**5. This query finds the least used parking lot**

This query finds the parking lot with the lowest number of bookings, allowing management to investigate reasons for low utilization and potentially optimize or repurpose the facility.

```
SELECT
   Parking_Lot.parking_lot_id,
   Parking_Lot.address,
   COUNT(parking_slot.parking_slot_id) AS total_bookings
FROM
   Parking_Lot
   LEFT JOIN parking_slot ON Parking_Lot.parking_lot_id = parking_slot.parking_lot_id
GROUP BY
   Parking_Lot.parking_lot_id, Parking_Lot.address
ORDER BY
   total_bookings ASC
LIMIT 1;
```

**Result:**

| parking_lot_id | address | total_bookings |
|---|---|---|
| **4** | 101 Pine Blvd | 0 |

6. **This query finds the top 3 vehicle owners with the highest total amount paid, along with their total amount paid**

This query identifies highest-paying vehicle owners, offering valuable insights into revenue generation and potential high-value customers for targeted marketing or loyalty programs.

```
SELECT
   vehicle_owner.vehicle_owner_name,
   SUM(payment.amount_paid) AS total_amount_paid
FROM
   vehicle_owner
   INNER JOIN booking ON vehicle_owner.vehicle_owner_id = booking.vehicle_owner_id
   INNER JOIN payment ON booking.payment_id = payment.payment_id
GROUP BY
   vehicle_owner.vehicle_owner_name
ORDER BY
   total_amount_paid DESC
LIMIT 3;
```

**Result:**

| vehicle_owner_name | total_amount_paid |
|---|---|
| **Jackson Turner** | 270 |
| **Sophia Harris** | 250 |
| **Benjamin Sanders** | 250 |

### 7. Query to Find Available Parking Slots by Category and Occupancy Status

This query provides a count of available and occupied slots per vehicle category, helping manage parking slot allocation and understand utilization patterns across different vehicle types.

```
SELECT vc.vehicle_category_name,
    SUM(CASE WHEN ps.parking_slot_status = 0 THEN 1 ELSE 0 END) AS available_slots,
    SUM(CASE WHEN ps.parking_slot_status = 1 THEN 1 ELSE 0 END) AS occupied_slots
FROM parking_slot ps
JOIN vehicle v ON ps.user_id = v.user_id
JOIN vehicle_category vc ON v.vehicle_category_id = vc.vehicle_category_id
GROUP BY vc.vehicle_category_name;
```

**Result:**

| vehicle_category_name | available_slots | occupied_slots |
|---|---|---|
| **Car** | 4 | 2 |
| **Truck** | 3 | 1 |
| **Motorcycle** | 2 | 3 |
| **Bicycle** | 1 | 3 |
| **Scooter** | 1 | 2 |
| **Van** | 0 | 4 |
| **SUV** | 2 | 1 |
| **Convertible** | 1 | 3 |
| **Minivan** | 1 | 2 |
| **Pickup Truck** | 1 | 2 |
| **ATV** | 3 | 0 |
| **Electric Vehicle** | 0 | 2 |
| **Golf Cart** | 2 | 1 |
| **Tractor** | 0 | 3 |
| **Bus** | 2 | 0 |

## 8. Query to Retrieve Most Frequent Users of Parking Slots

This query identifies top users, useful for understanding user behavior, developing customer profiles, and potentially creating personalized services or loyalty programs.

```
SELECT u.user_id,
    u.username,
    COUNT(ps.parking_slot_id) AS total_parking_events
FROM user u
JOIN vehicle v ON u.user_id = v.user_id
JOIN parking_slot ps ON v.vehicle_id = ps.parking_slot_id
GROUP BY u.user_id, u.username
ORDER BY total_parking_events DESC
LIMIT 10;
```

**Result**:

| user_id | username | total_parking_events |
|---|---|---|
| **6** | operator2 | 3 |
| **22** | security5 | 2 |
| **3** | customer1 | 2 |
| **5** | customer2 | 2 |
| **15** | security3 | 2 |
| **9** | customer3 | 2 |
| **14** | customer5 | 2 |
| **12** | operator4 | 2 |
| **13** | supervisor2 | 2 |
| **50** | security12 | 1 |

