CSC 540 Database Management Systems

Wolf Parking Management System

Project Report 1

26th September 2023

Team Members:

Aravinda Raman Jatavallabha

Prathyusha Kodali

Suraj Raghu Kumar

Yuktasree Muppala

Assumptions:

1. Vehicles are limited to one parking lot per visit and this will not change throughout their visit.
2. One driver can be associated with one or two vehicles.
3. A driver can hold multiple permits.
4. Drivers can have multiple citations associated with them.
5. One citation is associated with only violation at a time.
6. A vehicle can be issued only one permit at a time; with exactly one permit type (“residential”, “commuter”, “peak hours”, “special event”, and “Park & Ride”).
7. A handicapped person should only be assigned to a handicap space in the parking space, regardless of the vehicle type (eg. “electric”, “handicap”, “compact car”, or “regular).
8. A driver can be assigned to only one status at a time (eg. ‘S’, ‘E’, or ‘V’ depending on whether a student, or an employee or a visitor). For instance, a student working on campus will still be considered as a student and not an employee.
9. In the case of a student or an employee, UnivID is the value for the ID attribute for a Driver. Whereas, in the case of a Visitor, their phone number will be considered as a value for the ID.
10. Zones are included in the Parking Lot and within each zone there are dedicated parking spaces for vehicles.
11. A driver can submit at most one appeal per citation.

1. Problem Statement:

This problem involves designing the Wolf Parking Management System, a comprehensive database solution for efficiently managing driver information, parking lot information, zone information, space information, vehicle information, and citation information on a university campus.

The system performs four significant types of tasks, each comprising various operations: *Information processing*, which involves updating driver, parking lot, zone, space, and permit information, along with citation management; *maintaining permits and vehicle details for each drive*r, involving permit assignments based on

driver status, and managing vehicle ownership information. It also handles *generating and maintaining citations*, checking permit validity, and allowing drivers to pay or appeal citations. Additionally, the system *generates reports*, including citation and zone-related reports, car violation counts, employee permit statistics, and permit and space lookup functions.

A database system is a wiser choice than depending on a simple file structure because many people will view and change the data simultaneously. Employing a database and running data queries gives a more practical option for searching through files containing obsolete details, given the regular input and updates of significant data volumes. The necessity of re-entering complete data files or requiring system users to reload files before utilization could be more efficient. Thus, employing a simple set of files is impractical for the Wolf Parking Management System. Instead, opting for a database system eradicates the risk of users inadvertently overwriting each other's updates and prevents drivers from submitting incomplete information. This safeguard ensures data integrity and accuracy within the Wolf Parking System, eliminating the possibility of data loss or discrepancies.

2. Intended Users:

Administrators of the Parking System:

- Add Parking Lots: Administrators can add new parking lots to the system.
- Assign Zones and Spaces: They can assign specific zones and spaces within the parking lots.
- Assign Parking Permits: Administrators can assign parking permits to drivers based on their status (Student, Employee, or Visitor).
- Change Space Availability: Administrators can change the availability status of parking spaces.
- Verify Permit Validity: Administrators can check if a car has a valid permit to park in a particular lot

Drivers (Students, Employees, Visitors):

- Obtain and Display Permits: Drivers must obtain the necessary permits to park in specific zones and spaces designated for their status.
- Ensure Correct Permit Type: Drivers must ensure their permits match the appropriate zone and space types (e.g., electric car spaces).

Security Personnel:

- Citation Management: Security personnel create, update, and delete citations for vehicles that violate parking regulations.
- Payment Processing: They handle the processing of citation payments, updating payment status from unpaid to paid through a payment procedure.

These users collectively contribute to the efficient management of the parking system. Additionally, it is essential to note that students and visitors have certain limitations on the number of vehicles they can have on a permit, while employees have different allowances. Furthermore, special event permits and 'Park & Ride' permits are available to students and employees as additional options.

