



Module Code & Module Title
CC5051NT Databases

Assessment Weightage & Type
50% Individual Coursework

Year and Semester
2022-23 Spring 2

Student Name: Geenesh Acharya
Group: L2C4

London Met ID: 22016051
College ID: np05cp4s220023@iic.edu.np

Assignment Due Date: 05 January 2023
Assignment Submission Date: 05 January 2023

I confirm that I understand my coursework needs to be submitted online via Google Classroom under the relevant module page before the deadline for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a mark of zero will be awarded.

Contents

1.	Introduction	1
1.1.	Introduction of the Business	1
1.2.	Aims	2
1.3.	Objectives	2
1.4	Current Business Activities and Operation	2
1.5	Business Rules	3
2	Entity Description	4
2.1	Table of Customer	4
2.2	Table of Vehicle	4
2.3	Table of Service	5
2.4	Table of Driver.....	6
3	Initial ERD	6
4	Normalization	8
4.1	UNF (Unnormalized Form)	8
4.2	1NF (First Normal Form)	9
4.3	2NF (Second Normal Form)	10
4.4	3NF (Third Normal Form)	11
5	Final ERD.....	14
6	Database Implementation	15
6.4	Creating Tables for Elevate System	15
6.4.1	Creating user identification	16
6.4.2	Table for Customer	17
6.4.3	Table for Vehicle.....	18
6.4.4	Table for Driver	19

6.4.5	Table for Service	20
6.4.6	Table for Invoice	20
7	Populating Database.....	21
7.1	Inserting the data of Service.....	21
7.2	Inserting the data of Customer	22
7.3	Inserting the data of Vehicle.....	23
7.4	Inserting the data of Driver	24
7.5	Inserting the data of the Invoice	25
8	Database Query	26
8.1	Informational Queries.....	26
8.2	Transactional Queries with Relational Algebra.....	29
9	Critical Evaluation	34
9.1	Further discussion on the learning experience.....	34
9.2	Critical Assessment.....	34
10	Conclusion.....	36
11	Creating Dump File.....	37
12	References	39
13	Appendix	40
13.1	SQL Command Line code.....	40
13.2	Creating Table Code.....	40
13.2.1	Customer Data code	40
13.2.2	Vehicle Data code	40
13.2.3	Service Data code	41
13.2.4	Invoice Data code.....	41
13.3	Describe Table Code	42

13.3.1	Table of Customer	42
13.3.2	Table of Vehicle	42
13.3.3	Table of Service	42
13.3.4	Table of Invoice	42
13.4	Inserting Data Code	42
13.4.1	Inserting Customer Data	42
13.4.2	Inserting Vehicle Data	44
13.4.3	Inserting Service Data	46
13.4.4	Inserting Driver Data	46
13.4.5	Inserting Invoice Data	48
13.5	Showing Data Code	49
13.5.1	Customer Data	49
13.5.2	Vehicle Data	49
13.5.3	Service Data	50
13.5.4	Driver Data	50
13.5.5	Invoice Data	50
13.6	Informational Queries Code	50
13.7	Transactional Queries	50

List of Tables

Table 1 Entity Description -Table of Customer	4
Table 2 Entity Description -Table of Vehicle	5
Table 3 Entity Description -Table of Service	5
Table 4 Entity Description -Table of Driver	6

List of Figures

Figure 1 Initial ERD - ERD Cardinality	6
Figure 2 ERD - Initial ER Diagram	7
Figure 3 ERD - Final ER Diagram	14
Figure 4 Database Implementation - Creating user identification.....	16
Figure 5 Database Implementation - Data for Customer.....	17
Figure 6 Database Implementation - Table for Customer	17
Figure 7 Database Implementation - Data for Vehicle.....	18
Figure 8 Database Implementation - Table for Vehicle	18
Figure 9 Database Implementation - Data for Driver	19
Figure 10 Database Implementation - Table for Driver	19
Figure 11 Database Implementation - Data for Service.....	20
Figure 12 Database Implementation - Table for Service	20
Figure 13 Database Implementation - Data for Invoice	20
Figure 14 Database Implementation - Table for Invoice	21
Figure 15 Populating Database - Inserting Data of Service.....	21
Figure 16 Populating Database – Service Data	22
Figure 17 Populating Database - Inserting Data of Customer	22
Figure 18 Populating Database - Customer Data	23
Figure 19 Populating Database - Inserting Data of Vehicle	23
Figure 20 Populating Database - Vehicle data.....	24
Figure 21 Populating Database - Inserting Data of Driver	24
Figure 22 Populating Database - Driver Data.....	25
Figure 23 Populating Database - Inserting Data of Invoice	25
Figure 24 Populating Database - Invoice Data.....	26
Figure 25 Informational Queries - List of all Customer Category	26
Figure 26 Informational Queries - Model and Variants price in descending order	27
Figure 27 Informational Queries - Number of Vehicle using Petrol	27
Figure 28 Informational Queries - List of Issued Ticket.....	28
Figure 29 Informational Queries - List of names which have 's' in middle.....	28
Figure 30 Transactional Queries - Figure 1.....	29

Figure 31 Transactional Queries - Figure 2.....	30
Figure 32 Transactional Queries - Figure 3.....	31
Figure 33 Transactional Queries - Figure 4.....	32
Figure 34 Transactional Queries - Figure 5.....	33
Figure 35 Screenshot of Dump File	38

1. Introduction

1.1. Introduction of the Business

Elevate Limited is an online transportation system that will be started by Mr. Zenas Acharya, an international Businessman and real estate owner. This business operates over the internet as an application and is an excellent platform for Driver to give their Service to the Customer in front of the world. Customers have the benefits of reward points and can get a specific price after hitting a target. As a result, various Driver has been able to manage their livelihood through this online transportation system

The main aim of Elevate online transportation system is to provide different facilities to the Customer through the help of applications. In Elevate application, customers can reserve vehicles like Motorcycle, Taxi, Auto, and E-rickshaw. Among the other features of Elevate application, it now has a food or package delivery system for the Customer for better convenience. You can also work as a part-time or full-time service in an application. To work part-time, you should meet the criteria:

- i. The Vehicle should be in good condition.
- ii. Rider/Driver should have proper registration and licensee.
- iii. They should need to get a verified tick from the Application.

To maintain security in the Application, Customer can share their location with their relatives, and all verified drives will be tracked easily in case of emergency. After elevate worker provides every complete Service, an invoice will be generated, and the rating can be given to the driver/rider. If the driver/rider gets a continuously low rating, he will be suspended as a service giver.

After gathering opinions on the Elevate online transportation system from experts and the public, we learned that after a few months of operation, this transportation system would yield positive results. And if commerce keeps booming this way, the transportation system will have a great future. The business may very well be able to compete with other uber systems soon, like Pathao, Lyft, and others.

1.2. Aims

The Elevate Transportation system aims to provide fast, convenient, and efficient Service in delivering packages and booking transportation services by developing a website and mobile Application.

1.3. Objectives

Elevate is an online booking and delivery service for transportation services. In addition to booking and delivering transportation, these systems can also be used for commuting, traveling, and transporting goods. The Elevate Transportation System has the following objectives:

- i. To make it easier and more convenient for users of package delivery and transportation services.
- ii. To give customers various transportation choices, allowing them to evaluate costs and Availability.
- iii. To increase the efficiency of transportation operations and reduce costs.
- iv. To generate revenue through the sale of transportation services and related products.
- v. To keep track of all users, staff, vehicles, and services.

1.4 Current Business Activities and Operation

Our business activities highly depend on customers who can rent cars, place food orders, and use our package delivery service. They are the four important corporate assets of our business. The business activities that our transportation system "ELevate" will be conducting when it is officially started are listed below: -

- i. Elevate will offer job chances to all interested parties, including those who want to work part-time.
- ii. The interaction will be done through customer care support with technical support, where customers may also demand their desirable features for the Application.
- iii. Customers are one of the most significant assets to the transportation system. Thus, various offers and discounts are given out on special days to increase renting of transportation.

- iv. The enthusiastic employees working hard to represent our company are also paid a good monthly wage.
- v. The transportation system is promoted and presented to the world through various online auctions and advertisements.
- vi. After every complete Service, customer reward points will be increased.
- vii. Regular health check to verify driver/rider every three months.

As a result of these operations, the business will grow throughout the year by presenting outstanding services to the Customer and ensuring the Driver and staff member's lifestyles.

1.5 Business Rules

The collection of principles and instructions that direct how a company should conduct its business is known as business rules. We have already discussed current business activities and the operation of the transportation system. The firm now requires rules and regulations to efficiently carry out such tasks and functions. The following is a list of these business rules: -

- i. A driver may drive many vehicles, but only one Driver uses each Vehicle and Service at a time.
- ii. A driver writes a single invoice for each Service he provides.
- iii. Once the Customer books the Service, they cannot cancel the Service.
- iv. The service ticket is issued once the Customer books the Vehicle and the Service and will include details like driver name, type of Service, total charge, and estimated destination duration.
- v. The cost of the Vehicle and duration can vary depending on its type. For example, the price of riding a motorcycle service can be lower than riding a car.
- vi. The total Service charge is calculated according to the time taken and the Vehicle charged.
- vii. Customers have the right to choose a Vehicle according to their budget.
- viii. Customers and employees of companies pay different rates. Employees receive a 10% discount when they use the Service.

2 Entity Description

In simple terms, an entity is a group of easily recognizable objects, either animated or inanimate. For example, we have a hospital database where Doctors, Patients, Administration, Registration, and Health offices can all be considered entities. Each of these entities has some attributes or properties that make them unique.

In database management systems (DBMS), an attribute is a piece of data that describes an entity. Every attribute has its value, as the doctor entity may have the following attributes: Employee no., Name, Gender, Age, and Specialist. Our Elevate transportation system consists of entities, and their attributes are described below, along with their functions.

2.1 Table of Customer

Keys	Attributes of Customer	Description
P. K.	Customer_ID	Customer ID is the primary key, unique, and cannot be Null.
	Customer_Name	Name of the Customer
	Customer_Address	It is a place where the Customer lives.
	Customer_Phone_No.	It is the contact no of the Customer.
	Customer_Reward_Point	It is a certain point for every Service used.
	Customer_Category	It is a category of staff and Normal Customers.

Table 1 Entity Description -Table of Customer

2.2 Table of Vehicle

Keys	Attributes of Customer	Description
P. K.	Vehicle_ID	It is a Unique ID of the Vehicle.
	Vehicle_Name	It is the Name of the Vehicle.
	Vehicle_Model	It shows a model of the Vehicle.
	Vehicle_Type	Several types of vehicles like Bike, Scooter, cars, and many more
	Vehicle_Staff_Price	It is a price for staff who works for a company.

	Vehicle_Normal_Price	It is a price that normal people should pay for their Service used.
	Vehicle_Fuel_Type	There are two types of vehicle fuel, i.e., Petrol and Diesel
F. K	Customer_ID	It is a foreign key of the Customer table.

Table 2 Entity Description -Table of Vehicle

2.3 Table of Service

Keys	Attributes of Customer	Description
P. K.	Service_ID	It is a unique id of Service provided to an organization.
	Service_Name	It is the name of our three services.
	Service_Type	It is a type of Service used by Customers.
	Service_Date	It is the date of Service provided to the Customer.
	Service_Charge	It is a certain amount of charge to be paid after using its Service.
	Service_Ticket_No.	It is a unique no for every Service used.
	Service_Duration	It is the time taken to reach the Customer's destination.
	Service_Destination	It is a Destination of the Customer to be reached.
F. K	Customer_ID	It is a foreign key of the Customer and Vehicle table.
F. K	Vehicle_ID	It is a foreign key of the Customer and Vehicle table.

