DataEng Project Assignment 2 Submission Document

Construct a table showing each day for which your pipeline successfully, automatically processed one complete day's worth of sensor readings. The table should look like this:

Date	Day of Week	# Sensor Readings	# rows added to your database
08/05/2023	Monday	264505	264505
09/05/2023	Tuesday	163369	163369
10/05/2023	Wednesday	216463	216463
11/05/2023	Thursday	193455	193455
12/05/2023	Friday	288022	288022
13/05/2023	Saturday	256467	256467
14/05/2023	Sunday	251352	251352

Documentation of Each of the Original Data Fields

For each of the fields of the breadcrumb data, provide any documentation or information that you can determine about it. Include bounds or distribution data where appropriate. For example, for something like "Vehicle ID", say something more than "It is the identification number for the vehicle". Instead, add useful information such as "the integers in this field range from <min val> to <max val>, and there are <n> distinct vehicles identified in the data. Every vehicle is used on weekdays but only 50% of the vehicles are active on weekends."

EVENT_NO_TRIP
This field represents the event number of the trip. It is a unique identifier for each trip in the data. It is always an 8 digit integer. The current data in tables have it ranging from 228871317 to 233137897. There are a total of 3990 distinct trips identified in the data.

EVENT_NO_STOP
This field represents the event number of the trip. It is a unique identifier for each trip in the data. It is always a 9 digit integer. The current data in tables have it ranging from 228871319 to 233137914. There are hundreds of thousands of distinct stops identified in the data.

OPD_DATE
This field represents the date of the operation. It provides the date on which the trip occurred. The values in this field are in the format 'DDMONYYYY:HH:MI:SS'. The current data in tables have it ranging from 30DEC2022:00:00:00 to 05JAN2023:00:00:00. This does not give any information about time, the value is 00:00:00 for all readings.

VEHICLE_ID
This field represents the identification number of the vehicle. It uniquely identifies each vehicle in the data. It is always a 4 digit integer. The current data in tables have it ranging from 2910 to 4303. There are a total of 66 distinct vehicles identified in the data.

METERS
This field represents the distance covered in meters. It indicates the distance traveled between breadcrumb readings. The values in this field can be integers. The values in this field in the current data may range from 0 to 500,000. The number does not indicate the total distance covered by the vehicle in a single trip, but rather helps further in calculating the distance covered by it in each trip or over the course of many trips during the whole day.

ACT_TIME
This field represents the time of the breadcrumb reading. It provides information on when the breadcrumb reading was recorded. It is not a timestamp field by itself. It represents the number of seconds which have passed since 12 AM of a particular date. The values are always integers. The values in this field may range from 0 to 100,000 where any value greater than 86,400 means that it has crossed 24 hours from OPD_DATE of that reading. This value can further be used to calculate the actual timestamp of the breadcrumb reading.

GPS_LONGITUDE
This field represents the longitude coordinates obtained from GPS. It indicates the east-west position of the vehicle at the time of the breadcrumb reading. The values in this field are in decimal degrees format. The values in this field for the data in the tables range between -124 and -122, providing the bounds of the geographic area covered by the data. The negative sign in the value suggests that the vehicle is in the western hemisphere.

GPS_LATITUDE
This field represents the longitude coordinates obtained from GPS. It indicates the north-south position of the vehicle at the time of the breadcrumb reading. The values in this field are in decimal degrees format. The values in this field for the data in the tables range between 45 and 46, providing the bounds of the geographic area covered by the data. The positive sign in the value suggests that the vehicle is in the northern hemisphere.

GPS_SATELLITES
This field represents the number of GPS satellites used for obtaining the breadcrumb reading. It indicates the strength of the GPS signal during the reading. The number of satellites used can

vary for different breadcrumb readings. The values in this field can be integers. The values for the current data in the tables range from 0 to 12.

GPS_HDOP
This field represents the Horizontal Dilution of Precision (HDOP) obtained from GPS. The values in this field can be floats or decimals. It indicates the accuracy of the GPS horizontal positioning. The values in this field can vary, and a lower value indicates higher accuracy. The values for current data in the table range from 0 to 26.