3. Five Main Entities:

1) Driver - driverID, name, status
2) Vehicle - licenseNum, color, model, year, manufacturer
3) Security - securityID
4) Permit - permitID, permitType, spaceType, startDate, expirationDate, expirationTime
5) Parking Lot - lotID, name, address, category

4. Tasks and Operations- Realistic Situations:

- **<u>Situation 1</u>:** John, an employee, enters the WolfParking lot in his electric car, and his university ID and name are entered into the database. He is then assigned a permit, which has all the information about the parking lot, vehicle license number, and his permit type, which is residential. Finally, he parks his car in the assigned parking lot in Zone ‘A’ in the ‘electric’ space.

- **<u>Situation 2</u>:** A driver named Steffen parks his vehicle, a red 2023 Honda Civic, at the WolfParking lot. Upon returning, he noticed a $30 citation for an expired permit on his windshield. John appeals for security verification of the citation, and after it is confirmed to be valid, he promptly settles the payment. The security officer updates the payment status, effectively managing the citation process and ensuring accurate record-keeping.

5. Application Programming Interfaces:

Information Processing

- addDriverInfo(driverID, name, status):
    - Description: Adds or updates basic information about drivers.
    - Returns True if added/updated successfully, False otherwise.

- deleteDriveInfo(DriverID):
    - Description: Deletes driver information.
    - Returns True if deleted successfully, False otherwise.

- addParkingLotInfo(lotID, category, address, name):
    - Description: Adds parking lot information.
    - Returns True if added/updated successfully, False otherwise.

- deleteParkingLotInfo(lotID):
    - Description: Deletes parking lot information.
    - Returns True if deleted successfully, False otherwise.

- addZoneInfo(zoneID, lotID):
    - Description :Adds specified zones to lots.
    - Returns True if added successfully, False otherwise.

- deleteZoneInfo(zoneID):
    - Deletes zone information.
    - Returns True if deleted successfully, False otherwise.

- addSpaceInfo(spaceNum, zoneID, lotID, availabilityStatus)
    - Adds or updates space information to zones.
    - Returns True if added successfully, False otherwise.

- deleteSpaceInfo(spaceNum, zoneID, lotID):
    - Deletes space information from zones.
    - Returns True if deleted successfully, False otherwise

- checkPermitValidity(licenseNum, lotID):
    - Checks if a car has a valid permit for the specified lot.
    - Returns True if valid, False otherwise.

- addVehicleOwnership(licenseNum, driverID):
    - Checks if a vehicle is owned by the specified driver.
    - Returns True if owned, False otherwise.

Maintaining Permits and Vehicle Information

- assignPermitToDriver(driverID, permitID):
    - Assigns a permit to a driver.
    - Returns True if assigned successfully, False otherwise.

- updatePermitInfo(PermitID, PermitType, ExpirationTime, ExpirationDate, StartDate, SpaceType, Lot):
    - Updates permit information.
    - Returns True if added/updated successfully, False otherwise.

- updateVehicleInfo(licenseNum, color, manufacturer, model, year):
    - Updates vehicle information.
    - Returns True if added/updated successfully, False otherwise.

- addVehicleInfo(licenseNum):
    - Removes vehicles from the database.
    - Returns True if updated successfully, False otherwise.

- removeVehicleInfo(licenseNum):
    - Removes vehicle information from the database.
    - Returns True if updated successfully, False otherwise.

Generating and Maintaining Citations

- generateCitation(driverID, lotID, zoneID, spaceNum, category):
    - Generates a citation.
    - Returns True if generated successfully, False otherwise.

- settleCitation(driverID, citationNum):
    - Pays a citation.
    - Returns True if paid successfully, False otherwise.

- appealCitation(driverID, citationNum):
    - Appeals a citation.
    - Returns True if appealed successfully, False otherwise.

- updateCitationInfo(citationNum, paymentStatus, fee, lot, category):
    - Updates citation information.
    - Returns True if updated successfully, False otherwise.

