NC STATE UNIVERSITY

Wolf Parking Management System

CSC 540 Database Management Systems - Project 2

PROJECT TEAM A

Aditi Vakeel	avakeel
Boscosylvester Chittilapilly	bchitti
Janani Pradeep	jpradee
Nahed Aub Zaid	naabuzai

Assumptions:

1. Drivers will not have access to this database. They must go to the parking office for any parking-related concerns or requests.
2. Driver goes to the billing staff to raise an appeal or make a payment for a citation.
3. To fetch any data related to the driver other than name, UID external databases are to be used.
4. Admin gets to know (this knowledge is not concerned with this DB) about the availability status of spaces (lots, Zones, types) and updates data accordingly.
5. The only payments recorded in this database are the ones made for citations, and no other fee is applicable for getting a permit.
6. Billing staff will update payment status based on payments and the appeal table.
7. A new entity called 'Payments' is maintained solely by the billing staff to store information about the citation payment information.
8. Billing Staff's decisions/actions on appeal approvals are based on external knowledge.
9. The appeal table would be used to store information about the appeal description. and appeal status. It takes values 'In Progress',' Approved', and 'Rejected'. The default value 'In Progress'.
10. Each citation can be challenged by at most one appeal and can have at most one payment.
11. To deal with deletion anomalies and reduce data redundancy, we maintain separate entities for 'SpaceType,' 'PermitType,' and 'CitationCategory'.
12. The Year of the Vehicles is assumed to be in the range 1900-2300.
13. Drivers with Status 'S' and 'E' have UID_Phone a 9-digit number and Drivers with Status 'V' have 10-digit phone numbers.
14. The Availability status in Spaces takes two values - 'Available', 'Unavailable' with the default value as 'Available'.
15. The Payment status in Citations takes three values - 'Paid', 'Unpaid', 'Waived Off' with the default value as 'Unpaid'.
16. Apart from the three categories mentioned in the narrative for Citation Category, three additional categories have been added to handle 50% Handicap Discount. Whenever a new category is added, one additional entry for Handicap Discount needs to be entered.
17. The operation "generate a report for the total number of citations given in all zones in the lot for a given time range (e.g., monthly or annually)" was implemented with the assumption that a report is being generated for a lotwise count of citations for a particular date range.
18. No two Parking Lots have the same combination of name and address.
19. We would be storing invalid parking lot, invalid zone or invalid space details in the Citations table which would help in reporting citations registered in a specific location. The Citations table would be linked to the Spaces table by a One-Many relationship on the ParkinglotID, ZoneID and SpaceNumber.
20. Report for returning the available space number given a space type and parking lot returns only 1 space number even though there could be multiple spaces available because the narrative says
    "Return a.a available space number given a space type in a given parking lot."

1) Global Relational Database Schema:

- **Vehicles (CarlicenceNumber, Model, Year, Manufacturer, Color)**
  CarlicenceNumber - CarlicenceNumber, Model, Year, Manufacturer, Color
  This relation is in 3NF as it satisfies BCNF because the attribute CarlicenceNumber is a primary key and uniquely identifies Model, Year, Manufacturer, and Color. We cannot take any other combination of attributes to derive CarlicenceNumber because two or more cars can have the same color and year of manufacturing.

- **Drivers (Name, Status, UID_Phone)**
  UID_Phone - Name, Status, UID_Phone
  This relation is in 3NF as it satisfies BCNF because the attribute UID_Phone is a primary key. We cannot use Name or Status as a key because two or more drivers can have the same Name. Multiple students, employees, or visitors also can exist. Both of these cannot indicate the UID or Phone number of a driver.

- **Parkinglots (Name, ParkinglotID, Address)**
  ParkinglotID - Name, ParkinglotID, Address
  Name, Address - ParkinglotID, Name, Address
  This relation is in 3NF as it satisfies BCNF because the attribute ParkinglotID is a primary key and uniquely identifies the Address and Name of the ParkinglotID. Also, we can use Name and Address to identify a parking lot uniquely through its parkinglotID. Both functional dependencies satisfy the BCNF and hence 3NF conditions.

- **Zones (ZoneID, ParkinglotID)**
  ZoneID, ParkinglotID ---t ZoneID, ParkinglotID
  This relation is in 3NF as it satisfies BCNF because the relation contains 2 attributes only, which are combinedly superkey.

- **Spaces (SpaceNumber, ZoneID, ParkinglotID, AvailabilityStatus, SpaceTypeID)**
  ParkinglotID, ZoneID, SpaceNumber - ParkinglotID, ZoneID, SpaceNumber, AvailabilityStatus, SpaceTypeID
  This relation is in 3NF as it satisfies BCNF because the attributes SpaceNumber, ZoneID, and ParkinglotID are combined primary keys and uniquely identify the AvailabilityStatus, and SpaceTypeID. We cannot use AvailabilityStatus or SpaceTypeID as key as multiple spaces can be available at a time and can have the same SpaceTypeID

Also, we can use Name and Address to identify a parking lot uniquely through its parkinglotID.

- **SpaceTypes(SpaceType, <u>SpaceTypeID)</u>**
  SpaceTypeID - SpaceType, SpaceTypeID
  This relation is in 3NF as it satisfies BCNF because the relation contains 2 attributes only, of which SpaceTypeID is a primary key.

- **Permits <u>(PermitID,</u> StartDate, ExpirationDate, ExpirationTime, PermitTypeID, ParkinglotID, ZoneID, SpaceTypeID, UID_Phone, CarlicenseNumber)**
  PermitID - PermitID, StartDate, ExpirationDate, ExpirationTime, PermitTypeID, ParkinglotID, ZoneID, SpaceTypeID, UID_Phone, CarlicenseNumber
  This relation is in 3NF as it satisfies BCNF because the attribute PermitID is a primary key and uniquely identifies StartDate, ExpirationDate, ExpirationTime, PermitTypeID, ParkinglotID, ZoneID, SpaceTypeID, UID_Phone, CarlicenseNumber. We cannot take any other combination of attributes to identify a permit uniquely because a CarlicenseNumber can have multiple permits or two or more permits can have the same StartDate, ExpirationDate combination.

- **PermitTypes (PermitType, <u>PermitTypeID)</u>**
  PermitTypeID-PermitType, PermitTypeID
  This relation is in 3NF as it satisfies BCNF because the relation contains 2 attributes only, of which PermitTypeID is a primary key.

- **Citations <u>(CitationNumber</u>, CitationDate, CitationTime, PaymentStatus, CitationCategoryID, CarlicenseNumber, ParkinglotID, ZoneID, SpaceNumber)**
  CitationNumber- CitationNumber, CitationDate, CitationTime, PaymentStatus, CitationCategoryID, CarlicenseNumber, ParkinglotID, ZoneID, SpaceNumber
  This relation is in 3NF as it satisfies BCNF because the attribute CitationNumber is a primary key and uniquely identifies CitationDate, CitationTime, PaymentStatus, CitationCategoryID, CarlicenceNumber, ParkinglotID, ZoneID and SpaceNumber. We cannot take any other combination of attributes to identify a citation uniquely because a CarlicenseNumber can have multiple citations, or two or more citations can have the same CitationDate, CitationTime combination.

- **CitationCategories (<u>CitationCategoryID</u>, CitationCategory, Fee)**
  CitationCategoryID - CitationCategoryID, CitationCategory, Fee
  This relation is in 3NF as it satisfies BCNF because the relation contains 2 attributes only, of which CitationCategoryID is a primary key.

