Using MapReduce for Analysis of Price Variation, Volume and Value of S&P 500 Index

Ricardo Salomao da Silva Junior
Programming for Data Analytics
School of Computing
National College of Ireland
Dublin, Ireland
x18147607@student.ncirl.ie

***Abstract*—The analysis of stock market for business investments is an important practice for investors since they can make better business decisions based in the history of stock market behavior. S&P500 is an American Stock Market Index with 500 Stocks of big representativeness in the market. Processing the large amount of data collected from years of daily and hourly stock market analysis can be a computationally time demanding task. To overcome this problem, the Hadoop MapReduce framework for parallel computing is used in this study. Technologies for data storage (MySQL and HBase) and bulk load (Sqoop) are also used to assist the performed analysis. This work aims to perform an exploratory analysis of the S&P stock market index in respect to price variation, volume (number of transactions) and daily average price of stocks. At the conclusion of this report it is presented an automated way (shell scrip) to perform exploratory analysis of stock market. The graphic results of the processed data show important findings about price variation, volume of transactions and value of S&P500 stocks.**

Keywords—MySQL, Hadoop, Map Reduce, HBase, Sqoop, S&P500

I. INTRODUCTION

Stock market is an organization of buyers and sellers where shares and business can be negotiated in a network of economic transactions. Each Stock or Share opens the day with a market value and its price varies along the day accordingly to the business transactions.

The analysis of the Stock market has a great importance for businesses because the market history can be of a great predictor for good business decisions.

Standard & Poor's 500 (S&P500) [1] is an Index of the stock market composed by 500 assets that can be used by investors to investigate the market comparing the return of specific investments.

Some basic exploratory analysis of the historical data of S&P500 can lead investors to findings and insights about the business market. This analysis can be used to predict good business decisions investing to the right stocks and avoiding the ones likely to give no profits.

The proposed study aims to perform an exploratory analysis of the S&P500 historical data to answer the following questions:

1) *Which stocks are the most valuable by the years?*
2) *Which stocks have the highest price variation by the years?*
3) *Which stocks have the highest volume of transactions by the months, and by the years?*
4) *What is the average price of the famous S&P500 stocks Apple, Amazon, Facebook and Microsoft by the years?*

One of the big problems of dealing with sock market data is the large size of the dataset depending on how many stocks are in analysis. S&P500 alone has 500 stocks and to analyse 10 years of daily history of this index would represent 10(years)*365(days)*500(stocks), around 1.8M of records. If you include other Indexes or more years, the data increases in a fast rate and processing it in a centralized computing system is a time demanding.

Hadoop MapReduce [2] is a framework for parallel or distributed computing largely used for big data applications. This technology can deal with scalability computers providing an efficient and reliable parallel/distributed processing.

This document provides a step-by-step of how to use distributed computing based in Hadoop map reduce framework for exploratory analysis of stock market data.

In a first moment, a literature review of some related work is presented in the next session. Following that the methodology for the development of this work is detailed. The results of the analysis and conclusion are presented in the IV and V section, respectively.

II. LITERATURE REVIEW

The first version of Hadoop started to be developed by Apache Software Foundation following the ideas from [3] published in 2003. In this work, the Google File System was described by the authors for applications where the needs of large-scale data processing in commodity of hardware are essential. The proposed system provides fast recovering and data replication by actively controlling and monitoring applications.

The framework MapReduce for data processing on large clusters was proposed by [4], the main inspiration for Hadoop MapReduce framework. Hiding details of implementation of parallel computing, fault tolerance, balance of workload and optimization this framework provides two functions: Map-function, used to generate intermediate subsets for posteriorly processing by the Reduce-function, merging all the intermediate values.

The survey in [5] discusses the method of parallel data processing using map reduce framework. The authors provide a characterization of the map reduce aspects. The advantages of using this framework are highlighted e.g. high scalability, fault tolerance and flexibility. The authors also

list some pitfalls of this framework e.g. no high-level language like SQL in data base management systems.

