Used Auto Dealership Database System

CS 4318 – Dr. Shengli Yuan

FINAL REPORT

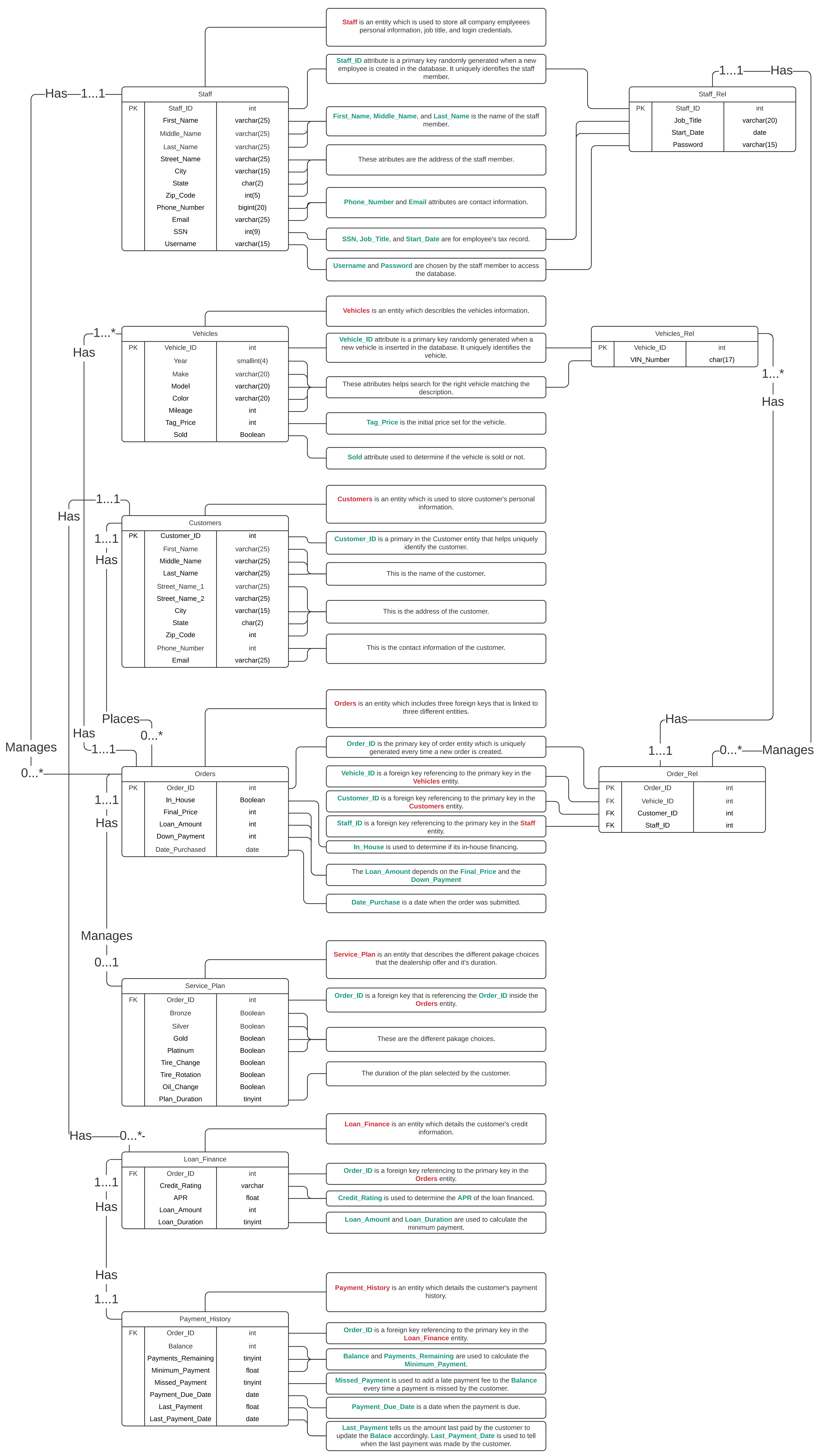
Team Members:

Haider Qazi

Anh Dang

Car Dealership DBMS ERD (UML Notation)

Anh Dang & Haider Qazi



Staff is an entity which is used to store all company emplyeees personal information, job title, and login credentials.

Staff		
PK	Staff_ID	int
	First_Name	varchar(25)
	Middle_Name	varchar(25)
	Last_Name	varchar(25)
	Street_Name	varchar(25)
	City	varchar(15)
	State	char(2)
	Zip_Code	int(5)
	Phone_Number	bigint(20)
	Email	varchar(25)
	SSN	int(9)
	Username	varchar(15)

Staff_ID attribute is a primary key randomly generated when a new employee is created in the database. It uniquely identifies the staff member.

First_Name, **Middle_Name**, and **Last_Name** is the name of the staff member.

These atributes are the address of the staff member.

Phone_Number and **Email** attributes are contact information.

SSN, **Job_Title**, and **Start_Date** are for employee's tax record.

Username and **Password** are chosen by the staff member to access the database.

Staff_Rel		
PK	Staff_ID	int
	Job_Title	varchar(20)
	Start_Date	date
	Password	varchar(15)

Vehicles is an entity which describles the vehicles information.

Vehicles		
PK	Vehicle_ID	int
	Year	smallint(4)
	Make	varchar(20)
	Model	varchar(20)
	Color	varchar(20)
	Mileage	int
	Tag_Price	int
	Sold	Boolean

Vehicle_ID attribute is a primary key randomly generated when a new vehicle is inserted in the database. It uniquely identifies the vehicle.

These attributes helps search for the right vehicle matching the description.

Tag_Price is the initial price set for the vehicle.

Sold attribute used to determine if the vehicle is sold or not.

Vehicles_Rel		
PK	Vehicle_ID	int
	VIN_Number	char(17)

Customers is an entity which is used to store customer's personal information.

Customers		
PK	Customer_ID	int
	First_Name	varchar(25)
	Middle_Name	varchar(25)
	Last_Name	varchar(25)
	Street_Name_1	varchar(25)
	Street_Name_2	varchar(25)
	City	varchar(15)
	State	char(2)
	Zip_Code	int
	Phone_Number	int
	Email	varchar(25)

Customer_ID is a primary in the Customer entity that helps uniquely identify the customer.

This is the name of the customer.

This is the address of the customer.

This is the contact information of the customer.

Orders is an entity which includes three foreign keys that is linked to three different entities.

Orders		
PK	Order_ID	int
	In_House	Boolean
	Final_Price	int
	Loan_Amount	int
	Down_Payment	int
	Date_Purchased	date

Order_ID is the primary key of order entity which is uniquely generated every time a new order is created.