- **Payments (<u>PaymentID,</u> PaymentDate, PaymentTime, Amount, CitationNumber)**
  PaymentID - PaymentID, PaymentDate, PaymentTime, Amount, CitationNumber
  CitationNumber - PaymentID, PaymentDate, PaymentTime, Amount, CitationNumber
  This relation is in 3NF as it satisfies BCNF as PaymentID uniquely identifies all the attributes, PaymentDate, PaymentTime, Amount, and CitationNumber. Multiple payments can have the same payment dates and times with the same amount. Thus, no combination of PaymentDate, PaymentTime, or Amount can uniquely identify a payment. CitationNumber uniquely identifies all the attributes, too. However, it acts as a foreign key from Citations relation. Both functional dependencies satisfy the BCNF and hence 3NF conditions.

- **Appeals (<u>AppealID,</u> AppealDescription, AppealStatus, CitationNumber)**
  AppealID-AppealID, AppealDescription, AppealStatus, CitationNumber
  CitationNumber-AppealID, AppealDescription, AppealStatus, CitationNumber
  This relation is in 3NF as it satisfies BCNF as AppealID uniquely determines the rest of the attributes AppealDescription, AppealStatus, and CitationNumber. Any combination of AppealDescription and AppealStatus cannot uniquely determine an appeal made by the driver because the AppealStatus has only three values, 'in progress,' 'approved,' and 'rejected,' which can be used by multiple payments and AppealDescription can be the same for two drivers who come under the same citation categories. In this relationship, too, CitationNumber acts as a foreign key and uniquely determines an appeal made by a driver. Both functional dependencies satisfy the BCNF and hence 3NF conditions.

2) Design Decisions for Global Schema:

All the entity sets in our diagrams are converted into relations based on the E/R viewpoints of the classes of users for this database. Weak entities and relationships in the schema have been converted by including the primary key of the strong entity set as the primary key of the weak entity set. Weak relationships do not require any other conversion process, as the process of converting weak entities covers weak relationships as well.

Many-to-one relationships were converted by taking the primary key on the one side and placing it as the foreign key on the many sides of the relationship. One-to-one relationships can be converted by moving the primary key from any one side to the other as the foreign key. In our case, we have two one-to-one relationships between Citations and Payments and Citations and Appeals. Moving the primary keys from Payments and Appeals into Citations as the foreign key would lead to a possibility of having NULL values in the Citations relation for the foreign key attributes. Hence, we have decided to move the primary key from Citations relation to Payments and Appeals relations as the foreign key.

During the ERO design process, we had created new entities for SpaceTypes, PermitTypes, CitationCategories, Payments and Appeals. These entities were created to deal with deletion anomalies and to reduce data redundancy. These entities were also converted into relations using the above-mentioned process.

- **Vehicles (<u>CarlicenceNumber</u>, Model, Year, Manufacturer, Color)**
    a. *Primary key:* CarlicenseNumber
    b. *Foreign keys:* none
    c. *Unique keys:* none
    d. *NULL constraints:*
        i. Model, Color are NOT NULL
        Explanation: Model and Color information is essential as they help the staff members in identifying a vehicle.
        ii. Year, Manufacturer attributes are allowed to be NULL
        Explanation: Year = NULL and Manufacturer = NULL would arise from a scenario where the admin is not sure of the manufacturer and year of the specific car. These NULL values would simply

mean that the data is not available now. In future, if required, the vehicle information can be updated to add these details.

e. *Check constraints:*
   i. Year: Year should be after 1900 and before 2300 as per our assumptions

- **Drivers (Name, Status, <u>UID_Phone)</u>**
  a. *Primary key:* UID_Phone
  b. *Foreign keys:* none
  c. *Unique keys:* none
  d. *NULL constraints:*
     i. All attributes are NOT NULL
     Explanation: These attributes cover the basic information for a driver that are also essential to the system and hence cannot be NULL
  e. *Check constraint:*
     i. Status: "S", "E" or "V" are the valid values for Status as mentioned in the narrative
     ii. UID_Phone: If the status is "S" or "E" indicating the driver is a student or an employee, then the UID_Phone should have a 9-digit ID number. If the status is "V" indicating the driver is a visitor, then the UID_Phone should have a 10-digit phone number.

- **Parkinglots <u>(ParkinglotID</u>, Name, Address)**
  a. *Primary key:* ParkingLotlD
  b. *Foreign keys:* none
  c. *Unique keys:* none
  d. *NULL constraints:*
     i. All attributes are NOT NULL
     Explanation: These attributes cover the basic information for a parking lot that are also essential to the system and hence cannot be NULL
  e. *Check constraint:*
     i. ParkingLotlD: This should be a positive number greater than 0.

- **Zones <u>(ZonelD, ParkinglotlD)</u>**
  a. *Primary key:* ParkingLotlD, ZonelD
  b. *Foreign keys:* ParkingLotlD
  c. *Unique keys:* none

d. *NULL constraints:*
    i. All attributes are NOT NULL
    Explanation: These attributes are the primary keys for this table and hence cannot be NULL
e. *Check constraints:*
    i. ZoneID: Valid values for ZoneID, as mentioned in the narrative are **"A", "B", "C", "D", "AS", "BS", "CS", "DS" and "V".**

- **Spaces <u>(SpaceNumber, ZoneID, ParkinglotID,</u> AvailabilityStatus, SpaceTypeID)**
    a. *Primary key:* SpaceNumber, ZoneID, ParkingLotID
    b. *Foreign keys:* ParkingLotID, ZoneID, SpaceTypeID
    c. *Unique keys:* none
    d. *NULL constraints:*
        i. All attributes are NOT NULL
        Explanation: These attributes cover the basic information for a space that are also essential to the system and hence cannot be NULL
    e. *Check constraint:*
        i. SpaceNumber: This should be a positive number greater than 0
        ii. AvailabilityStatus: Valid values for this attribute are "Available" and "Unavailable" as mentioned in the assumptions

- **SpaceTypes <u>(SpaceTypeID,</u> SpaceType)**
    a. *Primary key:* SpaceTypeID
    b. *Foreign keys:* none
    c. *Unique keys:* SpaceType
    d. *NULL constraints:*
        i. All attributes are NOT NULL
        Explanation: These attributes cover the basic information for a space type that are also essential to the system and hence cannot be NULL
    e. *Check constraints:* none

- **Permits <u>(PermitID,</u> StartDate, ExpirationDate, ExpirationTime, PermitTypeID, ParkinglotID, ZoneID, SpaceTypeID, UID_Phone, CarlicenseNumber)**
    a. *Primary key:* PermitID

b. *Foreign keys:* PermitTypeID, ParkingLotID, ZoneID, SpaceTypeID, UID_Phone, CarlicenseNumber
c. *Unique keys:* none
d. *NULL constraints:*
    i. All attributes are NOT NULL
    Explanation: These attributes cover the basic information for a permit that are also essential to the system and hence cannot be NULL
e. *Check constraint:* none

- **PermitTypes (PermitTypeID, PermitType)**
    a. *Primary key:* PermitTypeID
    b. *Foreign keys:* none
    c. *Unique keys:* PermitType
    d. *NULL constraints:*
        i. All attributes are NOT NULL
        Explanation: These attributes cover the basic information for a permit type that are also essential to the system and hence cannot be NULL
    e. *Check constraint:* none