Table 3 Entity Description -Table of Service

2.4 Table of Driver

Keys	Attributes of Customer	Description
P. K.	Driver_ID	Unique ID for Driver
	Driver_Name	It is the name of the vehicle owner.
	Driver_Address	Place where the Driver lives.
	Driver_Type	It is a type of Driver. (Half time and full-time)
	Driver_Availability	It shows the Driver's status

Table 4 Entity Description -Table of Driver

3 Initial ERD

An Initial Entity Relationship Diagram (ERD) is a beginning phase of a graphical representation that displays the relationship of entity sets stored in a database. It is based on three basic concepts: entities, attributes, and relationships. The main objectives of creating an Initial ER Diagram are to represent an appropriate final Entity relationship diagram with its correct details and relationship to entities. We create Entity Relationship Diagrams using cardinality, the most significant element of an ERD that shows how one entity is related to another. Several types of lines can represent relationships, and they are as follows: -

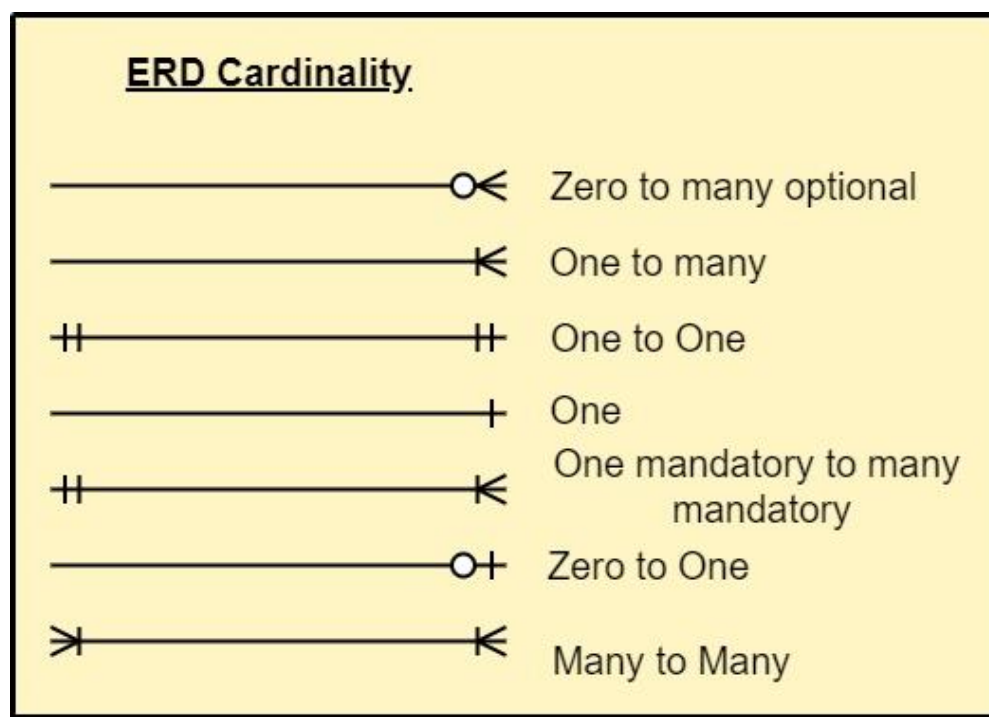


Figure 1 Initial ERD - ERD Cardinality

In my initial ERD, there are four entities with their attributes. The four entities are Customer, Driver, Service, and Vehicle. All entities are related to each. Through it, customers may make reservations for various Vehicle and services, and many-to-many relation between three entities is established. Still, one Driver can drive multiple vehicles, so one-to-many relation is established.

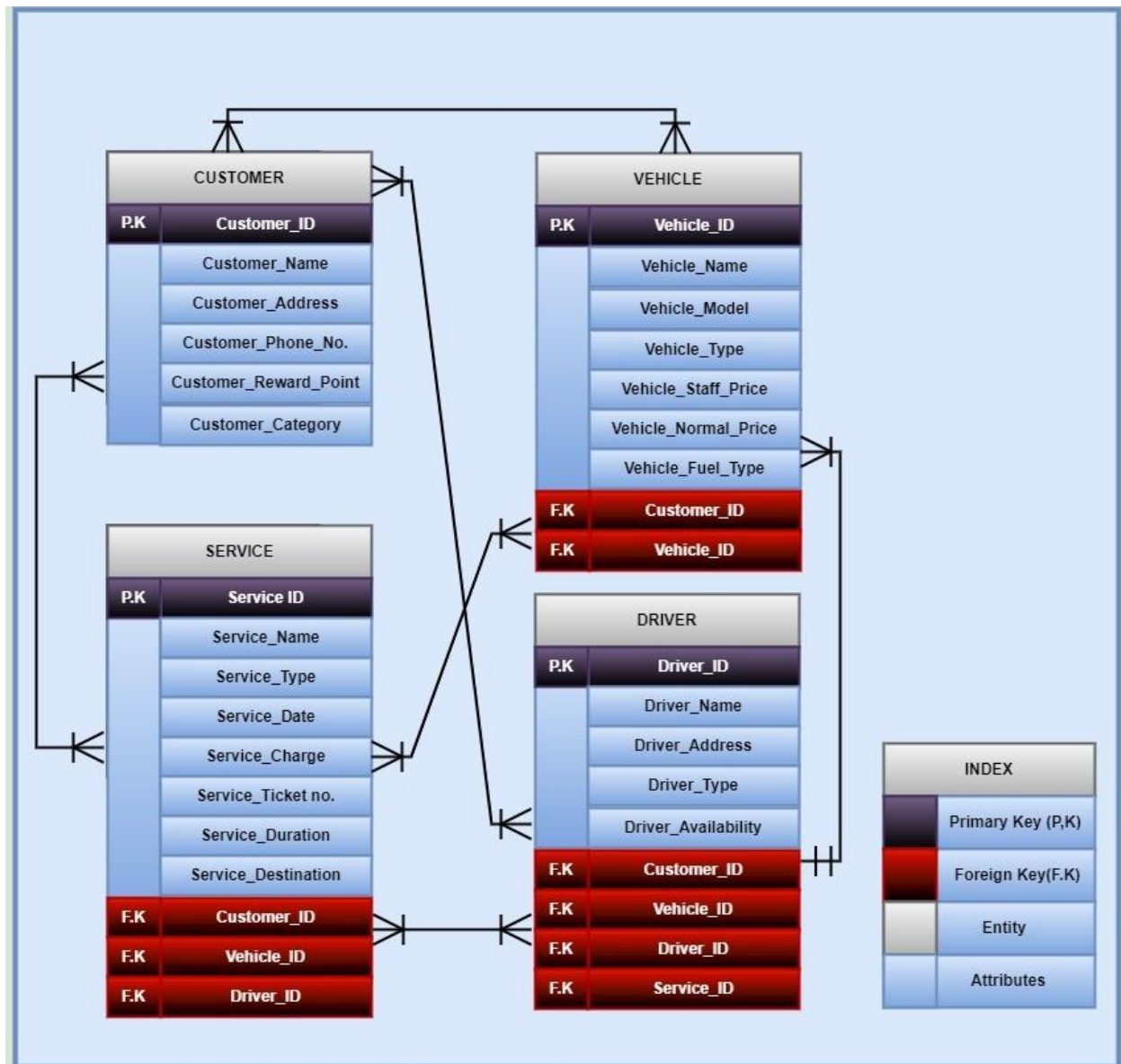


Figure 2 ERD - Initial ER Diagram

The above initial ERD has produced different anomalies and data redundancies since it was created. As a result of these anomalies, the quality of the Database would be degraded, and it wouldn't be acceptable. It may also lead to the downfall of the company itself. Hence, Normalization is done to eliminate data anomalies and duplicates.

4 Normalization

Normalization is organizing a table within a database, which reduces data redundancy and reflects separate entities, attributes, and relationships between them to eliminate unnecessary data duplication. The main objective of Normalization is to set the relationship between tables and make them more flexible by reducing redundancy and inconsistent dependency. Before doing Normalization, we should know why we are doing Normalization, so some points are listed:

- i. To make database design efficient.
- ii. To reduce the amount of data.
- iii. To make data free to update, Insertion, and Deletion Anomalies.
- iv. To show the pertinent relationship between entities.
- v. To reduce the restructure of data and to design and simplify data.

4.1 UNF (Unnormalized Form)

Un-normalization is an initial stage of Normalization, which we know as a structured frame representing a piece of organizational data or document. Every step of Normalization depends upon the previous form, which is vital for the starting stage. We set the right domains and data to ensure a smooth transition between the locations. We know that it contains various anomalies and redundancies, but still, there are advantages of practicing it because of the following points:

- i. It makes queries much simpler.
- ii. It helps us by making complex data structures in a simplified way.
- iii. The data can be organized more efficiently.

Presenting the repeating data and repeating groups:

➤ {Customer{Vehicle{Driver{Service}}}}

Customer {Customer_ID, Customer_Name, Customer_Address, Customer_Phone_no, Customer_Reward_Point, Customer_Category {**Vehicle:** Vehicle_ID, Vehicle_Name, Vehicle_Model, Vehicle_Type, Vehicle_Staff_price, Vehicle_Normal_Price, Vehicle_Fuel_Type {**Driver:**Driver_ID, Driver_Name, Driver_Address, Driver_Type, Driver_Availability {**Service:** Service_Ticket_no., Service_ID, Service_Name, Service_Charge, Service_Date, Service_Duration, Service_Destination}}}}

4.2 1NF (First Normal Form)

The first Normal Form (1NF) establishes the fundamental rules for normalizing databases and relates to a single table. In 1NF, we list all the primary keys and separate the repeating group and data. Some of the rules for 1NF are:

- i. It should have atomic-valued attributes or columns.
- ii. In a table, each column's name needs to be unique.

Normalizing 1NF

Customer (**Customer ID**, Customer_Name, Customer_Address, Customer_Phone_no, Customer_Reward_Point, Customer_Category)

Vehicle (**Vehicle ID**, Vehicle_Name, Vehicle_Model, Vehicle_Type, Vehicle_Staff_price, Vehicle_Normal_Price, Vehicle_Fuel_Type, **Customer ID***)

Driver (**Driver ID**, Driver_Name, Driver_Address, Driver_Type, Driver_Availability **Customer ID***, **Vehicle ID***)

Service (**Service Ticket no.**, Service_ID, Service_Name, Service_Charge, Service_Date, Service_Duration, Service_Destination, **Customer ID***, **Vehicle ID***, **Driver ID***)

4.3 2NF (Second Normal Form)

The second Normal Form (2NF) is the second step of normalizing a database, and it can be built after the creation of 1NF. We divide the entities into groups based on partial and complete dependency and then check whether any entity contains partial dependencies. Some of the rules for 2NF are:

- i. It shouldn't be a partial dependency.
- ii. It should be in 1NF.

Normalizing 2NF

Customer ID → Customer_Name, Customer_Address, Customer_Phone_no, Customer_Reward_Point, Customer_Category

Vehicle ID → Vehicle_Name, Vehicle_Model, Vehicle_Type, Vehicle_Staff_price, Vehicle_Normal_Price, Vehicle_Fuel_Type [Partial Dependency]

Vehicle_ID, Customer_ID → Vehicle_Name, Vehicle Model, Vehicle Type, Vehicle Staff price, Vehicle Normal Price, Vehicle Fuel Type

Driver ID → Driver Name, Driver Address, Driver Type, Driver_Availability

Customer_ID, Vehicle_ID, Driver_ID → Driver_Name, Driver_Address, Driver_Type, Driver_Availability

Service Ticket no. → Service_ID, Service_Name, Service_Charge, Service_Date, Service_Duration, Service_Destination

Customer_ID, Vehicle_ID, Driver_ID, Service_Ticket_no → Service ID, Service Name, Service Charge, Service Date, Service Duration, Service Destination

- There are partial dependencies in the vehicle table where every non-key is partially dependent on Vehicle_ID, so to break it down, I have listed another table: -

Listing 2NF in the table:

Customer (**Customer ID**, Customer_Name, Customer_Address, Customer_Phone_no, Customer_Reward_Point, Customer_Category)

Vehicle (**Vehicle ID**, Vehicle_Name, Vehicle_Model, Vehicle_Type, Vehicle_Staff_price, Vehicle_Normal Price, Vehicle_Fuel Type)

Customer, Vehicle (**Customer ID***, **Vehicle ID***,)

Driver (**Driver ID**, Driver_Name, Driver_Address, Driver_Type, Driver_Availability)

Customer, Vehicle, Driver (**Customer ID***, **Vehicle ID***, **Driver ID***)

Service (**Service Ticket no.**, Service_ID, Service Name, Service Charge, Service Date, Service Duration, Service Destination)

Customer, Vehicle, Driver, Service (**Customer ID***, **Vehicle ID***, **Driver ID***, **Service ID***)

4.4 3NF (Third Normal Form)

The third normal form (3NF) is part of a group of concepts for database normalization that also includes the first normal forms, 1NF and 2NF. Some of the rules for 3NF are:

- i. It shouldn't be a transitive dependency.
- ii. It should be in 2NF.