Vehicle_ID is a foreign key referencing to the primary key in the **Vehicles** entity.

Customer_ID is a foreign key referencing to the primary key in the **Customers** entity.

Staff_ID is a foreign key referencing to the primary key in the **Staff** entity.

In_House is used to determine if its in-house financing.

The **Loan_Amount** depends on the **Final_Price** and the **Down_Payment**

Date_Purchase is a date when the order was submitted.

Order_Rel		
PK	Order_ID	int
FK	Vehicle_ID	int
FK	Customer_ID	int
FK	Staff_ID	int

Service_Plan is an entity that describes the different pakage choices that the dealership offer and it's duration.

Service_Plan		
FK	Order_ID	int
	Bronze	Boolean
	Silver	Boolean
	Gold	Boolean
	Platinum	Boolean
	Tire_Change	Boolean
	Tire_Rotation	Boolean
	Oil_Change	Boolean
	Plan_Duration	tinyint

Order_ID is a foreign key that is referencing the **Order_ID** inside the **Orders** entity.

These are the different pakage choices.

The duration of the plan selected by the customer.

Loan_Finance is an entity which details the customer's credit information.

Loan_Finance		
FK	Order_ID	int
	Credit_Rating	varchar
	APR	float
	Loan_Amount	int
	Loan_Duration	tinyint

Order_ID is a foreign key referencing to the primary key in the **Orders** entity.

Credit_Rating is used to determine the **APR** of the loan financed.

Loan_Amount and **Loan_Duration** are used to calculate the minimum payment.

Payment_History is an entity which details the customer's payment history.

Payment_History		
FK	Order_ID	int
	Balance	int
	Payments_Remaining	tinyint
	Minimum_Payment	float
	Missed_Payment	tinyint
	Payment_Due_Date	date
	Last_Payment	float
	Last_Payment_Date	date

Order_ID is a foreign key referencing to the primary key in the **Loan_Finance** entity.

Balance and **Payments_Remaining** are used to calculate the **Minimum_Payment**.

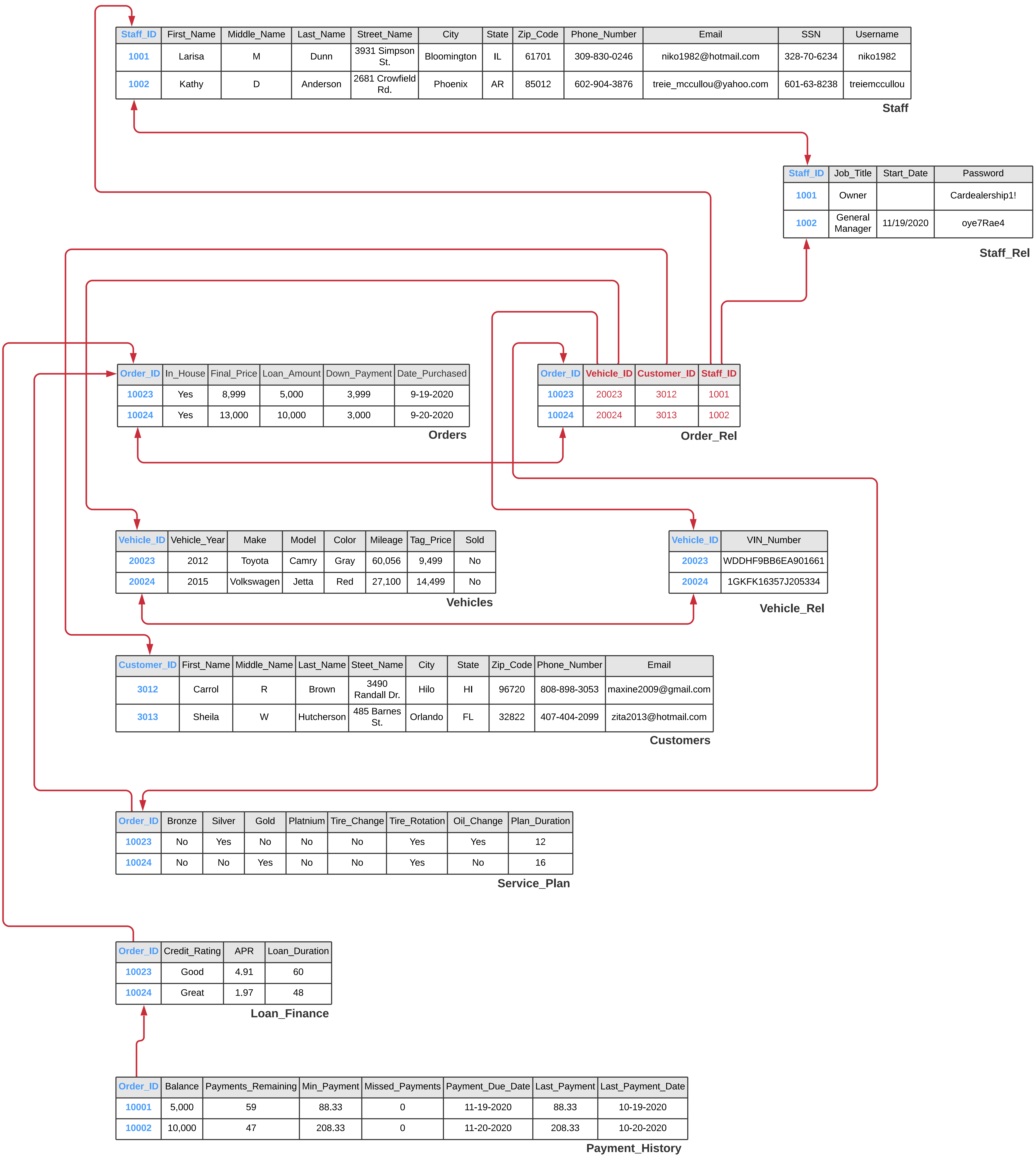
Missed_Payment is used to add a late payment fee to the **Balance** every time a payment is missed by the customer.

Payment_Due_Date is a date when the payment is due.

Last_Payment tells us the amount last paid by the customer to update the **Balace** accordingly. **Last_Payment_Date** is used to tell when the last payment was made by the customer.

