F28DM COURSEWORK 1

Database Design and Implementation

Group 17

Aashika Joshi (H00405238)

Ishwarya Gowri Sankar (H00403105)

Joshua Jacob (H00404266)

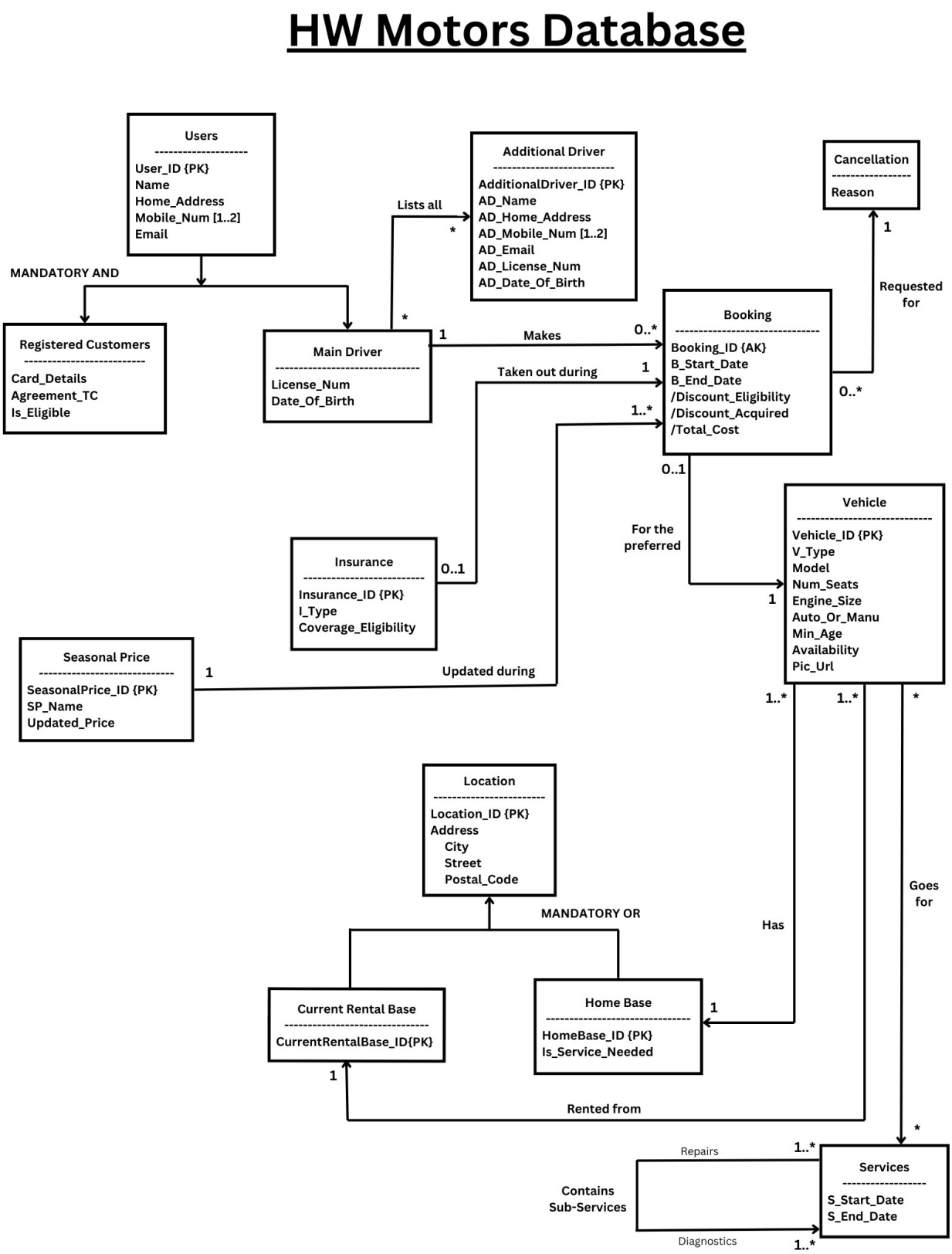
Keerthana B Nair (H00421150)

Rushaan Sahrush (H00376415)

Swapna Manikandan (H00396446)

TABLE OF CONTENTS

Task 1: Conceptual Model – Entity Relationship Diagram



NOTES:

⇒ The ER Diagram depicts a total of 13 entities. They are segregated as follows: -

>Strong Entities

Users
Additional Driver
Vehicle
Seasonal Price
Insurance
Location
Current Rental Base
Home Base

>Weak entities

Registered Customers
Main Driver
Booking
Cancellation
Services

⇒ There are total of 12 relationships segregated as follows: -

1 recursive relationship (Services contains sub-services)
1 (mandatory, and) specialisation
1 (mandatory, or) generalisation
9 general relationships

- Data from the **Users** entity is inherited by the **Registered Customers** entity and **Main Driver** entity**,** forming a **(mandatory, and) specialisation relationship** in the database.