Normalizing 3NF**1. Customer**

Customer ID → Customer Name → X

Customer ID → Customer Address → X

Customer ID → Customer Phone no. → X

Customer ID → Customer Reward Point → X

Customer ID → Customer Category → X

2. Vehicle

Vehicle ID → Vehicle Name → X

Vehicle ID → Vehicle Type → X

Vehicle ID → Vehicle Staff Price → X

Vehicle ID → Vehicle Normal Price → X

Vehicle ID → Vehicle Fuel Type → X

Vehicle ID → Vehicle Model → X

3. Driver

Driver ID → Driver Name → X

Driver ID → Driver Address → X

Driver ID → Driver Type → X

Driver ID → Driver Availability → X

4. Service

Service Ticket no. → Service ID → Service Name → X

Service Ticket no. → (Service ID, Service Name)

Service Ticket no. → Service Charge → X

Service Ticket no. → Service Date → X

Service Ticket no. → Service Duration → X

Service Ticket no. → Service Destination → X

5. Invoice

Invoice (**Invoice Ticket No.**, Service Charge, Service Date, Service Duration, Service Destination, Total_Charge, **Service ID***, **Customer ID***, **Vehicle ID***, **Driver ID***)

Listing all table

Customer (**Customer ID**, Customer Name, Customer Address, Customer Phone no, Customer Reward Point, Customer Category)

Vehicle (**Vehicle ID**, Vehicle_Name, Vehicle Model, Vehicle Type, Vehicle Staff price, Vehicle Normal Price, Vehicle Fuel Type)

Driver (**Driver ID**, Driver Name, Driver Address, Driver Type, Driver_Availability)

Service (**Service ID**, Service Name)

Invoice (**Invoice Ticket No**, Service Date, Service Charge, Service Duration, Service Destination, Total_Charge, **Customer ID***, **Vehicle ID***, **Driver ID***, **Service ID***)

5 Final ERD

A conceptual graphical representational model called an Entity Relationship Diagram (ERD) shows the structure and relationships of the entities maintained in a database. In the actual world, an entity is a thing or idea that exists separately. Entities are identical to database tables in a relational database, with each row of the table representing an instance of that object. An entity's attribute is a specific property that defines the entity. There are often one-to-one, one-to-many, or many-to-many relationships. (Techopedia, 2017)

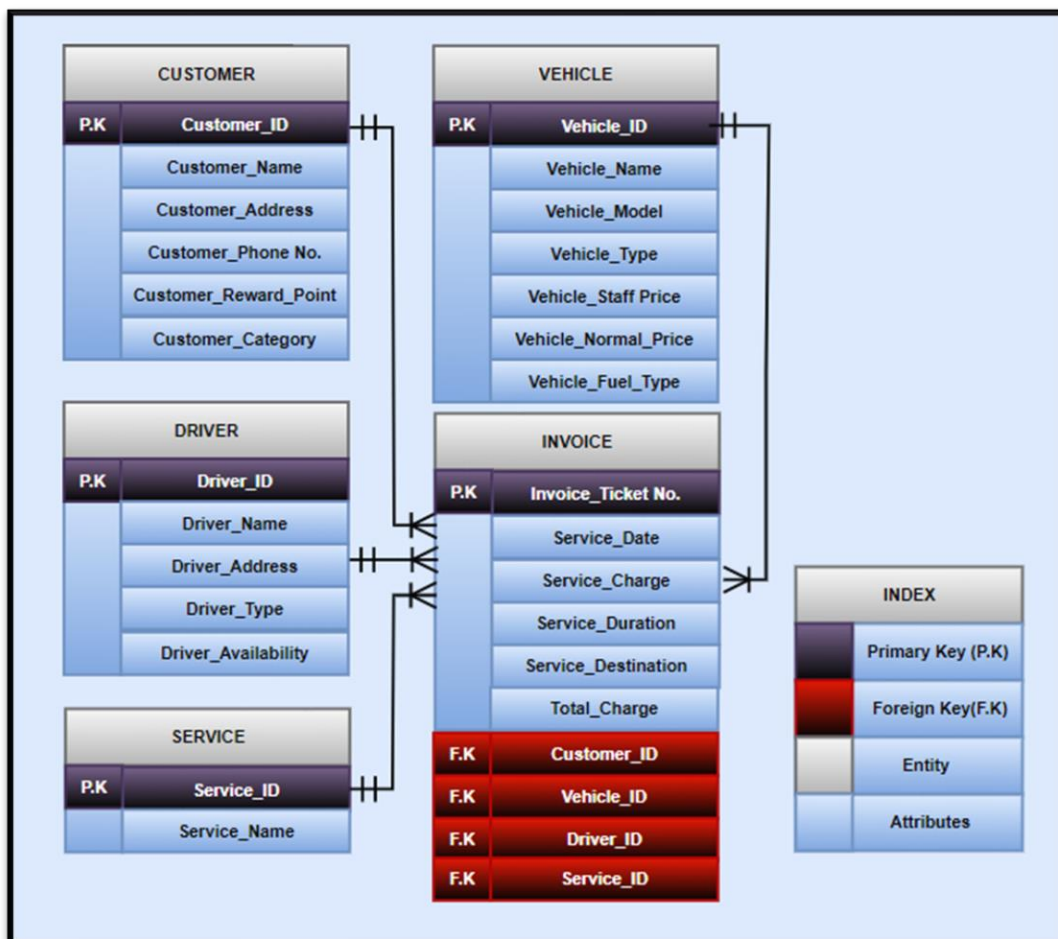


Figure 3 ERD - Final ER Diagram

A final ERD with fewer data abnormalities and redundancies is created here. A total of five entities are built for the Transportation System "Elevate" to store the attributes which contain information and data about it. All the properties are correctly set in their respective entities by considering every requirement the business has provided. The final ERD is created as a result.

6 Database Implementation

As we know, the process of setting up and configuring a database system to store, organize, and manage data is known as database implementation. As part of the process, the Database requirements are determined, logical and physical structures are designed, data is populated, and the Database is tested and optimized, ensuring high performance in the Database.

Implementing a database is an essential step in its lifecycle because it determines how the Database will be used and how effectively it will store, retrieve, and manage data. For a database to be reliable and successful, it must be implemented correctly.

All the entities required for the Elevate transportation system have been listed in a table. Primary keys are unique identifiers for tables and entities, and if the primary key is used in an entity, it cannot be null or repeated. PK is often used to denote it.

Likewise, A foreign key in a relational database table links data from two tables by creating a link between them. It refers to the primary key column of another table. In most cases, it is denoted by FK.

6.4 Creating Tables for Elevate System

A table is created by using the 'CREATE TABLE' statement. It can be used in the following ways:

```
CREATE TABLE <table_name> (  
    Column1 datatype,  
    Column2 datatype,  
    -----  
    Column n datatype);
```

The following is the SQL command line for creating the table for all entities with their descriptions:

6.4.1 Creating user identification

Before starting with SQL, we need to connect to the MYSQL server as a user with the necessary permissions to create users. We need to grant permission to the user to perform actions within the Database after creating a user account. Before we begin working on a user, there are several benefits to it, and some of them are: -

1. Enhance security: Authenticating users with a username and password when connecting to the Database ensures that only authorized users can access it.
2. Improve performance: Creating SQL users will allow different users to access different resources, improving database performance.

To create user identification, we need to do the following steps: -

Step 1: Run SQL Command Line.

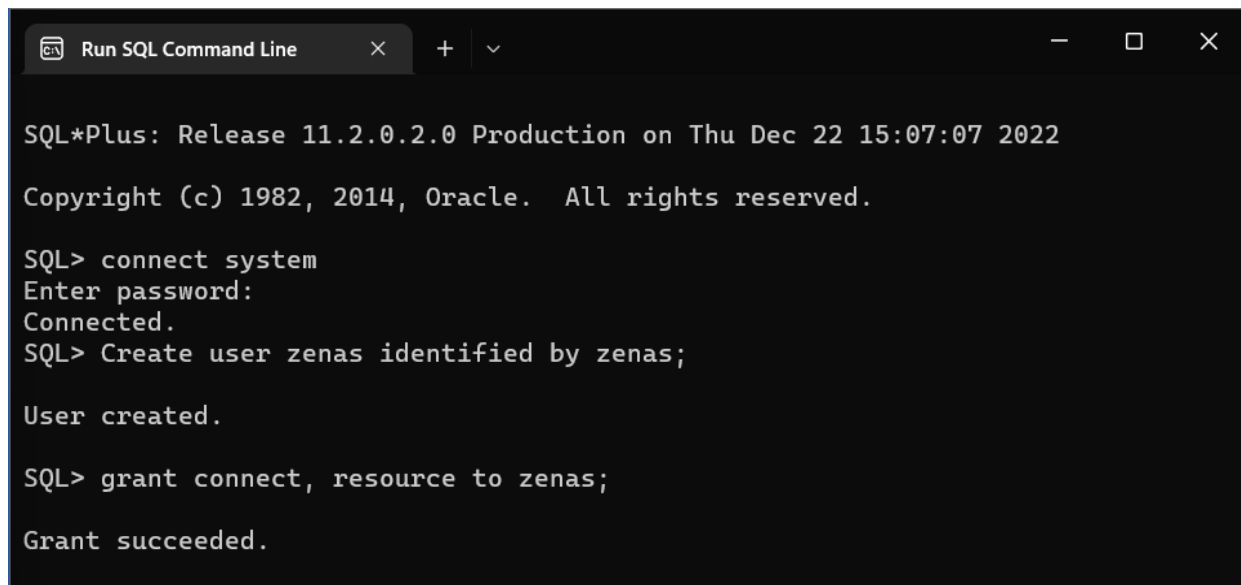
Step 2: Connect to the System and enter your oracle password.

Step 3: Create a user account.

Step 4: Grant permission to the user.

Step 5: Successfully user-created.

➤ I have written the following line statement of code to create a user:



```
SQL*Plus: Release 11.2.0.2.0 Production on Thu Dec 22 15:07:07 2022
Copyright (c) 1982, 2014, Oracle. All rights reserved.

SQL> connect system
Enter password:
Connected.
SQL> Create user zenas identified by zenas;

User created.

SQL> grant connect, resource to zenas;

Grant succeeded.
```

Figure 4 Database Implementation - Creating user identification

6.4.2 Table for Customer

- To create a customer table in SQL, I have written the following line statement of code using different clauses and datatype in capital letters.

```
SQL> CREATE TABLE Customer (  
  2 Customer_ID VARCHAR2(20) NOT NULL,  
  3 Customer_Name VARCHAR2(20) NOT NULL,  
  4 Customer_Address VARCHAR2(20) NOT NULL,  
  5 Customer_Phone_No INT NOT NULL,  
  6 Customer_Category VARCHAR2(20) NOT NULL,  
  7 Customer_Reward_Point INT NOT NULL,  
  8 CONSTRAINT PK_Customer PRIMARY KEY (Customer_ID));  
  
Table created.
```

Figure 5 Database Implementation - Data for Customer

- After creating the table, to show them in a more clear way, I used the following code: -

```
SQL> DESCRIBE Customer;  
Name                                         Null?    Type  
-----  
CUSTOMER_ID                                NOT NULL VARCHAR2(20)  
CUSTOMER_NAME                              NOT NULL VARCHAR2(20)  
CUSTOMER_ADDRESS                           NOT NULL VARCHAR2(20)  
CUSTOMER_PHONE_NO                           NOT NULL NUMBER(38)  
CUSTOMER_CATEGORY                          NOT NULL VARCHAR2(20)  
CUSTOMER_REWARD_POINT                       NOT NULL NUMBER(38)  
  
SQL> |
```

Figure 6 Database Implementation - Table for Customer

6.4.3 Table for Vehicle

- To create a Vehicle table in SQL, I have written the following line statement of code:

```
SQL> CREATE TABLE Vehicle (
  2  Vehicle_ID VARCHAR2(20) NOT NULL,
  3  Vehicle_Name VARCHAR2(20) NOT NULL,
  4  Vehicle_Model VARCHAR2(20) NOT NULL,
  5  Vehicle_Type VARCHAR2(20) NOT NULL,
  6  Vehicle_Staff_Price INT NOT NULL,
  7  Vehicle_Normal_Price INT NOT NULL,
  8  Vehicle_Fuel_Type VARCHAR2(20) NOT NULL,
  9  CONSTRAINT PK_Vehicle PRIMARY KEY (Vehicle_ID));

Table created.
```

Figure 7 Database Implementation - Data for Vehicle

- After creating the table, to show them more clearly, I used the following code:

```
SQL> DESCRIBE Vehicle;
Name                                     Null?      Type
-----
VEHICLE_ID                             NOT NULL   VARCHAR2(20)
VEHICLE_NAME                           NOT NULL   VARCHAR2(20)
VEHICLE_MODEL                           NOT NULL   VARCHAR2(20)
VEHICLE_TYPE                           NOT NULL   VARCHAR2(20)
VEHICLE_STAFF_PRICE                     NOT NULL   NUMBER(38)
VEHICLE_NORMAL_PRICE                     NOT NULL   NUMBER(38)
VEHICLE_FUEL_TYPE                       NOT NULL   VARCHAR2(20)

SQL> |
```

Figure 8 Database Implementation - Table for Vehicle

6.4.4 Table for Driver

- To create a Driver table in SQL, I have written the following line statement of code:

```
SQL> CREATE TABLE Driver (  
2 Driver_ID VARCHAR2(20) NOT NULL,  
3 Driver_Name VARCHAR2(20) NOT NULL,  
4 Driver_Address VARCHAR2(20) NOT NULL,  
5 Driver_Type VARCHAR2(20) NOT NULL,  
6 Driver_Availability VARCHAR2(20) NOT NULL,  
7 CONSTRAINT PK_Driver PRIMARY KEY (Driver_ID));  
  
Table created.
```

Figure 9 Database Implementation - Data for Driver

- After creating the table, to show them more clearly, I used the following code:

```
SQL> DESCRIBE Driver;  
Name                               Null?    Type  
-----  
DRIVER_ID                          NOT NULL VARCHAR2(20)  
DRIVER_NAME                        NOT NULL VARCHAR2(20)  
DRIVER_ADDRESS                     NOT NULL VARCHAR2(20)  
DRIVER_TYPE                        NOT NULL VARCHAR2(20)  
DRIVER_AVAILABILITY                NOT NULL VARCHAR2(20)  
  
SQL> |
```

Figure 10 Database Implementation - Table for Driver

6.4.5 Table for Service

- To create a Service table in SQL, I have written the following line statement of code:

```
SQL> CREATE TABLE Service (
  2 Service_ID VARCHAR2(20) NOT NULL,
  3 Service_Name VARCHAR2(20) NOT NULL,
  4 CONSTRAINT PK_Service PRIMARY KEY (Service_ID));

Table created.
```

