DBMS Project Report

PES University

Database Management Systems

UE18CS252

Submitted By

PES2201800116 Aniketh D Urs

ABSTRACT:

The main aim of the Car Rental DBMS project is to keep track of the day-to-day activities of a car rental organization. The Car Rental Database revolves around three main entities Car, User and Reservation. Car can be reserved from a rental location with a specific address. A Particular can either be ‘Economy’, ‘Standard’, ’SUV’, ‘Premium’ ‘Minivan’ or ‘Electric'. Car Type decides the rental price per day. A user can take Insurance per day for the rental car. Car type and Insurance Type decides the Insurance price per day. A user can reserve a car for a number of days. He can use any valid promotional code which is maintained by status. When a user books a car he mentions the start date and end date for which he needs the car. The total amount and net amount are calculated based on start date, end date, actual end date(Hypothetical), rental price per day, insurance price per day, and promotional code. A user is categorized as guest and customer. User can continue reserving car as guest as long as he has not registered as customer. A user is uniquely identified by his/her Driving license number. A registered customer will be provided with a login id and password. A customer can save card details for future payment.

This Database Schema has been implemented using various DBMS functions and operations such as : Insert, Update, Delete, Join, Triggers and Transactions.

Transactions have been implemented for the PAYMENT relation wherein in case of any discrepancy if the payment doesn’t go through it can be rolled back as a result of which the tuple won’t be stored in the PAYMENT relation. If the payment is successful then the transaction can be committed and the tuple can be inserted in the payment relation.

INTRODUCTION

The Car Rental Database has 11 entities namely:

1. CAR: The Car relation contains description of various cars being offered by the rental organization. Some of the attributes include Registation no., Model, Year, Color etc.

2. CAR INSUARANCE: This relation gives you a detailed insight of the insurance types and prices being offered for various car types. Attributes of this relation include Car Type, Insurance Type and Insurance Price.

3. CAR TYPE: This relation gives the Prices offered per day for various car types. Attributes include Car_Type and Price Per Day.

4. CAR USER: This relation contains personal details of the user who has reserved a car. It contains Attributes such as Fname ,Email ,Phone ,DOB etc.

5. CARD DETAILS: This relation contains the card details of the users who have registered themselves as customers with the organization. It contains attributes such as Card_No, Name_On_Card, CVV etc.

6. INSUARANCE: Details the Type of insurance and their coverages. Included attributes are Insurance_Type, Bodily_Coverage and Medical_Coverage.

7. OFFER DETAILS: Describes the various offers/discounts being offered by the organization including the promotional code. Includes attributes like Promo_Code, Description, Discounted_Amount etc.

8. PAYMENT: Contains detailed information of the amount paid by a Guest or a Customer who has rented a car. Attributes include Payment_ID, Amount_Paid etc.

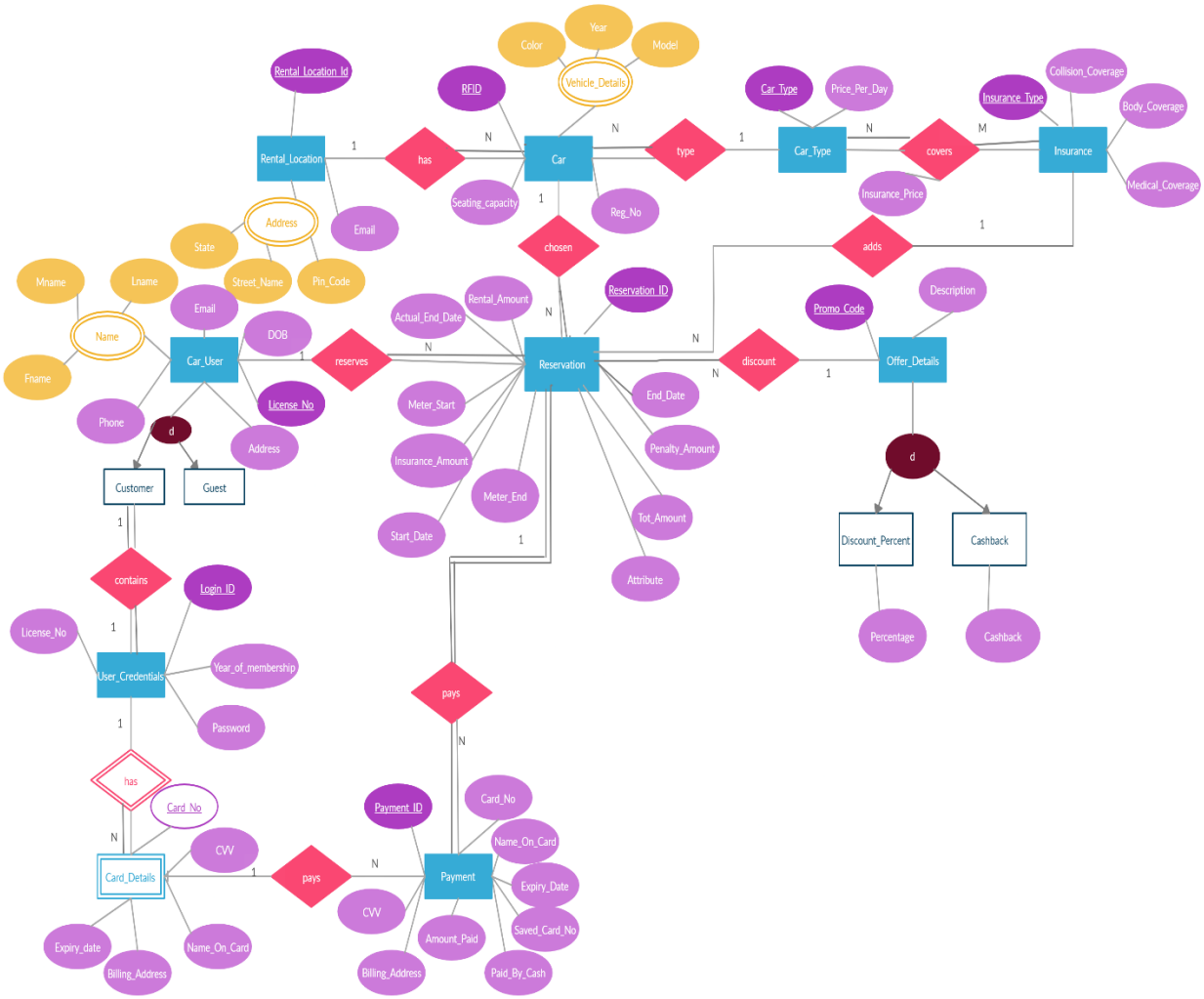
9. RENTAL_LOCATION: Contains information of the availability of the rental car including the state and street name. This contains Rental_Location_ID, Phone, Email, Street_Name, State, Pin_Code.

