

BIG DATA ANALYTICS USING HADOOP ECOSYSTEM

BY

MD. NAKIBUL HASSAN
ID: 132-15-2711

MD. AMINUL ISLAM
ID: 132-15-2704

AND

FARID OR RASHID
ID: 122-15-1981

This Report Presented in Partial Fulfillment of the Requirements for the
Degree of Bachelor of Science in Computer Science and Engineering

Supervised By

Md. Sadekur Rahman
Senior Lecturer
Department of CSE
Daffodil International University



DAFFODIL INTERNATIONAL UNIVERSITY

DHAKA, BANGLADESH

MAY 2017

APPROVAL

This project titled “**Big Data Analytics Using Hadoop Ecosystem**”, submitted by Md. Nakibul Hassan, Md. Aminul Islam, Farid Or Rashid to the Department of Computer Science and Engineering, Daffodil International University, has been accepted as satisfactory for the partial fulfilment of the requirements of the degree of B.Sc. in Computer Science and Engineering and approved as to its style and contents. The presentation has been held on 15th of May.

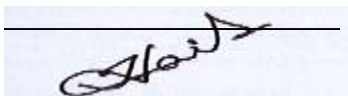
BOARD OF EXAMINERS



Dr. Sayed Akhter Hossain
Professor and Head

Department of Computer Science & Engineering
Faculty of Science & Information Technology
Daffodil International University

Chairman

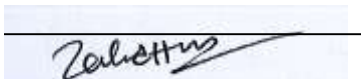


Dr. Sheak Rashed Haider Noori

Associate Professor and Associate Head

Department of Computer Science & Engineering
Faculty of Science & Information Technology
Daffodil International University

Internal Examiner



Md. Zahid Hasan

Associate Professor

Department of Computer Science & Engineering
Faculty of Science & Information Technology
Daffodil International University

Internal Examiner



Dr. Mohammad Shorif Uddin

Professor and Chairman

Department of Computer Science & Engineering
Jahangirnagar University

External Examiner

DECLARATION

We hereby declare that this project has been done by us under the supervision of **Md. Sadekur Rahman, Senior Lecturer**, Department of Computer Science & Engineering, Daffodil International University. We also declare that neither this project nor any part of this project has been submitted elsewhere for award of any degree or diploma.

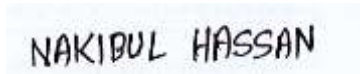
Supervised by:



Md. Sadekur Rahman
Senior Lecturer

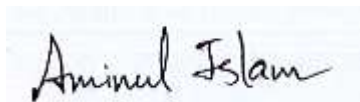
Department of Computer Science & Engineering
Faculty of Science & Information Technology
Daffodil International University

Submitted by:



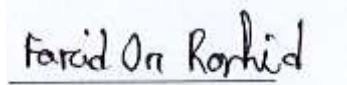
MD. Nakibul Hassan
ID: 132-15-2711

Department of Computer Science & Engineering
Faculty of Science & Information Technology
Daffodil International University



MD. Aminul Islam
ID: 132-15-2704

Department of Computer Science & Engineering
Faculty of Science & Information Technology
Daffodil International University



Farid Or Rashid
ID: 122-15-1981

Department of Computer Science & Engineering
Faculty of Science & Information Technology
Daffodil International University

ACKNOWLEDGEMENT

In the name of almighty **Allah**, the most Gracious and the most Merciful, All praises to Almighty Allah for giving us the strength, ability to being patient during our project and His blessing in completing our project.

There are few peoples whom we would like to thanks for completing and helping us to complete this project.

Special thanks to our honorable supervisor **Md. Sadekur Rahman, SeniorLecturer**, Department of Computer Science and Engineering, Daffodil International University for his continuous support and help. Whenever we were in trouble, we went to his desk and his desk was always open for to get helped. His endless patients, scholarly guidance, gentle behavior, continuous encouragement have made it possible to complete this project. We would like to thanks all the faculty members of CSE who helped us during this project.

We would like to express our heartiest gratitude to **Dr. Syed Akhter Hossain, Professor and Head**, Department of Computer Science and Engineering, for his kind help to finish our project and also to other faculty member and the staff of Computer Science and Engineering department of Daffodil International University.

We would also like to thanks all the friends of Daffodil International University for their support and help. We are also thankful to our senior brother of Daffodil International University for giving us advice to complete our project.

Last but not least, our deepest thanks to our parents to help us and for their kindness and giving us moral and financial support during project.

ABSTRACT

In 21st century, due to technological advancement, the amount of data getting generated is huge. The data volumes are exploding, around 90% of total data has been created past four to five years. Data is growing so faster that by the year 2020, about 1.7 megabytes information will be created each second for each human being on the planet. Enterprises worldwide will need to perform analytics on this huge data sets to make decision and stay competitive.

Previously traditional RDBMS (Relational Database Management System) was used to store data and analytics over data. But in today's world, RDBMS will not be efficient and time consuming to perform analytics over this huge data. Hadoop came into existence recently to rescue us and overcomes RDBMS limitations by providing simplified tools for huge data storage, faster processing on these huge data or we can say Big Data. Hadoop framework has two core components, one is HDFS (Hadoop Distributed File System) and another is Hadoop MapReduce a programming model. There are numbers of Big Data tools build on top of Hadoop which together form 'Hadoop Ecosystem'. One of the big Data tool is Apache Pig. Pig is a platform where large data set is analyzed using a data flow language, PigLatin.

The purpose of our project is to study different Hadoop ecosystems in details and perform data analytics on a movie data set using Hadoop. A movie data set consisting around fifty thousand records for analytics. Different analysis have been considered and analytics have been performed using MapReduce and Pig functionalities of Hadoop.

TABLE OF CONTENTS

CONTENT	Page No
Board of Examiners	i
Declaration	ii
Acknowledgement	iii
Abstract	iv
Table of Contents	v
List of Figures	vii
List of Tables	viii
 CHAPTER	
CHAPTER 1: INTRODUCTION	1-3
1.1 Introduction	1
1.2 Motivation	2
1.3 Aim of The Project	3
1.4 Expected Outcome	3
1.5 Report Layout	3
 CHAPTER 2: BACKGROUND	4-7
2.1 Definition of Big Data	4
2.2 Evolution of Big Data	4
2.3 Characteristics of Big Data	5
2.4 Scope of the Problem	6
2.5 Challenges	7

CHAPTER 3: REQUIREMENT SPECIFICATION	8-10
3.1 Hadoop	8
3.2 Hadoop Architecture	8
3.3 HDFS (Hadoop Distributed File system)	9
3.4 MapReduce	10
3.5 Pig	10
CHAPTER 4: DESIGN SPECIFICATION	13-18
4.1 Introduction	13
4.2 Virtual machine installation	13
4.3 Configure virtual machine	14
4.4 Loading Data into HDFS	16
CHAPTER 5: IMPLEMENTATION AND TESTING	19-30
5.1 Movie Data set	19
5.2 Movie Data Analytics	21
CHAPTER 6: CONCLUSION AND FUTURE SCOPE	31
6.1 Discussion and Conclusion	31
6.2 Scope for future Development	31
REFERENCES	32
Appendices	34
Appendix A: Project Reflection	34

LIST OF FIGURES

FIGURES	Page No
Figure 2.1: Characteristics of Big Data	5
Figure 3.1: Hadoop Architecture	9
Figure 3.2: Architecture of Apache Pig	12
Figure 4.1: Virtual Machine Installation	14
Figure 4.2: Home Screen of Edureka Virtual Machine	15
Figure 4.3: Checking all the Daemons of Hadoop	16
Figure 4.4: Copy data into HDFS from Local system	17
Figure 4.5: Data Sanity check in HDFS web log	18
Figure 5.1: Sample Movie data set in CSV format	20
Figure 5.2: Starting Pig grunt shell in MapReduce mode	22
Figure 5.3: Loading Data using Pig	23
Figure 5.4: Execution of the Pig Script of analysis 1	23
Figure 5.5: Successful execution of Pig Script of analysis 1	24
Figure 5.6: Output of the analysis 1 in HDFS	25
Figure 5.7: Pig Script of analysis 2	25
Figure 5.8: Successful execution of Pig script of analysis 2	26
Figure 5.9: Output of the analysis 2 in HDFS	27
Figure 5.10: Pig Script of the analysis 3	27
Figure 5.11: Successful execution of Pig script of analysis 3	28
Figure 5.12: Output of the Pig Script of analysis 3 in HDFS	28
Figure 5.13: Pig Script of the analysis 4	29
Figure 5.14: Successful execution of Pig script of analysis 4	29
Figure 5.15: Output of the Pig Script of analysis 4 in HDFS	30

LIST OF TABLES

TABLES

Page No

Table 5.1: Description of the Movie Data Set

19

CHAPTER 1

INTRODUCTION

1.1 Introduction

RDBMS (Relational Database Management System) was first developed by IBM in 1974. But first commercially available RDBMS was Oracle which was released in 1979. However RDBMS become popular in 1980's. By then, Oracle used a relational model system. The model was that they maintain a relation between tables using primary keys, foreign keys and indexes. However, in that time RDBMS was efficient for small and structure data sets as that time data generation speed was not so high because of lack of technology [1].

