WHAT WENT WELL?

Setting up Hadoop environment in local machine seemed intimidating at first, but I found couple resources online such as YouTube videos and Stack Overflow that offered adequate content. Throughout this project, I had done a bunch of trial and error and lot of trouble shooting. My curiosity also peaked due to a couple success HDFS installation over YouTube videos. This had given me a strong motivation that I could deployed the system successfully on my local machine.

Besides, I documented the project properly will help me to pick up latter where I left off. For example, there were couple errors when installing HDFS, I screenshotted and put a side note of the solution. This would remind instantly to pick up where I left off. Another example, I noticed working with virtual machine can be goofy specifically the IP address may change overtime. So, before running Hive, I may want to make sure that the each VMs is pinging

Lastly, I noticed that SQL and HiveQL have language similarities. I found a cheat sheet that I really liked (http://hortonworks.com/wp-content/uploads/2016/05/Hortonworks.CheatSheet.SQLtoHive.pdf) that explicitly compares the difference between the two. I brushed up my SQL skills from Code Academy and helped me tremendously when creating the JOIN function.

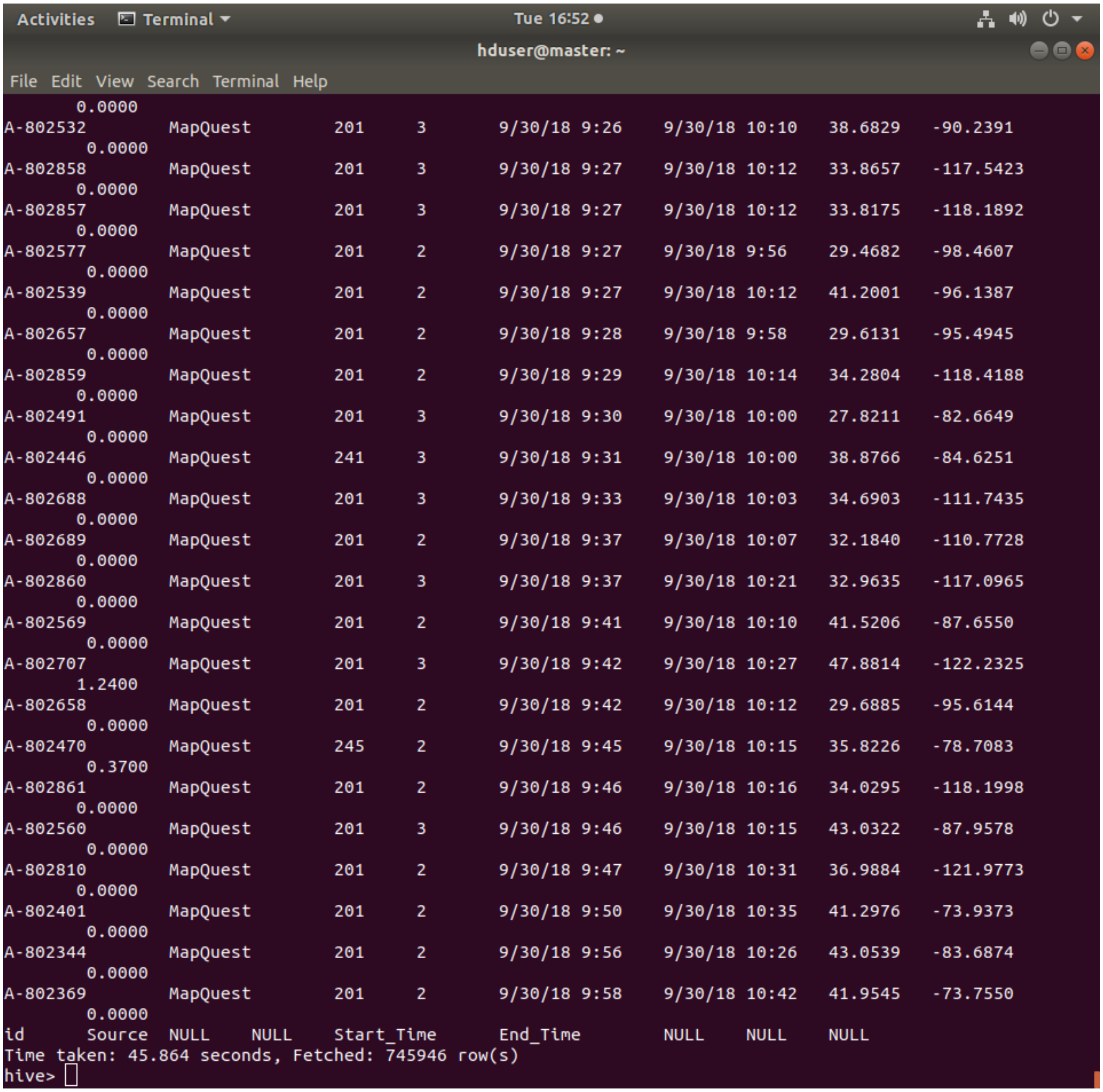
WHAT DID NOT GO WELL?

