Systems for Big Data Processing - Project 2
Taxis Dataset Analysis

1st Rafaela Cruz
nº 56926
rc.cruz@campus.fct.unl.pt

2nd André Bastos
nº 56969
afn.bastos@campus.fct.unl.pt

I. Introduction

In this project, we extracted some information about taxi rides in some city, using Apache Hive. In this report, we describe our approaches for extracting the required information, as well as the results obtained.

II. Data Pre-processing

For a more efficient querying, we decided to create a separate file to pre-process the dataset. The first thing that the file does is to create a table for the data rows, as they are originally from a csv file. Afterwards, we create another table that will contain the information for querying.

In this new table, we deleted the columns with information that was not necessary for the exercises, namely, the columns' number, vendor_id, pickup_datetime, dropoff_datetime, pickup_longitude, pickup_latitude, dropoff_longitude, dropoff_latitude, maximum_temperature, minimum_temperature, average_temperature and snow_depth.

Then, we added new columns to the table, with the information needed for the exercises. First, we had to extract the weekday from de pickup_datetime column, using the function `from_unixtime(unix_timestamp(pickup_datetime),'u')`. [1] Note that, if we obtain the weekdays in this way, Monday will correspond to number 1 and Sunday to number 7. Then, to obtain the pickup-area, we concatenated the pickup_longitude with the pickup_latitude, rounding them to two decimal cases (subtracting 0.005 so we always round down the values). [2] We did the same to obtain the dropoff-area and the route. Next, we used the function `substring` to obtain the hour from the pickup_datetime column. To obtain time intervals of 15 minutes, we used the pickup_datetime column and set a condition to obtain the intervals: if the minutes are between 0 and 14, then we replace them for 0, if the minutes are between 15 and 29, then we replace them for 15, and so on. To obtain the duration of a given ride, we subtracted the dropoff_datetime by the pickup_datetime. [3] For the optional exercise, we also needed to determine if a ride happened during day time or night time, so we also added a column with a 1 if the pickup_datetime hour was between 7 a.m. and 7 p.m. (day time) and 0 otherwise (night time).

The next step was to export the table to a directory called pre_processed. [4] This results in a folder with five files. Afterwards, we needed to change the extension of these files to csv and merge them into an unique csv file called pre_processed.csv, using an unix command (`cat`).

This is a good way to optimise space and processing time, as unused information was excluded from the beginning. It was also a way of simplifying the queries, since all the required information was previously generated.

III. Most Frequent Routes

In the first exercise we wanted to obtain the 10 most frequent routes given a weekday and hour. To do that, we needed to count the number of repeated routes that appeared for the same weekday, hour and route combinations grouped together. Therefore, we created a temporary table with the information needed to solve the problem. This approach turned the whole query a lot simpler, as the counting is already done.

Afterwards, we needed a query that created a ranking based on the count generated earlier. The `partition` feature from hive was used over the weekday and hour to isolate every combination of these two from the rest of the data for this query. [5] After partitioning, the function `row_number` was used to give an ascending number to each one of the rows. [6] Finally to finish the ranking, the ordering by the count column (`_c3`) was crucial.

As what was needed were the most frequent routes for the weekday and hour combination, selecting these two and the list of routes from the earlier query with another query was the next step. Groups were formed with weekday and hour, and limited the appearance of rows with row number less or equal to 10. This way, only the top 10 routes are shown for every weekday and hour.

In the end, we exported the results to a csv file named results_ex1.csv. This file contains 168 rows, one for each combination of weekday and hour. The taxi drivers can thus use this information the choose where they should work in a given weekday and hour, so they can have more profit.

IV. Expected Duration and Distance of a Taxi Ride

In this exercise, we wanted to obtain the expected duration and distance of a taxi ride, given the pickup-area, the weekday and time intervals of 15 minutes. The pickup-area, the weekday, the time intervals and the duration were obtained in the data pre-processing step (Section II). Therefore, we only needed to select these four columns, as well as the haversine_distance column, and compute the average value for

the duration and the distance, using the `avg` function. [7] In the end, we grouped the average values by the pickup-area, the weekday and the time intervals using a `group by` and exported the results to a csv file.

In this way, we obtained a csv file named results_ex2.csv, with multiple rows, one for each combination of pickup-area, weekday and 15 minutes interval. The taxi drivers can use this information to determine when and where the rides are longer, so they can have more profit.