Today, we are at the time of modern technology. Technology is getting advanced day by day and new technologies are adding our life day by day. By the advancement of technology, data generation speed is high and the size of data is getting bigger. Data sets are not only bigger, their format has variations. For example, text format, JSON format, audio, video, JPEG, XML etc. For this huge and unstructured data, traditional data analytics tools like RDBMS is inefficient and can't handle various types of data which we called unstructured data. RDBMS can't handle this large data. For this, modern world need a new technology to handle this large data sets.

As world needed a new technology, they start trying to find a new tools to handle this large data set. As a result they discover new tools like Hadoop to handle this large data set and this large data set now a days called Big Data. All the giant tech company like IBM, Microsoft, Facebook, Amazon are using big data technology. Big data is a new technology and Hadoop was invented near 2006. So research is ongoing on Big Data and Hadoop. There are few tool which is built on top of Hadoop to analyze Big Data which is called Hadoop ecosystem. By using big data analytics tools like Hadoop we can store large amount of data and analyze them by using different Hadoop ecosystem tools depending on one's need .There are lots project are developing on big data. By developing this project

one can store their data and analyze them and extract value out the large data set as their need.

1.2 Motivation

Big Data market is continuously getting larger each year rapidly. Each year data is getting doubled than precious year. In March, 2012, White House announced a national “Big Data Initiative” that consisted of Six Federal Departments committing more than \$200 million on Big Data projects and researches. In 2016, Big Data is one of the top emerging technology [4]

Global Pulse is a flagship innovation initiative of the UN secretary-general on Big Data. Global Pulse’s mission is to accelerate discovery, scaled adoption and development of Big Data innovation for sustainable development and humanities action across the UN systems. [4]

In today’s world, business is getting complex and competitive day by day. To survive in competition, business analytics need to predict about the market and analyze the market. To analyze the market, they need data. They need information out of huge volume of data which is called big data. Because data never lies. Traditional DBMS can’t analyze this large volume of data. This where big data tools come to gap the bridge. Big data and its analytical tools give them better opportunities to gain information out of data and give opportunities to better visualization into data.

Apart from business organization, all other sector like healthcare organizations, Entertainment organizations, Educational organizations and many other non-profitable organization are using big data analytics. Health care organizations are using big data analytics to get information of millions of patients they stored. Without big data analytics it is unimaginable to get valuable information out of this huge data. So, we can see that, in today’s world, the effect of big data analytics in all sectors of our life which motive us to do project on Big Data analytics.

1.3 Objectives

The first purpose of our projects is to learn deep about Big Data an emerging technology and Hadoop, a big data platform. We also learn about some tools of big data analytics which together form Hadoop ecosystem. We also set up a single node cluster using single machine where we can store data and run various jobs on them. The main purpose of our project is to learn how to analyze large data set depending on different analysis using Hadoop ecosystem. As a data set, we choose a movie data set which is consist of fifty thousand data of fifty thousand different movies and analyze them to get some valuable information.

1.4 Expected Outcome

Big data analytics is a process of analyze large data set where traditional DBMS failed. After completing this project we will be able to set up single node cluster for other project. We will be able to learn about big data and one of its popular platform Hadoop and its different tools. In this project we used a movie data set consists of fifty thousand records. After analyzing this data set, output will be stored into HDFS for further visualization. Here, output will be done depending on various analysis.

1.5 Report Layout

We describe the process on how a project can be done on big Data analytics using Hadoop ecosystem. In chapter one titled **Introduction** we discuss about the project introduction, motivation, aim of the project, expected outcome and report layout. In chapter two titled **Backgroud** we discuss about the definition of Big Data, Evolution of Big Data, characteristics of big Data, scope of the problem and challenges. In chapter three titled **Requirement Specification** we discuss about Hadoop, MapReduce, Pig. In chapter four titled **Design Specification** we discuss about virtual machine installation and setup and loading data into HDFS. In chapter five title **Implementation and Testing** we show implementation of various analysis. In chapter six named **Conclusion and Future Scope** we discuss about the future scope of the problem and conclusion of our project.

CHAPTER 2

BACKGROUND

2.1 Introduction to Big Data

Big Data is a term for data sets that are so large (several gigabytes/terabytes/petabytes) and so complex that traditional data processing tools and software is inadequate to deal with them. Big Data requires new set of tools, applications and frameworks to process and manage data. [5]

Big Data is nothing but a storehouse of so large and so complex data that become very tough to store, process, retrieve and analyze them. The definition of Big Data varies from company to company depending on its size, efficiency, capacity and so on. We can't say a data set as big data only depending on its size. Data sets also have to be complex and distributed that means data need to be distributed across the different machines.

2.2 Evolution of Big Data

From the begging of modern civilization, data has been always besides us here and there. From then we need to store, process and analyze them. Before modern civilization, data was stored in paper and people process and analyze them manually. At the begging of modern civilization, people used to store data using traditional DBMS. At that time, data was not so huge and complex because of limitations of technology. Row and column oriented data was enough to store data.

But today, due to technological advancement there is huge amount of data (several terabytes/petabytes) and these data is so complex. In the era of social media more unstructured data is added to digital world daily in exponential rate. These unstructured data includes comments as text format, search history on browser, audio and video format, XML file etc. These huge size of data which is complex in type and tough to process and analyze now a days is known as Big Data.

2.3 Characteristics of Big Data

The characteristics of big data are known as four V's of big data which is very popular in the big data world. By these four V's of big data we can learn about the characteristics of big data (See in figure 2.1).

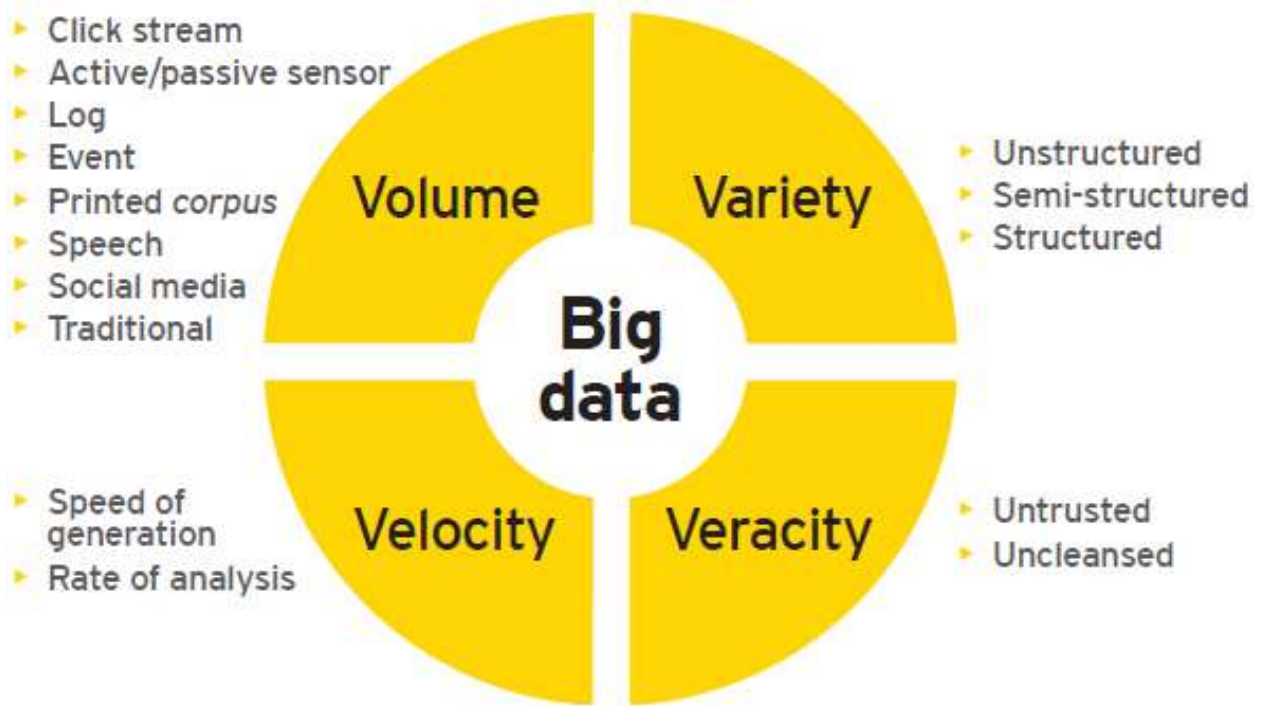


Figure 2.1 Characteristics of Big data [2]

2.3.1 Volume

Big data is primarily determined by its size. In the case of the four V's of big data, we refer to this as volume. Due to technological advancement, the amount of data being generated is growing rapidly. It could amount to hundreds of terabytes or even petabytes. Data is spread across different places, different formats in large volume ranging from gigabytes to petabytes. For instance, 15 terabytes of Facebook posts or 400 billion annual records of medical records could mean Big Data! Data is not only generated by humans but also

©Daffodil International University

generating by machines. For example, a sensor in commercial plane to check engines condition capture 500GB of data each flight [10].

2.3.2 Velocity

Velocity means the rate at which data is getting generated. As data is getting generated at high speed, Big data requires fast processing of data. Time factor plays a vital role in many organizations. For example, processing 2 million records at share market, evaluating exam paper at public examinations could mean big data [10].