- **Citations (CitationNumber, CitationDate, CitationTime, PaymentStatus, CarlicenseNumber, CitationCategoryID, ParkinglotID, ZoneID, SpaceNumber)**
    a. *Primary key:* CitationNumber
    b. *Foreign keys:* CarlicenseNumber, CitationCategoryID, ParkingLotID, ZoneID, SpaceNumber
    c. *Unique keys:* none
    d. *NULL constraints:*
        i. All attributes are NOT NULL
        Explanation: These attributes cover the basic information for a citation that are also essential to the system and hence cannot be NULL
    e. *Check constraints:*
        i. PaymentStatus: Valid values for payment status based on the assumptions are "Paid", "Unpaid" and "Waived Off"

- **CitationCategories (CitationCategoryID, CitationCategory, Fee)**

a. *Primary key:* CitationCategorylD
b. *Foreign keys:* none
c. *Unique keys:* CitationCategory
d. *NULL constraints:*
    i. All attributes are NOT NULL
    Explanation: These attributes cover the basic information for a citation category that are also essential to the system and hence cannot be NULL
e. *Check constraint:*
    i. Fee: Fee value should be greater than $0.

- **Payments (Payment!D, PaymentDate, PaymentTime, Amount, CitationNumber)**
    a. *Primary key:* PaymentlD
    b. *Foreign keys:* CitationNumber
    c. *Unique keys:* CitationNumber
    d. *NULL constraints:*
        i. All attributes are NOT NULL
        Explanation: These attributes cover the basic information for a payment that are also essential to the system and hence cannot be NULL
    e. Check constraint:
        i. Amount: Amount should be greater than $0.

- **Appeals (AppealID, AppealDescription, AppealStatus, CitationNumber)**
    a. *Primary key:* AppeallD
    b. *Foreign keys:* CitationNumber
    c. *Unique keys:* none
    d. *NULL constraints:*
        i. AppeallD, AppealStatus, CitationNumber are NOT NULL
        Explanation: AppeallD being the primary key cannot be NULL. When an appeal is created it would have a default value and hence it can never have a NULL value. An appeal can be created for a citation, so if an appeal is present then a citation is also present and hence CitationNumber cannot be NULL.
        ii. AppealDescription attribute is allowed to be NULL
        Explanation: AppealDescription = NULL would arise from a scenario where an appeal is generated without entering any

justification for the appeal. Even though this is not an ideal case, such a scenario could happen. These NULL values would simply mean that the description is empty.

e. Check constraints:
   i. AppealStatus: Valid values for appeal status based on the assumptions are "In Progress", "Approved" and "Rejected".

3) Base Relations:

- **Vehicles**

```
CREATE TABLE Vehicles
(
        CarlicenceNumber VARCHAR(10) PRIMARY KEY,
        Model VARCHAR(50) NOT NULL,
        Year INT CHECK (YEAR >= 1900 AND YEAR <= 2300),
        Manufacturer VARCHAR(50),
        Color VARCHAR(50) NOT NULL
);

SELECT * FROM Vehicles;
```

Output:

```
MySQL  classdb2.csc.ncsu.edu:3306  naabuzai  SQL > SELECT * FROM Vehicles;
+-----------------+------+---------+----------+--------------+
| CarLicenceNumber | Year | Color   | Model    | Manufacturer |
+-----------------+------+---------+----------+--------------+
| ABC555          | 2022 | Green   | Corolla  | Toyota       |
| DEF456          | 2019 | Silver  | F-150    | Ford         |
| EFG123          | 2012 | Cyan    | 3 Series | BMW          |
| FDE123          | 2020 | Red     | Corolla  | Toyota       |
| FDE666          | 2020 | Red     | Corolla  | Toyota       |
| GHI789          | 2017 | Blue    | Malibu   | Chevrolet    |
| HIJ456          | 2011 | Magenta | A4       | Audi         |
+-----------------+------+---------+----------+--------------+
7 rows in set (0.0017 sec)
```

- **Drivers**

```
CREATE TABLE Drivers
(
        Name VARCHAR(50) NOT NULL,
        Status CHAR(1) NOT NULL CHECK (Status IN ("S", "E", "V")),
        UID_Phone INT PRIMARY KEY,
        CHECK ((
                Status IN ("S", "E")
                AND UID PHONE>= 100000000
                AND UID  PHONE<= 999999999
            )OR(
                Status= "V"
                AND UID Phone>= 1000000000
                AND UID Phone< 9999999999
        ))
);

SELECT * FROM Drivers;
```

Output:

```
MySQL  classdb2.csc.ncsu.edu:3306  naabuzai  SQL > select * from Drivers;
+---------------+--------+------------+
| Name          | Status | UID_Phone  |
+---------------+--------+------------+
| Robert Wilson | E      | 200132323  |
| Sami Sami     | E      | 200144322  |
| Lisa Davis    | E      | 200286437  |
| John Doe      | E      | 200520022  |
| Jane Smith    | E      | 200620043  |
| Mark Johnson  | E      | 200786754  |
| Sami Sami     | E      | 200944322  |
| Grace Moore   | S      | 204535678  |
| Khalid Ali    | S      | 204985678  |
| Eva Brown     | S      | 764985678  |
| Laura Harris  | V      | 9192914323 |
| Alice Smith   | S      | 984985678  |
+---------------+--------+------------+
12 rows in set (0.0020 sec)
```

- **Parkinglots**

```
CREATE TABLE ParkingLots
(
        Name VARCHAR(50) NOT NULL,
        ParkingLotID INT PRIMARY KEY CHECK (ParkingLotID > 0),
        Address VARCHAR(100) NOT NULL
);

SELECT * FROM ParkingLots;
```

Output:

```
MySQL  classdb2.csc.ncsu.edu:3306  naabuzai  SQL > SELECT * FROM ParkingLots;
+---------------+--------------+-----------------+
| Name          | ParkingLotID | Address         |
+---------------+--------------+-----------------+
| Parking Lot H |            1 | 435 Main Street |
| Parking Lot B |            2 | 456 Elm St      |
| Parking Lot C |            3 | 789 Oak St      |
| Parking Lot D |            4 | 101 Pine St     |
| Parking Lot E |            5 | 202 Maple St    |
| Parking Lot F |            6 | 303 Birch St    |
| Parking Lot G |            7 | 404 Cedar St    |
| Parking Lot K |           11 | 808 Sycamore St |
| Parking Lot A |           18 | 123 Main Street |
+---------------+--------------+-----------------+
9 rows in set (0.0017 sec)
```

- **Zones**

```
CREATE TABLE Zones
(
        ZoneID VARCHAR(2)
        CHECK (ZoneID IN ("A", "B", "C", "D", "AS", "BS", "CS", "OS", "V")),
        ParkingLotID INT,
        PRIMARY KEY(ZoneID, ParkingLotID),
        FOREIGN KEY (ParkingLotID) REFERENCES ParkingLots(ParkingLotID)
        ON UPDATE CASCADE
);

SELECT * FROM Zones;
```

Output:

```
MySQL  classdb2.csc.ncsu.edu:3306  naabuzai  SQL > select * from Zones;
+--------+--------------+
| ZoneID | ParkingLotID |
+--------+--------------+
| A      |            1 |
| AS     |            5 |
| B      |            2 |
| BS     |            6 |
| C      |            3 |
| CS     |            1 |
| D      |            1 |
| D      |            4 |
| V      |            1 |
+--------+--------------+
9 rows in set (0.0016 sec)
```