- deleteCitation(citationNum):
    - Deletes a citation.
    - Returns True if deleted successfully, False otherwise.

- getCitationsForDriver(driverID):
    - Returns a list of citations associated with the specified driver.

Reports

- generateCitationReport(timeRange, lotID):
    - Returns a detailed report of all citations within the specified time range for the specified lot.

- fetchZoneList(zoneID, lotID)
    - Return the list of zones for each lot as tuple pairs (lot, zone).

- getAvailableSpacesInLot(lotID):
    - Returns a list of available parking spaces in the specified lot.

6. Description of Views:

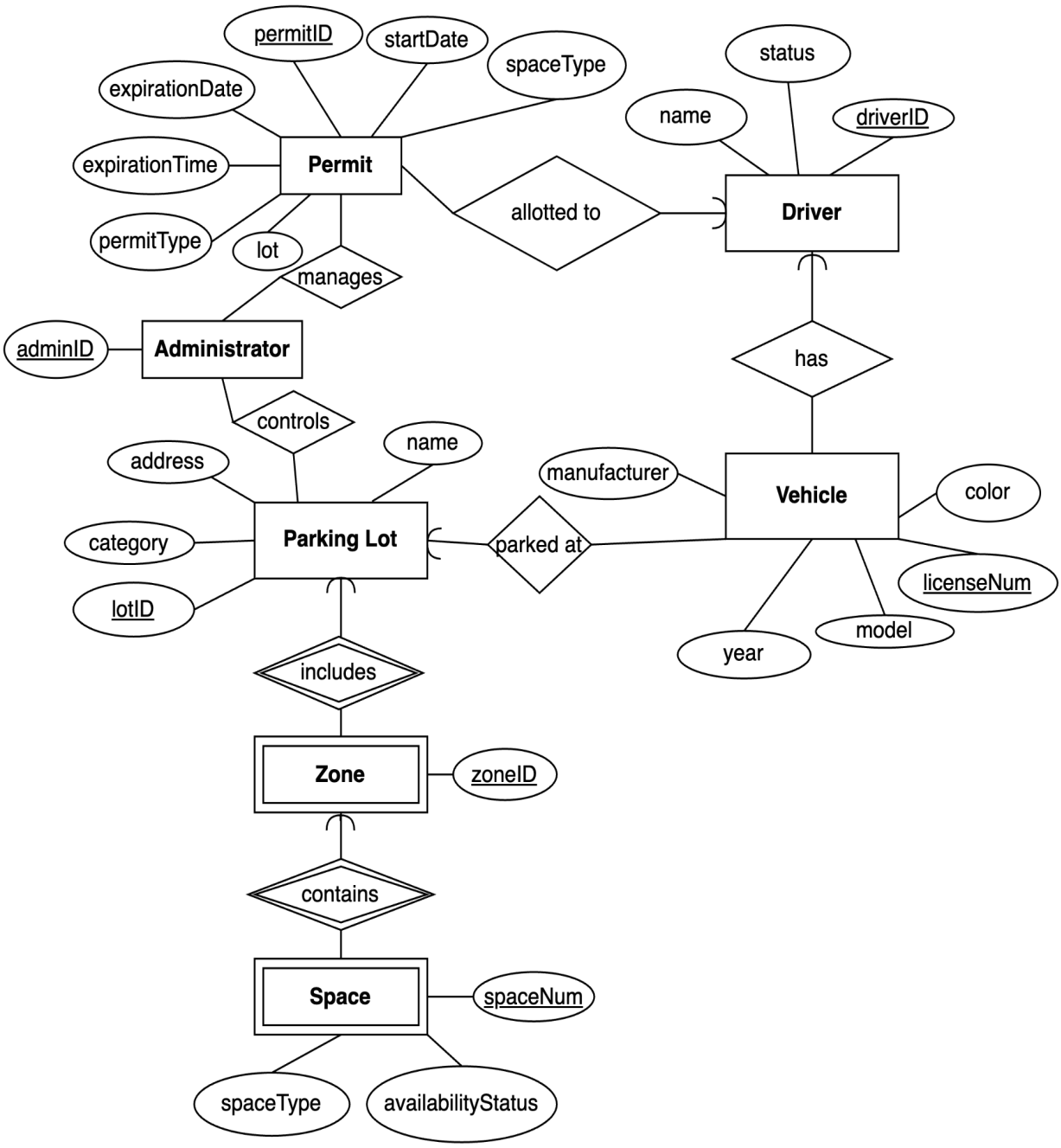
Driver's View: Drivers can have vehicles parked in the parking lot. They are allowed to get the permits and can appeal for the citations.

