LONDON METROPOLITAN UNIVERSITY

Islington college
(इस्लिङ्टन कलेज)

CC5051NI Databases

50% Individual Coursework

2022 Autumn

Student Name: Nahush Man Singh Karki

London Met ID: 22015811

Assignment Submission Date: Thursday, January 5, 2023

Word Count: 242

I confirm that I understand my coursework needs to be submitted online via Google Classroom under the relevant module page before the deadline in order 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.

Table of Content

Table of Figures

Table of Tables

Introduction

This is the initial project handed out for the database module. The purpose of this project is to create a delivery system that delivers people as well as items to the designated location. To achieve the result, a service has been created. The name of the service is called **Reached.** The purpose of this service is to reserve a vehicle to drop customers off at the specified location as per the customer's need and provide services that include food delivery, package delivery, and more.

This service picks and drops people based on the pointed-out location as well as drops packages alongside foods. As the client successfully finishes a trip, they get rewarded with points. This company works with stores to get the items delivered to the destinated place. When a customer puts in an order, the driver receives the order and gets sent out to deliver the item.

Business rules

Since it takes time and money to deliver to a far destination, it only sends the item within a city or close by cities. The office is in Kathmandu and functions around the neighbouring cities but within the covering distance.

When a customer finishes a ride, they receive an invoice or bill. The charge given to the customer is based on the distance covered, i.e., the rate, which is calculated in rupees/km. The charge also varies depending on the type of vehicle.

There are vehicles such as cars, bikes, bicycles, trucks, and scooters available. Cars and Trucks have an engine type that uses diesel, bikes and scooters have an engine type that uses petrol, but, as we already know, a bicycle does not use any gas.

One invoice is received per completion of a service. There are three types of service, personal delivery, food delivery, and item delivery/Courier service. If an order is

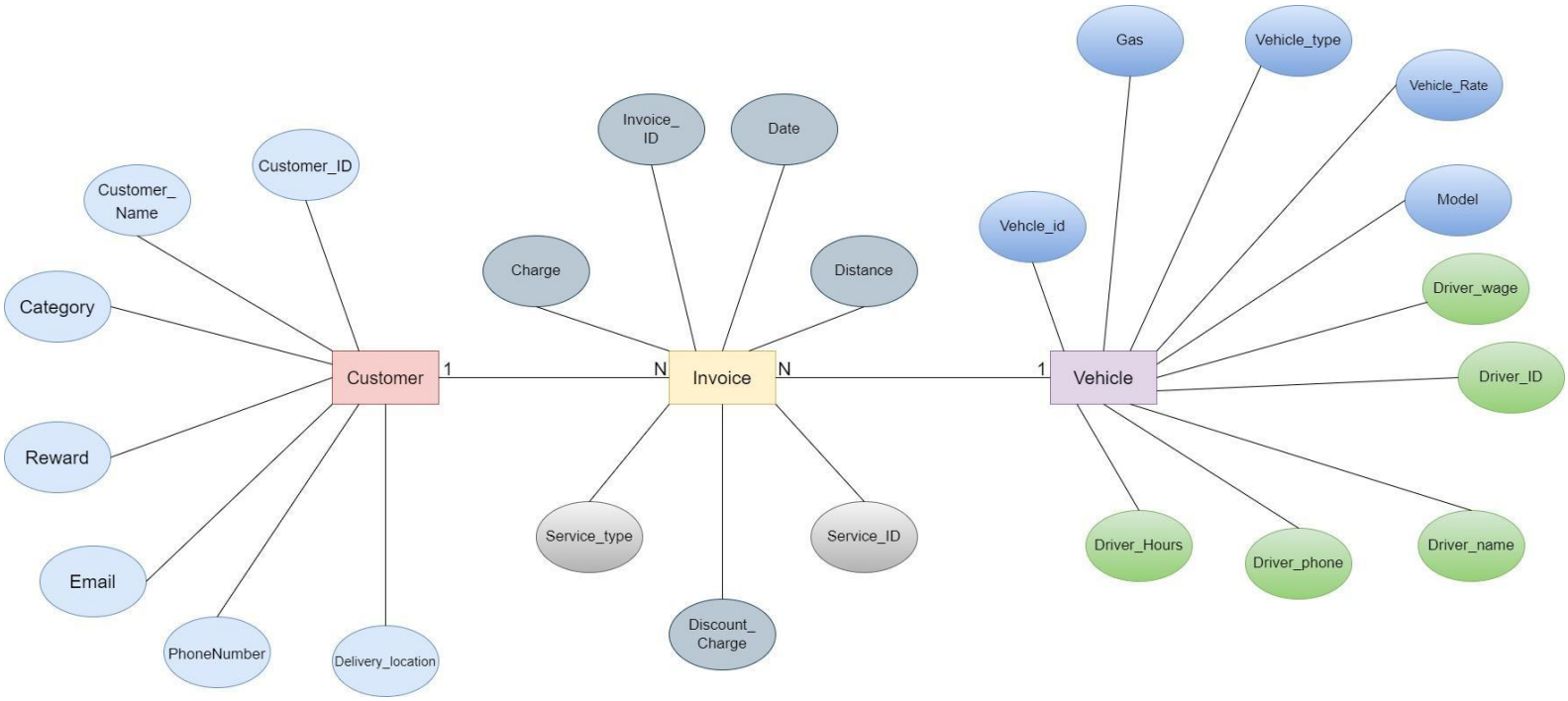
received, either one of these vehicles can be used to deliver the item or person based on customer wants.