- It is mandatory for a user in the **Users** entity to be a **registered customer** and **a main driver**, ensuring that no user can make a booking as the main driver without prior registration, mitigating unauthorised bookings.

- In the **Registered Customer** entity, the **Is_Eligible Boolean** attribute distinguishes between individual and commercial users during registration. If a user represents a commercial entity, they are automatically deemed ineligible to make bookings.

- The **Mobile_Num** in **Users** entity and the **AD_Mobile_Num** in the **Additional Driver** entity are the **repeating attributes**, storing one to two mobile numbers.

- In the **Seasonal price** entity, the **Updated_Price** attribute reflects the current price applied for a vehicle during a particular season, dynamically changing with the current season. The **Booking** entity will refer to this and update the vehicle price on the website as and when required.

- In the **Booking** entity the B_Start_Date, B_End_Date and User_ID (from the **User** entity**)** combine to make the **composite primary key**. The **Booking** attribute in the entity is an **alternate key**, which is further used in the **Cancellation** entity.

- The **Discount_Eligibilty** attribute in the **Booking** entity is a **derived attribute**, derived from **B_Start_Date** and **B_End_Date** in the **Booking** entity. If the duration for the vehicle hire is for 7 days or more, the Discount_Eligibilty attribute will be true, indicating that the booking is eligible to receive a discount, else if will remain false, indicating that no discount can be provided.

- The **Discount_Acquired** attribute in the **Booking** entity is a **derived attribute**, derived by calculating 20% of the **Updated_Price attribute** in the **Seasonal Price entity**. This attribute calculates the discount amount acquired by a specific booking, remaining null if Discount_Eligibility is false.

- The **Total_Cost** attribute in the **Booking** entity is a **derived attribute**, derived by subtracting the **Updated_Price** attribute in the **Seasonal Price** entity and **Discount_Acquired** attribute in the **Booking** entity. This attribute indicates the final price the main driver must pay to confirm the booking.

- Data from the **Home Base** entity and **Current Rental Base** entity is generalised into the **Location** entity, forming a **(mandatory, or) generalisation relationship** in the database.

- It is mandatory for a location coordinate in the **Location** entity to either exist as a **home base location** or a **current rental base location**. This is implemented to keep a record of the home base for each vehicle and to keep a track of the current location of the vehicle, while also associating a location coordinate to each service base in the Services entity.

- The **Address** attribute in the **Location** entity is a **composite attribute** consisting of sub-attributes City, Street and Postal_Code.

- Every vehicle in the **Vehicle** entity has a home base recorded by the **Home Base** entity.

- Every vehicle in the **Vehicle** entity has a current location where it resides presently at, recorded by the **Current Rental Base** entity.

- If the **HomeBase_ID** attribute in **Home Base** entity and **CurrentRentalBase_ID** attribute in **Current Rental Base** entity coincide when associated with a vehicle in the **Vehicle** entity, it implies that the vehicle has been returned to its home base. This ensures that only vehicles located at their home base receive servicing.

- If a vehicle resides at its home base, only then is it eligible for servicing. The **Services** entity provides **2 types of services**, first being **repairs** which contain sub-services engine repairs, tire repairs and the second being **diagnostics** which contain sub-services engine diagnostics, electrical diagnostics. This implements the **recursive relationship** for the **Services** entity.

Task 2: Translation into Relational Schema

Strong entities:

Vehicle(Vehicle_ID:int(10) ,
HomeBase_ID: int(10),
CurrentRentalBase_ID: int(10),
V_Type: varchar (30),
Model: varchar(30),
Num_Seats: int(10),
Engine_Size: float,
Auto_Or_Manu: [A|M],
Min_Age: int(3),
Availability: [Y|N],
Pic_Url: varchar(255)
FOREIGN KEY (HomeBase_ID) REFERENCES Home Base(HomeBase_ID),
FOREIGN KEY (CurrentRentalBase_ID) REFERENCES Current Rental Base(CurrentRentalBase_ID)
);

SeasonalPrice(SeasonalPrice_ID: int (10),
SP_Name: varchar(100),
Vehicle_ID: int(10),
Updated_Price: decimal(10,2)
FOREIGN KEY (Vehicle_ID) REFERENCES Vehicle(Vehicle_ID)
);

Insurance(Insurance_ID: int (10),
I_Type: varchar(30),
Coverage_Eligibility: [Y|N]);

