Daxuan Shu
2*******1
CS 143
Spring 2018

Homework 1

```
a)
    CREATE TABLE Bird_Scooter (
        scooter_id                      SMALLINT ,
            -- 32,767 > 10,000
        flag                            ENUM['online','offline','lost'],
            -- True means online, Flase means offline and Null means lost/stolen
        home_location                   VARCHAR(50) NOT NULL ,
            -- a home location that rarely changes
        PRIMARY KEY (scooter_id)
    )

    CREATE TABLE User (
        user_id                         MEDIUMINT ,
            -- 8,388,607 > 500,000
        credit_card_number              VARCHAR(20) ,
            -- at most 20 digits for credit card number, null if do not have a credit card on file.
        expiration_date                 DATE ,
        email_addr                      VARCHAR(100) ,
        PRIMARY KEY (user_id)
    );

b)
    CREATE TABLE Trip (
        trip_id                         VARCHAR(20) ,
            -- to identify a trip
        user_id                         MEDIUMINT NOT NULL,
            -- Who is using the scooter in this trip
            -- Foreign Key
        scooter_id                      SMALLINT NOT NULL,
            -- 32,767 > 10,000
            -- Foreign Key
        time_start                      TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP ,
            -- time the trip start in format like 2018-04-11 08:00:00
        time_end                        TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP ,
            -- time the trip end in format like 2018-04-11 08:00:00
        start_latitude                  DECIMAL(12,9) NOT NULL,
        start_longtitude                DECIMAL(12,9) NOT NULL,
        end_latitude                    DECIMAL(12,9) NOT NULL,
        end_longtitude                  DECIMAL(12,9) NOT NULL,
        PRIMARY KEY (trip_id)
    );
```

c)
Method I:
Advantages: Since the modification process is done from our server, we can make sure its security and accuracy.

Disadvantages: The server might have a large workload during the rush hours of using scooters. Meanwhile, we have to keep opening the server all the time to write

to our database. If the server somehow accidently shut down, we might loss data and the ability to write to database.
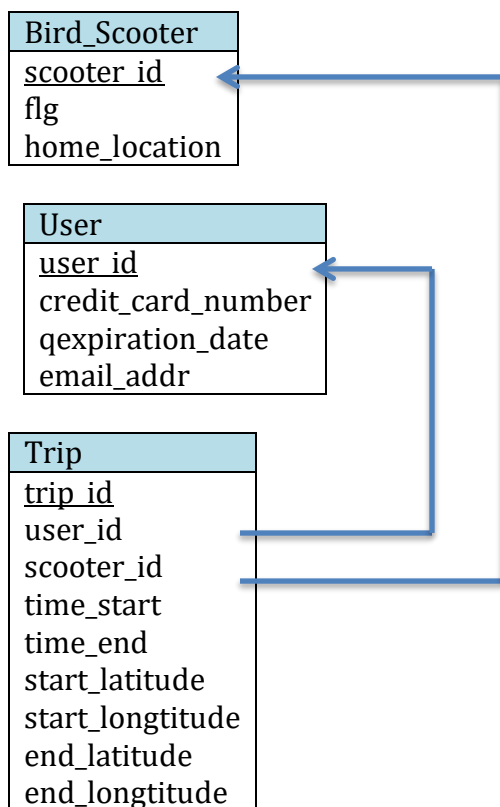
Method II:
Advantages: This method relieves the pressure for our server because the server does not need to open all the time. It is easier for us to maintain the server.

Disadvantages: Since we cache the ride data on the phone, the users might write to the cache data and make some changes to the cache data. This will cause potential data integrity issues.

If I was an employee at Bird, I prefer the first method. Because this method is safe and most server has a probability >95%'s stable state.

Better way, instead of caching the trip data on the user's phone, we can cache the data on the scooter. Since the scooter is our company's device and it is controlled by us. It is safer than user's phone to store the data.

d)

Bird_Scooter
scooter_id
flg
home_location

User
user_id
credit_card_number
qexpiration_date
email_addr

Trip
trip_id
user_id
scooter_id
time_start
time_end
start_latitude
start_longtitude
end_latitude
end_longtitude

Some other things
I wound not. Because we can calculate the number of minuets by the time_start and time_end attibutes, which can save some space.

Including charge
Pros: Easy to access, easy for statistics
Cons: waste space, not safe

Part 2:

a)

```
SELECT
        HOUR(DateTime) AS hour ,
        SUM(Throughput) AS trips
FROM
        rides2017
GROUP BY hour;
```

b)

```
SELECT
        Origin,
        Destination
FROM
        rides2017
WHERE (WEEKDAY(DateTime) < 5)

GROUP BY Origin, Destination
ORDER BY SUM(Throughput) DESC
LIMIT 1;
```

c)

```
SELECT
        Destination,
        AVG(Throughput) AS averages
FROM
        rides2017
Where (WEEKDAY(DateTime) = 0 && HOUR(DateTime) > 7 && HOUR(DateTime) <
9 || (HOUR(DateTime) = 10 && MINUTE(DateTime) = 0 && SECOND (DateTime) =
0))
GROUP BY Destination
ORDER BY averages DESC
LIMIT 5;
```

d)
```
SELECT DISTINCT
        Origin
FROM(
SELECT
        Origin,
        HOUR(DateTime) AS hour,
        MAX(Throughput) AS max_tp,
        AVG(Throughput) AS avg_tp
FROM
        rides2017
GROUP BY Origin, hour
)aggregate
WHERE aggregate.max_tp / 100 > aggregate.avg_tp;
```

e)$\Pi_{hour,trips/100}(\sigma_{(hour\geq7 \wedge hour < 10)\vee(hour \geq 17 \vee hour < 19)}(hourly\_ridership))$

f)
$\Pi_{Station,DateTime,Riders,Condition}(\sigma_{weather.Condition="sunny" \vee weather.Condition = "cloudy"}(Occupancy \bowtie Weather))$