In 2009 was published an overview of column-oriented database systems detailed in [6]. Column-oriented databases can deal with large scale data-intensive applications by storing each column from a table separately compressing their attributes. This way, it becomes faster to select a subset from a column avoiding excessive disk head seeking in comparison to traditional relational database where entire records (rows) are stored.

III. METHODOLOGY

The aim of this section is to present the steps followed in this study to create the final shell script in Linux for data acquisition, storage, processing and presentation of S&P Index.

A. Data Acquisition

The historical data from market capitalisation of the 500 companies from S&P index used in this work was obtained from kaggle.com and can be accessed through the link: https://www.kaggle.com/camnugent/sandp500.

The market stocks price dataset is composed by 7 columns in a coma-separated values file. The data description of the columns in this file is presented in the following structure:

- Date (YYYY-MM-DD): Representing the year (YYYY), month (MM) and day (DD) of the current record.
- Open: This is a decimal value corresponding to the price (USD) of the stock at the time that the market was open.
- High: This is a decimal value with the maximum price (USD) achieved by the current stock during the day of this record.
- Low: This is a decimal value with the lowest price (USD) of the current stock during the day of this record.
- Close: This is a decimal value corresponding to the price (USD) of the stock at the time that the market was closing.
- Volume: This is a numeric value representing the total number of transactions of the stock at the day of the record.
- Name: This is a text value with the name of the current S&P stock present in this record.

The dataset is composed by daily market prices data from 500 stocks between the years of 2013 and 2018. The total number of rows or records is above 619K consuming 28MB of the disk.

To access this public dataset directly from Kaggle.com it is required the user to log in to the platform. To avoid this and other undesired steps performing the tasks for this analysis in an automatic way, the data was stored in a cloud computing storage platform (https://www.dropbox.com/s/kl5hvy3e4n7y7w5/all_stocks_5yr.csv) where it was possible to download simply using the Linux command ***wget*** in a shell script.

B. Storing Historical Stock Price Data

The selected data base management system for storing the S&P500 stock pricing data was MySQL [7]. The rationale of choosing this database instead of using the csv file is that besides the traditional ACID (Atomicity, Consistency, Isolation, Durability) properties, it is possible to share the data with other local, remote or distributed applications, providing more security, drill down and faster reporting capabilities.

Another reason for selecting this database is that there are currently available tools for transferring data from the MySQL to the Hadoop database, HDFS. This way, using MySQL commands and bulk load tools, selected columns, subsets of the data or even the entire stored data can be load into the HDFS to later perform a Hadoop MapReduce task.

A database called SandP500 with the same data structure from the csv file was created in the DBMS.

TABLE I. STOCK TABLE MYSQL DATA DESCRIPTION

Stock Table Data Description		
Name	***Type***	***Description***
ID	INTEGER	A unique and auto incremental field corresponding to the record identifier.
DATE	DATE	Date of the record
OPEN	DECIMAL(8, 2)	Price of the Stock when the market opened
HIGH	DECIMAL(8, 2)	Higher price of the Stock in the current date
LOW	DECIMAL(8, 2)	Lower price of the Stock in the current date
CLOSE	DECIMAL(8, 2)	Price of the Stock when the market closed
VOLUME	INTEGER	Number of transactions of the stock in the current date
NAME	VARCHAR(255)	Name of the Stock from S&P500

The Stock table was loaded by the command ***load data infile*** present in MySQL. This operator receives the path to the csv file (origin), the table Stock and the chosen columns (destination).

C. Transferring Data from MySQL to HDFS

To use Hadoop for a MapReduce task it is required that the data is present in the Hadoop Distributed file System (HDFS). The tool used in this project for transferring data between MySQL and HDFS in an efficiently manner was Apache Sqoop [8], an implementation for load bulk between relational data base and HDFS.

The inputs used for command ***sqoop import***, are the database connection, the table Stock (origin) and the HDFS target directory (destination). A MapReduce task is started and the entire data from table Stock in MySQL is efficiently transferred to the selected HDFS directory.

D. MapReduce Structure

In this section the proposed MapReduce tasks to help to answer the proposed questions in the introduction for exploratory analysis of S&P Index are detailed.