10. RESERVATION: This relation contains records of clients who have reserved a car and also generates the total amount to be paid by the person inclusive of [Insurance and Penalty Amount](if any) purchased. Attributes of the reservation relation include Reservation_ID, Start_Date,End_Date, Meter_Start, Meter_End ,Total_Amount, Additional_amount etc.

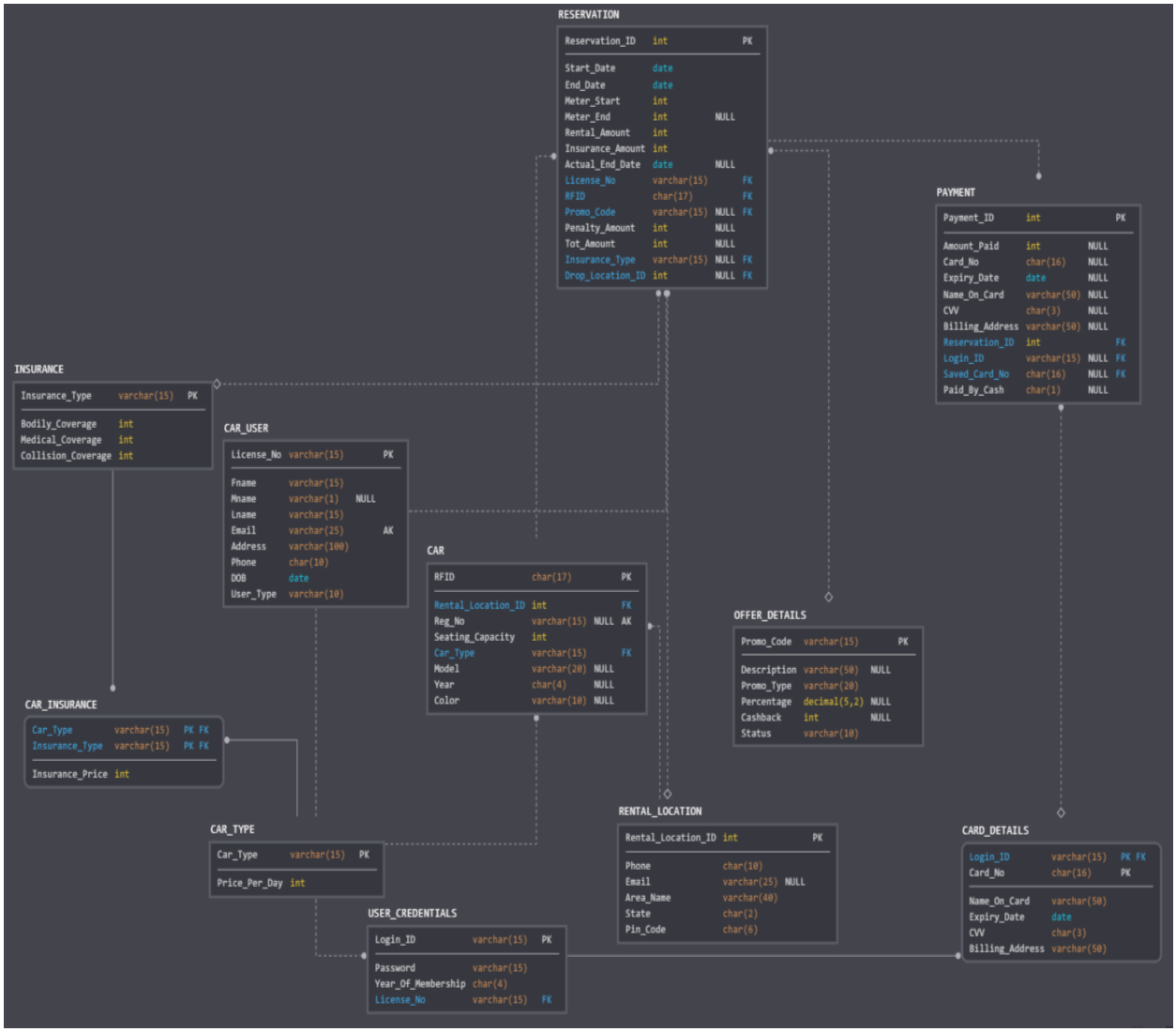
11. USER_CREDENTIALS: Contains information of the Users registered as customers with the car rental organization. Attributes include Login_ID, Password, Year_Of_Membership and Driving Licence No.

DATA MODEL

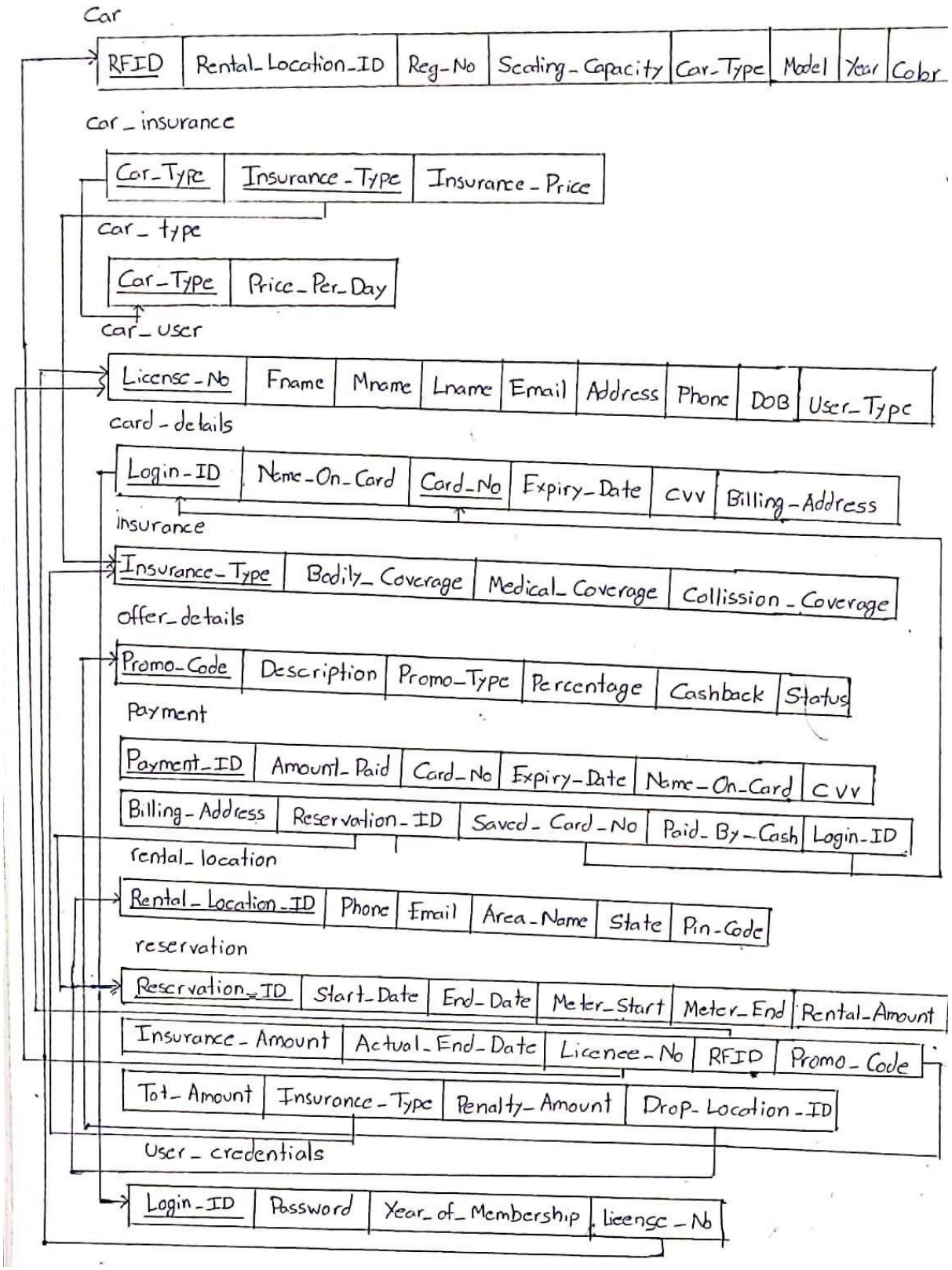
ER DIAGRAM:



RELATIONAL SCHEMA:



MAPPING:


Car
RFID | Rental_Location_ID | Reg_No | Seating_Capacity | Car_Type | Model | Year | Color
car_insurance
Car_Type | Insurance_Type | Insurance_Price
car_type
Car_Type | Price_Per_Day
car_user
License_No | Fname | Mname | Lname | Email | Address | Phone | DOB | User_Type
card_details
Login_ID | Name_On_Card | Card_No | Expiry_Date | CVV | Billing_Address
insurance
Insurance_Type | Bodily_Coverage | Medical_Coverage | Collission_Coverage
offer_details
Promo_Code | Description | Promo_Type | Percentage | Cashback | Status
payment
Payment_ID | Amount_Paid | Card_No | Expiry_Date | Name_On_Card | CVV
Billing_Address | Reservation_ID | Saved_Card_No | Paid_By_Cash | Login_ID
rental_location
Rental_Location_ID | Phone | Email | Area_Name | State | Pin_Code
reservation
Reservation_ID | Start_Date | End_Date | Meter_Start | Meter_End | Rental_Amount
Insurance_Amount | Actual_End_Date | Licence_No | RFID | Promo_Code
Tot_Amount | Insurance_Type | Penalty_Amount | Drop_Location_ID
User_credentials
Login_ID | Password | Year_of_Membership | license_No

ENTITY DESCRIPTION [Data Types, Keys]

1.RENTAL_LOCATION:

- Rental_Location_ID INT
- Phone CHAR(10)
- Email VARCHAR(25)
- Street_Name VARCHAR(40)
- State CHAR(2)
- Pin_Code CHAR(6)

Rental_Location_ID is chosen as the primary key for this relation.

2. CAR_TYPE:

- Car_Type VARCHAR(15)
- Price_Per_Day INT

Car_Type is chosen as the primary key for this relation

3. INSURANCE:

- Insurance_Type VARCHAR(15)
- Bodily_Coverage INT
- Medical_Coverage INT
- Collision_Coverage INT

Insurance_Type is the primary key for this relation.

4. CAR_INSURANCE:

- Car_Type VARCHAR(15)
- Insurance_Type VARCHAR(15)
- Insurance_Price INT

The primary key for this relation is (Car_Type,Insurance_Type).

The foreign key for this relation are:

1. (Car_Type) references (Car_Type) in CAR_TYPE relation
2. (Insurance_Type) references (Insurance_Type) in INSUARANCE relation

5. CAR_USER:

- License_No VARCHAR(15)
- Fname VARCHAR(15)
- Mname VARCHAR(1)
- Lname VARCHAR(15)
- Email VARCHAR(25)

- Address VARCHAR(100)
- Phone CHAR(10)
- DOB DATE
- User_Type VARCHAR(10)

Licence_No is the primary Key.

6. USER_CREDENTIALS:

- Login_ID VARCHAR(15)
- Password VARCHAR(15)
- Year_Of_Membership Char(4)
- License_No VARCHAR(15)

The primary key of this relation is Login_ID.

The foreign key of this relation is (Licence_No) references (Licence_No) in CAR_USER relation

7. CARD_DETAILS:

- Login_ID VARCHAR(15)
- Name_On_Card VARCHAR(50)
- Card_No CHAR(16)
- Expiry_Date DATE
- CVV CHAR(3)
- Billing_Address VARCHAR(50)

The primary Key for the relation is (Login_ID,Card_No).

The foreign key for this relation is (Login_ID) references (Login_ID) in USER_CREDENTIALS relation.

8. CAR:

- VIN CHAR(17)
- Rental_Location_ID INT
- Reg_No VARCHAR(15)
- Seating_Capacity INT
- Car_Type VARCHAR(15)
- Model VARCHAR(20)
- Year CHAR(4)
- Color VARCHAR(10)

The primary key for this relation is VIN.

The foreign keys for this relation are:

1. (Car_Type) references (Car_Type) in CAR_TYPE Relation.

2. (Rental_Location_ID) references (Rental_Location_ID) in RENTAL_LOCATION relation.

9. OFFER_DETAILS:

- Promo_Code VARCHAR(15)
- Description VARCHAR(50)
- Promo_Type VARCHAR(20)
- Percentage DECIMAL(5,2)
- Discounted_Amount INT
- Status VARCHAR(10)

Promo_Code is the primary key for this relation.

10. RESERVATION:

- Reservation_ID INT
- Start_Date DATE
- End_Date DATE
- Meter_Start INT
- Meter_End INT
- Rental_Amount INT
- Insurance_Amount INT
- Actual_End_Date DATE
- License_No VARCHAR(15)
- RFID CHAR(17)
- Promo_Code VARCHAR(15)
- Tot_Amount INT
- Insurance_Type VARCHAR(15)
- Penalty_Amount INT
- Drop_Location_ID INT

The primary key for this relation is Reservation_ID.