## 9. Query for User Behavior and Vehicle Type Analysis

```
SELECT
    u.user_id,
    u.fullname,
    COUNT(DISTINCT v.vehicle_id) AS owned_vehicles,
    COUNT(DISTINCT b.booking_id) AS total_bookings,
    AVG(cf.rating) AS avg_feedback_rating
FROM  user u
JOIN user_group ug ON u.user_group_id = ug.user_group_id
LEFT JOIN vehicle_owner vo ON u.user_id = vo.user_id
LEFT JOIN vehicle v ON vo.vehicle_owner_id = v.vehicle_owner_id
LEFT JOIN booking b ON vo.vehicle_owner_id = b.vehicle_owner_id
LEFT JOIN Customer_Feedback cf ON u.user_id = cf.user_id
WHERE ug.group_name = 'Customer'
GROUP BY u.user_id
HAVING total_bookings > 0
ORDER BY total_bookings DESC, avg_feedback_rating DESC;
```

**Result:**

| user_id | fullname | owned_vehicles | total_bookings | avg_feedback_rating |
|---|---|---|---|---|
| **3** | Jane Smith | 2 | 1 | 5.0000 |
| **11** | Nina Lopez | 1 | 1 | 5.0000 |
| **28** | Amelia Martinez | 1 | 1 | 5.0000 |
| **32** | Zoe Perez | 1 | 1 | 5.0000 |
| **9** | Lily Thomas | 2 | 1 | 4.0000 |
| **17** | Jacob Young | 2 | 1 | 4.0000 |
| **21** | Harper Nelson | 1 | 1 | 4.0000 |
| **39** | Ethan Murphy | 0 | 1 | 4.0000 |
| **45** | Oliver Rivera | 1 | 1 | 4.0000 |
| **5** | Bob Johnson | 2 | 1 | 3.0000 |
| **25** | Mason Martin | 2 | 1 | 3.0000 |
| **36** | Noah Evans | 0 | 1 | 3.0000 |
| **42** | Grace Reed | 1 | 1 | 3.0000 |
| **48** | Michael Hughes | 1 | 1 | 3.0000 |
| **14** | Lucas Walker | 0 | 1 | 2.0000 |

**10. Query for Security Incidents with Parking Lot Details**

```
SELECT si.incident_id, si.description, si.date_time, si.reported_by,
    pl.address, pl.zone
FROM Security_Incident si
JOIN Parking_Lot pl ON si.parking_lot_id = pl.parking_lot_id
LIMIT 10;
```

**Result:**

| incident_id | description | date_time | reported_by | address | zone |
|---|---|---|---|---|---|
| **1** | Suspicious activity near entrance | 2023-01-15 18:30:00 | John Doe | 123 Main St | Downtown |
| **2** | Attempted vehicle break-in | 2023-02-12 20:45:00 | Jane Smith | 456 Elm St | Suburbs |
| **3** | Vandalism in parking lot | 2023-03-22 11:10:00 | Emily Davis | 789 Oak Ave | City Center |
| **4** | Unauthorized vehicle in reserved spot | 2023-04-05 14:00:00 | Michael Brown | 101 Pine Blvd | Industrial Area |
| **5** | Fight reported between two drivers | 2023-04-20 16:30:00 | Sarah Wilson | 202 Maple Dr | Downtown |
| **6** | Loitering near parking area | 2023-05-05 19:20:00 | Chris Lee | 303 Birch Rd | Suburbs |
| **7** | Suspicious package found | 2023-06-01 09:15:00 | Laura Kim | 404 Cedar St | City Center |
| **8** | Vehicle fire incident | 2023-06-15 08:45:00 | Daniel Garcia | 505 Redwood Blvd | Industrial Area |
| **9** | Unauthorized vendor solicitation | 2023-07-10 10:30:00 | James Scott | 606 Spruce St | Downtown |
| **10** | Vehicle hit-and-run reported | 2023-08-02 13:15:00 | Sophia Turner | 707 Willow Ln | Suburbs |

**11. Query for Employee Shifts with Performance Metrics**

This query provides details about employee shifts and basic information, helping track employee work patterns and potentially supporting workforce management strategies.

```
SELECT es.shift_id, es.shift_start, es.shift_end,
    e.employee_name, e.job_title, e.hire_date
FROM Employee_Shift es
JOIN employee e ON es.employee_id = e.employee_id
LIMIT 10;
```