Additional Driver(AdditionalDriver_ID: int (10),
AD_Name: varchar(100),
AD_Home_Address: varchar(255),
AD_Email: varchar(255),
AD_License_Num: varchar(30),
AD_Date_Of_Birth: date);

Mandatory AND:

Users(User_ID: int(10),
Name: varchar(100),
Home_Address: varchar(255),
Email: varchar(255),
Registered CustomersFlag: [Y|N],

Main DriverFlag: [Y|N],
Card_Details: int(30),
Agreement_TC: [Y|N],
Is_Eligible: [Y|N],
License_Num: int(30),
Date_Of_Birth: date,
AdditionalDriver_ID: int(10)
FOREIGN KEY (AdditionalDriver_ID) REFERENCES
AdditionalDriver(AdditionalDriver_ID));

Mandatory OR:

Current Rental Base(Location_ID: int(10),
City: varchar (30),
Street: varchar(30),
Postal_Code: int (10),
CurrentRentalBase_ID: int (10)
FOREIGN KEY (Location_ID) REFERENCES Location(Location_ID));

Home Base(Location_ID: int(10),
City: varchar(30),
Street: varchar(30),
Postal_Code: int(10),
HomeBase_ID: int (10),
Is_Service_Needed: [Y|N]
FOREIGN KEY (Location_ID) REFERENCES Location(Location_ID));

Weak entities:

Registered Customers Mobile_Num(Registered_Customers_User_ID: int(10),
Registered_Customers_Mobile_Num: int(30));

Main Driver Mobile_Num(Main_Driver_User_ID: int(10),
Main_Driver_Mobile_Num: int(30));

Additional Driver Mobile_Num(AdditionalDriver_ID: int(10), AD_Mobile_Num: int(30)**);**

Booking(Booking_ID: int (10),
User_ID : int (10),
B_Start_Date: date,
B_End_Date: date,
Vehicle_ID: int (10),
Insurance_ID: int(10),
AdditionalDriver_ID: int(10),
SeasonalPrice_ID: int(10),

UNIQUE (Booking_ID)
FOREIGN KEY (User_ID) REFERENCES User(User_ID),
FOREIGN KEY (Vehicle_ID) REFERENCES Vehicle(Vehicle_ID),
FOREIGN KEY (Insurance_ID) REFERENCES Insurance(Insurance_ID),
FOREIGN KEY (AdditionalDriver_ID) REFERENCES
AdditionalDriver(AdditionalDriver_ID),
FOREIGN KEY (SeasonalPrice_ID) REFERENCES SeasonalPrice(SeasonalPrice_ID));

Cancellation(Booking_ID: int(10),
Reason: varchar(100)
FOREIGN KEY (Booking_ID) REFERENCES Booking(Booking_ID));

Services(Location_ID: int(10),
Vehicle_ID: int(10),
S_Start_Date: date,
S_End_Date: date
FOREIGN KEY (Location_ID) REFERENCES Location(Location_ID),
FOREIGN KEY (Vehicle_ID) REFERENCES Vehicle(Vehicle_ID));

NOTES:

1. In the Vehicle table, both **CurrentRentalBase_ID** and **HomeBase_ID** serve as foreign keys for precise tracking of the vehicle's current location and designated home base, respectively.

2. In the Main Driver table, **AdditionalDriver_ID** functions as a foreign key to keep track of any additional drivers linked with the main driver for a given booking.

3. The Seasonal Price Table employs **Vehicle_ID** as a foreign key to assign specific seasonal prices to corresponding vehicles.

4. The Cancellation Table utilizes **Booking_ID** as a foreign key to document cancellations associated with specific bookings.

5. The Current Rental Base table incorporates **Location_ID** as a foreign key to record the location coordinates corresponding to the vehicle's current rental base.

6. In the Home Base table, **Location_ID** functions as a foreign key to record the location coordinates assigned to the vehicle's designated home base.

7. In the Booking table, the foreign key associations are as follows:
   -**User_ID** ensures a one-to-one mapping between users and specific bookings.
   -**Vehicle_ID** links users with their preferred vehicle for a booking.
   -**Insurance_ID** correlates the preferred insurance type with the user's chosen vehicle for a booking.
   -**AdditionalDriver_ID** tracks any additional drivers accompanying the main driver in a booking.
   -**SeasonalPrice_ID** maintains a record of the current seasonal price applicable to the vehicle during the booking period.

8. In the Services table, foreign key associations are as follows:
   -**Location_ID** corresponds to the location coordinates of the service centers.
   -**Vehicle_ID** serves as the foreign key linking specific vehicles to their respective service centers.

9. The Derived Attributes are **Discount_Eligibility**, **Discount_Acquired** and **Total_Cost** in the Booking Table.

10. Composite Attribute: The **Address** attribute in Location entity is a composite attribute consisting of sub-attributes **City**, **Street** and **Postal_Code**.