Data Validation Assertions

List 20 or more data validation assertion statements here. These should be English language sentences similar to “The speed of a TriMet bus should not exceed 100 miles per hour” . You will only implement a subset of them, so feel free to write assertions that might be difficult to evaluate. Create assertions for all of the fields, even those, like GPS_HDOP, that might not be used in your database schema.

1. Validate that the latitude values fall within the appropriate range for latitude coordinates
2. Ensure that each breadcrumb reading has a valid trip_id that references an existing trip in the Trip table.
3. Check that the speed values are non-negative and within a reasonable range for bus speeds.
4. Ensure that the tstamp column has a valid timestamp value for each breadcrumb reading.
5. OPD_DATE is a string and in the format "DD MMM YYYY:HH:MM:SS".
6. Each event_no_trip should have only one unique vehicle_id.
7. Vehicle_id should be int type and 4 digits.
8. Ensure that service_key is one of the specified valid values ('Weekday', 'Saturday', or 'Sunday')
9. Year should be within 2022 and 2023.
10. GPS_LONGITUDE is a float between -180 and 180,GPS_LATITUDE is a float between -90 and 90.
11. At Least 50 trips for every vehicle in a day.
12. Every trip id will have a length of 9 digits and starts with digit 2.
13. For every vehicle there will be a vehicle id and trip id.
14. More than 50% of vehicles speed will be under 15 mph.
15. Every trip will have different stops which recorded details at different locations (latitude and longitude).
16. Direction of each trip which took place cannot be anything other than 0 or 1.
17. Hundreds of trips take place each day.
18. Meters covered by any vehicle cannot be negative.
19. There cannot exist a trip without a vehicle id.
20. GPS_HDOP values are always decimals within the range of 0 to 24.

Data Transformations

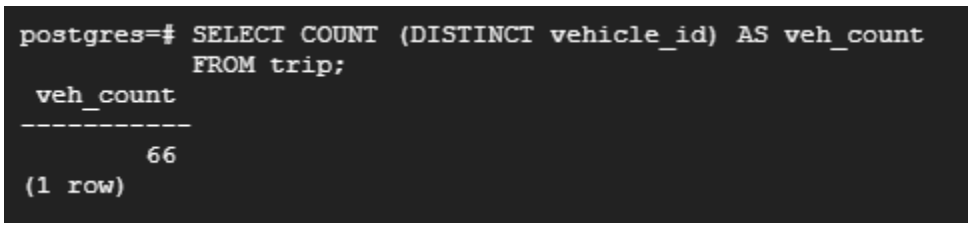
Describe any transformations that you implemented either to react to validation violations or to shape your data to fit the schema. For each, give a brief description of the transformation along with a reason for the transformation.

- We have transformed a few columns from the consumed data like OPD_DATE and ACT_TIME and combined them to get a new column TIMESTAMP in a meaningful way and calculated the speed of each vehicle using METERS and TIMESTAMP values.
- Populated speed column by 0 by default before actual speed calculation to avoid NaN speed value for the first record of every trip.
- Used math.inNan to make sure we do not divide by NaN or null value to avoid undefined or infinite values in speed columns.

Example Queries

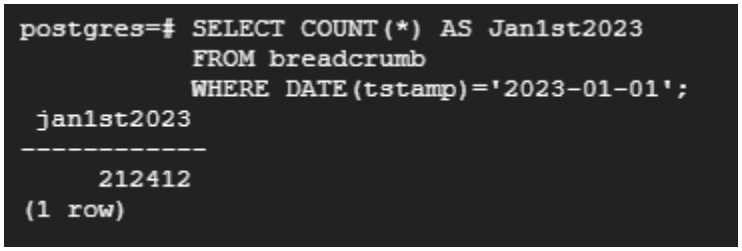
1. How many vehicles are there in the TriMet system?