The foreign keys for this relation are:

1. (Drop_Location_ID) references (Rental_Location_ID) in RENTAL_LOCATION relation.
2. (License_No) references (License_No) in CAR_USER relation.
3. (Promo_Code) references (Promo_Code) in OFFER_DETAILS relation.
4. (Insurance_Type) references (Insurance_Type) in INSUARANCE relation.

11. PAYMENT:

- Payment_ID INT
- Amount_Paid INT
- Card_No CHAR(16)
- Expiry_Date DATE
- Name_On_Card VARCHAR(50)

- CVV CHAR(3)
- Billing_Address VARCHAR(50)
- Reservation_ID INT
- Login_ID VARCHAR(15)
- Saved_Card_No CHAR(16)
- Paid_By_Cash CHAR(1)

The primary key for this relation is Payment_ID.

The foreign keys for this relation are:

1. (Reservation_ID) references (Reservation_ID) in RESERVATION relation.
2. (Login_ID,Saved_Card_No) references (Login_ID,Card_No) in CARD_DETAILS relation.

DATA DEFINITION LANGUAGE:

```
CREATE TABLE RENTAL_LOCATION
(
  Rental_Location_ID INT PRIMARY KEY,
  Phone CHAR(10) NOT NULL,
  Email VARCHAR(25),
  Area_Name VARCHAR(40) NOT NULL,
  State CHAR(2) NOT NULL
  CHECK (State ='KA'),
  Pin_Code CHAR(6) NOT NULL
);

CREATE TABLE CAR_TYPE
(
  Car_Type VARCHAR(15) PRIMARY KEY,
  Price_Per_Day  INT NOT NULL
);
```

```
CREATE TABLE INSURANCE
(
  Insurance_Type VARCHAR(15) PRIMARY KEY,
  Bodily_Coverage  INT NOT NULL,
  Medical_Coverage  INT NOT NULL,
  Collision_Coverage  INT NOT NULL
);

CREATE TABLE CAR_INSURANCE
(
  Car_Type VARCHAR(15),
  Insurance_Type VARCHAR(15),
  Insurance_Price  INT NOT NULL,
  PRIMARY KEY(Car_Type,Insurance_Type),
  CONSTRAINT CARTYPEFK
  FOREIGN KEY (Car_Type) REFERENCES CAR_TYPE(Car_Type)
          ON DELETE CASCADE,
  CONSTRAINT INSURANCETYPEFK
  FOREIGN KEY (Insurance_Type) REFERENCES INSURANCE(Insurance_Type)
          ON DELETE CASCADE
);

CREATE TABLE CAR_USER
(
  License_No VARCHAR(15) PRIMARY KEY,
  Fname VARCHAR(15) NOT NULL,
  Mname VARCHAR(1),
  Lname VARCHAR(15) NOT NULL,
  Email VARCHAR(25) NOT NULL UNIQUE,
  Address VARCHAR(100) NOT NULL,
  Phone CHAR(10) NOT NULL,
```

```
  DOB DATE NOT NULL,
  User_Type VARCHAR(10) NOT NULL
);

CREATE TABLE USER_CREDENTIALS
(
  Login_ID VARCHAR(15) PRIMARY KEY,
  Password VARCHAR(15) NOT NULL,
  Year_Of_Membership Char(4) NOT NULL
  CHECK (Year_of_Membership>2000),
  License_No VARCHAR(15) NOT NULL,
  CONSTRAINT USRLIC
  FOREIGN KEY (License_No) REFERENCES CAR_USER(License_No)
          ON DELETE CASCADE
);

CREATE TABLE CARD_DETAILS
(
  Login_ID VARCHAR(15) NOT NULL,
  Name_On_Card VARCHAR(50) NOT NULL,
  Card_No CHAR(16) NOT NULL,
  Expiry_Date DATE NOT NULL,
  CVV CHAR(3) NOT NULL,
  Billing_Address VARCHAR(50) NOT NULL,
  PRIMARY KEY(Login_ID,Card_No),
  CONSTRAINT USRCARDFK
  FOREIGN KEY (Login_ID) REFERENCES USER_CREDENTIALS(Login_ID)
          ON DELETE CASCADE
);

CREATE TABLE CAR (
```

```
  RFID CHAR(17) PRIMARY KEY,
  Rental_Location_ID INT NOT NULL,
  Reg_No VARCHAR(15) UNIQUE,
  Seating_Capacity INT NOT NULL,
  Car_Type VARCHAR(15) NOT NULL,
  Model VARCHAR(20),
  Year CHAR(4),
  Color VARCHAR(10),
  CONSTRAINT CARRFIDTYPEFK
  FOREIGN KEY (Car_Type) REFERENCES CAR_TYPE(Car_Type)
          ON DELETE CASCADE,
  CONSTRAINT CARRFIDRENTALFK
  FOREIGN KEY (Rental_Location_ID) REFERENCES RENTAL_LOCATION(Rental_Location_ID)
          ON DELETE CASCADE ,
  CONSTRAINT CARSEATING CHECK (Seating_Capacity>2)
);

CREATE TABLE OFFER_DETAILS
(
  Promo_Code VARCHAR(15) PRIMARY KEY,
  Description VARCHAR(50),
  Promo_Type VARCHAR(20) NOT NULL,
  Percentage DECIMAL(5,2),
  Cashback  INT,
  Status VARCHAR(10) NOT NULL
);

CREATE TABLE RESERVATION
(
  Reservation_ID INT PRIMARY KEY,
  Start_Date DATE NOT NULL,
```

```
 End_Date DATE NOT NULL,
 Meter_Start INT NOT NULL,
 Meter_End INT,
 Rental_Amount INT NOT NULL,
 Insurance_Amount  INT NOT NULL,
 Actual_End_Date DATE NULL,
 Status VARCHAR(10) NOT NULL,
 License_No VARCHAR(15) NOT NULL,
 RFID CHAR(17) NOT NULL,
 Promo_Code VARCHAR(15),
 Penalty_Amount  INT  DEFAULT 0,
 Tot_Amount  INT DEFAULT 0,
 Insurance_Type VARCHAR(15),
 Drop_Location_ID INT,
 CONSTRAINT RSERVLOCATIONFK
 FOREIGN KEY (Drop_Location_ID) REFERENCES RENTAL_LOCATION(Rental_Location_ID)
        ON DELETE CASCADE,
 CONSTRAINT RESLICENSEFK
 FOREIGN KEY (License_No) REFERENCES CAR_USER(License_No)
        ON DELETE CASCADE,
 CONSTRAINT RFIDRESERVATIONFK
 FOREIGN KEY (RFID) REFERENCES CAR(RFID)
        ON DELETE CASCADE,
 CONSTRAINT PROMORESERVATIONFK
 FOREIGN KEY (Promo_Code) REFERENCES OFFER_DETAILS(Promo_Code)
        ON DELETE CASCADE,
 CONSTRAINT INSURESERVATIONFK
 FOREIGN KEY (Insurance_Type) REFERENCES INSURANCE(Insurance_Type)
        ON DELETE CASCADE
)engine=InnoDB;
```

```
CREATE TABLE PAYMENT
(
  Payment_ID INT PRIMARY KEY,
  Amount_Paid INT ,
  Card_No CHAR(16),
  Expiry_Date DATE,
  Name_On_Card VARCHAR(50),
  CVV CHAR(3),
  Billing_Address VARCHAR(50),