11. Composite Primary Key: In the Booking entity the **B_Start_Date**, **B_End_Date** and **User_ID** (from the User entity) combine to make the composite primary key.

12. **Booking_ID** functions as an alternate key in the Cancellation table, providing a secondary means of uniquely identifying canceled bookings alongside the composite primary key.

13. **Mobile_Num[1..2]** is a multi-valued attribute in the User's Table and AD_Mobile_Num[1..2] in the Additional Driver Table.

    The multi-valued attributes - **Registered Customers_Mobile_Num**, **Main Driver_Mobile_Num**, and **Additional Driver_Mobile_Num** have been segregated into separate tables using normalization. These attributes have been derived from Mobile_Num attribute in the tables Users and Additional Driver. Accessible via User_ID, which has been renamed for clarity: Registered_Customers_User_ID, Main_Driver_User_ID, and AdditionalDriver_ID

14. **Mandatory AND** Relationship

    The Users table serves as a superclass, incorporating attributes mandatory for all user registrations. User_ID, Name, Home_Address, Email, License_Num, and Date_Of_Birth, Agreement_TC, Card_Details and Is_Eligible are important attributes for user identification and contact information. Additionally, the presence of flags such as Registered CustomersFlag and Main DriverFlag signify that both conditions must be met for certain privileges. The common attributes from Registered Customers and Main Driver have been included in the superclass Users.

15. **Mandatory OR** Relationship

    There are two tables representing locations: Current Rental Base and Home Base. Both tables share the same attributes from the Location Table: Location_ID, City, Street, and Postal_Code.This indicates that a location can be either a current rental base or a home base. Each location must be categorized as one or the other; it cannot be both simultaneously. This relationship ensures that every location in the system is assigned a specific role, either acting as a current rental base or as a home base.

DATA DICTIONARIES:

VEHICLE

Attribute	Description	Domain	Null?	Primary Key	Foreign Key
Vehicle_ID	Unique identifier of the vehicle.	int(10)	**N**	**Y**	
HomeBase_ID	Identifier of the vehicle's home base.	int(10)	**N**	**N**	homebase. homebase_ID
CurrentRentalBase_ID	Identifier of where the vehicle is currently located at.	int(10)	**N**	**N**	currentrentalbase.currentrentalbase_ID
V_Type	Type of vehicle the user is hiring.	varchar(30)	**N**	**N**	
Model	Model of the vehicle.	varchar(30)	**N**	**N**	
Num_Seats	Number of seats in the vehicle.	int(10)	**N**	**N**	
Engine_Size	Engine size of the vehicle.	float	**N**	**N**	
Auto_Or_Manu	If the vehicle is automatic or manual.	enum	**N**	**N**	
Min_Age	The minimum age required to book the specific vehicle.	int(3)	**N**	**N**	
Availability	If the car is currently available	boolean	**N**	**N**	
Pic_Url	The URL of the vehicle's picture	varchar(255)	**N**	**N**	

SEASONAL PRICE

Attribute	Description	Domain	Null?	Primary Key	Foreign Key
SeasonalPrice_ID	Unique Identifier for the seasonal price.	int(10)	**N**	**Y**	
SP_Name	Name of the season.	varchar(100)	**N**	**N**	
Vehicle_ID	Identifier of the vehicle associated with the seasonal price.	int(10)	**N**	**N**	vehicle. vehicle_ID
Updated_Price	Updated price for the seasonal period.	decimal(10,2)	**N**	**N**	

INSURANCE

Attribute	Description	Domain	Null?	Primary Key	Foreign Key
Insurance_ID	Unique Identifier for the insurance of the user.	int(10)	**N**	**Y**	
I_Type	Type of insurance.	varchar(30)	**N**	**N**	
Coverage_Eligibility	If the user is eligible for insurance coverage.	boolean	**N**	**N**	

ADDITIONAL DRIVER

Attribute	Description	Domain	Null?	Primary Key	Foreign Key
AdditionalDriver_ID	Unique Identifier for the Additional Driver.	int(10)	**N**	**Y**	
AD_Name	Name of the Additional Driver.	varchar(100)	**N**	**N**	
AD_Home_Address	Home Address of the additional driver.	varchar(255)	**N**	**N**	
AD_Email	Email of the Additional Driver.	varchar(255)	**N**	**N**	

AD_License_Num	License number of the Additional Driver.	varchar(30)	**N**	**N**	
AD_Date_Of_Birth	Date of birth of the Additional Driver.	date	**N**	**N**	

