DataEng: Project Assignment 2 (v 2.0)

Validate, Transform, Enhance and Store

Assignment date: January 26, 2021
Due date: February 14, 2021 @10pm PT
Submit: assignment submission form

By now your pipeline should be automatically gathering and transferring C-Tran breadcrumb records daily but not yet doing much with the data. The next step in building your data pipeline is to get your data in shape for analysis. This includes:

A. understanding the contents of the bread crumb data
B. reviewing the needed schema for the data
C. validating the data
D. transforming the data
E. storing the data into a database server
F. example queries

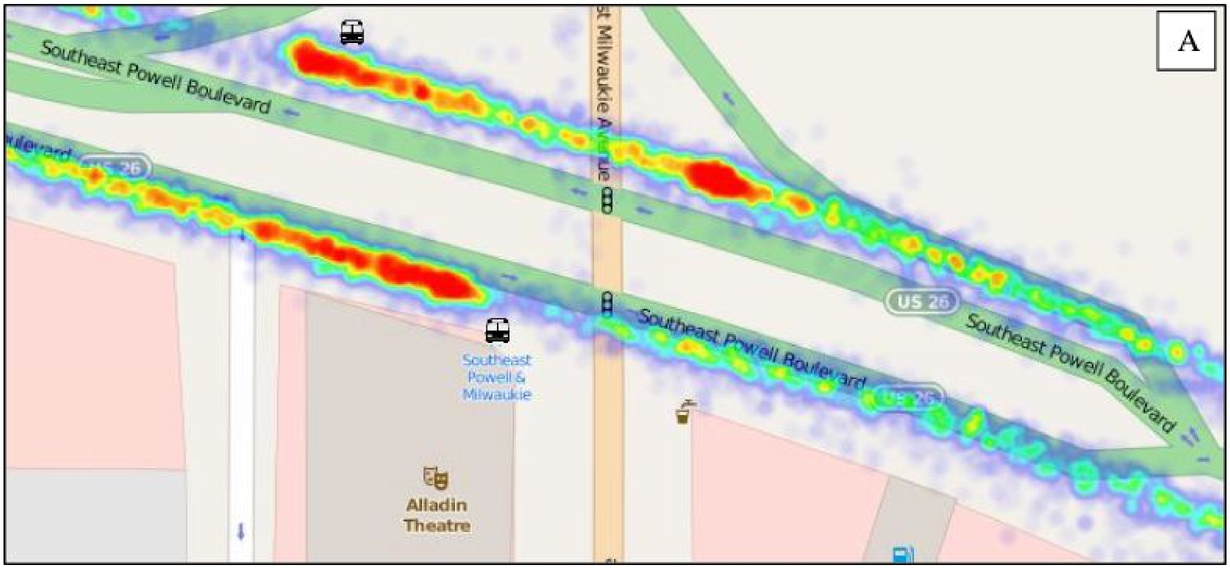
Your job is to enhance your Kafka consumer to perform steps A-E so that your pipeline ingests high-quality data daily into a database server in preparation for visualization.

A. Review the Data

Have a look at a few of the data records and see if you can determine the meaning of each record attribute. Also, have a look at the only documentation that we have for the data (provided to us by C-Tran). The documentation is not very good, so we are counting on you to give better descriptions of each field. In your submission document, provide an improved description of each data field.

B. Database Schema

We plan to develop a visualization application that builds visualizations similar to this:


A
Southeast Powell Boulevard
ulevard
US 26
st Milwaukie Avenue
US 26
Southeast Powell Boulevard
Southeast Powell Boulevard
US 26
Southeast Powell & Milwaukie
Alladin Theatre

This diagram shows vehicle speeds as a heatmap overlaid on a street map for a given latitude/longitude rectangle, route, date range and time range. Colors in the heatmap correspond to average speeds of buses at each GPS location for the selected route, date range and/or time range.

To achieve this we need you to build database tables (or views) that conform to the following schema.