2.3.3 Variety

As its name is big data it may not belongs to a specific format. Variety refers to the different formats in which data is getting generated. Apart from structure data like spreadsheet and flat format, there is large number of unstructured data is getting generated like text, audio, video, log files, mails, sensor data, social media etc. New research shows that large amount of an organizations data is not numeric which is equally important for decision making. Business organizations are making use of both structure and unstructured data for data analysis and thereby making better business decisions [10].

2.3.4 Veracity

Veracity refers to the uncertainty of data availability. Sometimes available data can get messy and difficult to trust. As there are many forms of big data, quality and accuracy are hard to control like the twitter posts with emoticons and hashtags, typos and abbreviations. But now big data technology allows us to work with these types of data. Due to uncertainty, 1 of 3 business man do not trust the information they used in decision making [10].

2.4 Scope of the problem

Our scope of the project was to run analytics on a data depending on various analysis. That's why we needed some tools of Hadoop like and we needed to learn them. Time scheduling was one of the difficult part of our project because we failed to do our project in time we would be great danger. We spend almost half of our time to analyze our problem

that means to study about big data and its tools. It took almost four months to do our analysis. We spent one month to designing our system. Coding part took one month also and implementation part took almost one and a half month to complete. In total it took seven and a half months.

2.5 Challenges

As big data analytics is analytics of large size of data which is complex in type, we faced so many challenges to work with big data analytics. As we are not a business organizations first challenge was to data collections. After collections of data we faced so many challenges that we needed to overcome [11].

As big data analytics is a young technology and resource of big data in not enough in internet, we had to face so many trouble in completing our projects. To complete our project, we first needed to learn deep about big data and big data analytics framework Hadoop and their ecosystems which was not easy for us.

Another challenge was the skills for the projects. At the beginning of the project our skill was almost zero about big data analytics. As a new technology resource was not available in the internet and project also had to deliver his skills to the other members which was tough because of communication gap. However we had to overcome all these challenges and complete our projects.

CHAPTER 3

REQUIREMENT SPECIFICATION

3.1 Hadoop

Hadoop is an open source software framework used for storing and processing Big Data in a distributed manner on large clusters of commodity hardware. Hadoop is licensed under Apache v2 therefore Hadoop is known as apache Hadoop. Hadoop was developed based on a Google file system paper which was published on October 2003. This paper was generated from another research paper of Google named MapReduce: Simplified data processing on large clusters. Hadoop is written in java and one of the highest level Apache projects. Hadoop was developed by Doug Cutting Michael J. Cafarella [6].

3.2 Hadoop Architecture

There were two major challenges in Big Data problem. One was data storage and another was data processing. As Hadoop came up as a big data solution, it solve the storage and processing issues. In Hadoop architecture, there are two major components. One is HDFS (Hadoop Distributed File System) and another is YARN (Yet Another Resource Negotiator). HDFS solved the storage issue and YARN solved the processing issues. Once data is brought into HDFS, mapreduce done with the processing part. YARN does all the resource management and scheduling these jobs. The Hadoop architecture is illustrated in figure 3.1

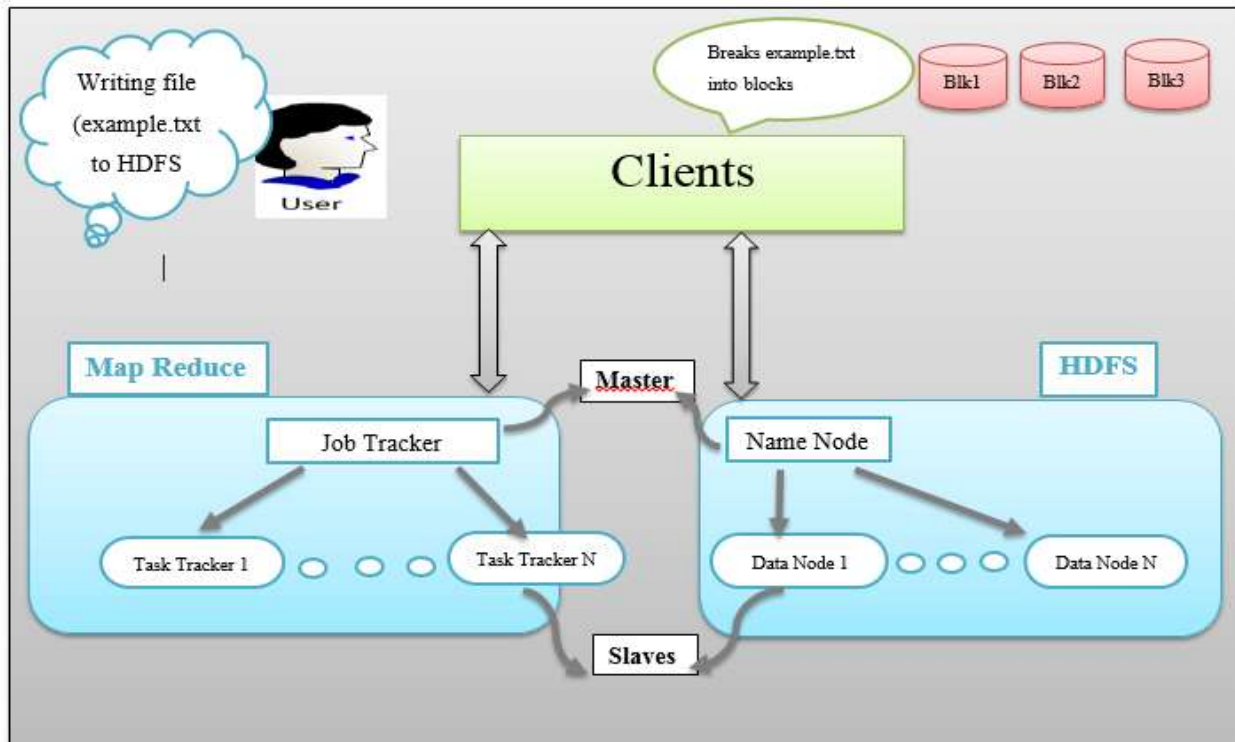


Figure 3.1: Hadoop architecture [1]

3.3 HDFS (Hadoop Distributed File System)

HDFS is one of the Hadoop's major components and it refers to the storage part. Hadoop distributed file system is a java based distributed file system that allow us to store large data across multiple nodes in a Hadoop cluster which allow to multiple access and parallel processing. So if anyone install Hadoop he/she will get HDFS as storage system in the distributed environment.

HDFS allow its user horizontal scaling. Horizontal scaling means one can upgrade its storage system without parching additional components like RAM or CPU. They just need to add more nodes to the cluster to upgrade storage. HDFS also allow multiple replication across various nodes.

HDFS works in master/slave fashion.

NameNode: Name node is the master node in a cluster. There is only one master node per cluster. It contains the metadata information of all the slave nodes.

DataNode: Data node is the slave node which actually store the data. There are multiple data node a single cluster.

3.4 MapReduce

MapReduce is the processing part in Hadoop. It performs complex analysis on large set of data in a parallel fashion. MapReduce gives us flexibility writing code without bothering design issues of the system. MapReduce consists of two distinct tasks, one is Map and another is Reduce. As name suggest, reducer phase start after the finishing of mapper phase. First is the map job where blocks of data is read and process to produce key value pairs as an intermediate output. Then the output of the mapper job is the input to the reducer job. The reducer receive key value pairs input from multiple mapper output. Then reducer aggregates those intermediate data tuples into smaller set of tuples or key value pairs which is the final output. In a simple mar reduce program written in java has three different class. Mapper class, Reduce class and Driver class. Mapper phase implements into Mapper class and reducer phase implements into Reducer class and driver class define the type of input output formats, input output key class, output path and all other stuffs are implemented into driver class. Like HDFS, MapReduce works in master/slave fashion [12].

Job Tracker: Job tracker keeps the track of the assign jobs of the every data nodes in a cluster. Its responsibility is also to reassign failed job to the slave nodes and keep information about the jobs. There is only single job tracker in a cluster.

Task Tracker: Tasks tracker runs in the data nodes and its main responsibility is to complete the jobs it is assigned by the job tracker. There are multiple task tracker in a single cluster.

3.5 PIG

Apache Pig is a high level data flow system for creating programs that run on top of Hadoop. Apache Pig is designed to provide an abstraction over MapReduce and reducing

the complexity of writing MapReduce program. Pig was developed by Yahoo in 2006 to allow developer using Hadoop focusing more on analyzing large data sets and spend less time on writing MapReduce programs. As name suggest Pig which eat almost everything, Apache Pig was designed to handle all kinds of data – hence the name!

Pig gives flexibility to programmers to write MapReduce job who don't know programming language like java or python. Pig helps developers to write MapReduce job as approximately 10 lines of Pig script is equal to 200 lines of MapReduce program! Hence it reduce the development times 20 times shorter. That is why every developer chose Pig as a data processing language over MapReduce java program although they know java or python [13].

3.5.1 Pig Architecture

Apache Pig is made of two major components, one is the language itself which is known as **PigLatin**, and another one is the runtime environment where PigLatin programs are executed which is known as **PigLatin compiler**. Just like java virtual machine and java application. The architecture of apache Pig is illustrate in fig 3.2.

3.5.1.1 PigLatin Script

PigLatin script is a programming language of apache Pig. There are three way to run piglatin script. First one is grunt shell, where we write piglatin script in the grunt shell and execute it. Second one is to write the script into a script file and run it in the command line. Another one is Pig user defined function which extends the pigs built in functionality.

