Computer Science 143, Homework 1

Michael Wu
UID: 404751542

April 18th, 2018

Part 1

a)

```
CREATE TABLE scooter (
  id INT PRIMARY KEY NOT NULL,
  online BOOLEAN NOT NULL,
  homeLocation VARCHAR(100) NOT NULL
);

CREATE TABLE user (
  id INT PRIMARY KEY NOT NULL,
  creditCard CHAR(16),
  expiration DATE,
  email VARCHAR(100) NOT NULL
);
```

b)

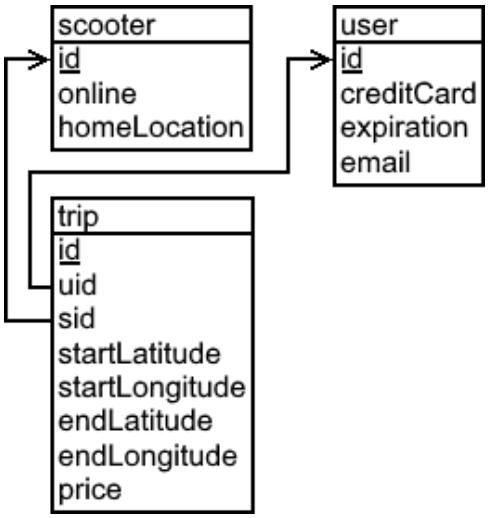
```
CREATE TABLE trip (
  id INT PRIMARY KEY NOT NULL,
  uid INT NOT NULL,
  sid INT NOT NULL,
  startLatitude DOUBLE NOT NULL,
  startLongitude DOUBLE NOT NULL,
  endLatitude DOUBLE NOT NULL,
```

```
  endLongitude DOUBLE NOT NULL,
  startTime DATETIME NOT NULL,
  endTime DATETIME NOT NULL,
  price DECIMAL(20,2) NOT NULL,
  FOREIGN KEY uid REFERENCES user(id),
  FOREIGN KEY sid REFERENCES scooter(id)
);
```

The attributes `uid` and `sid` are foreign keys.

c) The first way of inserting at the beginning of a trip and updating later has the advantage that a record of the trip is created as soon as the trip begins. So even if a phone fails to communicate with the server anymore, the company will know that a trip occurred and could possibly charge the user a fee. However it will require changing the database, which means it could require increased processing power. The second way of inserting at the end of the trip has the advantage that no updates will be necessary in the database, reducing the amount of work that the database has to do. However, since trip information is cached, if a phone crashes trip information may be lost. I would prefer the first way so that we can keep track of partial information and avoid losing data. An alternative solution would be to create a `startTrip` table and an `endTrip` table that contain the relative information generated at the start and end of the trip. This would make it so that trip information could be split up in a more intuitive way.

d) The table schema is shown below.



Part 2

a)

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

b)

```
SELECT Origin, Destination
FROM rides2017
WHERE DAYOFWEEK(DateTime)>1 AND DAYOFWEEK(DateTime)<7
GROUP BY Origin, Destination
ORDER BY SUM(Throughput) DESC
LIMIT 1;
```

c)

```
SELECT Destination, AVG(Throughput) AS Average
FROM
  (SELECT Destination, SUM(Throughput) AS Throughput
  FROM rides2017
  WHERE DAYOFWEEK(DateTime)=2
    AND HOUR(DateTime)>=7
    AND HOUR (DateTime)<10
  GROUP BY Destination, DATE(DateTime))
  AS mondayRushHourThroughput
GROUP BY Destination
ORDER BY Average DESC
LIMIT 5;
```

d)

```
SELECT Origin
FROM
  (SELECT Origin, SUM(Throughput) AS Throughput
  FROM rides2017
  GROUP BY Origin, DateTime) AS hourlyThroughput
GROUP BY Origin
HAVING MAX(Throughput)>100*AVG(Throughput);
```

e)

$$\pi_{\text{hour,trips}/100}(\sigma_{(\text{hour}>=7 \wedge \text{hour}<10) \vee (\text{hour}>=17 \wedge \text{hour}<19)}(\text{hourly\_ridership}))$$

f)

$$\pi_{\text{Station,DateTime,Riders,Condition}}(\sigma_{\text{Condition=sunny} \vee \text{Condition=rainy}}(\text{Occupancy} \bowtie \text{Weather}))$$