**Result**:

| shift_id | shift_start | shift_end | employee_name | job_title | hire_date |
|---|---|---|---|---|---|
| **1** | 2023-10-01 08:00:00 | 2023-10-01 16:00:00 | Emily Davis | Security Officer | 2023-03-05 |
| **2** | 2023-10-01 09:00:00 | 2023-10-01 17:00:00 | Mark Lee | Parking Manager | 2020-09-10 |
| **3** | 2023-10-01 10:00:00 | 2023-10-01 18:00:00 | Anna Thompson | Technician | 2021-05-25 |
| **4** | 2023-10-01 11:00:00 | 2023-10-01 19:00:00 | John Doe | Customer Service Representative | 2022-07-14 |
| **5** | 2023-10-01 12:00:00 | 2023-10-01 20:00:00 | Sophia Kim | Maintenance Worker | 2021-11-23 |
| **6** | 2023-10-01 13:00:00 | 2023-10-01 21:00:00 | Michael Brown | Security Supervisor | 2019-12-12 |
| **7** | 2023-10-01 14:00:00 | 2023-10-01 22:00:00 | Sarah Wilson | Parking Attendant | 2022-02-18 |
| **8** | 2023-10-01 15:00:00 | 2023-10-01 23:00:00 | David Green | IT Specialist | 2020-03-30 |
| **9** | 2023-10-01 16:00:00 | 2023-10-01 00:00:00 | Olivia Johnson | Operations Manager | 2018-06-04 |
| **10** | 2023-10-01 17:00:00 | 2023-10-01 01:00:00 | Liam Garcia | Technician | 2023-01-15 |

**12. Vehicle Owners with Frequent Violations and Their Booking History.**

This query identifies vehicle owners with multiple parking violations and provides their booking history:

```
SELECT
    vo.vehicle_owner_id,
    vo.vehicle_owner_name,
    COUNT(DISTINCT pv.violation_id) AS violation_count,
    GROUP_CONCAT(DISTINCT pv.violation_type) AS violation_types,
    SUM(pv.fine_amount) AS total_fines,
    COUNT(DISTINCT b.booking_id) AS total_bookings,
    AVG(p.amount_paid) AS average_payment
FROM
    vehicle_owner vo
JOIN
    vehicle v ON vo.vehicle_owner_id = v.vehicle_owner_id
LEFT JOIN
    Parking_Violation pv ON v.vehicle_id = pv.vehicle_id
LEFT JOIN
    booking b ON vo.vehicle_owner_id = b.vehicle_owner_id
LEFT JOIN
    payment p ON b.payment_id = p.payment_id
GROUP BY
    vo.vehicle_owner_id, vo.vehicle_owner_name
HAVING
    violation_count > 1
ORDER BY
    violation_count DESC, total_fines DESC
LIMIT 10;
```

**Result:**

| vehicle_owner_id | vehicle_owner_name | violation_count | violation_types | total_fines | total_bookings | average_payment |
|---|---|---|---|---|---|---|
| **2** | Bob Johnson | 4 | Blocking Pedestrian Access,Illegal Parking,No Parking Permit | 410 | 1 | 150 |
| **1** | Jane Smith | 4 | Fire Lane Violation,Loading Zone Violation,Overstay | 355 | 1 | 100 |
| **3** | John Doe | 4 | Exceeding Time Limit,Expired Meter,Unpaid Parking | 220 | 1 | 200 |
| **44** | William Price | 3 | Improper Parking,Parking in a Handicap Zone | 370 | 1 | 200 |
| **16** | Ryan Adams | 3 | No License Plate,No Parking Permit | 270 | 1 | 120 |
| **5** | Alice Brown | 2 | Failure to Pay Fee,Fire Lane Violation | 285 | 1 | 180 |
| **10** | Olivia Green | 2 | Blocking Driveway,Unpaid Citation | 215 | 1 | 170 |
| **45** | Sophia Gray | 2 | Reserved Space Violation,Unauthorized Area | 205 | 1 | 150 |
| **6** | Charlie Wilson | 2 | Blocking Entrance,Double Parking | 185 | 1 | 125 |
| **38** | Benjamin Sanders | 2 | Expired Meter,Parking on Grass | 85 | 1 | 250 |