3.5.1.2 PigLatin Compiler

Pig Latin compiler converts the pig latin codes into executable code. The executable code is in form of MapReduce jobs. The sequence of MapReduce programs enable pig programs to do data processing and analyzing in parallel.

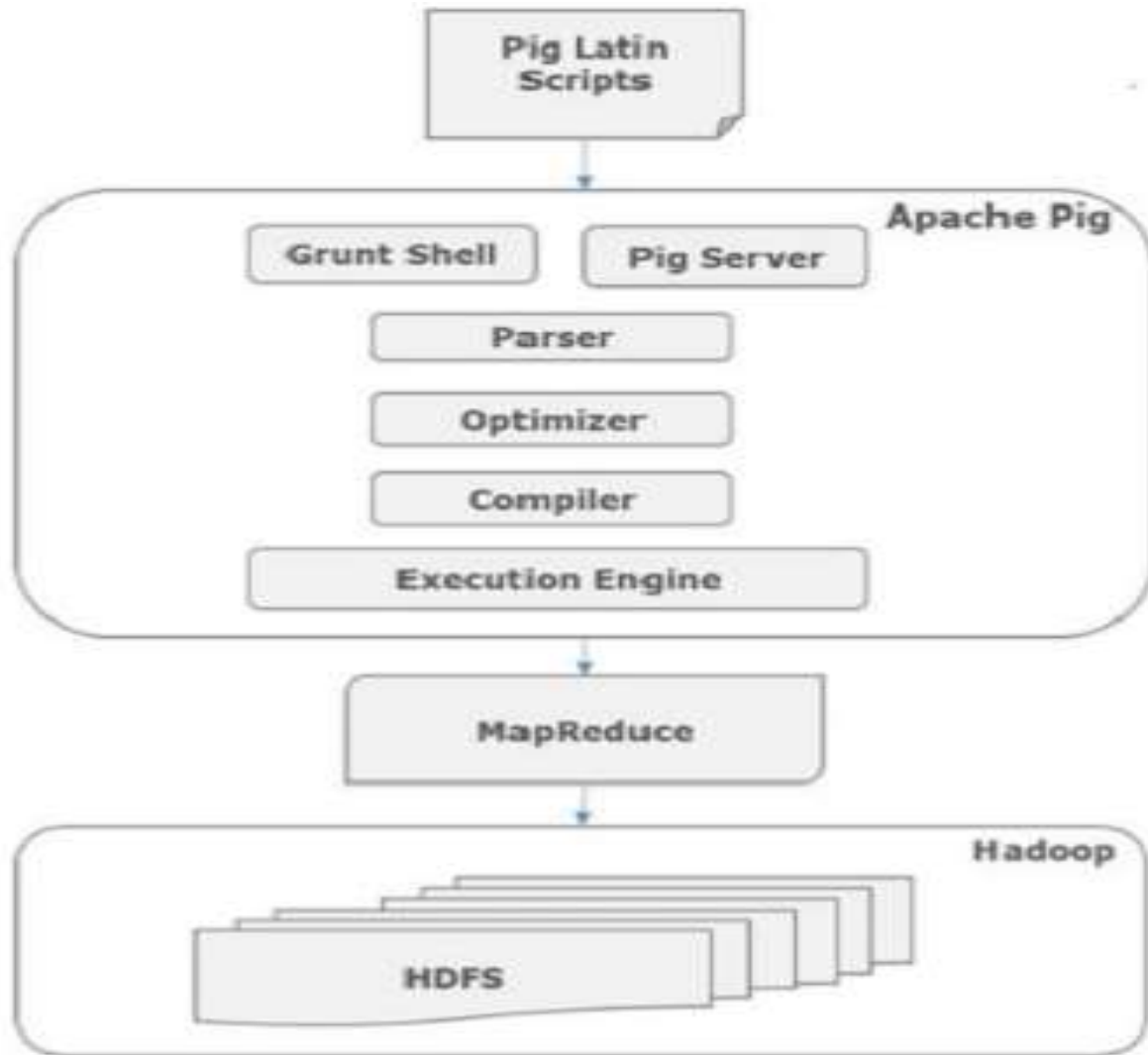


Figure 3.2: Architecture of Apache Pig [3]

The difference between MapReduce and PigLatin is that we can achieve everything with MapReduce and we can achieve almost everything with PigLatin. That means in some cases PigLatin may not work properly like so many loops need to be used. Otherwise Pig Latin is not very mature and it was developed to handle ad-hoc level data processing.

CHAPTER 4

DESIGN SPECIFICATION

4.1 Introduction

We need to install Hadoop to work with Big Data. As most of us are not comfortable with UNIX operating system we will take help of virtual machine. A Virtual machine is a software computer like physical a computer that runs an operating system and applications. Virtual machine comes up with some build in big data tools like Hadoop, Hbase, Spark, Impala etc. There are so many virtual machine in the internet like cloudera vm, sandbox vm, edureka vm. Cloudera vm is most popular of them. We use edureka virtual machine for our project. Edureka virtual machine comes with CentOS 6.5 operating system and all other Big Data tools.

4.2 Virtual Machine Installation

Installation of Edureka Virtual machine is very useful for working with Hadoop as it comes with some pre-built Big Data tools.

Minimum hardware requirements for installing Virtual machine on Windows are

- Minimum 4 GB RAM or above
- Dual core processor or above
- 20 GB free Hard Disk space to run VM smoothly

These step by step need to be followed to install virtual machine:

Step-1: Download open source virtual box from below link [7]

<https://www.virtualbox.org/wiki/Downloads>

Step-2: Download Edureka virtual machine. As it is not an open source virtual machine, normal user can't download it, only some user who paid for it can download it. Other can download Cloudera virtual machine from the below link [8]

https://www.cloudera.com/downloads/quickstart_vms/5-8.html

Step-3: After downloading both first we need to install virtual box. After installing below screen illustrate in figure 4.1 will appear. Then we need to import **.ova** file into the virtual box.

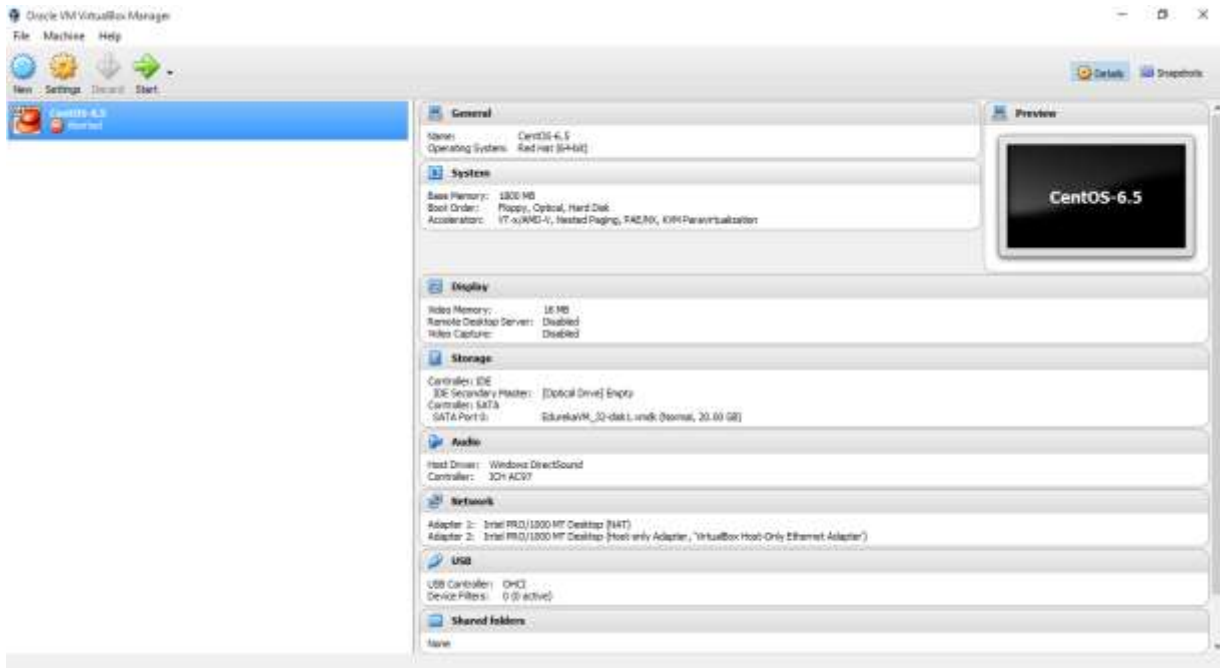


Figure 4.1: Virtual machine installation

4.3 Virtual machine configuration

After importing virtual machine we need to adjust hardware and network configuration.

First go to setting and then go to system setting. Path is **setting->system**

Then we need to specify **Base RAM**. While selecting base ram always select **30-35%** of total ram to run vm smoothly.

Then we need to specify network setting. Go to **setting->network** and select adapter 1 as **NAT** and Adapter 2 as **Host-only-Adapter**.

Then click start in start button. It will take some times and virtual machine home screen will appear.