The cost of a bike and a scooter is 21 rupees per kilometer, a car costs 26 rupees per kilometer, a bicycle costs 26 rupees per kilometer and a tuck costs 26 rupees per kilometer. Cars, bikes, and scooters are used to deliver people while all the vehicles mentioned are used to deliver items like food or object.

One driver may drive many vehicles but should complete a ride using one vehicle. On the next ride, the driver may use a different vehicle as per need. A driver may work part-time or full-time. Full-time drivers work 40 hours per week while part-time drivers work 20 hours per week. Drivers get paid hourly. Per hour the driver who drives a car gets 200 rupees. Drivers who drive bikes, scooters, and bicycles get 175 per hour. Drivers who drive trucks make more than drivers who drive cars, scooters, and bikes. They get 250 rupees per hour. Wage is calculated per month.

There are 2 categories of customers, Staff, and normal. The staff gets a 20% discount and normal customers are asked for a normal rate.

Initial ERD



The figure above is a rough diagram of an ERD for the service. Here three Entities are present, Customer, Invoice, and Vehicle. The attributes are also involved alongside its entity.

An entity has been constantly described as a real-world object which gets stored in the database as tables. Each of these tables has properties that describe this real-world object which are called attributes.

To convert this diagram into a structured database system, a step called Normalization must be carried out.

Normalization

Normalization is the technique of reducing data redundancy and removing unwanted characteristics like insertion, Update, and Deletion Anomalies. This process is done to organize data in a database. A huge table is divided into smaller ones during the normalization steps and these tables are linked through relationships. The well-known advantage of normalization is to remove anomalies producing a well-organized, flexible, consistent database design. As mentioned before, there are three anomalies, Insertion (difficulty inserting a new tuple into a relationship because of lack of data), Deletion (deletion of important data after removing some data), Updatation (multiple rows of data required to update just to update single data).

There are forms of normalization that must be present to complete the process of normalization. The first form is called UNF, the un-normalized form, and is the simplest database model. In this form, all attributes are gathered and listed. There are two groups on this form, Repetitive group, and repetitive data. The second form is called 1NF which eliminates repetitive groups. 2NF is the second form and it eliminates partial functional dependencies (part of a key giving a value of non-key). 3NF is third and this eliminates transitive dependencies (part of a key gives values to a non-key and this non-key value gives value to another non-key). 4NF and 5NF eliminate multi-value dependencies and join dependencies respectively.

UNF

In this form, the repetitive data, also known as Customers, and the repetitive group, also known as Invoice and Vehicle, are separated. (Customer {Invoice, Vehicle}). These are the two groups.

Customer – Customer_ID, Customer_name, Customer_phonenumber, Reward, Customer_email, Category, Delivery_location {Invoice_ID, Date, Charger, Distance, Discount_charge, Service_ID, Service_type, Vehicle_ID, Vehicle_name, Gas, Vehicle_type, Model, Rate, Driver_ID, Driver_name, Hours, Driver_phonenumber, Driver_wage}

Here the primary key is Customer_ID.

1NF

In this form, the two groups are separated and written in two tables.

Customer – Customer_ID, Customer_name, Customer_phonenumber, Reward, Customer_email, Category, Delivery_location

Here the primary key is Customer_ID.

Customer-Invoice_Driver – Invoice_ID, Customer_ID, Date, Charger, Distance, Discount_charge, Service_ID, Service_type, Vehicle_ID, Gas, Vehicle_type, Model, Vehicle_Rate, Driver_ID, Driver_name, Hours, Driver_phonenumber, Driver_wage

Here the two keys are Invoice_ID and Customer_ID.

These are the two tables formed in 1NF.

2NF

In this form, partial dependencies are located and removed from tables. If a part of the key gives a non-key value, this is called partial dependency. But if there is only one key in a table then that table automatically goes to the next form.

To locate partial dependencies, we must take two keys from tables, and since the customer table has only 1 it automatically goes to the next form.

Taking two keys in the customer-invoice table, we get

Invoice_ID **Invoice_ID**, Date, Charger, Distance, Discount_charge, Service_ID, Service_type, Vehicle_ID, Gas, Vehicle_type, Model, Vehicle_rate, Driver_ID, Driver_name, Hours, Driver_phonenumber, Driver_wage

Customer_ID XXX

Invoice_ID, Customer_ID

Invoice_ID gives the values shown above. Customer_ID gives nothing while Invoice_ID and Customer_ID are their tables. Hence in this form, we have 3 tables:

Customer – Customer_ID, Customer_name, Customer_phonenumber, Reward, Customer_email, Category, Delivery_location

Here the primary key is Customer_ID.

Invoice – Invoice_ID, Date, Charger, Distance, Discount_charge, Service_ID, Service_type, Vehicle_ID, Gas, Vehicle_type, Model, Vehicle_rate, Driver_ID, Driver_name, Hours, Driver_phonenumber, Driver_wage

Here the key is Invoice_ID.

Invoice-Customer – Invoice_ID, Customer_ID

Here the keys are Invoice_ID and Customer_ID

Therefore, we have 3 tables in 2NF.

3NF

In this form, the goal is to remove any transitive dependencies. When a part of a key gives value to a non-key and this non-key gives value to another non-key then this is called transitive dependency. The only table that will have transitive dependencies is the invoice table. The rest two get selected as the final table.