Figure 11 Database Implementation - Data for Service

- After creating the table, to show them more clearly, I used the following code: -

```
SQL> DESCRIBE Service;
Name                                         Null?    Type
-----
SERVICE_ID                                NOT NULL VARCHAR2(20)
SERVICE_NAME                              NOT NULL VARCHAR2(20)

SQL> |
```

Figure 12 Database Implementation - Table for Service

6.4.6 Table for Invoice

- To create an Invoice table in SQL, which also have four foreign key and one primary key, I have written the following line statement of code:

```
SQL> CREATE TABLE Invoice (
  2 Invoice_Ticket_No VARCHAR2(20) NOT NULL,
  3 Service_Date DATE NOT NULL,
  4 Service_Charge VARCHAR2(20) NOT NULL,
  5 Service_Duration VARCHAR2(20) NOT NULL,
  6 Service_Destination VARCHAR2(20) NOT NULL,
  7 Total_Charge INT NOT NULL,
  8 Customer_ID VARCHAR2(20) NOT NULL,
  9 Vehicle_ID VARCHAR2(20) NOT NULL,
  10 Driver_ID VARCHAR2(20) NOT NULL,
  11 Service_ID VARCHAR2(20) NOT NULL,
  12 CONSTRAINT PK_Invoice_Ticket_No PRIMARY KEY (Invoice_Ticket_No),
  13 CONSTRAINT FK_Customer_ID FOREIGN KEY (Customer_ID) REFERENCES Customer (Customer_ID),
  14 CONSTRAINT FK_Vehicle_ID FOREIGN KEY (Vehicle_ID) REFERENCES Vehicle (Vehicle_ID),
  15 CONSTRAINT FK_Driver_ID FOREIGN KEY (Driver_ID) REFERENCES Driver (Driver_ID),
  16 CONSTRAINT FK_Service_ID FOREIGN KEY (Service_ID) REFERENCES Service (Service_ID));

Table created.

SQL> |
```

Figure 13 Database Implementation - Data for Invoice

- After creating the table, to show them more clearly, I used the following code: -

```
SQL> DESCRIBE Invoice;
Name                                     Null?      Type
-----
INVOICE_TICKET_NO                       NOT NULL   VARCHAR2(20)
SERVICE_DATE                           NOT NULL   DATE
SERVICE_CHARGE                         NOT NULL   VARCHAR2(20)
SERVICE_DURATION                       NOT NULL   VARCHAR2(20)
SERVICE_DESTINATION                   NOT NULL   VARCHAR2(20)
TOTAL_CHARGE                           NOT NULL   NUMBER(38)
CUSTOMER_ID                            NOT NULL   VARCHAR2(20)
VEHICLE_ID                             NOT NULL   VARCHAR2(20)
DRIVER_ID                              NOT NULL   VARCHAR2(20)
SERVICE_ID                             NOT NULL   VARCHAR2(20)

SQL> |
```

Figure 14 Database Implementation - Table for Invoice

7 Populating Database

The process of populating a database is to add data to its tables. The data can be manually entered into the tables or automatically imported from external sources such as text files.

Populating a database is one of the most important steps in setting up and using a database system. This is because it allows you to store and manage the data for which the Database will be used. All the necessary data for the Elevate transportation system is inserted here with their screenshots.

7.1 Inserting the data of Service

To insert the data in Service, I have written the following line of code: -

```
SQL> INSERT ALL
2   INTO Service (Service_ID, Service_Name)
3   VALUES ('S201', 'Food Delivery')
4   INTO Service (Service_ID, Service_Name)
5   VALUES ('S202', 'Package Delivery')
6   INTO Service (Service_ID, Service_Name)
7   VALUES ('S203', 'Courier Service')
8   SELECT * FROM dual;

3 rows created.
```

Figure 15 Populating Database - Inserting Data of Service

- We have used a line of code below to show the inserted Service data more clearly: -

SERVICE_ID	SERVICE_NAME
S201	Food Delivery
S202	Package Delivery
S203	Courier Service

Figure 16 Populating Database – Service Data

7.2 Inserting the data of Customer

To insert the data in Customer, I have written the following line of code: -

```
SQL> INSERT ALL
2 INTO Customer (Customer_ID, Customer_Name, Customer_Address, Customer_Phone_No, Customer_Category, Customer_Reward_Point)
3 VALUES ('C201', 'Sricha Parajuli', 'Dharan', '9852055474', 'Normal', '50')
4 INTO Customer (Customer_ID, Customer_Name, Customer_Address, Customer_Phone_No, Customer_Category, Customer_Reward_Point)
5 VALUES ('C202', 'Shrasta Niraula', 'Biratnagar', '9842112943', 'Normal', '20')
6 INTO Customer (Customer_ID, Customer_Name, Customer_Address, Customer_Phone_No, Customer_Category, Customer_Reward_Point)
7 VALUES ('C203', 'Shreya Bhandari', 'Belbari', '9804042626', 'Staff', '35')
8 INTO Customer (Customer_ID, Customer_Name, Customer_Address, Customer_Phone_No, Customer_Category, Customer_Reward_Point)
9 VALUES ('C204', 'Alisha Rai', 'Urlabari', '9842020991', 'Normal', '60')
10 INTO Customer (Customer_ID, Customer_Name, Customer_Address, Customer_Phone_No, Customer_Category, Customer_Reward_Point)
11 VALUES ('C205', 'Shrasta Niraula', 'Biratnagar', '9842112943', 'Normal', '20')
12 INTO Customer (Customer_ID, Customer_Name, Customer_Address, Customer_Phone_No, Customer_Category, Customer_Reward_Point)
13 VALUES ('C206', 'Deepika Rijal', 'Salakpur', '9842056621', 'Staff', '80')
14 INTO Customer (Customer_ID, Customer_Name, Customer_Address, Customer_Phone_No, Customer_Category, Customer_Reward_Point)
15 VALUES ('C207', 'Geenesh Acharya', 'Halgada', '9800967301', 'Staff', '80')
16 INTO Customer (Customer_ID, Customer_Name, Customer_Address, Customer_Phone_No, Customer_Category, Customer_Reward_Point)
17 VALUES ('C208', 'Sampada Regmi', 'Birtamode', '9802775699', 'Normal', '65')
18 INTO Customer (Customer_ID, Customer_Name, Customer_Address, Customer_Phone_No, Customer_Category, Customer_Reward_Point)
19 VALUES ('C209', 'Shrasta Niraula', 'Biratnagar', '9842112943', 'Normal', '20')
20 INTO Customer (Customer_ID, Customer_Name, Customer_Address, Customer_Phone_No, Customer_Category, Customer_Reward_Point)
21 VALUES ('C210', 'Shrasta Niraula', 'Biratnagar', '9842112943', 'Normal', '20')
22 INTO Customer (Customer_ID, Customer_Name, Customer_Address, Customer_Phone_No, Customer_Category, Customer_Reward_Point)
23 VALUES ('C211', 'Ranjita Limbu', 'Bhadrapur', '9852136547', 'Normal', '55')
24 INTO Customer (Customer_ID, Customer_Name, Customer_Address, Customer_Phone_No, Customer_Category, Customer_Reward_Point)
25 VALUES ('C212', 'Dikshya Dhimal', 'Inaruwa', '9864532147', 'Normal', '10')
26 INTO Customer (Customer_ID, Customer_Name, Customer_Address, Customer_Phone_No, Customer_Category, Customer_Reward_Point)
27 VALUES ('C213', 'Geenesh Acharya', 'Halgada', '9800967301', 'Staff', '80')
28 INTO Customer (Customer_ID, Customer_Name, Customer_Address, Customer_Phone_No, Customer_Category, Customer_Reward_Point)
29 VALUES ('C214', 'Sushmeeta Jha', 'Lahan', '9832456176', 'Normal', '20')
30 INTO Customer (Customer_ID, Customer_Name, Customer_Address, Customer_Phone_No, Customer_Category, Customer_Reward_Point)
31 VALUES ('C215', 'Shrasta Niraula', 'Biratnagar', '9842112943', 'Normal', '20')
32 SELECT * FROM dual;

15 rows created.
```

Figure 17 Populating Database - Inserting Data of Customer

- We have used a line of code below to show the inserted customer data more clearly: -

```
SQL> SELECT * FROM Customer;
```

CUSTOMER_ID	CUSTOMER_NAME	CUSTOMER_ADDRESS	CUSTOMER_PHONE_NO	CUSTOMER_CATEGORY	CUSTOMER_REWARD_POINT
C201	Sricha Parajuli	Dharan	9852055474	Normal	50
C202	Shrasta Niraula	Biratnagar	9842112943	Normal	20
C203	Shreya Bhandari	Belbari	9804042626	Staff	35
C204	Alisha Rai	Urlabari	9842020991	Normal	60
C205	Shrasta Niraula	Biratnagar	9842112943	Normal	20
C206	Deepika Rijal	Salakpur	9842056621	Staff	80
C207	Geenesh Acharya	Halgada	9800967301	Staff	80
C208	Sampada Regmi	Birtamode	9802775699	Normal	65
C209	Shrasta Niraula	Biratnagar	9842112943	Normal	20
C210	Shrasta Niraula	Biratnagar	9842112943	Normal	20
C211	Ranjita Limbu	Bhadrapur	9852136547	Normal	55
C212	Dikshya Dhimal	Inaruwa	9864532147	Normal	10
C213	Geenesh Acharya	Halgada	9800967301	Staff	80
C214	Sushmeeta Jha	Lahan	9832456176	Normal	20
C215	Shrasta Niraula	Biratnagar	9842112943	Normal	20

15 rows selected.

Figure 18 Populating Database - Customer Data

7.3 Inserting the data of Vehicle

To insert the data in Vehicle, I have written the following line of code: -

```
SQL> INSERT ALL
2  INTO Vehicle (Vehicle_ID, Vehicle_Name, Vehicle_Model, Vehicle_Type, Vehicle_Staff_Price, Vehicle_Normal_Price, Vehicle_Fuel_Type)
3  VALUES ('V201', 'Toyota', 'Prius', 'Car', '450', '500', 'Diesel')
4  INTO Vehicle (Vehicle_ID, Vehicle_Name, Vehicle_Model, Vehicle_Type, Vehicle_Staff_Price, Vehicle_Normal_Price, Vehicle_Fuel_Type)
5  VALUES ('V202', 'Volkswagen', 'Golf GTI', 'Car', '540', '600', 'Diesel')
6  INTO Vehicle (Vehicle_ID, Vehicle_Name, Vehicle_Model, Vehicle_Type, Vehicle_Staff_Price, Vehicle_Normal_Price, Vehicle_Fuel_Type)
7  VALUES ('V203', 'TVS', 'Jupiter', 'Scooter', '720', '800', 'Petrol')
8  INTO Vehicle (Vehicle_ID, Vehicle_Name, Vehicle_Model, Vehicle_Type, Vehicle_Staff_Price, Vehicle_Normal_Price, Vehicle_Fuel_Type)
9  VALUES ('V204', 'Audi', 'A3', 'Car', '855', '950', 'Diesel')
10 INTO Vehicle (Vehicle_ID, Vehicle_Name, Vehicle_Model, Vehicle_Type, Vehicle_Staff_Price, Vehicle_Normal_Price, Vehicle_Fuel_Type)
11 VALUES ('V205', 'BMW', 'Adventure', 'Bike', '891', '990', 'Petrol')
12 INTO Vehicle (Vehicle_ID, Vehicle_Name, Vehicle_Model, Vehicle_Type, Vehicle_Staff_Price, Vehicle_Normal_Price, Vehicle_Fuel_Type)
13 VALUES ('V206', 'Hero', 'Destini 125', 'Scooter', '720', '800', 'Petrol')
14 INTO Vehicle (Vehicle_ID, Vehicle_Name, Vehicle_Model, Vehicle_Type, Vehicle_Staff_Price, Vehicle_Normal_Price, Vehicle_Fuel_Type)
15 VALUES ('V207', 'Volkswagen', 'Golf GTI', 'Car', '351', '390', 'Diesel')
16 INTO Vehicle (Vehicle_ID, Vehicle_Name, Vehicle_Model, Vehicle_Type, Vehicle_Staff_Price, Vehicle_Normal_Price, Vehicle_Fuel_Type)
17 VALUES ('V208', 'Benelli', 'Imperiale', 'Bike', '180', '200', 'Petrol')
18 INTO Vehicle (Vehicle_ID, Vehicle_Name, Vehicle_Model, Vehicle_Type, Vehicle_Staff_Price, Vehicle_Normal_Price, Vehicle_Fuel_Type)
19 VALUES ('V209', 'Lamborghini', 'Murcielago', 'Car', '774', '860', 'Diesel')
20 INTO Vehicle (Vehicle_ID, Vehicle_Name, Vehicle_Model, Vehicle_Type, Vehicle_Staff_Price, Vehicle_Normal_Price, Vehicle_Fuel_Type)
21 VALUES ('V210', 'Jaguar', 'XF', 'Car', '900', '1000', 'Diesel')
22 INTO Vehicle (Vehicle_ID, Vehicle_Name, Vehicle_Model, Vehicle_Type, Vehicle_Staff_Price, Vehicle_Normal_Price, Vehicle_Fuel_Type)
23 VALUES ('V211', 'Volkswagen', 'Golf GTI', 'Car', '351', '390', 'Diesel')
24 INTO Vehicle (Vehicle_ID, Vehicle_Name, Vehicle_Model, Vehicle_Type, Vehicle_Staff_Price, Vehicle_Normal_Price, Vehicle_Fuel_Type)
25 VALUES ('V212', 'Hero', 'Destini 125', 'Scooter', '720', '800', 'Petrol')
26 INTO Vehicle (Vehicle_ID, Vehicle_Name, Vehicle_Model, Vehicle_Type, Vehicle_Staff_Price, Vehicle_Normal_Price, Vehicle_Fuel_Type)
27 VALUES ('V213', 'Benelli', 'Imperiale', 'Bike', '180', '200', 'Petrol')
28 SELECT * FROM dual;
```

13 rows created.