Select Edureka and type password as **“edureka”**. In case of Cloudera type “cloudera” as password. Then home screen will apear like figure 4.2

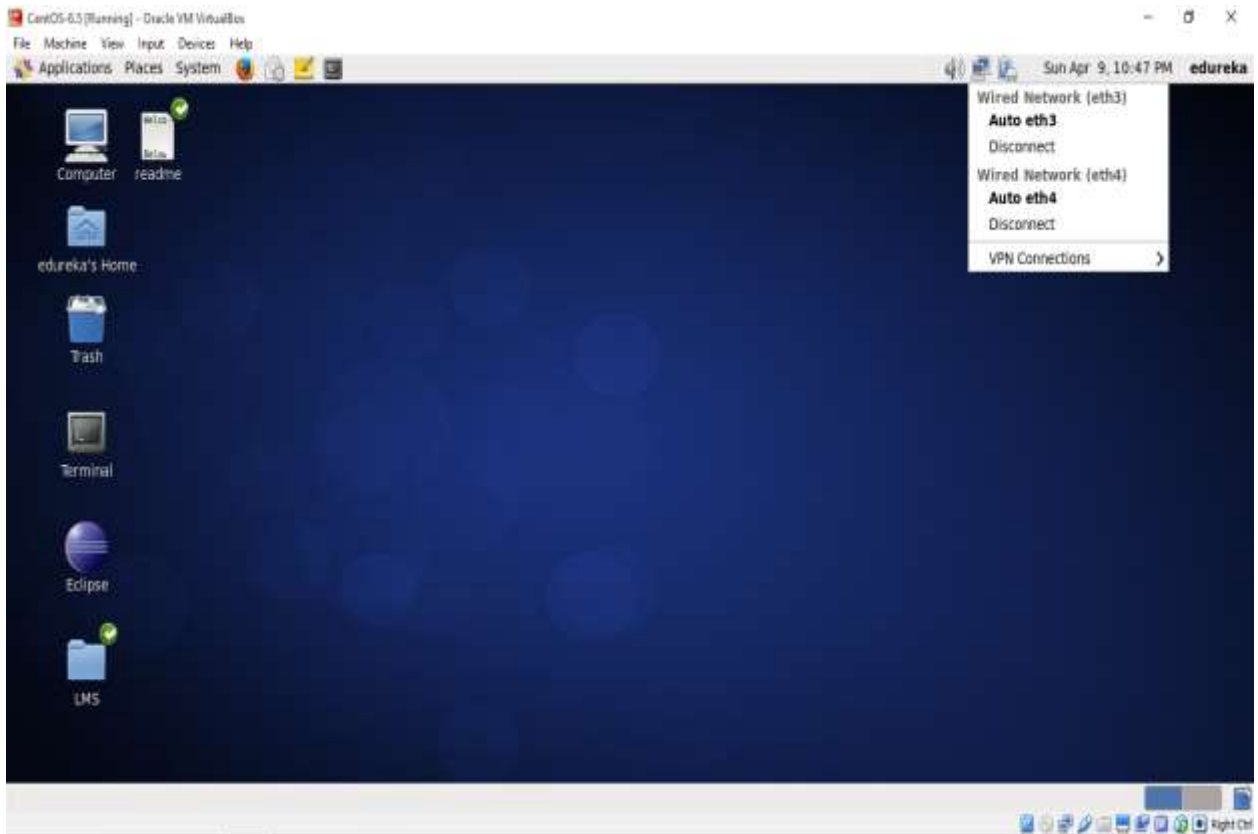
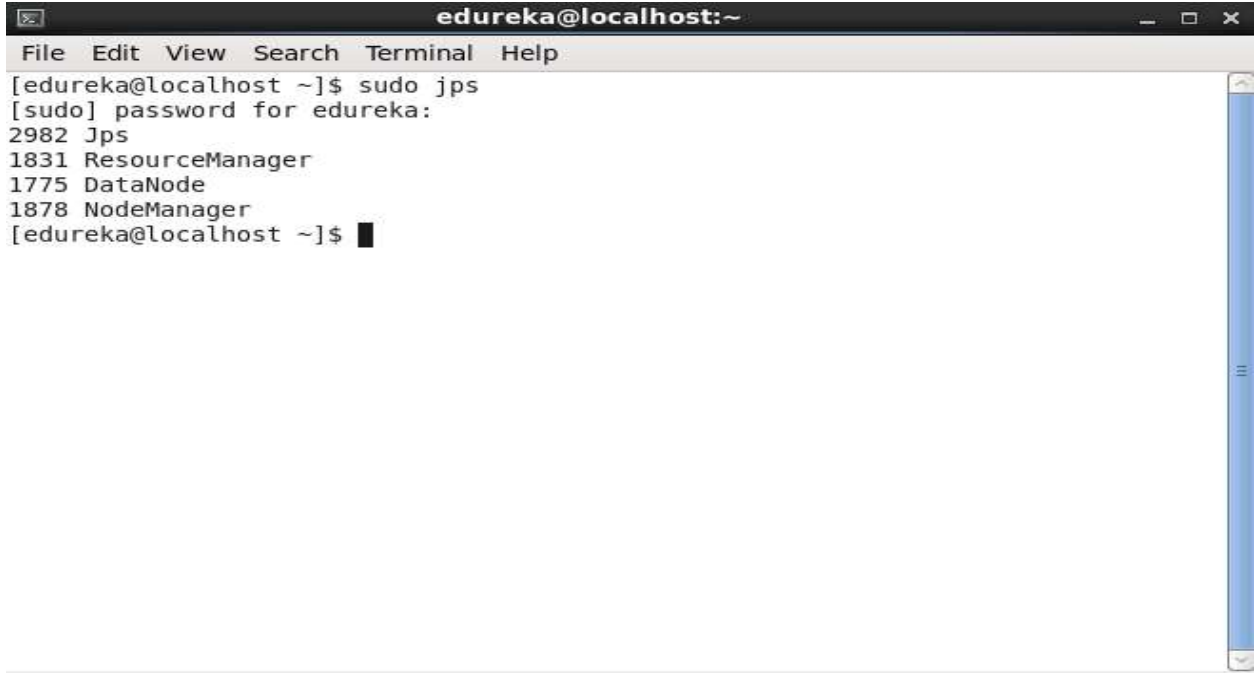


Figure 4.2: Home screen of Edureka Virtual Machine

4.4 Loading Data into HDFS

Before working with Hadoop we need to check if all the daemons are running properly or not. The command is **sudo jps** and give password as “edureka”. The output will be like figure 4.4. As we are working on pseudo distributed mode that means name node and data node are in the same machine the output will be like exactly the figure 4.4.



```
edureka@localhost:~  
File Edit View Search Terminal Help  
[edureka@localhost ~]$ sudo jps  
[sudo] password for edureka:  
2982 Jps  
1831 ResourceManager  
1775 DataNode  
1878 NodeManager  
[edureka@localhost ~]$
```

Figure 4.3: Checking all the daemons of Hadoop

First we need transfer file from local machine to virtual machine using **FileZilla** a FTP utility which is used to transfer file over the internet.

Following Hadoop commands is that are used to perform different operation on HDFS.

\$hadoop version: to check Hadoop version

\$hadoop dfs -mkdir: to create directory into HDFS

\$hadoop dfs -copyFromLocal <file_path> hdfs:/ : to upload file into HDFS

\$hadoop dfs -rm -r: to remove a file from HDFS

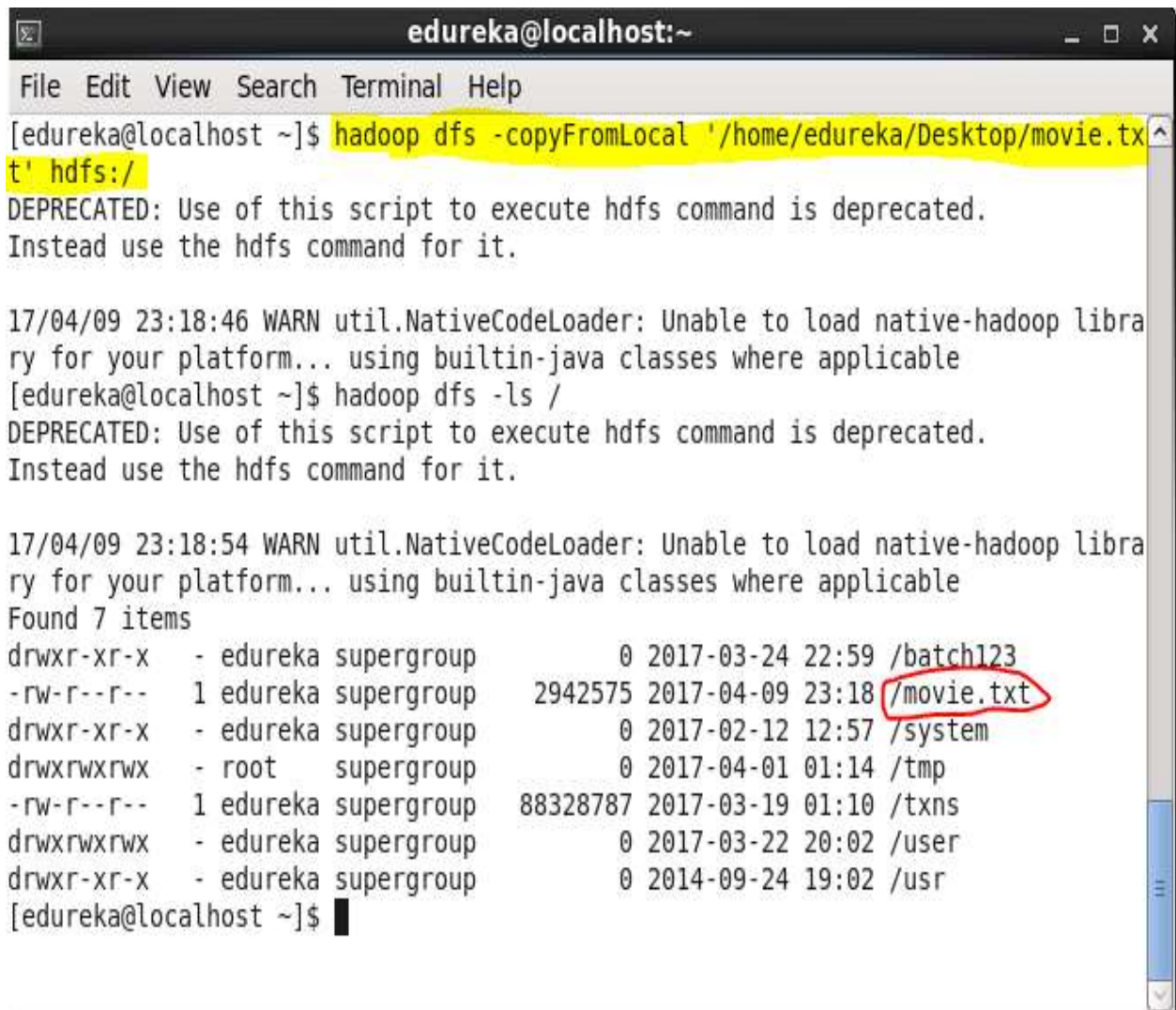
\$hadoop dfs -ls: check the file list in the HDFS