Relational Model

Anh Dang & Haider Qazi



Staff_ID	First_Name	Middle_Name	Last_Name	Street_Name	City	State	Zip_Code	Phone_Number	Email	SSN	Username
1001	Larisa	M	Dunn	3931 Simpson St.	Bloomington	IL	61701	309-830-0246	niko1982@hotmail.com	328-70-6234	niko1982
1002	Kathy	D	Anderson	2681 Crowfield Rd.	Phoenix	AR	85012	602-904-3876	treie_mccullou@yahoo.com	601-63-8238	treiemccullou

Staff

Staff_ID	Job_Title	Start_Date	Password
1001	Owner		Cardealership1!
1002	General Manager	11/19/2020	oye7Rae4

Staff_Rel

Order_ID	In_House	Final_Price	Loan_Amount	Down_Payment	Date_Purchased
10023	Yes	8,999	5,000	3,999	9-19-2020
10024	Yes	13,000	10,000	3,000	9-20-2020

Orders

Order_ID	Vehicle_ID	Customer_ID	Staff_ID
10023	20023	3012	1001
10024	20024	3013	1002

Order_Rel

Vehicle_ID	Vehicle_Year	Make	Model	Color	Mileage	Tag_Price	Sold
20023	2012	Toyota	Camry	Gray	60,056	9,499	No
20024	2015	Volkswagen	Jetta	Red	27,100	14,499	No

Vehicles

Vehicle_ID	VIN_Number
20023	WDDHF9BB6EA901661
20024	1GKFK16357J205334

Vehicle_Rel

Customer_ID	First_Name	Middle_Name	Last_Name	Steet_Name	City	State	Zip_Code	Phone_Number	Email
3012	Carrol	R	Brown	3490 Randall Dr.	Hilo	HI	96720	808-898-3053	maxine2009@gmail.com
3013	Sheila	W	Hutcherson	485 Barnes St.	Orlando	FL	32822	407-404-2099	zita2013@hotmail.com

Customers

Order_ID	Bronze	Silver	Gold	Platinum	Tire_Change	Tire_Rotation	Oil_Change	Plan_Duration
10023	No	Yes	No	No	No	Yes	Yes	12
10024	No	No	Yes	No	No	Yes	No	16

Service_Plan

Order_ID	Credit_Rating	APR	Loan_Duration
10023	Good	4.91	60
10024	Great	1.97	48

Loan_Finance

Order_ID	Balance	Payments_Remaining	Min_Payment	Missed_Payments	Payment_Due_Date	Last_Payment	Last_Payment_Date
10001	5,000	59	88.33	0	11-19-2020	88.33	10-19-2020
10002	10,000	47	208.33	0	11-20-2020	208.33	10-20-2020

Payment_History

Abstract: This project is a database system design for a used car dealership. Dealerships without a database system will take a long time to query data. Most used car dealership companies utilize Microsoft Excel or hand-written spreadsheets. The database will optimize query time and structure data so that it is easily understood. The staff will access the database securely by using the correct username and password. Information in the database will include customers, orders, vehicles, loan finance, payment history, service plan, and staff. The database will be implemented in MySQL using Workbench. The staff will be able to manage the orders and other details.

Mission Statement: Our customer database system will help access and manage customer records effectively, efficiently, and securely with the help of a user-friendly interface application design.

Mission Objectives:

- To maintain (enter, update, and delete) data on Staff.
- To maintain (enter, update, and delete) data on Vehicles.
- To maintain (enter, update, and delete) data on Customers.
- To maintain (enter, update, and delete) data on Orders.
- To maintain (enter, update, and delete) data on Service Plan.
- To maintain (enter, update, and delete) data on Loan Finance.
- To maintain (enter, update, and delete) data on Payment History.
- To perform searches on Staff.
- To perform searches on Vehicles.
- To perform searches on Customers.
- To perform searches on Orders.
- To perform searches on Service Plan.
- To perform searches on Loan Finance.
- To perform searches on Payment History.
- To track the status of Sold Vehicles.
- To track the status of Payment History.
- To track the status of Loan Finance.
- To report on Staff.
- To report on Vehicles.
- To report on Customers.
- To report on Orders.
- To report on a Service Plan.
- To report on Loan Finance.
- To report on Payment History.

Major User Views:

Data	Access Type	Owner	General Manager	Assistant Manager	Salesman	Assistant
Staff Staff_Rel	Maintain	X	X			
	Query	X	X	X		
	Report	X	X			
Customers	Maintain		X	X		
	Query		X	X	X	
	Report		X	X	X	
Vehicles Vehicle_Rel	Maintain		X	X		
	Query		X	X	X	
	Report		X	X		
Orders Order_Rel	Maintain		X	X		
	Query		X	X	X	
	Report		X	X	X	

Service Plan	Maintain		X			
	Query		X	X	X	
	Report		X	X	X	
Loan Finance	Maintain		X			
	Query		X	X		
	Report		X	X		
Payment History	Maintain		X			
	Query		X	X		
	Report		X	X		

Use Cases:

Actors/Users:

1. Owner
2. General Manager
3. Assistant Manager
4. Salesman
5. Assistant

Use Cases:

1. Use case name: INSERT a new Staff Member

   Actor/User: Owner
   Steps:
   a) User clicks on Staff Tab.
   b) User clicks on “New Staff” button.
   c) A new Staff ID is generated and displayed.
   d) Prompt user to enter Name, Address, Contact Information, DOB, Gender, Job Title, Salary, SSN, and Start Date.
   e) All information is displayed. Ask for confirmation.
   f) User clicks on “Confirm” button.
   g) New Staff entry is created, and his/her login credentials are generated.

   ```
   INSERT INTO Staff (`Staff_ID`,`First_Name`,  `Middle_Name`,
   `Last_Name`,`Street_Name`, `City`, `State`,  `Zip_Code`, `Phone_Number`,
   `SSN`, `Email`, `Username`) VALUES ('9999','Justin', 'R', 'Smith', '12345
   Street Name Dr', 'Houston', 'TX', '77370', '2811236950', '435134354',
   'vkleynd@cnet.com', 'vkleynd');
   INSERT INTO Staff_Rel (`Staff_ID`, `Job_Title`,`Start_Date`,`Password`)
   VALUES ('9999', 'Assistant Manager', '2020-02-08', 'fzox8Smj2E4B');
   ```

2. Use case name: DELETE a Staff Member

   Actor/User: Owner
   Steps:
   a) User clicks on Staff Tab.
   b) List of Staff Members is displayed.
   c) User selects the Staff Member to delete.
   d) User clicks on “Delete Staff” button.
   e) User clicks on “Confirm” button.
   f) Selected Staff Member is deleted from the Staff Table.