Taking the invoice table, we get:

Invoice – Invoice_ID, Date, Charger, Distance, Discount_charge, Service_ID, Service_type, Vehicle_ID, Gas, Vehicle_type, Model, Vehicle_rate, Driver_ID, Driver_name, Hours, Driver_phonenumber, Driver_wage

Service table: Invoice_ID(FK) □ Service_ID (PK) □ Service_type

Invoice_ID □ Vehicle_ID □ Vehicle_type □ Model □ Vehicle_rate □ Gas □ Driver_ID □ Driver_name □ Hours □ Driver_phonenumber □ Driver_wage

Vehicle table: Vehicle_ID (PK) □ Vehicle_type □ Vehicle_rate □ Model □ Driver_ID (FK) □ Gas

Driver table: Driver_ID (PK) □ Driver_name □ Hours □ Driver_phonenumber □ Driver_wage

Invoice table: - Invoice_ID (PK) □ Service_ID (FK) □ Vehicle_ID (FK) □ Distance □ Charge □ Discount_charge

Therefore, the tables created after this form are:

Customer table – Customer_ID, Customer_name, Customer_phonenumber, Reward, Customer_email, Category, Delivery_location

Invoice_Date table – Date_ID, Customer_ID (FK), Invoice_ID (FK), Date

Invoice table – Invoice_ID, Service_ID (FK), Vehicle_ID (FK), Customer_ID (FK), Distance, Charge, Discount_charge

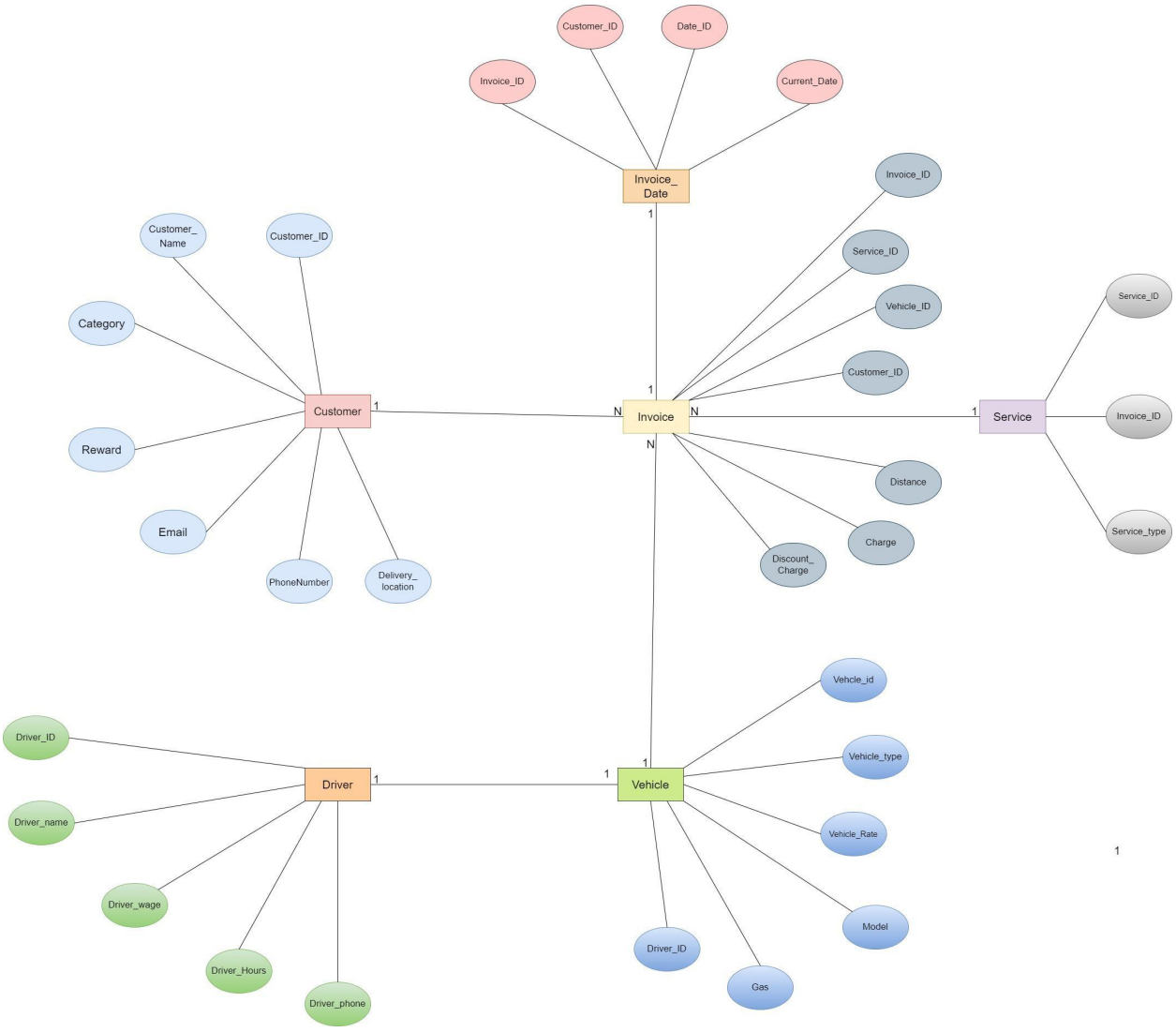
Service table – Service_ID, Invoice_ID(FK), Service_type

Vehicle table – Vehicle_ID, Vehicle_type, Vehicle_rate, Model, Driver_ID(FK), Gas

Driver table – Driver_ID, Driver_name, Hours, Driver_phonenumber, Driver_wage

After this form, we have 6 tables.