Step-1: Move file name movite.txt into HDFS from local system using below command

\$hadoop dfs -copyFromLocal '/home/edureka/Desktop/movie.txt' hdfs:/ asfigure 4.5

Step-2 : check whether file has uploaded into HDFS or not using below command

\$hadoop dfs -ls / as figure 4.5



```
edureka@localhost:~  
File Edit View Search Terminal Help  
[edureka@localhost ~]$ hadoop dfs -copyFromLocal '/home/edureka/Desktop/movie.tx  
t' hdfs:/  
DEPRECATED: Use of this script to execute hdfs command is deprecated.  
Instead use the hdfs command for it.  
  
17/04/09 23:18:46 WARN util.NativeCodeLoader: Unable to load native-hadoop libra  
ry for your platform... using builtin-java classes where applicable  
[edureka@localhost ~]$ hadoop dfs -ls /  
DEPRECATED: Use of this script to execute hdfs command is deprecated.  
Instead use the hdfs command for it.  
  
17/04/09 23:18:54 WARN util.NativeCodeLoader: Unable to load native-hadoop libra  
ry for your platform... using builtin-java classes where applicable  
Found 7 items  
drwxr-xr-x - edureka supergroup 0 2017-03-24 22:59 /batch123  
-rw-r--r-- 1 edureka supergroup 2942575 2017-04-09 23:18 /movie.txt  
drwxr-xr-x - edureka supergroup 0 2017-02-12 12:57 /system  
drwxrwxrwx - root supergroup 0 2017-04-01 01:14 /tmp  
-rw-r--r-- 1 edureka supergroup 88328787 2017-03-19 01:10 /txns  
drwxrwxrwx - edureka supergroup 0 2017-03-22 20:02 /user  
drwxr-xr-x - edureka supergroup 0 2014-09-24 19:02 /usr  
[edureka@localhost ~]$
```

Figure 4.4: Copy data into HDFS from local system

Step-3 : After copying file from local file system to HDFS a sanity check can be performed to check for the data present in the HDFS file browser or not. Steps are below

First open Mozilla Firefox

Then in the search button type **localhost:50075** and press enter which will take us to the NameNode WebUI.

Then go to browse the file system and check if it is available or not. Output will be exactly the figure 4.6

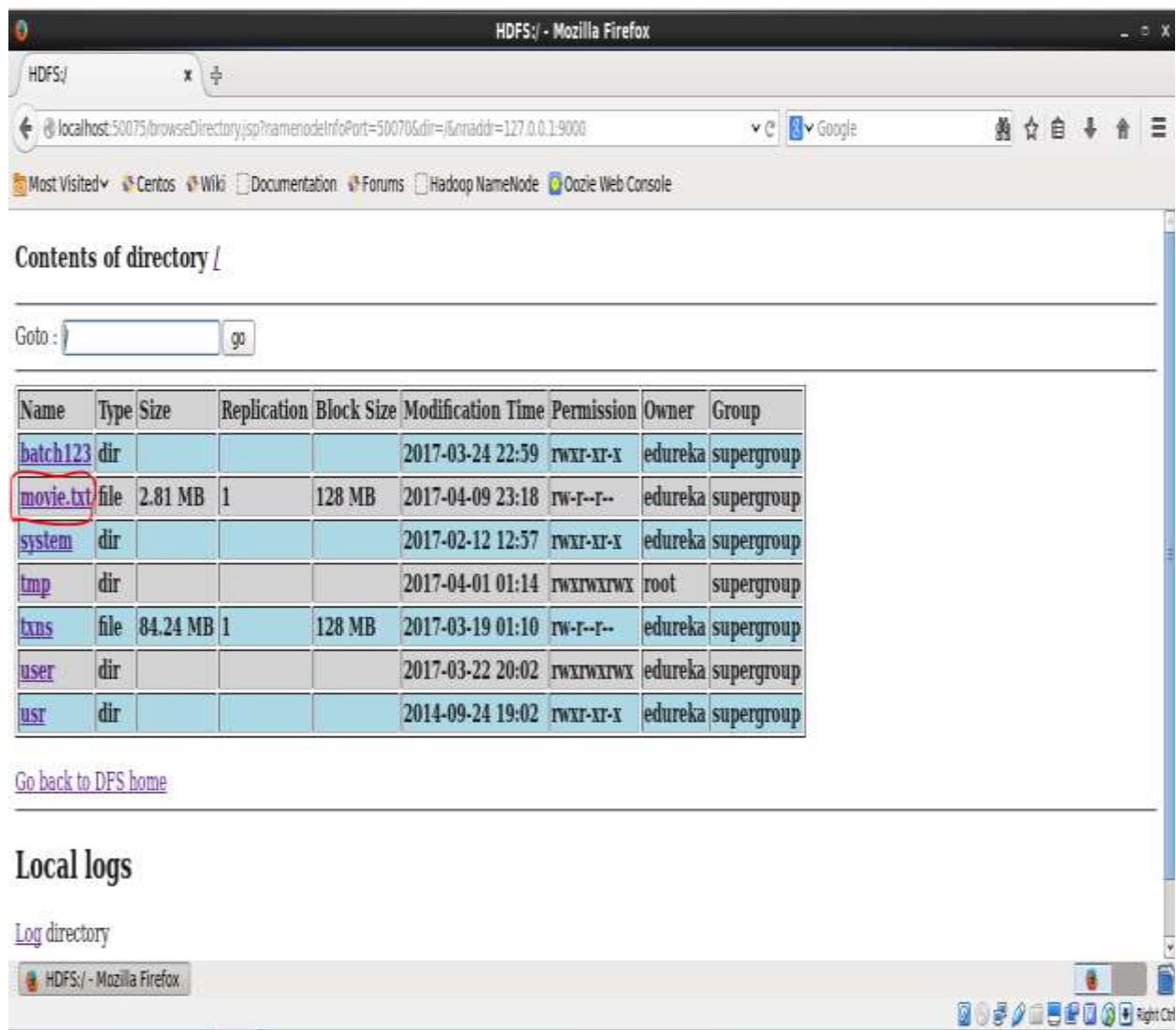


Figure 4.5: Data sanity check in HDFS web log

CHAPTER 5

IMPLEMENTATION AND TESTING

In this chapter we will describe about movie data set we are using for our analysis and each steps of analysis.

5.1 Movie data set

Movie data set consists of almost fifty thousand records of movie collection from 1913 to 2014. We collect our test data set from the following link.