Table: **BreadCrumb**

tstamp	latitude	longitude	direction	speed	trip_id

The BreadCrumb table keeps track of each individual breadcrumb reading. Each row of the BreadCrumb table corresponds to a single GPS sensor reading from a single bus.

tstamp: the moment at which the event occurred, i.e., at which the bus's GPS location was recorded. This should be a Postgres "timestamp" type of column.
//OPD_DATE: Need to add **ACT_TIME to make datetime stamp.**

latitude: a float value indicating fractional degrees of latitude. Latitude and Longitude together indicate the location of the bus at time=tstamp.
//GPS_LATITUDE

longitude: a float value indicating the fractional degrees of longitude. Latitude and Longitude indicate the location of the bus at time=tstamp.
//GPS_LONGITUDE

direction: an integer value in the range 0..359 indicating the forward facing direction of the bus at time=tstamp. Direction is measured in degrees, 0 equals north.
\\DIRECTION

speed: a float value indicating the speed of the bus at time=tstamp. Speed is measured in miles per hour.
\\VELOCITY: Need to convert meter/sec to Miles/hour

trip_id: this integer column indicates the identifier of the trip in which the bus was participating. A trip is a single run of a single bus servicing a single route. This column is a foreign key referencing trip_id in the Trip table.
\\EVENT_NO_TRIP

Table: **Trip**

trip_id	route_id	vehicle_id	service_key	direction

The Trip table keeps track of information about each bus trip. A “trip” is a single run of a single bus over a single route.

trip_id: unique integer identifier for the trip.
\\ EVENT_NO_TRIP from breadcrumbs data

route_id: integer identifying the route that the trip was servicing. For this project we do not have a separate Route table, but if we did, then this would be a foreign key reference to the Route table. Note that you will not have enough information to properly populate this field during assignment #2.

vehicle_id: integer identifying the vehicle that serviced the trip. A trip is serviced by only one vehicle but a vehicle services potentially many trips.
\\VEHICLE_ID

service_key: one of ‘Weekday’, ‘Saturday’, or ‘Sunday’ depending on the day of the week on which the trip occurred OR depending on whether the trip occurred on a Holiday recognized by C-Tran. Note that you will not have enough information to properly populate this field during assignment #2.

direction: either ‘0’ or ‘1’ indicating whether the trip traversed the route from beginning to end (‘0’) or from end to beginning (‘1’). Note that this “direction” has a completely different meaning than the same-named column in the BreadCrumb table (sorry about that). Also note that you will not have enough information to properly populate this field during assignment #2.

See the ER diagram and DDL code for the schema. Please implement this schema precisely in your database so that the visualization tool will work for you.

Note to Winter 2021 Students

The course is new, the project is untested, and we might need to update this data specification again. Sorry about that, we will try to minimize changes and communicate changes via slack or email as well as updates to this document.

C. Data Validation

Create 20 distinct data validation assertions for the bread crumb data. These should be in English (not python). Include them in your submission document (see below).

Then implement at least 10 of the assertions in your Kafka consumer code. Implement a variety of different types of assertions so that you can experience with each of the major types of data validation assertions: existence, limit, intra-record check, inter-record check, summary, referential integrity, and distribution assertions.

Existence assertion

- Every record should have a Event-No-trip
- Every record should have an Event-No-stop.
- The Act_time shouldn't be empty
- Trip id is unique and not null
- Every record should have a latitude value other than 0
- Every record should have a longitude value other than 0
- Every record should have a time value of when the bread crumb was taken
- Every record should have a date value

Limit assertion

- Velocity should be greater than 0 and less than 100
- The Act_time should be greater than 0 and less than 86400
- Dates should be greater than 2020
- Longitude values should be between -180 and 180
- Latitude values should be between -90 and 99

Intra-record assertion

- Every record should have a velocity
- Every record should have a longitude
- Every record should have a latitude
- Every record should have a Direction

Inter-record assertion

- Velocity is distance in metres divided by act_time in seconds(difference from the previous entry)
- The total distance travelled by a vehicle with a unique vehicle id is the distance recorded as meters for the last entry for the day.
- Service key is either weekday, saturday or a sunday to indicate the day of travel

Summary assertion

- Every vehicle should have a unique id
- Every trip should have a unique vehicle id

Statistical assertion

- Max velocity of a vehicle should be greater than the average velocity of the vehicle on any given day
- The min velocity of a vehicle should be less than the average velocity of the vehicle on any given day

D. Data Transformation

Add code to your Kafka consumer to transform your data. Your transformations should be driven by either the need to resolve validation assertion violations or the need to shape the data for the schema. Describe any/all needed transformations in your submission document.