- **Space Types**

```
CREATE TABLE SpaceTypes
(
        SpaceType VARCHAR(50) NOT NULL UNIQUE,
        SpaceTypeID INT PRIMARY KEY AUTO_INCREMENT
);

SELECT*  FROM  SpaceTypes;
```

Output:

```
MySQL  classdb2.csc.ncsu.edu:3306  naabuzai  SQL > select * from SpaceTypes;
+-------------+-------------+
| SpaceType   | SpaceTypeID |
+-------------+-------------+
| electric    |           1 |
| handicap    |           2 |
| compact car |           3 |
| regular     |           4 |
+-------------+-------------+
4 rows in set (0.0012 sec)
```

- **Spaces**

```
CREATE TABLE Spaces
(
        SpaceNumber INT NOT NULL CHECK (SpaceNumber > 0),
        ZoneID VARCHAR(2),
        ParkingLotID  INT,
```

```
        AvailabilityStatus VARCHAR(20) NOT NULL DEFAULT "Available"
        CHECK (AvailabilityStatus IN ("Available", "Unavailable")),
        SpaceTypeID INT NOT NULL DEFAULT 4,
        PRIMARY KEY (SpaceNumber,ZoneID,ParkingLotID),
        FOREIGN KEY (ParkingLotID, ZoneID)
        REFERENCES Zones(ParkingLotID, ZoneID)
        ON UPDATE CASCADE,
        FOREIGN KEY (SpaceTypeID) REFERENCES
        SpaceTypes(SpaceTypeID) ON UPDATE CASCADE
);
SELECT * FROM Spaces;
```

Output:

```
MySQL  classdb2.csc.ncsu.edu:3306  naabuzai  SQL > select * from Spaces;
+-------------+--------+--------------+--------------------+-------------+
| SpaceNumber | ZoneID | ParkingLotID | AvailabilityStatus | SpaceTypeID |
+-------------+--------+--------------+--------------------+-------------+
|           9 | AS     |            5 | Available          |           4 |
|          10 | A      |            1 | Unavailable        |           1 |
|          11 | BS     |            6 | Available          |           2 |
|          12 | C      |            3 | Unavailable        |           3 |
|         101 | B      |            2 | Available          |           1 |
|         103 | D      |            1 | Unavailable        |           1 |
+-------------+--------+--------------+--------------------+-------------+
6 rows in set (0.0015 sec)
```

- **Permit Types**

```
CREATE TABLE PermitTypes
(
        PermitType VARCHAR(50) NOT NULL UNIQUE,
        PermitTypeID INT PRIMARY KEY AUTOINCREMENT
);

SELECT * FROM PermitTypes;
```

Output:

```
MySQL  classdb2.csc.ncsu.edu:3306  naabuzai  SQL > select * from PermitTypes;
+---------------+--------------+
| PermitType    | PermitTypeID |
+---------------+--------------+
| Peak Hours    |            1 |
| Commuter      |            2 |
| Peak Hours    |            3 |
| Special Event |            4 |
| Park & Ride   |            5 |
+---------------+--------------+
5 rows in set (0.0021 sec)
```

- **Permits**

```
CREATE TABLE Permits (
        PermitID INT PRIMARY KEY AUTOINCREMENT,
        StartDate DATE NOT NULL,
        ExpirationDate DATE NOT NULL,
        ExpirationTime TIME NOT NULL,
        PermitTypeID INT,
        ParkingLotID INT,
        ZoneID VARCHAR(2),
        SpaceTypeID INT,
        UID_Phone INT,
        CarlicenceNumber INT,
        FOREIGN KEY (PermitTypeID) REFERENCES
        PermitTypes(PermitTypeID) ON UPDATE CASCADE,
        FOREIGN KEY (ParkingLotID, ZoneID) REFERENCES
        Zones(ParkingLotID, ZoneID)
        ON UPDATE CASCADE,
        FOREIGN KEY (SpaceTypeID)  REFERENCES
        SpaceTypes(SpaceTypeID) ON UPDATE CASCADE,
        FOREIGN KEY (UID_Phone) REFERENCES Drivers(UID_Phone)
        ON UPDATE CASCADE,
        FOREIGN KEY (CarlicenceNumber) REFERENCES
        Vehicles(CarlicenceNumber) ON UPDATE CASCADE
);

SELECT * FROM Permits;
```

Output:

```
MySQL classdb2.csc.ncsu.edu:3306 naabuzai SQL > select * from Permits;
+----------+------------+---------------+---------------+--------------+--------+------------+------------+------------------+--------------+
| PermitID | StartDate  | ExpirationDate | ExpirationTime | PermitTypeID | ZoneID | SpaceTypeID | UID_Phone  | CarLicenseNumber | ParkingLotID |
+----------+------------+---------------+---------------+--------------+--------+------------+------------+------------------+--------------+
|        1 | 2023-10-01 | 2023-12-31    | 23:59:59      |            1 | C      |          4 | 200132323  | ABC555           |            3 |
|        2 | 2023-02-01 | 2023-11-30    | 13:00:00      |            2 | A      |          4 | 200144322  | DEF456           |            1 |
|        3 | 2023-11-01 | 2023-12-31    | 23:59:59      |            3 | CS     |          4 | 764985678  | FDE123           |            1 |
|        4 | 2023-04-01 | 2023-09-30    | 15:00:00      |            4 | B      |          1 | 200286437  | GHI789           |            2 |
|        5 | 2023-08-01 | 2023-10-31    | 17:30:00      |            1 | V      |          2 | 9192914323 | GHI789           |            1 |
+----------+------------+---------------+---------------+--------------+--------+------------+------------+------------------+--------------+
5 rows in set (0.0017 sec)
```

- **Citation Categories**

```
CREATE TABLE CitationCategories
(
        CitationCategorylD INT AUTO_INCREMENT PRIMARY KEY,
        CitationCategory VARCHAR(50)  NOT NULL UNIQUE,
        Fee DECIMAL(5,2)  NOT NULL CHECK  (Fee> 0.00)
);

SELECT * FROM CitationCategories;
```

Output:

```
MySQL classdb2.csc.ncsu.edu:3306 naabuzai SQL > SELECT * FROM CitationCategories;
+--------------------+-----------------------------------+-------+
| CitationCategoryID | CitationCategory                  | Fee   |
+--------------------+-----------------------------------+-------+
|                  1 | Invalid Permit                    | 25.00 |
|                  2 | Expired Permit                    | 30.00 |
|                  3 | No Permit                         | 40.00 |
|                  4 | Handicap Discount - Invalid Permit | 12.50 |
|                  5 | Handicap Discount - Expired Permit | 15.00 |
|                  6 | Handicap Discount - No Permit     | 20.00 |
+--------------------+-----------------------------------+-------+
6 rows in set (0.0033 sec)
```