Final ERD



The final form of the structured database system of this service has been achieved. The Figure above shows the ERD diagram of this finished system.

Implementation

After the finished result, the components of it should be entered into a database management system. This will give structure to the service which helps update, edit, and maintain the service. Oracle is a database management system that uses SQL commands to carry out its work. Using this system creating tables and giving values to the attributes should be an easy task.

According to normalization, the following tables have been created:

Customer

Table 1. Customer table

Attributes	Data Type	Constraints	Description
Customer_ID	varchar2(7)	Primary key, Unique	This portion holds a unique id given to a customer.
Customer_name	varchhar2(25)	NOT NULL	This portion holds the customer's name.
Category	varchar2(10)	NOT NULL	This portion tells of whether a customer is a staff or just a regular customer.
Reward	Int		This portion holds the number of points a customer has.
Email	varchar2(25)		This portion holds the email address of the customer.
PhoneNumber	varchar2(15)	NOT NULL	This portion holds the phone number of the customer.
Delivery_location	varchar2(50)	NOT NULL	This portion holds the address given by the customer.

The customer is the clients who run this business. They are the main glue to the service, so their details will be saved in a table called Customer.

```
SQL> select * from customer;

CUSTOME CUSTOMER_NAME             CATEGORY       REWARD EMAIL                     PHONENUMBER     DELIVERY_LOCATION
------- ------------------------- ---------- ---------- ------------------------- --------------- ------------------------------------------------
C01     Sagar Thapa               Staff            2500 sagar_thapa1@gmail        9768345621      Thamel, Kathmandu
C02     Susil Magar               Regular           400 susil_magar12@yahoo       9456712323      Dhumbarai, Kathmandu
C03     Shreya Shrestha           Regular          3400 shreya_shrestha34@yahoo   9857716353      Dhapasi, Kathmandu
C04     Kapal Chettri             Regular          2200 kapal_chettri35@gmail     9756786751      New Baneshwor, Kathmandu
C05     Monica Acharaya           Staff              50 monica_acharaya06@gmail   9759364718      Putalisadak, Kathmandu
C06     Dhiraz Adhikari           Regular          1000 dhiraz_adhikari16@gmail   9857162738      Chabel, Kathmandu
C07     Samant Giri               Regular          5000 samant_giri77@gmail       9847656732      Gausala, Kathmandu

7 rows selected.
```

*Figure SEQ Figure * ARABIC 4. Customer table values*

```
CUSTOMER_NAME                             NOT NULL VARCHAR2(25)
CATEGORY                                  NOT NULL VARCHAR2(10)
REWARD                                             NUMBER(38)
EMAIL                                              VARCHAR2(25)
PHONENUMBER                               NOT NULL VARCHAR2(15)
DELIVERY_LOCATION                         NOT NULL VARCHAR2(50)
```

*Figure SEQ Figure * ARABIC 3. Customer table creation*

These are the values for the customer table.

Driver

Table 2. Driver table

Attribute	Data Type	Constraints	Description
Driver_ID	varchar2(7)	Primary Key, Unique	This portion holds a unique id given to a driver.
Driver_name	varchar2(25)	NOT NULL	This portion holds the driver's name.
Driver_wage	Int	NOT NULL	This portion holds the amount of wage the driver has.

Driver_hours	varchar2(10)	NOT NULL	This portion tells if the driver is full-time or part-time.
Driver_phone	varchar2(15)	NOT NULL	This portion holds the driver's phone number.

The table above is for the driver table with its data type, constraints, and description.

```
SQL*Plus: Release 11.2.0.2.0 Production on Sat Dec 31 16:25:59 2022

Copyright (c) 1982, 2010, Oracle.  All rights reserved.

SQL> conn cw_NahushmanSinghkarki
Enter password:
Connected.
SQL> create table driver (
  2  Driver_ID varchar2(7) Primary key,
  3  Driver_name varchar2(25) NOT NULL,
  4  Driver_wage int NOT NULL,
  5  Driver_hours varchar2(10) NOT NULL,
  6  Driver_phone varchar2(15) NOT NULL);

Table created.

SQL> desc driver;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 DRIVER_ID                                 NOT NULL VARCHAR2(7)
 DRIVER_NAME                               NOT NULL VARCHAR2(25)
 DRIVER_WAGE                               NOT NULL NUMBER(38)
 DRIVER_HOURS                              NOT NULL VARCHAR2(10)
 DRIVER_PHONE                              NOT NULL VARCHAR2(15)

SQL>
```

The figure above shows the creation of the driver table.

```
SQL> insert all into driver values ('D01', 'Sishir Thapa', 40000, 'Full-time', '9456872341')
  2  into driver values ('D02', 'Sarthak rai', 32000, 'Full-time', '9849864581')
  3  into driver values('D03', 'Simant adhikari', 28000, 'Full-time', '9768593418')
  4  into driver values('D04', 'Bir sapkota', 20000, 'Part-time', '9867341652')
  5  into driver values('D05', 'Silesh thakuri', 16000, 'Part-time', '9705489235')
  6  into driver values('D06' , 'Dipesh magar', 14000, 'Part-time', '9853127469')
  7  into driver values('D07', 'Sailesh basnet', 20000, 'Part-time', '9714263845')
  8  select * from dual;

7 rows created.
```