Figure 19 Populating Database - Inserting Data of Vehicle

- We have used a line of code below to show the inserted Vehicle data more clearly: -

```
SQL> SELECT * FROM Vehicle;
```

VEHICLE_ID	VEHICLE_NAME	VEHICLE_MODEL	VEHICLE_TYPE	VEHICLE_STAFF_PRICE	VEHICLE_NORMAL_PRICE	VEHICLE_FUEL_TYPE
V201	Toyota	Prius	Car	450	500	Diesel
V202	Volkswagen	Golf GTI	Car	540	600	Diesel
V203	TVS	Jupiter	Scooter	720	800	Petrol
V204	Audi	A3	Car	855	950	Diesel
V205	BMW	Adventure	Bike	891	990	Petrol
V206	Hero	Destini 125	Scooter	720	800	Petrol
V207	Volkswagen	Golf GTI	Car	351	390	Diesel
V208	Benelli	Imperiale	Bike	180	200	Petrol
V209	Lamborghini	Murcielago	Car	774	860	Diesel
V210	Jaguar	XF	Car	900	1000	Diesel
V211	Volkswagen	Golf GTI	Car	351	390	Diesel
V212	Hero	Destini 125	Scooter	720	800	Petrol
V213	Benelli	Imperiale	Bike	180	200	Petrol

13 rows selected.

Figure 20 Populating Database - Vehicle data

7.4 Inserting the data of the Driver

To insert the data in Driver, I have written the following line of code: -

```
SQL> INSERT ALL
2   INTO Driver (Driver_ID, Driver_Name, Driver_Address, Driver_Type, Driver_Availability)
3   VALUES ('D201', 'Ayush Dhimal', 'Itahari', 'Full-Time', 'Yes')
4   INTO Driver (Driver_ID, Driver_Name, Driver_Address, Driver_Type, Driver_Availability)
5   VALUES ('D202', 'Sanjey Gurung', 'Biratnagar', 'Part-Time', 'No')
6   INTO Driver (Driver_ID, Driver_Name, Driver_Address, Driver_Type, Driver_Availability)
7   VALUES ('D203', 'Sunil Acharya', 'Dharan', 'Full-Time', 'Yes')
8   INTO Driver (Driver_ID, Driver_Name, Driver_Address, Driver_Type, Driver_Availability)
9   VALUES ('D204', 'Anish Rai', 'Salakpur', 'Full-Time', 'No')
10  INTO Driver (Driver_ID, Driver_Name, Driver_Address, Driver_Type, Driver_Availability)
11  VALUES ('D205', 'Ashim Jha', 'Halgada', 'Full-Time', 'Yes')
12  INTO Driver (Driver_ID, Driver_Name, Driver_Address, Driver_Type, Driver_Availability)
13  VALUES ('D206', 'Numa Limbu', 'Dhamak', 'Part-Time', 'No')
14  INTO Driver (Driver_ID, Driver_Name, Driver_Address, Driver_Type, Driver_Availability)
15  VALUES ('D207', 'Apil Rai', 'Inaruwa', 'Part-Time', 'No')
16  INTO Driver (Driver_ID, Driver_Name, Driver_Address, Driver_Type, Driver_Availability)
17  VALUES ('D208', 'Sujan Basnet', 'Birtamode', 'Part-Time', 'Yes')
18  INTO Driver (Driver_ID, Driver_Name, Driver_Address, Driver_Type, Driver_Availability)
19  VALUES ('D209', 'Birat Kholi', 'Lahan', 'Part-Time', 'No')
20  INTO Driver (Driver_ID, Driver_Name, Driver_Address, Driver_Type, Driver_Availability)
21  VALUES ('D210', 'Anushka Kholi', 'Dhulhari', 'Full-Time', 'Yes')
22  SELECT * FROM dual;
```

10 rows created.

Figure 21 Populating Database - Inserting Data of Driver

- We have used a line of code below to show the inserted Driver data more clearly:

DRIVER_ID	DRIVER_NAME	DRIVER_ADDRESS	DRIVER_TYPE	DRIVER_AVAILABILITY
D201	Ayush Dhimal	Itahari	Full-Time	Yes
D202	Sanje Gurung	Biratnagar	Part-Time	No
D203	Sunil Acharya	Dharan	Full-Time	Yes
D204	Anish Rai	Salakpur	Full-Time	No
D205	Ashim Jha	Halgada	Full-Time	Yes
D206	Numa Limbu	Dhamak	Part-Time	No
D207	Apil Rai	Inaruwa	Part-Time	No
D208	Sujan Basnet	Birtamode	Part-Time	Yes
D209	Birat Kholi	Lahan	Part-Time	No
D210	Anushka Kholi	Dhulari	Full-Time	Yes

10 rows selected.

Figure 22 Populating Database - Driver Data

7.5 Inserting the data of the Invoice

To insert the data in the Invoice, I have written the following line of code: -

```
SQL> INSERT ALL
2   INTO Invoice (Invoice_Ticket_No, Service_Date, Service_Charge, Service_Duration, Service_Destination, Total_Charge, Customer_ID, Vehicle_ID, Driver_ID, Service_ID)
3   VALUES ('I201', '25-Jan-2022', '500', '1 hour', 'Dharan', '500', 'C201', 'V201', 'D201', 'S201')
4   INTO Invoice (Invoice_Ticket_No, Service_Date, Service_Charge, Service_Duration, Service_Destination, Total_Charge, Customer_ID, Vehicle_ID, Driver_ID, Service_ID)
5   VALUES ('I202', '31-Jan-2022', '600', '2 hour', 'Biratnagar', '1200', 'C202', 'V208', 'D203', 'S201')
6   INTO Invoice (Invoice_Ticket_No, Service_Date, Service_Charge, Service_Duration, Service_Destination, Total_Charge, Customer_ID, Vehicle_ID, Driver_ID, Service_ID)
7   VALUES ('I203', '12-Feb-2022', '950', '6 hour', 'Lahan', '5700', 'C204', 'V205', 'D205', 'S202')
8   INTO Invoice (Invoice_Ticket_No, Service_Date, Service_Charge, Service_Duration, Service_Destination, Total_Charge, Customer_ID, Vehicle_ID, Driver_ID, Service_ID)
9   VALUES ('I204', '08-March-2022', '200', '3 hour', 'Salakpur', '600', 'C208', 'V208', 'D208', 'S203')
10  INTO Invoice (Invoice_Ticket_No, Service_Date, Service_Charge, Service_Duration, Service_Destination, Total_Charge, Customer_ID, Vehicle_ID, Driver_ID, Service_ID)
11  VALUES ('I205', '12-March-2022', '180', '1 hour', 'Halgada', '180', 'C213', 'V210', 'D210', 'S202')
12  INTO Invoice (Invoice_Ticket_No, Service_Date, Service_Charge, Service_Duration, Service_Destination, Total_Charge, Customer_ID, Vehicle_ID, Driver_ID, Service_ID)
13  VALUES ('I206', '25-March-2022', '180', '2 hour', 'Bhadrapur', '360', 'C213', 'V208', 'D208', 'S201')
14  INTO Invoice (Invoice_Ticket_No, Service_Date, Service_Charge, Service_Duration, Service_Destination, Total_Charge, Customer_ID, Vehicle_ID, Driver_ID, Service_ID)
15  VALUES ('I207', '03-April-2022', '800', '8 hour', 'Inaruwa', '6400', 'C212', 'V205', 'D205', 'S202')
16  INTO Invoice (Invoice_Ticket_No, Service_Date, Service_Charge, Service_Duration, Service_Destination, Total_Charge, Customer_ID, Vehicle_ID, Driver_ID, Service_ID)
17  VALUES ('I208', '12-Nov-2022', '200', '2 hour', 'Salakpur', '400', 'C208', 'V208', 'D208', 'S203')
18  INTO Invoice (Invoice_Ticket_No, Service_Date, Service_Charge, Service_Duration, Service_Destination, Total_Charge, Customer_ID, Vehicle_ID, Driver_ID, Service_ID)
19  VALUES ('I209', '19-Dec-2022', '390', '2 hour', 'Dhulari', '780', 'C211', 'V203', 'D203', 'S203')
20  INTO Invoice (Invoice_Ticket_No, Service_Date, Service_Charge, Service_Duration, Service_Destination, Total_Charge, Customer_ID, Vehicle_ID, Driver_ID, Service_ID)
21  VALUES ('I210', '06-Dec-2022', '200', '4 hour', 'Biratnagar', '800', 'C208', 'V201', 'D201', 'S202')
22  SELECT * FROM dual;

10 rows created.
```

Figure 23 Populating Database - Inserting Data of Invoice

- We have used a line of code below to show the inserted Invoice data more clearly: -

INVOICE_TICKET_NO	SERVICE_D	SERVICE_CHARGE	SERVICE_DURATION	SERVICE_DESTINATION	TOTAL_CHARGE	CUSTOMER_ID	VEHICLE_ID
DRIVER_ID	SERVICE_ID						
I201 D201	25-JAN-22 S201	500	1 hour	Dharan	500	C201	V201
I202 D203	31-JAN-22 S201	600	2 hour	Biratnagar	1200	C202	V208
I203 D205	12-FEB-22 S202	950	6 hour	Lahan	5700	C204	V205
I204 D208	08-MAR-22 S203	200	3 hour	Salakpur	600	C208	V208
I205 D210	12-MAR-22 S202	180	1 hour	Halgada	180	C213	V210
I206 D208	25-MAR-22 S201	180	2 hour	Bhadrapur	360	C213	V208
I207 D205	03-APR-22 S202	800	8 hour	Inaruwa	6400	C212	V205
I208 D208	12-NOV-22 S203	200	2 hour	Salakpur	400	C208	V208
I209 D203	19-DEC-22 S203	390	2 hour	Dhulari	780	C211	V203
I210 D201	06-DEC-22 S202	200	4 hour	Biratnagar	800	C208	V201

10 rows selected.

Figure 24 Populating Database - Invoice Data

8 Database Query

8.1 Informational Queries

- a. List all customers according to category.

```
SQL> SELECT * FROM Customer
2 ORDER BY Customer_Category;
```

CUSTOMER_ID	CUSTOMER_NAME	CUSTOMER_ADDRESS	CUSTOMER_PHONE_NO	CUSTOMER_CATEGORY	CUSTOMER_REWARD_POINT
C201	Sricha Parajuli	Dharan	9852055474	Normal	50
C215	Shrasta Niraula	Biratnagar	9842112943	Normal	20
C202	Shrasta Niraula	Biratnagar	9842112943	Normal	20
C214	Sushmeeta Jha	Lahan	9832456176	Normal	20
C212	Dikshya Dhimal	Inaruwa	9864532147	Normal	10
C211	Ranjita Limbu	Bhadrapur	9852136547	Normal	55
C204	Alisha Rai	Urlabari	9842020991	Normal	60
C205	Shrasta Niraula	Biratnagar	9842112943	Normal	20
C208	Sampada Regmi	Birtamode	9802775699	Normal	65
C209	Shrasta Niraula	Biratnagar	9842112943	Normal	20
C210	Shrasta Niraula	Biratnagar	9842112943	Normal	20
C206	Deepika Rijal	Salakpur	9842056621	Staff	80
C213	Geenesh Acharya	Halgada	9800967301	Staff	80
C207	Geenesh Acharya	Halgada	9800967301	Staff	80
C203	Shreya Bhandari	Belbari	9804042626	Staff	35

15 rows selected.

Figure 25 Informational Queries - List of all Customer Category

- b. Find model and vehicle variants and sort by price in descending order.

```
SQL> SELECT Vehicle_Model, Vehicle_Type, Vehicle_Staff_Price
2 FROM Vehicle
3 ORDER BY Vehicle_Staff_Price DESC;
```

VEHICLE_MODEL	VEHICLE_TYPE	VEHICLE_STAFF_PRICE
XF	Car	900
Adventure	Bike	891
A3	Car	855
Murcielago	Car	774
Jupiter	Scooter	720
Destini 125	Scooter	720
Destini 125	Scooter	720

VEHICLE_MODEL	VEHICLE_TYPE	VEHICLE_STAFF_PRICE
Golf GTI	Car	540
Prius	Car	450
Golf GTI	Car	351
Golf GTI	Car	351
Imperiale	Bike	180
Imperiale	Bike	180

13 rows selected.