- **Citations**

```
CREATE TABLE Citations
(
        CitationNumber INT AUTO  INCREMENT  PRIMARY  KEY,
        CitationDate DATE NOT NULL,
        CitationTime TIME NOT NULL,
        PaymentStatus  VARCHAR(20) NOT NULL CHECK (PaymentStatus IN
        ("In Progress", "Approved", "Rejected")) DEFAULT "In Progress",
```

```
        CitationCategoryID INT,
        CarlicenseNumber VARCHAR(10),
        ParkingLotID INT NOT NULL,
        ZoneID VARCHAR(2) NOT NULL,
        SpaceNumber INT NOT NULL,
        FOREIGN KEY (CarlicenseNumber)
        REFERENCES (Vehicles)
        ON UPDATE CASCADE,
        FOREIGN KEY (CitationCategoryID)
        REFERENCES CitationCategories(CitationCategoryID)
        ON UPDATE CASCADE,
        FOREIGN KEY (ParkingLotID, ZoneID, SpaceNumber)
        REFERENCES Spaces(ParkingLotID, ZoneID, SpaceNumber)
        ON UPDATE CASCADE,
);

SELECT * FROM Citations;
```

Output

```
MySQL  classdb2.csc.ncsu.edu:3306  naabuzai  SQL > SELECT * FROM Citations;
+----------------+-------------+-------------+---------------+-------------------+-----------------+-------------+--------+-------------+
| CitationNumber | CitationDate | CitationTime | PaymentStatus | CitationCategoryID | CarLicenseNumber | ParkingLotID | ZoneID | SpaceNumber |
+----------------+-------------+-------------+---------------+-------------------+-----------------+-------------+--------+-------------+
|              1 | 2023-03-01  | 14:00:00    | Paid          |                 9 | DEF456          |           6 | DS     |          11 |
|              2 | 2023-04-01  | 15:00:00    | Paid          |                 3 | EFG123          |           5 | AS     |           9 |
|              3 | 2023-03-01  | 14:00:00    | Paid          |                 9 | DEF456          |           6 | DS     |          11 |
|              4 | 2023-04-01  | 15:00:00    | Paid          |                 3 | EFG123          |           5 | AS     |           9 |
|              5 | 2023-05-01  | 16:00:00    | Paid          |                 5 | GHI789          |           1 | D      |         103 |
|              6 | 2023-06-01  | 17:00:00    | Paid          |                 3 | HIJ456          |           3 | C      |          12 |
|              7 | 2023-07-01  | 18:00:00    | Unpaid        |                 3 | ABC555          |           1 | D      |         103 |
+----------------+-------------+-------------+---------------+-------------------+-----------------+-------------+--------+-------------+
7 rows in set (0.0032 sec)
```

- **Payments**

```
CREATE TABLE Payments
(
        PaymentID INT AUTO_INCREMENT PRIMARY KEY,
        Paymentoate DATE NOT NULL,
        PaymentTime TIME NOT NULL,
        Amount DECIMAL(8,2) NOT NULL CHECK (Amount> 0.00),
        CitationNumber INT NOT NULL UNIQUE,
        FOREIGN KEY (CitationNumber)
```

```
        REFERENCES Citations(CitationNumber) ON UPDATE CASCADE
);

SELECT * FROM Payments;
```

Output:

```
MySQL  classdb2.csc.ncsu.edu:3306  naabuzai  SQL > select * from Payments;
+-----------+-------------+-------------+--------+----------------+
| PaymentID | PaymentDate | PaymentTime | Amount | CitationNumber |
+-----------+-------------+-------------+--------+----------------+
|         1 | 2023-03-01  | 14:00:00    |  40.00 |              1 |
|         2 | 2023-04-01  | 15:00:00    |  25.00 |              2 |
|         3 | 2023-03-01  | 14:00:00    |  40.00 |              3 |
|         4 | 2023-04-01  | 15:00:00    |  25.00 |              4 |
|         5 | 2023-05-01  | 16:00:00    |  30.00 |              5 |
|         6 | 2023-06-01  | 17:00:00    |  40.00 |              6 |
+-----------+-------------+-------------+--------+----------------+
6 rows in set (0.0014 sec)
```

- **Appeals**

```
CREATE TABLE Appeals
(
        AppealID INT AUTO_INCREMENT PRIMARY KEY,
        AppealDescription VARCHAR(255),
        AppealStatus VARCHAR(50) NOT NULL CHECK (AppealStatus IN ("In
        Progress", "Approved", "Rejected")) DEFAULT "In Progress",
        CitationNumber INT NOT NULL UNIQUE,
        FOREIGN KEY (CitationNumber)
        REFERENCES Citations(CitationNumber) ON UPDATE CASCADE
);

SELECT * FROM Appeals;
```

Output:

```
MySQL  classdb2.csc.ncsu.edu:3306  naabuzai  SQL > select * from Appeals;
+----------+----------------------+--------------+----------------+
| AppealID | AppealDescription    | AppealStatus | CitationNumber |
+----------+----------------------+--------------+----------------+
|        1 | Appeal Description 2 | Pending      |              1 |
|        2 | Appeal Description 2 | Pending      |              2 |
|        3 | Appeal Description 3 | Pending      |              3 |
|        4 | Appeal Description 5 | Pending      |              4 |
|        5 | Appeal Description 5 | Pending      |              5 |
|        6 | Appeal Description 6 | Rejected     |              6 |
+----------+----------------------+--------------+----------------+
6 rows in set (0.0013 sec)
```

4) SQL Queries:

4.1

Operation 1: Information Processing

- **Add Driver**
  SOL> INSERT INTO Drivers (UID_Phone, Name, Status) VALUES (200144322, 'Sarni Sarni', 'E')·,
  Query OK, 1 row affected (0.0113 sec)

- **Update Driver**
  SOL> UPDATE Drivers SET Name= 'Khalid Ali', Status= 'S' WHERE UID_Phone = 984985678;
  Query OK, 1 row affected (0.0079 sec)
  Rows matched: 1 Changed: 1 Warnings: 0

- **Delete Driver**
  SOL> DELETE FROM Permits WHERE UID_Phone = 98498567$.
  Query OK, 1 row affected (0.0028 sec)

- **Add Parking Lot**
  SOL> INSERT INTO ParkingLots (ParkingLotID, Name, Address) VALUES (18, 'Parking Lot A', '123 Main Street');
  Query OK, 1 row affected (0.0115 sec)

- **Update Parking Lot**
  SQL> UPDATE ParkingLots SET Name= 'Parking Lot BB', Address= '456 Elm MS' WHERE ParkingLotID = 18;
  Query OK, 1 row affected (0.0091 sec)
  Rows matched: 1 Changed: 1 Warnings: 0

- **Delete Parking Lot**
  SQL> DELETE FROM ParkingLots where ParkingLotID=18;
  Query OK, 1 row affected (0.0028 sec)

- **Add Zone**
  SQL> INSERT INTO Zones (ParkingLotID, ZoneID) VALUES (1, 'D');
  Query OK, 1 row affected (0.0037 sec)

- **Delete Zone**
  SQL> DELETE FROM Zones WHERE ZoneID = 'A' and ParkingLotID=18;
  Query OK, 1 row affected (0.0029 sec)

- **Add Space**
  SQL> INSERT INTO Spaces (ParkingLotID, ZoneID, SpaceNumber, AvailabilityStatus, SpaceTypeID) VALUES (1, 'D', 103, 'Unavailable', 1);
  Query OK, 1 row affected (0.0029 sec)

- **Update Space**
  SQL> UPDATE Spaces Set AvailabilityStatus = 'Available' where ZoneID = 'C' and ParkingLotID = 2 AND SpaceNumber = 6;
  Query OK, 1 row affected (0.0013 sec)
  Rows matched: 1 Changed: 1 Warnings: 0