   SELECT COUNT (DISTINCT vehicle_id) AS veh_count
   FROM trip;



2. How many breadcrumb reading events occurred on January 1, 2023?

   SELECT COUNT(*) AS Jan1st2023
   FROM breadcrumb
   WHERE DATE(tstamp)='2023-01-01';



3. How many breadcrumb reading events occurred on January 2, 2023?

   SELECT COUNT(*) AS Jan2nd2023
   FROM breadcrumb
   WHERE DATE(tstamp)='2023-01-02';

```
postgres=# SELECT COUNT(*) AS Jan2nd2023
           FROM breadcrumb
           WHERE DATE(tstamp)='2023-01-02';
 jan2nd2023
------------
     197816
(1 row)
```

4. On average, how many breadcrumb readings are collected on each day of the week?

   SELECT COUNT (*)/7 AS avg_readings
   FROM breadcrumb
   WHERE DATE(tstamp) BETWEEN '2022-12-30' AND '2023-01-05';

```
postgres=# SELECT COUNT (*)/7 AS avg_readings
           FROM breadcrumb
           WHERE DATE(tstamp) BETWEEN '2022-12-30' AND '2023-01-05';
 avg_readings
--------------
       232650
(1 row)
```

5. List the TriMet trips that traveled a section of I-205 between SE Division and SE Powell on January 1, 2023. To find this, search for all trips that have breadcrumb readings that occurred within a lat/lon bounding box such as [(45.497805, -122.566576), (45.504025, -122.563187)].

   SELECT DISTINCT t.trip_id FROM trip t
   JOIN breadCrumb b ON t.trip_id = b.trip_id
   WHERE DATE(b.tstamp) = '2023-01-01'
   AND b.latitude BETWEEN 45.497805 AND 45.504025
   AND b.longitude BETWEEN -122.566576 AND -122.563187;

```
postgres=# SELECT DISTINCT t.trip_id
           FROM trip t
           JOIN breadCrumb b ON t.trip_id = b.trip_id
           WHERE DATE(b.tstamp) = '2023-01-01'
           AND b.latitude BETWEEN 45.497805 AND 45.504025
           AND b.longitude BETWEEN -122.566576 AND -122.563187;
  trip_id
-----------
 229984997
 230003642
 230025626
 230026822
 230117920
 230119964
 230172309
 230185095
(8 rows)
```

6. List all breadcrumb readings on a section of US-26 west side of the tunnel (bounding box: [(45.506022, -122.711662), (45.516636, -122.700316)]) during Mondays between 4pm and 6pm. Order the readings by tstamp. Then list readings for Sundays between 6am and 8am. How do these two time periods compare for this particular location?

