CS 6360.002 Database Design

Uber

Team:

Harshita Rastogi (HXR190001)

Shivam Gupta (SXG190040)

Tejas Gupta (TXG180021)

PROJECT DESCRIPTION:

Uber is a ride providing (car-for-hire) company that provides taxi service to the customers. Unlike regular taxi service, Uber lets you join as a driver with your personal registered vehicle and one can make money by completing the customer request.

Important Components of the System

The main important actors in the domain are - Drivers and Customers, which form the very basic foundation of the Database Model.

Driver - a person above the age of 18 having an authorized unexpired driver’s license and SSN, possessing a registered vehicle under insurance

Customer - a person having an account with Uber who needs to be picked from a certain location and dropped to a desired location

The working of Uber System

Uber is a real time application that allows a customer to request for a ride from the current location of the customer to their desired destination.

- ➢ A registered customer on Uber can have various types of accounts such as Ordinary, Platinum, Gold based on the number of completed rides. On the basis of the account, the customer is eligible for an offer by Uber which can be availed by filling in the promo code.
- ➢ The customer requests for a ride and has the flexibility of choosing the type of ride (UberX, UberXL). The estimated fare price is also visible to the customer based on the type of ride chosen. The customer gets allotted a ride according to the availability of the driver in the current location.
- ➢ Once the trip request has been finalized, a driver arrives at the location of the customer to pick up the customer.
- ➢ After reaching the desired final destination, the driver ends the trip and the customer can see the final fare of the trip.
- ➢ Once the trip has been completed, the fare gets reduced from the payment method option chosen by the customer at the beginning of the trip. The customer has the option of adding tip amount to the driver. Moreover, both the driver and the customer can rate each other based on the trip and provide comments/feedback if necessary.

PROJECT DATA REQUIREMENTS

A User entity has been created and the user type can be : a customer or a driver which are both registered with Uber and regarded as Uber Users.

The Uber system must store the personal information about the **Uber user** such as name, date of birth, address, email and phone number. It must also identify the users uniquely by a system generated identification code (UberID). A user in the system can either be a customer or a driver. A driver can also be a customer when he/she is not riding their vehicle.

The **customer** must be identified by a customer id which has to be derived from the UberID. The type(ordinary, gold, platinum) of the customer must also be recorded.

Just like the Customer ID (CID), the driver must be identified by a driver ID (DID) is derived from the UberID. Additional information regarding the **driver** must also be stored such as Driver’s License Number, DL Expiry date and SSN.

Every **driver** will be having separate shifts, so a **shift** table is also required which store the Information regarding the driver’s shifts which will contain the Driver ID, the time of login and logout which will denote during which time the driver is available because it will be required during the time of allotting the rides to the drivers.

Vehicle will contain the Vehicle identification number (VID), model of the car, the manufacturing year that when it was created, purchase date, whether the car is active or not, condition of the car which will be utilized for maintenance if the condition is bad., the capacity of the car that how many seats are available, the insurance number and the insurance expiry date which will be required for Insurance renewal and also the date which tells that when the car was last checked for maintenance.

Trip Requests will be identified with some specific id (Trip ID) and will be containing the information about the type of the trip, pickup location, dropoff location, ride d, estimated fare. It will also contain the driver ID who will be allotted a ride and the customer ID who will be booking the ride.

In our Uber Dababase system the trips can be **completed** or **incompleted**.

Completed trips will be having the information about the time when the driver arrived, the pickup time when the rise actually got started, the drop off time to the destination when the rise gets finished. It also contains tip given by the customer, the surge time which denotes the high traffic which will be utilized in calculating the actual fare.

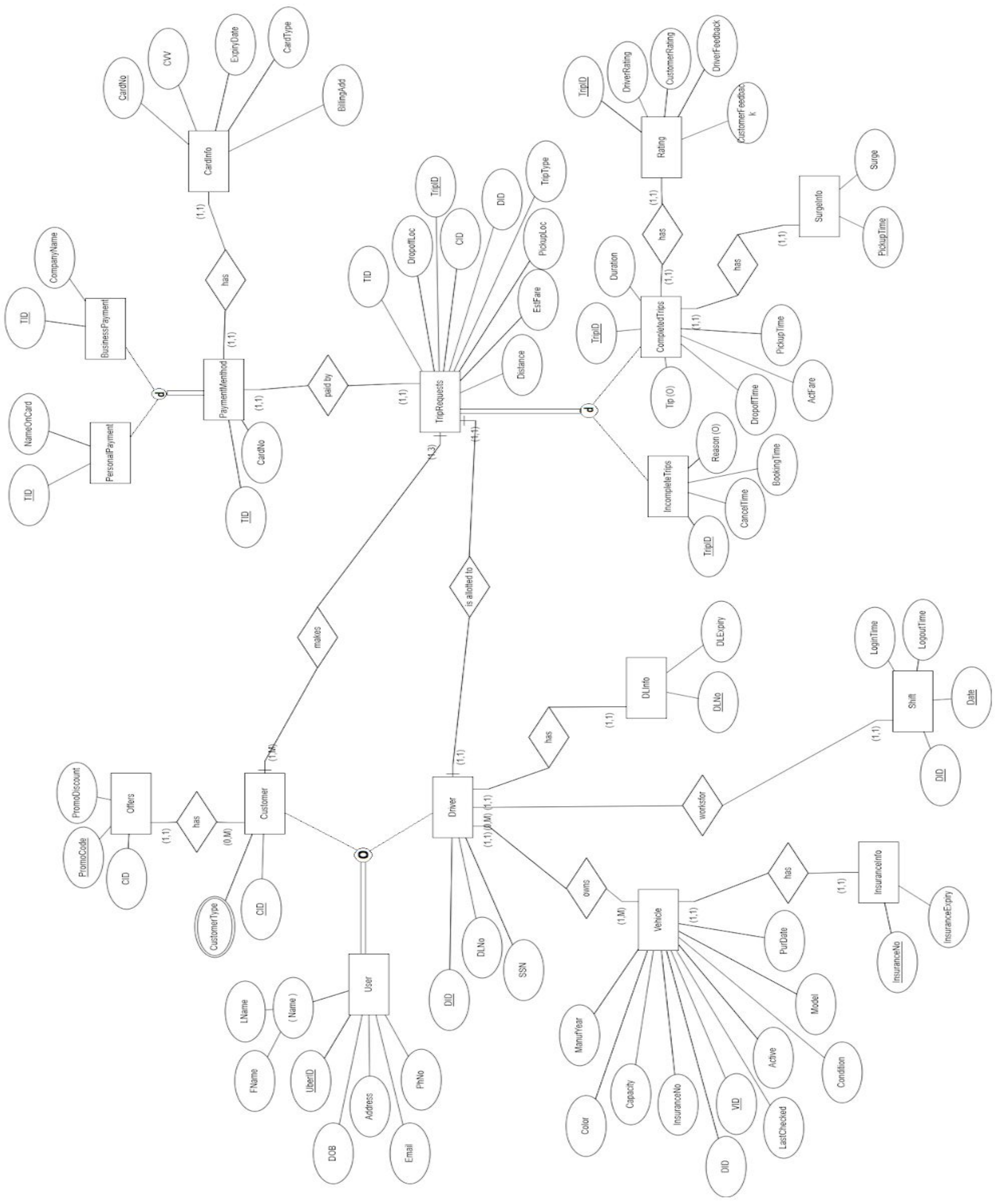
I**ncompleted trips** will be having the information about the time when the booking got done and the cancelation time and also the reason for the cancellation.

Payment method will contain the transaction ID (TID), the information the the card like card number, CVV number, expiry date, the type of the card like visa or mastercard, the billing address where the bill invoice is to be sent.

The payment can be done through personal account or through a company. That's why it is divided into two separate tables: **Personal payment** and the **business payment.** Personal payment will have additional information like the name written on the card and the business payment will have additional information like the name of the company who is paying on behalf of the customer.

In the end The feedback is very important for Uber which helps the people to know about the review of the whole business. So **Rating** table has been constructed which will contain the trip ID, the ratings given to the driver and the customer which will contain the number of stars out of 10 along the the feedback for the driver and the customer.

EER DIAGRAM (After Normalisation)



<u>MAPPING EER DIAGRAM TO RELATIONAL SCHEMA</u>

*Primary Key - **Bold** *Foreign Key - *Italics and Underlined*

- UberUser { **UberID**, FName, LName, PhNo, Email, Address , DOB }

- Customer { ***<u>CID</u>***, CustomerType, PromoCode, PromoDiscount }
  FOREIGN KEY (CID) **REFERENCES** UberUser(UberID)

- Driver { ***<u>DID</u>*** , SSN, DLNo, DLExpiry }
  FOREIGN KEY (DID) **REFERENCES** UberUser(UberID)

- Vehicle {**VID**, *<u>DID</u>*, Model, Color , ManufYear, PurDate, Active, Condition , Cpty, InsuranceNo, InsuranceExpiry, LastChecked }
  FOREIGN KEY (DID) **REFERENCES** Driver(DID)

- TripRequests { ***<u>TripID</u>***, *<u>CID</u>*, *<u>DID</u>*, TripType, PickupLoc, DropoffLoc, Distance, EstFare, *<u>TID</u>* }
  FOREIGN KEY (TID) **REFERENCES** PaymentMethod(TID)
  FOREIGN KEY (CID) **REFERENCES** Customer(CID)
  FOREIGN KEY (DID) **REFERENCES** Driver(DID)

- CompletedTrips { ***<u>TripID</u>***, DriverArrAt, PickupTime, DropOffTime, ActFare, Tip, Surge }
  FOREIGN KEY (TripID) **REFERENCES** TripRequests(TripID)

- IncompleteTrips { **TripID**, BookingTime, CancelTime, Reason }
  FOREIGN KEY (TripID) **REFERENCES** TripRequests(TripID)

- PaymentMethod { ***<u>TID</u>***, CardNo, CVV, ExpDate, CardType, BillingAdd }

- PersonalPayment { **TID** , NameOnCard }
  FOREIGN KEY (TID) **REFERENCES** PaymentMethod(TID)

- BusinessPayment { **TID**, CompanyName }
  FOREIGN KEY (TID) **REFERENCES** PaymentMethod(TID)

- Rating { **TripID** , DriverRating, CustomerRating, DriverFeedback, CustomerFeedback }
  FOREIGN KEY (TripID) **REFERENCES** CompletedTrips(TripID)

- Shift { **DID** , LoginTime, LogoutTime }
  FOREIGN KEY (DID) **REFERENCES** Driver(DID)

❖ <u>FUNCTIONAL DEPENDENCIES AND NORMALIZATION</u>

All the tables contain atomic values. There does not exist any partial dependency in the tables. Therefore, the schema is already obeys 1NF and 2NF.

Driver: There exists transitive dependency. FD2 violates 3NF.

Driver { **DID** , SSN, DLNo, DLExpiry }

FD1: DID---> DLNo

FD2: DLNo---> DLExpiry

So, the new tables are:

Driver{ **DID**, SSN, *<u>DLNo</u>*}

DrivingLicenceInfo{ **DLNo**, DLExpiry}

Customer: There exists transitive dependency. FD2 violates 3NF.

Customer { **CID**, CustomerType, PromoCode, PromoDiscount }

FD1: CID ---> PromoCode

FD2: PromoCode ---> PromoDiscount

So, the new tables are:

Customer { **CID**, CustomerType, *<u>PromoCode</u>* }

Offer { **PromoCode**, PromoDiscount }

Vehicle: There exists transitive dependency. FD2 violates 3NF.

Vehicle { **VID**, *DID*, Model, Color , ManufYear, PurDate, Active, Condition , Cpty, InsuranceNo, InsuranceExpiry, LastChecked }

FD1: VID---> InsuranceNo

FD2: InsuranceNo> InsuranceExpiry

So, the new tables are:

Vehicle{ **VID**, *DID*, Model, Color , ManufYear, PurDate, Active, Condition , Cpty, InsuranceNo, LastChecked }

InsuranceInfo{ **InsuranceNo**, InsuranceExpiry}

PaymentMethod: There exists transitive dependency. FD2 violates 3NF.

PaymentMethod{ **TID**, CardNo, CVV, ExpDate, CardType, BillingAdd }

FD1: TID ---> CardNo

FD2: CardNo ---> CVV, ExpDate, CardType, BillingAdd

So, the new tables are:

PaymentMethod { **TID** , CardNo }

CardInfo { **CardNo**, CVV, ExpDate, CardType, BillingAdd }

CompletedTrips: There exists transitive dependency. FD2 violates 3NF.

CompletedTrips { **TripID,** DriverArrAt, PickupTime, DropOffTime, ActFare, Tip, Surge }

FD1: TripID ---> PickupTime

FD2: PickupTime ---> Surge

So the new tables are:

CompletedTrips { **TripID,** DriverArrAt, *PickupTime*, DropOffTime, ActFare, Tip }

SurgeInfo { **PickupTime** , Surge }

NORMALIZED RELATIONAL SCHEMA

*Primary Key - **Bold**

*Foreign Key - *Italics and Underlined*

- UberUSer { ***UberID***, FName, LName, PhNo, Email, Address , DOB}
- Customer { ***CID***, CustomerType, *PromoCode*}
  FOREIGN KEY (CID) **REFERENCES** UberUser(UberID)
- Offer { **PromoCode**, PromoDiscount }
  FOREIGN KEY (PromoCode) **REFERENCES** Customer(PromoCode)
- Driver { ***DID*** , SSN, *DLNo*}
  FOREIGN KEY (DID) **REFERENCES** UberUser(UberID)
- DrivingLicenceInfo{ **DLNo**, DLExpiry}
  FOREIGN KEY (DLNo) **REFERENCES** Driver(DLNo)
- Vehicle{ **VID**, *DID*, Model, Color , ManufYear, PurDate, Active, Condition , Cpty, *InsuranceNo*, LastChecked }
  FOREIGN KEY (DID) **REFERENCES** Driver(DID)
- InsuranceInfo {**InsuranceNo**, InsuranceExpiry}
  FOREIGN KEY (InsuranceNo) **REFERENCES** Vehicle(InsuranceNo)
- TripRequests { ***TripID***, *CID*, *DID*, TripType, PickupLoc, DropoffLoc, Distance, EstFare, *TID* }
  FOREIGN KEY (TID) **REFERENCES** PaymentMethod(TID)
  FOREIGN KEY (CID) **REFERENCES** Customer(CID)
  FOREIGN KEY (DID) **REFERENCES** Driver(DID)
- CompletedTrips { ***TripID*, **DriverArrAt, *PickupTime*, DropOffTime, ActFare, Tip }
  FOREIGN KEY (TripID) **REFERENCES** TripRequests(TripID)
- SurgeInfo { **PickupTime** , Surge }
  FOREIGN KEY (PickupTime) **REFERENCES** CompletedTrips(PickupTime)
- IncompleteTrips { **TripID**, BookingTime, CancelTime, Reason }
  FOREIGN KEY (TripID) **REFERENCES** TripRequests(TripID)

- PaymentMethod { ***<u>TID</u>***, *<u>CardNo</u>* }

- CardInfo { **CardNo**, CVV, ExpDate, CardType, BillingAdd }
  FOREIGN KEY (CardNo) **REFERENCES** PaymentMethod(CardNo)

- PersonalPayment { **TID** , NameOnCard }
  FOREIGN KEY (TID) **REFERENCES** PaymentMethod(TID)

- BusinessPayment { **TID**, CompanyName }
  FOREIGN KEY (TID) **REFERENCES** PaymentMethod(TID)

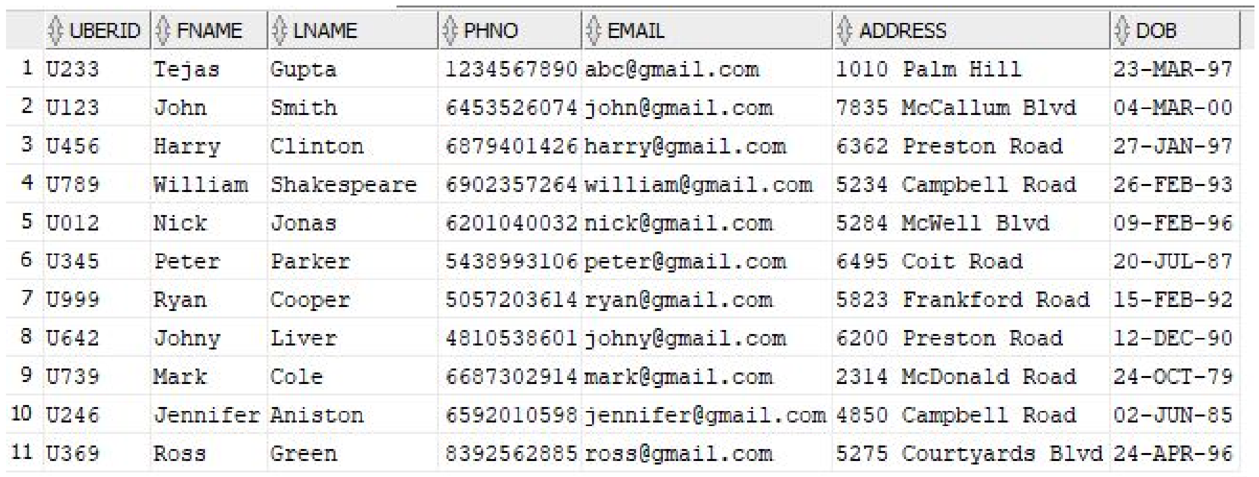
- RATING { **TripID** , DriverRating, CustomerRating, DriverFeedback, CustomerFeedback }
  FOREIGN KEY (TripID) **REFERENCES** CompletedTrips(TripID)

- SHIFT { **DID** , LoginTime, LogoutTime }
  FOREIGN KEY (DID) **REFERENCES** Driver(DID)

RESULTS:

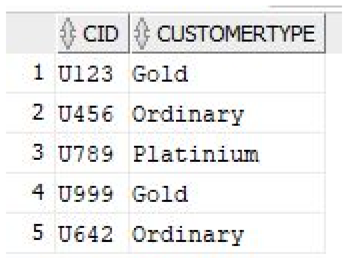
- **Snapshots of the Tables created on Oracle SQL Developer**

UberUser:



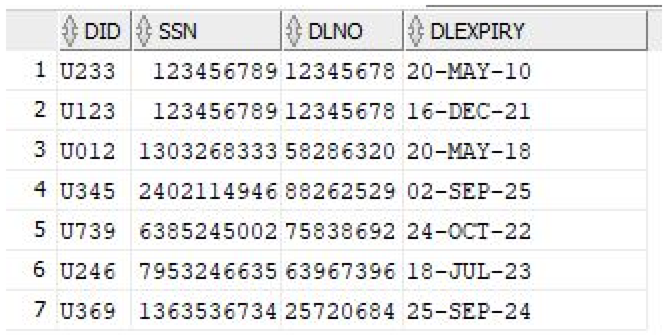
	UBERID	FNAME	LNAME	PHNO	EMAIL	ADDRESS	DOB
1	U233	Tejas	Gupta	1234567890	abc@gmail.com	1010 Palm Hill	23-MAR-97
2	U123	John	Smith	6453526074	john@gmail.com	7835 McCallum Blvd	04-MAR-00
3	U456	Harry	Clinton	6879401426	harry@gmail.com	6362 Preston Road	27-JAN-97
4	U789	William	Shakespeare	6902357264	william@gmail.com	5234 Campbell Road	26-FEB-93
5	U012	Nick	Jonas	6201040032	nick@gmail.com	5284 McWell Blvd	09-FEB-96
6	U345	Peter	Parker	5438993106	peter@gmail.com	6495 Coit Road	20-JUL-87
7	U999	Ryan	Cooper	5057203614	ryan@gmail.com	5823 Frankford Road	15-FEB-92
8	U642	Johny	Liver	4810538601	johny@gmail.com	6200 Preston Road	12-DEC-90
9	U739	Mark	Cole	6687302914	mark@gmail.com	2314 McDonald Road	24-OCT-79
10	U246	Jennifer	Aniston	6592010598	jennifer@gmail.com	4850 Campbell Road	02-JUN-85
11	U369	Ross	Green	8392562885	ross@gmail.com	5275 Courtyards Blvd	24-APR-96

Customer:



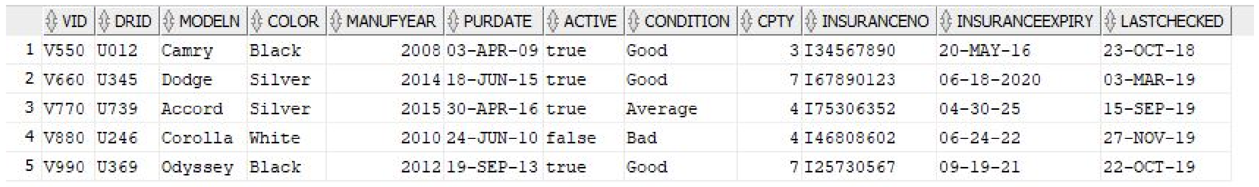
	CID	CUSTOMERTYPE
1	U123	Gold
2	U456	Ordinary
3	U789	Platinium
4	U999	Gold
5	U642	Ordinary

Driver:



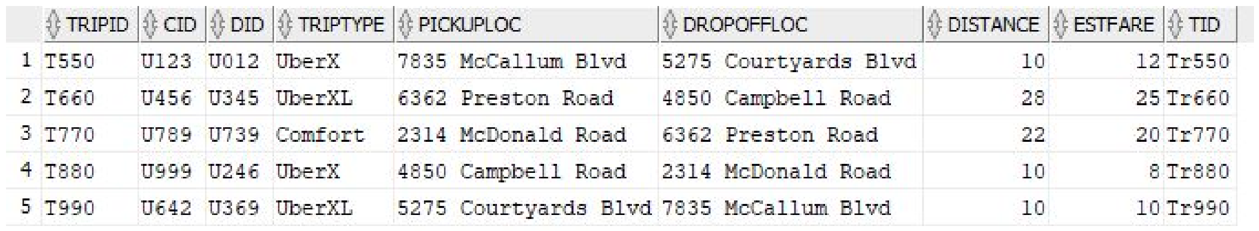
	DID	SSN	DLNO	DLEXPIRY
1	U233	123456789	12345678	20-MAY-10
2	U123	123456789	12345678	16-DEC-21
3	U012	1303268333	58286320	20-MAY-18
4	U345	2402114946	88262529	02-SEP-25
5	U739	6385245002	75838692	24-OCT-22
6	U246	7953246635	63967396	18-JUL-23
7	U369	1363536734	25720684	25-SEP-24

Vehicle:



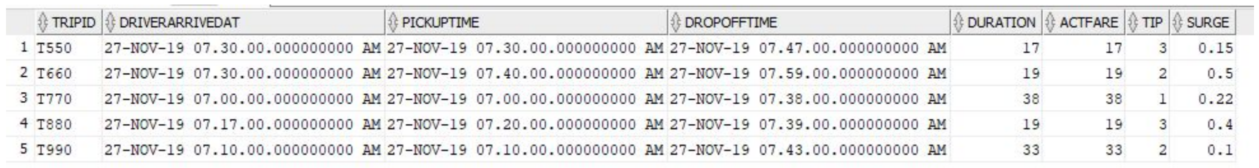
	VID	DRID	MODELN	COLOR	MANUFYEAR	PURDATE	ACTIVE	CONDITION	CPTY	INSURANCENO	INSURANCEEXPIRY	LASTCHECKED
1	V550	U012	Camry	Black	2008	03-APR-09	true	Good	3	I34567890	20-MAY-16	23-OCT-18
2	V660	U345	Dodge	Silver	2014	18-JUN-15	true	Good	7	I67890123	06-18-2020	03-MAR-19
3	V770	U739	Accord	Silver	2015	30-APR-16	true	Average	4	I75306352	04-30-25	15-SEP-19
4	V880	U246	Corolla	White	2010	24-JUN-10	false	Bad	4	I46808602	06-24-22	27-NOV-19
5	V990	U369	Odyssey	Black	2012	19-SEP-13	true	Good	7	I25730567	09-19-21	22-OCT-19

TripRequests:



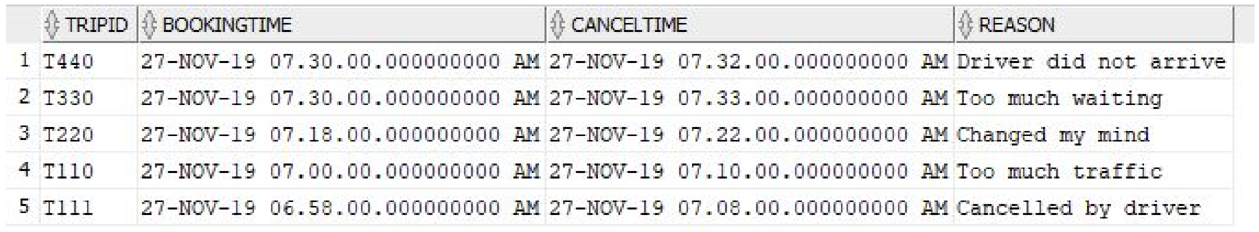
	TRIPID	CID	DID	TRIPTYPE	PICKUPLOC	DROPOFFLOC	DISTANCE	ESTFARE	TID
1	T550	U123	U012	UberX	7835 McCallum Blvd	5275 Courtyards Blvd	10	12	Tr550
2	T660	U456	U345	UberXL	6362 Preston Road	4850 Campbell Road	28	25	Tr660
3	T770	U789	U739	Comfort	2314 McDonald Road	6362 Preston Road	22	20	Tr770
4	T880	U999	U246	UberX	4850 Campbell Road	2314 McDonald Road	10	8	Tr880
5	T990	U642	U369	UberXL	5275 Courtyards Blvd	7835 McCallum Blvd	10	10	Tr990

CompletedTrips:



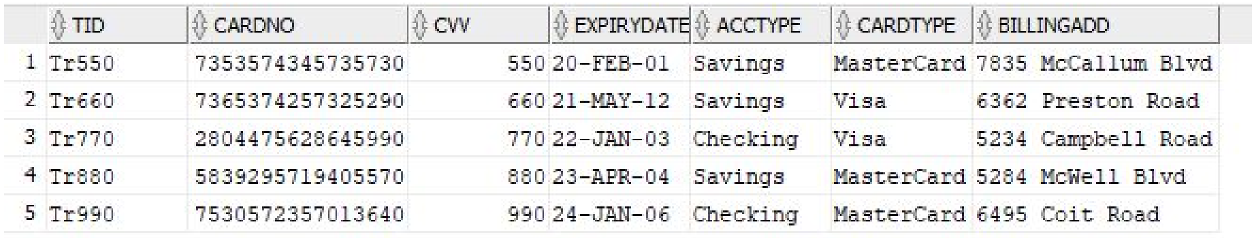
	TRIPID	DRIVERARRIVEDAT	PICKUPTIME	DROPOFFTIME	DURATION	ACTFARE	TIP	SURGE
1	T550	27-NOV-19 07.30.00.000000000 AM	27-NOV-19 07.30.00.000000000 AM	27-NOV-19 07.47.00.000000000 AM	17	17	3	0.15
2	T660	27-NOV-19 07.30.00.000000000 AM	27-NOV-19 07.40.00.000000000 AM	27-NOV-19 07.59.00.000000000 AM	19	19	2	0.5
3	T770	27-NOV-19 07.00.00.000000000 AM	27-NOV-19 07.00.00.000000000 AM	27-NOV-19 07.38.00.000000000 AM	38	38	1	0.22
4	T880	27-NOV-19 07.17.00.000000000 AM	27-NOV-19 07.20.00.000000000 AM	27-NOV-19 07.39.00.000000000 AM	19	19	3	0.4
5	T990	27-NOV-19 07.10.00.000000000 AM	27-NOV-19 07.10.00.000000000 AM	27-NOV-19 07.43.00.000000000 AM	33	33	2	0.1

IncompleteTrips:



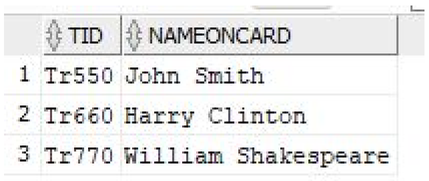
	TRIPID	BOOKINGTIME	CANCELTIME	REASON
1	T440	27-NOV-19 07.30.00.000000000 AM	27-NOV-19 07.32.00.000000000 AM	Driver did not arrive
2	T330	27-NOV-19 07.30.00.000000000 AM	27-NOV-19 07.33.00.000000000 AM	Too much waiting
3	T220	27-NOV-19 07.18.00.000000000 AM	27-NOV-19 07.22.00.000000000 AM	Changed my mind
4	T110	27-NOV-19 07.00.00.000000000 AM	27-NOV-19 07.10.00.000000000 AM	Too much traffic
5	T111	27-NOV-19 06.58.00.000000000 AM	27-NOV-19 07.08.00.000000000 AM	Cancelled by driver

PaymentMethod:



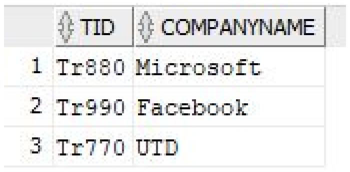
	TID	CARDNO	CVV	EXPIRYDATE	ACCTYPE	CARDTYPE	BILLINGADD
1	Tr550	7353574345735730	550	20-FEB-01	Savings	MasterCard	7835 McCallum Blvd
2	Tr660	7365374257325290	660	21-MAY-12	Savings	Visa	6362 Preston Road
3	Tr770	2804475628645990	770	22-JAN-03	Checking	Visa	5234 Campbell Road
4	Tr880	5839295719405570	880	23-APR-04	Savings	MasterCard	5284 McWell Blvd
5	Tr990	7530572357013640	990	24-JAN-06	Checking	MasterCard	6495 Coit Road

PersonalPayment:



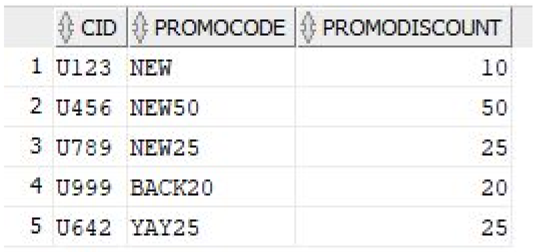
	TID	NAMEONCARD
1	Tr550	John Smith
2	Tr660	Harry Clinton
3	Tr770	William Shakespeare

BusinessPayment:



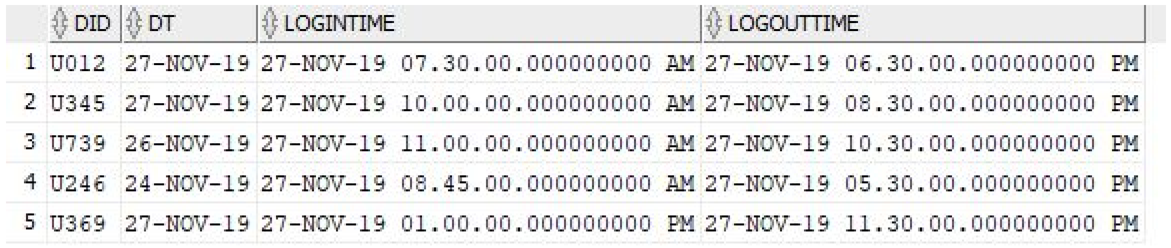
	TID	COMPANYNAME
1	Tr880	Microsoft
2	Tr990	Facebook
3	Tr770	UTD

Offers:



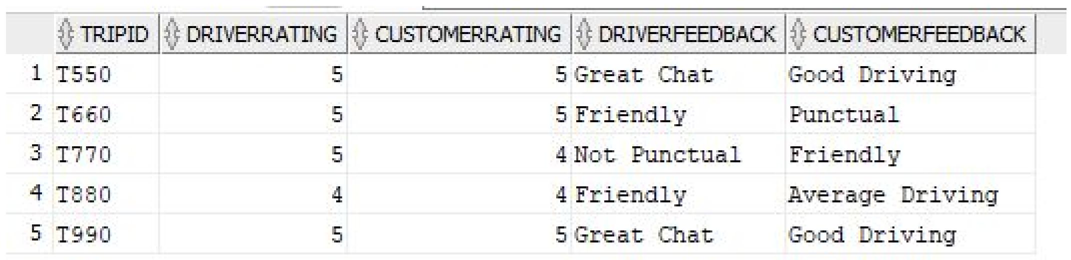
	CID	PROMOCODE	PROMODISCOUNT
1	U123	NEW	10
2	U456	NEW50	50
3	U789	NEW25	25
4	U999	BACK20	20
5	U642	YAY25	25

Shift:



	DID	DT	LOGINTIME	LOGOUTTIME
1	U012	27-NOV-19	27-NOV-19 07.30.00.000000000 AM	27-NOV-19 06.30.00.000000000 PM
2	U345	27-NOV-19	27-NOV-19 10.00.00.000000000 AM	27-NOV-19 08.30.00.000000000 PM
3	U739	26-NOV-19	27-NOV-19 11.00.00.000000000 AM	27-NOV-19 10.30.00.000000000 PM
4	U246	24-NOV-19	27-NOV-19 08.45.00.000000000 AM	27-NOV-19 05.30.00.000000000 PM
5	U369	27-NOV-19	27-NOV-19 01.00.00.000000000 PM	27-NOV-19 11.30.00.000000000 PM

Rating:



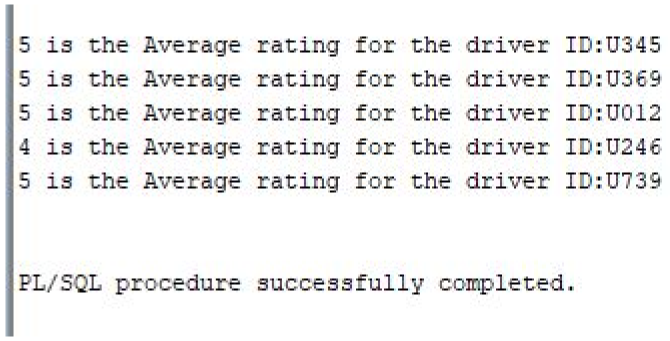
	TRIPID	DRIVERRATING	CUSTOMERRATING	DRIVERFEEDBACK	CUSTOMERFEEDBACK
1	T550	5	5	Great Chat	Good Driving
2	T660	5	5	Friendly	Punctual
3	T770	5	4	Not Punctual	Friendly
4	T880	4	4	Friendly	Average Driving
5	T990	5	5	Great Chat	Good Driving

- **Snapshots of running stored procedures with output and code created on Oracle SQL Developer**

1. **Stored Procedure to Calculate Average Ratings of all Drivers:**

```
create or replace PROCEDURE Average_Rating AS
CURSOR DrivRating IS SELECT AVG(R.DriverRating) as AvgRating, T.DID FROM
TripRequests T, Rating R WHERE T.TripID=R.TripID GROUP BY T.DID;
thisRating DrivRating%ROWTYPE;
BEGIN
OPEN DrivRating;
LOOP
FETCH DrivRating INTO thisRating;
EXIT WHEN (DrivRating%NOTFOUND);
dbms_output.put_line(thisRating.AvgRating || ' is the Average rating for the driver ID:'
|| thisRating.DID);
END LOOP;
CLOSE DrivRating;
END;
```

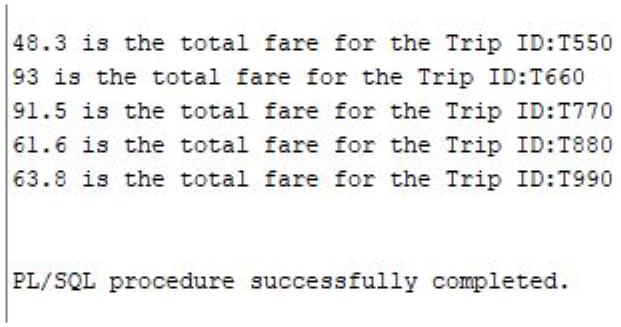
```
begin
Average_Rating;
End;
```



```
5 is the Average rating for the driver ID:U345
5 is the Average rating for the driver ID:U369
5 is the Average rating for the driver ID:U012
4 is the Average rating for the driver ID:U246
5 is the Average rating for the driver ID:U739


PL/SQL procedure successfully completed.
```

2. Stored Procedure to Calculate Total Fare for a given Ride:

```
create or replace PROCEDURE Calculate_Fare(Base_fare IN number, Service_Tax IN number,
Cost_per_mile IN number, Cost_per_min IN number) AS
CURSOR Trip_total_fare IS
SELECT
"A1". "TRIPID"    "TRIPID",
"A1"."DURATION"   "DURATION",:"A2"."DISTANCE"   "DISTANCE",
"A1"."SURGE"      "SURGE"
FROM
"TRIPREQUESTS"     "A2",
"COMPLETEDTRIPS"   "A1"
WHERE
"A2"."TRIPID" = "A1"."TRIPID";
thisTrip Trip_total_fare%rowtype;
thisTotalFare TripRequests.EstFare%TYPE;
BEGIN
OPEN Trip_total_fare;
LOOP
FETCH Trip_total_fare INTO thisTrip;
EXIT WHEN (Trip_total_fare%NOTFOUND);
thisTotalFare:= (Base_fare + Service_Tax + Cost_per_mile*thisTrip.distance +
Cost_per_min*thisTrip.duration )*(1 + thisTrip.Surge);
dbms_output.put_line(thisTotalFare || ' is the total fare for the Trip ID:' || thisTrip.TripID);
END LOOP;
CLOSE Trip_total_fare;
END;

Begin
Calculate_Fare(5,10,1,1);
End;
```



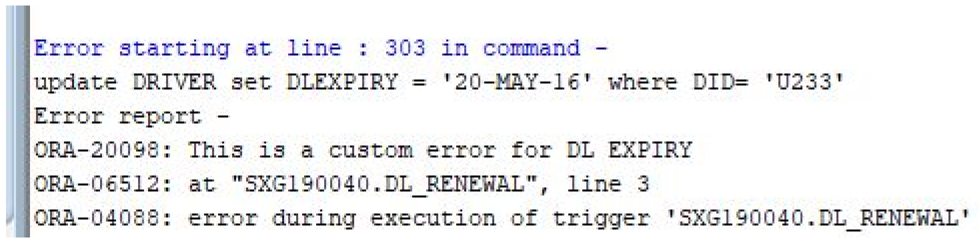
```
48.3 is the total fare for the Trip ID:T550
93 is the total fare for the Trip ID:T660
91.5 is the total fare for the Trip ID:T770
61.6 is the total fare for the Trip ID:T880
63.8 is the total fare for the Trip ID:T990


PL/SQL procedure successfully completed.
```

- **<u>Snapshots of running Triggers with errors fired and code on Oracle SQL Developer</u>**

1. Trigger to check that the Driver's License should not have expired:

create or replace TRIGGER DL_Renewal
before insert or update
on DRIVER for each row
Begin
if (:new.DLEXPIRY < sysdate) then
raise_application_error(-20098, 'This is a custom error for DL EXPIRY');
end if;
End;

<u>Query:</u> update DRIVER set DLEXPIRY = '20-MAY-16' where DID= 'U233';

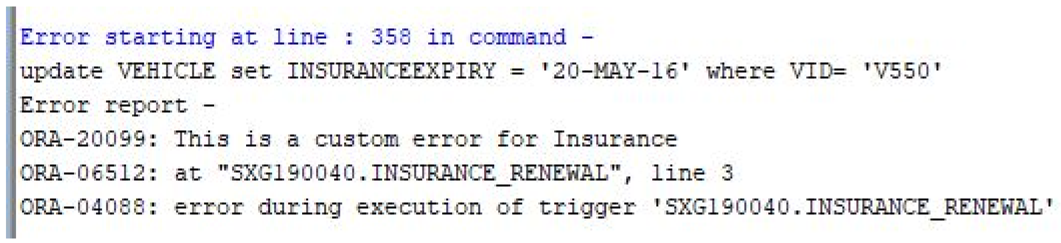


```
Error starting at line : 303 in command -
update DRIVER set DLEXPIRY = '20-MAY-16' where DID= 'U233'
Error report -
ORA-20098: This is a custom error for DL EXPIRY
ORA-06512: at "SXG190040.DL_RENEWAL", line 3
ORA-04088: error during execution of trigger 'SXG190040.DL_RENEWAL'
```

2. Trigger to check that the Insurance for the vehicle should not have expired:

create or replace TRIGGER Insurance_Renewal
before insert or update
on Vehicle for each row
Begin
if (:new.INSURANCEEXPIRY < sysdate) then
raise_application_error(-20099, 'This is a custom error for Insurance');
end if;
End;

<u>Query</u>: Update VEHICLE set INSURANCEEXPIRY = '20-MAY-16' where VID= 'V550';

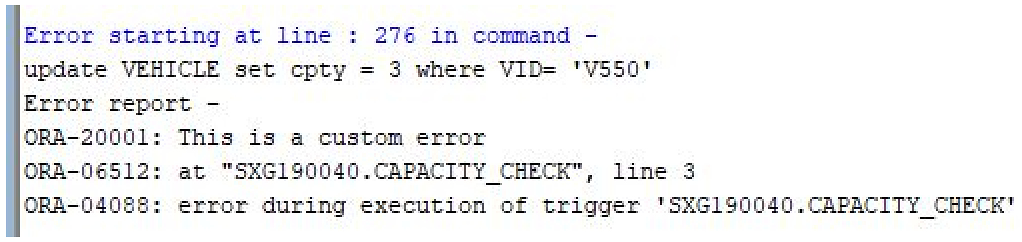


```
Error starting at line : 358 in command -
update VEHICLE set INSURANCEEXPIRY = '20-MAY-16' where VID= 'V550'
Error report -
ORA-20099: This is a custom error for Insurance
ORA-06512: at "SXG190040.INSURANCE_RENEWAL", line 3
ORA-04088: error during execution of trigger 'SXG190040.INSURANCE_RENEWAL'
```

3. Trigger to check that the capacity of a vehicle has to be greater than 4:

```
create or replace TRIGGER Capacity_Check
before update
on Vehicle for each row
Begin
if (:new.cpty < 4) then
raise_application_error( -20001, 'This is a custom error for Capacity');
end if;
End;
```

Query: update VEHICLE set cpty = 3 where VID= 'V550';



```
Error starting at line : 276 in command -
update VEHICLE set cpty = 3 where VID= 'V550'
Error report -
ORA-20001: This is a custom error
ORA-06512: at "SXG190040.CAPACITY_CHECK", line 3
ORA-04088: error during execution of trigger 'SXG190040.CAPACITY_CHECK'
```

APPENDIX CODE:

Creating Tables Query

```
-------------------------
--1)- Create User table
-------------------------

CREATE TABLE UberUSER
(
UberID      varchar(15)    NOT NULL,
FName       varchar(50)    NOT NULL,
LName       varchar(50)    NOT NULL,
  PhNo        int       NOT NULL,
  Email       varchar(50)    NOT NULL,
  Address      varchar(50)    NOT NULL,
  DOB         DATE       NOT NULL,

  PRIMARY KEY(UberID)
);

--------------------------
--2)- Create Customer table
--------------------------
CREATE TABLE Customer
(
  CID             varchar(15) NOT NULL,
  CustomerType      varchar(15) NOT NULL ,

  PRIMARY KEY(CID),
  FOREIGN KEY (CID) REFERENCES UberUser(UberID) ON DELETE CASCADE
);

DROP TABLE driver;

--3)- Create Driver table
----------------------
----------------------
CREATE TABLE Driver
(
  DID    varchar(15)     NOT NULL,
  SSN        int         NOT NULL,
  DLNo       varchar(50)     NOT NULL,
  DLExpiry    DATE       NOT NULL,

  PRIMARY KEY(DID),
  FOREIGN KEY (DID) REFERENCES UberUser(UberID) ON DELETE CASCADE
);

-----------------------
```

```
--4)- Create Vehicle table
-----------------------
CREATE TABLE Vehicle
(
  VID           varchar(15)    NOT NULL,
  DrID           varchar(50)    NOT NULL,
  ModelN         varchar(50)    NOT NULL,
  Color          varchar(20)    NOT NULL,
  ManufYear      int          NOT NULL,
  PurDate        DATE         NOT NULL,
  Active         varchar(18)       NOT NULL,
  Condition      varchar(15)    NOT NULL,
  Cpty            int       NOT NULL,
  InsuranceNo     varchar(15)    NOT NULL,
  InsuranceExpiry varchar(15)    NOT NULL,
  LastChecked     DATE        NOT NULL,

  PRIMARY KEY(VID),
  FOREIGN KEY (DrID) REFERENCES Driver(DID) ON DELETE CASCADE
);

-----------------------
--5)- Create TripRequests table
-----------------------
CREATE TABLE TripRequests
(
  TripID      varchar(15)    NOT NULL,
  CID         varchar(50)    NOT NULL,
  DID         varchar(50)    NOT NULL,
  TripType     varchar(50)    NOT NULL,
  PickupLoc    varchar(50)    NOT NULL,
  DropoffLoc   varchar(50)    NOT NULL,
  Distance      float      NOT NULL,
  EstFare      float      NOT NULL,
  TID         varchar(15)    NOT NULL,

  PRIMARY KEY(TripID),
  FOREIGN KEY (TID) REFERENCES PaymentMethod(TID) ON DELETE CASCADE,
  FOREIGN KEY (CID) REFERENCES Customer(CID) ON DELETE CASCADE,
  FOREIGN KEY (DID) REFERENCES Driver(DID)  ON DELETE CASCADE
);
------------------------
--6)- Create CompletedTrips table
------------------------

CREATE TABLE CompletedTrips
(
  TripID          varchar(15)    NOT NULL,
  DriverArrivedAt   TIMESTAMP         NOT NULL,
  PickupTime       TIMESTAMP         NOT NULL,
  DropoffTime      TIMESTAMP        NOT NULL,
  duration         int          NOT NULL,
  ActFare          float     NOT NULL,
  Tip              float     NOT NULL,
  Surge       float     NULL,

  PRIMARY KEY(TripID),
  FOREIGN KEY (TripID) REFERENCES TripRequests(TripID) ON DELETE CASCADE
);
------------------------
--7)- Create IncompleteTrips table
------------------------

CREATE TABLE IncompleteTrips
(
  TripID          varchar(15)    NOT NULL,
  BookingTime      TIMESTAMP                NOT NULL,
  CancelTime       TIMESTAMP         NOT NULL,
  Reason          varchar(30)          NOT NULL,

PRIMARY KEY(TripID),
FOREIGN KEY (TripID) REFERENCES TripRequests(TripID) ON DELETE CASCADE
);


----------------------
--8)- Create PaymentMethod table
----------------------
CREATE TABLE PaymentMethod
(
  TID         varchar(50)     NOT NULL,
  CardNo      int       NOT NULL,
  CVV         int       NOT NULL,
  ExpiryDate    DATE          NOT NULL,
  AccType     varchar(50)    NOT NULL,
  CardType    varchar(50)    NOT NULL,
  BillingAdd   varchar(50)    NOT NULL,

  PRIMARY KEY(TID)
);


----------------------
--9)- Create PersonalPayment table
----------------------
CREATE TABLE PersonalPayment
(
  TID         varchar(15)    NOT NULL,
  NameOnCard    varchar(50)    NOT NULL,
  PRIMARY KEY(TID),
  FOREIGN KEY (TID) REFERENCES PaymentMethod(TID) ON DELETE CASCADE
);

-------------------------
--10)- Create BusinessPayment table
-------------------------
CREATE TABLE BusinessPayment
(
  TID          varchar(15)    NOT NULL,
  CompanyName   varchar(50)    NOT NULL,

  PRIMARY KEY(TID),
  FOREIGN KEY (TID) REFERENCES PaymentMethod(TID) ON DELETE CASCADE
);

-------------------------
--11)- Create Rating table
-------------------------

CREATE TABLE Rating
(
  TripID             varchar(15)    NOT NULL,
  DriverRating      int        NOT NULL,
  CustomerRating    int        NOT NULL,
  DriverFeedback    varchar(15)    NOT NULL,
  CustomerFeedback  varchar(15)    NOT NULL,

  PRIMARY KEY(TripID),
  FOREIGN KEY (TripID) REFERENCES CompletedTrips(TripID) ON DELETE CASCADE
);

-------------------------
--12)- Create Shift table
-------------------------


CREATE TABLE Shift
(
  DID          varchar(15)    NOT NULL,
  DT         DATE          NOT NULL,
  LoginTime     TIMESTAMP                  NOT NULL,
  LogoutTime    TIMESTAMP         NOT NULL,

  PRIMARY KEY(DID, DT),
  FOREIGN KEY (DID) REFERENCES Driver(DID) ON DELETE CASCADE
);

-------------------------
--13)- Create Offers table
```

```
-------------------------

CREATE TABLE Offers
(
  CID           varchar(15)    NOT NULL,
  PromoCode      varchar(15)    NOT NULL,
  PromoDiscount  float      NOT NULL,

  PRIMARY KEY(CID),
  FOREIGN KEY (CID) REFERENCES Customer(CID) ON DELETE CASCADE
);
```