  Reservation_ID INT NOT NULL,
  Login_ID VARCHAR(15),
  Saved_Card_No CHAR(16),
  Paid_By_Cash CHAR(1),
  CONSTRAINT PAYMENTRESERVATIONFK
  FOREIGN KEY (Reservation_ID) REFERENCES RESERVATION(Reservation_ID)
        ON DELETE CASCADE,
  CONSTRAINT PAYMENTLOGINFK
  FOREIGN KEY (Login_ID,Saved_Card_No) REFERENCES CARD_DETAILS(Login_ID,Card_No)
        ON DELETE CASCADE
);

INSERT INTO RENTAL_LOCATION
(Rental_Location_ID,Phone,Email,Area_Name,State,Pin_Code)
VALUES
(101,'9726031111','zoom1@gmail.com','Basaweshwarnagar','KA',560079),
(102,'9726032222','zoom2@gmail.com',' Majestic','KA',560009),
(103,'9721903121','zoom3@gmail.com',' Whitefield','KA',560066),
(104,'721903121','zoom4@gmail.com','Electronic City','KA',560100),
(105,'9026981045','zoom5@gmail.com','Brigade Road','KA',560001);
```

```
INSERT INTO CAR_TYPE
(Car_Type,Price_Per_Day)
VALUES
('Economy',100),
('Standard',200),
('SUV',300),
('MiniVan',400),
('Premium',500),
('Electric',1000);

INSERT INTO INSURANCE
(Insurance_Type,Bodily_Coverage,Medical_Coverage,Collision_Coverage)
VALUES
('Liability',2500.00,5000.00,0.00),
('Comprehensive',5000.00,5000.00,5000.00);

INSERT INTO CAR_INSURANCE
(Car_Type,Insurance_Type,Insurance_Price)
VALUES
('Economy','Liability',100),
('Standard','Liability',110),
('SUV','Liability',120),
('MiniVan','Liability',140),
('Premium','Liability',190),
('Economy','Comprehensive',200),
('Standard','Comprehensive',250),
('SUV','Comprehensive',300),
('MiniVan','Comprehensive',350),
('Premium','Comprehensive',500),
('Electric','Comprehensive',900);
```

```
INSERT INTO CAR_USER

(License_No,FName,MName,Lname,Email,Address,Phone,DOB,USER_TYPE)

VALUES

('KA031983642','Patric','G','Cummins','patric.c@gmail.com','#10 Wilson
Garden,Bangalore','9022196058',('1970-01-10'),'Guest'),

('KA021495370','Brad',NULL,'Pitt','brad.pitt@gmail.com','#25 MG
Road,Bangalore','8697891045',('1980-03-20'),'Customer'),

('KA028341025','Glenn',NULL,'Maxwell','glenm@gmail.com','#30 Richmond
Road,Bangalore','8590125607',('1984-11-11'),'Customer'),

('KA029046432','Christiano',NULL,'Ronaldo','cr7@gmail.com','#7 VM
Road,Bangalore','7048015647',('1987-04-24'),'Guest'),

('KA029785313','Leonardo',NULL,'Decaprio','ldc@gmail.com','#1 Church
Street,Bangalore','9056010687',('1987-04-24'),'Customer');


INSERT INTO USER_CREDENTIALS

(Login_ID,Password,Year_Of_Membership,License_No)

VALUES

('Brad_P','ouatih','2019','KA021495370'),

('RDJ','marvel','2014','KA028341025'),

('LDC','inception','2015','KA029785313'),

('Chris_E','cap','2016','KA029046432'),

('Ben','strange','2016','KA021495370');


INSERT INTO CARD_DETAILS

(Login_ID,Name_On_Card,Card_No,Expiry_Date,CVV,Billing_Address)

VALUES

('Brad_P','Brad Pitt','4735111122223333',('2022-01-15'),'833','#25 MG Road,Bangalore'),

('LDC','Leo_De_Cap','4233908110921001',('2021-12-31'),'419','#1 Church Street,Bangalore'),

('Chris_E','Chris Evans','5123408110921001',('2022-10-31'),'820','#50 Ecity,Bangalore'),

('Ben','Benedict','3785032136469082',('2023-05-12'),'121','221 B Baker Street,Bangalore');
```

```
INSERT INTO CAR
(RFID,Rental_Location_ID,Reg_No,Seating_Capacity,Car_Type,Model,Year,Color)
VALUES
('F152206785240289',101,'KAF101',5,'Economy','i20','2007','Gold'),
('T201534710589051',101,'KYQ101',5,'Standard','Toyota Camry','2012','Grey'),
('E902103289341098',102,'XYZ671',5,'Premium','BMW X5','2015','Black'),
('R908891209418173',103,'DOP391',4,'SUV','Fortuner','2014','White'),
('N892993994858292',104,'RAC829',10,'MiniVan','Ertiga','2013','Black');

INSERT INTO OFFER_DETAILS
(PROMO_CODE,DESCRIPTION,PROMO_TYPE,PERCENTAGE,Cashback,Status)
VALUES
('CHRISTMAS10','Christmas 10% offer','Percentage',10.00,NULL,'Available'),
('July25','July Rs.250.00 discount','Cashback',NULL,250.00,'Expired'),
('MayDay1','May Day Rs.500 offer','Cashback',NULL,500.00,'Available'),
('NewYear10','New Year 10% offer','Percentage',10.00,NULL,'Available'),
('EASTER12','Easter 15% offer','Percentage',15.00,NULL,'Expired');

INSERT INTO RESERVATION
(Reservation_ID,Start_Date,End_Date,Meter_Start,Meter_End,Actual_End_Date,License_No
,RFID,Promo_Code,Insurance_Type,Drop_Location_ID,Tot_Amount)
VALUES
(1,('2019-11-06'),('2019-11-12'),81256,81300,('2019-11-
12'),'KA031983642','F152206785240289','NEWYEAR10','Liability',101,0),
(2,('2019-10-20'),('2019-10-24'),76524,76590,('2019-10-
24'),'KA021495370','T201534710589051','EASTER12','Liability',101,0),
(3,('2019-12-06'),('2019-12-12'),82001,82222,('2019-12-
15'),'KA021495370','N892993994858292','CHRISTMAS10','Comprehensive',104,0),
(4,('2019-09-01'),('2019-09-02'),51000,51100,('2019-09-
02'),'KA021495370','R908891209418173','MAYDAY1','Comprehensive',103,0),
(5,('2019-08-13'),('2019-08-15'),51000,51100,('2019-08-
15'),'KA029046432','E902103289341098','MAYDAY1','Comprehensive',105,0);
```