V. Factor of Clients - Wet Weather *versus* Dry Weather

In this exercise, the objective was to determine the factor of additional clients when it is raining or snowing *versus* dry weather, given the pickup-area. To do this, we started by creating a temporary table with the pickup-area and two new columns, wet_weather and dry_weather, given two conditions: if a given row has a precipitation or a snow-fall value higher than zero, we add the number of passengers (passenger_count) to the wet_weather column, else we add a zero to that column. If a given row has a precipitation or a snow-fall value equal to zero, we add the number of passengers (passenger_count) to the dry_weather column, else we add a zero to that column.

Afterwards, we use this temporary table to generate the results. To do that, we select the pickup-area and divide the sum of the number of passengers on the wet weather column by the sum of the number of passengers in the dry weather column, by pickup-area. Note that, if the sum of the number of passengers for the wet weather column is zero, then the factor wet weather *versus* dry weather will be zero, which is not a problem. However, if the sum of the number of passengers in the dry weather column is zero, then the factor will be a number divided by zero, which is not possible. To solve this problem, we set a condition: if the sum of the number of passengers in the dry weather column is zero, then the result will simply be one.

In the end, we exported the results to a csv file named results_ex3.csv. Since there is a result for every pickup-area, this file has 720 rows. The taxi drivers can use this information to check whether it is a good idea to go to work when it is raining or snowing, given a certain area. Analysing the results, we observe that, contrary to what would be expected, there are not many areas where the factor wet weather *versus* dry weather is higher than one. Therefore, we can conclude that there is not a great advantage in working when it is raining or snowing.

VI. Optional Exercise - Best period of time to work as a taxi driver

For the optional exercise, we decided that one useful query for the taxi drivers would be at which periods of time (day of the week, and day or night time) it is more profitable to work. We considered the number of taxi rides as the indicator of whether it is more or less profitable.

In the data pre-processing step (Section II), we computed whether it is day time (1) or night time (0) into a column called isDayTime.

Afterwards, we grouped together the weekday and the isDayTime columns, and computed the number of taxi rides for each period of time, by applying the `count` function to the id column.

In the end, we exported the results to a csv file named results_ex4.csv. This file only contains 14 rows, one for each combination of weekday, day and night time. The taxi drivers will find this information very useful as they can get the number of drives occurring given a weekday and day or night. This is more useful than using the number of passengers in the ride, as the price will be the same independently of this number. What is more profitable is the number of rides that a taxi driver can get, as there can be a lot of competition between the taxi drivers. Observing the results, we can see that it is always more profitable to work during the day then during the night. We can also conclude that the day where it is more profitable to work at is at Thursdays and the night where it is more profitable to work at is at Saturdays.

VII. Conclusion

With this project, we can conclude that Apache Hive is convenient to extract useful information from a given dataset. Using this technology, we extracted different kinds of information that would be useful for taxi drivers working in the city from which this data was collected. For example, taxi drivers can use our results to determine which routes are more frequent in a given weekday and hour, which pickup-areas they should pickup clients from so that the duration of the ride is longer, which pickup-areas have more clients when it is raining or snowing and which period of time has more clients.

References

[1] Stack Overflow: Hive date function to achieve day of week. Retrieved from https://stackoverflow.com/questions/22982904/hive-date-function-to-achieve-day-of-week. Last accessed in Nov 22, 2019.

[2] Stack Overflow: Apache Hive How to round off to 2 decimal places? Retrieved from https://stackoverflow.com/questions/34672818/apache-hive-how-to-round-off-to-2-decimal-places. Last accessed in Nov 22, 2019.

[3] Stack Overflow: Hive's unix_timestamp and from_unixtime functions Retrieved from https://stackoverflow.com/questions/31701847/hives-unix-timestamp-and-from-unixtime-functions. Last accessed in Nov 23, 2019.

[4] Stack Overflow: How to export a Hive table into a CSV file? Retrieved from https://stackoverflow.com/questions/17086642/how-to-export-a-hive-table-into-a-csv-file. Last accessed in Dec 7, 2019.

[5] DataFlair: Hive Partitions, Types of Hive Partitioning with Examples. Retrieved from https://data-flair.training/blogs/apache-hive-partitions/. Last accessed in Dec 3, 2019.

[6] Java, SQL and jOOQ: The Difference Between ROW_NUMBER(), RANK(), and DENSE_RANK(). Retrieved from https://blog.jooq.org/2014/08/12/the-difference-between-row_number-rank-and-dense_rank/. Last accessed in Dec 3, 2019.

[7] Arm Treasure Data: Hive Built-in Aggregate Functions. Retrieved from https://support.treasuredata.com/hc/en-us/articles/360001457367-Hive-Built-in-Aggregate-Functions. Last accessed in Dec 1, 2019.