USERS

Attribute	**Description**	**Domain**	**Null?**	**Primary Key**	**Foreign Key**
User_ID	Unique Identifier for the user.	int(10)	**N**	**Y**	
Name	Name of the user.	varchar(100)	**N**	**N**	
Home_Address	Home Address of the user.	varchar(255)	**N**	**N**	
Email	Email of the user.	varchar(255)	**N**	**N**	
Registered CustomersFlag	Flag to check if the attribute is a part of the Registered Customers table.	boolean	**N**	**N**	
Main DriverFlag	Flag to check if the attribute is a part of the Main Driver table.	boolean	**N**	**N**	
Card_Details	Card details of the user.	int(30)	**N**	**N**	
Agreement_TC	Terms and conditions to be accepted by the user.	boolean	**N**	**N**	
Is_Eligible	To check if the user is part of the public or is a corporate firm.	boolean	**N**	**N**	
License_Num	License Number of the main driver.	int(30)	**N**	**N**	
Date_Of_Birth	Date of birth of the main driver.	date	**N**	**N**	
AdditionalDriver_ID	Identifier for the additional driver.	int(10)	**N**	**Y**	additionaldriver.additionaldriver_ID

CURRENT RENTAL BASE

Attribute	Description	Domain	Null?	Primary Key	Foreign Key
Location_ID	Unique Identifier for the vehicle's location.	int(10)	**N**	**Y**	location. location_ID
City	City name where the user lives.	varchar(30)	**N**	**N**	
Street	Street name.	varchar(30)	**N**	**N**	
Postal_Code	Postal Code of the user.	int(10)	**N**	**N**	
CurrentRentalBase _ID	Unique Identifier of where the vehicle is currently located at.	int(10)	**N**	**Y**	

HOME BASE

Attribute	Description	Domain	Null?	Primary Key	Foreign Key
Location_ID	Identifier of the location.	int(10)	**N**	**Y**	location. location_ID
City	Name of the city.	varchar(30)	**N**	**N**	
Street	Name of the street.	varchar(30)	**N**	**N**	
Postal_Code	Postal code of the city.	int(10)	**N**	**N**	
HomeBase_ID	Unique Identifier for the home base.	int(10)	**N**	**Y**	
Is_Service_Needed	If the service is required or not at the home base.	boolean	**N**	**N**	

REGISTERED CUSTOMERS MOBILE_NUM

Attribute	Description	Domain	Null?	Primary Key	Foreign Key
Registered_Customers_User_ID	User ID of the registered customers.	int(10)	**N**	**N**	
Registered_Customers_Mobile_ Num	Mobile number(s) of the registered customers.	int(30)	**N**	**N**	

MAIN DRIVER MOBILE_NUM

Attribute	Description	Domain	Null?	Primary Key	Foreign Key
Main_Driver_User_ID	User ID of the main driver.	int(10)	**N**	**N**	
Main_Driver_Mobile_Num	Mobile number(s) of the main driver.	int(30)	**N**	**N**	

ADDITIONAL DRIVER MOBILE_NUM

Attribute	Description	Domain	Null?	Primary Key	Foreign Key
AdditionalDriver_ID	User ID of the additional driver.	int(10)	**N**	**N**	
AD_Mobile_Num	Mobile number(s) of the additional driver.	int(30)	**N**	**N**	

BOOKING

Attribute	Description	Domain	Null?	Primary Key	Foreign Key
Booking_ID	Unique identifier for the booking.	int(10)	**N**	**N**	
User_ID	Unique identifier for the user.	int(10)	**N**	**Y**	user.user_ID
B_Start_Date	Start date of the booking.	date	**N**	**Y**	
B_End_Date	End date of the booking.	date	**N**	**Y**	
Vehicle_ID	Identifier of the vehicle.	int(10)	**N**	**N**	vehicle.vehicle_ID
Insurance_ID	Identifier of the insurance.	int(10)	**N**	**N**	insurance.insurance_ID

AdditionalDriver_ID	Identifier of the additional driver.	int(10)	**N**	**N**	additionaldriver.additionaldriver_ID
SeasonalPrice_ID	Identifier of the seasonal price.	int(10)	**N**	**N**	seasonalprice.seasonalprice_ID

CANCELLATION

Attribute	**Description**	**Domain**	**Null?**	**Primary Key**	**Foreign Key**
Booking_ID	Identifier for the booking for cancellation.	int(10)	**N**	**N**	booking.booking_ID
Reason	Reason for booking cancellation.	varchar(100)	**N**	**N**	

SERVICES

Attribute	**Description**	**Domain**	**Null?**	**Primary Key**	**Foreign Key**
Location_ID	Identifier of the location.	int(10)	**N**	**N**	location.location_ID
Vehicle_ID	Identifier of the Vehicle.	int(10)	**N**	**N**	vehicle.vehicle_ID
S_Start_Date	Start Date of the service.	date	**N**	**N**	
S_End_Date	End Date of the Service.	date	**N**	**N**	