The data above shows the values inserted inside the driver table.

```
SQL> update driver
  2  set Driver_wage = 14000 where Driver_ID = 'D07';

1 row updated.
```

Here, a few values have been edited.

```
SQL> select * from driver;

DRIVER_ DRIVER_NAME               DRIVER_WAGE DRIVER_HOU DRIVER_PHONE
------- ------------------------- ----------- ---------- ---------------
D01     Sishir Thapa                    40000 Full-time  9456872341
D02     Sarthak rai                     32000 Full-time  9849864581
D03     Simant adhikari                 28000 Full-time  9768593418
D04     Bir sapkota                     20000 Part-time  9867341652
D05     Silesh thakuri                  16000 Part-time  9705489235
D06     Dipesh magar                    14000 Part-time  9853127469
D07     Sailesh basnet                  14000 Part-time  9714263845
```

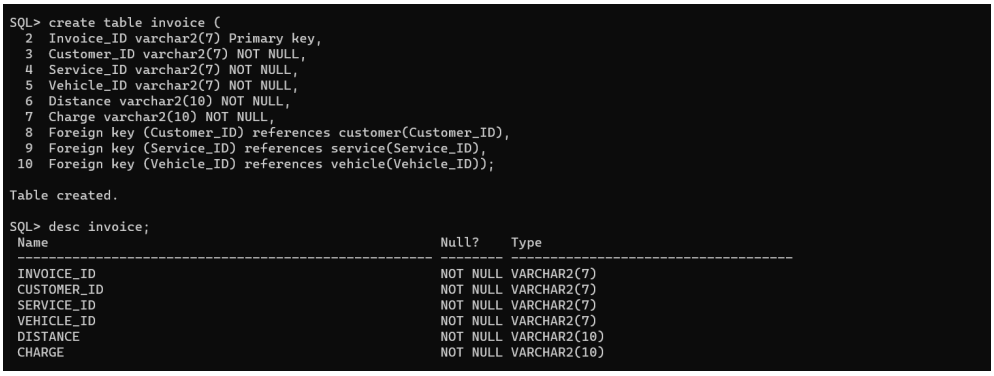
This is what the final driver table looks like.

Invoice

Table 3. Invoice Table

Attribute	Data Type	Constraints	Description
Invoice_ID	varchar2(7)	Primary key, Unique	This portion holds a unique id given to an invoice.
Customer_ID	varchar2(7)	Foreign Key, NOT NULL	This portion holds the customer id and acts as a foreign key which is taken from the customer table.
Service_ID	varchar2(7)	Foreign Key, NOT NULL	This portion holds the service id and acts as a foreign key which is taken from the service table.
Vehicle_ID	varchar2(7)	Foreign Key, NOT NULL	This portion holds the vehicle id and acts as a foreign key which is taken from the vehicle table.
Distance	varchar2(10)	NOT NULL	This portion holds the distance it took to complete the service.

Charge	varchar2(10)	NOT NULL	This portion holds the total amount that needs to be paid by the customer for the service.
Discount_charge	varchar2(10)	NULL	This portion holds the total charge after applying the discount.



```
SQL> create table invoice (
  2  Invoice_ID varchar2(7) Primary key,
  3  Customer_ID varchar2(7) NOT NULL,
  4  Service_ID varchar2(7) NOT NULL,
  5  Vehicle_ID varchar2(7) NOT NULL,
  6  Distance varchar2(10) NOT NULL,
  7  Charge varchar2(10) NOT NULL,
  8  Foreign key (Customer_ID) references customer(Customer_ID),
  9  Foreign key (Service_ID) references service(Service_ID),
 10  Foreign key (Vehicle_ID) references vehicle(Vehicle_ID));

Table created.

SQL> desc invoice;
 Name                                      Null?    Type
 ----------------------------------------- -------- ------------------------------------
 INVOICE_ID                                NOT NULL VARCHAR2(7)
 CUSTOMER_ID                               NOT NULL VARCHAR2(7)
 SERVICE_ID                                NOT NULL VARCHAR2(7)
 VEHICLE_ID                                NOT NULL VARCHAR2(7)
 DISTANCE                                  NOT NULL VARCHAR2(10)
 CHARGE                                    NOT NULL VARCHAR2(10)
```

The figure above shows the creation of the invoice table. Since the foreign key was set before inserting values. The not null constraint was removed, and the foreign key column was left empty. Since service_ID and vehicle_ID is a foreign key in this table, the service and vehicle table were given values first then the related values for service_id and vehicle_id were given to this table.

```
SQL> alter table invoice
  2  add discount_charge varchar2(10);

Table altered.

SQL> select * from invoice;

INVOICE CUSTOME SERVICE DISTANCE   CHARGE     VEHICLE DISCOUNT_C
------- ------- ------- ---------- ---------- ------- ----------
I01     C04     S01     4 km       Rs 104     V04
I02     C02     S02     7 km       Rs 182     V06
I03     C06     S03     8 km       Rs 208     V05
I04     C03     S04     11 km      Rs 286     V07
I05     C01     S05     6 km       Rs 126     V02
I06     C07     S06     5 km       Rs 130     V03
I07     C05     S07     9 km       Rs 189     V01

7 rows selected.

SQL> update invoice
  2  set discount_charge = 'Rs 100' where Customer_id = 'C01';

1 row updated.

SQL> update invoice
  2  set discount_charge = 'Rs 150' where Customer_id = 'C05';

1 row updated.
```

Discount_charge has been added to the table to display the bill with a discount for the staff members.