Figure 26 Informational Queries - Model and Variants price in descending order

- c. Display the number of total vehicles that use petrol.

```
SQL> SELECT COUNT ('Vehicle_ID')
2 As Vehicle_Fuel_Type
3 FROM Vehicle
4 WHERE Vehicle_Fuel_Type='Petrol';

VEHICLE_FUEL_TYPE
-----
6

SQL>
```

Figure 27 Informational Queries - Number of Vehicle using Petrol

- d. List all tickets issued from 2022/03/05 to 2022/04/05.

```
SQL> Set linesize 150;
SQL> SELECT * FROM Invoice
  2   WHERE Service_Date BETWEEN
  3   '5-March-2022' AND '5-April-2022';
```

INVOICE_TICKET_NO	SERVICE_D	SERVICE_CHARGE	SERVICE_DURATION	SERVICE_DESTINATION	CUSTOMER_ID	VEHICLE_ID
I204 D208	08-MAR-22 200 S203		3 hour	Salakpur	C208	V208
I205 D210	12-MAR-22 180 S202		1 hour	Halgada	C213	V210
I206 D208	25-MAR-22 180 S201		2 hour	Bhadrapur	C213	V208
I207 D205	03-APR-22 800 S202		8 hour	Inaruwa	C212	V205

Figure 28 Informational Queries - List of Issued Ticket

- e. List the name of the Driver with the character 's' between their words.

```
SQL> SELECT Driver_Name FROM Driver
  2   WHERE REGEXP_Like(Driver_Name, '[s]');

DRIVER_NAME
-----
Ayush Dhimal
Anish Rai
Ashim Jha
Sujan Basnet
Anushka Kholi
```

Figure 29 Informational Queries - List of names which have 's' in middle.

8.2 Transactional Queries with Relational Algebra

- a. Show the total cost and the type of Service of a particular customer in a year who has used the Service.

```
SQL> SELECT CUSTOMER.CUSTOMER_NAME,
2  SUM (INVOICE.TOTAL_CHARGE) AS total_cost,
3  Service.Service_Name
4  FROM Invoice
5  JOIN CUSTOMER ON CUSTOMER.CUSTOMER_ID = INVOICE.CUSTOMER_ID
6  JOIN SERVICE ON SERVICE.SERVICE_id = INVOICE.SERVICE_id
7  WHERE customer.customer_Id = 'C202'
8  AND TO_CHAR(SERVICE_DATE, 'YYYY')='2022'
9  GROUP BY CUSTOMER.CUSTOMER_NAME, Service.SERVICE_Name;
```

CUSTOMER_NAME	TOTAL_COST	SERVICE_NAME
Shrasta Niraula	1200	Food Delivery

```
SQL> |
```

Figure 30 Transactional Queries - Figure 1

➤ Relational Algebra

R1: σ Customer.Customer_Id='C01' AND To_char(Service_date, 'yyyy')='2022'
(Invoice)

R2: $R1 \bowtie R1.customer_id = customer.customer_id$ (Customer)

R3: $R2 \bowtie R2.service_id = services.service_id$ (Services)

R4: π customer_name, SUM(TotalCharge) as Total_cost (**R3**)

R5: **R4** GROUP BY service_Id, Service_type, .Driver_name (**R4**)

- b. List the details of services provided by a driver for the current month whose first name starts with the letter 'A.'

```
SQL> SELECT service.service_Id, service.Service_Name, driver.driver_name
2 FROM INVOICE
3 JOIN CUSTOMER ON CUSTOMER.CUSTOMER_ID = INVOICE.CUSTOMER_ID
4 JOIN DRIVER ON DRIVER.DRIVER_ID = INVOICE.DRIVER_ID
5 JOIN SERVICE ON SERVICE.SERVICE_ID = INVOICE.SERVICE_ID
6 WHERE DRIVER.DRIVER_NAME LIKE 'A%'
7 AND TO_CHAR(SERVICE_DATE, 'MON')='MAR'
8 GROUP BY SERVICE.SERVICE_ID, SERVICE.Service_Name, DRIVER.DRIVER_NAME;
```

SERVICE_ID	SERVICE_NAME	DRIVER_NAME
S202	Package Delivery	Anushka Kholi

Figure 31 Transactional Queries - Figure 2

➤ Relational Algebra

R1: σ DRIVER.DRIVER_NAME LIKE 'A%' AND TO_CHAR(SERVICE_DATE,'MON')='MAR' (Invoice)

R2: $R1 \bowtie R1.customer_id = customer.customer_id$ (Customer)

R3: $R2 \bowtie R2.service_id = services.service_id$ (Services)

R4: $R3 \bowtie R3.Driver_id = Driver.Driver_Id$ (Driver)

R5: π Service_Id,Service_Type,Driver_name (**R4**)

R6: R5 GROUP BY service_Id, Service_type, Driver_name (**R5**)

- c. List the details of customers who have used only courier service and their delivery location.

```
SQL> Set linesize 200;
SQL> SELECT Service.Service_ID S_ID,
2 Service.Service_Name,
3 Customer.Customer_ID CID,
4 Customer.Customer_Name,
5 Invoice.Service_Destination
6 FROM INVOICE, Customer, Service
7 WHERE Customer.Customer_Id = Invoice.Customer_ID
8 AND Service.service_id= Invoice.service_ID
9 AND Invoice.Customer_Id IN (
10 SELECT Customer_ID FROM Service,
11 Invoice WHERE Service.Service_Id = Invoice.Service_Id
12 AND Service.Service_Name= 'Food Delivery' MINUS
13 SELECT Customer_Id FROM Service,
14 Invoice WHERE Service.Service_Id = Invoice.Service_Id
15 AND Service.Service_Name != 'Food Delivery');
```

S_ID	SERVICE_NAME	CID	CUSTOMER_NAME	SERVICE_DESTINATION
S201	Food Delivery	C201	Sricha Parajuli	Dharan
S201	Food Delivery	C202	Shrasta Niraula	Biratnagar

Figure 32 Transactional Queries - Figure 3

➤ Relational Algebra

R1: Service \bowtie Service.Service_Id= Invoice.Service_Id (Invoice)

R2: σ Service_type != 'Food Delivery'(R1)

R3: π Customer_Id(R2)

R4: σ R1.service_type = 'Food Delivery' (R1)

R5: π Customer_Id (R4)

R6: R5-R3

R7: Customer \bowtie Customer.Customer_Id = R1.customer_Id (R1)

R8: π Service_Id, Service_type, Customer_Id, Customer_name, Destination (R7)

R9: Customer_Id IN R6 (R8)

- d. List all the details of the top 3 highest earning drivers.

```
SQL> SELECT * FROM
  2 (SELECT SUM(Total_charge) AS Total_Income,
  3 Driver.Driver_Id,
  4 Driver.Driver_Name,
  5 Driver.Driver_Address,
  6 Driver.Driver_Type
  7 FROM Invoice
  8 JOIN Driver on Driver.Driver_ID = Invoice.Driver_ID
  9 GROUP BY (Driver.Driver_ID, Driver.Driver_Name,
  10 Driver.Driver_Address,
  11 Driver.Driver_Type,
  12 Invoice.Driver_ID)
  13 ORDER BY SUM(Total_charge) DESC)
  14 WHERE rownum <= 3;
```

TOTAL_INCOME	DRIVER_ID	DRIVER_NAME	DRIVER_ADDRESS	DRIVER_TYPE
12100	D205	Ashim Jha	Halgada	Full-Time
3180	D203	Sunil Acharya	Dharan	Full-Time
1360	D208	Sujan Basnet	Birtamode	Part-Time

```
SQL> |
```

Figure 33 Transactional Queries - Figure 4

➤ Relational Algebra

R1: Invoice \bowtie Driver.driverID = Invoice.driverID (Driver)

R2: π sum(totalCharge) As Average_Income, Driver.driverID, Driver.driverName, Driver.emergencyCont, Driver.home_location, Driver.Working_Status (**R1**)

R3: R2 Group By (Driver.driverID, Driver.driverName, Driver.emergencyCont, Driver.home_location, Driver.Working_Status, Invoice.driverID)

R4: R3 Order By Sum(totalCharge) Desc)

R5: σ rowNum <= 3 (**R4**)

- e. Display the rate of all vehicles for staff and normal customers on a particular destination.

```
SQL> SELECT
  2 Invoice.Invoice_Ticket_No,
  3 Customer.Customer_Name,
  4 Customer.Customer_Category,
  5 Vehicle.Vehicle_Type,
  6 Vehicle.Vehicle_Model,
  7 Invoice.Service_Destination,
  8 Invoice.Service_Duration,
  9 Invoice.Total_charge
10 FROM Vehicle
11 JOIN Invoice on Invoice.vehicle_ID = Vehicle.vehicle_ID
12 JOIN Customer ON Customer.Customer_ID = Invoice.Customer_ID
13 WHERE Invoice.Service_Destination = 'Biratnagar';
```

INVOICE_TICKET_NO	CUSTOMER_NAME	CUSTOMER_CATEGORY	VEHICLE_TYPE	VEHICLE_MODEL	SERVICE_DESTINATION	SERVICE_DURATION	TOTAL_CHARGE
I202	Shrasta Niraula	Normal	Bike	Imperiale	Biratnagar	2 hour	1200
I210	Sampada Regmi	Normal	Car	Prius	Biratnagar	4 hour	800
I211	Suyog Shakya	Staff	Bike	Imperiale	Biratnagar	2 hour	1080

Figure 34 Transactional Queries - Figure 5

➤ Relational Algebra

R1: σ Invoice.Destination='Biratnagar'

R2: $R1 \bowtie R1.Vehicle_id = Invoice.Vehicle_Id$ (Invoice)

R3: $R2 \bowtie R2.Customer_id = Customer_customer_Id$ (Customer)

R4: π Customer_name, category, vehicle_varient, vehicle_brand, pickup, destination, duration, Total_charge (**R3**)

9 Critical Evaluation

9.1 Further discussion on the learning experience

This course was focused on developing a database for storing transportation company data. When this coursework was assigned, I experienced difficulty and confusion in understanding the scenarios presented in the coursework. I had a difficult time analyzing the case scenario. I was required to visit multiple websites to research various transportation companies. The most challenging aspect of this training was identifying entities and their attributes from the scenario.

This course was quite challenging for me. Despite knowing databases, I was unable to implement them. As a result of the problematic scenarios described in the coursework, I simultaneously found it incredibly educational and challenging. More than other parts, normalization processes were unfamiliar to me, and I found it challenging to understand them in class. Normalization was the most challenging aspect of this assignment, and analyzing and identifying entities and attributes was even more complicated. With the help of my friends and the module leaders, I overcame the obstacles. Throughout my assignment, I read every piece of information I could find on the internet. Moreover, I gained a better understanding of the significance of Normalization and ER due to this course.

9.2 Critical Assessment

Databases are one of the most significant modules for us because it teaches us about normalizing, case scenarios, and drawing ER diagrams, among other things. As a student, learning how to manage massive amounts of data is critical. The knowledge I gained from this course has also been quite beneficial. This coursework and other classes have improved my comprehension of effective problem-solving and data summarization.

As a result of taking this module, I gained a deeper understanding of Normalization, which plays a vital role in database development. Data anomalies and redundancy are reduced by dividing entities. As a result of the information gathered in this module, we have been able to create the entity relational ship diagram. We have also completed the database storage for the software application, which we have made for the software coursework.

This report relates to the software engineering module. Software engineering aims to analyze user needs and then design, construct, and test software applications to meet those needs. Thus, this Database can be used to develop software that provides back-end services. We can implement these services in the Network Operating System Module for storing user information using the Active Directory System. Users and administrators can easily access the information to enhance their skills in other modules.

10 Conclusion

Finally, I concluded my coursework on Database; I was a little worried and scared before I started the coursework because of some challenges in Queries. After resolving faults by researching them on internet sources and looking through samples, I found it more fascinating. This coursework on creating a database has dramatically helped me learn different things about Databases. This gives me a clear and profound concept of syntax and methods used to create and insert a database in the SQL command line.

This coursework gives me more confidence in creating tables, inserting values, adding rows, or deleting columns. This coursework helped all students to properly understand the topic and help develop and understand the right ideas for this Database. This coursework focuses on Queries, Normalization, and ER diagrams. By creating a Database, we have knowledge of Normalization, Entity relation diagram, and queries. Each student has a good understanding of topics related to creating databases and entity-relationship diagrams. There were times when I had a hard time making the Database, but the help of my teacher and the content I uploaded to Google Classroom helped me.

Moreover, through it, we gained experience in several things about database Queries, Normalization, ER diagrams, and many more, which also helped us advance our skills. In this manner, I completed my task with the assistance of my classmate, module instructor Ronal Niraula want to express my gratitude to everyone for their support.