Security's View: Security can view and handle citation information, approve or reject the citation appeals, can generate monthly or annual citation reports.

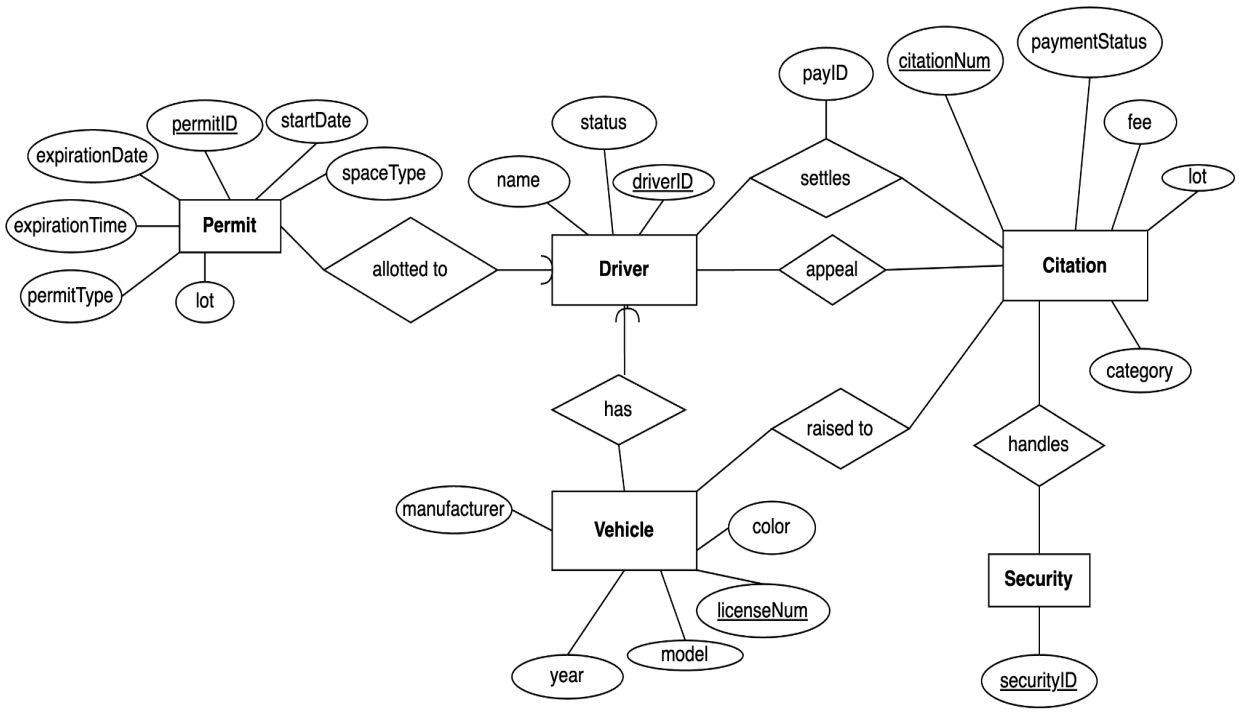
Administrator's View: Administrators can add parking lots to the system, assign zones and spaces to the lots, assign a parking permit to a driver, change the space availability, and verifies if a car has a valid permit.

