***Dataset***

The dataset provided is from the second year of Bay Area Bike Share's operation. Files contain data about bike trips starting from 2014-9-1 to 2015-08-31. Both files in the dataset are comma delimited and do not contain header rows.

* *station\_data.csv*

|  |  |
| --- | --- |
| ***Column Number*** | ***Column Definition*** |
| 1 | Station ID number |
| 2 | Name of station |
| 3 | Latitude |
| 4 | Longitude |
| 5 | Number of total docks at station |
| 6 | City (San Francisco, Redwood City, Palo Alto, Mountain View, San Jose) |
| 7 | Original date that station was installed |

* *trip\_data.csv*

*-*

|  |  |
| --- | --- |
| ***Column Number*** | ***Column Definition*** |
| *1* | Numeric ID of bike trip |
| *2* | Duration of trip in seconds |
| *3* | *S*tart date of trip with date and time |
| *4* | *S*tation name of start station (corresponds to 'Column 2' in the station\_data.csv dataset) |
| *5* | Numeric reference for start station (corresponds to 'Column 1' in the station\_data.csv dataset) |
| *6* | End date of trip with date and time |
| *7* | *S*tation name for end station (corresponds to Column 2 in the station\_data.csv dataset) |
| *8* | Numeric reference for end station (corresponds to 'Column 1' in the station\_data.csv dataset) |
| *9* | Numeric ID of bike used |
| *10* | Customer type.  'Subscriber' = annual or 30-day member;  'Customer' = 24-hour or 3-day member |
| *11* | Home zip code of subscriber (customers can choose to manually enter zip at kiosk however data is unreliable) |

**SECTION 1: SQL**

Please complete this part of the test using a word processor. You can use any flavor of SQL you feel comfortable with, although PostgreSQL is preferred. Please indicate the SQL flavor you use in a comment in your work.  
Save your answers in a .sql file <your\_name.sql>

1. Write the DDL statements to create a database named 'bikeshare' and the schema for the two datasets above. Name one of the tables *stations* and the other *trips*. You can name fields as you see fit. Skip any index, foreign key or permission definitions.
2. Write the command/script to load the data in the csv files into the relevant tables of the database.

Write the SQL queries to extract the information requested below. Note that Data Engineers typically have only read-access to the data sources and are not allowed to create temporary tables. Avoid making use of temporary tables.

1. The id of the ten 'most popular' bikes, i.e. the bikes that have made the highest number of trips and the numbers of corresponding trips.
2. The number of trips originating from each city. Your output should include the city name and the number of trips originating from it.
3. The number of trips crossing cities, i.e. trips that originate in one city and end in another. Your output should include the originating and ending city names and the number of trips between them.
4. The date and bike id of the most recent ride for each individual bike.

**SECTION 2: R or PYTHON**

Please complete this part of the test using R or Python. Save your answers in either a .py or .R file with your name as the file name. We should be able to run your script from start to finish to get the answers for each question.

1. Read in the csv files
2. Clean up the trip data by removing trips that start and end at the same station within 60 seconds.
3. Add an indicator column to the trip data and populate it with ‘1’ for inter-city trips (as described in question 5) and ‘0’ for intra-city (within the city) trips.
4. Create a table comparing cities by:
   * The number of stations located in that city,
   * The total number of intra-city (within the city) trips,
   * Average duration of intra-city (within the city) trips,
   * Total duration of intra-city (within the city) trips in bike-hours.
5. Plot a graph of number of trips per month broken down by customer type.
6. Plot a graph of average duration of trips on weekdays vs. weekends, broken down by customer type.