   ```
   DELETE Staff.*, Staff_Rel.*
   FROM Staff
   LEFT JOIN Staff_Rel ON Staff.Staff_ID = Staff_Rel.Staff_ID
   WHERE Staff.Staff_ID = 103 AND Staff_Rel.Staff_ID = 103;
   ```

3. Use case name: UPDATE a Staff Member information.

   Actor/User: Owner
   Steps:
   a) Click on Staff Tab.
   b) List of Staff Members is displayed.
   c) User selects the Staff Member to update.
   d) Staff Member information is displayed.
   e) User updates the Staff Member information.
   f) User clicks “Update” button.
   g) User clicks “Confirm” button.
   h) Staff Member information is updated.

```
UPDATE `new_schema`.`Staff`
SET `Middle_Name` = 'T'
WHERE (`Staff_ID` = '104');
```

4. Use case name: QUERY a Staff Member.
   Actor/User: Owner
   Steps:
   a) User clicks on Search Staff Tab
   b) User is prompted with a search form fields.
   c) User enters Staff ID.
   d) The Staff Member is Highlighted.
   e) User Selects the Staff Member and information is displayed.

```
SELECT * FROM new_schema.Staff
WHERE Staff_ID
IN (104);
```

5. Use case name: INSERT a new Vehicle

   Actor/User: General Manager
   Steps:
   a) User clicks on Vehicle Tab.
   b) User clicks on “New Vehicle” button.
   c) A new Vehicle ID is generated and displayed.
   d) Prompt user to enter VIN number, Year, Make, Model, Color, Mileage, and Tag Price.
   e) All information is displayed. Ask for confirmation.
   f) User clicks on “Confirm” button.
   g) New vehicle entry is created.

```
INSERT INTO `new_schema`.`Vehicle` (`VIN_Number`, `Year`, `Make`,
`Model`, `Color`, `Mileage`, `Tag_Price`)
VALUES ('7GJDV75HF75HF75FH', '2019', 'Toyota', 'Camry', 'Silver', '52000',
'19000');
```

6. Use case name: DELETE a Vehicle

   Actor/User: General Manager
   Steps:
   a) User clicks on Vehicle Tab.
   b) List of Vehicles is displayed.
   c) User selects the Vehicle to delete.
   d) User clicks on “Delete Vehicle” button.
   e) User clicks on “Confirm” button.
   f) Selected Vehicle is deleted from the Vehicle Table.

```
DELETE FROM `new_schema`.`Vehicle`
WHERE (`Vehicle_ID` = '4');
```

7. Use case name: UPDATE a Vehicle information.

   Actor/User: General Manager
   Steps:
   a) Click on Vehicle Tab.
   b) List of Vehicle is displayed.
   c) User selects the Vehicle to update.
   d) Vehicle information is displayed.
   e) User updates the Vehicle information.
   f) User clicks “Update” button.
   g) User clicks “Confirm” button.
   h) Vehicle information is updated.

```
UPDATE `new_schema`.`Vehicle`
SET `Mileage` = '9000', `Tag_Price` = '17000'
WHERE (`Vehicle_ID` = '4');
```

8. Use case name: QUERY a Vehicle.
   Actor/User: General Manager
   Steps:
   a) User clicks on Search Vehicle Tab
   b) User is prompted with a search form fields.
   c) User enters Vehicle ID.
   d) The Vehicle is Highlighted.
   e) User Selects the Vehicle and information is displayed.

```
SELECT * FROM new_schema.Vehicle
WHERE Vehicle_ID
IN (4);
```

9. Use case name: INSERT a new Customer

   Actor/User: Assistant Manager
   Steps:
   a) User clicks on Customer Tab.

b) User clicks on “New Customer” button.
c) A new Customer ID is generated and displayed.
d) Prompt user to enter Name, Address, and Contact Information.
e) All information is displayed. Ask for confirmation.
f) User clicks on “Confirm” button.
g) New Customer entry is created.

```
INSERT INTO `new_schema`.`Customer` (`First_Name`, `Middle_Name`,
`Last_Name`, `Street_Name_1`, `City`, `State`, `Zip_Code`,
`Phone_Number`)
VALUES ('Tom', 'T', 'Cruz', '4355 Hollywood Blvd', 'Los Angeles', 'CA',
'34368', '3123456789');
```

10. Use case name: DELETE a Customer

Actor/User: Assistant Manager
Steps:

a) User clicks on Customer Tab.
b) List of Customers is displayed.
c) User selects the Customer to delete.
d) User clicks on “Delete Customer” button.
e) User clicks on “Confirm” button.
f) Selected Customer is deleted from the Customer Table.

```
DELETE FROM `new_schema`.`Customer`
WHERE (`Customer_ID` = '3');
```

11. Use case name: UPDATE a Customer information.

Actor/User: Assistant Manager
Steps:

a) Click on Customer Tab.
b) List of Customer is displayed.
c) User selects the Customer to update.
d) Customer information is displayed.
e) User updates the Customer information.
f) User clicks “Update” button.
g) User clicks “Confirm” button.
h) Customer information is updated.

```
UPDATE `new_schema`.`Customer`
SET `Phone_Number` = '8569785234'
WHERE (`Customer_ID` = '3');
```

12. Use case name: QUERY a Customer.

Actor/User: Assistant
Steps:

a) User clicks on Search Customer Tab

b) User is prompted with a search form fields.
c) User enters Customer ID.
d) The Customer is Highlighted.
e) User Selects the Customer and information is displayed.

```
SELECT * FROM new_schema.Customer
WHERE Customer_ID
IN (3);
```

13. Use case name: INSERT a new Order

Actor/User: Salesman
Steps:

a) User clicks on Order Tab.
b) User clicks on “New Order” button.
c) A new Order ID is generated and displayed.
d) Prompt user to enter Vehicle ID, Customer ID, and Financial Information.
e) All information is displayed. Ask for confirmation.
f) User clicks on “Confirm” button.
g) New Order entry is created.

```
INSERT INTO `new_schema`.`Order` (`Vehicle_ID`, `Customer_ID`, `Staff_ID`,
`Down_Payment`, `Final_Price`, `In_House`, `Loan_Amount`)
VALUES ('104', '100', '0004', '9000', '15000', '1', '6000');
```

14. Use case name: DELETE an Order

Actor/User: Salesman
Steps:

a) User clicks on Order Tab.
b) List of Order is displayed.
c) User selects the Order to delete.
d) User clicks on “Delete Order” button.
e) User clicks on “Confirm” button.
f) Selected Order is deleted from the Order Table.

```
DELETE FROM `new_schema`.`Order`
WHERE (`Order_ID` = '5');
```

15. Use case name: UPDATE an Order information.

Actor/User: Salesman
Steps:

a) Click on Order Tab.
b) List of Order is displayed.
c) User selects the Order to update.
d) Order information is displayed.
e) User updates the Order information.

f) User clicks “Update” button.
g) User clicks “Confirm” button.
h) Order information is updated.

```
UPDATE `new_schema`.`Order`
SET `Down_Payment` = '2000'
WHERE (`Order_ID` = '3');
```