After inserting the value for discount_charge, the following table gets generated.

```
SQL> select * from invoice;

INVOICE CUSTOME SERVICE DISTANCE   CHARGE     VEHICLE DISCOUNT_C
------- ------- ------- ---------- ---------- ------- ----------
I01     C04     S01     4 km       Rs 104     V04
I02     C02     S02     7 km       Rs 182     V06
I03     C06     S03     8 km       Rs 208     V05
I04     C03     S04     11 km      Rs 286     V07
I05     C01     S05     6 km       Rs 126     V02     Rs 100
I06     C07     S06     5 km       Rs 130     V03
I07     C05     S07     9 km       Rs 189     V01     Rs 150

7 rows selected.
```

Vehicle

Table 4. Vehicle Table

Attribute	Data Type	Constraints	Description

Vehicle_ID	varchar2(7)	Primary Key, Unique	This portion holds a unique id given to a vehicle.
Vehicle_type	varchar2(10)	NOT NULL	This portion holds the type of vehicle.
Model	varchar2(25)	NOT NULL	This portion holds the model of the vehicle.
Vehicle_rate	varchar2(10)	NOT NULL	This portion holds the cost of the vehicle per kilometer.
Gas	varchar2(10)	NOT NULL	This portion holds the type of gas the engine uses.
Driver_ID	varchar2(7)	Foreign Key, NOT NULL	This portion holds the driver's id and acts as a foreign key taken from the driver's table.

```
SQL> create table vehicle (
  2  Vehicle_ID varchar2(7) Primary key,
  3  Vehicle_type varchar2(10) NOT NULL,
  4  Model varchar2(25) NOT NULL,
  5  Vehicle_rate varchar2(10) NOT NULL,
  6  Gas varchar2(10) NOT NULL,
  7  Driver_ID varchar2(7) NOT NULL,
  8  Foreign key (Driver_ID) references driver(Driver_ID));

Table created.

SQL> desc vehicle;
 Name                                              Null?    Type
 ------------------------------------------------- -------- ------------------------------------
 VEHICLE_ID                                        NOT NULL VARCHAR2(7)
 VEHICLE_TYPE                                      NOT NULL VARCHAR2(10)
 MODEL                                             NOT NULL VARCHAR2(25)
 VEHICLE_RATE                                      NOT NULL VARCHAR2(10)
 GAS                                               NOT NULL VARCHAR2(10)
 DRIVER_ID                                         NOT NULL VARCHAR2(7)
```

The figure above shows the creation of a vehicle table.

```
SQL> insert all into vehicle values ('V01', 'Bike', 'Yamaha Saluto UBS', 'Rs 21/km', 'Petrol', 'D03')
  2  into vehicle values ('V02', 'Scooter', 'Honda Activa 6G', 'Rs 21/km', 'Petrol', 'D06')
  3  into vehicle values ('V03', 'Bicycle', 'Benz Mtb Carbon', 'Rs 26/km', 'N/A', 'D07')
  4  into vehicle values ('V04', 'Truck', 'Maruti Suzuki EECO Cargo', 'Rs 26/km', 'Disel' , 'D01')
  5  into vehicle values ('V05', 'Truck', 'Maruti Suzuki EECO Cargo', 'Rs 26/km', 'Disel' , 'D04')
  6  into vehicle values ('V06', 'Car', 'Datsun GO', 'Rs 26/km', 'Disel' , 'D02')
  7  into vehicle values ('V07', 'Car', 'Nissan Magnite', 'Rs 26/km', 'Disel' , 'D05')
  8  select * from dual;

7 rows created.

SQL> select * from vehicle;

VEHICLE VEHICLE_TY MODEL                    VEHICLE_RA GAS        DRIVER_
------- ---------- ------------------------ ---------- ---------- -------
V01     Bike       Yamaha Saluto UBS        Rs 21/km   Petrol     D03
V02     Scooter    Honda Activa 6G          Rs 21/km   Petrol     D06
V03     Bicycle    Benz Mtb Carbon          Rs 26/km   N/A        D07
V04     Truck      Maruti Suzuki EECO Cargo Rs 26/km   Disel      D01
V05     Truck      Maruti Suzuki EECO Cargo Rs 26/km   Disel      D04
V06     Car        Datsun GO                Rs 26/km   Disel      D02
V07     Car        Nissan Magnite           Rs 26/km   Disel      D05

7 rows selected.
```

Figure 13. Value insertion

The values have been inserted into the table.

Service

Table 5. Service Table

Attribute	Data Type	Constraints	Description
Service_ID	varchar2(7)	Primary Key, Unique	This portion holds a unique id given to a vehicle.
Invoice_ID	varchar2(7)	Foreign Key, NOT NULL	This portion holds the invoice id and acts as a foreign key which is taken from the invoice table.
Service_Type	varchar2(10)	NOT NULL	This portion holds the type of service the customer chose.