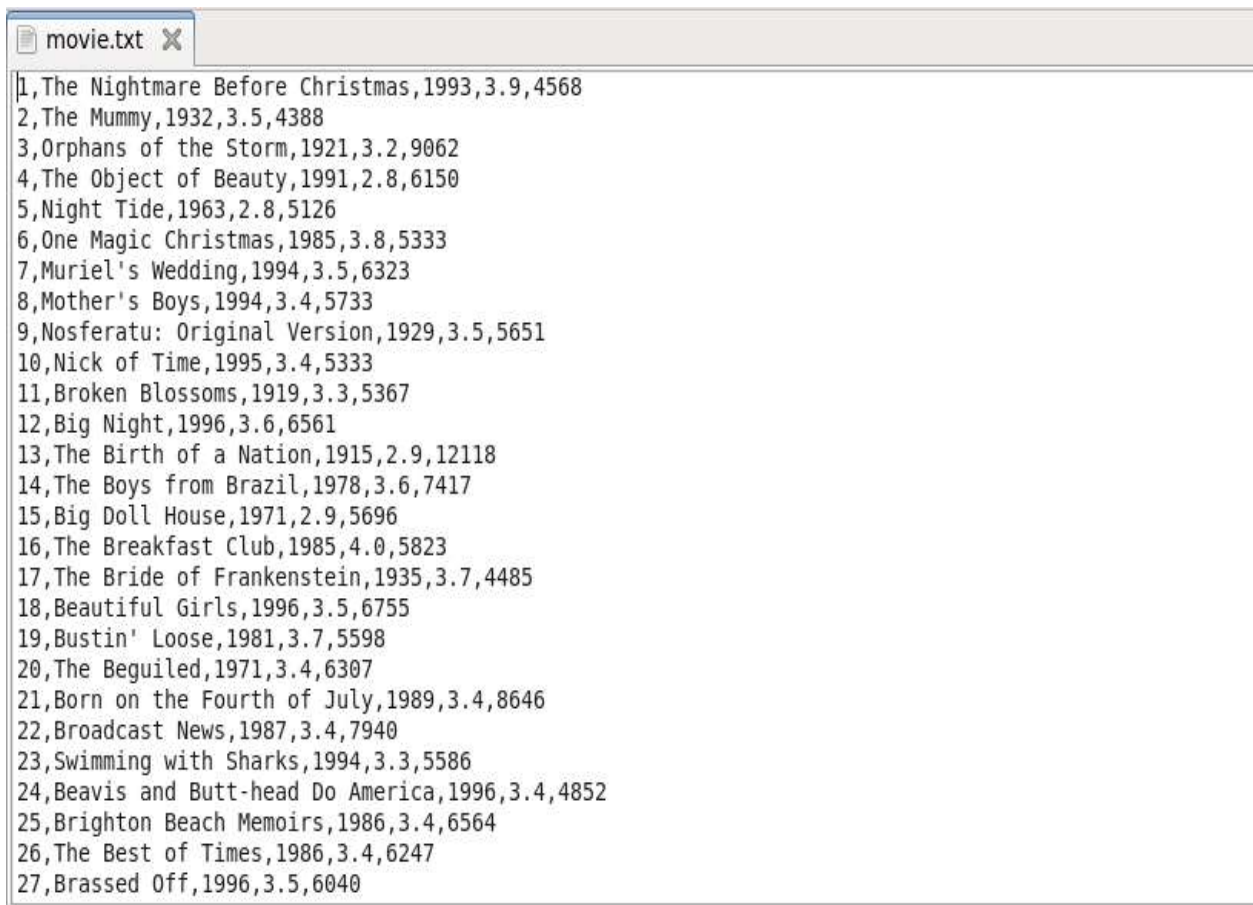
<https://edureka.wistia.com/medias/7qd5lgmko4>

Although data is in comma separated value (CSV) which is a structured data we will analyze our data at Big Data analytics fashion. A detailed description of the movie data set is described in table 5.1

Table 5.1: Description of the movie data set

Field	Description of the field
Column 1	Id of the movie
Column 2	Name of the movie
Column 3	Year of released of the movie
Column 4	Rating of the movie
Column 5	Duration in seconds of the movie

From description we can see that data set is structured although Hadoop was found for unstructured data analytics. We will analyze our data set in a Big data analytics fashion as we are not in the production environment and collecting of unstructured data is so tough. We chose Pig as an analytical tool as we are not from a Java background and Pig helped us in reducing development time. A snippet of the CSV (comma separated value) format movie data set is shown in the Figure 5.1



```
1,The Nightmare Before Christmas,1993,3.9,4568
2,The Mummy,1932,3.5,4388
3,Orphans of the Storm,1921,3.2,9062
4,The Object of Beauty,1991,2.8,6150
5,Night Tide,1963,2.8,5126
6,One Magic Christmas,1985,3.8,5333
7,Muriel's Wedding,1994,3.5,6323
8,Mother's Boys,1994,3.4,5733
9,Nosferatu: Original Version,1929,3.5,5651
10,Nick of Time,1995,3.4,5333
11,Broken Blossoms,1919,3.3,5367
12,Big Night,1996,3.6,6561
13,The Birth of a Nation,1915,2.9,12118
14,The Boys from Brazil,1978,3.6,7417
15,Big Doll House,1971,2.9,5696
16,The Breakfast Club,1985,4.0,5823
17,The Bride of Frankenstein,1935,3.7,4485
18,Beautiful Girls,1996,3.5,6755
19,Bustin' Loose,1981,3.7,5598
20,The Beguiled,1971,3.4,6307
21,Born on the Fourth of July,1989,3.4,8646
22,Broadcast News,1987,3.4,7940
23,Swimming with Sharks,1994,3.3,5586
24,Beavis and Butt-head Do America,1996,3.4,4852
25,Brighton Beach Memoirs,1986,3.4,6564
26,The Best of Times,1986,3.4,6247
27,Brassed Off,1996,3.5,6040
```

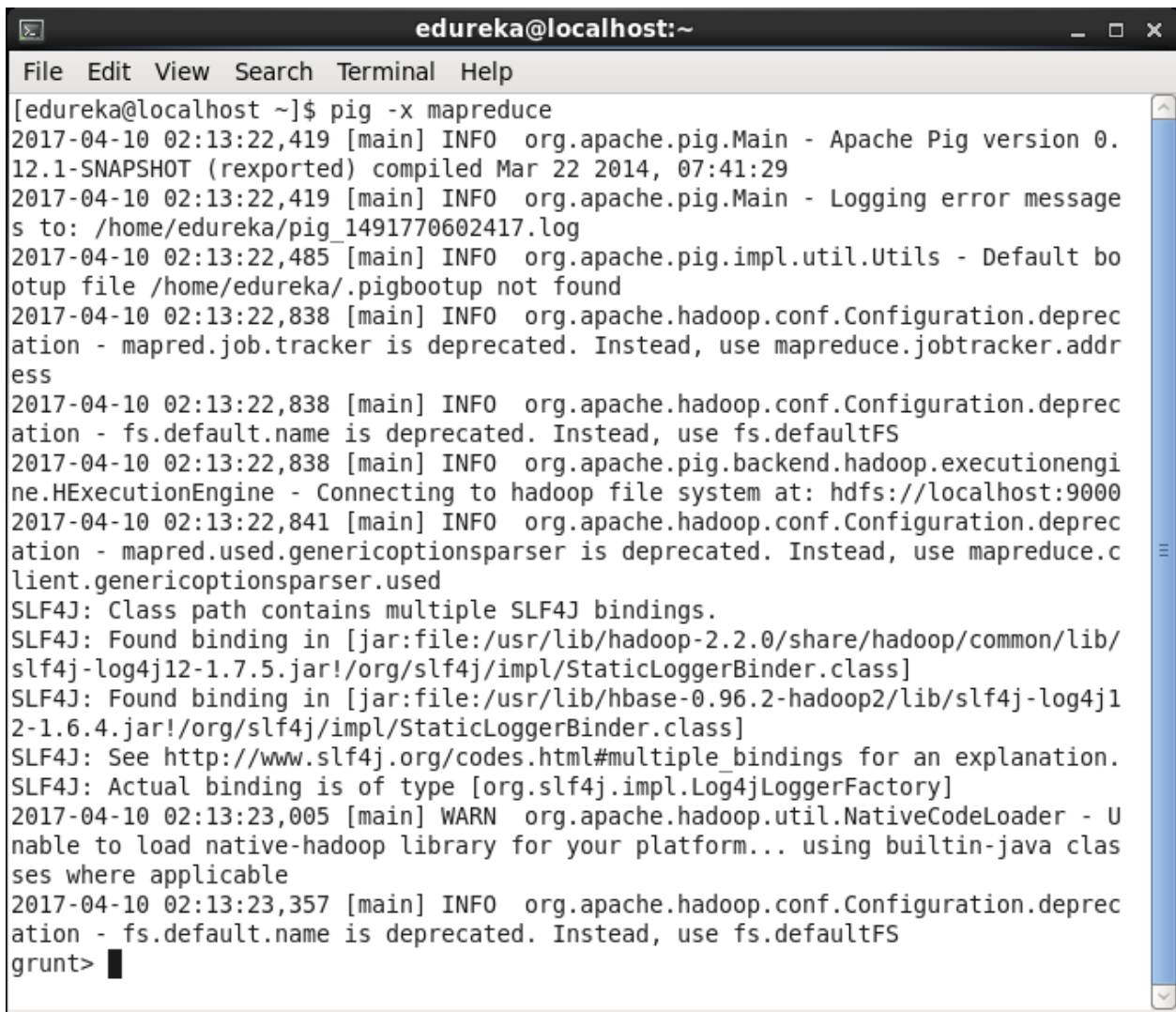
Figure 5.1 Sample movie data set in CSV format

5.2 Movie Data Analytics

Different analysis have been considered to perform data analytics on movie data set. Detailed on the different analysis and implementation details have been provided below. The output of the data analytics also have been provided.