16. Use case name: QUERY an Order.

Actor/User: Assistant
Steps:

a) User clicks on Search Order Tab
b) User is prompted with a search form fields.
c) User enters Order ID.
d) The Order is Highlighted.
e) User Selects the Order and information is displayed.

```
SELECT * FROM new_schema.Order
WHERE Order_ID
IN (3);
```

17. Use case name: INSERT a new Service Plan

Actor/User: Assistant Manager
Steps:

a) User clicks on Order Tab.
b) User clicks “Search Order” button.
c) User input Order ID.
d) Order information is displayed.
e) User clicks on add service plan.
f) User is prompted to select the Bronze, Silver, Gold, Platinum and its duration.
g) User clicks on “Confirm” button.
h) New Service Plan entry is created.

```
INSERT INTO `new_schema`.`Service_Plan` (`Order_ID1`, `Bronze`, `Silver`,
`Gold`, `Platinum`, `Plan_Duration`,)
VALUES ('0005', '1', '0', '0', '0', '12');
```

18. Use case name: DELETE a Service Plan

Actor/User: Assistant Manager
Steps:

a) User clicks on Order Tab.
b) User clicks “Search Order” button.
c) User input Order ID.
d) Order information is displayed.
e) User clicks on Delete service plan.
f) User clicks on “Confirm” button.

g) Service Plan is Deleted.

```
DELETE FROM `new_schema`.`Service_Plan`
WHERE (`Order_ID1` = '0005');
```

19. Use case name: UPDATE Service Plan duration.

Actor/User: Assistant Manager
Steps:
a) User clicks on Order Tab.
b) User clicks “Search Order” button.
c) User input Order ID.
d) Order information is displayed.
e) User clicks on Update service plan.
f) User change service plan duration.
g) User clicks on “Confirm” button.
h) Service Plan is Updated.

```
UPDATE `new_schema`.`Service_Plan`
SET `Plan_Duration` = '24'
WHERE (`Order_ID1` = '0006');
```

20. Use case name: QUERY a Service Plan.
Actor/User: Assistant
Steps:
a) User clicks on Search Order Tab.
b) User is prompted with a search form fields.
c) User enters Order ID.
d) User clicks on “Service Plan” button.
e) Service Plan information is displayed.

```
SELECT * FROM new_schema.Service_Plan
WHERE Order_ID1
IN (0007);
```

21. Use case name: INSERT a new Loan Finance.

Actor/User: General Manager
Steps:
a) User clicks on Loan Finance Tab.
b) User clicks on “New Loan” button.
c) New Loan ID is generated and displayed.
d) Prompt user to enter Order ID, Credit Rating, APR, Loan Amount, and Loan Duration.
e) All information is displayed. Ask user for confirmation.
f) User clicks on “Confirm” button.
g) New Loan Finance entry is created.

```
INSERT INTO `new_schema`.`Loan_Finance` (`Loan_ID`, `Order_ID`,
`Loan_Amount`, `Loan _Duration`, `Credit_Rating`, `APR`, `Balance`)
VALUES ('0005', '0007', '9000', '12', 'Good', '12', '15000');
```

22. Use case name: DELETE a Loan Finance

    Actor/User: General Manager
    Steps:

    a) User clicks on Loan Finance Tab.
    b) List of Loans is displayed.
    c) User selects the Loan to delete.
    d) User clicks on “Delete Loan” button.
    e) User clicks on “Confirm” button.
    f) Selected Loan entry is deleted from the Loan Finance Table.

```
DELETE FROM `new_schema`.`Loan_Finance`
WHERE (`Loan_ID` = '00020');
```

23. Use case name: UPDATE Loan Finance.

    Actor/User: General Manager
    Steps:

    a) User clicks on Loan Finance Tab.
    b) List of Loans is displayed.
    c) User selects the Loan to update.
    d) Loan Finance information is displayed.
    e) User updates the Loan Finance information.
    f) User clicks “Update” button.
    g) User clicks “Confirm” button.
    h) Loan information is updated.

```
UPDATE `new_schema`.`Loan_Finance`
SET `Loan_Duration` = '16'
WHERE (`Loan_ID` = '00067');
```

24. Use case name: QUERY a Loan Finance.

    Actor/User: Assistant
    Steps:

    a) User clicks on Loan Finance Tab.
    b) User clicks on “Search Loan” button.
    c) User is prompted with a search form fields.
    d) User enters Loan ID.
    e) The Loan is Highlighted.
    f) User Selects the Loan and information is displayed.

```
SELECT * FROM new_schema.Loan_Finance
WHERE Loan_ID
IN (00076);
```

25. Use case name: INSERT Last Payment to Payment History Table.

Actor/User: Assistant Manager
Steps:

a) User clicks on Loan Finance Tab.
b) User clicks on “Search Loan” button.
c) User is prompted with a search form field.
d) User enters the Loan ID.
e) All information is displayed.
f) User clicks on “Payment History” button.
g) Payment History information is displayed.
h) User clicks on Last Payment entry.
i) User updates the Last Payment information value.
j) Ask for confirmation.
k) User clicks on “Confirm” button.
l) New last payment entry is created.

```
INSERT INTO `new_schema`.`Payment_History (`’Last_Payment’)
VALUES (‘315’);
```

26. Use case name: DELETE Balance from Payment History Table.

Actor/User: Assistant Manager
Steps:

a) User clicks on Loan Finance Tab.
b) User clicks on “Search Loan” button.
c) User is prompted with a search form field.
d) User enters the Loan ID.
e) All information is displayed.
f) User clicks on “Payment History” button.
g) Payment History information is displayed.
h) User clicks on “Delete Balance” button.
i) Balance attribute value turns to NULL.

```
DELETE FROM `new_schema`.`Payment_History`
WHERE (`Loan_ID` = '00027');
```

27. Use case name: UPDATE Payment History.

Actor/User: Assistant Manager
Steps:

a) User clicks on Loan Finance Tab.
b) User clicks on “Search Loan” button.
c) User is prompted with a search form field.
d) User enters the Loan ID.
e) All information is displayed.
f) User clicks on “Payment History” button.
g) Payment History information is displayed.
h) User clicks on Last Payment entry.

i) User deletes the Last Payment information value.
j) Ask for confirmation.
k) User clicks on “Confirm” button.
l) Last Payment information is updated in the Payment History Table.

```
UPDATE `new_schema`.`Payment_History`
SET `Last_Payment` = '400'
WHERE (`Loan_ID` = '00050');
```