1) Stock Avg. Price by Years: Calculates for each one of the 500 Stoks S&P the average price in each year.

a) Map: Filter the records based in the stock name and year. Key: "NAME-YYYY", Value: "PRICE" (MEAN(OPEN, CLOSE)).

Reduce: *Calculate the mean of the prices from the current stock and year.*

2) Stock Avg. Price Variation by Years: Calculates for each one of the 500 Stoks S&P the average price variation in each year.

a) Map: Filter the records based in the stock name and year. Key: "NAME-YYYY", Value: "PRICE_VARIATION" (HIGH - LOW).

b) Reduce: Calculate the mean of the price variation from the current stock and year.

3) Stock Avg. Volume Transactions by Month: Calculates for each one of the 500 Stoks S&P the average number of transactions in each month.

a) Map: Filter the records based in the stock name, the year and month. Key: "NAME-YYYY-MM", Value: "VOLUME" (VOLUME OF TRANSACTIONS).

b) Reduce: Calculate the mean of the number of transactions from the current stock, year and month.

4) Stock Avg. Volume of Transactions by Year: This taks is basically the same as the previous one "2) Stock Avg. Volume of Transactions by Month". The difference is in the Key from Map function where the records are not filtered by month ("MM") only by year using the Key: "NAME-YYYY".

E. MapReduce Implementation

For a more complete exploratory analysis in total 14 map reduce tasks were implemented. But for the purpose of this report only 4 will be used. The other tasks are present in the code and jar files available with this report.

The MapReduce tasks were implemented using java and eclipse with the classes SandP500Driver.Java, SandP500Mapper.Java, SandP500Reducer.Java. The Jar files from each task were created and placed into the local path of the project.

The script of the project was updated with the Hadoop command lines to execute automatically each MapReduce task in order of dependency (as some tasks depends on the others).

F. Storing MapReduce Output

The selected tecnology for storing the MapReduce tasks output results was the column oriented distributed database Hbase [9].

Each one of the 8 tasks generated a separated output folder in HDFS with repective MapReduce output result. For future analysis of this output, the data was trasnfered to the Hbase database using the command ***ImportTsv*** from Hbase. It was also included to the script of the project the create database commands before transfering data.

G. The Final Script

Each step covered until now, was added to the project script called ***runSandP500.sh*** to perform actions automatically with less user interventions possible, avoinding human mistakes and speeding up the time of execution of the entire process.

The inputs of the script:

HADOOP_PATH= path where Hadoop is installed
LOCAL_PATH= path where the project
HDFS_PATH= path in the HDFS
HBASE_PATH= path where HBase is installed
MYSQL_PSSWRD= MySQL Password from user root.

IV. Results

In this section the output results from the MapReduce tasks after executing the shell script created during the methodology are investigated using graphics elements and charts for visual analysis and comparison.

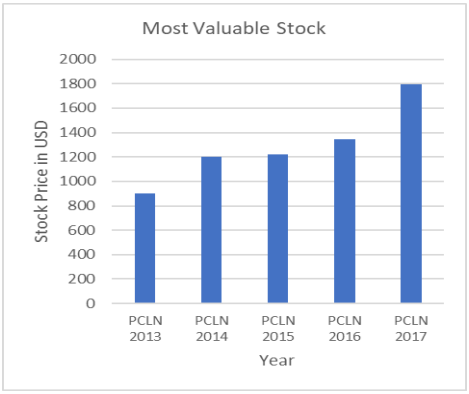

Fig. 1 Most valuable stocks

The first chart observed represents the stocks with highest price average during the years. The output result of the 1st Map Reduce task stored in the table "stockpricesbyyear" from HBase contains the stocks and average price for year. The chart in Fig. 1 was created by selecting the stocks with maximum average price by year. The Priceline Group (PCLN) was the most valuable stock during the five years of data. PCLN presents a rising rate of stock price, and that's is even more evident wen comparing the average stock price in 2013 and 2017, when the stock price doubled.