Task 3: Implementation of the Schema in MariaDB

```
CREATE DATABASE HW_Motors;
USE HW_Motors;

/*Stores information about users.
Each user is assigned a unique identifier (User_ID) which auto-increments with each new entry. */
CREATE TABLE Users (
User_ID INT(10) AUTO_INCREMENT PRIMARY KEY NOT NULL, /*Unique Identifier for each users */
Name VARCHAR (100) NOT NULL, /* Name of the user, cannot be NULL */
Home_Address VARCHAR (255) NOT NULL, /* Home address of the user, cannot be NULL*/
Mobile_Num1 INT(30) NOT NULL, /* Primary mobile number of the user, cannot be NULL */
Mobile_Num2 INT(30), /* Mobile Number 2 is optional for the users so there is no default value in this case */
Email VARCHAR (255) NOT NULL /* Email address of the user, cannot be NULL */
);

/*Stores information about the registered customers*/
CREATE TABLE RegisteredCustomers (
User_ID INT(10), /* Foreign key referencing the User_ID from the Users table */
Card_Details VARCHAR(30) NOT NULL, /* Stores the card details of the customer, cannot be NULL */
Agreement_TC BOOLEAN DEFAULT TRUE NOT NULL, /* Represents whether the customer has agreed to the terms and conditions; default is TRUE */
Is_Eligible BOOLEAN DEFAULT TRUE NOT NULL /* Indicates whether the customer is part of the public or a commercial business; default is TRUE indicating the customer is assumed to be part of the public */
);

/*Stores information about the main driver*/
CREATE TABLE MainDriver (
User_ID INT(10), /* Foreign key referencing the User_ID from the Users table */
License_Num VARCHAR(30) NOT NULL, /* License number of the main driver, cannot be NULL */
Date_Of_Birth DATE NOT NULL, /* Date of birth of the main driver, cannot be NULL */
AdditionalDriver_ID INT(10) /* Foreign key referencing the AdditionalDriver_ID from the AdditionalDrivers table */
);
```

```
/*Stores information about the additional driver.
Each additional driver has a unique identifier (AdditionalDriver_ID) which auto-increments with each new entry. */
CREATE TABLE AdditionalDriver (
   AdditionalDriver_ID INT(10) AUTO_INCREMENT PRIMARY KEY NOT NULL, /* Unique identifier for each additional driver */
   AD_Name VARCHAR(100) NOT NULL, /* Name of the additional driver, cannot be NULL */
   AD_Home_Address VARCHAR(255) NOT NULL, /* Home address of the additional driver, cannot be NULL */
   AD_Mobile_Num1 INT(30) NOT NULL, /* Primary mobile number of the additional driver, cannot be NULL */
   AD_Mobile_Num2 INT(30), /* Mobile Number 2 is optional for the users so there is no default value in this case */
   AD_Email VARCHAR(255) NOT NULL, /* Email address of the additional driver, cannot be NULL */
   AD_License_Num VARCHAR(30) NOT NULL, /* License number of the additional driver, cannot be NULL */
   AD_Date_Of_Birth DATE NOT NULL /* Date of birth of the additional driver, cannot be NULL */
);

/*Stores information about the bookings.
Each Booking has a unique identifier (Booking_ID ) which auto-increments with each new entry. */
CREATE TABLE Booking (
Booking_ID INT(10) AUTO_INCREMENT NOT NULL UNIQUE, /* Unique identifier for each booking, it is an alternate key not a primary key */
User_ID INT(10), /* Foreign key referencing the User_ID from Users table*/
B_Start_Date DATE NOT NULL, /* Start date of the booking, cannot be NULL */
B_End_Date DATE NOT NULL, /* End date of the booking, cannot be NULL */
PRIMARY KEY(User_ID,B_Start_Date,B_End_Date), /* Composite primary key to identify a booking*/
Vehicle_ID INT(10),  /* Foreign key referencing the Vehicle_ID from the Vehicle table */
Insurance_ID INT(10),  /* Foreign key referencing the Insurance_ID from the Insurance table */
AdditionalDriver_ID INT(10), /* Foreign key referencing the AdditionalDriver_ID from the AdditionalDrivers table */
SeasonalPrice_ID INT(10), /* Foreign key referencing the SeasonalPrice_ID from the SeasonalPrice table */
Discount_Eligibility BOOLEAN DEFAULT FALSE NOT NULL, /*Eligible for discount if booking is for more than 7 days, default is FALSE */
Discount_Acquired INT DEFAULT NULL, /* Calculated discount value (Updated_Price x 20%) . It can be set to null as the discount is applied only if its more than 7 days */
Total_Cost INT NOT NULL /* Total cost of the booking (Updated_Price) - (Discount Acquired) */
);
```

```
/*Stores information about the cancellations.
Each entry in this table records the cancellation reason for a specific booking.*/
CREATE TABLE Cancellation (
Booking_ID INT(10), /* Foreign key referencing the Booking_ID from the Bookings table */
Reason VARCHAR(100) /* Reason for cancellation*/
);

/*Stores information about the insurance.
Each insurance option is assigned a unique identifier (Insurance_ID) which auto-increments with each new entry.*/
CREATE TABLE Insurance (
Insurance_ID INT(10) AUTO_INCREMENT PRIMARY KEY NOT NULL, /* Unique identifier for each insurance option */
I_Type VARCHAR(30), /* Type of insurance */
Coverage_Eligibility BOOLEAN DEFAULT TRUE NOT NULL /* Indicates whether the insurance coverage is possible or not (default: TRUE) */
);

/*Stores information about the Vehicles.