Details on the service table are provided in the above table.

```
SQL> create table service (
  2  Service_ID varchar2(7) Primary key,
  3  Invoice_ID varchar2(7) NOT NULL,
  4  Service_type varchar2(10) NOT NULL);

Table created.
```

```
SQL> set linesize 100;
SQL> desc service;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 SERVICE_ID                                NOT NULL VARCHAR2(7)
 INVOICE_ID                                NOT NULL VARCHAR2(7)
 SERVICE_TYPE                              NOT NULL VARCHAR2(10)
```

```
SQL> select * from service;

SERVICE INVOICE SERVICE_TY
------- ------- ----------
S01     I01     Item
S02     I02     Person
S03     I03     Item
S04     I04     Person
S05     I05     Food
S06     I06     Food
S07     I07     Food

7 rows selected.
```

Invoice_Date

Table 6. Date Table

Attribute	Data Type	Constraints	Description
Date_ID	varchar2(7)	Primary key, Unique	This portion holds the unique id given to a particular service on a certain date.
Customer_ID	varchar2(7)	Foreign Key, NOT NULL	This portion holds the customer id and acts as a foreign key which is taken from the customer table.
Invoice_ID	varchar2(7)	Foreign Key, NOT NULL	This portion holds the invoice id and acts as a foreign key which is taken from the invoice table.
Date	date	NOT NULL	This portion holds the date.

The above tables have the details for the invoice_date table.

```
SQL> create table Invoice_date (
  2  Date_ID varchar2(7) Primary key,
  3  Customer_ID varchar2(7) NOT NULL,
  4  Invoice_ID varchar2(7) NOT NULL,
  5  Current_Date varchar2(10) NOT NULL,
  6  Foreign key (Customer_ID) references customer(Customer_ID));

Table created.

SQL> desc Invoice_date;
 Name                                              Null?    Type
 ------------------------------------------------- -------- ------------------------------------
 DATE_ID                                           NOT NULL VARCHAR2(7)
 CUSTOMER_ID                                       NOT NULL VARCHAR2(7)
 INVOICE_ID                                        NOT NULL VARCHAR2(7)
 CURRENT_DATE                                      NOT NULL VARCHAR2(10)
```

The figure above shows the creation of the invoice_date table.

```
SQL> update invoice_date
  2  set current_date = to_date('2021/12/12', 'yyyy/mm/dd') where date_id = 'Dt01';

1 row updated.

SQL> update invoice_date
  2  set current_date = to_date('2022/01/04', 'yyyy/mm/dd') where date_id = 'Dt02';

1 row updated.

SQL> update invoice_date
  2  set current_date = to_date('2022/03/15', 'yyyy/mm/dd') where date_id = 'Dt03';

1 row updated.

SQL> update invoice_date
  2  set current_date = to_date('2022/04/01', 'yyyy/mm/dd') where date_id = 'Dt04';

1 row updated.

SQL> update invoice_date
  2  set current_date = to_date('2022/05/22', 'yyyy/mm/dd') where date_id = 'Dt05';

1 row updated.

SQL> update invoice_date
  2  set current_date = to_date('2022/06/03', 'yyyy/mm/dd') where date_id = 'Dt06';

1 row updated.

SQL> update invoice_date
  2  set current_date = to_date('2022/07/14', 'yyyy/mm/dd') where date_id = 'Dt07';

1 row updated.
```

The values for the invoice_date table have been successfully inserted.

```
SQL> select * from invoice_date;

DATE_ID CUSTOME INVOICE CURRENT_D
------- ------- ------- ---------
Dt01    C04     I01     12-DEC-21
Dt02    C02     I02     04-JAN-22
Dt03    C06     I03     15-MAR-22
Dt04    C03     I04     01-APR-22
Dt05    C01     I05     22-MAY-22
Dt06    C07     I06     03-JUN-22
Dt07    C05     I07     14-JUL-22

7 rows selected.
```

This is the final invoice_date table.

Database Querying

To detect if the final system is working perfectly, applying queries will be helpful. Some commands generate needed values and data. The query required by this project are below:

List of customers based on category.

```
SQL> select * from customer order by category;

CUSTOME CUSTOMER_NAME             CATEGORY       REWARD EMAIL                     PHONENUMBER     DELIVERY_LOCATION
------- ------------------------- ---------- ---------- ------------------------- --------------- --------------------------------------------------
C04     Kapal Chettri             Regular          2200 kapal_chettri35@gmail     9756786751      New Baneshwor, Kathmandu
C06     Dhiraz Adhikari           Regular          1000 dhiraz_adhikari16@gmail   9857162738      Chabel, Kathmandu
C02     Susil Magar               Regular           400 susil_magar12@yahoo       9456712323      Dhumbarai, Kathmandu
C03     Shreya Shrestha           Regular          3400 shreya_shrestha34@yahoo   9857716353      Dhapasi, Kathmandu
C07     Samant Giri               Regular          5000 samant_giri77@gmail       9847656732      Gausala, Kathmandu
C05     Monica Acharaya           Staff              50 monica_acharaya06@gmail   9759364718      Putalisadak, Kathmandu
C01     Sagar Thapa               Staff            2500 sagar_thapa1@gmail        9768345621      Thamel, Kathmandu

7 rows selected.
```

There are two categories of customers. Regular and staff. The above diagram shows the list of customers according to the category.

```
SQL> select model, vehicle_type, vehicle_rate from vehicle order by vehicle_rate desc;

MODEL                     VEHICLE_TY VEHICLE_RA
------------------------- ---------- ----------
Maruti Suzuki EECO Cargo  Truck      Rs 26/km
Datsun GO                 Car        Rs 26/km
Benz Mtb Carbon           Bicycle    Rs 26/km
Nissan Magnite            Car        Rs 26/km
Maruti Suzuki EECO Cargo  Truck      Rs 26/km
Honda Activa 6G           Scooter    Rs 21/km
Yamaha Saluto UBS         Bike       Rs 21/km

7 rows selected.
```