7. Local E/R Diagrams:

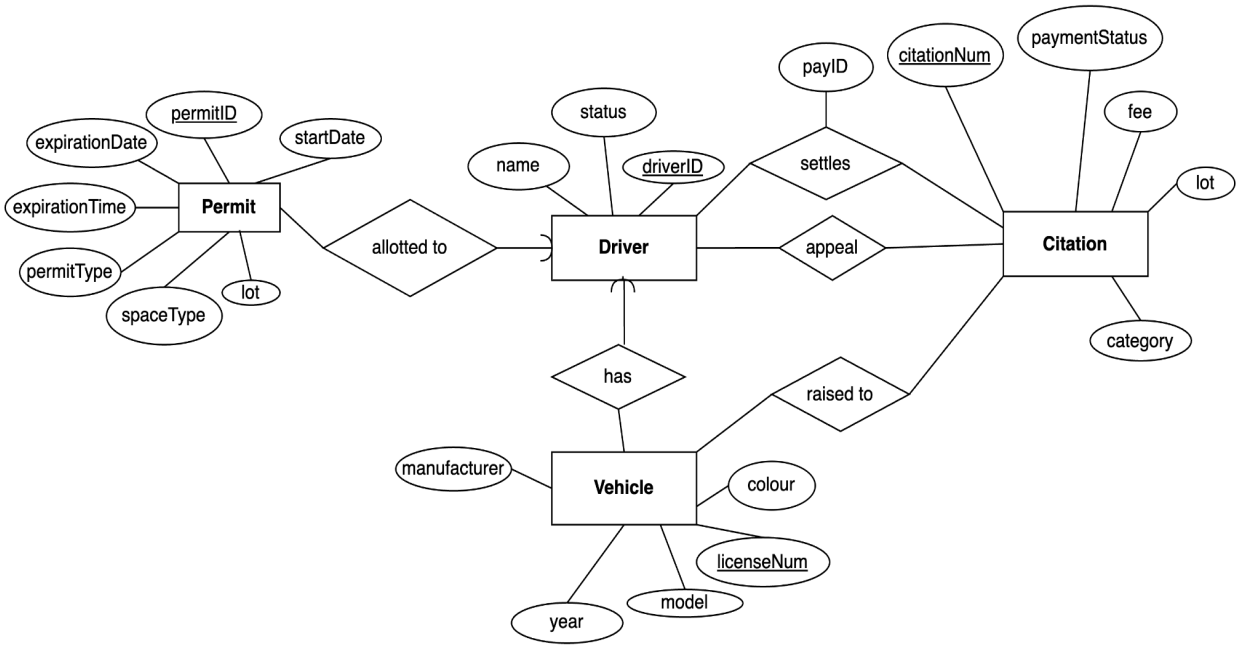
Administrator's View:



Security's View:



Driver's View:



8. Description of Local E/R diagrams:

- Every driver who arrives at the parking lot has a status (student, employee, or visitor), a unique driverID, and a name.
- One driver can have multiple vehicles, as per assumption #2**,** which a unique licenseNum can identify**.**
- A driver can also be associated with multiple permits as per assumption #3**,** where a unique permitID can identify the permit, as one particular permitID cannot be related to numerous drivers.
- Citations are identified by a unique citationNum associated with a particular driver raised on their vehicle.
- Drivers and citations have a many-many relationship. That means that many drivers can have multiple citations associated with them for their vehicles they own, according to assumption #4.
- Administrators identified by a unique adminID manage the permit, assign a parking permit to a driver, control the parking lot by assigning spaces to drivers based on availability, and check if a car has a valid permit in their lot.
- As per assumption #1, a vehicle can be parked only in a single parking lot, identified by a lotID, with an assigned zone and a space.
- Zones are also identified by a zoneID, which is unique as no two zones can have the same zoneID within a parking lot.
- Spaces are also identified by a spaceNum which is unique, as no two spaces in a zone inside a parking lot can have the same spaceNum.
- Parking lot is a strong entity associated with Zones, a weak entity further connected to another weak entity, Spaces. This is as per assumption #10**.**
- Security, who can be identified by a unique securityID, handles the citation information that violates parking regulations.
- Security also handles changing the status of payment made on the citation by the driver.