- **Delete Space**
  SQL> DELETE FROM Spaces WHERE ParkingLotID = 2 AND ZoneID = 'C' AND SpaceNumber = 6;
  Query OK, 1 row affected (0.0032 sec)

- **Update Space Type in Space**
  SQL> UPDATE Spaces SET SpaceTypeID =1 WHERE ParkingLotID = 3 AND ZoneID = 'AS' AND SpaceNumber = 10;
  Query OK, 1 row affected (0.0061 sec)
  Rows matched: 1 Changed: 1 Warnings: 0

- **Add Space Type**
  SQL> INSERT INTO SpaceTypes (SpaceType) VALUES ('electric');
  Query OK, 1 row affected (0.0060 sec)

- **Update Space Type**
  SQL> UPDATE SpaceTypes SET SpaceType ='handicapped' WHERE SpaceTypeID =2;
  Query OK, 1 row affected (0.0031 sec)
  Rows matched: 1 Changed: 1 Warnings: 1

Operation 2: Maintaining Permits and Vehicle Information

- **Add Vehicle**
  SQL> INSERT INTO Vehicles (CarlicenceNumber, Model, Year, Manufacturer, Color) VALUES ('FDE666', 'Corolla', 2020, 'Toyota', 'Red');
  Query OK, 1 row affected (0.0024 sec)

- **Update Vehicle**
  SQL> UPDATE Vehicles
  SET Model= 'Corolla', Year= 2022, Manufacturer= 'Toyota', Color= 'Green'
  WHERE CarlicenceNumber = 'ABC555';
  Query OK, 1 row affected (0.0026 sec)
  Rows matched: 1 Changed: 1 Warnings: 0

- **Delete Vehicle**
  SQL> DELETE FROM Vehicles WHERE CarlicenseNumber = 'DEF456';
  Query OK, 1 row affected (0.0034 sec)

- **Create Permit**
  SQL> INSERT INTO Permits (PermitID, StartDate, ExpirationDate, ExpirationTime, PermitTypeID, ParkingLotID, ZoneID, SpaceTypeID, UID_Phone, CarlicenseNumber)
  VALUES (1, '2023-10-01', '2023-12-31', '23:59:59', 1, 2, 'CS', 4, 984985678, 'ABC555');
  Query OK, 1 row affected (0.0033 sec)

- **Update Permit**
  SQL> UPDATE Permits
  SET StartDate = '2023-03-22',ExpirationDate = '2023-12-31',

```
ExpirationTime = '10:59:59',PermitTypeID = 1, ZoneID = 'CS',
SpaceTypeID = 4,UID_Phone = 984985678,
CarlicenseNumber = 'ABC555',ParkingLotID = 2
WHERE PermitID = 1;
Query OK, 1 row affected (0.0025 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

- **Delete Permit**

```
SQL> DELETE FROM Permits WHERE PermitID = 1;
Query OK, 1 row affected (0.0030 sec)
```

- **Add Permit Type**

```
SQL> INSERT INTO PermitTypes (PermitType,PermitTypeID)
VALUES ('Residential',4);
Query OK, 1 row affected (0.0027 sec)
```

- **Update Permit Type**

```
SQL>UPDATE PermitTypes
SET PermitType = 'Peak Hours'
WHERE PermitTypeID = 1;
Query OK, 1 row affected (0.0031 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

- **Update Vehicle in Permit**

```
SQL>  UPDATE Permits
SET CarlicenseNumber ='BCD890'
WHERE PermitID = 2;
Query OK, 1 row affected (0.0025 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

- **Update Vehicle of a driver in all Permits**

```
SQL> UPDATE Permits AS P
JOIN Drivers AS DON  P.UID_ Phone= D.UID_ Phone
SET P.CarlicenseNumber = 'DEF456'
WHERE D.UID_Phone = 984985678;
Query OK, 1 row affected (0.0036 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

Operation 3: Generating and Mantaining Citations

- **Create Citation**
  SQL> INSERT INTO Citations (CitationNumber, CitationDate, CitationTime, PaymentStatus, CitationCategoryID, CarlicenseNumber, ParkingLotID, ZoneID, SpaceNumber)
  VALUES (7, '2023-07-01', '18:00:00', 'Unpaid', 2, 'ABC555', 5, 'AS', 9);
  Query OK, 1 row affected (0.0059 sec)

- **Update Citation**
  SQL> UPDATE Citations
  SET CitationCategoryID = 3, CitationDate= '2023-10-15',
  CitationTime = '14:30:00', PaymentStatus = 'Paid'
  WHERE CitationNumber = 20;
  Query OK, 1 row affected (0.0036 sec)
  Rows matched: 1 Changed: 1 Warnings: 0

- **Delete Citation**
  SQL> DELETE FROM Citations
  WHERE CitationNumber = 20;
  Query OK, 1 row affected (0.0049 sec)

- **Add Payment**
  SQL> INSERT INTO Payments (PaymentID, CitationNumber, Amount, Paymentoate, PaymentTime)
  VALUES (22, 2, 50.00, '2023-10-02', '14:30:00');
  Query OK, 1 row affected (0.0034 sec)

- **Delete Payment**
  SQL> DELETE FROM Payments WHERE PaymentID = 1;
  Query OK, 1 row affected (0.0032 sec)

- **Add Appeal**
  SQL> INSERT INTO Appeals (CitationNumber, AppealDescription, AppealStatus) VALUES (7, 'Parking in the wrong zone', 'Under Review');
  Query OK, 1 row affected (0.0030 sec)

- **Update Appeal Status**
  SQL> UPDATE Appeals
  SET AppealStatus = 'Approved' WHERE AppealID = 11;
  Query OK, 1 row affected (0.0032 sec)

Rows matched: 1 Changed: 1 Warnings: 0

- **Delete Appeal**
  SQL> DELETE FROM Appeals WHERE AppeallD =16;
  Query OK, 1 row affected (0.0032 sec)

- **Update Citation CategorylD**
  SQL> UPDATE Citations SET CitationCategorylD = 1
  WHERE CitationNumber =2;
  Query OK, 1 row affected (0.0034 sec)
  Rows matched: 1 Changed: 1 Warnings: 0

- **Add Citation Category**
  SQL> INSERT INTO CitationCategories (CitationCategory, Fee)
  VALUES ('No Permit', 40.00);
  Query OK, 1 row affected (0.0033 sec)

- **Update Citation Category**
  SQL> UPDATE CitationCategories SET CitationCategory =' Handicap Discount - Invalid Permit', Fee= 6.25 WHERE CitationCategorylD = 1;
  Query OK, 1 row affected (0.0034 sec)
  Rows matched: 1 Changed: 1 Warnings: 0