INSERT INTO PAYMENT

(Payment_ID,Card_NO,Expiry_Date,Name_On_Card,CVV,Billing_Address,Reservation_ID,Log in_ID,Saved_Card_No,Paid_By_Cash)

VALUES

(1001,'4735111122223333',('2022-01-15'),'Brad Pitt','100','#25 MG Road,Bangalore',1,'Brad_P','4735111122223333','N'),

(1002,'4233908110921001',('2021-12-31'),'Leo_De_Cap','419','#1 Church Street,Bangalore',5,'LDC','4233908110921001','N'),

(1003,NULL,NULL,NULL,NULL,NULL,5,NULL,NULL,'Y'),

(1004,NULL,NULL,NULL,NULL,NULL,3,NULL,NULL,'Y'),

(1005,NULL,NULL,NULL,NULL,NULL,4,NULL,NULL,'Y');

FUNCTIONAL DEPENDENCIES:

A set of attributes X *functionally determines* a set of attributes Y if the value of X determines a unique value for Y

Rental_Location_ID —> {Phone, Email, Street_Name, State, Pin_Code}

Car_Type —> Price_Per_Day

Insurance_Type —> {Bodily_Coverage, Medical_Coverage, Collision_Coverage}

{Car_Type, Insurance_Type} —> {Insurance_Price}

{License_No} —> {FName, Mname, Lname, Email,Address, Phone,DOB, User_Type}

{Login_ID} —> {Password, Year_Of_Membership, License_No}

{Login_ID, Card_No} —> {Name_On_Card, Expiry_Date, CVV, Billing_Address}

Reservation_ID —> {Start_Date, End_Date, Meter_Start, Meter_End, Rental_Amount, Insurance_Amount, License_No, VIN, Promo_Code, Tot_Amount, Insurance_Type}

Payment_ID —> {Amount_Paid, Card_No, Expiry_Date, Name_On_Card, CVV, Billing_Address, Reservation_ID, Login_ID, Saved_Card_No, Paid_By_Cash}

Promo_Code —> {Description, Promo_Type, Percentage, Discounted_Amount}

NORMALISATION:

Normalisation is done to remove redundant information in tuples and to prevent update anomalies.

We will get a normalized set of relations if we develop using ER Model and converting ER to schema approach.

Hence, All the relations would be in 1^{st}, 2^{nd}, 3^{rd} ,BCNF Normal Form.

However , We can illustrate the violation by adding a column to an existing relation.

2^{nd} NORMAL FORM

1.In the relation CARD_DETAILS 2^{nd} NF will be violated by the addition of The (License_No) attribute present in USER_CREDENTIALS relation. This is because the attribute (License_No) is functionally dependent on the prime attribute (Login_ID)

i.e [Login_ID->License_No] but the primary key of the relation CARD_DETAILS is [Login_ID,Card_No]. In this case , The non-key attribute [Licence_No] is not fully functional dependent on the primary key [Login_ID,Card_No]. Hence 2^{nd} NF is violated.

2.In the relation CAR_INSUARANCE 2^{nd} NF will be violated by the addition of the attribute (Price_Per_Day) present in the relation CAR_TYPE and (Medical_Coverage) present in the relation INSUARANCE. This is because the attribute (Price_Per_Day) is fully functional dependent on the prime attribute(Car_Type) and the attribute (Medical_Coverage) is fully functional dependent on the prime attribute(Insuarance_type) .

i.e[Car_Type->Price_Per_Day] and [Insuarance_Type->Medical_Coverage] respectively.But the primary key of the relation car_insuarance is [Car_Type,Insurance_Type].In this relation,the non-key attributes (Price_Per_Day) and (Medical_Coverage) are not fully functional dependent on the primary key (Car_Type,Insurance_Type).

Hence 2^{nd} NF is violated.

In order to convert the relation into 2^{nd} NF We need to decompose the car_insurance relation into

R1(Car_Type,Price_Per_Day)

R2(Car_Type,Insurance Type)

R3(Insurance_Type,Medical_Coverage)

The above Decomposition is Loseless , Attribute and Functional Dependency preserving.

3rd NORMAL FORM

A relation is in third normal form if it is in 2NF and no non key attribute is transitively dependent on the primary key.

If we add the attribute License_No from the relation user_credentials relation into payment relation,We get a violation in the 3rd NF.

The following dependencies exist:

1.Amount_Paid,Card_No,expiry_Date,Name_On_Card,CVV,Billining_Address,Reservation_ID , Login_ID, Saved_Card_No, Paid_By_Cash are functionally dependent on Payment_ID i.e (Payment_ID-->Amount_Paid,Card_No,expiry_Date,Name_On_Card,CVV,Billining_Address, Reservation_ID, Login_ID, Saved_Card_No, Paid_By_Cash).

2. License_No is functionally dependent on Login_ID (Login_ID --> License_No)

The table in this example is in 1NF and in 2NF. But there is a transitive dependency between Login_ID and License_No, because Login_ID is not the primary key of this relation.

Hence,This Relation is not in 3rd NF

Here, License_NO is dependent on Login_ID and Login_ID is dependent on Payment_ID. The non-prime attribute (Lisence_NO) is transitively dependent on super key(Payment_ID). It violates the rule of third normal form.

That's why we need to move the License_NO to a new relation, with Login_ID as a Primary key.These relations are now in the 3rd NF and the decomposition is Loseless ,Attribute and Functional Dependency Preserving.

TRIGGERS:

Triggers are the SQL codes that are automatically executed in response to certain events on a particular table. These are used to maintain the integrity of the data.