   SELECT * FROM breadcrumb
   WHERE EXTRACT (DOW FROM tstamp) = 1
   AND EXTRACT (HOUR FROM tstamp) BETWEEN 16 AND 18
   AND latitude BETWEEN 45.506022 AND 45.516636
   AND longitude BETWEEN -122.711662 AND -122.700316
   ORDER BY tstamp;

```
postgres=# SELECT * FROM breadcrumb
           WHERE EXTRACT (DOW FROM tstamp) = 1
           AND EXTRACT (HOUR FROM tstamp) BETWEEN 16 AND 18
           AND latitude BETWEEN 45.506022 AND 45.516636
           AND longitude BETWEEN -122.711662 AND -122.700316
           ORDER BY tstamp;
      tstamp        | latitude  |  longitude  | speed |  trip_id
--------------------+-----------+-------------+-------+-----------
 2023-01-02 17:58:00 |  45.51312 | -122.702532 |  20.6 | 230519233
 2023-01-02 17:58:00 | 45.512552 | -122.703557 |  20.2 | 230519233
 2023-01-02 17:58:00 | 45.512085 |  -122.70464 |  19.6 | 230519233
 2023-01-02 17:58:00 | 45.514637 | -122.700998 |  20.8 | 230519233
 2023-01-02 17:58:00 | 45.513853 | -122.701723 |  20.6 | 230519233
 2023-01-02 17:59:00 | 45.509053 | -122.708388 |  18.6 | 230519233
 2023-01-02 17:59:00 | 45.508282 | -122.708938 |    19 | 230519233
 2023-01-02 17:59:00 | 45.507137 | -122.710782 |  19.6 | 230519233
 2023-01-02 17:59:00 |  45.50758 | -122.709698 |  19.6 | 230519233
 2023-01-02 17:59:00 | 45.511625 | -122.705677 |    19 | 230519233
 2023-01-02 17:59:00 | 45.511107 | -122.706602 |  18.6 | 230519233
 2023-01-02 17:59:00 | 45.510505 |  -122.70734 |  17.4 | 230519233
 2023-01-02 17:59:00 | 45.509815 | -122.707907 |  17.6 | 230519233
 2023-01-02 18:53:00 | 45.514565 | -122.700653 |  22.8 | 230530162
 2023-01-02 18:53:00 | 45.507522 | -122.709478 |    22 | 230530162
 2023-01-02 18:53:00 |   45.5083 |  -122.7086  |  22.2 | 230530162
 2023-01-02 18:53:00 | 45.509187 | -122.707982 |    22 | 230530162
 2023-01-02 18:53:00 | 45.510112 | -122.707332 |    23 | 230530162
 2023-01-02 18:53:00 | 45.510963 |  -122.70638 |  24.2 | 230530162
 2023-01-02 18:53:00 | 45.511635 |  -122.7051  |  24.6 | 230530162
 2023-01-02 18:53:00 | 45.512213 |  -122.70375 |  24.4 | 230530162
 2023-01-02 18:53:00 | 45.512852 | -122.702468 |  24.4 | 230530162
 2023-01-02 18:53:00 |  45.51368 | -122.701452 |  24.2 | 230530162
 2023-01-02 18:53:00 |  45.50704 | -122.710697 |  22.4 | 230530162
(24 rows)
```

```
SELECT * FROM breadcrumb
WHERE EXTRACT (DOW FROM tstamp) = 0
AND EXTRACT (HOUR FROM tstamp) BETWEEN 6 AND 8
AND latitude BETWEEN 45.506022 AND 45.516636
AND longitude BETWEEN -122.711662 AND -122.700316
ORDER BY tstamp;
```

```
postgres=# SELECT * FROM breadcrumb
           WHERE EXTRACT (DOW FROM tstamp) = 0
           AND EXTRACT (HOUR FROM tstamp) BETWEEN 6 AND 8
           AND latitude BETWEEN 45.506022 AND 45.516636
           AND longitude BETWEEN -122.711662 AND -122.700316
           ORDER BY tstamp;
       tstamp        | latitude  |  longitude   | speed |  trip_id
---------------------+-----------+--------------+-------+-----------
 2023-01-01 06:00:00 | 45.507005 | -122.711178 |  23.2 | 230045092
 2023-01-01 06:00:00 | 45.507425 | -122.709808 |  23.4 | 230045092
 2023-01-01 06:00:00 | 45.508192 | -122.708822 |  22.8 | 230045092
 2023-01-01 06:00:00 | 45.509147 | -122.708142 |  23.6 | 230045092
 2023-01-01 06:00:00 |  45.51007 | -122.707495 |    23 | 230045092
 2023-01-01 06:00:00 |  45.51093 | -122.706577 |  23.8 | 230045092
 2023-01-01 06:00:00 |  45.51214 |  -122.70409 |  23.5 | 230045092
 2023-01-01 06:00:00 |  45.51275 | -122.702773 |  24.4 | 230045092
 2023-01-01 06:01:00 | 45.513562 | -122.701678 |  24.8 | 230045092
 2023-01-01 06:01:00 | 45.514458 | -122.700847 |  23.4 | 230045092
(10 rows)
```

This location has 2 different trips going through it between 4 PM and 6 PM on Monday, whereas just one trip between 6 AM and 8AM on Sunday. Hence the first query is giving out over twice as many readings as the second.