**13. Parking Lot Utilization and Revenue Analysis**

This query analyzes the utilization and revenue of parking lots, including violation data:

```
SELECT
    pl.parking_lot_id,
    pl.address,
    pl.zone,
    COUNT(DISTINCT b.booking_id) AS total_bookings,
    AVG(p.amount_paid) AS average_payment,
    SUM(p.amount_paid) AS total_revenue,
    COUNT(DISTINCT pv.violation_id) AS violation_count,
    SUM(pv.fine_amount) AS total_fines,
    (pl.total_slots - pl.available_slots) / pl.total_slots * 100 AS occupancy_rate,
    AVG(cf.rating) AS average_rating
FROM Parking_Lot pl
LEFT JOIN
    parking_slot ps ON pl.parking_lot_id = ps.parking_lot_id
LEFT JOIN
    booking b ON ps.booking_id = b.booking_id
LEFT JOIN
    payment p ON b.payment_id = p.payment_id
LEFT JOIN
    vehicle v ON b.vehicle_id = v.vehicle_id
LEFT JOIN
    Parking_Violation pv ON v.vehicle_id = pv.vehicle_id
LEFT JOIN
    Customer_Feedback cf ON b.vehicle_owner_id = cf.user_id
GROUP BY pl.parking_lot_id, pl.address, pl.zone, pl.total_slots, pl.available_slots
HAVING total_bookings > 0
ORDER BY total_revenue DESC, occupancy_rate DESC
LIMIT 10;
```

**Result:**

| parking_lot_id | address | zone | total_bookings | average_payment | total_revenue | violation_count | total_fines | occupancy_rate | average_rating |
|---|---|---|---|---|---|---|---|---|---|
| **1** | 123 Main St | Downtown | 2 | 160 | 640 | 3 | 270 | 50 | 3.25 |
| **13** | 1010 Fir Ave | Downtown | 1 | 200 | 600 | 3 | 195 | 30.7692 | 5 |
| **8** | 505 Redwood Blvd | Industrial Area | 1 | 150 | 450 | 3 | 330 | 25 | 2 |
| **25** | 2222 Fir Blvd | Downtown | 1 | 220 | 440 | 2 | 205 | 35.7143 | 4 |
| **6** | 303 Birch Rd | Suburbs | 1 | 120 | 360 | 3 | 370 | 33.3333 | 4 |
| **34** | 3131 Palm Blvd | Suburbs | 1 | 170 | 340 | 2 | 185 | 22.7273 | 2 |
| **20** | 1717 Redwood St | Industrial Area | 1 | 150 | 300 | 2 | 215 | 23.8095 | 5 |
| **50** | 4747 Birch Rd | Suburbs | 1 | 100 | 300 | 3 | 155 | 20 | 5 |
| **29** | 2626 Maple Blvd | Downtown | 1 | 270 | 270 | 1 | 100 | 31.25 | 5 |
| **45** | 4242 Fir Blvd | Downtown | 1 | 130 | 260 | 2 | 285 | 45.4545 | 2 |

# Stored procedures

**1. CalculateMembershipDiscount**

**/* This procedure calculates the final membership discount rate based on the user's membership status. */**

```
DELIMITER //
CREATE PROCEDURE CalculateMembershipDiscount(
   IN p_user_id INT,
   OUT p_final_rate DECIMAL(10, 2)
)
BEGIN
   DECLARE pricing_id INT;
   DECLARE discount_percentage DECIMAL(5, 2);
   DECLARE standard_rate DECIMAL(10, 2);
   -- Set the standard price for non-members
   SET standard_rate = 300.00;
   -- Check if the user has an active membership
   IF EXISTS (
      SELECT 1 FROM membership M
      WHERE M.user_id = p_user_id AND M.member_status = 'active'
   ) THEN
      -- Get the pricing_id for the user
      SELECT M.pricing_id INTO pricing_id FROM membership M
      WHERE M.user_id = p_user_id AND M.member_status = 'active';
      -- Set the discount percentage based on pricing_id
      CASE pricing_id
         WHEN 1 THEN SET discount_percentage = 0.30; -- Basic (30% discount)
         WHEN 2 THEN SET discount_percentage = 0.40; -- Premium (40% discount)
         WHEN 3 THEN SET discount_percentage = 0.50; -- VIP (50% discount)
         WHEN 4 THEN SET discount_percentage = 0.60; -- Corporate (60% discount)
         ELSE SET discount_percentage = 0;          -- No discount if no valid pricing_id
```

```
        END CASE;

    -- Calculate the discounted rate from the standard rate
        SET p_final_rate = standard_rate * (1 - discount_percentage);
    -- Return user_id and the final discounted rate directly
        SELECT p_user_id AS user_id, p_final_rate AS final_rate;
    ELSE
        -- If no active membership, set the rate to standard price
        SET p_final_rate = standard_rate;
        -- Return user_id and the standard rate directly
        SELECT p_user_id AS user_id, p_final_rate AS final_rate;
    END IF;
END //
DELIMITER ;
```