Each vehicle is assigned a unique identifier (Vehicle_ID) which auto-increments with each new entry.*/
CREATE TABLE Vehicle (
Vehicle_ID INT(10) AUTO_INCREMENT PRIMARY KEY NOT NULL, /* Unique identifier for each vehicle */
HomeBase_ID INT(10), /* Foreign key referencing the HomeBase_ID from the HomeBase table */
CurrentRentalBase_ID INT(10), /* Foreign key referencing the CurrentRentalBase_ID from the RentalBase table */
V_Type VARCHAR(30) NOT NULL, /* Type of vehicle, cannot be NULL */
Model VARCHAR(30) NOT NULL, /* Model of the vehicle, cannot be NULL */
Num_Seats INT(10) NOT NULL, /* Number of seats in the vehicle, cannot be NULL */
Engine_Size FLOAT NOT NULL, /* Size of the vehicle's engine, cannot be NULL */
Auto_Or_Manu ENUM('A', 'M'),  /* Indicates whether the vehicle has automatic (A) or manual (M) transmission */
Min_Age INT(3) NOT NULL CHECK (Min_Age >= 25), /* Minimum age requirement for renting the vehicle, must be 25 or older */
Availability BOOLEAN DEFAULT TRUE NOT NULL, /* Indicates whether the vehicle is available for rent (default: TRUE) */
Pic_URL VARCHAR(255) /* URL of a picture of the vehicle */
);

/*Stores information about the Seasonal Price.
Each SeasonalPrice is assigned a unique identifier (SeasonalPrice_ID ) which auto-increments with each new entry.*/
CREATE TABLE SeasonalPrice (
SeasonalPrice_ID INT(10) AUTO_INCREMENT PRIMARY KEY NOT NULL, /* Unique identifier for each seasonal price, cannot be NULL */
SP_Name VARCHAR(100) NOT NULL, /* Name of the season, cannot be NULL */
Vehicle_ID INT(10), /* Foreign key referencing the User_ID from the Users table */
```

```
Updated_Price DECIMAL(10,2) NOT NULL /* Updated current price for the seasonal period, cannot be NULL */
);

/*Stores information about the Location.
Each location is assigned a unique identifier (Location_ID) which auto-increments with each new entry.*/
CREATE TABLE Location (
Location_ID INT(10) AUTO_INCREMENT PRIMARY KEY NOT NULL, /* Unique identifier for each location */
City VARCHAR(30) NOT NULL, /* City of the location, cannot be NULL */
Street VARCHAR(30) NOT NULL, /* Street of the location, cannot be NULL */
Postal_Code VARCHAR(30) NOT NULL  /* Postal code of the location, cannot be NULL */
);

/*Stores information about the Home Base.
Each Homel base is assigned a unique identifier (HomeBase_ID ) which auto-increments with each new entry.*/
CREATE TABLE HomeBase (
HomeBase_ID INT(10) AUTO_INCREMENT PRIMARY KEY NOT NULL, /* Unique identifier for each home base, cannot be NULL */
Location_ID INT(10), /* Foreign key referencing the Location_ID from the Location table */
Is_Service_Needed BOOLEAN DEFAULT FALSE NOT NULL /* Indicates whether service is needed at the home base, default is FALSE */
);

/*Stores information about the Rental Base.
Each rental base is assigned a unique identifier (CurrentRentalBase_ID) which auto-increments with each new entry.*/
CREATE TABLE CurrentRentalBase (
CurrentRentalBase_ID INT(10) AUTO_INCREMENT PRIMARY KEY NOT NULL,  /* Unique identifier for each rental base */
Location_ID INT(10) /* Foreign key referencing the Location_ID from the Location table */
);

/*Stores information about the Vehicle servicing*/
CREATE TABLE Services (
Location_ID INT(10), /* Foreign key referencing the Location_ID from the Location table */
Vehicle_ID INT(10), /* Foreign key referencing the Vehicle_ID from the Vehicle table */
S_Start_Date DATE NOT NULL, /* Start date of the service, cannot be NULL */
S_End_Date DATE NOT NULL /* End date of the service, cannot be NULL */
);
```

```
/* Adding foreign key constraints after creating tables because multiple tables depend on each other*/

ALTER Table RegisteredCustomers ADD Foreign KEY(User_ID) REFERENCES Users(User_ID);
ALTER Table MainDriver ADD FOREIGN KEY(User_ID) REFERENCES Users(User_ID);
ALTER Table MainDriver ADD FOREIGN KEY(AdditionalDriver_ID) REFERENCES AdditionalDriver(AdditionalDriver_ID);
ALTER Table Cancellation ADD FOREIGN KEY(Booking_ID) REFERENCES Booking(Booking_ID);
ALTER Table Booking ADD FOREIGN KEY(User_ID) REFERENCES Users(User_ID);
ALTER Table Booking ADD FOREIGN KEY(Vehicle_ID) REFERENCES Vehicle(Vehicle_ID);
ALTER Table Booking ADD FOREIGN KEY(Insurance_ID) REFERENCES Insurance(Insurance_ID);
ALTER Table Booking ADD FOREIGN KEY(AdditionalDriver_ID) REFERENCES AdditionalDriver(AdditionalDriver_ID);
ALTER Table Booking ADD FOREIGN KEY(SeasonalPrice_ID) REFERENCES SeasonalPrice(SeasonalPrice_ID);
ALTER Table SeasonalPrice ADD Foreign KEY(Vehicle_ID) REFERENCES Vehicle(Vehicle_ID);
ALTER Table HomeBase ADD FOREIGN KEY(Location_ID) REFERENCES Location(Location_ID);
ALTER Table CurrentRentalBase ADD FOREIGN KEY(Location_ID) REFERENCES Location(Location_ID);
ALTER Table Vehicle ADD FOREIGN KEY(HomeBase_ID) REFERENCES HomeBase(HomeBase_ID);
ALTER Table Vehicle ADD FOREIGN KEY(CurrentRentalBase_ID) REFERENCES CurrentRentalBase (CurrentRentalBase_ID);
ALTER Table Services ADD FOREIGN KEY(Location_ID) REFERENCES Location(Location_ID);
ALTER Table Services ADD FOREIGN KEY(Vehicle_ID) REFERENCES Vehicle(Vehicle_ID);
```

