

Title: Populating and Querying Citi Bike Database with SQL

Client: Mayor of New York City, Bill de Blasio

Objective: Help the mayor get a better understanding of Citi Bike ridership by answering a few queries using SQL. Through these queries, we hope to gain insights on the usage of Citi Bikes in the month of October and provide suggestions to the mayor on how to improve

The Citi Bike data for every month starting September 2013 is freely available to download on <https://www.citibikenyc.com/system-data> . Following is the information that is recorded every month.

Trip Duration (seconds)	How long a trip lasted
Start Time and Date	Self-explanatory
Stop Time and Date	Self-explanatory
Start Station Name	Self-explanatory
End Station Name	Self-explanatory
Station ID	Unique identifier for each station
Station Lat/Long	Coordinates
Bike ID	unique identifier for each bike
User Type (Customer = 24 day pass user; Subscriber = Annual Member)	hour pass or 3 Customers are usually
tourists, subscribers are usually NYC residents	
10. Gender (Zero=unknown; 1=male; 2=female)	Usually
unknown for customers since they often sign up at a kiosk	
11. Year of Birth	Self entered, not validated by an ID.

Note that -

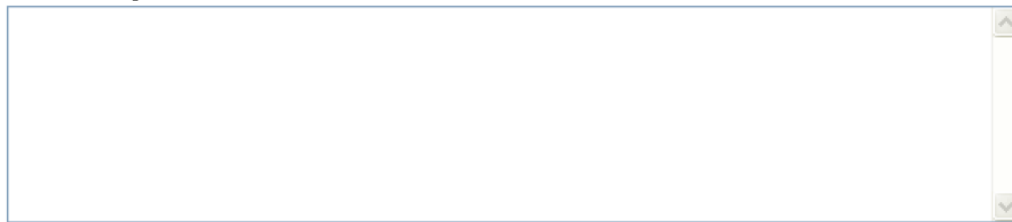
- Trip count and mileage estimates include trips with a duration of greater than one minute.
- Mileage estimates are calculated using an assumed speed of 7.456 miles per hour, up to two hours. Trips over two hours max-out at 14.9 miles. Once you opt into Ride Insights, the Citi Bike app will use your phone's location to record the route you take between your starting and ending Citi Bike station to give exact mileage.
- We only include trips that begin at publicly available stations (thereby excluding trips that originate at our depots for rebalancing or maintenance purposes).

1. **Project Description:** In this project, you will create a database for New York' Citi Bikes. The system maintains the information about trips, stations, and users. You are tasked with creating and populating the Citi Bikes database with the information available in the Excel CSV file. Based on your understanding of the real world, decompose the given relation in the excel sheet to achieve a high normal form (desired forms will be 3NF or BCNF). Since there is a lot of data, you can work with only a subset (to be discussed in the class).

2. **Access to PHP and MySQL:** All CS students at SSU have a personal space (blue.cs.sonoma.edu/~yourSSUID) on the Blue web server which has PHP support. You will need to upload your project files to the web server for your peers and the instructor to view and access them. The instructor will create an account on Blue's MySQL server for each student and login credentials will be available on Canvas soon.
3. **Populating the Database:** Once the given relation has been normalized, you will need to populate the normalized tables with the data available via Excel files. You may optionally first create your user interface and populate the database through your interface.
4. **Interface Implementation (50 pts):** Implement an interface by using PHP and HTML to query and modify the data. A simple example is as shown in the following figure, which includes a text box to accept a SQL statement and then submit it to the MySQL database. You will be required to provide SQL code for the 10 queries given in a later section.

Database Query Form by John Doe

Please input SQL Statement here



Submit Clear

Interface Requirements:

- i. Your interface should not accept SQL DROP statements.
 - ii. For any other SQL statements, your interface should not only accept it, but also return the execution result. For example, a select statement will return the query results (including the attribute name for each column) and the number of rows retrieved. A create/delete/update/insert statement will display "Table Created/Updated", "Row Inserted" or "Row(s) Deleted" messages on your interface.
 - iii. An error message is displayed if an incorrect SQL statement is submitted.
 - iv. You should have a title for this interface, indicating your team name.
5. **Executing SQL queries (30 pts):** First, you need to write the SQL statements for the following queries. Then, you must test each of them through your interface to ensure that these generate the correct result.
- i. Top 5 stations with the most starts (showing # of starts)
 - ii. Number of male users over 50 years of age
 - iii. Trip duration by user type

- iv. Most popular trips based on start station and stop station
- iv. Rider performance by Gender and Age based on avg trip distance (station to station), median speed (trip duration / distance traveled).
- v. What is the busiest bike in NYC in October 2018? How many times was it used? How many minutes was it in use?
- vi. Number of round trips (trips that start and end on the same station)
- vii. Ratio of subscribers to regular customers
- viii. Number of female teenagers that used bikes.
- ix. Number of broken bikes.

(Assumption: A bike may be broken if the where a trip lasted less than 90 seconds and the start station == end station. 90 seconds is an arbitrary choice based on how long it would take a rider to realize a bike isn't working properly and coming back to the station to return it and take a new one.)

- x. How many Day-Pass users, not subscribers, must have paid additional fee? Your query should return the number of people and not the amount they were charged.

(Assumption: The first 30 minutes of each ride is included for Day Pass users. If they want to keep a bike out for longer, it's only an extra \$4 for each additional 15 minutes.)

6. Report (30 pts): Through a brief write (no more than 1 page), please describe your

- i. Normalization strategy and why do you think it is the best. Is the decomposed relation lossless and dependency preserving?
- ii. Implementation strategy. In specific, how did you start developing this application.
- iii. What were some challenges faced by you and how did you overcome those as a team?

7. Material to submit: Please submit the following in a zip file YourGroupID.zip to Canvas by the deadline.

- i. URL of your PHP/HTML interface in a file named url.txt. Please make sure the interface can be used to query the database. The instructor and three groups will use the interface to verify that the database has been populated with correct schema and data.
- ii. Your code for implementing the interface.
- iii. The SQL statements to create and populate your tables in queries.txt. Please have queries numbered and in the same order as queries given.