Operation 4: Reports

- **Generate report to return all citations**
  SQL> SELECT c.CitationNumber, c.CitationDate, c.CitationTime, c.PaymentStatus, cc.CitationCategory, c.CarlicenseNumber, pl.Name AS ParkingLotName, c.ZoneID, c.SpaceNumber
  FROM Citations c JOIN CitationCategories cc JOIN ParkingLots pl ON c.ParkingLotID = pl.ParkingLotID and c.CitationCategorylD = cc.CitationCategorylD;

```
MySQL  classdb2.csc.ncsu.edu:3306  naabuzai  SQL > SELECT c.CitationNumber, c.CitationDate, c.CitationTime, c.PaymentStatus, cc.CitationCategory, c.CarLicenseNumber,
l.Name AS ParkingLotName, c.ZoneID, c.SpaceNumber
                                                -> FROM Citations c JOIN CitationCategories cc JOIN ParkingLots pl ON c.ParkingLotID = pl.ParkingLotID and c.Citation
tegoryID = cc.CitationCategoryID;
+----------------+-------------+-------------+---------------+---------------------------------+-----------------+----------------+--------+-------------+
| CitationNumber | CitationDate | CitationTime | PaymentStatus | CitationCategory                | CarLicenseNumber | ParkingLotName | ZoneID | SpaceNumber |
+----------------+-------------+-------------+---------------+---------------------------------+-----------------+----------------+--------+-------------+
|              3 | 2023-03-01  | 14:00:00    | Paid          | Invalid Permit                  | DEF456          | Parking Lot F  | DS     |          11 |
|              4 | 2023-04-01  | 15:00:00    | Paid          | No Permit                       | EFG123          | Parking Lot E  | AS     |           9 |
|              5 | 2023-05-01  | 16:00:00    | Paid          | Handicap Discount - Expired Permit | GHI789          | Parking Lot H  | D      |         103 |
|              6 | 2023-06-01  | 17:00:00    | Paid          | No Permit                       | HIJ456          | Parking Lot C  | C      |          12 |
|              7 | 2023-07-01  | 18:00:00    | Unpaid        | No Permit                       | ABC555          | Parking Lot H  | D      |         103 |
+----------------+-------------+-------------+---------------+---------------------------------+-----------------+----------------+--------+-------------+
5 rows in set (0.0035 sec)
```

- **Create Citation reports in a given time range for all zones in each lot**

  SQL> SELECT c.ParkingLotID, p.Name, p.Address, COUNT(*)
  FROM Citations c JOIN ParkingLots p ON c.ParkingLotID = p.ParkingLotID
  WHERE c.CitationDate BETWEEN '2023-01-01' AND '2023-12-31'
  GROUP BY c.ParkingLotID, p.Name, p.Address;

```
MySQL  classdb2.csc.ncsu.edu:3306  naabuzai  SQL > SELECT c.ParkingLotID, p.Name, p.Address , COUNT(*)
                                                -> FROM Citations c JOIN ParkingLots p ON c.ParkingLotID = p.ParkingLotID
                                                -> WHERE c.CitationDate BETWEEN '2023-01-01' AND '2023-12-31'
                                                -> GROUP BY c.ParkingLotID, p.Name, p.Address ;
+--------------+---------------+-----------------+----------+
| ParkingLotID | Name          | Address         | COUNT(*) |
+--------------+---------------+-----------------+----------+
|            1 | Parking Lot H | 435 Main Street |        2 |
|            3 | Parking Lot C | 789 Oak St      |        1 |
|            5 | Parking Lot E | 202 Maple St    |        1 |
|            6 | Parking Lot F | 303 Birch St    |        1 |
+--------------+---------------+-----------------+----------+
4 rows in set (0.0035 sec)
```

- **Report showing zones along with their corresponding parking lots as a tuple**

  SQL> SELECT ParkingLotID, ZoneID
  FROM Zones
  ORDER BY ParkingLotID, ZoneID;

```
MySQL  classdb2.csc.ncsu.edu:3306  naabuzai  SQL > SELECT ParkingLotID, ZoneID
                                               -> FROM Zones
                                               -> ORDER BY ParkingLotID, ZoneID;
+--------------+--------+
| ParkingLotID | ZoneID |
+--------------+--------+
|            1 | A      |
|            1 | CS     |
|            1 | D      |
|            1 | V      |
|            2 | B      |
|            3 | C      |
|            4 | D      |
|            5 | AS     |
|            6 | BS     |
+--------------+--------+
9 rows in set (0.0033 sec)
```

- **Generate report to get active citation violation count**
  SQL> SELECT COUNT(DISTINCT CarlicenseNumber) AS CitationViolationCount
  FROM Citations
  WHERE PaymentStatus = 'Unpaid';

```
MySQL  classdb2.csc.ncsu.edu:3306  naabuzai  SQL > SELECT COUNT(DISTINCT CarLi
                                                -> FROM Citations
                                                -> WHERE PaymentStatus = 'Unpa
+------------------------+
| CitationViolationCount |
+------------------------+
|                      1 |
+------------------------+
1 row in set (0.0032 sec)
```

- **Generate report to get number of employees having permits for a given parking zone**
  SQL> SELECT COUNT(*) AS EmployeeCount
  FROM Permits P JOIN Drivers DON P.UID_Phone = D.UID_Phone
  WHERE D.Status = 'E' AND ParkingLotID = 3;

```
IBE.!m  classdb2.csc.ncsu.edu:3306  naabuzai  SQL
```

- **Generate report to get permit information for given UID_Phone**
  SQL> SELECT p.PermitID, p.StartDate, p.ExpirationDate, p.ExpirationTime, pt.PermitType, pl.Name AS ParkingLotName, p.ZoneID, st.SpaceType, p.UID_Phone, p.CarLicenceNumber
  FROM Permits p JOIN PermitTypes pt JOIN SpaceTypes st JOIN ParkingLots pl ON p.PermitTypeID = pt.PermitTypeID AND p.SpaceTypeID = st.SpaceTypeID AND p.ParkingLotID = pl.ParkingLotID
  WHERE UID_Phone = 200132323;

```
MySQL  classdb2.csc.ncsu.edu:3306  naabuzai  SQL > SELECT p.PermitID, p.StartDate, p.ExpirationDate, p.ExpirationTime, pt.PermitType, pl.Name
ID, st.SpaceType, p.UID_Phone, p.CarLicenceNumber
                                                -> FROM Permits p JOIN PermitTypes pt JOIN SpaceTypes st JOIN ParkingLots pl ON p.PermitTypeI
SpaceTypeID = st.SpaceTypeID AND p.ParkingLotID = pl.ParkingLotID
                                                -> WHERE UID_Phone = 200132323;
+----------+------------+---------------+---------------+------------+---------------+--------+-----------+-----------+------------------+
| PermitID | StartDate  | ExpirationDate | ExpirationTime | PermitType | ParkingLotName | ZoneID | SpaceType | UID_Phone | CarLicenceNumber |
+----------+------------+---------------+---------------+------------+---------------+--------+-----------+-----------+------------------+
|        1 | 2023-10-01 | 2023-12-31    | 23:59:59      | Peak Hours | Parking Lot C | C      | regular   | 200132323 | ABC555           |
+----------+------------+---------------+---------------+------------+---------------+--------+-----------+-----------+------------------+
1 row in set (0.0040 sec)
```

- **Generate report to get an available space number in a given parking lot for a given space type**

  SQL> SELECT SpaceNumber
  FROM Spaces S JOIN SpaceTypes ST ON S.SpaceTypeID = ST.SpaceTypeID
  WHERE ParkingLotID = 6 AND SpaceType = 'regular' AND AvailabilityStatus = 'Available' LIMIT 1;

  classdb2.csc.ncsu.edu:3306 naabuzai SQL

Query 1:Generate report to get permit information for given UID_Phone

(1) SQL Query

SQL> EXPLAIN SELECT* FROM Permits WHERE UID_Phone = 200132323;

