MapReduce Programming Assignment

**Introduction**

You were introduced to the MapReduce Optional Assignment, where you practised writing MapReduce codes using a relatively small data set. In this assignment, you will use the NYC TLC yellow taxi data set for the year 2017 and perform various operations using the big data tools that you have learnt about so far.

This session contains the following segments -

* Introduction
* Dataset Description
* Tasks
* Evaluation Rubric
* Final Submission

In addition, optional sessions are also included, in which you will be reintroduced to some of the concepts covered in the course. You will learn all about MRJob, a popular library created by Yelp for simplifying the process of writing MapReduce code. You will use Apache Sqoop and Apache HBase. You will learn how to work with AWS RDS (Relational Database Service).

The data set for the assignment can be downloaded from these links:

https://nyc-tlc-upgrad.s3.amazonaws.com/yellow\_tripdata\_2017-01.csv

https://nyc-tlc-upgrad.s3.amazonaws.com/yellow\_tripdata\_2017-02.csv

https://nyc-tlc-upgrad.s3.amazonaws.com/yellow\_tripdata\_2017-03.csv

https://nyc-tlc-upgrad.s3.amazonaws.com/yellow\_tripdata\_2017-04.csv

https://nyc-tlc-upgrad.s3.amazonaws.com/yellow\_tripdata\_2017-05.csv

https://nyc-tlc-upgrad.s3.amazonaws.com/yellow\_tripdata\_2017-06.csv

**NOTE:**The size of each individual csv file is huge. Since processing all the files is cumbersome, we recommend using just one csv file for testing your MapReduce queries and debugging your code and the final code against all the datasets.

**Dataset Description**

The data set that will be used for this assignment is the TLC trip record data for yellow taxis.

**Please note that the input data is for the year 2017.**

|  |  |
| --- | --- |
| **Field Name** | **Description** |
| VendorID | A code indicating the TPEP provider that provided the record. 1= Creative Mobile Technologies, LLC; 2= VeriFone Inc. |
| tpep\_pickup\_datetime | The date and time when the meter was engaged |
| tpep\_dropoff\_datetime | The date and time when the meter was disengaged |
| passenger\_count | The number of passengers in the vehicle. This is a driver-entered value |
| trip\_distance | The elapsed trip distance in miles reported by the taximeter |
| RatecodeID | The final rate code in effect at the end of the trip.  1= Standard rate  2=JFK  3=Newark  4=Nassau or Westchester  5=Negotiated fare 6=Group ride |
| store\_and\_fwd\_flag | This flag indicates whether the trip record was held in vehicle memory before sending to the vendor, aka “store and forward,” because the vehicle did not have a connection to the server.  Y= store and forward trip  N= not a store and forward trip |
| PULocationID | The ID of the location from where the passenger was picked. |
| DOLocationID | The ID of the location from where the passenger was dropped. |
| payment\_type | A numeric code signifying how the passenger paid for the trip.  1= Credit card  2= Cash  3= No charge  4= Dispute  5= Unknown  6= Voided trip |
| fare\_amount | The time-and-distance fare calculated by the meter. |
| extra | Miscellaneous extras and surcharges. Currently, this only includes the $0.50 and $1 rush hour and overnight charges. |
| mta\_tax | $0.50 MTA tax that is automatically triggered based on the metered rate in use. |
| tip\_amount | This field is automatically populated for credit card tips. Cash tips are not included. |
| tolls\_amount | Total amount of all tolls paid in trip . |
| improvement\_surcharge | $0.30 improvement surcharge assessed trips at the flag drop. The improvement surcharge began being levied in 2015. |
| total\_amount | The total amount charged to passengers. Does not include cash tips. |
| Airport\_fee | $1.25 for pick up only at LaGuardia and John F. Kennedy Airports |

 The fields present in the data and their meaning can also be referred from [here](https://www1.nyc.gov/assets/tlc/downloads/pdf/data_dictionary_trip_records_yellow.pdf).

**Tasks**

You will use the following big data tools for working with the assignment - Hadoop Framework, Apache HBase and Apache Sqoop. You'll be required to use an AWS EMR instance with all the services and install additional services depending on the tasks.

You will need to complete the following tasks after downloading the files onto your EMR Cluster. We recommend that you use the m4.xlarge cluster with ample space, since you will be working with a huge data set.

Once the dataset has been downloaded onto your instance, perform the following tasks:

Tasks:

**Data Ingestion Tasks:**

**Task 1.** Create an RDS instance in your AWS account and upload the data using MySQL.

Since the dataset is huge, you need to upload the information from only two files from the dataset.