E. Storage in Database Server

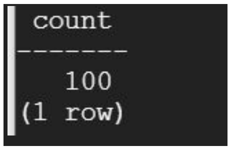
1. Install and configure a PostgreSQL database server on your Virtual Machine. Refer this document to learn how.
2. Enhance your data pipeline to load your transformed data into your database server. You are free to use any of the data loading techniques that we discussed in class. The data should be loaded ASAP after your Kafka consumer receives the data, validates the data and transforms the data so that you have a reliable end-to-end data pipeline running daily, automatically.

F. Example Queries

Answer the following questions about the C-Tran system using your sensor data database. In your submission document include your query code, number of rows in each query result (if applicable) and first five rows of the result (if applicable).

1. How many vehicles are there in the C-Tran system?

   select count(distinct vehicle_id) from trip;

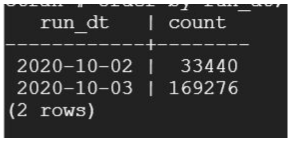
   ```
    count
   -------
      100
   (1 row)
   ```

   Ans: 100

2. How many bread crumb reading events occurred on October 2, 2020?

   select date(tstamp) as run_dt, count(*)
   from breadcrumb
   where date(tstamp) IN ('2020-10-02','2020-10-03')

```
group by date(tstamp)
order by run_dt;
```



```
   run_dt   | count
------------+--------
 2020-10-02 |  33440
 2020-10-03 | 169276
(2 rows)
```

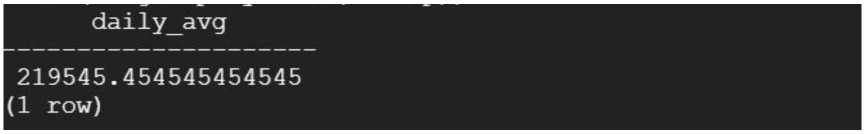
Ans: 33440

3. How many bread crumb reading events occurred on October 3, 2020?

   Ans: 169276

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

```
select avg( A.cnt) as daily_avg
from (
        select date(tstamp) run_dt, count(*) cnt
        from breadcrumb
        group by date(tstamp)) A;
```

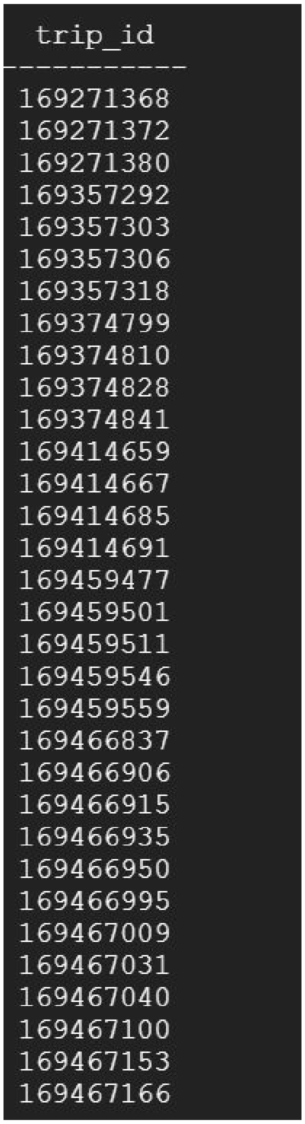


```
      daily_avg
---------------------
 219545.454545454545
(1 row)
```