11 Creating Dump File

In order to create a dumb file, I use the command HOST to go to the System, and then I use the CD command to locate its location where it should be saved. In the next step, I used the exp user and password, then entered the file's name as ".dmp." The dump file was successfully created.

```
SQL> Host
Microsoft Windows [Version 10.0.22621.963]
(c) Microsoft Corporation. All rights reserved.

C:\oracle\app\oracle\product\11.2.0\server\bin>cd C:\Users\geene\Desktop\22016051-Database-Geenesh Acharya
C:\Users\geene\Desktop\22016051-Database-Geenesh Acharya>exp zenas/zenas file=22016051.dmp

Export: Release 11.2.0.2.0 - Production on Thu Jan 5 09:02:08 2023

Copyright (c) 1982, 2009, Oracle and/or its affiliates. All rights reserved.

Connected to: Oracle Database 11g Express Edition Release 11.2.0.2.0 - 64bit Production
Export done in WE8MSWIN1252 character set and AL16UTF16 NCHAR character set
server uses AL32UTF8 character set (possible charset conversion)
. exporting pre-schema procedural objects and actions
. exporting foreign function library names for user ZENAS
. exporting PUBLIC type synonyms
. exporting private type synonyms
. exporting object type definitions for user ZENAS
About to export ZENAS's objects ...
. exporting database links
. exporting sequence numbers
. exporting cluster definitions
. about to export ZENAS's tables via Conventional Path ...
. . exporting table          CUSTOMER          16 rows exported
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
. . exporting table          DRIVER            10 rows exported
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
. . exporting table          INVOICE           11 rows exported
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
. . exporting table          SERVICE            3 rows exported
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
. . exporting table          VEHICLE           13 rows exported

EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
. exporting synonyms
. exporting views
. exporting stored procedures
. exporting operators
. exporting referential integrity constraints
. exporting triggers
. exporting indextypes
. exporting bitmap, functional and extensible indexes
```

```
. exporting bitmap, functional and extensible indexes
. exporting posttables actions
. exporting materialized views
. exporting snapshot logs
. exporting job queues
. exporting refresh groups and children
. exporting dimensions
. exporting post-schema procedural objects and actions
. exporting statistics
Export terminated successfully with warnings.

C:\Users\geene\Desktop\22016051-Database-Geenesh Acharya>
```

Figure 35 Screenshot of Dump File

12 References

Techopedia. (2017). *Entity-Relationship Diagram (ERD)*. Retrieved 12 18, 2022, from <https://www.techopedia.com/definition/1200/entity-relationship-diagram-erd>

13 Appendix

13.1 SQL Command Line code

In the Elevate transportation system, the following code is used in the SQL command line, and they are: -

13.2 Creating Table Code

13.2.1 Customer Data code

```
CREATE TABLE Customer (  
  
    Customer_ID VARCHAR2(20) NOT NULL,  
  
    Customer_Name VARCHAR2(20) NOT NULL,  
  
    Customer_Address VARCHAR2(20) NOT NULL,  
  
    Customer_Phone_No INT NOT NULL,  
  
    Customer_Category VARCHAR2(20) NOT NULL,  
  
    Customer_Reward_Point INT NOT NULL,  
  
    CONSTRAINT PK_Customer PRIMARY KEY (Customer_ID));
```

13.2.2 Vehicle Data code

```
CREATE TABLE Vehicle (  
  
    Vehicle_ID VARCHAR2(20) NOT NULL,  
  
    Vehicle_Name VARCHAR2(20) NOT NULL,  
  
    Vehicle_Model VARCHAR2(20) NOT NULL,  
  
    Vehicle_Type VARCHAR2(20) NOT NULL,  
  
    Vehicle_Staff_Price INT NOT NULL,  
  
    Vehicle_Normal_Price INT NOT NULL,  
  
    Vehicle_Fuel_Type VARCHAR2(20) NOT NULL,  
  
    CONSTRAINT PK_Vehicle PRIMARY KEY (Vehicle_ID));
```

13.2.3 Service Data code

```
CREATE TABLE Service (  
  
    Service_ID VARCHAR2(20) NOT NULL,  
  
    Service_Name VARCHAR2(20) NOT NULL,  
  
    CONSTRAINT PK_Service PRIMARY KEY (Service_ID));
```

13.2.4 Invoice Data code

```
CREATE TABLE Invoice (  
  
    Invoice_Ticket_No VARCHAR2(20) NOT NULL,  
  
    Service_Date DATE NOT NULL,  
  
    Service_Charge VARCHAR2(20) NOT NULL,  
  
    Service_Duration VARCHAR2(20) NOT NULL,  
  
    Service_Destination VARCHAR2(20) NOT NULL,  
  
    Total_Charge INT NOT NULL,  
  
    Customer_ID VARCHAR2(20) NOT NULL,  
  
    Vehicle_ID VARCHAR2(20) NOT NULL,  
  
    Driver_ID VARCHAR2(20) NOT NULL,  
  
    Service_ID VARCHAR2(20) NOT NULL,  
  
    CONSTRAINT PK_Invoice_Ticket_No PRIMARY KEY (Invoice_Ticket_No),  
  
    CONSTRAINT FK_Customer_ID FOREIGN KEY (Customer_ID) REFERENCES  
Customer (Customer_ID),  
  
    CONSTRAINT FK_Vehicle_ID FOREIGN KEY (Vehicle_ID) REFERENCES Vehicle  
(Vehicle_ID),  
  
    CONSTRAINT FK_Driver_ID FOREIGN KEY (Driver_ID) REFERENCES Driver  
(Driver_ID),
```



```
CONSTRAINT FK_Service_ID FOREIGN KEY (Service_ID) REFERENCES Service  
(Service_ID));
```

13.3 Describe Table Code

13.3.1 Table of Customer

```
DESCRIBE Customer;
```

13.3.2 Table of Vehicle

```
DESCRIBE Vehicle;
```

13.3.3 Table of Service

```
DESCRIBE Service;
```

13.3.4 Table of Invoice

```
DESCRIBE Invoice;
```

13.4 Inserting Data Code

13.4.1 Inserting Customer Data

```
INSERT ALL
```

```
    INTO Customer (Customer_ID, Customer_Name, Customer_Address,  
Customer_Phone_No, Customer_Category, Customer_Reward_Point)
```

```
VALUES ('C201', 'Sricha Parajuli', 'Dharan', '9852055474', 'Normal', '50')
```

```
    INTO Customer (Customer_ID, Customer_Name, Customer_Address,  
Customer_Phone_No, Customer_Category, Customer_Reward_Point)
```

```
VALUES ('C202', 'Shrasta Niraula', 'Biratnagar', '9842112943', 'Normal', '20')
```

```
    INTO Customer (Customer_ID, Customer_Name, Customer_Address,  
Customer_Phone_No, Customer_Category, Customer_Reward_Point)
```

```
VALUES ('C203', 'Shreya Bhandari', 'Belbari', '9804042626', 'Staff', '35')
```

```
    INTO Customer (Customer_ID, Customer_Name, Customer_Address,  
Customer_Phone_No, Customer_Category, Customer_Reward_Point)
```

```
VALUES ('C204', 'Alisha Rai', 'Urlabari', '9842020991', 'Normal', '60')
```

```
INTO Customer (Customer_ID, Customer_Name, Customer_Address,  
Customer_Phone_No, Customer_Category, Customer_Reward_Point)
```

```
VALUES ('C205', 'Shrasta Niraula', 'Biratnagar', '9842112943', 'Normal', '20')
```

```
INTO Customer (Customer_ID, Customer_Name, Customer_Address,  
Customer_Phone_No, Customer_Category, Customer_Reward_Point)
```

```
VALUES ('C206', 'Deepika Rijal', 'Salakpur', '9842056621', 'Staff', '80')
```

```
INTO Customer (Customer_ID, Customer_Name, Customer_Address,  
Customer_Phone_No, Customer_Category, Customer_Reward_Point)
```

```
VALUES ('C207', 'Geenesh Acharya', 'Halgada', '9800967301', 'Staff', '80')
```

```
INTO Customer (Customer_ID, Customer_Name, Customer_Address,  
Customer_Phone_No, Customer_Category, Customer_Reward_Point)
```

```
VALUES ('C208', 'Sampada Regmi', 'Birtamode', '9802775699', 'Normal', '65')
```

```
INTO Customer (Customer_ID, Customer_Name, Customer_Address,  
Customer_Phone_No, Customer_Category, Customer_Reward_Point)
```

```
VALUES ('C209', 'Shrasta Niraula', 'Biratnagar', '9842112943', 'Normal', '20')
```

```
INTO Customer (Customer_ID, Customer_Name, Customer_Address,  
Customer_Phone_No, Customer_Category, Customer_Reward_Point)
```

```
VALUES ('C210', 'Shrasta Niraula', 'Biratnagar', '9842112943', 'Normal', '20')
```

```
INTO Customer (Customer_ID, Customer_Name, Customer_Address,  
Customer_Phone_No, Customer_Category, Customer_Reward_Point)
```

```
VALUES ('C211', 'Ranjita Limbu', 'Bhadrapur', '9852136547', 'Normal', '55')
```

```
INTO Customer (Customer_ID, Customer_Name, Customer_Address,  
Customer_Phone_No, Customer_Category, Customer_Reward_Point)
```

```
VALUES ('C212', 'Dikshya Dhimal', 'Inaruwa', '9864532147', 'Normal', '10')
```

```
INTO Customer (Customer_ID, Customer_Name, Customer_Address,  
Customer_Phone_No, Customer_Category, Customer_Reward_Point)
```

```
VALUES ('C213', 'Geenesh Acharya', 'Halgada', '9800967301', 'Staff', '80')
```

```
INTO Customer (Customer_ID, Customer_Name, Customer_Address,  
Customer_Phone_No, Customer_Category, Customer_Reward_Point)
```

```
VALUES ('C214', 'Sushmeeta Jha', 'Lahan', '9832456176', 'Normal', '20')
```

```
INTO Customer (Customer_ID, Customer_Name, Customer_Address,  
Customer_Phone_No, Customer_Category, Customer_Reward_Point)
```

```
VALUES ('C215', 'Shrasta Niraula', 'Biratnagar', '9842112943', 'Normal', '20')
```

```
INTO Customer (Customer_ID, Customer_Name, Customer_Address,  
Customer_Phone_No, Customer_Category, Customer_Reward_Point)
```

```
VALUES ('C216', 'Suyog Shakya', 'Biratnagar', '9842112943', 'Staff', '20')
```

```
SELECT * FROM dual;
```

13.4.2 Inserting Vehicle Data

```
INSERT ALL
```

```
INTO Vehicle (Vehicle_ID, Vehicle_Name, Vehicle_Model, Vehicle_Type,  
Vehicle_Staff_Price, Vehicle_Normal_Price, Vehicle_Fuel_Type)
```

```
VALUES ('V201', 'Toyota', 'Prius', 'Car', '450', '500', 'Diesel')
```

```
INTO Vehicle (Vehicle_ID, Vehicle_Name, Vehicle_Model, Vehicle_Type,  
Vehicle_Staff_Price, Vehicle_Normal_Price, Vehicle_Fuel_Type)
```

```
VALUES ('V202', 'Volkswagen', 'Golf GTI', 'Car', '540', '600', 'Diesel')
```

```
INTO Vehicle (Vehicle_ID, Vehicle_Name, Vehicle_Model, Vehicle_Type,  
Vehicle_Staff_Price, Vehicle_Normal_Price, Vehicle_Fuel_Type)
```

```
VALUES ('V203', 'TVS', 'Jupiter', 'Scooter', '720', '800', 'Petrol')
```

```
INTO Vehicle (Vehicle_ID, Vehicle_Name, Vehicle_Model, Vehicle_Type,  
Vehicle_Staff_Price, Vehicle_Normal_Price, Vehicle_Fuel_Type)
```

```
VALUES ('V204', 'Audi', 'A3', 'Car', '855', '950', 'Diesel')
```

```
INTO Vehicle (Vehicle_ID, Vehicle_Name, Vehicle_Model, Vehicle_Type,  
Vehicle_Staff_Price, Vehicle_Normal_Price, Vehicle_Fuel_Type)
```

```
VALUES ('V205', 'BMW', 'Adventure', 'Bike', '891', '990', 'Petrol')
```

```
INTO Vehicle (Vehicle_ID, Vehicle_Name, Vehicle_Model, Vehicle_Type,  
Vehicle_Staff_Price, Vehicle_Normal_Price, Vehicle_Fuel_Type)
```

```
VALUES ('V206', 'Hero', 'Destini 125', 'Scooter', '720', '800', 'Petrol')
```

```
INTO Vehicle (Vehicle_ID, Vehicle_Name, Vehicle_Model, Vehicle_Type,  
Vehicle_Staff_Price, Vehicle_Normal_Price, Vehicle_Fuel_Type)
```

```
VALUES ('V207', 'Volkswagen', 'Golf GTI', 'Car', '351', '390', 'Diesel')
```

```
INTO Vehicle (Vehicle_ID, Vehicle_Name, Vehicle_Model, Vehicle_Type,  
Vehicle_Staff_Price, Vehicle_Normal_Price, Vehicle_Fuel_Type)
```

```
VALUES ('V208', 'Benelli', 'Imperiale', 'Bike', '180', '200', 'Petrol')
```

```
INTO Vehicle (Vehicle_ID, Vehicle_Name, Vehicle_Model, Vehicle_Type,  
Vehicle_Staff_Price, Vehicle_Normal_Price, Vehicle_Fuel_Type)
```

```
VALUES ('V209', 'Lamborghini', 'Murcielago', 'Car', '774', '860', 'Diesel')
```

```
INTO Vehicle (Vehicle_ID, Vehicle_Name, Vehicle_Model, Vehicle_Type,  
Vehicle_Staff_Price, Vehicle_Normal_Price, Vehicle_Fuel_Type)
```

```
VALUES ('V210', 'Jaguar', 'XF', 'Car', '900', '1000', 'Diesel')
```

```
INTO Vehicle (Vehicle_ID, Vehicle_Name, Vehicle_Model, Vehicle_Type,  
Vehicle_Staff_Price, Vehicle_Normal_Price, Vehicle_Fuel_Type)
```

```
VALUES ('V211', 'Volkswagen', 'Golf GTI', 'Car', '351', '390', 'Diesel')
```

```
INTO Vehicle (Vehicle_ID, Vehicle_Name, Vehicle_Model, Vehicle_Type,  
Vehicle_Staff_Price, Vehicle_Normal_Price, Vehicle_Fuel_Type)
```

```
VALUES ('V212', 'Hero', 'Destini 125', 'Scooter', '720', '800', 'Petrol')
```

```
INTO Vehicle (Vehicle_ID, Vehicle_Name, Vehicle_Model, Vehicle_Type,  
Vehicle_Staff_Price, Vehicle_Normal_Price, Vehicle_Fuel_Type)
```

```
VALUES ('V213', 'Benelli', 'Imperiale', 'Bike', '180', '200', 'Petrol')
```

```
SELECT * FROM dual;
```