**Note: You will need to create an appropriate schema for the data sets to upload them to RDS (you can find the data dictionary in the previous segments The steps to work with RDS in given in the Additional Resource).**

**Task 2.**Use Sqoop command to ingest the data from RDS into the HBase Table.

**Task 3.**Bulk import data from next two files in the dataset on your EMR cluster to your HBase Table using the relevant codes.

**Note: For the above task 3, you just need to import data from the next 2 csv files (**yellow\_tripdata\_2017-03.csv & yellow\_tripdata\_2017-04.csv**) on your EMR cluster.**

**MapReduce Tasks:**

**Task 4.**Write MapReduce codes to perform the tasks using the files you’ve downloaded on your EMR Instance:

1. Which vendors have the most trips, and what is the total revenue generated by that vendor?
2. Which pickup location generates the most revenue?
3. What are the different payment types used by customers and their count? The final results should be in a sorted format.
4. What is the average trip time for different pickup locations?
5. Calculate the average tips to revenue ratio of the drivers for different locations in sorted format.
6. How does revenue vary over time? Calculate the average trip revenue per month - analysing it by hour of the day (day vs night) and the day of the week (weekday vs weekend).

**Optional Task:**

**Task 5.**Use Sqoop export command to export the results of each MapReduce tasks above to your RDS instance. Use the RDS connection string connection to visualise the dataset using a dashboarding tool (Google Data Studio, Tableau or PowerBI) ***(Optional)***

**NOTE:**Please note that Task 5 is optional and purely to demonstrate how RDS and Sqoop.

**NOTE:**The Data Ingestion tasks and the MapReduce tasks are separate. The MapReduce tasks must be run with the local data downloaded to the cluster.

**Final Submission**

Submissions Required

Upload a ZIP file containing the following:

* **Document-01:**A PDF document (RDS.pdf) containing the codes, with the explanations, used for loading the datasets mentioned into an AWS RDS instance. This should have the code along with the screenshots of the EMR instance showing the table creation.
* **Document-02:**A PDF document (Ingestiontask.pdf) containing the code to create the HBase table. The file should also include the Sqoop command to ingest data from RDS into the HBase table. The document should be well commented explaining the code.
* The Python code (**batch\_ingest.py**) used to ingest the batch data to the HBase table.
* The Python codes used for the MapReduce tasks. The files should be labelled as **mrtask\_a.py**, **mrtask\_b.py,**and so on based on the relevant question number. This scripts should also contain the relevant and proper comments, explaining all the steps taken. Answers to the query and the screenshots of the results of the MapReduce tasks must be included in a separate document (**MapReducetasks.pdf**) in sequence.

All the above files must be combined into **one zip file**before uploading.

For submissions obtained within 1 week of the deadline, there will be a 30% penalty. Submissions beyond 1 week of the deadline will be provided only feedback and score zero marks (100% penalty).

Make sure you have not made any changes to the original dataset provided to you. Your  code should work on the dataset given to you as part of the problem statement. You are not allowed to make modifications in data set using any means.

During grading we will be running your code on the dataset provided by us, in case your code gives errors with that, then marks will be deducted accordingly.

All penalties are automatically applied by the system based on time of submission. Hence, submissions that are late, even by a second, will attract penalties.

For e.g.- If the deadline is 2nd August 2019, 11:59:00 PM IST, the submissions at 2nd August 2019, 11:59:01 PM IST will attract a penalty of 30%. Hence we recommend that assignments are submitted at least 30 minutes before the deadline to avoid any last minute issues.

Also, note the all the deadlines are in IST (UTC +5.5), hence, if you are in a different time zone, then your deadline may vary according to local time. For eg - If you are in London and following BST (British Summer Time) which is UTC +1 then deadline for you in local time would be 7:29:00 PM BST when the deadline in India is 11:59:00 PM IST.

Make sure you click “Submit for Grading” only if you are 100% sure, else you must just upload your file and leave it there. It will be automatically submitted before the deadline. If you click “Submit for grading” after uploading the file, under no circumstance you will be allowed to resubmit or change your upload.

Here are the steps that you must follow during submitting any assignment-

Collect all the files (if there are multiple files) and compress(zip file) them together.

Try to upload this compressed file latest by 11:30 PM

Download your submission and check that you have included all the required files.

Check that none of the files or the zip is corrupt. If it is found to be corrupt during grading, you will NOT be allowed to re-submit.

If you are 100% sure that you will not need to make any more changes in the assignment, click “Submit for Grading”, else, just let it be. Unless you remove it, it will be automatically submitted at the upcoming deadline.