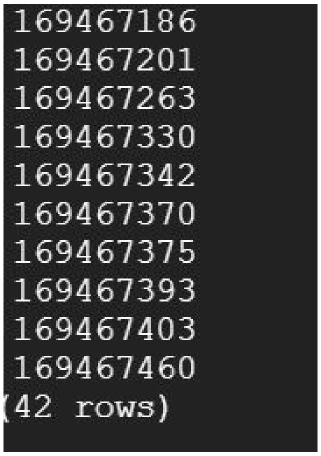
   Ans: 219545.45

5. List the C-Tran trips that crossed the I-5 bridge on October 2, 2020. To find this, search for all trips that have bread crumb readings that occurred within a lat/lon bounding box such as [(45.620460, -122.677744), (45.615477, -122.673624)].

```
select distinct(trip_id)
from breadcrumb
where latitude between 45.615477 and 45.620460
        and longitude between -122.677744 and -122.673624
        and date(tstamp) = '2020-10-02';
```



```
  trip_id
-----------
 169271368
 169271372
 169271380
 169357292
 169357303
 169357306
 169357318
 169374799
 169374810
 169374828
 169374841
 169414659
 169414667
 169414685
 169414691
 169459477
 169459501
 169459511
 169459546
 169459559
 169466837
 169466906
 169466915
 169466935
 169466950
 169466995
 169467009
 169467031
 169467040
 169467100
 169467153
 169467166
```



```
 169467186
 169467201
 169467263
 169467330
 169467342
 169467370
 169467375
 169467393
 169467403
 169467460
(42 rows)
```

6. List all bread crumb readings for a specific portion of Highway 14 (bounding box: [(45.610794, -122.576979), (45.610794, -122.576979)]) 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 count(*)
   from breadcrumb
   where latitude between 45.615477 and 45.610794
   and longitude between -122.576979 and -122.569501
   and to_char(tstamp,'day') = 'monday';
   ```

   Ans: 0

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

   ```
   select max(speed) from breadcrumb;
   ```

   Ans: 319.891

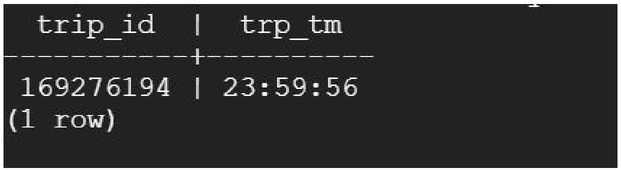
8. List all possible directions and give a count of the number of vehicles that faced precisely that direction during at least one trip. Sort the list by most frequent direction to least frequent.

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

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

   ```
   select A.trip_id, A.trp_tm
   from (
           select trip_id,max(cast(tstamp as time)) - min(cast(tstamp as time))as trp_tm
           from breadcrumb group by trip_id )A
           order by A.trp_tm desc
           fetch first 1 rows only;
   ```

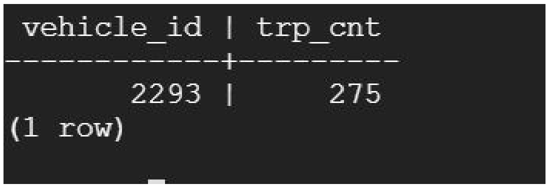
   Ans: 169276194



```
  trip_id   |  trp_tm
-----------+----------
 169276194 | 23:59:56
(1 row)
```

11. Which vehicle is the fastest? “Fastest” in this case should be measured in miles per hour averaged from the beginning of a trip to the end of the trip. That is, the total distance of the trip divided by the total time of the trip. This then should be averaged over all trips that each vehicle serviced.

   Ans : Vehicle 2293 and the distance is 275

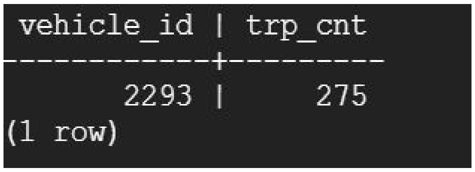


```
 vehicle_id | trp_cnt
------------+---------
       2293 |     275
(1 row)
```

12. Devise three new, interesting questions about the C-Tran bus system that can be answered by your bread crumb data. Show your questions, their answers, the SQL you used to get the answers and the results of running the SQL queries on your data (the number of result rows, and first five rows returned).

(1) which vehicle had the most number of trips by Weekday, Saturday and Sunday

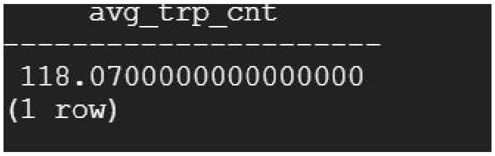
```
select vehicle_id, count(distinct(trip_id)) trp_cnt
from trip
group by vehicle_id
order by trp_cnt desc
fetch first 1 rows only;
```