13.4.3 Inserting Service Data

```
INSERT ALL
```

```
INTO Service (Service_ID, Service_Name)
```

```
VALUES ('S201', 'Food Delivery')
```

```
INTO Service (Service_ID, Service_Name)
```

```
VALUES ('S202', 'Package Delivery')
```

```
INTO Service (Service_ID, Service_Name)
```

```
VALUES ('S203', 'Courier Service')
```

```
SELECT * FROM dual;
```

13.4.4 Inserting Driver Data

```
INSERT ALL
```

```
INTO Driver (Driver_ID, Driver_Name, Driver_Address, Driver_Type,  
Driver_Availability)
```

```
VALUES ('D201', 'Ayush Dhimal', 'Itahari', 'Full-Time', 'Yes')
```

```
INTO Driver (Driver_ID, Driver_Name, Driver_Address, Driver_Type,  
Driver_Availability)
```

```
VALUES ('D202', 'Sanjeey Gurung', 'Biratnagar', 'Part-Time', 'No')
```

```
INTO Driver (Driver_ID, Driver_Name, Driver_Address, Driver_Type,  
Driver_Availability)
```

```
VALUES ('D203', 'Sunil Acharya', 'Dharan', 'Full-Time', 'Yes')
```

```
INTO Driver (Driver_ID, Driver_Name, Driver_Address, Driver_Type,  
Driver_Availability)
```

```
VALUES ('D204', 'Anish Rai', 'Salakpur', 'Full-Time', 'No')
```

```
INTO Driver (Driver_ID, Driver_Name, Driver_Address, Driver_Type,  
Driver_Availability)
```

```
VALUES ('D205', 'Ashim Jha', 'Halgada', 'Full-Time', 'Yes')
```

```
INTO Driver (Driver_ID, Driver_Name, Driver_Address, Driver_Type,  
Driver_Availability)
```

```
VALUES ('D206', 'Numa Limbu', 'Dhamak', 'Part-Time', 'No')
```

```
INTO Driver (Driver_ID, Driver_Name, Driver_Address, Driver_Type,  
Driver_Availability)
```

```
VALUES ('D207', 'Apil Rai', 'Inaruwa', 'Part-Time', 'No')
```

```
INTO Driver (Driver_ID, Driver_Name, Driver_Address, Driver_Type,  
Driver_Availability)
```

```
VALUES ('D208', 'Sujan Basnet ', 'Birtamode', 'Part-Time', 'Yes')
```

```
INTO Driver (Driver_ID, Driver_Name, Driver_Address, Driver_Type,  
Driver_Availability)
```

```
VALUES ('D209', 'Birat Kholi', 'Lahan', 'Part-Time', 'No')
```

```
INTO Driver (Driver_ID, Driver_Name, Driver_Address, Driver_Type,  
Driver_Availability)
```

```
VALUES ('D210', 'Anushka Kholi', 'Dhulari', 'Full-Time', 'Yes')
```

```
SELECT * FROM dual;
```

13.4.5 Inserting Invoice Data

INSERT ALL

INTO Invoice (Invoice_Ticket_No, Service_Date, Service_Charge, Service_Duration, Service_Destination, Total_Charge, Customer_ID, Vehicle_ID, Driver_ID, Service_ID)

VALUES ('I201', '25-Jan-2022', '500', '1 hour', 'Dharan' , '500 ', 'C201', 'V201', 'D201', 'S201')

INTO Invoice (Invoice_Ticket_No, Service_Date, Service_Charge, Service_Duration, Service_Destination, Total_Charge, Customer_ID, Vehicle_ID, Driver_ID, Service_ID)

VALUES ('I202', '31-Jan-2022', '600', '2 hour', 'Biratnagar', '1200', 'C202', 'V208', 'D203', 'S201')

INTO Invoice (Invoice_Ticket_No, Service_Date, Service_Charge, Service_Duration, Service_Destination, Total_Charge, Customer_ID, Vehicle_ID, Driver_ID, Service_ID)

VALUES ('I203', '12-Feb-2022', '950', '6 hour', 'Lahan', '5700', 'C204', 'V205', 'D205', 'S202')

INTO Invoice (Invoice_Ticket_No, Service_Date, Service_Charge, Service_Duration, Service_Destination, Total_Charge, Customer_ID, Vehicle_ID, Driver_ID, Service_ID)

VALUES ('I204', '08-March-2022', '200', '3 hour', 'Salakpur', '600', 'C208', 'V208', 'D208', 'S203')

INTO Invoice (Invoice_Ticket_No, Service_Date, Service_Charge, Service_Duration, Service_Destination, Total_Charge, Customer_ID, Vehicle_ID, Driver_ID, Service_ID)

VALUES ('I205', '12-March-2022', '180', '1 hour', 'Halgada', '180', 'C213', 'V210', 'D210', 'S202')

INTO Invoice (Invoice_Ticket_No, Service_Date, Service_Charge, Service_Duration, Service_Destination, Total_Charge, Customer_ID, Vehicle_ID, Driver_ID, Service_ID)

VALUES ('I206', '25-March-2022', '180', '2 hour', 'Bhadrapur', '360', 'C213', 'V208', 'D208', 'S201')

```
INTO Invoice (Invoice_Ticket_No, Service_Date, Service_Charge, Service_Duration,  
Service_Destination, Total_Charge, Customer_ID, Vehicle_ID, Driver_ID, Service_ID)
```

```
VALUES ('I207', '03-April-2022', '800', '8 hour', 'Inaruwa', '6400', 'C212', 'V205',  
'D205', 'S202')
```

```
INTO Invoice (Invoice_Ticket_No, Service_Date, Service_Charge, Service_Duration,  
Service_Destination, Total_Charge, Customer_ID, Vehicle_ID, Driver_ID, Service_ID)
```

```
VALUES ('I208', '12-Nov-2022', '200', '2 hour', 'Salakpur', '400', 'C208', 'V208',  
'D208', 'S203')
```

```
INTO Invoice (Invoice_Ticket_No, Service_Date, Service_Charge, Service_Duration,  
Service_Destination, Total_Charge, Customer_ID, Vehicle_ID, Driver_ID, Service_ID)
```

```
VALUES ('I209', '19-Dec-2022', '390', '2 hour', 'Dhulari', '780', 'C211', 'V203',  
'D203', 'S203')
```

```
INTO Invoice (Invoice_Ticket_No, Service_Date, Service_Charge, Service_Duration,  
Service_Destination, Total_Charge, Customer_ID, Vehicle_ID, Driver_ID, Service_ID)
```

```
VALUES ('I210', '06-Dec-2022', '200', '4 hour', 'Biratnagar', '800', 'C208', 'V201',  
'D201', 'S202')
```

```
INTO Invoice (Invoice_Ticket_No, Service_Date, Service_Charge, Service_Duration,  
Service_Destination, Total_Charge, Customer_ID, Vehicle_ID, Driver_ID, Service_ID)
```

```
VALUES ('I211', '31-Jan-2022', '600', '2 hour', 'Biratnagar', '1200', 'C216', 'V208',  
'D203', 'S201');
```

```
SELECT * FROM dual;
```

13.5 Showing Data Code

13.5.1 Customer Data

```
SELECT * FROM Customer;
```

13.5.2 Vehicle Data

```
SELECT * FROM Vehicle;
```


13.5.3 Service Data

```
SELECT * FROM Service;
```

13.5.4 Driver Data

```
SELECT * FROM Driver;
```

13.5.5 Invoice Data

```
SELECT * FROM Invoice;
```

13.6 Informational Queries Code

1. List all customers according to category.

```
SELECT * FROM Customer  
ORDER BY Customer_Category;
```

2. Find model and vehicle variants and sort by price in descending order.

```
SELECT COUNT ('Vehicle_ID')  
As Vehicle_Fuel_Type  
FROM Vehicle  
WHERE Vehicle_Fuel_Type='Petrol';
```

3. Display the number of total vehicles that use petrol.

```
SELECT Vehicle_Model, Vehicle_Type, Vehicle_Staff_Price  
FROM Vehicle  
ORDER BY Vehicle_Staff_Price DESC;
```

4. List all tickets issued from 2022/03/05 to 2022/04/05.

```
SELECT * FROM Invoice  
WHERE Service_Date BETWEEN  
'5-March-2022' AND '5-April-2022';
```

5. List the name of the Driver who has the character ' s' between their names.

```
SELECT Driver_Name FROM Driver  
  
WHERE REGEXP_Like(Driver_Name,['s']);
```

13.7 Transactional Queries

- a. Show the total cost and the type of Service of a particular customer in a year that has used the Service.

```
SELECT CUSTOMER.CUSTOMER_NAME,  
SUM (INVOICE.TOTAL_CHARGE) AS total_cost,  
Service.Service_Name  
FROM Invoice  
JOIN      CUSTOMER      ON      CUSTOMER.CUSTOMER_ID      =  
INVOICE.CUSTOMER_ID  
JOIN SERVICE ON SERVICE.SERVICE_id = INVOICE.SERVICE_id  
WHERE customer.customer_Id = 'C202'  
AND TO_CHAR(SERVICE_DATE, 'YYYY')='2022'  
GROUP BY CUSTOMER.CUSTOMER_NAME, Service.SERVICE_Name;
```

- b. Show the total cost and the type of Service of a particular customer in a year that has used the Service.

```
SELECT Service.service_Id, Service.Service_Name, driver.driver_name  
FROM INVOICE  
  
JOIN CUSTOMER ON CUSTOMER.CUSTOMER_ID = INVOICE.CUSTOMER_ID  
JOIN DRIVER ON DRIVER.DRIVER_ID = INVOICE.DRIVER_ID  
JOIN SERVICE ON SERVICE.SERVICE_ID = INVOICE.SERVICE_ID  
WHERE DRIVER.DRIVER_NAME LIKE 'A%'  
AND TO_CHAR(SERVICE_DATE, 'MON')='MAR'  
  
GROUP      BY      SERVICE.SERVICE_ID,      SERVICE.SERVICE_Name,  
DRIVER.DRIVER_NAME;
```

- c. List the details of customers who have used only courier service and their location of delivery.

```
SELECT Service.Service_ID S_ID,  
Service.Service_Name,  
Customer.Customer_ID CID,
```

```
Customer.Customer_Name,  
Invoice.Service_Destination  
FROM INVOICE, Customer, Service  
WHERE Customer.Customer_Id = Invoice.Customer_ID  
AND Service.service_id= Invoice.service_ID  
AND Invoice.Customer_Id IN (  
SELECT Customer_ID FROM Service,  
Invoice WHERE Service.Service_Id = Invoice.Service_Id  
AND Service.Service_Name= 'Food Delivery' MINUS  
SELECT Customer_Id FROM Service,  
Invoice WHERE Service.Service_Id = Invoice.Service_Id  
AND Service.Service_Name != 'Food Delivery');
```

d. List all the details of the top 3 highest earning drivers.

```
SELECT * FROM  
(SELECT SUM(Total_charge) AS Total_Income,  
Driver.Driver_Id,  
Driver.Driver_Name,  
Driver.Driver_Address,  
Driver.Driver_Type  
FROM Invoice  
JOIN Driver on Driver.Driver_ID = Invoice.Driver_ID  
GROUP BY (Driver.Driver_ID, Driver.Driver_Name,  
Driver.Driver_Address,
```

Driver.Driver_Type,

Invoice.Driver_ID)

ORDER BY SUM(Total_charge) DESC)

WHERE rownum <= 3;

- e. Display the rate of all vehicles for staff and normal customers on a particular destination.

SELECT

Invoice.Invoice_Ticket_No,

Customer.Customer_Name,

Customer.Customer_Category,

Vehicle.Vehicle_Type,

Vehicle.Vehicle_Model,

Invoice.Service_Destination,

Invoice.Service_Duration,

Invoice.Total_charge

FROM Vehicle

JOIN Invoice on Invoice.vehicle_ID = Vehicle.vehicle_ID

JOIN Customer ON Customer.Customer_ID = Invoice.Customer_ID

WHERE Invoice.Service_Destination = 'Biratnagar';