(2) Execution Plan (without indexing)

```
MySQL classdb2.csc.ncsu.edu:3306 naabuzai SQL > EXPLAIN SELECT * FROM Permits WHERE UID_Phone = 200132323;
+----+-------------+---------+------+---------------+---------------+---------+-------+------+-------+
| id | select_type | table   | type | possible_keys | key           | key_len | ref   | rows | Extra |
+----+-------------+---------+------+---------------+---------------+---------+-------+------+-------+
|  1 | SIMPLE      | Permits | ref  | Permits_ibfk_4 | Permits_ibfk_4 | 5       | const |    1 |       |
+----+-------------+---------+------+---------------+---------------+---------+-------+------+-------+
1 row in set (0.0025 sec)
```

(3) Index-creation statement

SQL> CREATE INDEX uidPhoneIndex ON Permits(UID_Phone);

(4) Execution Plan (after indexing)

```
MySQL classdb2.csc.ncsu.edu:3306 naabuzai SQL > CREATE INDEX uidPhoneIndex ON Permits(UID_Phone);
Query OK, 0 rows affected (0.0218 sec)

Records: 0  Duplicates: 0  Warnings: 0
MySQL classdb2.csc.ncsu.edu:3306 naabuzai SQL > EXPLAIN SELECT * FROM Permits WHERE UID_Phone = 200132323;
+----+-------------+---------+------+---------------+---------------+---------+-------+------+-------+
| id | select_type | table   | type | possible_keys | key           | key_len | ref   | rows | Extra |
+----+-------------+---------+------+---------------+---------------+---------+-------+------+-------+
|  1 | SIMPLE      | Permits | ref  | uidPhoneIndex | uidPhoneIndex | 5       | const |    1 |       |
+----+-------------+---------+------+---------------+---------------+---------+-------+------+-------+
1 row in set (0.0023 sec)
```

Query 2: Generate report to return all citations

(1) SQL Query

SQL> EXPLAIN SELECT ParkingLotID, ZoneID FROM Zones ORDER BY ParkingLotID, ZoneID;

(2) Execution Plan (without indexing)

```
MySQL  classdb2.csc.ncsu.edu:3306  naabuzai  SQL > EXPLAIN SELECT ParkingLotID, ZoneID
                                               -> FROM Zones
                                               -> ORDER BY ParkingLotID, ZoneID;
+----+-------------+-------+-------+---------------+--------------+---------+------+------+-------------+
| id | select_type | table | type  | possible_keys | key          | key_len | ref  | rows | Extra       |
+----+-------------+-------+-------+---------------+--------------+---------+------+------+-------------+
|  1 | SIMPLE      | Zones | index | NULL          | Zones_ibfk_1 | 6       | NULL |    9 | Using index |
+----+-------------+-------+-------+---------------+--------------+---------+------+------+-------------+
1 row in set (0.0035 sec)
```

(3) Index-creation statement

SQL> CREATE INDEX citationIndex ON Citations(CitationNumber);

(4) Execution Plan (after indexing)

```
MySQL  classdb2.csc.ncsu.edu:3306  naabuzai  SQL > explain SELECT ParkingLotID, ZoneID
                                               -> FROM Zones
                                               -> ORDER BY ParkingLotID, ZoneID;
+----+-------------+-------+-------+---------------+--------------+---------+------+------+-------------+
| id | select_type | table | type  | possible_keys | key          | key_len | ref  | rows | Extra       |
+----+-------------+-------+-------+---------------+--------------+---------+------+------+-------------+
|  1 | SIMPLE      | Zones | index | NULL          | ParkingIndex | 6       | NULL |    9 | Using index |
+----+-------------+-------+-------+---------------+--------------+---------+------+------+-------------+
1 row in set (0.0030 sec)
```

4.3

y $Gammala = Selectionlp = Renameloo_8 = Theta]oinlTI = Projection$

1. **Generate a report to get permit information for a given UID_Phone.**

SQL Query:

SQL> SELECT p.PermitID, p.StartDate, p.ExpirationDate, p.ExpirationTime, pt.PermitType, pl.Name AS ParkingLotName, p.ZoneID, st.SpaceType, p.UID_Phone, p.CarlicenceNumber
FROM Permits p JOIN PermitTypes pt JOIN SpaceTypes st JOIN ParkingLots pl

ON p.PermitTypelD = pt.PermitTypelD AND p.SpaceTypelD = st.SpaceTypelD
AND p.ParkinglotlD = pl.ParkinglotlD
WHERE UID_Phone = 200132323;

Relational Algebra:

$$\pi_{p.Permit!D,\ p.StartDate,\ p.ExpirationDate,\ p.ExpirationTime,\ pt.PermitType,\ p.UID\_Phone,\ p.CarLicenseNumber,\ p.Zone!D,\ pl.ParkingLotName,\ st.SpaceType}($$

$$\rho_{p.Permit!D,\ p.StartDate,\ p.ExpirationDate,\ p.ExpirationTime,\ pt.PermitType,\ p.UID\_Phone,\ p.CarLicenseNumber,\ p.Zone!D,\ pl.ParkingLotName,\ st.SpaceType}($$

$$(a(UID\_Phone=200132323)$$

$$(\rho_{p}(Permits) \bowtie_{p.PermitTypelD=pt.PermitTypeID} \rho_{pt}(PermitTypes)$$

$$\bowtie_{p.SpaceTypel\ D=st.SpaceTypelD} \rho_{st}(SpaceTypes)$$

$$\bowtie_{p.ParkingLotl\ D=pl.Parking\ Lot!D} \rho_{pl}(ParkingLots))))$$

Correctness Proof:

Suppose p is any tuple in Permits relation, pt is any tuple in PermitType, st is any tuple in SpaceType relation and pl is any tuple in Parkinglots table such that p.PermitTypelD and pt.PermitTypelD are equal, p.SpaceTypelD and st.SpaceTypelD are same and finally p.ParkinglotlD and pl.ParkinglotlD are also equal. The combination of p, pt ,st and pl gives the permit information for a given UID_Phone.The query returns the permit information like PermitlD, StartDate, ExpirationDate, ExpirationTime, UID_Phone, CarlicenseNumber, permit type information like PermitType, parking lot information like ParkinglotName, ZoneId, and space type information like SpaceType for a given UID_phone.This is exactly what we are required to retrieve.

2. **Generate report to get number of employees having permits for a given parking zone**

SQL Query:

```
SQL> SELECT COUNT(*) AS EmployeeCount
     FROM Permits P JOIN Drivers D ON
     P.UID_ Phone= D.UID_ Phone
     WHERE D.Status = 'E' AND ParkinglotID = 4;
```

Relational Algebra:

$$TT_{EmployeeCount}(p(E_{mp\ oyee}C_{oun}\ t)(yCOUNT(*)(\ CJ(Status='£'\ AND\ Parkinglot1D='4')$$

$$(p_{P}(Permits)oo_{P.UID\_?hone=D.UID\_Phone}\ p_{D}(Drivers))))$$

Correctness Proof:

Suppose p is any tuple in Permits and d in any tuple in Drivers such that the values of p.UID_Phone and d.UID_Phone are equal. The combination of the tuples p and d gives the permit information of a driver. This query returns the count of the tuples whose driver has the requested status and for a requested parking lot based on the driver's UID_Phone being equal to permit's UID_Phone. This is exactly what we are required to retrieve.