The first data set that I downloaded from Kaggle apparently has poor data quality specifically no ID. This means if I would like to perform JOIN function, it is nearly impossible. The only way to solve it is to create a random table. I tried my best to solve it, but apparently no luck. I decided to change my dataset because I was afraid that I was running out of time.

Even though the second dataset worked well, I failed to process the timestamp. My dataset timestamp format has a mix between DD-MM-YYYY hh:mm and MM/DD/YYYY hh:mm. According to Hive Language Manual, there are two ways to process timestamp:

1. YYYY-MM-DD hh:mm:ss
2. Unix timestamp function

I tried to custom format on Microsoft Excel to YYYY-MM-DD hh:mm, but for some of the reasons some rows that have MM/DD/YYYY hh:mm format was not converted. I decided to create my own unix timestamp function with a help of different forums, but still failed. Apparently, when I tried to process the time in HiveQL, it still works; however, I highly doubt it shows all the data correctly.



```
       0.0000
A-802532        MapQuest        201     3       9/30/18 9:26    9/30/18 10:10   38.6829   -90.2391
        0.0000
A-802858        MapQuest        201     3       9/30/18 9:27    9/30/18 10:12   33.8657   -117.5423
       0.0000
A-802857        MapQuest        201     3       9/30/18 9:27    9/30/18 10:12   33.8175   -118.1892
       0.0000
A-802577        MapQuest        201     2       9/30/18 9:27    9/30/18 9:56    29.4682   -98.4607
        0.0000
A-802539        MapQuest        201     2       9/30/18 9:27    9/30/18 10:12   41.2001   -96.1387
        0.0000
A-802657        MapQuest        201     2       9/30/18 9:28    9/30/18 9:58    29.6131   -95.4945
        0.0000
A-802859        MapQuest        201     2       9/30/18 9:29    9/30/18 10:14   34.2804   -118.4188
       0.0000
A-802491        MapQuest        201     3       9/30/18 9:30    9/30/18 10:00   27.8211   -82.6649
        0.0000
A-802446        MapQuest        241     3       9/30/18 9:31    9/30/18 10:00   38.8766   -84.6251
        0.0000
A-802688        MapQuest        201     3       9/30/18 9:33    9/30/18 10:03   34.6903   -111.7435
       0.0000
A-802689        MapQuest        201     2       9/30/18 9:37    9/30/18 10:07   32.1840   -110.7728
       0.0000
A-802860        MapQuest        201     3       9/30/18 9:37    9/30/18 10:21   32.9635   -117.0965
       0.0000
A-802569        MapQuest        201     2       9/30/18 9:41    9/30/18 10:10   41.5206   -87.6550
        0.0000
A-802707        MapQuest        201     3       9/30/18 9:42    9/30/18 10:27   47.8814   -122.2325
       1.2400
A-802658        MapQuest        201     2       9/30/18 9:42    9/30/18 10:12   29.6885   -95.6144
        0.0000
A-802470        MapQuest        245     2       9/30/18 9:45    9/30/18 10:15   35.8226   -78.7083
        0.3700
A-802861        MapQuest        201     2       9/30/18 9:46    9/30/18 10:16   34.0295   -118.1998
       0.0000
A-802560        MapQuest        201     3       9/30/18 9:46    9/30/18 10:15   43.0322   -87.9578
        0.0000
A-802810        MapQuest        201     2       9/30/18 9:47    9/30/18 10:31   36.9884   -121.9773
       0.0000
A-802401        MapQuest        201     2       9/30/18 9:50    9/30/18 10:35   41.2976   -73.9373
        0.0000
A-802344        MapQuest        201     2       9/30/18 9:56    9/30/18 10:26   43.0539   -83.6874
       0.0000
A-802369        MapQuest        201     2       9/30/18 9:58    9/30/18 10:42   41.9545   -73.7550
        0.0000
id      Source  NULL    NULL    Start_Time      End_Time        NULL    NULL    NULL
Time taken: 45.864 seconds, Fetched: 745946 row(s)
hive> 
```

WHAT WOULD I TRY DIFFERENTLY NEXT TIME?

There are three main takeaways from this project:

First, I would make a deeper research between Hive and Spark. After having a conversation with a senior data scientist, he mentioned that nowadays most companies are using Spark rather Hive and have better compatibility with machine learning programs such as Python. In fact, Spark has better documentation that Hive.

Second, I would pick three different datasets that I would be interested to work on. The main purpose is to have some back up in case if I choose poor dataset quality (e.g. No ID). It sounds like that I am irresilient the fact that I hate to jump over if I could not solve a problem.

Third, when creating the unix timestamp, I should have documented every code that I tried. This would give me a comparison between codes that I have written and not repeating the same codes that I might have written.