**Result:**

```
CALL CalculateMembershipDiscount(3, @final_rate);
```

| user_id | final_rate |
|---|---|
| **3** | 210.00 |

**2. CalculateParkingAvailability**

**/* This procedure calculates the number of available parking slots, filled parking slots, and available charging stations for a given parking lot. */**

```
DELIMITER //
CREATE PROCEDURE CalculateParkingAvailability(
   IN ParkingLotID INT,
   OUT AvailableSlots INT,
   OUT FilledSlots INT,
   OUT AvailableChargingStations INT )
BEGIN
   -- Get available slots for the specific parking lot
   SELECT available_slots INTO AvailableSlots
   FROM parking_lot
   WHERE parking_lot_id = ParkingLotID;
   -- Calculate filled slots for the specific parking lot
   SELECT (total_slots - available_slots) INTO FilledSlots
   FROM parking_lot
   WHERE parking_lot_id = ParkingLotID;
   -- Calculate available charging stations for the specific parking lot
   SELECT COUNT(*) INTO AvailableChargingStations
   FROM Charging_Station
   WHERE parking_lot_id = ParkingLotID AND status = 'Available';
   -- Return the parking lot ID along with the calculated details
   SELECT ParkingLotID AS parking_lot_id,
        AvailableSlots,
        FilledSlots,
        AvailableChargingStations;
END//
DELIMITER ;
```

**Result:**

CALL CalculateParkingAvailability(30, @AvailableSlots, @FilledSlots,

@AvailableChargingStations);

| parking_lot_id | AvailableSlots | FilledSlots | AvailableChargingStations |
|---|---|---|---|
| **30** | 160 | 50 | 1 |

**3. CheckParkingViolations**

**/* This stored procedure checks the number of parking violations for a given vehicle. If the vehicle has more than 3 violations, it bans the vehicle from parking; otherwise, it returns the count of violations. */**

```
DELIMITER //
CREATE PROCEDURE CheckParkingViolations(IN vehicle_id_param INT)
BEGIN
   DECLARE violation_count INT;
   -- Calculate the number of violations for the given vehicle
   SELECT COUNT(*) INTO violation_count
   FROM parking_violation
   WHERE vehicle_id = vehicle_id_param;  -- Use the correct input parameter name
   -- Check if the vehicle has exceeded the violation limit (3 violations)
   IF violation_count > 3 THEN
      -- Vehicle is banned due to excessive violations
      SELECT CONCAT('Vehicle ID ', vehicle_id_param, ' is banned from parking due to ',
violation_count, ' excessive violation(s).') AS ban_status;
   ELSE
      -- Vehicle has fewer violations, not banned
      SELECT CONCAT('Vehicle ID ', vehicle_id_param, ' has ', violation_count, ' violation(s)
and is not banned.') AS violation_status;
   END IF;
END //
DELIMITER ;
```

**Result:**

```
Call CheckParkingViolations(1);
```

| violation_status |
| --- |
| Vehicle ID 1 has 3 violation(s) and is not banned. |

# Functions

1. **Function to see available Parking slots**

```
DELIMITER //
CREATE FUNCTION AvailableParkingSlots(ParkingLotID INT)
RETURNS INT
READS SQL DATA
BEGIN
   DECLARE TotalSlots INT;
   DECLARE OccupiedSlots INT;
   SELECT total_slots INTO TotalSlots FROM Parking_Lot
   WHERE parking_lot_id = ParkingLotID;
   SELECT COUNT(*) INTO OccupiedSlots FROM parking_slot
   WHERE parking_lot_id = ParkingLotID AND parking_slot_status = 1;
   RETURN TotalSlots - OccupiedSlots;
END//
DELIMITER ;
```