28. Use case name: QUERY a Minimum Payment from Payment History Table.
Actor/User: Assistant
Steps:
a) User clicks on Loan Finance Tab.
b) User clicks on “Search Loan” button.
c) User is prompted with a search form field.
d) User enters the Loan ID.
e) All information is displayed.

```
SELECT * FROM new_schema.Payment_History
WHERE Loan_ID
IN (00085);
```

29. Use case name: UPDATE BOTH Loan Finance and Payment History with ONE entry.

Actor/User: Salesman
Steps:
a) User clicks on Loan Finance Tab.
b) User clicks on “Search Loan” button.
c) User enters the Loan ID and Selects the Loan.
d) Loan Finance information is displayed.
e) User changes the Loan Duration attribute value.
f) User clicks on “Update” button.
g) Loan Duration entry is Updated in the Loan Finance Table.
h) The Payments Remaining and Minimum Payment is also updated under Payment History.

```
UPDATE `new_schema`.`Loan_Finance`
SET `Loan_Duration` = '16'
WHERE (`Loan_ID` = '00060');
```

30. Use case name: UPDATE BOTH Service Plan and Payment History Balance with ONE ENTRY.

Actor/User: Salesman
Steps:
a) User clicks on Order Tab.
b) User clicks on “Search Order” button.
c) User enters Order ID.

d) Order is highlighted and user selects the order.
e) Order information is displayed.
f) User clicks on “Service Plan” button.
g) User updates the service package value.
h) User clicks “Confirm”.
i) Balance under Payment History entity is updated also.

```
UPDATE `new_schema`.`Service_Plan`
SET `Silver` = '1’
WHERE (`Order_ID’ = '0006');
```

31. Use case name: UPDATE BOTH Order Down Payment and Payment History Balance with one ENTRY.

    Actor/User: Salesman
    Steps:

    a) User clicks on Order Tab.
    b) User clicks “Search Order” button.
    c) User enters Order ID.
    d) Order is highlighted and user selects the order.
    e) Order information is displayed.
    f) User update Down Payment attribute value.
    g) User clicks “Update” button.
    h) User clicks “Confirm” button.
    i) Balance attribute under Payment History is also updated.

```
UPDATE `new_schema`.`Order`
SET `Down_Payment` = '12000'
WHERE (`Order_ID` = '0007');
```

32. Use case name: CREATE Order entry and Sold attribute under Vehicle is UPDATED.

    Actor/User: Salesman
    Steps:

    a) User clicks on Order Tab.
    b) User clicks on “New Order” button.
    c) User enters Vehicle ID and order information.
    d) New Order entry is created.
    e) Sold attribute under Vehicle is updated to True.

```
INSERT INTO `new_schema`.`Order’ (‘Vehicle_ID’)
VALUES (‘007’);
```

33. Use case name: QUERY Vehicle information inside Order.

    Actor/User: Assistant
    Steps:

    a) User clicks on Order Tab.

b) User clicks “Search Order” button.
c) User enters Order ID.
d) Order is highlighted and user selects the order.
e) Vehicle information is displayed inside Order.

```
SELECT * FROM new_schema.Order
WHERE Order_ID
IN (00087);
```

34. Use case name: QUERY Staff information inside Order.

Actor/User: Assistant
Steps:
a) User clicks on Order Tab.
b) User clicks “Search Order” button.
c) User enters Order ID.
d) Order is highlighted and user selects the order.
e) Staff information is displayed inside Order.

```
SELECT * FROM new_schema.Order
WHERE Order_ID
IN (0008);
```

35. Use case name: UPDATE Staff information inside Staff_Rel.

Actor/User: General Manager
Steps:
a) User clicks on Staff Tab.
b) User clicks “Search Staff” button.
c) User enters Staff ID.
d) Staff is highlighted and user selects the Staff.
e) User modifies the Staff Information.
f) User click on “Update” button.
g) Staff information is updated inside Staff_Rel Table.

```
UPDATE Staff_Rel
SET `Job_Title` = 'Assistant Manager'
WHERE (`Staff_ID` = '104');
```

36. Use case name: UPDATE Orders information inside Order_Rel.

Actor/User: Salesman
Steps:
a) User clicks on Order Tab.
b) User clicks “Search Order” button.
c) User enters Order ID.
d) Order is highlighted and user selects the Order.
e) User modifies the Order Information.
f) User click on “Update” button.

g) Order information is updated inside Order _Rel Table.

```
UPDATE Order_Rel
SET `Staff_ID` = '1005'
WHERE (`Order_ID` = '3001');
```

37. Use case name: UPDATE Vehicles information inside Vehicles _Rel.

Actor/User: General Manager
Steps:

a) User clicks on Vehicles Tab.
b) User clicks “Search Vehicles” button.
c) User enters Vehicles ID.
d) Vehicle is highlighted and user selects the Vehicle.
e) User modifies the Vehicle Information.
f) User click on “Update” button.
g) Vehicle information is updated inside Vehicle _Rel Table.

```
UPDATE Vehicle_Rel
SET `VIN_Number` = '2D4F5G9D2D556S12D'
WHERE (`Vehicle_ID` = '3031');
```

38. Use case name: QUERY Staff information from Staff _Rel.

Actor/User: General Manager
Steps:

a) User clicks on Staff Tab.
b) User clicks “Search Staff” button.
c) User enters Staff ID.
d) Staff is highlighted and user selects the Staff.
e) Staff information is displayed from Staff _Rel Tables.

```
SELECT * FROM Staff_Rel
WHERE Staff_ID = '1002'
```

39. Use case name: QUERY Orders information from Order_Rel.

Actor/User: Salesman
Steps:

a) User clicks on Orders Tab.
b) User clicks “Search Orders” button.
c) User enters Order ID.
d) Order is highlighted and user selects the Order.
e) Order information is displayed from Order_Rel Tables.

```
SELECT * FROM Order_Rel
WHERE Order_ID = '4002'
```

40. Use case name: QUERY Vehicles information from Vehicle_Rel.

Actor/User: Salesman
Steps:

a) User clicks on Vehicles Tab.
b) User clicks “Search Vehicles” button.
c) User enters Vehicle ID.
d) Vehicle is highlighted and user selects the Vehicle.
e) Vehicle information is displayed from Vehicle_Rel Tables.

```
SELECT * FROM Vehicle_Rel
WHERE Vehicle_ID = '3002'
```

Aggregate Function Queries Use Cases:

41. Use case name: QUERY avg APR from Loan Finance Table.

Actor/User: Assistant Manager
Steps:

a) User clicks on Loan Finance Tab.
b) User clicks “Average APR” button.
c) The Average APR is displayed.

```
SELECT FORMAT(AVG(APR),2) Avg_APR
FROM Loan_Finance;
```