Before Triggers have been used in this Database:

TRIGGER 1:

```
delimiter |

CREATE TRIGGER rent_amt before insert on reservation

FOR EACH ROW

        BEGIN

                SET @carType=(select C.Car_Type from car as C where new.RFID=C.RFID);

    SET @price=(select Price_Per_Day from car_type where Car_Type=@carType);
```

```
      set new.Rental_Amount=@price*(datediff(new.Actual_End_Date,new.Start_Date));
    END |
delimiter ;
```

TRIGGER 2:

```
delimiter |
CREATE TRIGGER ins_amt before insert on reservation
FOR EACH ROW
          BEGIN
                SET @carType=(select C.Car_Type from car  C where new.RFID=C.RFID);
      SET @price=(select C.Insurance_Price from car_insurance C where((
C.Car_Type=@carType) and (C.Insurance_Type=new.Insurance_Type)));
      set new.Insurance_Amount=@price*(datediff(new.Actual_End_Date,new.Start_Date));
    END |
delimiter ;
```

TRIGGER 3:

```
delimiter |
CREATE TRIGGER date_check before insert on reservation
FOR EACH ROW
          BEGIN
                if(DATEDIFF(new.Actual_End_Date,new.End_date)>0) then
           set new.Penalty_Amount=(DATEDIFF(new.Actual_End_Date,new.End_date)*50);
       END IF;
    END |
delimiter ;
```

TRIGGER 4:

```
delimiter |
CREATE TRIGGER Tot_Amount before insert on reservation
FOR EACH ROW
```

```
BEGIN
        DECLARE v1,v2 INT;
    select offer.Percentage,offer.Cashback into v1,v2 from offer_details as offer where new.Promo_Code=offer.Promo_Code;
        SET @availability=(select offer.STATUS from offer_details as offer where offer.Promo_Code=new.Promo_Code);
    if v1 is not null then
      if (@availability='Available') then
        set new.Tot_Amount=(new.Rental_Amount+new.Insurance_Amount)-((v1/100)*(new.Rental_Amount+new.Insurance_Amount))+new.Penalty_amount;
      else
set new.Tot_Amount =(new.Rental_Amount+new.Insurance_Amount) +new.Penalty_amount;
       END IF;
    else
        if (@availability='Available') then
            set new.Tot_Amount=(new.Rental_Amount+new.Insurance_Amount+new.Penalty_Amount) -v2;
          else
            set new.Tot_Amount=(new.Rental_Amount+new.Insurance_Amount)+new.Penalty_amount;
           END IF;
    END IF;
      END |
delimiter ;
```

With the help of Triggers 1,2,3,4 it is possible to retrieve and calculate Rental amount, Insurance amount, Penalty Amount from particular relations into reservation relation in the database.

IMPLEMENTATION:

```
INSERT INTO RESERVATION
(Reservation_ID,Start_Date,End_Date,Meter_Start,Meter_End,Actual_End_Date,License_No,RFID,Promo_Code,Insurance_Type,Drop_Location_ID,Tot_Amount)
```

VALUES

(1,('2019-11-06'),('2019-11-12'),81256,81300,('2019-11-12'),'KA031983642','F152206785240289','NEWYEAR10','Liability',101,0),

(2,('2019-10-20'),('2019-10-24'),76524,76590,('2019-10-24'),'KA021495370','T201534710589051','EASTER12','Liability',101,0),

(3,('2019-12-06'),('2019-12-12'),82001,82222,('2019-12-15'),'KA021495370','N892993994858292','CHRISTMAS10','Comprehensive',104,0),

(4,('2019-09-01'),('2019-09-02'),51000,51100,('2019-09-02'),'KA021495370','R908891209418173','MAYDAY1','Comprehensive',103,0),

(5,('2019-08-13'),('2019-08-15'),51000,51100,('2019-08-15'),'KA029046432','E902103289341098','MAYDAY1','Comprehensive',105,0);

Reservation_ID	Start_Date	End_Date	Meter_Start	Meter_End	Rental_Amount	Insurance_Amount	Actual_End_Date	License_No	RFID	Promo_Code	Penalty_Amount	Tot_Amount	Insurance_Type	Drop_Location_ID
1	2019-11-06	2019-11-12	81256	81300	600	600	2019-11-12	KA031983642	F152206785240289	NEWYEAR10	0	1080	Liability	101
2	2019-10-20	2019-10-24	76524	76590	800	440	2019-10-24	KA021495370	T201534710589051	EASTER12	0	1240	Liability	101
3	2019-12-06	2019-12-12	82001	82222	3600	3150	2019-12-15	KA021495370	N892993994858292	CHRISTMAS10	150	6225	Comprehensive	104
4	2019-09-01	2019-09-02	51000	51100	300	300	2019-09-02	KA021495370	R908891209418173	MAYDAY1	0	100	Comprehensive	103
5	2019-08-13	2019-08-15	51000	51100	1000	1000	2019-08-15	KA029046432	E902103289341098	MAYDAY1	0	1500	Comprehensive	105

TRIGGER 5:

```
Delimiter |

CREATE TRIGGER amt_paid before insert on payment

FOR EACH ROW

        BEGIN

            DECLARE a1 INT;

            select r.Tot_Amount INTO a1 from reservation r where
new.Reservation_ID=r.Reservation_ID;

      SET new.Amount_Paid =a1;

   END |

delimiter ;
```

Trigger 5 is used to retrieve Total Amount from reservation relation into Amount_Paid attribute of payment relation.

IMPLEMENTATION:

INSERT INTO PAYMENT

(Payment_ID,Card_NO,Expiry_Date,Name_On_Card,CVV,Billing_Address,Reservation_ID,Login_ID,Saved_Card_No,Paid_By_Cash)

VALUES

(1001,'4735111122223333',('2022-01-15'),'Brad Pitt','100','#25 MG Road,Bangalore',1,'Brad_P','4735111122223333','N'),

(1002,'4233908110921001',('2021-12-31'),'Leo_De_Cap','419','#1 Church Street,Bangalore',5,'LDC','4233908110921001','N'),

(1003,NULL,NULL,NULL,NULL,NULL,5,NULL,NULL,'Y'),

(1004,NULL,NULL,NULL,NULL,NULL,3,NULL,NULL,'Y'),

(1005,NULL,NULL,NULL,NULL,NULL,4,NULL,NULL,'Y');

Payment_ID	Amount_Paid	Card_No	Expiry_Date	Name_On_Card	CVV	Billing_Address	Reservation_ID	Login_ID	Saved_Card_No	Paid_By_Cash
1001	1080	4735111122223333	2022-01-15	Brad Pitt	100	#25 MG Road,Bangalore	1	Brad_P	4735111122223333	N
1002	1500	4233908110921001	2021-12-31	Leo_De_Cap	419	#1 Church Street,Bangalore	5	LDC	4233908110921001	N
1003	1500	NULL	NULL	NULL	NULL	NULL	5	NULL	NULL	Y
1004	6225	NULL	NULL	NULL	NULL	NULL	3	NULL	NULL	Y
1005	100	NULL	NULL	NULL	NULL	NULL	4	NULL	NULL	Y

TRANSACTIONS:

A transaction is a sequence of operations performed (using one or more SQL statements) on a database as a single logical unit of work. The effects of all the SQL statements in a transaction can be either all committed (applied to the database) or all rolled back (undone from the database). A database transaction must be atomic, consistent, isolated and durable.