7. What is the maximum velocity reached by any bus in the system?

   SELECT MAX(speed) as max_speed FROM breadcrumb;

```
postgres=# SELECT MAX(speed) as max_speed FROM breadcrumb;
 max_speed
-----------
       236
(1 row)
```

8. List all speeds and give a count of the number of vehicles that move precisely at that speed during at least one trip. Sort the list by most frequent speed to least frequent.

```
SELECT speed, COUNT(speed) AS count
FROM breadcrumb
GROUP BY speed
ORDER BY COUNT(speed) DESC;
```

```
 speed | count
-------+-------
     0 | 59447
    10 | 25460
    11 | 25225
  10.4 | 24957
  10.2 | 24801
```

1544 rows

9. Which is the longest (in terms of time) trip of all trips in the data?

```
SELECT trip_id, trip_duration
FROM (
   SELECT trip_id, MAX(tstamp) - MIN(tstamp) AS trip_duration
   FROM breadcrumb
   GROUP BY trip_id
) AS trip_durations
ORDER BY trip_duration DESC
LIMIT 1;
```

```
postgres=# SELECT trip_id, trip_duration
           FROM (
               SELECT trip_id, MAX(tstamp) - MIN(tstamp) AS trip_duration
               FROM breadcrumb
               GROUP BY trip_id
           ) AS trip_durations
           ORDER BY trip_duration DESC
           LIMIT 1;
  trip_id  | trip_duration
-----------+---------------
 231812405 | 02:52:34
(1 row)
```

10. Find the average speed and total distance traveled for each vehicle during the month of January 2023 and list these results in descending order of average speed.

```
SELECT t.vehicle_id, AVG(b.speed) AS avg_speed, SUM (b.speed) AS distance
FROM trip t
JOIN breadcrumb b ON t.trip_id = b.trip_id
WHERE EXTRACT(YEAR FROM b.tstamp) = 2023
```

```
AND EXTRACT(MONTH FROM b.tstamp) = 1
GROUP BY t.vehicle_id
ORDER BY avg_speed DESC;
```

```
 vehicle_id |     avg_speed      |      distance
------------+--------------------+--------------------
       3721 | 10.907324936370147 | 167132.93999999977
       3724 | 10.388930684283201 | 210729.07000000044
       3212 | 10.275801633063738 | 265536.99000000005
       3414 |  9.655590674385843 | 123417.75999999985
       3015 |  9.635997873245405 | 226542.30999999944
```

66 rows

11. Find the 5 trips with the highest average speeds.

```
SELECT trip_id, AVG(speed) AS average_speed
FROM breadcrumb
GROUP BY trip_id
ORDER BY average_speed DESC
LIMIT 5;
```

```
postgres=# SELECT trip_id, AVG(speed) AS average_speed
           FROM breadcrumb
           GROUP BY trip_id
           ORDER BY average_speed DESC
           LIMIT 5;
  trip_id  |   average_speed
-----------+--------------------
 231174055 | 24.595310344827578
 230117920 |  23.27933333333333
 230118902 | 21.867402597402588
 229984997 | 21.821898734177218
 229615525 |  21.79956790123456
(5 rows)
```

12. Which is the west-most point touched by any vehicle during any trip? When did this occur? Give the corresponding trip id and vehicle id as well.

```
SELECT DISTINCT b.longitude, b.tstamp, b.trip_id, t.vehicle_id
FROM breadcrumb b
JOIN trip t ON b.trip_id = t.trip_id
WHERE b.longitude = (
   SELECT MIN(longitude) AS longitude
   FROM breadcrumb
);
```

```
postgres=# SELECT DISTINCT b.longitude, b.tstamp, b.trip_id, t.vehicle_id
           FROM breadcrumb b
           JOIN trip t ON b.trip_id = t.trip_id
           WHERE b.longitude = (
               SELECT MIN(longitude) AS longitude
               FROM breadcrumb
           );
  longitude  |       tstamp        |  trip_id  | vehicle_id
-------------+---------------------+-----------+------------
 -123.115858 | 2023-01-01 17:09:00 | 229979327 |       3015
 -123.115858 | 2023-01-02 10:33:00 | 230299029 |       3267
(2 rows)
```

Your Code

Provide a reference to the repository where you store your python code. If you are keeping it private then share it with the Professor (bruce.irvin@gmail.com) and TA (mina8@pdx.edu).

Project Repository - https://github.com/kajal8879/Trimet_Project