Task 4: INDEXES

-- Index on Email
CREATE INDEX userEmail_idx ON Users(Email);
This index, userEmail_idx, is created on the Email attribute of the Users table. It helps in speeding up searches for users by their email addresses, making the process more efficient.

-- Index on Vehicle's Type
CREATE INDEX vehicleType_idx ON Vehicle (Type);
This index is created on the Type attribute of the Vehicle table. It supports lookups of vehicles based on their types. Queries that involve filtering for vehicles by their types benefit from this index.

-- Index on Insurance's Type
CREATE INDEX insuranceType_idx ON Insurance (Type);
This index is created on the Type attribute of the Insurance table. It enables rapid retrieval of insurance records based on their types.

-- Index on Service dates
CREATE INDEX servicesDates_idx ON Services (S_Start_Date, S_End_Date);
This index servicesDates_idx, is created on the S_Start_Date and S_End_Date attributes in the Services table. It facilitates efficient retrieval of service records within specific date ranges. By indexing both the start and end dates, queries related to analysing service usage patterns or identifying services scheduled during particular time periods can be executed more efficiently.

-- Index on Home Base ID
CREATE INDEX vehicleHomebaseID_idx ON Vehicle (HomeBase_ID);
This index vehicleHomebaseID_idx is created on the HomeBase_ID attribute in the Vehicle table. It streamlines the process retrieving of the vehicles based on their associated home bases. By indexing the HomeBase_ID, queries seeking to find which vehicles are located at each home base can be executed more efficiently.

CONTRIBUTION:

Rushaan conceptualized the model using Entity Relationship Diagrams (ERD) and provided accompanying notes to explain the conceptual design. Ishwarya contributed to the ER Diagrams and worked on implementing indexes, while also actively participating in commenting within MariaDB. Aashika and Swapna developed relational schemas and data dictionaries along with their respective notes, detailing the structure and relationships within their databases. Joshua took charge of creating table queries for MariaDB and added relevant comments to enhance clarity and understanding. Additionally, Keerthana created a few table queries for MariaDB.