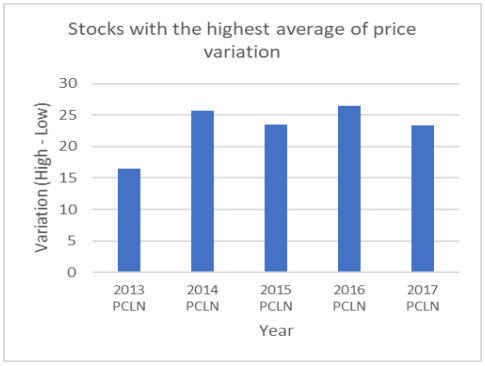

Fig. 2 Highest Price Variation

The output result of the 2nd Map Reduce task stored in the table "pricevariationbyyear" from HBase contains the stocks and average of price variation by year. The chart in Fig. 2 was created by selecting the stocks with maximum average price variation by year. Here is observed that the most valuable stock is also the one with highest average of price variation. In other words, there is a big difference between the lowest and highest daily prices that the stock PCLN achieves during the years.

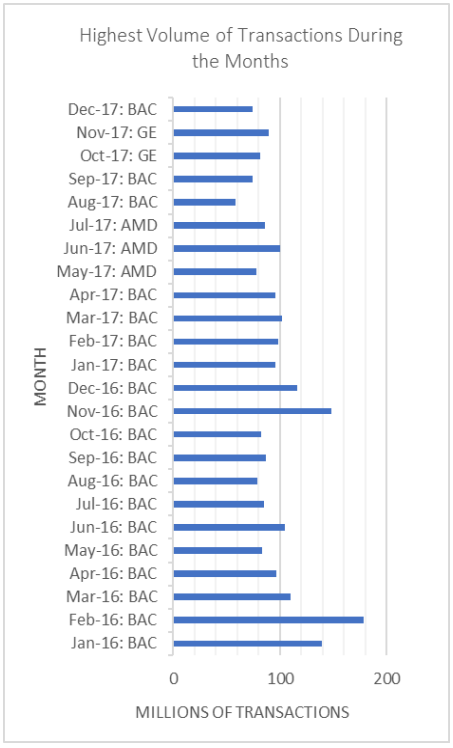

Fig. 3 Highest Volume of Transaction by Month

The third chart shows each month of 2016 and 2017 with the respective stock with highest volume of transactions in the month. The output result of the 3rd Map Reduce task stored in the table "avgvolumebymonth" from HBase contains the stocks and average of number of transactions by month. The chart in Fig. 3 was created by selecting the stocks with maximum average of transactions by month during the years of 2016 and 2017. Bank of America (BAC) is the most present stock in this chart. It means that this share is one of the most popular between the traders achieving the number of 180 million of transactions in February of 2016.

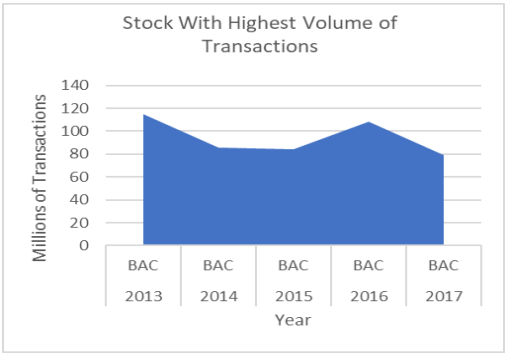

Fig. 4 Highest Volume of Transactions

The output result of the 4th Map Reduce task stored in the table "avgvolumebyyear" from HBase contains the stocks and average of volume by year. The chart in Fig. 4 was created by selecting the stocks with maximum average volume by year. The fourth chart shows that BAC was the most traded stock between the years of 2013 and 2017 with number of transactions as high as 100 million in 2013 and 2016.