42. Use case name: QUERY avg Plan Duration from Service Plan Table.

Actor/User: General Manager
Steps:

a) User clicks on Service Plan Tab.
b) User clicks “Average Plan Duration” button.
c) The Average Plan Duration is displayed.

```
SELECT FORMAT(AVG(Plan_Duration),2) Average_Plan_Duration
FROM Service_Plan;
```

43. Use case name: QUERY the Staff based on Job Title.

Actor/User: General Manager
Steps:

a) User clicks on Staff Tab.
b) User clicks “Number of Staff” button.
c) The Number of Staff Members by Job Title is displayed.

```
SELECT City, COUNT(*) AS Count_City
FROM Staff
GROUP BY City
ORDER BY City;
```

44. Use case name: QUERY Total Customers from Customers Table.

    Actor/User: Assistant Manager
    Steps:
    a) User clicks on Customer Tab.
    b) The Total Number of Customers is displayed.

```
SELECT COUNT(*) AS Total_Customers
FROM Customers;
```

45. Use case name: QUERY Max Final Price from Orders Table.

    Actor/User: Assistant Manager
    Steps:
    a) User clicks on Orders Tab.
    b) User clicks on “Sort Final Price” Button.
    c) Max Final Price is displayed.

```
SELECT MAX(Final_Price) Max_Final_Price
FROM Orders;
```

46. Use case name: QUERY Lowest Payments Remaining from Payment History Table.

    Actor/User: General Manager
    Steps:
    a) User clicks on Payment History Tab.
    b) User clicks on “Sort Payment Remaining” Button.
    c) Lowest Payment Remaining is displayed.

```
SELECT MIN(Payments_Remaining) Min_Payments_Rem
FROM Payment_History;
```

47. Use case name: QUERY Total Tag Price from Vehicles Table.

    Actor/User: General Manager
    Steps:
    a) User clicks on Vehicles Tab.
    b) User clicks on “Total Vehicles Price” Button.
    c) Total Vehicles Tag Price is displayed.

```
SELECT FORMAT(SUM(Tag_Price),0) Total_Inventory_Price
FROM Vehicles;
```

48. Use case name: QUERY the Staff based on Job Title.

    Actor/User: Assistant Manager
    Steps:
    d) User clicks on Staff Tab.

e) User clicks “Number of Staff” button.
f) The Number of Staff Members by Job Title is displayed.

```
SELECT Job_Title, COUNT(*) AS Job_Title_Count
FROM Staff_Rel
GROUP BY Job_Title
ORDER BY Job_Title;
```

49. Use case name: QUERY Amount of Orders Sold by a Staff Member.

Actor/User: Assistant Manager
Steps:
a) User clicks on Staff Tab.
b) User clicks on “Search Staff” Button.
c) User enters Staff_ID.
d) Staff is highlighted and User selects the Staff.
e) User clicks on the “Vehicles Sold” button.
f) Amount of Vehicles Sold by Staff Member is displayed.

```
SELECT Staff_ID, COUNT(*) AS Vehicles_Sold
FROM Order_Rel
WHERE Staff_ID = '1003';
```

50. Use case name: QUERY Amount of Vehicles from Vehicle_Rel Table.

Actor/User: Salesman
Steps:
a) User clicks on Vehicles Tab.
b) Amount of Vehicles is displayed.

```
SELECT Vehicle_ID, COUNT(*) AS Vehicles_Count
FROM Vehicle_Rel;
```

Joint Queries Use Cases:

51. Use case name: QUERY Staff Name and Order_ID.

Actor/User: Salesman
Steps:
a) User clicks on Staff Tab.
b) User clicks on “Staff Order Details” Button.
c) Staff Name and Order_ID is displayed.

```
SELECT Order_Rel.Staff_ID, CONCAT(Staff.First_Name,' ', Staff.Last_Name)
AS 'Staff_Name', Order_Rel.Order_ID
FROM Order_Rel
INNER JOIN Staff
ON Order_Rel.Staff_ID=Staff.Staff_ID;
```

52. Use case name: QUERY Staff Username and Password.

    Actor/User: General Manager
    Steps:
    a) User clicks on Staff Tab.
    b) User search Staff by Staff_ID.
    c) Staff is highlighted and User select the Staff Member.
    d) User clicks on “Staff Details” Button.
    e) Staff Username and Password is displayed.

```
SELECT Staff.Staff_ID, Staff.Username, Staff_Rel.Password
FROM Staff_Rel
INNER JOIN Staff
ON Staff.Staff_ID=Staff_Rel.Staff_ID;
```

53. Use case name: QUERY Order_ID, Staff_ID, and Vehicle information.

    Actor/User: Salesman
    Steps:
    a) User clicks on Orders Tab.
    b) User search Orders by Order_ID.
    c) Order is highlighted and User selects the Order.
    d) User clicks on “Order Details” Button.
    e) Staff and Vehicle information is displayed.

```
SELECT Order_Rel.Order_ID, Vehicles.Vehicle_ID,
CONCAT(Vehicles.Vehicle_Year,' ', Vehicles.Make,' ', Vehicles.Model)
AS Vehicle, Order_Rel.Staff_ID
FROM Vehicles INNER JOIN Order_Rel
ON Order_Rel.Vehicle_ID=Vehicles.Vehicle_ID;
```

54. Use case name: QUERY Order_ID, Staff_ID, and Job_Title.

    Actor/User: General Manager
    Steps:
    a) User clicks on Orders Tab.
    b) User search Orders by Order_ID.
    c) Order is highlighted and User selects the Order.
    d) User clicks on “Order Details” Button.
    e) Staff ID and Job Title information is displayed.

```
SELECT Order_Rel.Order_ID, Order_Rel.Staff_ID, Staff_Rel.Job_Title
FROM Staff_Rel INNER JOIN Order_Rel
ON Staff_Rel.Staff_ID=Order_Rel.Staff_ID;
```

55. Use case name: QUERY Order_ID, Customer_ID, Address, and Contact Info.

    Actor/User: Salesman
    Steps:

a) User clicks on Orders Tab.
b) User search Orders by Order_ID.
c) Order is highlighted and User selects the Order.
d) User clicks on “Order Details” Button.
e) Order ID, Customer_ID, Address, and Contact Info is displayed.

```
SELECT Order_Rel.Order_ID, Order_Rel.Customer_ID,
CONCAT(Customers.First_Name, " ", Customers.Last_Name) AS
Customer_Name, CONCAT(Customers.Street_Name, " ", Customers.City, " ",
Customers.State, " ", Customers.Zip_Code) AS Address,
Customers.Phone_Number, Customers.Email
FROM Order_Rel INNER JOIN Customers ON
Order_Rel.Customer_ID=Customers.Customer_ID;
```