```
 vehicle_id | trp_cnt
------------+---------
       2293 |     275
(1 row)
```

(2) what is the average number of trips per vehicle by Weekday, Saturday and Sunday

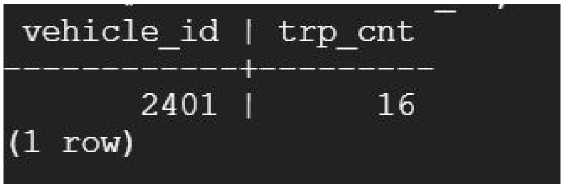
```
select avg(A.trp_cnt) avg_trp_cnt from ( select vehicle_id, count(distinct(trip_id)) trp_cnt from trip group by vehicle_id) A;
```



```
     avg_trp_cnt
----------------------
 118.0700000000000000
(1 row)
```

(3) which vehicle had the least number of trips by Weekday, Saturday and Sunday

```
select vehicle_id, count(distinct(trip_id)) trp_cnt from trip group by vehicle_id
order by trp_cnt fetch first 1 rows only;
```



```
 vehicle_id | trp_cnt
------------+---------
       2401 |      16
(1 row)
```

Submission

Congratulations! Your data pipeline is now working end-to-end.

To submit your completed Assignment 2, create a google document containing the table shown below. Share the document as viewable by anybody at PSU who has the link. You do NOT need to share it with the individual instructors. Then include the URL of the document in the DataEng Project Assignment Submission form.

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	# updates/insertions into your database
9-Feb	Tuesday	371613	369312
10-Feb	Wednesday	369312	370638
11-Feb	Thursday	366506	366610
13-Feb	Saturday	176418	173399
14-Feb	Sunday	135160	130913
16-Feb	Tuesday	376844	240840
17-Feb	Wednesday	373161	358089
18-Feb	Thursday	373161	372923
19-Feb	Friday	385208	362654
20-Feb	Saturday	173849	70298

Documentation of Each of the Original Data Fields

For each of the fields of the bread crumb 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
EVENT_NO_STOP
OPD_DATE
VEHICLE_ID
METERS

ACT_TIME
VELOCITY
DIRECTION
RADIO_QUALITY
GPS_LONGITUDE
GPS_LATITUDE:
GPS_SATELLITES
GPS_HDOP
SCHEDULE_DEVIATION

Data Validation Assertions

List 20 or more data validation assertion statements here. These should be English language sentences similar to “The VELOCITY field
exceeds 5000000”. 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 RADIO_QUALITY) that might not be used in your database schema.

- Every record should have a Event-No-trip
- Every record should have an Event-No-stop.
- The Act_time shouldn’t be empty
- Trip id is unique and not null
- Every record should have a latitude value other than 0
- Every record should have a longitude value other than 0
- Every record should have a time value of when the bread crumb was taken
- Every record should have a date value
- Velocity should be greater than 0 and less than 100
- The Act_time should be greater than 0 and less than 86400
- Dates should be greater than 2020
- Longitude values should be between -180 and 180
- Latitude values should be between -90 and 99
- Every record should have a velocity
- Every record should have a longitude
- Every record should have a latitude
- Every record should have a Direction
- Velocity is distance in metres divided by act_time in seconds(difference from the previous entry)
- The total distance travelled by a vehicle with a unique vehicle id is the distance recorded as meters for the last entry for the day.
- Service key is either weekday, saturday or a sunday to indicate the day of travel
- Every vehicle should have a unique id
- Every Trip must only have one vehicle associated with it.

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.
Timestamp column: This column creation required transformation of OPD_DATE to date format and ACT_TIME to time format. Then the date and time was combined to create a timestamp column.

Speed: Had to change the velocity column from string and float and fill the empty values with 0. Then multiplied the column with 2.237 to convert it to miles/hour.

Creates breadcrumb data frame for Breadcrumb table.

Example Queries

Provide your responses to the questions listed in Section E above. For each question, provide the SQL you used to answer the questions along with the count of the number of rows returned (where applicable) and a listing of the first 5 rows returned (where applicable).

https://github.com/sayeghmutaz-001/Data-Engineering-Prog2

Your Code

Provide a reference to the repository where you store your python code. If you are keeping it private then share it with Bruce (bruce.irvin@gmail.com), David and Aman (github references TBD).

This code needs to be integrated with consumer.py

https://github.com/Raghu-Srungavarapu/dataengineering/tree/main/Project%20Assignment%20
2