**Result:**

Select AvailableParkingSlots(1);

| ParkingLotID | AvailableParkingSlots |
|---|---|
| 1 | 100 |

**2. Function to identify the memebership status of a person**

```
DELIMITER //
CREATE FUNCTION GetMembershipStatus (UserID INT)
RETURNS VARCHAR(20)
READS SQL DATA
BEGIN
```

```
    DECLARE Status VARCHAR(20);
    SELECT member_status INTO Status FROM membership WHERE user_id = UserID;
    RETURN Status;
END//
DELIMITER ;
```

**Result:**

Select GetMembershipStatus(2);

| UserID | GetMembershipStatus |
|---|---|
| 2 | pending |

**3. Function to assign an empty parking slot to a person**

```
DELIMITER //
CREATE FUNCTION AssignParkingSlot (ParkingLotID INT)
RETURNS INT
READS SQL DATA
BEGIN
   DECLARE AvailableSlot INT;
   SELECT parking_slot_id INTO AvailableSlot FROM parking_slot
   WHERE parking_lot_id = ParkingLotID AND parking_slot_status = 0
   LIMIT 1;
 RETURN AvailableSlot;
END//
DELIMITER ;
```

**Result:**

Select AssignParkingSlot(1);

| ParkingLotID | AssignParkingSlot |
|---|---|
| 1 | 1 |

# Triggers

1. **To prevent duplicate vehicle numbers:**

```
DELIMITER $$
CREATE TRIGGER before_vehicle_insert
BEFORE INSERT ON vehicle FOR EACH ROW
BEGIN
IF EXISTS (SELECT 1 FROM vehicle WHERE vehicle_number = NEW.vehicle_number)
THEN
SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Vehicle number already exists!';
END IF;
END$$
DELIMITER ;
```

**Result:**

```
INSERT INTO VEHICLE VALUE (2, 2, 'XYZ5678', 2, 5);
```

Error Code: 1644. Vehicle number already exists!

2. **To update parking slot status**

```
DELIMITER $$
CREATE TRIGGER after_parking_slot_update
AFTER UPDATE ON parking_slot FOR EACH ROW
BEGIN
   IF NEW.parking_slot_status != OLD.parking_slot_status THEN
      INSERT INTO parking_slot_log (parking_slot_id, old_status, new_status, log_time)
      VALUES (NEW.parking_slot_id, OLD.parking_slot_status, NEW.parking_slot_status,
NOW());
   END IF;
END$$
DELIMITER ;
```

3. **To mark the parking space as available when a customer checks out and updates the available spots in the lot**

```
DELIMITER //
CREATE TRIGGER FreeUpSpace
AFTER INSERT ON parking_slot
FOR EACH ROW
BEGIN
  IF NEW.parking_slot_status = 0 THEN
    UPDATE Parking_lot
    SET available_slots = available_slots + 1
    WHERE parking_lot_id = NEW.parking_lot_id;
  END IF;
END//
DELIMITER ;
```

4. **To make sure the user choose a unique username**

```
DELIMITER //
CREATE TRIGGER UniqueUserName
AFTER INSERT ON user
FOR EACH ROW
BEGIN
  IF EXISTS (SELECT 1 FROM user WHERE username = NEW.username) THEN
    SIGNAL SQLSTATE '45000'
    SET MESSAGE_TEXT = 'Username already exists. Please choose a different username.';
  END IF;
END //
DELIMITER ;
```

# Conclusion

The **Parking Lot Management System** is a transformative solution designed to address inefficiencies in parking operations and enhance the user experience. By leveraging real-time monitoring, automation, and advanced analytics, the system provides a seamless way for parking lot operators to manage spaces and for drivers to find available spots quickly and conveniently. Its scalability and open-source design ensure adaptability to a wide range of parking lot sizes and layouts, making it suitable for diverse environments such as malls, offices, and public facilities.

This project not only optimizes space utilization and reduces operational costs but also contributes to broader goals such as minimizing congestion, reducing environmental impact, and aligning with smart city initiatives. With its focus on automation and data-driven decision-making, the system paves the way for modernized parking management that benefits both operators and customers. By addressing operational, technical, and customer-facing challenges, the Parking Lot Management System offers a forward-thinking, efficient, and sustainable solution to a critical urban need.