The figure above shows the model and type of vehicle based on the vehicle rate in descending order.

```
SQL> select count(*) Petrol_Users from vehicle where gas = 'Petrol';

PETROL_USERS
------------
           2
```

The figure above shows the total number of vehicles that uses petrol.

```
SQL> select driver_name from driver where driver_name like '%s%';

DRIVER_NAME
-------------------------
Sishir Thapa
Bir sapkota
Silesh thakuri
Dipesh magar
Sailesh basnet
```

Figure 23. Driver name

To locate customers with s in between their names the command above should be entered. Since the name includes first and last name, any s inside the full name will be searched for.

Transactional Queries and Relational Algebra

Relational Algebra is a query language that explains the process of extracting a certain relation as a result. The process involves many operators to acquire the desired result.

σ = SELECT

This operator acts as the select command.

R = relation

π = projection

This operator is used to display.

⋈ = join

This operator joins two tables.

```
SQL> select c.customer_id, c.customer_name, c.category, c.reward, c.email, c.phonenumber, c.delivery_location from customer c join invoice i on i.customer_i
d = c.customer_id join service s on i.invoice_id = s.invoice_id where service_type = 'Item';

CUSTOME CUSTOMER_NAME             CATEGORY       REWARD EMAIL                     PHONENUMBER     DELIVERY_LOCATION
------- ------------------------- ---------- ---------- ------------------------- --------------- --------------------------------------------------
C04     Kapal Chettri             Regular          2200 kapal_chettri35@gmail     9756786751      New Baneshwor, Kathmandu
C06     Dhiraz Adhikari           Regular          1000 dhiraz_adhikari16@gmail   9857162738      Chabel, Kathmandu
```

The query above shows the details of customers who only used courier service (item).

The relational algebra for this query is as follows,

R1 : $\sigma_{\text{service\_type = 'Item'}} \bowtie$ (Service)

R2: R1 $_{\text{R1.invoice\_id = Invoice.invoice\_id}} \bowtie$ (Invoice)

$_{\text{Invoice.customer\_id = Customer.customer\_id}} \bowtie$ (Customer)

R3: π customer_id, customer_name, category, reward, email, phonenumber, delivery_location(R3)

```
SQL> select * from driver where rownum < 4 order by driver_wage desc;

DRIVER_ DRIVER_NAME               DRIVER_WAGE DRIVER_HOU DRIVER_PHONE
------- ------------------------- ----------- ---------- ---------------
D01     Sishir Thapa                    40000 Full-time  9456872341
D02     Sarthak rai                     32000 Full-time  9849864581
D03     Simant adhikari                 28000 Full-time  9768593418
```

Figure 25. Drivers wage

The figure above displays the top 3 highest earning driver. There is no relation algebra solution for order by.

Critical Evaluation

Overall, the service was created and implemented successfully. The main purpose of this service was to deliver the required person or item to the designated location. To achieve the result all, the required entities, attributes, and values were to be structured and organized in a database system properly. After finishing normalization some part of the result was achieved. The tables were created with the help of normalization was another part of the result. After inserting the tables in the database system values were inserted inside the table. Using these values relations were appointed between tables. To make sure the system would work, queries were used to verify the effectiveness of the service.

All these were the requirement for the service and the project. Completing every task was useful because it taught more concepts about the process of normalization, table creation, data insertion, and SQL queries. These concepts were a heavy topic; still, a lot more is left to learn about them. But this project provided enough knowledge to grab the required and basic idea.

While creating the service many problems were stumbled upon. Normalization was one of the hardest parts of this project. To finish this process some time with teachers and friends was spent. Business rules worked side by side with normalization because the attributes were involved in both processes. Creating tables were supposed to be easy but small mistakes were found like spelling error, wrong constraint, inserting dates, and assigning foreign key before inserting values. Because of these minor errors, table creation took longer than it was supposed to. Queries were another hard topic that needed a lot of research and help from the internet and friends. After spending a fair amount of time, the answers that were understood was recorded. Since the difficulty level was very hard few queries are unanswered.

In conclusion, the effort did not go to waste and this project was a success.

Reference

Admin. “Bikes Price in Nepal (2022 Updated).” *NepalETrend*, 13 June 2022, https://www.nepaletrend.com/bikes-price-in-nepal/.

https://www.nepaletrend.com/bikes-price-in-nepal/

“DBMS Normalization: 1NF, 2NF, 3NF and BCNF with Examples - Javatpoint.” *Www.javatpoint.com*, https://www.javatpoint.com/dbms-normalization.

https://www.javatpoint.com/dbms-normalization

NewsTeam. “Maruti Suzuki Eeco Cargo Van Launched in Nepali Market.” *ktm2day.Com*, 14 Sept. 2020, https://www.ktm2day.com/maruti-suzuki-eeco-price-nepal/amp/.

https://www.ktm2day.com/maruti-suzuki-eeco-price-nepal/amp/

“Cars Price in Nepal - 2022: Car Price List in Nepal: Cars in Nepal (2022).” *Car Nepal*, 23 Apr. 2022, https://carnepal.com/car-price-in-nepal/.

https://carnepal.com/car-price-in-nepal/