Database Design | Final Report

Project Name: UBER DATABASE DESIGN

Submission Date: 12/12/2023

Submitted By: Group 26

Group Members :

1. **Name:** Bhanu Marisa **Student Id:** 02083895
2. **Name:** Teja Krishna Reddy Yarram **Student Id:** 02083419
3. **Name:** Shaik Sajid Ali **Student Id:** 02081970

Database Design | Final Report

Project Description :

A lot of real-time data is generated within Uber data centers. This data originates in different sources such as end-user applications (driver/rider) or the backend microservices. Some of this data consists of application or system logs continuously emitted as part of day-to-day operations. Uber database management system is developed to focus on helping transportation service passengers by allowing them to hail rides at some charges depending on the distance covered. The database becomes a key component by helping in keeping track of customer data, providing Uber location via GPS, doing payments, allocating jobs to employees, generating payment receipts, controlling revenues, creating an even login in new customers and management staff in a more secure, convenient and user-friendly manner than the manual systems. Using the database we can organize the data in an effective way which helps in accessing it quickly.

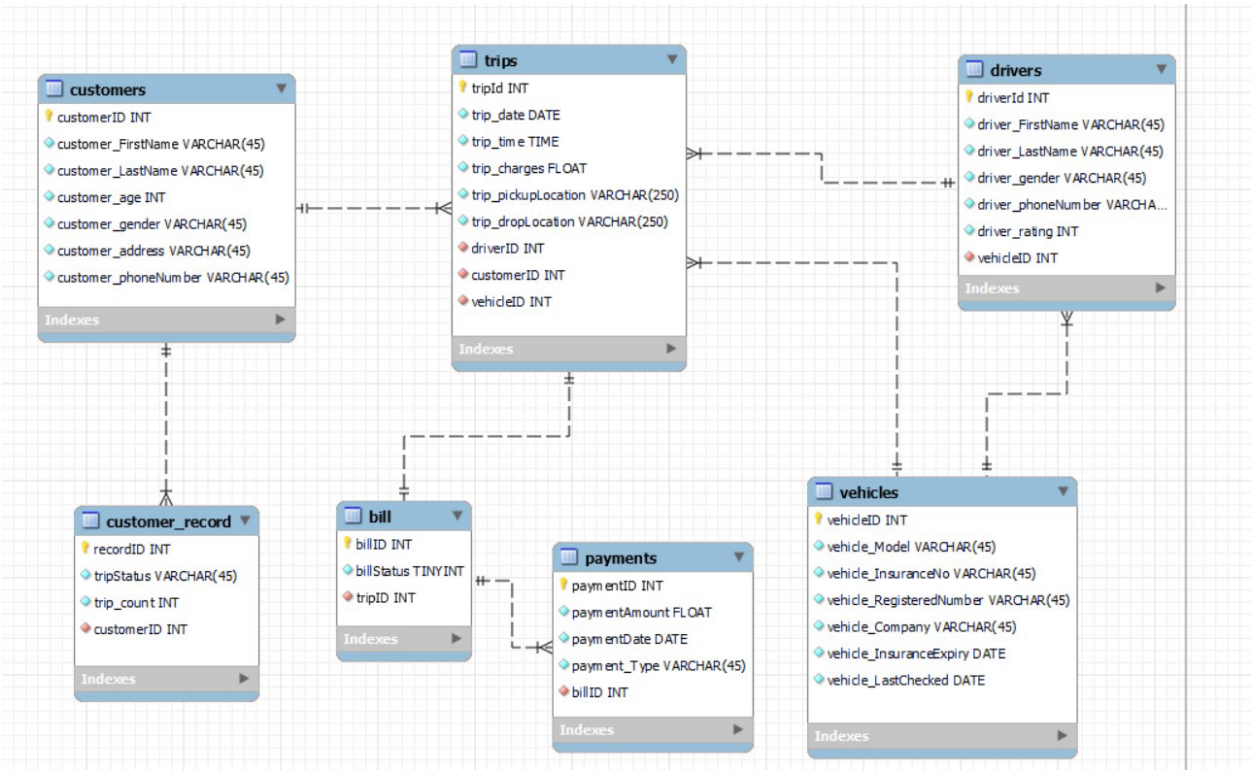
Database Requirements:

The following tables contains information for the database collection:

1. Customer Information Table
2. Driver Information Table
3. Trip Details Table
4. Vehicle Information Table
5. Customer Record Table
6. Billing Table
7. Payment Table

Database Design | Final Report

Entity-Relationship Diagram :



There are 7 tables in our project namely,

1.Customers: In this table the customers data is stored such as customerID, customer_FirstName, customer_LastName, customer_age, customer_gender, customer_address, customer_phoneNumber. customerID is the primary key in the table.

2.Trips: In this table the customer trips will be stored. The attributes in this table are trip_Id, trip_date, trip_time, trip_charges, trip_pickupLocation, pickup_dropLocation. trip_Id is the primary key in this table.

3.Drivers: In this table the drivers data is stored. The attributes in this table are driver_Id, driver_Firstname, driver_LastName, driver_gender, driver_phoneNumber, phone_rating. driver-Id is the primary key.

4.Customer_record: In this table the customer trips record will be stored. record_Id is the primarykey in this table. tripStatus, trip_count are the attributes in this table.

5.Bill: In this the bill related details are stored in the table. billID is the primary key and billStatus attribute is also present.

6.Payments: In this table the customer payment method details are stored. payment_ID, ,payment_Amount, payment_Date, payment_Type are the attributes in the table.

7.Vehicles: In this table the data related to vehicle is stored. vehicleID is the primary key in this table. Vehicle_Model, ,vehicle_InsuaranceNo, vehicle_RegisteredNumber, vehicle_Company, vehicle_InsuranceExpiry, vehicle_LastChecked are the attributes in this table.

Relational Schema :

The relationship between all the tables is Non-Identifying as there are no weak entities. The relationship between each table is explained below

1.customers table and trips table:

The relationship between these two tables is 1:n because a single customer can have multiple trips in a single day and the customerID is the foreign key in the trips table.

2.customers table and customers_record table:

The relationship between these two tables is 1:n because customers_record have past trips of the customer, so it will be more than one and CustomerID is the foreign key in the customer_record table.

3.trips table and bill table:

The relationship between these two tables is 1:1 because each trip can have only one bill generated and tripID is the primary key in bills table.

4.bills table and payments table:

The relationship between these two tables is 1:n because a single bill can be paid through different methods, such as card, cash, wallet and billId is the foreign key in the payments table.

5.vehicles table and trips table:

The relation ship between these two tables is 1:n as a single vehicle can do multiple trips in a single day and vehicleID is the foreign key in trips table.

6.drivers table and trips table:

The relationship between these two tables is 1:n because a driver can do n number trips in a single day and driverID is the foreign key in the trips table.

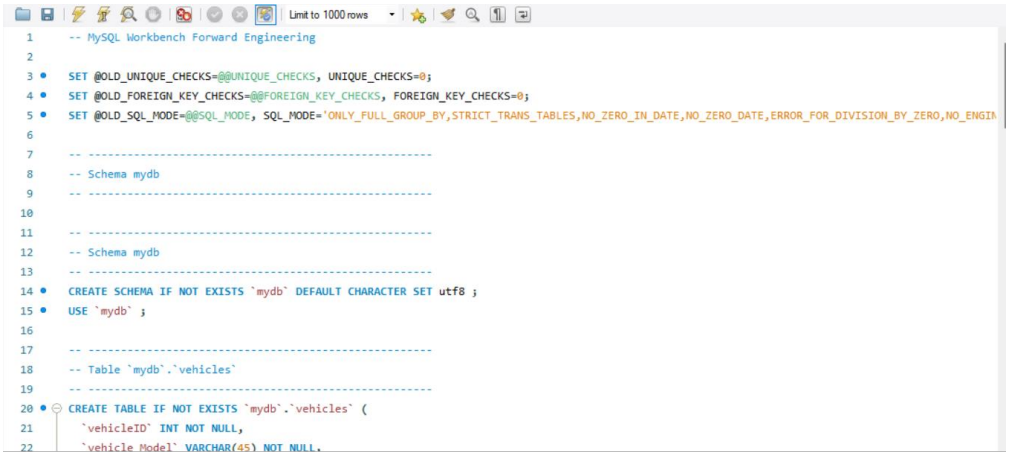
7.vehicles table and driver table:

The relationship between these two tables is 1:n because a single vehicle can be driven by n number of drivers and vehicleId is the foreign key in the drivers table.

Database Design | Final Report

Database Implementation:

Using Forward Engineering technique, the below schema is generated.



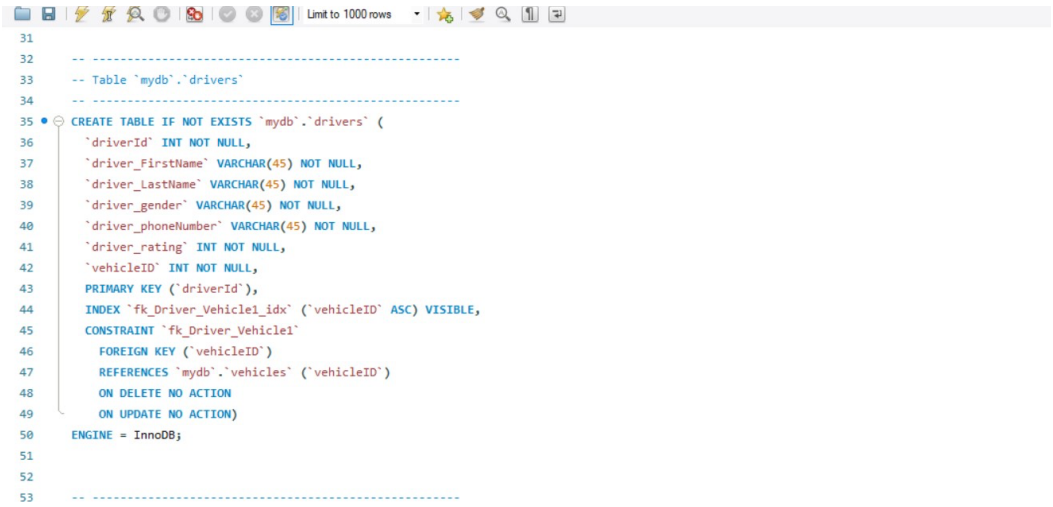
```
-- MySQL Workbench Forward Engineering

SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_ENGIN

-- -----------------------------------------------------
-- Schema mydb
-- -----------------------------------------------------

-- -----------------------------------------------------
-- Schema mydb
-- -----------------------------------------------------
CREATE SCHEMA IF NOT EXISTS `mydb` DEFAULT CHARACTER SET utf8 ;
USE `mydb` ;

-- -----------------------------------------------------
-- Table `mydb`.`vehicles`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`vehicles` (
  `vehicleID` INT NOT NULL,
  `vehicle_Model` VARCHAR(45) NOT NULL,
```



```

-- -----------------------------------------------------
-- Table `mydb`.`drivers`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`drivers` (
  `driverId` INT NOT NULL,
  `driver_FirstName` VARCHAR(45) NOT NULL,
  `driver_LastName` VARCHAR(45) NOT NULL,
  `driver_gender` VARCHAR(45) NOT NULL,
  `driver_phoneNumber` VARCHAR(45) NOT NULL,
  `driver_rating` INT NOT NULL,
  `vehicleID` INT NOT NULL,
  PRIMARY KEY (`driverId`),
  INDEX `fk_Driver_Vehicle1_idx` (`vehicleID` ASC) VISIBLE,
  CONSTRAINT `fk_Driver_Vehicle1`
    FOREIGN KEY (`vehicleID`)
    REFERENCES `mydb`.`vehicles` (`vehicleID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


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

```
51
52
53      -- -----------------------------------------------------
54      -- Table `mydb`.`customers`
55      -- -----------------------------------------------------
56  •   CREATE TABLE IF NOT EXISTS `mydb`.`customers` (
57        `customerID` INT NOT NULL,
58        `customer_FirstName` VARCHAR(45) NOT NULL,
59        `customer_LastName` VARCHAR(45) NOT NULL,
60        `customer_age` INT NOT NULL,
61        `customer_gender` VARCHAR(45) NOT NULL,
62        `customer_address` VARCHAR(45) NOT NULL,
63        `customer_phoneNumber` VARCHAR(45) NOT NULL,
64        PRIMARY KEY (`customerID`))
65      ENGINE = InnoDB;
66
67
```

```
71  •   CREATE TABLE IF NOT EXISTS `mydb`.`trips` (
72        `tripId` INT NOT NULL,
73        `trip_date` DATE NOT NULL,
74        `trip_time` TIME NOT NULL,
75        `trip_charges` FLOAT NOT NULL,
76        `trip_pickupLocation` VARCHAR(250) NOT NULL,
77        `trip_dropLocation` VARCHAR(250) NOT NULL,
78        `driverID` INT NOT NULL,
79        `customerID` INT NOT NULL,
80        `vehicleID` INT NOT NULL,
81        PRIMARY KEY (`tripId`),
82        INDEX `fk_Trip_Driver_idx` (`driverID` ASC) VISIBLE,
83        INDEX `fk_Trip_Customer1_idx` (`customerID` ASC) VISIBLE,
84        INDEX `fk_trips_vehicles1_idx` (`vehicleID` ASC) VISIBLE,
85        CONSTRAINT `fk_Trip_Driver`
86          FOREIGN KEY (`driverID`)
87          REFERENCES `mydb`.`drivers` (`driverId`)
88          ON DELETE NO ACTION
89          ON UPDATE NO ACTION,
90        CONSTRAINT `fk_Trip_Customer1`
91          FOREIGN KEY (`customerID`)
92          REFERENCES `mydb`.`customers` (`customerID`)
93          ON DELETE NO ACTION
94          ON UPDATE NO ACTION,
95        CONSTRAINT `fk_trips_vehicles1`
96          FOREIGN KEY (`vehicleID`)
97          REFERENCES `mydb`.`vehicles` (`vehicleID`)
98          ON DELETE NO ACTION
99          ON UPDATE NO ACTION)
100     ENGINE = InnoDB;
101
```

```
106 •   CREATE TABLE IF NOT EXISTS `mydb`.`customer_record` (
107       `recordID` INT NOT NULL,
108       `tripStatus` VARCHAR(45) NOT NULL,
109       `trip_count` INT NOT NULL,
110       `customerID` INT NOT NULL,
111       PRIMARY KEY (`recordID`),
112       INDEX `fk_customer_record_Customer1_idx` (`customerID` ASC) VISIBLE,
113       CONSTRAINT `fk_customer_record_Customer1`
114         FOREIGN KEY (`customerID`)
115         REFERENCES `mydb`.`customers` (`customerID`)
116         ON DELETE NO ACTION
117         ON UPDATE NO ACTION)
118     ENGINE = InnoDB;
119
120
121     -- -----------------------------------------------------
122     -- Table `mydb`.`bill`
123     -- -----------------------------------------------------
124 •   CREATE TABLE IF NOT EXISTS `mydb`.`bill` (
125       `billID` INT NOT NULL,
126       `billStatus` TINYINT NOT NULL,
127       `tripID` INT NOT NULL,
128       PRIMARY KEY (`billID`),
129       INDEX `fk_bill_Trip1_idx` (`tripID` ASC) VISIBLE,
130       CONSTRAINT `fk_bill_Trip1`
131         FOREIGN KEY (`tripID`)
132         REFERENCES `mydb`.`trips` (`tripId`)
133         ON DELETE NO ACTION
134         ON UPDATE NO ACTION)
135     ENGINE = InnoDB;
136
```

```
136
137
138     -- -----------------------------------------------------
139     -- Table `mydb`.`payments`
140     -- -----------------------------------------------------
141 ●   CREATE TABLE IF NOT EXISTS `mydb`.`payments` (
142       `paymentID` INT NOT NULL,
143       `paymentAmount` FLOAT NOT NULL,
144       `paymentDate` DATE NOT NULL,
145       `payment_Type` VARCHAR(45) NOT NULL,
146       `billID` INT NOT NULL,
147       PRIMARY KEY (`paymentID`),
148       INDEX `fk_Payment_bill1_idx` (`billID` ASC) VISIBLE,
149       CONSTRAINT `fk_Payment_bill1`
150         FOREIGN KEY (`billID`)
151         REFERENCES `mydb`.`bill` (`billID`)
152         ON DELETE NO ACTION
153         ON UPDATE NO ACTION)
154     ENGINE = InnoDB;
155
156
157 ●   SET SQL_MODE=@OLD_SQL_MODE;
158 ●   SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
159 ●   SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
160
```

Inserting data into Tables:

Inserting data into customers table:

```
1 ●   use `mydb`;
2
3 ●   INSERT INTO customers (customerID, customer_FirstName, customer_LastName, customer_age, customer_gender, customer_address, customer_phoneNumber
4     VALUES ('100', 'Jon', 'Doe', '19', 'Male', '123 street', '456788876');
5 ●   INSERT INTO customers (customerID, customer_FirstName, customer_LastName, customer_age, customer_gender, customer_address, customer_phoneNumber
6      VALUES ('101', 'Johny', 'Salvatore', '19', 'Male', '123 street', '456788876');
7 ●   INSERT INTO customers (customerID, customer_FirstName, customer_LastName, customer_age, customer_gender, customer_address, customer_phoneNumber
8      VALUES ('102', 'Sidd', 'Salvatore', '16', 'Male', '123 ,plays garden', '489788876');
9 ●   INSERT INTO customers (customerID, customer_FirstName, customer_LastName, customer_age, customer_gender, customer_address, customer_phoneNumber
10     VALUES ('103', 'Johny', 'rakesh', '20', 'Female', 'graphic era', '696788876');
11 ●  INSERT INTO customers (customerID, customer_FirstName, customer_LastName, customer_age, customer_gender, customer_address, customer_phoneNumber
12     VALUES ('104', 'subha', 'Salvatore', '19', 'Male', '123 street', '456788876');
13 ●  INSERT INTO customers (customerID, customer_FirstName, customer_LastName, customer_age, customer_gender, customer_address, customer_phoneNumber
14     VALUES ('105', 'suja', 'Salvatore', '19', 'Female', '123 street', '456778876');
15 ●  INSERT INTO customers (customerID, customer_FirstName, customer_LastName, customer_age, customer_gender, customer_address, customer_phoneNumber
16     VALUES ('106', 'Johny', 'Salvatore', '19', 'Male', '123 street', '456788876');
17 ●  INSERT INTO customers (customerID, customer_FirstName, customer_LastName, customer_age, customer_gender, customer_address, customer_phoneNumber
18     VALUES ('107', 'manav', 'Salvatore', '41', 'Male', '123 street', '676788876');
19 ●  INSERT INTO customers (customerID, customer_FirstName, customer_LastName, customer_age, customer_gender, customer_address, customer_phoneNumber
20     VALUES ('108', 'doe', 'Coven', '19', 'Male', '1234 streets', '456756876');
21 ●  INSERT INTO customers (customerID, customer_FirstName, customer_LastName, customer_age, customer_gender, customer_address, customer_phoneNumber
22     VALUES ('109', 'Joy', 'Tribbiani', '19', 'Male', '123 street', '456788876');
23 ●  INSERT INTO customers (customerID, customer_FirstName, customer_LastName, customer_age, customer_gender, customer_address, customer_phoneNumber
24     VALUES ('110', 'Jsed', 'Coven', '19', 'Male', '123 street', '453488876');
```

Data populating in customers table:

```
81
82 •   select * from customers ;
83
84
85
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

customerID	customer_FirstName	customer_LastName	customer_age	customer_gender	customer_address	customer_phoneNumber
100	Jon	Doe	19	Male	123 street	456788876
101	Johny	Salvatore	19	Male	123 street	456788876
102	Sidd	Salvatore	16	Male	123 ,plays garden	489788876
103	Johny	rakesh	20	Female	graphic era	696788876
104	subha	Salvatore	19	Male	123 street	456788876
105	suja	Salvatore	19	Female	123 street	456778876
106	Johny	Salvatore	19	Male	123 street	456788876
107	manav	Salvatore	41	Male	123 street	676788876
108	doe	Coven	19	Male	1234 streets	456756876
109	Joy	Tribbiani	19	Male	123 street	456788876
110	Jsed	Coven	19	Male	123 street	453488876
111	Virat	Kohli	33	Male	859 new street	0000000000
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Inserting data into vehicles table:

```
28
29 •  INSERT INTO vehicles (vehicleID, vehicle_Model, vehicle_InsuranceNo, vehicle_RegisteredNumber, vehicle_Company, vehicle_InsuranceExpiry, vehicl
30     VALUES ('1234', 'AUDI', '123445', 'KK', 'Audi', '2003-11-11', '2003-11-11');
31 •  INSERT INTO vehicles (vehicleID, vehicle_Model, vehicle_InsuranceNo, vehicle_RegisteredNumber, vehicle_Company, vehicle_InsuranceExpiry, vehicl
32     VALUES ('12456734', 'MG', '123423445', 'KK', 'Audi', '2003-11-11', '2003-11-11');
33 •  INSERT INTO vehicles (vehicleID, vehicle_Model, vehicle_InsuranceNo, vehicle_RegisteredNumber, vehicle_Company, vehicle_InsuranceExpiry, vehicl
34     VALUES ('1254734', 'Maruti', '123345445', 'KK', 'Audi', '2003-11-11', '2003-11-11');
35 •  INSERT INTO vehicles (vehicleID, vehicle_Model, vehicle_InsuranceNo, vehicle_RegisteredNumber, vehicle_Company, vehicle_InsuranceExpiry, vehicl
36     VALUES ('1245734', 'Suzuki', '123453445', 'KK', 'Audi', '2003-11-11', '2003-11-11');
37 •  INSERT INTO vehicles (vehicleID, vehicle_Model, vehicle_InsuranceNo, vehicle_RegisteredNumber, vehicle_Company, vehicle_InsuranceExpiry, vehicl
38     VALUES ('1225334', 'Tata', '12346845', 'KK', 'Audi', '2001-11-11', '2008-11-11');
39 •  INSERT INTO vehicles (vehicleID, vehicle_Model, vehicle_InsuranceNo, vehicle_RegisteredNumber, vehicle_Company, vehicle_InsuranceExpiry, vehicl
40     VALUES ('123674', 'AUDI', '12345683445', 'KK', 'Audi', '2001-11-11', '2007-11-11');
41
```

Data populating in vehicles table:

```
83
84 •  select * from vehicles ;
85
86
87
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

vehicleID	vehicle_Model	vehicle_InsuranceNo	vehicle_RegisteredNumber	vehicle_Company	vehicle_InsuranceExpiry	vehicle_LastChecked
1234	AUDI	123445	KK	Audi	2003-11-11	2003-11-11
1267	BENZ	12348945	KK	Benz	2008-05-11	2008-06-11
12987	Honda	987564	KK	Civic	2010-05-09	2010-08-12
123674	AUDI	12345683445	KK	Audi	2001-11-11	2007-11-11
1225334	Tata	12346845	KK	Audi	2001-11-11	2008-11-11
1245734	Suzuki	123453445	KK	Audi	2003-11-11	2003-11-11
1254734	Maruti	123345445	KK	Audi	2003-11-11	2003-11-11
12456734	MG	123423445	KK	Audi	2003-11-11	2003-11-11
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Inserting data into drivers table:

```
44
45 •  INSERT INTO drivers(driverID, driver_FirstName, driver_LastName, driver_gender, driver_phoneNumber, driver_rating, vehicleID)
46     VALUES ('200', 'Jone', 'Rock', 'Male', '9685743210', '3', '123674');
47 •  INSERT INTO drivers (driverID, driver_FirstName, driver_LastName, driver_gender, driver_phoneNumber, driver_rating, vehicleID)
48     VALUES ('201','John', 'Doe', 'Male', '4567888762', '3', '1254734');
49 •  INSERT INTO drivers (driverID, driver_FirstName, driver_LastName, driver_gender, driver_phoneNumber, driver_rating, vehicleID)
50     VALUES ('202','Sidd', 'Sriram', 'Male', '4897888762','5', '1254734');
51 •  INSERT INTO drivers (driverID, driver_FirstName, driver_LastName, driver_gender, driver_phoneNumber, driver_rating, vehicleID)
52     VALUES ('203','Elena', 'Gilbert', 'Female', '6967288876', '2', '123674');
53 •  INSERT INTO drivers (driverID, driver_FirstName, driver_LastName, driver_gender, driver_phoneNumber, driver_rating, vehicleID)
54     VALUES ('204', 'Joey', 'Tribbiani','Male', '4567882876', '1', '1254734');
55 •  INSERT INTO drivers (driverID, driver_FirstName, driver_LastName, driver_gender, driver_phoneNumber, driver_rating, vehicleID)
56     VALUES ('205','Phoebe', 'Buffay', 'Female', '4562778876','2', '123674');
57
58
```

Data populating in drivers table:

```
85
86 •  select * from drivers ;
87
88
89
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

driverId	driver_FirstName	driver_LastName	driver_gender	driver_phoneNumber	driver_rating	vehicleID
200	Jone	Rock	Male	9685743210	3	123674
201	John	Doe	Male	4567888762	3	1254734
202	Sidd	Sriram	Male	4897888762	5	1254734
203	Elena	Gilbert	Female	6967288876	2	123674
204	Joey	Tribbiani	Male	4567882876	1	1254734
205	Phoebe	Buffay	Female	4562778876	2	123674
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Inserting data into trips table:

```
62 •  INSERT INTO trips(tripID,trip_date, trip_time, trip_charges, trip_pickupLocation, trip_dropLocation,driverID,customerID,vehicleID)
63    VALUES ('300', '2002-11-11', '15:50:00', '700', 'NewBedford', 'FallRiver','201','102','1234');
64 •  INSERT INTO trips(tripID,trip_date, trip_time, trip_charges, trip_pickupLocation, trip_dropLocation,driverID,customerID,vehicleID)
65    VALUES ('301', '2003-09-11', '16:50:00', '500', 'FallRiver', 'NewBedford','202','103','1234');
66 •  INSERT INTO trips(tripID,trip_date, trip_time, trip_charges, trip_pickupLocation, trip_dropLocation,driverID,customerID,vehicleID)
67    VALUES ('302', '2005-10-11', '17:50:00', '550', 'NewBedford', 'UMass','205','105','123674');
68 •  INSERT INTO trips(tripID,trip_date, trip_time, trip_charges, trip_pickupLocation, trip_dropLocation,driverID,customerID,vehicleID)
69    VALUES ('303', '2006-01-11', '18:50:00', '600', 'UMass', 'FallRiver','203','110','1234');
70 •  INSERT INTO trips(tripID,trip_date, trip_time, trip_charges, trip_pickupLocation, trip_dropLocation,driverID,customerID,vehicleID)
71    VALUES ('304', '2008-07-10', '19:26:00', '650', 'UMass', 'NewBedford','200','108','1254734');
72 •  INSERT INTO trips(tripID,trip_date, trip_time, trip_charges, trip_pickupLocation, trip_dropLocation,driverID,customerID,vehicleID)
73    VALUES ('305', '2009-11-11', '20:32:00', '250', 'FallRiver', 'UMass','202','106','123674');
74 •  INSERT INTO trips(tripID,trip_date, trip_time, trip_charges, trip_pickupLocation, trip_dropLocation,driverID,customerID,vehicleID)
75    VALUES ('306', '2010-11-11', '22:52:00', '350', 'NewBedford', 'FallRiver','205','106','1234');
```

Data populating in trips table:

```
88
89 •  select * from trips ;
90
91
92
```

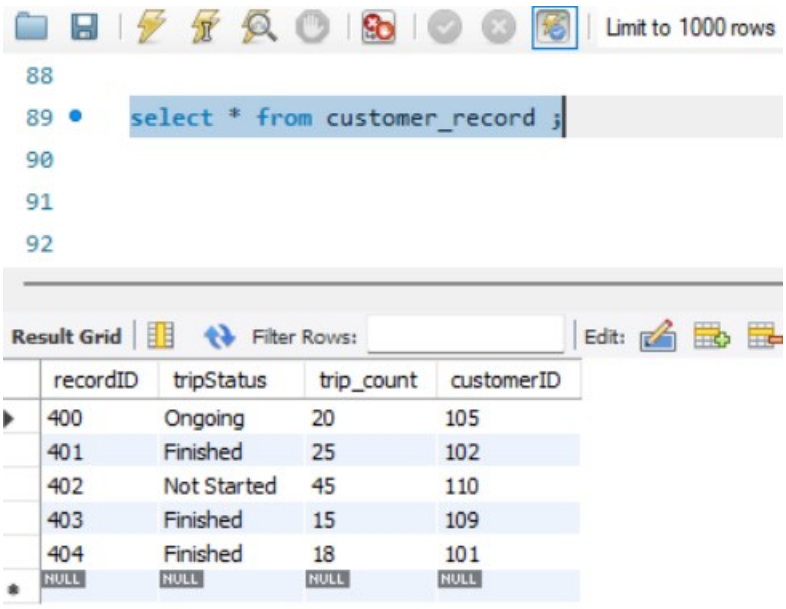
Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

tripId	trip_date	trip_time	trip_charges	trip_pickupLocation	trip_dropLocation	driverID	customerID	vehicleID
300	2002-11-11	15:50:00	7000	NewBedford	FallRiver	201	102	1234
301	2003-09-11	16:50:00	500	FallRiver	NewBedford	202	103	1234
302	2005-10-11	17:50:00	550	NewBedford	UMass	205	105	123674
303	2006-01-11	18:50:00	600	UMass	FallRiver	203	110	1234
304	2008-07-10	19:26:00	650	UMass	NewBedford	200	108	1254734
305	2009-11-11	20:32:00	250	FallRiver	UMass	202	106	123674
306	2010-11-11	22:52:00	350	NewBedford	FallRiver	205	106	1234
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Inserting data into customer_record table:

```
80
81 •  INSERT INTO customer_record(recordID, tripStatus, trip_count, customerID) VALUES ('400', 'Ongoing' , '20','105');
82 •  INSERT INTO customer_record(recordID, tripStatus, trip_count, customerID) VALUES ('401', 'Finished' , '25','102');
83 •  INSERT INTO customer_record(recordID, tripStatus, trip_count, customerID) VALUES ('402', 'Not Started' , '45','110');
84 •  INSERT INTO customer_record(recordID, tripStatus, trip_count, customerID) VALUES ('403', 'Finished' , '15','109');
85 •  INSERT INTO customer_record(recordID, tripStatus, trip_count, customerID) VALUES ('404', 'Finished' , '18','101');
86
```

Data populating in customer_record table:

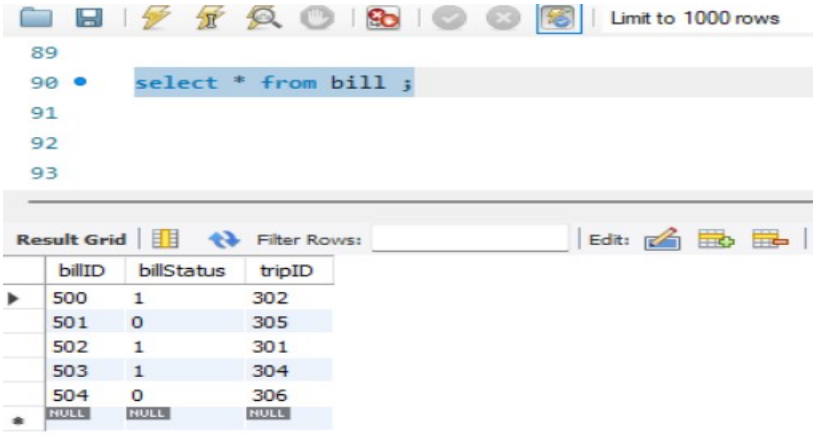


recordID	tripStatus	trip_count	customerID
400	Ongoing	20	105
401	Finished	25	102
402	Not Started	45	110
403	Finished	15	109
404	Finished	18	101
NULL	NULL	NULL	NULL

Inserting data into bill table:

```
89  INSERT INTO bill(billID, billStatus, tripID) VALUES ('500', '1', '302');
90  INSERT INTO bill(billID, billStatus, tripID) VALUES ('501', '0', '305');
91  INSERT INTO bill(billID, billStatus, tripID) VALUES ('502', '1', '301');
92  INSERT INTO bill(billID, billStatus, tripID) VALUES ('503', '1', '304');
93  INSERT INTO bill(billID, billStatus, tripID) VALUES ('504', '0', '306');
94
```

Data populating in bill table:

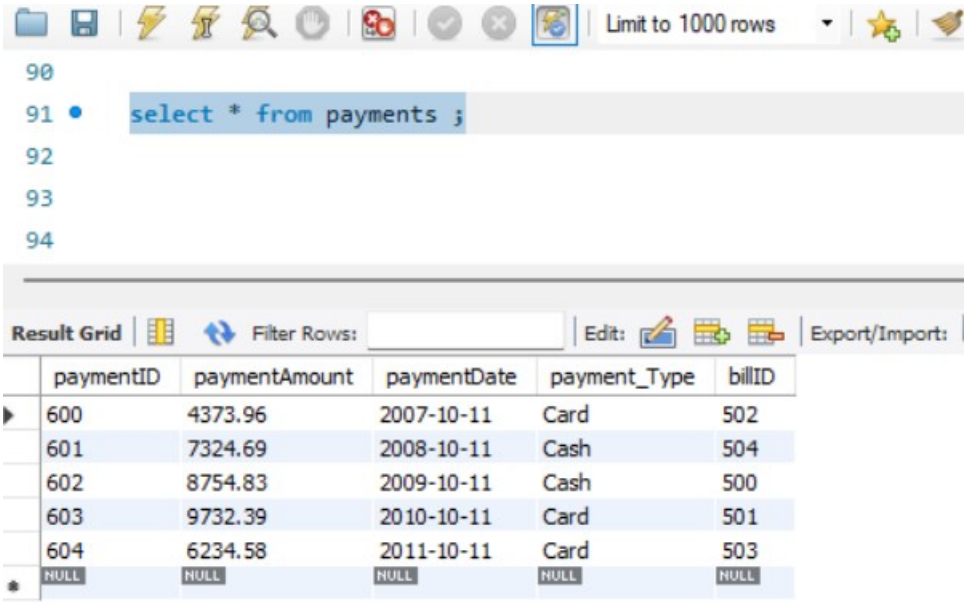


billID	billStatus	tripID
500	1	302
501	0	305
502	1	301
503	1	304
504	0	306
NULL	NULL	NULL

Inserting data into payments table:

```
INSERT INTO payments (paymentID, paymentAmount, paymentDate, payment_Type, billID)
VALUES ('600', '4373.96', '2007-10-11', 'Card','502');
INSERT INTO payments (paymentID, paymentAmount, paymentDate, payment_Type, billID)
VALUES ('601', '7324.69', '2008-10-11', 'Cash','504');
INSERT INTO payments (paymentID, paymentAmount, paymentDate, payment_Type, billID)
VALUES ('602', '8754.83', '2009-10-11', 'Cash','500');
INSERT INTO payments (paymentID, paymentAmount, paymentDate, payment_Type, billID)
VALUES ('603', '9732.39', '2010-10-11', 'Card','501');
INSERT INTO payments (paymentID, paymentAmount, paymentDate, payment_Type, billID)
VALUES ('604', '6234.58', '2011-10-11', 'Card','503');
```

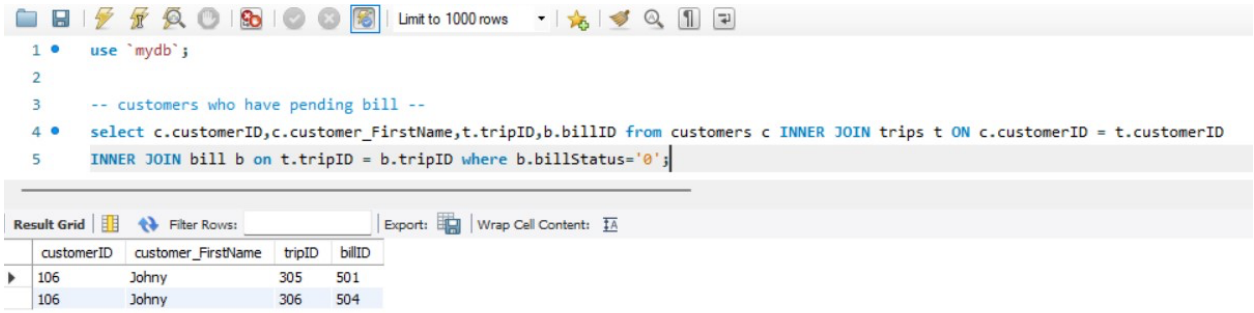
Data populating in payments table:



paymentID	paymentAmount	paymentDate	payment_Type	billID
600	4373.96	2007-10-11	Card	502
601	7324.69	2008-10-11	Cash	504
602	8754.83	2009-10-11	Cash	500
603	9732.39	2010-10-11	Card	501
604	6234.58	2011-10-11	Card	503
NULL	NULL	NULL	NULL	NULL

Querying database:

Query1: In this query we retrieve customers first name and customer Id whose bill is pending.



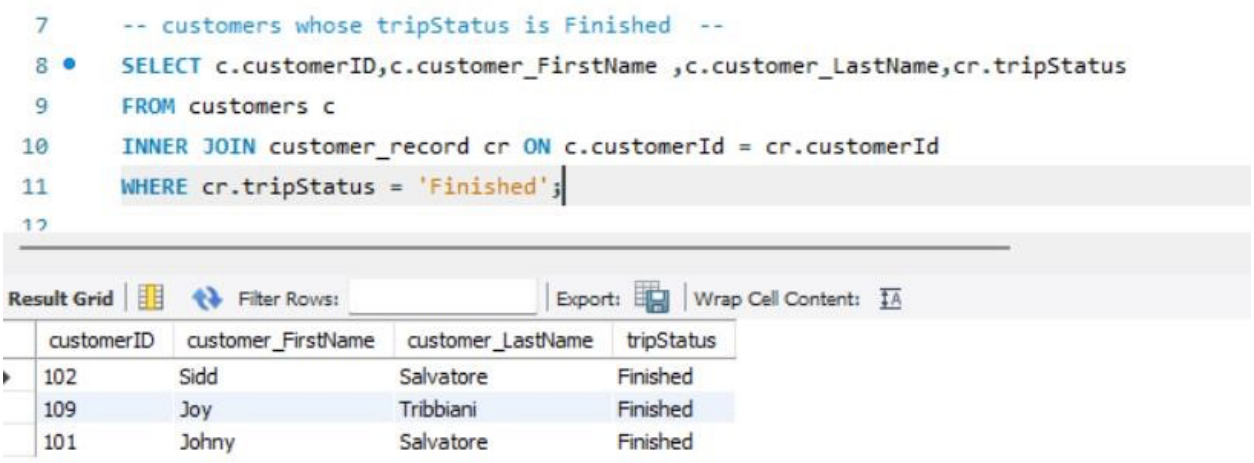
```
use `mydb`;

-- customers who have pending bill --
select c.customerID,c.customer_FirstName,t.tripID,b.billID from customers c INNER JOIN trips t ON c.customerID = t.customerID
INNER JOIN bill b on t.tripID = b.tripID where b.billStatus='0';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customerID	customer_FirstName	tripID	billID
106	Johny	305	501
106	Johny	306	504

Query2: In this query we are retrieve customers first name and customer Id whose trip status is finished



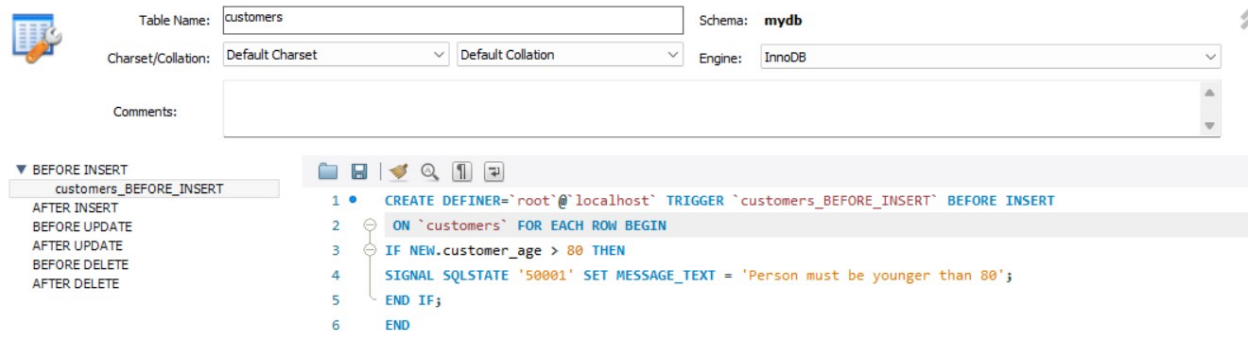
```
-- customers whose tripStatus is Finished  --
SELECT c.customerID,c.customer_FirstName ,c.customer_LastName,cr.tripStatus
FROM customers c
INNER JOIN customer_record cr ON c.customerId = cr.customerId
WHERE cr.tripStatus = 'Finished';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

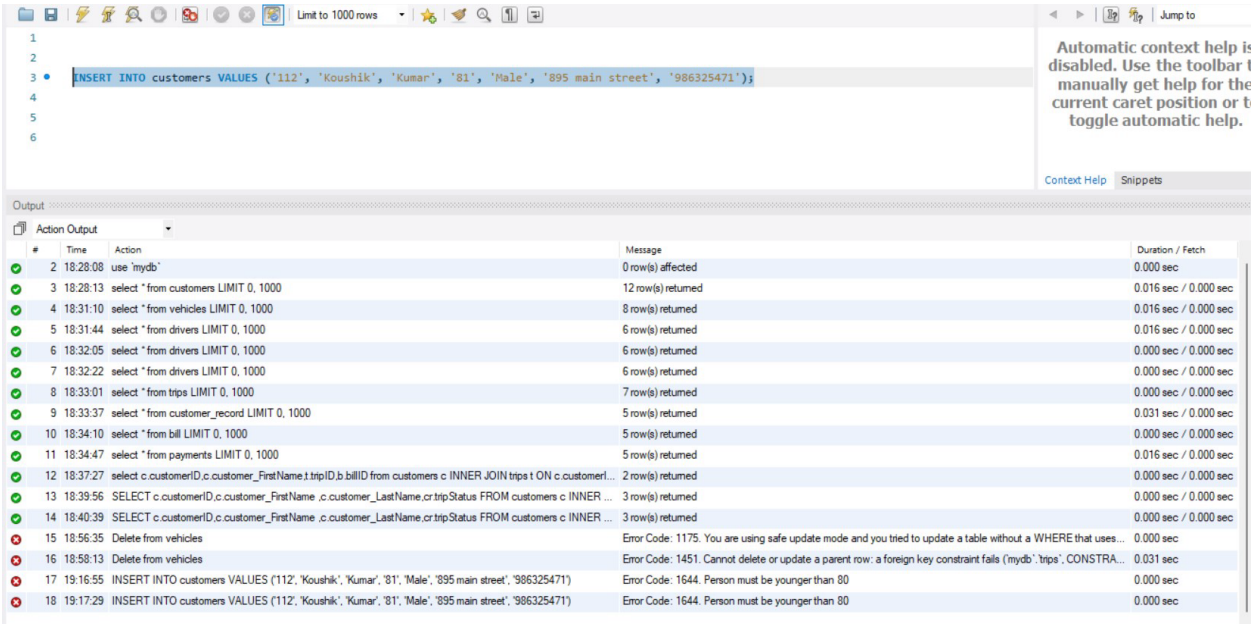
customerID	customer_FirstName	customer_LastName	tripStatus
102	Sidd	Salvatore	Finished
109	Joy	Tribbiani	Finished
101	Johny	Salvatore	Finished

Triggers and Stored Procedures:

Before_Insert Trigger – If the customers age is greater than 80 then the data should not be inserted in the table and it should show below message.



Trying to insert customers data whose age is greater than 80, which throws error as trigger executes



Stored Procedure:

We have two stored procedures in this project, one stored procedure is used for retrieving customers data whose age is less than 20 and the other stored procedure is used for inserting data into vehicles table.

1. Stored procedure for retrieving data of teenagers

```
USE `mydb`;
DROP procedure IF EXISTS `mydb`.`GetTeenageCustomers`;
;

DELIMITER $$
USE `mydb`$$
CREATE DEFINER=`root`@`localhost` PROCEDURE `GetTeenageCustomers`()
BEGIN
    Select customer_FirstName,customer_age from customers where customer_age < 20 ;

END$$

DELIMITER ;
;
```

Calling stored procedure for retrieving teenagers data

```
CALL GetTeenageCustomers();
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_FirstName	customer_age
Jon	19
Johny	19
Sidd	16
subha	19
suja	19
Johny	19
doe	19

2.Stored procedure for inserting data into vehicles table

```
USE `mydb`;
DROP procedure IF EXISTS `mydb`.`InsertNewVehicle`;
;

DELIMITER $$
USE `mydb`$$
CREATE DEFINER=`root`@`localhost` PROCEDURE `InsertNewVehicle`(
    vehicleID INT ,
    vehicle_Model VARCHAR(45),
    vehicle_InsuranceNo VARCHAR(45),
    vehicle_RegisteredNumber VARCHAR(45),
    vehicle_Company VARCHAR(45),
    vehicle_InsuranceExpiry DATE,
    vehicle_LastChecked DATE

)
BEGIN
 INSERT INTO vehicles
 VALUES (vehicleID ,
    vehicle_Model ,
    vehicle_InsuranceNo ,
    vehicle_RegisteredNumber,
    vehicle_Company ,
    vehicle_InsuranceExpiry,
    vehicle_LastChecked);
END$$

DELIMITER ;
```

Calling stored procedure for inserting data into vehicles table

```
CALL InsertNewVehicle('12990', 'Benz', '987569', 'GG', 'Mrcedes', '2015-09-10', '2019-08-27');
```

manually get help for th
current caret position or
toggle automatic help.

Context Help | Snippets

Output

Action Output

#	Time	Action	Message	Duration / Fetch
5	18:31:44	select * from drivers LIMIT 0, 1000	6 row(s) returned	0.016 sec / 0.000 sec
6	18:32:05	select * from drivers LIMIT 0, 1000	6 row(s) returned	0.000 sec / 0.000 sec
7	18:32:22	select * from drivers LIMIT 0, 1000	6 row(s) returned	0.000 sec / 0.000 sec
8	18:33:01	select * from trips LIMIT 0, 1000	7 row(s) returned	0.000 sec / 0.000 sec
9	18:33:37	select * from customer_record LIMIT 0, 1000	5 row(s) returned	0.031 sec / 0.000 sec
10	18:34:10	select * from bill LIMIT 0, 1000	5 row(s) returned	0.000 sec / 0.000 sec
11	18:34:47	select * from payments LIMIT 0, 1000	5 row(s) returned	0.016 sec / 0.000 sec
12	18:37:27	select c.customerID,c.customer_FirstName,t.tripID,b.billID from customers c INNER JOIN trips t ON c.customerI...	2 row(s) returned	0.000 sec / 0.000 sec
13	18:39:56	SELECT c.customerID,c.customer_FirstName ,c.customer_LastName,cr.tripStatus FROM customers c INNER ...	3 row(s) returned	0.000 sec / 0.000 sec
14	18:40:39	SELECT c.customerID,c.customer_FirstName ,c.customer_LastName,cr.tripStatus FROM customers c INNER ...	3 row(s) returned	0.000 sec / 0.000 sec
15	18:56:35	Delete from vehicles	Error Code: 1175. You are using safe update mode and you tried to update a table without a WHERE that uses...	0.000 sec
16	18:58:13	Delete from vehicles	Error Code: 1451. Cannot delete or update a parent row: a foreign key constraint fails (`mydb`.`trips`, CONSTRA...	0.031 sec
17	19:16:55	INSERT INTO customers VALUES ('112', 'Koushik', 'Kumar', '81', 'Male', '895 main street', '986325471')	Error Code: 1644. Person must be younger than 80	0.000 sec
18	19:17:29	INSERT INTO customers VALUES ('112', 'Koushik', 'Kumar', '81', 'Male', '895 main street', '986325471')	Error Code: 1644. Person must be younger than 80	0.000 sec
19	19:21:25	CALL GetTeenageCustomers()	9 row(s) returned	0.000 sec / 0.000 sec
20	19:25:07	CALL InsertNewVehicle('12987', 'Honda', '987564', 'KK', 'Civic', '2010-05-09', '2010-08-12')	Error Code: 1062. Duplicate entry '12987' for key 'vehicles.PRIMARY'	0.000 sec
21	19:26:39	CALL InsertNewVehicle('12990', 'Benz', '987569', 'GG', 'Mrcedes', '2015-09-10', '2019-08-27')	1 row(s) affected	0.000 sec

Tuple Relation Calculus :

Here's the TRC for the Join Queries:

Customers who have pending bills:

```
select c.customerID,c.customer_FirstName,t.tripID,b.billID from customers c INNER JOIN trips t ON
c.customerID = t.customerID
INNER JOIN bill b on t.tripID = b.tripID where b.billStatus='0';
```

{ c.customerID, c.customer_FirstName, t.tripID, b.billID | ∃ c ∃ t ∃ b (customers(c) ∧ trips(t) ∧ bill(b) ∧ c.customerID = t.customerID ∧ t.tripID = b.tripID ∧ b.billStatus = '0') }

Customers whose trip status is finished:

```
SELECT c.customerID,c.customer_FirstName ,c.customer_LastName,cr.tripStatus
FROM customers c
INNER JOIN customer_record cr ON c.customerId = cr.customerId
WHERE cr.tripStatus = 'Finished';
```

{ c.customerID, c.customer_FirstName, c.customer_LastName, cr.tripStatus | ∃c∃cr (customers(c) ∧ customer_record(cr) ∧ c.customerID = cr.customerID ∧ cr.tripStatus = 'Finished') **}**

Limitations:

Data Security and Privacy: Protecting the security and privacy of personal data, which is collected, including location, payment information, and personal information, is crucial. In order to prevent data breaches, the system needs to employ strong security measures and adhere to data protection standards such as GDPR.

Data Scalability and Volume: A lot of real-time data might come from many sources. In order to accommodate massive volumes of data without experiencing performance deterioration, the database must be scalable, which may necessitate considerable infrastructure and maintenance work.

Real-Time Data Processing: It might be challenging to develop and maintain a high-performance system with low latency when handling tasks like ride allocation and GPS location updates.

Data Accuracy and Consistency: To prevent errors in trip details, invoicing, and payments, it is crucial to guarantee the accuracy and consistency of data across all tables, particularly while handling concurrent operations.

Time Constraints: The project's scope and quality may be impacted by time restrictions related to the database system's development, deployment, and maintenance.

Git-Hub Repository:

https://github.com/bhanunath/Uber-Datatbase-design-DB-Project