It can be implemented on a particular relation using InnoDB Engine.

start transaction ;

INSERT INTO PAYMENT

(Payment_ID,Card_NO,Expiry_Date,Name_On_Card,CVV,Billing_Address,Reservation_ID,Login_ID,Saved_Card_No,Paid_By_Cash)

values (1001,'4735111122223333',('2022-01-15'),'Brad Pitt','100','#25 MG Road,Bangalore',1,'Brad_P','4735111122223333','N');

savepoint transaction1;

INSERT INTO PAYMENT

(Payment_ID,Card_NO,Expiry_Date,Name_On_Card,CVV,Billing_Address,Reservation_ID,Login_ID,Saved_Card_No,Paid_By_Cash)

values (1002,'4233908110921001',('2021-12-31'),'Leo_De_Cap','419','#1 Church Street,Bangalore',5,'LDC','4233908110921001','N');

rollback to savepoint transaction1;

commit;

Without Rollback:

Payment_ID	Amount_Paid	Card_No	Expiry_Date	Name_On_Card	CVV	Billing_Address	Reservation_ID	Login_ID	Saved_Card_No	Paid_By_Cash
1001	1080	4735111122223333	2022-01-15	Brad Pitt	100	#25 MG Road,Bangalore	1	Brad_P	4735111122223333	N
1002	1500	4233908110921001	2021-12-31	Leo_De_Cap	419	#1 Church Street,Bangalore	5	LDC	4233908110921001	N
1003	1500	NULL	NULL	NULL	NULL	NULL	5	NULL	NULL	Y
1004	6225	NULL	NULL	NULL	NULL	NULL	3	NULL	NULL	Y
1005	100	NULL	NULL	NULL	NULL	NULL	4	NULL	NULL	Y

With Rollback:

Payment_ID	Amount_Paid	Card_No	Expiry_Date	Name_On_Card	CVV	Billing_Address	Reservation_ID	Login_ID	Saved_Card_No	Paid_By_Cash
1001	1080	4735111122223333	2022-01-15	Brad Pitt	100	#25 MG Road,Bangalore	1	Brad_P	4735111122223333	N
1003	1500	NULL	NULL	NULL	NULL	NULL	5	NULL	NULL	Y
1004	6225	NULL	NULL	NULL	NULL	NULL	3	NULL	NULL	Y
1005	100	NULL	NULL	NULL	NULL	NULL	4	NULL	NULL	Y

SQL QUERIES:

CO-RELATED NESTED QUERIES:

1.Write a Query to select the phone number of the rental location where a car has not yet been allocated

```
select r.Phone
from rental_location r
where not exists( select c.Rental_Location_ID
        from car c
        where c.Rental_Location_ID=r.Rental_Location_ID);
```

Phone
9026981045

2.Write a Query to select card name of those users who have a valid login ID and are members of the rental service since 2015.

```
select c.Name_On_Card
from card_details c
where exists( select u.Login_ID
        from user_credentials u
        where c.Login_ID=u.Login_ID and Year_Of_Membership>2015);
```

Name_On_Card
Benedict
Brad Pitt
Chris Evans

AGGREGATE QUERIES:

1. Write a query to display the payment ID wherein the average amount paid is greater than 1000.

   Select p.Payment_ID,p.Amount_Paid
   from payment p
   group by Payment_ID
   having avg(Amount_Paid)>1000;

Payment_ID	Amount_Paid
1001	1080
1002	1500
1003	1500
1004	6225

2. Write a query to display the Insurance Type and maximum Insurance Price for each car type.

   Select *
   from car_insurance
   group by Car_Type
   having max(Insurance_Price)
   order by Insurance_Price asc;

Car_Type	Insurance_Type	Insurance_Price
Economy	Comprehensive	200
Standard	Comprehensive	250
SUV	Comprehensive	300
MiniVan	Comprehensive	350
Premium	Comprehensive	500
Electric	Comprehensive	900

JOIN QUERIES:

1) select *

from card_details c

right join user_credentials u using(Login_ID);

Login_ID	Password	Year_Of_Membership	License_No	Name_On_Card	Card_No	Expiry_Date	CVV	Billing_Address
Ben	strange	2016	KA021495370	Benedict	3785032136469082	2023-05-12	121	221 B Baker Street,Bangalore
Brad_P	ouatih	2019	KA021495370	Brad Pitt	4735111122223333	2022-01-15	833	#25 MG Road,Bangalore
Chris_E	cap	2016	KA029046432	Chris Evans	5123408110921001	2022-10-31	820	#50 Ecity,Bangalore
LDC	inception	2015	KA029785313	Leo_De_Cap	4233908110921001	2021-12-31	419	#1 Church Street,Bangalore
RDJ	marvel	2014	KA028341025	NULL	NULL	NULL	NULL	NULL

2) select *

from user_credentials

left join car_user

using(License_No);

License_No	Login_ID	Password	Year_Of_Membership	Fname	Mname	Lname	Email	Address	Phone	DOB	User_Type
KA021495370	Ben	strange	2016	Brad	NULL	Pitt	brad.pitt@gmail.com	#25 MG Road,Bangalore	8697891045	1980-03-20	Customer
KA021495370	Brad_P	ouatih	2019	Brad	NULL	Pitt	brad.pitt@gmail.com	#25 MG Road,Bangalore	8697891045	1980-03-20	Customer
KA029046432	Chris_E	cap	2016	Christiano	NULL	Ronaldo	cr7@gmail.com	#7 VM Road,Bangalore	7048015647	1987-04-24	Guest
KA029785313	LDC	inception	2015	Leonardo	NULL	Decaprio	ldc@gmail.com	#1 Church Street,Bangalore	9056010687	1987-04-24	Customer
KA028341025	RDJ	marvel	2014	Glenn	NULL	Maxwell	glenm@gmail.com	#30 Richmond Road,Bangalore	8590125607	1984-11-11	Customer

CONCLUSION:

The project was done to understand the working of an actual car rental organization. Implementations of various queries was made possible to retrieve information from databases. This database currently has only a few tuples in all of the relations to depict a clearer working of the System.

FUTURE ENHANCEMENTS:

- Greater No. of tuples can be fed into the database.
- Making Use of more number of triggers.
- Providing Front-end and Back-end with phpMyadmin support to the database to create a real-time Car-Rental website.
- Inclusion of accessories to be purchased along with the car.