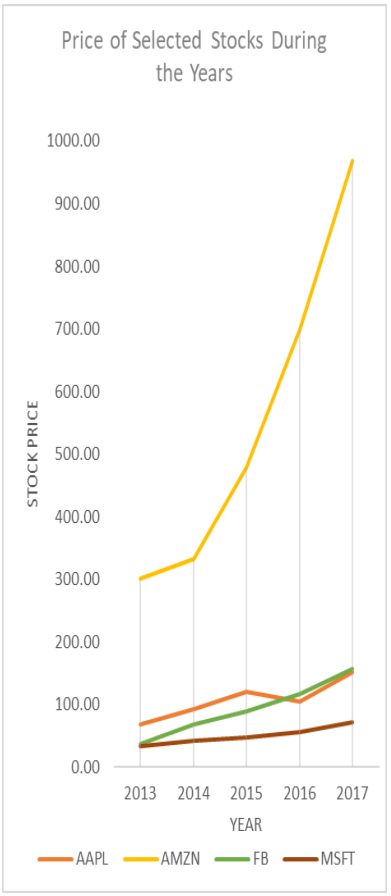

Fig. 5 Price of Famous Stocks

To create the chart in Fig. 5, the stocks AAPL, AMZN, FB and MSFT were selected from the output result from the Map Reduce task 1 stored in the table "stockpricesbyyear" from HBase. The chart number five shows the price of the selected stocks S&P500 Apple (AAPL), Amazon (AMZN), Facebook (FB) and Microsoft (MSFT) during the 5 years in

analysis. Amazon is one of the most expensive stocks demonstrating an exponential increasing rate.

V. Conclusions and Future Work

This work provides a shell script tool for distributed processing using Hadoop MapReduce framework for automated exploratory analysis of stock market index.

The database management system MySQL was selected for storing the stock market data before the map reduce tasks. And, to transfer the data between MySQL and HDFS the tool Sqoop for bulk load was used.

In total 14 jars files were created to perform different MapReduce tasks. The output files after executing these jars with Hadoop were transferred from HDFS to the column-oriented NoSQL database HBase for posterior analysis.

Results shown that the Priceline Group (PCLN) is one the most valuable and the stock with highest price variation in the selected years. The Bank of America (BAC) is the stock with highest number of transactions for 5 years consecutively.

Finally, the last chart shows stock price analysis of four popular companies Apple, Amazon, Facebook and Microsoft from the S&P500.

For future work it is suggested a time series analysis using distributed computing for stock market price forecast collecting a larger data set, for a better generalization.

References

[1] SP Indices, "S&P U.S. Indices Methodology", 2019. [Online]. Available: https://us.spindices.com/documents/methodologies/methodology-sp-us-indices.pdf. [Accessed: 28- Apr- 2019].

[2] Apache Software Foundation, "Apache Hadoop", Apache Hadoop. [Online]. Available: https://hadoop.apache.org/. [Accessed: 28- Apr- 2019].

[3] S. Ghemawat, H. Gobioff and S. Leung, "The Google File System", Google AI, 2003. [Online]. Available: https://ai.google/research/pubs/pub51. [Accessed: 28- Apr- 2019].

[4] J. Dean and S. Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters", Google AI, 2004. [Online]. Available: https://ai.google/research/pubs/pub62. [Accessed: 28- Apr- 2019].

[5] K. Lee, Y. Lee, H. Choi, Y. Chung and B. Moon, "Parallel Data Processing with MapReduce: A Survey", cite seer, 2012. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.227.3146. [Accessed: 27- Apr- 2019].

[6] D. Abadi, P. Boncz and S. Harizopoulos, "Column-oriented Database Systems", ACM DL, 2009. [Online]. Available: https://doi.org/10.14778/1687553.1687625. [Accessed: 28- Apr- 2019].

[7] Oracle Corporation, "MySQL", MySQL. [Online]. Available: https://www.mysql.com/. [Accessed: 28- Apr- 2019].

[8] Apache Software Foundation, "Apache Sqoop", Apache Sqoop. [Online]. Available: https://sqoop.apache.org/. [Accessed: 28- Apr- 2019].

[9] Apache Software Foundation, "Apache HBase", Apache HBase. [Online]. Available: https://hbase.apache.org/. [Accessed: 28- Apr- 2019].