56. Use case name: QUERY Loan_ID and Payment Info.

Actor/User: Assistant Manager
Steps:

a) User clicks on Loan_Finance Tab.
b) User search Loan by Loan_ID.
c) Loan is highlighted and User selects the Loan.
d) User clicks on “Loan Details” Button.
e) Loan ID and Payment Info is displayed.

```
SELECT Loan_Finance.Order_ID, Loan_Finance.Loan_Duration,
Payment_History.Balance, Payment_History.Payments_Remaining,
Payment_History.Min_Payment, Payment_History.Missed_Payments,
Payment_History.Payment_Due_Date, Payment_History.Last_Payment,
Payment_History.Last_Payment_Date
FROM Payment_History
INNER JOIN Loan_Finance ON
Loan_Finance.Order_ID=Payment_History.Order_ID;
```

57. Use case name: QUERY Order Details and Loan Details.

Actor/User: Assistant Manager
Steps:

a) User clicks on Loan_Finance Tab.
b) User search Loan by Loan_ID.
c) Loan is highlighted and User selects the Loan.
d) User clicks on “Loan Details” Button.
e) Loan ID and Order Info is displayed.

```
SELECT Loan_Finance.Order_ID, Orders.Final_Price, Orders.Loan_Amount,
Orders.Down_Payment, Orders.Date_Purchased,Loan_Finance.APR,
Loan_Finance.Loan_Duration
FROM Loan_Finance
INNER JOIN Orders
ON Orders.Order_ID=Loan_Finance.Order_ID;
```

58. Use case name: QUERY Order Details and Service Plan Details.

Actor/User: Assistant Manager
Steps:

a) User clicks on Order Tab.
b) User search Order by Order_ID.
c) Order is highlighted and User selects the Order.
d) User clicks on “Order Details” Button.
e) Order info and Service Plan Info is displayed.

```
SELECT Orders.Order_ID, Orders.Final_Price, Orders.Date_Purchased,
Service_Plan.Bronze, Service_Plan.Silver, Service_Plan.Gold,
Service_Plan.Platinum, Service_Plan.Tire_Change, Service_Plan.Tire_Rotation,
Service_Plan.Oil_Change, Service_Plan.Plan_Duration
FROM Service_Plan
INNER JOIN Orders ON Orders.Order_ID=Service_Plan.Order_ID;
```

59. Use case name: QUERY Vehicle VIN and Details information.

Actor/User: Salesman
Steps:

a) User clicks on Orders Tab.
b) User search Orders by Order_ID.
c) Order is highlighted and User selects the Order.
d) User clicks on “Order Details” Button.
e) Staff and Vehicle information is displayed.

```
SELECT Vehicles.Vehicle_Year, Vehicles.Make, Vehicles.Model,
Vehicle_Rel.VIN_Number
FROM Vehicles INNER JOIN Vehicle_Rel
ON Vehicle_Rel.Vehicle_ID=Vehicles.Vehicle_ID;
```

60. Use case name: QUERY Order_ID, Final Price and Customer ID

Actor/User: Salesman
Steps:

a) User clicks on Orders Tab.
b) User search Orders by Order_ID.
c) Order is highlighted and User selects the Order.
d) User clicks on “Order Details” Button.
e) Order ID, Final_Price, Customer ID is displayed.

```
SELECT Orders.Order_ID, Orders.Final_Price, Order_Rel.Customer_ID
FROM Orders INNER JOIN Order_Rel ON
Orders.Order_ID=Order_Rel.Order_ID;
```

61. Use case name: QUERY Vehicle ID, Order ID and Vin Number

Actor/User: Salesman

Steps:

a) User clicks on Orders Tab.
b) User search Orders by Order_ID.
c) Order is highlighted and User selects the Order.
d) User clicks on “Order Details” Button.
e) Order ID, Vehicle ID and Vin Number is displayed.

```
SELECT Order_Rel.Order_ID, Order_Rel.Vehicle_ID, Vehicle_Rel.Vin_Number
FROM Order_Rel INNER JOIN Vehicle_Rel ON
Order_Rel.Vehicle_ID=Vehicle_Rel.Vehicle_ID;
```

62. Use case name: QUERY Order ID, Vehicle ID and Service Plan Details.

Actor/User: Assistant Manager
Steps:

a) User clicks on Order Tab.
b) User search Order by Order_ID.
c) Order is highlighted and User selects the Order.
d) User clicks on “Order Details” Button.
e) Order ID, Vehicle ID, and Service Plan Info is displayed.

```
SELECT Order_Rel.Order_ID, Order_Rel.Vehicle_ID, Service_Plan.Bronze,
Service_Plan.Silver, Service_Plan.Gold, Service_Plan.Platinum,
Service_Plan.Tire_Change, Service_Plan.Tire_Rotation,
Service_Plan.Oil_Change, Service_Plan.Plan_Duration
FROM Service_Plan
INNER JOIN Order_Rel ON Order_Rel.Order_ID=Service_Plan.Order_ID;
```

Conclusion:

The final project has a total of 10 entities. Some of the entities from phase 1 and 2 were in 1NF, 2NF, and 3NF. Therefore, they had to be converted to BCNF. There is a total of 40 simple use cases, with each 10 entities being able to QUERY, UPDATE, DELETE, and INSERT. There is a total of 10 aggregate function use cases, one for each entity. There is a total of 12 joint query for direct entity relationship use cases, one for each arrow in our relational model. All use cases are 62 in total. After phase 2, we changed from 7 entities to 10 entities. We separated the Staff, Order, and Vehicle table. This will eliminate any update, insert, or delete anomalies. The ERD, relational model, and database also changed because of these modifications. This project has taught us how to implement Database Lifecycle. It taught tools such as MySQL and how to use Workbench.

Reference:

MySQL :: MySQL 8.0 Reference Manual :: 12.20.1 Aggregate Function Descriptions. (2020). MySQL. https://dev.mysql.com/doc/refman/8.0/en/aggregate-functions.html

Database Normalization Overview. (2020). MariaDB KnowledgeBase. https://mariadb.com/kb/en/database-normalization-overview/

MySQL Join Made Easy For Beginners. (2020). MySQL Tutorial. https://www.mysqltutorial.org/mysql-join/#:%7E:text=To%20join%20tables%2C%20you%20use,the%20FULL%20OUTER%20JOIN%20yet.

C. (2016, May 7). *Database Analysis and the DreamHome Case Study-pert18-sbd*. Catatanuniv. https://catatanuniv.wordpress.com/2016/05/07/database-analysis-and-the-dreamhome-case-study-pert18-sbd/