9. Local Relational Schemas:

Driver's View:

Driver (driverID, name, status)
Vehicle (licenseNum, manufacturer, color, model, year)
Permit (permitID, spaceType, startDate, expirationDate, permitType, expirationTime, lot)
Citation (citationNum, lot, fee, paymentStatus, category)

AllotedTo (permitID, driverID)
Appeals (driverID, citationNum)
Settles (driverID, citationNum, payID)
RaisedTo (citationNum, licenseNum)
Has(driverID, licenseNum)

Security's View:

Security (securityID)
Citation (citationNum, lot, fee, paymentStatus, category)
Driver (driverID, name, status)
Permit (permitID, spaceType, startDate, expirationDate, permitType, expirationTime, lot)
Vehicle (licenseNum, manufacturer, color, model, year)

Appeals (driverID, citationNum)
RaisedTo (citationNum, licenseNum)
Settles (driverID, citationNum, payID)
Handles (securityID, citationNum)
AllotedTo (permitID, driverID)
Has(driverID, licenseNum)

Administrator's View:

Driver (driverID, name, status)
Vehicle (licenseNum, manufacturer, color, model, year)

Permit (permitID, spaceType, startDate, expirationDate, permitType, expirationTime, lot)
Parking Lot (lotID, category, address, name)
Zone (zoneID, lotID)
Space (spaceNum, zoneID, lotID, spaceType, availabilityStatus)
Administrator (adminID)

Manages (permitID, adminID)
ParkedAt (lotID, licenseNum)
Controls (adminID, lotID)
AllotedTo (permitID, driverID)
Has(driverID, licenseNum)

10. Local Schema Documentation:

Entity Sets to Relations:
The entity sets Permit, Driver,Citation, Parking Lot, Security, were converted into relations with attributes users, drivers, vehicles, permits, citations, and parking lots.

Combining Many - One Relationships:

Weak entity set- “Zone” was made into a relation with all its attributes plus the lotID attribute, which is a foreign key from the Parking Lot entity set. Comprising lotID represents the “includes” relationship in the diagram between Zone and Parking lot; therefore, we do not have a separate relationship in the schema for “includes”. Zone was made into a weak entity set because it has a many-one relationship where we need to know the lotID in combination with the zoneID to look it up.

Weak entity set- “Space” was made into a relation with all its attributes plus the ZoneID and lotID attributes, which are foreign keys from the Zone and Parking Lot entity sets. Additional attributes zoneID and lotID represent the “contains” relationship in the diagram between the Zone and Space Entity set; therefore, we do not have a separate relationship in the schema for “Contains”. Space was made into

a weak entity set because it has a many-one relationship with Zone where we need to know the lotID, ZoneID and combination with the SpaceID to look it up.

Relationships to Relations:
Relationships parked at, allotted to, settles, appeal, raised to, controls, manages, handles has from ER diagram have all been turned into relations. The attributes in each relation are the primary keys of the entities they are connecting plus the attributes of that relationship itself. Settles relation has payID attribute as it is common to both the entities it is connecting, Driver and Citation.

Foreign Keys:
In Permit entity, lot attribute is a foreign key referencing the Parking Lot entity and spaceType attribute references the Space entity. Similarly, lot attribute in Citation entity is a foreign key that references Parking Lot entity. Through these foreign keys, we are ensuring that referential integrity is maintained and query performance is improved.

11. Inconsistency identified in the given problem prompt:

A handicapped person visiting the parking lot in an electric car or any other car type which has a dedicated space for parking in the lot must be allocated the handicapped space only for parking their car and not any other space type. If this is not followed, we will be violating the 1NF condition. Therefore we are following assumption #7, thereby resolving the inconsistency.

12. An Overview of the Complete ER Diagram: