Carpooling

Team : The Normalizers
Members : Rohan Oswal -W1385206
Priyanka Mehta -W1443846

Guided by – Prof. Dr. Shailesh Agarwal

TABLE OF CONTENTS

1. Business Application Description

1.1 Introduction

With the increase of environmental concerns and the congestion of roads, carpooling has gained a lot of popularity when it comes to environment-friendly and cheap ways of travelling. An average commuter travels about fifteen miles or for 26.4 minutes a day to reach his/her workplace. Carpooling is a sustainable way to travel as it reduces air pollution, carbon emissions, traffic congestions and need for parking spaces. From cost perspective it reduces fuel costs, tolls and stress of driving. The purpose of this project is to develop an understanding about carpooling and the use of databases in carpool industry.

1.2 Objective

The objectives of carpooling are:

1. **Efficient use of resources**: Carpooling facilitates efficient use of the vehicle and gas.

2. **Reduce cost**: The cost of commuting is minimized as it is shared by multiple users rather than a single commuter.

3. **Reduce carbon footprint**: Emissions will be reduced as the number of cars on the road will go down due to sharing.

4. **Reduce traffic congestions and parking issues**: Situations of heavy traffic and limited parking spaces can be controlled due to fewer vehicles.

1.3 Scope

For this project, we are considering the carpooling application for San Francisco Bay Area. The idea is to start the application first locally and then globally. We have implemented a database which records the transactions that happens in a real life carpooling app.

2. User Types or Entities

We have considered the following users for our application:

Drivers-

Drivers are the people who owns the car and wants to go from one place to another. They publish their trips on an application to find passengers to share the ride with, in return of mutually agreed fare.

Passengers-

Passengers are the people who would like to share a ride with other people. They do not drive the car, but share a ride with people who are willing to drive to their desired destination at an agreed fare.

Customer Support-

Customer support is a representative who would be a point of contact for passengers and drivers. They would also take care of refunds and complaints raised by the users.

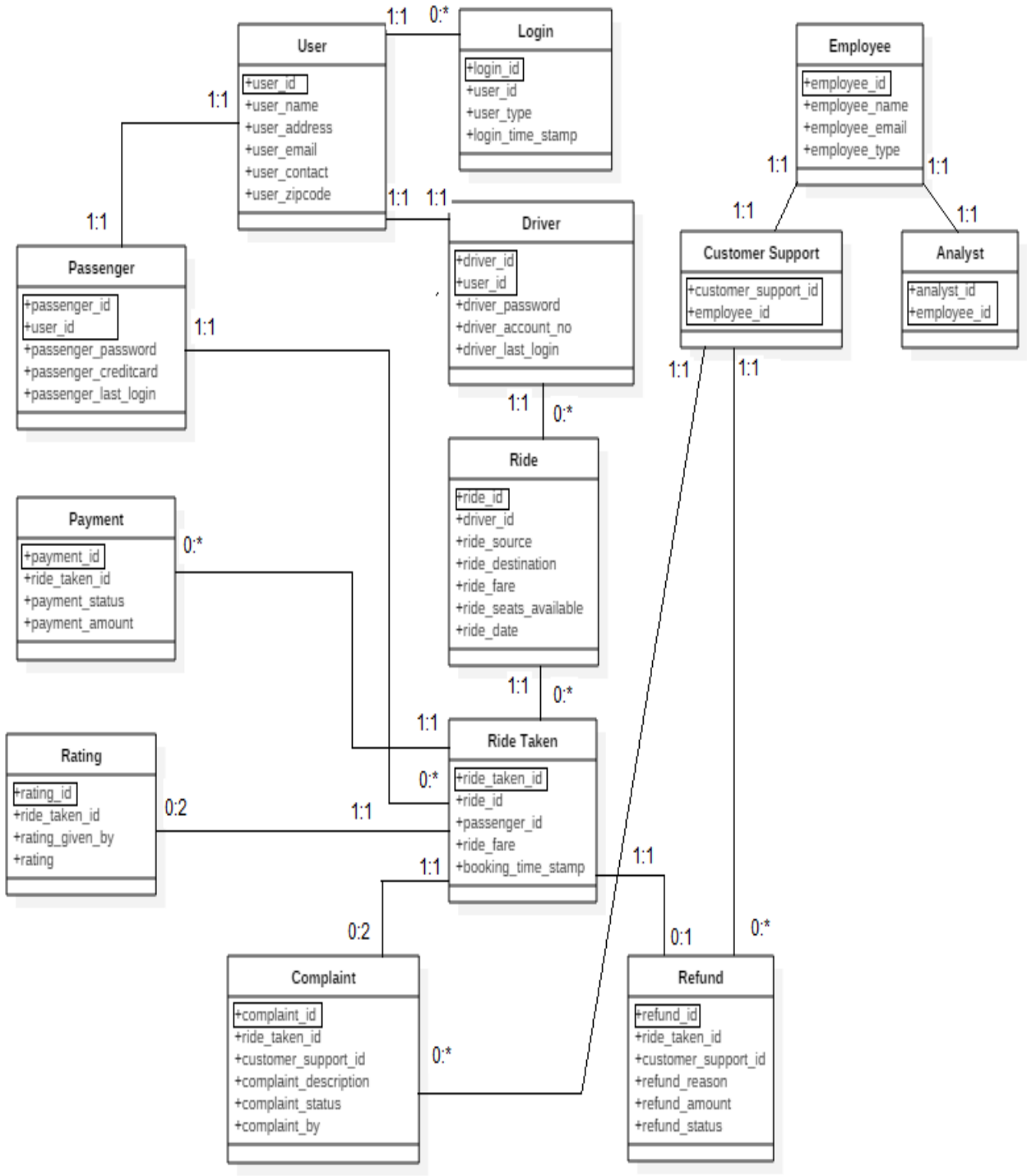
Analyst-

Analysts are the people who design and review businesses strategies. They are the people who bring in new strategies by analyzing current business cases and user generated data.

3. Tables

1. User
2. Passenger
3. Driver
4. Login
5. Ride
6. Ride Taken
7. Payment
8. Rating
9. Employee
10. Customer Support
11. Analyst
12. Complaint
13. Refund

4. Logical Schema – UML Model



5. **Use Cases**

1. Passenger
    a. Login
    b. Profile Management
    c. Book a seat
    d. Cancel a seat
    e. Make payments
    f. Apply coupons
    g. Ask for refund
    h. Give ratings
    i. Make complaints

2. Driver
    a. Login
    b. Profile Management
    c. Accept Ride
    d. Cancel Ride
    e. Reject Ride (Delays)
    f. Publish Trips
    g. Receive Payments
    h. Rate Passengers
    i. Make Complaints

3. Customer Support
    a. Manage Refunds
    b. Manage Complaints
    c. Manage Billings

4. Analyst
    a. Analyze search patterns
    b. Analyze trips
    c. Plan new strategies
    d. Analyze profile rating

6. Physical Schema - Database Dictionary

1. USER

USER			
NAME	**TYPE**	**CONSTRAINT**	**DESCRIPTION**
user_id	Integer	Primary Key	
user_name	Varchar (50)	Not Null	
user_address	Varchar (100)	Not Null	
user_email	Varchar (100)	Not Null	
user_contact	Varchar (15)	Not Null	
user_zipcode	Varchar (5)	Not Null	

2. PASSENGER

PASSENGER			
NAME	**TYPE**	**CONSTRAINT**	**DESCRIPTION**
passenger_id	Integer	Primary Key	
user_id	Integer	Primary Key, Foreign Key	
passenger_password	Varchar (12)	Not Null	
passenger_creditcard	Varchar (20)	Not Null	
passenger_last_login	Timestamp	Default 0	

3. DRIVER

DRIVER			
NAME	**TYPE**	**CONSTRAINT**	**DESCRIPTION**
driver_id	Integer	Primary Key	
user_id	Integer	Primary Key, Foreign Key	
driver_password	Varchar (12)	Not Null	
driver_account_no	Varchar (15)	Not Null	
driver_last_login	Timestamp	Default 0	

4. LOGIN

LOGIN			
NAME	**TYPE**	**CONSTRAINT**	**DESCRIPTION**
login_id	Integer	Primary Key	
user_id	Integer	Foreign Key	
user_type	varchar (10)		Domain -> Passenger, Driver
login_time_stamp	Timestamp	Default 0	

5. RIDE

RIDE			
NAME	**TYPE**	**CONSTRAINT**	**DESCRIPTION**
ride_id	Integer	Primary Key	
driver_id	Integer	Foreign Key	
ride_source	Varchar(50)	Not Null	
ride_destination	Varchar(50)	Not Null	
ride_fare	Numeric (10,2)	Not Null	
ride_seats_available	Integer	Not Null	
ride_date	Date	Not Null	

6. RIDE TAKEN

RIDE TAKEN			
NAME	**TYPE**	**CONSTRAINT**	**DESCRIPTION**
ride_taken_id	Integer	Primary Key	
ride_id	Integer	Foreign Key	
passenger_id	integer	Foreign Key	
ride_fare	Numeric (10,2)	Not Null	
booking_time_stamp	Timestamp	Not Null	

7. PAYMENT

PAYMENT			
NAME	**TYPE**	**CONSTRAINT**	**DESCRIPTION**
payment_id	Integer	Primary Key	
ride_taken_id	Integer	Foreign Key	
payment_status	Varchar(12)		Domain -> Paid, Outstanding
payment_amount	Numeric (10,2)	Not Null	

8. RATING

RATING			
NAME	**TYPE**	**CONSTRAINT**	**DESCRIPTION**
rating_id	Integer	Primary Key	
ride_taken_id	Integer	Foreign Key	
rating_given_by	Varchar (10)		Domain-> Passenger, Driver
Rating	Integer	Default Null	Domain ->1-5

9. EMPLOYEE

EMPLOYEE			
NAME	**TYPE**	**CONSTRAINT**	**DESCRIPTION**
employee_id	Integer	Primary Key	
employee_name	Varchar (50)	Not Null	
employee_email	Varchar (100)	Not Null	
employee_type	Varchar (16)		Domain -> Analyst, Customer Support

10. CUSTOMER SUPPORT

CUSTOMER SUPPORT			
NAME	**TYPE**	**CONSTRAINT**	**DESCRIPTION**
customer_support_id	Integer	Primary Key	
employee_id	Integer	Primary Key, Foreign Key	

11. ANALYST

ANALYST			
NAME	**TYPE**	**CONSTRAINT**	**DESCRIPTION**
analyst_id	Integer	Primary Key	
employee_id	Integer	Primary Key, Foreign Key	

12. COMPLAINT

COMPLAINT			
NAME	**TYPE**	**CONSTRAINT**	**DESCRIPTION**
complaint_id	Integer	Primary Key	
ride_taken_id	Integer	Foreign Key	
customer_support_id	Integer	Foreign Key	
complaint_description	Text	Default - Null	
complaint_status	Varchar (10)		Domain-> Initiated, Pending, Resolved
complaint_by	Varchar (10)		Domain-> Passenger, Driver

13. REFUND

REFUND			
NAME	**TYPE**	**CONSTRAINT**	**DESCRIPTION**
refund_id	Integer	Primary Key	
ride_taken_id	Integer	Foreign Key	
customer_support_id	Integer	Foreign Key	
refund_reason	Text	Not Null	
refund_amount	NUMERIC(10,2)	Not Null	
refund_status	Varchar(12)		Domain-> Initiated, Pending, Resolved

7. Queries

USER

1.Search for users who live in the same zipcode.

```
mysql> SELECT U.user_name, U.user_zipcode
    -> FROM User U
    -> WHERE U.user_zipcode = ANY
    -> (SELECT U1.user_zipcode
    -> FROM User U1
    -> WHERE U1.user_id <> U.user_id)
    -> ORDER BY U.user_zipcode;
+--------------------+--------------+
| user_name          | user_zipcode |
+--------------------+--------------+
| Clinton Guzman     | 95008        |
| Astra Everett      | 95008        |
| Kiona Ashley       | 95014        |
| Ingrid Mullins     | 95014        |
| Virginia Cunningham | 95014        |
| Kaye Bird          | 95083        |
| Malachi Marsh      | 95083        |
+--------------------+--------------+
7 rows in set (0.00 sec)
```

2. Insert a new user as a passenger

Before Insert

```
mysql> SELECT *
    -> FROM User;
+---------+--------------------+-------------------------------+-----------------------------------------------+--------------+--------------+
| user_id | user_name          | user_address                  | user_email                                    | user_contact | user_zipcode |
+---------+--------------------+-------------------------------+-----------------------------------------------+--------------+--------------+
|       1 | Kiona Ashley       | P.O. Box 730, 224 Ut, Road    | natoque.penatibus.et@commodoipsumSuspendisse.net | 474-0158     | 95014        |
|       2 | Bryar Hopkins      | 8482 Massa Rd.                | non.sollicitudin@Donecdignissimmagna.org      | 995-5229     | 95086        |
|       3 | Ingrid Mullins     | 582-7339 Vitae Ave            | amet.lorem@loremDonecelementum.org            | 291-8876     | 95014        |
|       4 | Clinton Guzman     | 923-1666 Parturient St.       | semper.pretium.neque@Phasellus.edu            | 871-7377     | 95008        |
|       5 | Astra Everett      | Ap #563-4042 Aliquam Av.      | nibh@lectusasollicitudin.com                  | 1-900-162-0731 | 95008        |
|       6 | Kaye Bird          | P.O. Box 391, 9738 Aliquam St. | nec.diam.Duis@Aliquamauctorvelit.com          | 237-7845     | 95083        |
|       7 | Aladdin Harmon     | 410-9599 Aliquam Ave          | Curabitur@vulputatelacus.co.uk                | 1-127-328-7250 | 95623        |
|       8 | Malachi Marsh      | 5395 Duis St.                 | nascetur.ridiculus@ridiculus.edu              | 1-168-120-7039 | 95083        |
|       9 | Virginia Cunningham | 7803 Dui Avenue               | eleifend.vitae@arcuVestibulum.edu             | 905-2791     | 95014        |
|      10 | Evangeline Gaines  | 2030 Ridiculus St.            | In@estarcuac.org                              | 588-5223     | 95432        |
+---------+--------------------+-------------------------------+-----------------------------------------------+--------------+--------------+
10 rows in set (0.00 sec)
```

After Insert

```
mysql> INSERT INTO User (user_id, user_name, user_address, user_email, user_contact, user_zipcode)
    -> VALUES (11, "Brinne Gomes", "115 North First Street", "gomes@selllasr.com", "465-9874", "95876");
Query OK, 1 row affected (0.00 sec)

mysql> SELECT *
    -> FROM User;
+---------+--------------------+-------------------------------+-----------------------------------------------+--------------+--------------+
| user_id | user_name          | user_address                  | user_email                                    | user_contact | user_zipcode |
+---------+--------------------+-------------------------------+-----------------------------------------------+--------------+--------------+
|       1 | Kiona Ashley       | P.O. Box 730, 224 Ut, Road    | natoque.penatibus.et@commodoipsumSuspendisse.net | 474-0158     | 95014        |
|       2 | Bryar Hopkins      | 8482 Massa Rd.                | non.sollicitudin@Donecdignissimmagna.org      | 995-5229     | 95086        |
|       3 | Ingrid Mullins     | 582-7339 Vitae Ave            | amet.lorem@loremDonecelementum.org            | 291-8876     | 95014        |
|       4 | Clinton Guzman     | 923-1666 Parturient St.       | semper.pretium.neque@Phasellus.edu            | 871-7377     | 95008        |
|       5 | Astra Everett      | Ap #563-4042 Aliquam Av.      | nibh@lectusasollicitudin.com                  | 1-900-162-0731 | 95008        |
|       6 | Kaye Bird          | P.O. Box 391, 9738 Aliquam St. | nec.diam.Duis@Aliquamauctorvelit.com          | 237-7845     | 95083        |
|       7 | Aladdin Harmon     | 410-9599 Aliquam Ave          | Curabitur@vulputatelacus.co.uk                | 1-127-328-7250 | 95623        |
|       8 | Malachi Marsh      | 5395 Duis St.                 | nascetur.ridiculus@ridiculus.edu              | 1-168-120-7039 | 95083        |
|       9 | Virginia Cunningham | 7803 Dui Avenue               | eleifend.vitae@arcuVestibulum.edu             | 905-2791     | 95014        |
|      10 | Evangeline Gaines  | 2030 Ridiculus St.            | In@estarcuac.org                              | 588-5223     | 95432        |
|      11 | Brinne Gomes       | 115 North First Street        | gomes@selllasr.com                            | 465-9874     | 95876        |
+---------+--------------------+-------------------------------+-----------------------------------------------+--------------+--------------+
11 rows in set (0.00 sec)
```

3. Find users who are both driver and passenger

```
mysql> SELECT U.user_id, U.user_name
    -> FROM User U
    -> WHERE U.user_id
    -> IN
    -> (SELECT D.user_id
    -> FROM Driver D
    -> INNER JOIN
    -> Passenger P ON
    -> D.user_id = P.user_id);
+---------+----------------+
| user_id | user_name      |
+---------+----------------+
|       3 | Ingrid Mullins |
|       5 | Astra Everett  |
+---------+----------------+
2 rows in set (0.00 sec)
```

PASSENGER

4. Insert of user as a passenger (As there is a new entry in User table, there will be a corresponding entry in passenger table)

Before Insert

```
mysql> SELECT *
    -> FROM Passenger;
+--------------+---------+--------------------+----------------------+----------------------+
| passenger_id | user_id | passenger_password | passenger_last_login | passenger_credit_card |
+--------------+---------+--------------------+----------------------+----------------------+
|            1 |       2 | WMU77ENO8EJ        | 2016-10-11 07:01:39  | 550944 187153 8246   |
|            2 |       4 | TXT41CDB7AT        | 2017-05-09 08:08:51  | 5366762894955097     |
|            3 |       6 | VAT45LWG8TK        | 2017-06-11 08:38:13  | 5220 6203 6385 0403  |
|            4 |       8 | FQP92WMI3NP        | 2017-09-25 03:48:52  | 5506 6049 2834 4230  |
|            5 |       9 | RUP13QQF8NP        | 2017-10-04 15:01:09  | 5588692189271185     |
|            6 |       3 | RFQ31PRE2EU        | 2017-04-02 22:16:59  | 523021 5607228470    |
|            7 |       5 | ELT54MGG0FG        | 2017-05-11 07:44:48  | 531926 4560784214    |
+--------------+---------+--------------------+----------------------+----------------------+
7 rows in set (0.00 sec)
```

After Insert

```
mysql> INSERT INTO Passenger (passenger_id, user_id, passenger_password, passenger_last_login, passenger_credit_card) VALUES (8, 11, "KLU7657AS", "2017-11-22 08:46:55",
"4565565678798112");
Query OK, 1 row affected (0.01 sec)
```

```
mysql> SELECT *
    -> FROM Passenger;
+--------------+---------+--------------------+----------------------+----------------------+
| passenger_id | user_id | passenger_password | passenger_last_login | passenger_credit_card |
+--------------+---------+--------------------+----------------------+----------------------+
|            1 |       2 | WMU77ENO8EJ        | 2016-10-11 07:01:39  | 550944 187153 8246   |
|            2 |       4 | TXT41CDB7AT        | 2017-05-09 08:08:51  | 5366762894955097     |
|            3 |       6 | VAT45LWG8TK        | 2017-06-11 08:38:13  | 5220 6203 6385 0403  |
|            4 |       8 | FQP92WMI3NP        | 2017-09-25 03:48:52  | 5506 6049 2834 4230  |
|            5 |       9 | RUP13QQF8NP        | 2017-10-04 15:01:09  | 5588692189271185     |
|            6 |       3 | RFQ31PRE2EU        | 2017-04-02 22:16:59  | 523021 5607228470    |
|            7 |       5 | ELT54MGG0FG        | 2017-05-11 07:44:48  | 531926 4560784214    |
|            8 |      11 | KLU7657AS          | 2017-11-22 08:46:55  | 4565565678798112     |
+--------------+---------+--------------------+----------------------+----------------------+
8 rows in set (0.00 sec)
```

5. Find Name, Address of passengers who have taken ride provided by driver_id – 3

```
mysql> SELECT DISTINCT U.user_name,U.user_address
    -> FROM User U,Passenger P,Ride R,Ride_Taken RT
    -> WHERE U.user_id = P.user_id and
    -> P.passenger_id = RT.passenger_id and
    -> R.ride_id = RT.ride_id and
    -> R.driver_id = 3;
+--------------------+------------------------+
| user_name          | user_address           |
+--------------------+------------------------+
| Clinton Guzman     | 923-1666 Parturient St. |
| Ingrid Mullins     | 582-7339 Vitae Ave     |
| Virginia Cunningham | 7803 Dui Avenue       |
| Bryar Hopkins      | 8482 Massa Rd.         |
| Malachi Marsh      | 5395 Duis St.          |
+--------------------+------------------------+
5 rows in set (0.00 sec)
```

6. Find users who have taken maximum number of rides

```
mysql> SELECT U.user_name,count(ride_taken_id) as 'Ride_Count'
    -> FROM User U, Passenger P, Ride_Taken RT
    -> WHERE P.passenger_id = RT.passenger_id and P.user_id = U.user_id
    -> GROUP BY P.passenger_id
    -> HAVING Ride_Count >= ALL
    -> (SELECT COUNT(ride_taken_id)
    -> FROM Ride_Taken RT1
    -> WHERE P.passenger_id <> RT1.passenger_id
    -> GROUP BY RT1.passenger_id);
+--------------------+------------+
| user_name          | Ride_Count |
+--------------------+------------+
| Clinton Guzman     |          5 |
| Virginia Cunningham |          5 |
+--------------------+------------+
2 rows in set (0.00 sec)
```

DRIVER

7. Find names of drivers and the number of rides posted till now

```
mysql> SELECT U.user_name,count(R.ride_id) AS 'No. of Rides Published'
    -> FROM User U,Driver D,Ride R
    -> WHERE U.user_id = D.user_id and
    -> D.driver_id = R.driver_id
    -> GROUP BY D.driver_id;
+-------------------+------------------------+
| user_name         | No. of Rides Published |
+-------------------+------------------------+
| Kiona Ashley      |                      2 |
| Astra Everett     |                      3 |
| Aladdin Harmon    |                      3 |
| Evangeline Gaines |                      2 |
+-------------------+------------------------+
4 rows in set (0.00 sec)
```

8. Find names of drivers who has not published any ride till now

```
mysql> SELECT U.user_name
    -> FROM User U
    -> WHERE U.user_id =
    -> (SELECT D.user_id
    -> FROM Driver D
    -> WHERE D.driver_id =
    -> (SELECT D.driver_id
    -> FROM Driver D
    -> WHERE D.driver_id NOT IN
    -> (SELECT DISTINCT R.driver_id
    -> FROM Ride R)));
+----------------+
| user_name      |
+----------------+
| Ingrid Mullins |
+----------------+
1 row in set (0.00 sec)
```

CUSTOMER SUPPORT

9. Find number of refunds, complaint processed by each customer representative.

Complaints Served

```
mysql> SELECT E.employee_name as 'Representative_Name',count(*) as 'Complaints_Served'
    -> FROM Employee E
    -> INNER JOIN Customer_Support CS
    -> on E.employee_id = CS.employee_id
    -> INNER JOIN Complaint C
    -> on CS.customer_support_id = C.customer_support_id
    -> GROUP BY CS.customer_support_id;
+---------------------+-------------------+
| Representative_Name | Complaints_Served |
+---------------------+-------------------+
| Chelsea Winters     |                 1 |
| Indigo Butler       |                 1 |
| Aidan Wong          |                 1 |
| Camille Bradley     |                 1 |
| Steven Sandoval     |                 1 |
+---------------------+-------------------+
5 rows in set (0.00 sec)
```

Refunds Served

```
mysql> SELECT E.employee_name as 'Representative_Name',count(*) as 'Refund_Requests_Served'
    -> FROM Employee E
    -> INNER JOIN Customer_Support CS
    -> on E.employee_id = CS.employee_id
    -> INNER JOIN Refund R
    -> on CS.customer_support_id = R.customer_support_id
    -> GROUP BY CS.customer_support_id;
+---------------------+------------------------+
| Representative_Name | Refund_Requests_Served |
+---------------------+------------------------+
| Ulric Terry         |                      1 |
| Chelsea Winters     |                      1 |
| Indigo Butler       |                      1 |
| Erica Shannon       |                      1 |
+---------------------+------------------------+
4 rows in set (0.00 sec)
```

RIDES

10. Find rides available for month of September 2017

```
mysql> SELECT ride_source,ride_destination,ride_fare
    -> FROM Ride
    -> WHERE ride_date like '2017-09%';
+------------------+------------------+-----------+
| ride_source      | ride_destination | ride_fare |
+------------------+------------------+-----------+
| Fermentum Street | Non St.          |      4.82 |
| Quam St.         | Libero Avenue    |      4.89 |
| Convallis Rd.    | Ante St.         |      2.66 |
+------------------+------------------+-----------+
3 rows in set, 1 warning (0.00 sec)
```

11. Find the source and destination from ride with lowest fare

```
mysql> SELECT R.ride_source,R.ride_destination,R.ride_fare
    -> FROM Ride R where R.ride_fare < ALL
    -> (SELECT R1.ride_fare
    -> FROM Ride R1
    -> WHERE R.ride_id <> R1.ride_id);
+-------------+------------------+-----------+
| ride_source | ride_destination | ride_fare |
+-------------+------------------+-----------+
| Erat St.    | Libero Avenue    |      1.75 |
+-------------+------------------+-----------+
1 row in set (0.00 sec)
```

PAYMENT

12. Amount paid to every driver as per rides given.

```
mysql> SELECT D.driver_id,U.user_name,sum(payment_amount) as 'Amount_Earned'
    -> FROM User U,Driver D,Ride R,Ride_Taken RT,Payment P
    -> WHERE U.user_id = D.user_id and
    -> D.driver_id = R.driver_id and
    -> R.ride_id = RT.ride_id and
    -> RT.ride_taken_id = P.ride_taken_id
    -> and P.payment_status = 'Paid'
    -> GROUP BY D.driver_id;
+-----------+------------------+---------------+
| driver_id | user_name        | Amount_Earned |
+-----------+------------------+---------------+
|         1 | Kiona Ashley     |         10.18 |
|         3 | Astra Everett    |         25.50 |
|         4 | Aladdin Harmon   |         13.93 |
|         5 | Evangeline Gaines |         12.44 |
+-----------+------------------+---------------+
4 rows in set (0.00 sec)
```

RATING

13. Update rating for ride taken id

Before Update

```
mysql> SELECT *
    -> FROM Rating;
+-----------+---------------+-----------------+--------+
| rating_id | ride_taken_id | rating_given_by | rating |
+-----------+---------------+-----------------+--------+
|         1 |             2 | Passenger       |      4 |
|         2 |             5 | Passenger       |      4 |
|         3 |            10 | Passenger       |      3 |
|         4 |             9 | Driver          |      5 |
|         5 |             1 | Passenger       |      2 |
|         6 |             3 | Driver          |      5 |
|         7 |             8 | Passenger       |      5 |
|         8 |            12 | Driver          |      4 |
|         9 |            15 | Passenger       |      2 |
|        10 |            19 | Driver          |      2 |
+-----------+---------------+-----------------+--------+
10 rows in set (0.00 sec)
```

After Update

```
mysql> UPDATE Rating
    -> SET Rating = '4'
    -> WHERE ride_taken_id = 15 AND
    -> rating_given_by = 'Passenger';
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> SELECT *
    -> FROM Rating;
+-----------+---------------+-----------------+--------+
| rating_id | ride_taken_id | rating_given_by | rating |
+-----------+---------------+-----------------+--------+
|         1 |             2 | Passenger       |      4 |
|         2 |             5 | Passenger       |      4 |
|         3 |            10 | Passenger       |      3 |
|         4 |             9 | Driver          |      5 |
|         5 |             1 | Passenger       |      2 |
|         6 |             3 | Driver          |      5 |
|         7 |             8 | Passenger       |      5 |
|         8 |            12 | Driver          |      4 |
|         9 |            15 | Passenger       |      4 |
|        10 |            19 | Driver          |      2 |
+-----------+---------------+-----------------+--------+
10 rows in set (0.00 sec)
```

REFUND

14. Total Amount refunded by the company till date

```
mysql> SELECT SUM(refund_amount) AS 'Refund_till_date'
    -> FROM Refund
    -> WHERE refund_status = 'Resolved';
+------------------+
| Refund_till_date |
+------------------+
|             4.98 |
+------------------+
1 row in set (0.00 sec)
```

15. Delete Refund row where refund id = 4

Before Delete

```
mysql> SELECT *
    -> FROM Refund;
+-----------+--------------+--------------------+-----------------------------------------------------------------------------------------------+---------------+---------------+
| refund_id | ride_taken_id | customer_support_id | refund_reason                                                                                 | refund_status | refund_amount |
+-----------+--------------+--------------------+-----------------------------------------------------------------------------------------------+---------------+---------------+
|         1 |             2 |                  3 | lobortis tellus justo sit amet nulla. Donec non justo.                                        | Resolved      |          3.51 |
|         2 |            10 |                  1 | elit elit fermentum risus, at fringilla purus mauris a nunc. In at pede. Cras vulputate       | Resolved      |          1.47 |
|         3 |            15 |                  7 | rhoncus. Donec est. Nunc ullamcorper, velit in aliquet lobortis, nisi                         | Resolved      |          0.00 |
|         4 |            19 |                  2 | tristique senectus et netus et malesuada fames ac turpis egestas.                             | Pending       |          5.93 |
+-----------+--------------+--------------------+-----------------------------------------------------------------------------------------------+---------------+---------------+
4 rows in set (0.00 sec)
```

After Delete

```
mysql> DELETE
    -> FROM Refund
    -> WHERE refund_id = '4';
Query OK, 1 row affected (0.01 sec)

mysql> SELECT *
    -> FROM Refund;
+-----------+--------------+--------------------+-----------------------------------------------------------------------------------------------+---------------+---------------+
| refund_id | ride_taken_id | customer_support_id | refund_reason                                                                                 | refund_status | refund_amount |
+-----------+--------------+--------------------+-----------------------------------------------------------------------------------------------+---------------+---------------+
|         1 |             2 |                  3 | lobortis tellus justo sit amet nulla. Donec non justo.                                        | Resolved      |          3.51 |
|         2 |            10 |                  1 | elit elit fermentum risus, at fringilla purus mauris a nunc. In at pede. Cras vulputate       | Resolved      |          1.47 |
|         3 |            15 |                  7 | rhoncus. Donec est. Nunc ullamcorper, velit in aliquet lobortis, nisi                         | Resolved      |          0.00 |
+-----------+--------------+--------------------+-----------------------------------------------------------------------------------------------+---------------+---------------+
3 rows in set (0.00 sec)
```

8. INDEXES

1. All the Primary keys in the tables can be Indexed.
2. The ride_source and ride_destination in the ride table can be Indexed.

9. TRIGGERS

Use 1:

Whenever a user logs into the system again as a driver or a passenger, the user's login last timestamp can be updated in respective driver or passenger table using a trigger.

When a user with id=5 logs in as a Driver:

```
mysql> INSERT INTO Login (user_id, user_type, login_time_stamp)
    -> VALUES (5, "Driver", NOW());
Query OK, 1 row affected (0.01 sec)
```

New entry in login table with current timestamp of login:

```
mysql> select * from Login order by login_time_stamp desc;
+----------+---------+-----------+---------------------+
| login_id | user_id | user_type | login_time_stamp    |
+----------+---------+-----------+---------------------+
|       21 |       5 | Driver    | 2017-11-26 21:57:18 |
|       20 |      10 | Driver    | 2017-10-25 12:55:32 |
|       19 |       9 | Passenger | 2017-10-04 15:01:09 |
|       18 |       8 | Passenger | 2017-09-25 03:48:52 |
```

So, the last login timestamp value in table driver for user_id = 5 needs to be updated to this current timestamp value and this can be done using a trigger.

This is how the Driver table would look after the trigger runs.

```
mysql> select * from Driver;
+-----------+---------+-----------------+---------------------+-------------------+
| driver_id | user_id | driver_password | driver_last_login   | driver_account_no |
+-----------+---------+-----------------+---------------------+-------------------+
|         1 |       1 | YZI48GPI3FC     | 2016-09-27 20:16:17 | 375785884022735   |
|         2 |       3 | TFI02GFY6RG     | 0000-00-00 00:00:00 | 3748 340697 577   |
|         3 |       5 | COG99IAG0GL     | 2017-11-26 21:57:18 | 3785 692506 043   |
|         4 |       7 | TXL72WYL6PE     | 2017-07-13 22:02:39 | 3774 461104 073   |
|         5 |      10 | MLL71HXY1AB     | 2017-10-25 12:55:32 | 3753 195785 249   |
+-----------+---------+-----------------+---------------------+-------------------+
5 rows in set (0.00 sec)

mysql>
```

Use 2:

We would also need triggers to update seats available for a particular ride when a passenger confirms a seat for the ride. The new available count should be one less than than the earlier value.

10.VIEWS

1. Details of passengers and their rides taken

CREATE VIEW

```
mysql> CREATE VIEW Ride_details AS
    -> SELECT U.user_id, P.passenger_id, U.user_name, U.user_email, RT.ride_taken_id, R.ride_id, R.ride_source, R.ride_destination
    -> FROM User U, Passenger P, Ride_Taken RT, Ride R
    -> WHERE P.passenger_id = RT.passenger_id AND U.user_id = P.user_id AND R.ride_id = RT.ride_id;
Query OK, 0 rows affected (0.01 sec)
```

DISPLAY VIEW

```
mysql> SELECT *
    -> FROM Ride_details;
+---------+--------------+--------------------+-----------------------------------+---------------+---------+-----------------+------------------+
| user_id | passenger_id | user_name          | user_email                        | ride_taken_id | ride_id | ride_source     | ride_destination |
+---------+--------------+--------------------+-----------------------------------+---------------+---------+-----------------+------------------+
|       2 |            1 | Bryar Hopkins      | non.sollicitudin@Donecdignissimmagna.org |            13 |       8 | Quam St.        | Libero Avenue    |
|       3 |            6 | Ingrid Mullins     | amet.lorem@loremDonecelementum.org |             7 |       3 | Quam. St.       | Sed Road         |
|       3 |            6 | Ingrid Mullins     | amet.lorem@loremDonecelementum.org |            10 |       5 | Erat St.        | Libero Avenue    |
|       3 |            6 | Ingrid Mullins     | amet.lorem@loremDonecelementum.org |            12 |       7 | Fermentum Street | Non St.          |
|       4 |            2 | Clinton Guzman     | semper.pretium.neque@Phasellus.edu |             2 |       1 | Montes Street   | Ante St.         |
|       4 |            2 | Clinton Guzman     | semper.pretium.neque@Phasellus.edu |             5 |       2 | Sed Avenue      | Libero Avenue    |
|       4 |            2 | Clinton Guzman     | semper.pretium.neque@Phasellus.edu |             6 |       3 | Quam. St.       | Sed Road         |
|       4 |            2 | Clinton Guzman     | semper.pretium.neque@Phasellus.edu |             9 |       5 | Erat St.        | Libero Avenue    |
|       4 |            2 | Clinton Guzman     | semper.pretium.neque@Phasellus.edu |            15 |       8 | Quam St.        | Libero Avenue    |
|       5 |            7 | Astra Everett      | nibh@lectusasollicitudin.com      |             8 |       4 | Erat St.        | Libero Avenue    |
|       5 |            7 | Astra Everett      | nibh@lectusasollicitudin.com      |            18 |      10 | Duis Ave        | Magna. Rd.       |
|       6 |            3 | Kaye Bird          | nec.diam.Duis@Aliquamauctorvelit.com |             1 |       1 | Montes Street   | Ante St.         |
|       6 |            3 | Kaye Bird          | nec.diam.Duis@Aliquamauctorvelit.com |            19 |      10 | Duis Ave        | Magna. Rd.       |
|       8 |            4 | Malachi Marsh      | nascetur.ridiculus@ridiculus.edu  |             4 |       2 | Sed Avenue      | Libero Avenue    |
|       8 |            4 | Malachi Marsh      | nascetur.ridiculus@ridiculus.edu  |            14 |       8 | Quam St.        | Libero Avenue    |
|       9 |            5 | Virginia Cunningham | eleifend.vitae@arcuVestibulum.edu |             3 |       2 | Sed Avenue      | Libero Avenue    |
|       9 |            5 | Virginia Cunningham | eleifend.vitae@arcuVestibulum.edu |            11 |       6 | Luctus Av.      | Neque. Rd.       |
|       9 |            5 | Virginia Cunningham | eleifend.vitae@arcuVestibulum.edu |            16 |       8 | Quam St.        | Libero Avenue    |
|       9 |            5 | Virginia Cunningham | eleifend.vitae@arcuVestibulum.edu |            17 |       9 | Convallis Rd.   | Ante St.         |
|       9 |            5 | Virginia Cunningham | eleifend.vitae@arcuVestibulum.edu |            20 |      10 | Duis Ave        | Magna. Rd.       |
+---------+--------------+--------------------+-----------------------------------+---------------+---------+-----------------+------------------+
20 rows in set (0.00 sec)
```

Queries:

1. Find the details of passengers who have taken Ride_id - 8

```
mysql> SELECT RD.user_name, RD.user_email
    -> FROM Ride_details RD
    -> WHERE RD.ride_id = 8;
+--------------------+-----------------------------------------+
| user_name          | user_email                              |
+--------------------+-----------------------------------------+
| Bryar Hopkins      | non.sollicitudin@Donecdignissimmagna.org |
| Malachi Marsh      | nascetur.ridiculus@ridiculus.edu        |
| Clinton Guzman     | semper.pretium.neque@Phasellus.edu      |
| Virginia Cunningham | eleifend.vitae@arcuVestibulum.edu      |
+--------------------+-----------------------------------------+
4 rows in set (0.00 sec)
```

2. Find passengers whose destination is Libero

```
mysql> SELECT DISTINCT RD.user_name, RD.user_email
    -> FROM Ride_details RD
    -> WHERE RD.ride_destination LIKE "Libero%";
+---------------------+-----------------------------------------+
| user_name           | user_email                              |
+---------------------+-----------------------------------------+
| Bryar Hopkins       | non.sollicitudin@Donecdignissimmagna.org |
| Ingrid Mullins      | amet.lorem@loremDonecelementum.org      |
| Clinton Guzman      | semper.pretium.neque@Phasellus.edu      |
| Astra Everett       | nibh@lectusasollicitudin.com            |
| Malachi Marsh       | nascetur.ridiculus@ridiculus.edu        |
| Virginia Cunningham | eleifend.vitae@arcuVestibulum.edu       |
+---------------------+-----------------------------------------+
6 rows in set (0.00 sec)
```

2. Details of drivers and their ratings

CREATE VIEW

```
mysql> CREATE view Driver_Ratings as
    -> SELECT U.user_name,D.driver_id,RD.ride_source,RD.ride_destination,RT.passenger_id,R.rating
    -> FROM User U, Driver D, Rating R, Ride_Taken RT, Ride RD
    -> WHERE U.user_id = D.user_id and
    -> D.driver_id = RD.driver_id and
    -> RD.ride_id = RT.ride_id and
    -> R.ride_taken_id = RT.ride_taken_id
    -> and R.rating_given_by = 'Passenger';
Query OK, 0 rows affected (0.01 sec)
```

DISPLAY VIEW

```
mysql> SELECT *
    -> FROM Driver_Ratings;
+---------------+-----------+---------------+------------------+--------------+--------+
| user_name     | driver_id | ride_source   | ride_destination | passenger_id | rating |
+---------------+-----------+---------------+------------------+--------------+--------+
| Kiona Ashley  |         1 | Montes Street | Ante St.         |            2 |      4 |
| Aladdin Harmon |         4 | Sed Avenue    | Libero Avenue    |            2 |      4 |
| Aladdin Harmon |         4 | Erat St.      | Libero Avenue    |            6 |      3 |
| Kiona Ashley  |         1 | Montes Street | Ante St.         |            3 |      2 |
| Aladdin Harmon |         4 | Erat St.      | Libero Avenue    |            7 |      5 |
| Astra Everett |         3 | Quam St.      | Libero Avenue    |            2 |      4 |
+---------------+-----------+---------------+------------------+--------------+--------+
6 rows in set (0.00 sec)
```

Queries

1. Find drivers who went to Libero Avenue and have got good ratings (more than 3)

```
mysql> SELECT DISTINCT user_name
    -> FROM Driver_Ratings
    -> WHERE ride_destination = 'Libero Avenue' and rating > 3;
+----------------+
| user_name      |
+----------------+
| Aladdin Harmon |
| Astra Everett  |
+----------------+
2 rows in set (0.00 sec)
```

2. Find the driver with the lowest rating

```
mysql> SELECT DISTINCT DR1.user_name
    -> FROM Driver_Ratings DR1
    -> WHERE rating <= ALL
    -> (SELECT DR2.rating
    -> FROM Driver_Ratings DR2);
+--------------+
| user_name    |
+--------------+
| Kiona Ashley |
+--------------+
1 row in set (0.00 sec)
```

11.BUSINESS METRICS

1. Active users

- Drivers - Drivers who have posted ride in last 3 months

```
mysql> SELECT DISTINCT U.user_name AS 'Active_Drivers'
    -> from User U, Driver D, Ride R
    -> WHERE U.user_id = D.user_id AND
    -> D.driver_id = R.driver_id AND
    -> R.ride_date BETWEEN DATE_SUB(NOW(), INTERVAL 90 DAY) AND NOW();
+--------------------+
| Active_Drivers     |
+--------------------+
| Evangeline Gaines  |
| Astra Everett      |
| Kiona Ashley       |
+--------------------+
3 rows in set (0.00 sec)
```

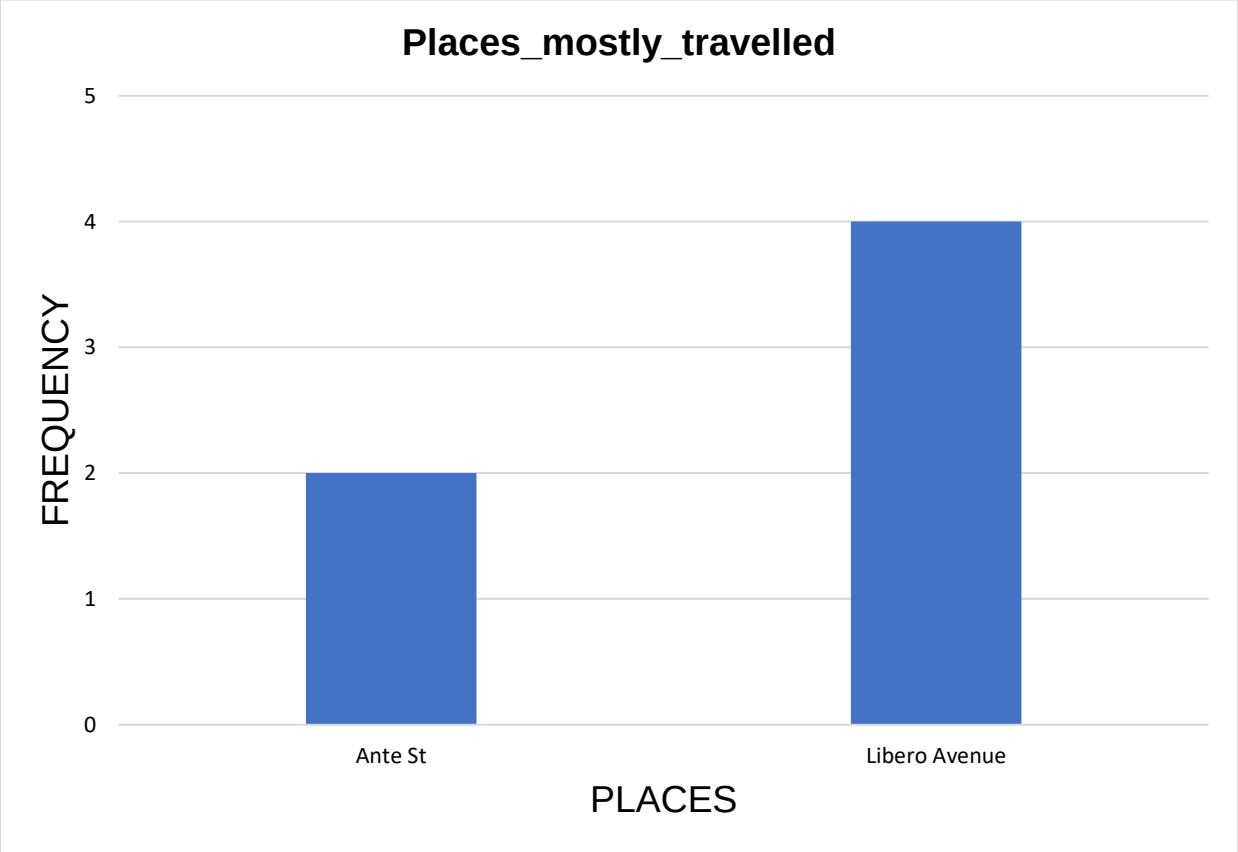
- Passengers - Passenger who have taken ride in last 2 months

```
mysql> SELECT distinct U.user_name as 'Active_Passengers'
    -> FROM User U,Passenger P,Ride_Taken RT
    -> WHERE U.user_id = P.user_id and
    -> P.passenger_id = RT.passenger_id and
    -> RT.booking_time_stamp BETWEEN DATE_SUB(NOW(), INTERVAL 60 DAY) AND NOW();
+---------------------+
| Active_Passengers   |
+---------------------+
| Astra Everett       |
| Kaye Bird           |
| Virginia Cunningham |
+---------------------+
3 rows in set (0.04 sec)
```

2. Places travelled frequently

```
mysql> Select R.ride_destination, Count(R.ride_id) As 'Places_mostly_travelled'
    -> FROM Ride R
    -> GROUP BY R.ride_destination
    -> HAVING Places_mostly_travelled > (
    -> SELECT AVG(S.Total)
    -> FROM
    -> (Select Count(R.ride_id) AS 'Total'
    -> FROM Ride R
    -> GROUP BY R.ride_destination)S);
+------------------+-------------------------+
| ride_destination | Places_mostly_travelled |
+------------------+-------------------------+
| Ante St.         |                       2 |
| Libero Avenue    |                       4 |
+------------------+-------------------------+
2 rows in set (0.00 sec)
```

GRAPHICAL REPRESENTATION



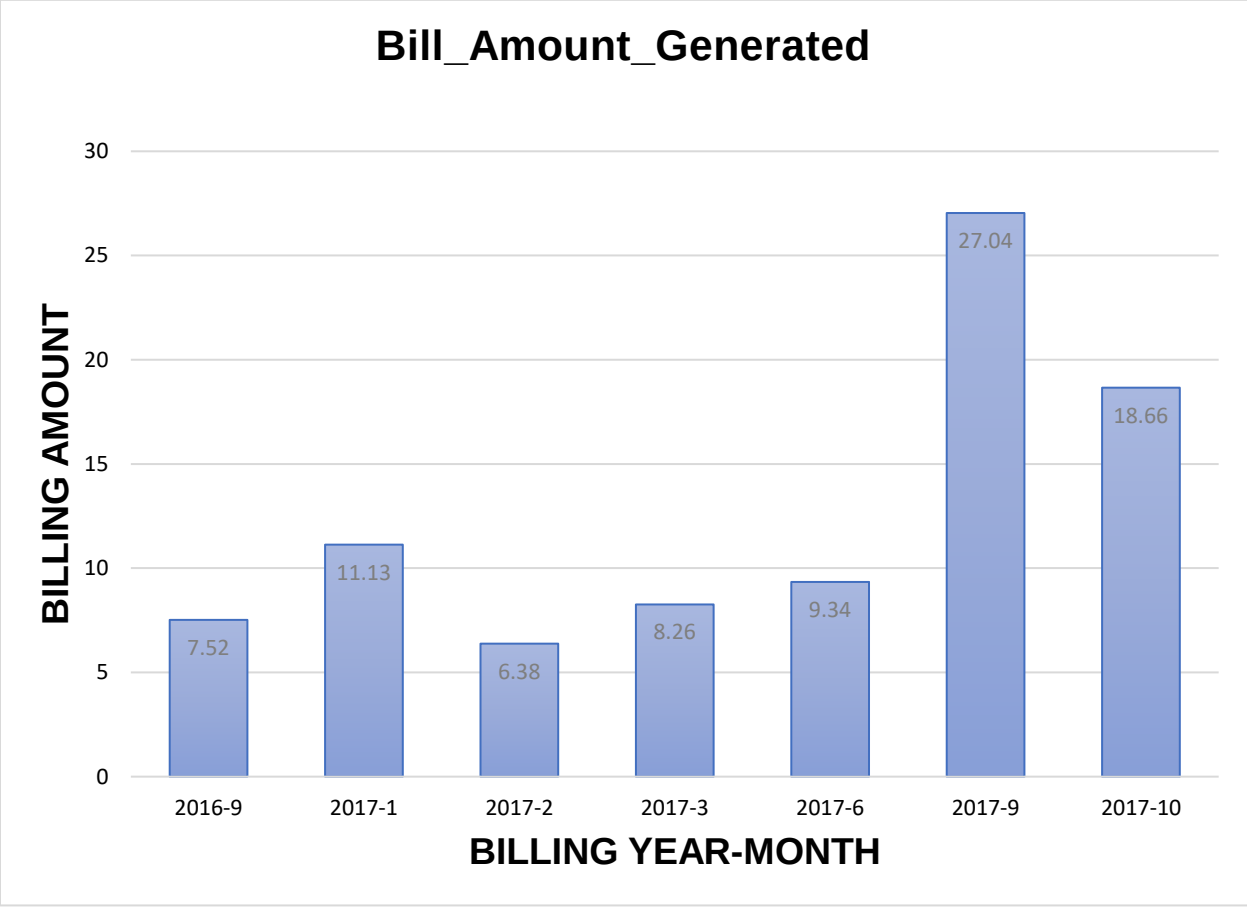
3. Billing

-per month/per day/per year

```
mysql> SELECT CONCAT(YEAR(booking_time_stamp),'-',MONTH(booking_time_stamp)) as 'Booking_Period',SUM(`ride_fare`) as 'Bill_Amount_Generated'
    -> FROM Ride_Taken
    -> GROUP BY Booking_Period
    -> ORDER BY Booking_Period;
+----------------+------------------------+
| Booking_Period | Bill_Amount_Generated |
+----------------+------------------------+
| 2016-9         |                   7.52 |
| 2017-1         |                  11.13 |
| 2017-10        |                  18.66 |
| 2017-2         |                   6.38 |
| 2017-3         |                   8.26 |
| 2017-6         |                   9.34 |
| 2017-9         |                  27.04 |
+----------------+------------------------+
7 rows in set (0.00 sec)
```

(We have considered per month)

GRAPHICAL REPRESENTATION



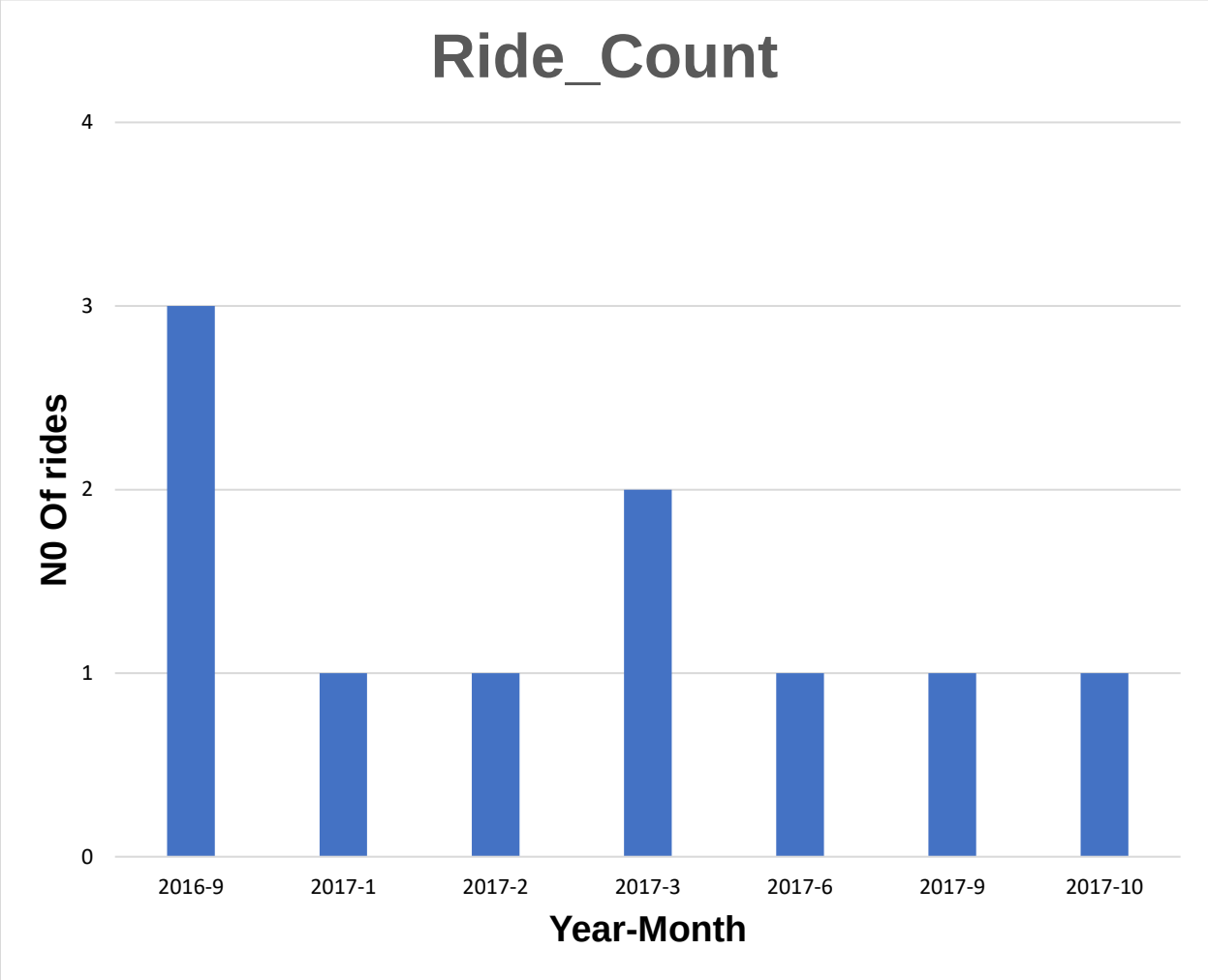
4. Rides

-Offered every month/day/year

```
mysql> SELECT CONCAT(YEAR(ride_date),'-',MONTH(ride_date)) as 'Ride_Period',Count(`ride_id`) as 'Ride_Count'
    -> FROM Ride
    -> GROUP BY Ride_Period
    -> ORDER BY Ride_Period desc;
+-------------+------------+
| Ride_Period | Ride_Count |
+-------------+------------+
| 2017-9      |          3 |
| 2017-6      |          1 |
| 2017-3      |          2 |
| 2017-2      |          1 |
| 2017-10     |          1 |
| 2017-1      |          1 |
| 2016-9      |          1 |
+-------------+------------+
7 rows in set (0.00 sec)
```

(We have considered per month)

GRAPHICAL REPRESENTATION



12.PROJECT SUMMARY

12.1 Summarize your experience with this exercise.

Our aim for this project was to design and implement a database for a carpooling app. Numerous such apps already exist in the market and are generating a lot of data. This data needs to be captured properly using a database and can be queried and analyzed using a DBMS.

While designing the database for our app we have applied all concepts that we learnt in class. We have always studied DBMS as a subject focused on query languages but with this course we had a profound learning of how a database is actually designed.

12.2 What was the hardest part of this project?

Designing the schema, populating the data, adding the Primary and Foreign keys were the hard part of this project.

But the hardest part of the project was making the UML and deciding the content and queries we wanted to in the project. We were trying to put into our project the concept taught in class, and run queries on basis of that. It was challenging. For the UML assigning the cardinalities and avoiding redundancies was the hardest part of this project.

12.3 What problems did you run against in this project?

1. Finalizing the Conceptual Model was difficult. There were certain other tables that could have been considered but then we had to restrict it here to not complicate the model.
2. Managing time between project was also a challenge
3. Deciding on the metrics to be presented on the slide was difficult.
4. We realized that certain additional columns were required while doing the queries to get our desired outputs.

12.4 How did you solve these problems?

1. We took carpool rides to better understand the use cases.
2. We worked together in a team, divided the tasks. We also met regularly to discuss the problems we were encountering and solved them.
3. We took guidance of Prof Agarwal to help us throughout the project specially while doing UML model.
4. We also kept refining our physical model to accommodate new columns which helped us to get better reports.

12.5 If you were to do this project again, what methodology would you follow?

1. We would think more in depth about the columns each table should contain and how these columns could be used for analytics.
2. The data dump file had to be changed as per certain requirements and having it version controlled would be a good idea.
3. Generate more data which would help us get a greater number of metrics.
4. Implementing the triggers in the project.

12.6 Suggestions how to refine this project for the next class?

The project is an ideal way by which all concepts taught in theory can be applied practically. We found the project really challenging and a great learning experience and we think that the project is apt. Only suggestion would be getting a better insight on how to choose metrics for the project to make sure that the necessary details are captured while designing the database.