5.2.1 Analysis 1: Number of movies released between 1950 and 1960

First we need to load our data into HDFS which we have done in previous chapter. Now we need to enter into Pig grunt shell. There are two mode of Pig grunt shell, one is local mode and another is mapreduce mode. To enter into the local mode we will type **pig -x local** and to enter into mapreduce mode we will type **pig -x mapreduce**. As we are working on mapreduce mode we will type **pig -x mapreduce** and then figure illustrate in figure 5.2 will appear.



```
edureka@localhost:~
File Edit View Search Terminal Help
[edureka@localhost ~]$ pig -x mapreduce
2017-04-10 02:13:22,419 [main] INFO org.apache.pig.Main - Apache Pig version 0.12.1-SNAPSHOT (rexported) compiled Mar 22 2014, 07:41:29
2017-04-10 02:13:22,419 [main] INFO org.apache.pig.Main - Logging error messages to: /home/edureka/pig_1491770602417.log
2017-04-10 02:13:22,485 [main] INFO org.apache.pig.impl.util.Utils - Default bootstrap file /home/edureka/.pigbootstrap not found
2017-04-10 02:13:22,838 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.address
2017-04-10 02:13:22,838 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2017-04-10 02:13:22,838 [main] INFO org.apache.pig.backend.hadoop.executionengine.HExecutionEngine - Connecting to hadoop file system at: hdfs://localhost:9000
2017-04-10 02:13:22,841 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - mapred.used.genericoptionsparser is deprecated. Instead, use mapreduce.client.genericoptionsparser.used
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/lib/hadoop-2.2.0/share/hadoop/common/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/hbase-0.96.2-hadoop2/lib/slf4j-log4j12-1.6.4.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
2017-04-10 02:13:23,005 [main] WARN org.apache.hadoop.util.NativeCodeLoader - Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
2017-04-10 02:13:23,357 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
grunt>
```

Figure 5.2: Starting Pig grunt shell in MapReduce mode

Loading data using pig is shown in figure 5.3

A terminal window titled 'edureka@localhost:~' with a menu bar (File, Edit, View, Search, Terminal, Help). The command prompt shows a Pig script: 'grunt> data = load '/movie.txt' using PigStorage(',') as (id : int, name : chararray, year : int, rating : float, duration : int);' followed by a new line 'grunt>'.

```
edureka@localhost:~  
File Edit View Search Terminal Help  
grunt> data = load '/movie.txt' using PigStorage(',') as (id : int, name : chararray, year : int, rating : float, duration : int);  
grunt>
```

Figure 5.3 Loading data using Pig

A Pig script was written in grunt shell to accomplish this which is visible in figure 5.4

A terminal window titled 'edureka@localhost:~' with a menu bar (File, Edit, View, Search, Terminal, Help). The command prompt shows a Pig script: 'grunt> data = load '/movie.txt' using PigStorage(',') as (id : int, name : chararray, year : int, rating : float, duration : int);' followed by several lines of Pig code. The output shows log messages from the Pig engine, including 'Pig features used in the script: GROUP BY, FILTER' and 'Columns pruned for data: \$0, \$3, \$4'.

```
edureka@localhost:~  
File Edit View Search Terminal Help  
grunt> data = load '/movie.txt' using PigStorage(',') as (id : int, name : chararray, year : int, rating : float, duration : int);  
grunt> a = foreach data generate name, year;  
grunt> b = filter a by year >= 1950;  
grunt> c = filter b by year <= 1960;  
grunt> d = foreach c generate 1 as one;  
grunt> e = group d all;  
grunt> f = foreach e generate SUM(d.one);  
grunt> store f into '/useCase1' using PigStorage('\t');  
2017-04-10 02:31:19,870 [main] INFO org.apache.pig.tools.pigstats.ScriptState - Pig features used in the script: GROUP BY, FILTER  
2017-04-10 02:31:19,871 [main] INFO org.apache.pig.newplan.logical.optimizer.LogicalPlanOptimizer - {RULES_ENABLED=[AddForEach, ColumnRewrite, GroupByConstParallelSetter, ImplicitSplitInserter, LimitOptimizer, LoadTypeCastInserter, MergeFilter, MergeForEach, NewPartterOptimizer, PushDownForEachFlatten, PushUpFilter, SplitFilter, StreamTypeCastInserter], RULES_DISABLED=[FilterLogicExpressionSimplif  
2017-04-10 02:31:19,873 [main] INFO org.apache.pig.newplan.logical.rules.ColumnPruneVisitor - Columns pruned for data: $0, $3, $4  
2017-04-10 02:31:19,883 [main] INFO org.apache.pig.newplan.logical.rules.ColumnPruneVisitor - Columns pruned for data: $0, $3, $4
```

Figure 5.4 Execution of the script of the analysis 1

Figure 5.5 show the successfully execution of the script

```

edureka@localhost:~
File Edit View Search Terminal Help
er at /0.0.0.0:8832
2017-04-18 02:31:24,317 [JobControl] INFO org.apache.hadoop.mapreduce.Job - The url to track the job: http://localhost:8888/proxy/application_1491759183721_08802/
2017-04-18 02:31:24,317 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - HadoopJobId: job_1491759183721_08802
2017-04-18 02:31:24,317 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Processing aliases b,d,data,e,f
2017-04-18 02:31:24,317 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - detailed locations: M: data[11,7],data[-1,-1],b[13,4],d[-1,-1],f[-1,-1],e[16,4] C: f[-1,-1],e[16,4] R: f[-1,-1]
2017-04-18 02:31:24,382 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 0% complete
2017-04-18 02:31:36,639 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 50% complete
2017-04-18 02:31:44,615 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 100% complete
2017-04-18 02:31:44,616 [main] INFO org.apache.pig.tools.pigstats.SimplePigStats - Script Statistics:

HadoopVersion PigVersion UserId StartedAt FinishedAt Features
2.2.0 0.12.1-SNAPSHOT edureka 2017-04-18 02:31:19 2017-04-18 02:31:44 GROUP_BY,FILTER

Success!

Job Stats (time in seconds):
JobId Maps Reduces MaxMapTime MinMapTime AvgMapTime MedianMapTime MaxReduceTime MinReduceTime AvgReduceTime MedianReduceTime Alias F
eature Outputs
job_1491759183721_08802 1 1 3 3 3 3 3 3 3 3 b,d,data,e,f GROUP_BY,COMBINER /useCase1,

Input(s):
Successfully read 49590 records (2942925 bytes) from: "/movie.txt"

Output(s):
Successfully stored 1 records (4 bytes) in: "/useCase1"

Counters:
Total records written : 1
Total bytes written : 4
Spillable Memory Manager spill count : 0
Total bags proactively spilled: 0
Total records proactively spilled: 0

Job DAG:
job_1491759183721_08802

2017-04-18 02:31:44,713 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Success!
grunt>

```

Figure 5.5 Successful execution of the script of analysis 1

Below screenshot show the result of script stored into HDFS.

File: [/useCase1/part-r-00000](#)

Goto :

[Go back to dir listing](#)
[Advanced view/download options](#)

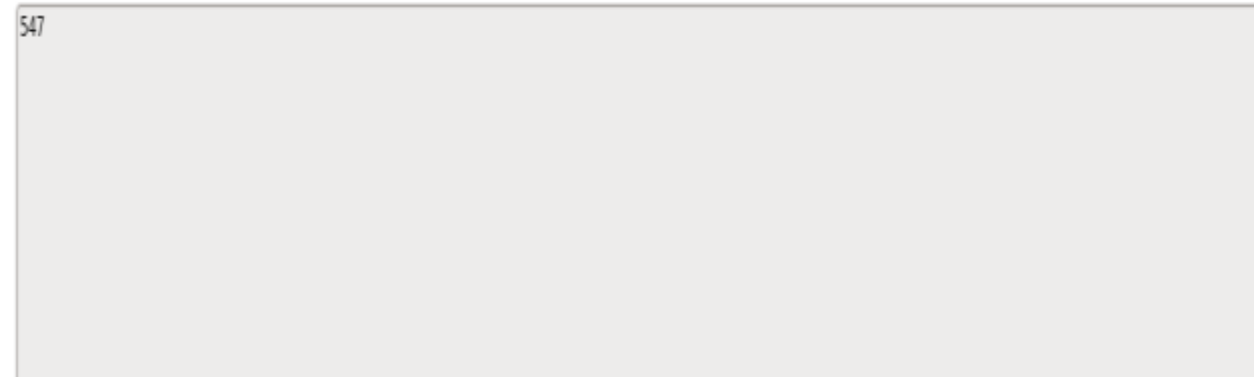


Figure 5.6 Output of the analysis 1 in HDFS

5.2.2 Analysis 2: Find movies having rating more than 4

As we already load data using data alias we don't need to load data again into pig. We just need to write another script to find the movies having rating more than 4. Figure 5.7 illustrate the script.

```
edureka@localhost:~$ cat script2.pig
grunt> data = load '/movie.txt' using PigStorage(',') as (id : int, name : chararray , year : int , rating : float , duration : int);
grunt> a = foreach data generate name,rating;
grunt> b = filter a by rating>=4.0F;
grunt> store b into '/useCase2' using PigStorage(',');
2017-04-10 03:40:25,359 [main] INFO org.apache.pig.tools.pigstats.ScriptState - Pig features used in the script: FILTER
2017-04-10 03:40:25,360 [main] INFO org.apache.pig.newplan.logical.optimizer.LogicalPlanOptimizer - {RULES_ENABLED=[AddForEach, ColumnMapKeyPrune, DuplicateForEachColumnRewrite, GroupByConstParallelSetter, ImplicitSplitInserter, LimitOptimizer, LoadTypeCastInserter, MergeFilter, MergeForEach, NewPartitionFilterOptimizer, PartitionFilterOptimizer, PushDownForEachFlatten, PushUpFilter, SplitFilter, StreamTypeCastInserter], RULES_DISABLED=[FilterLogicExpressionSimplifier]}
2017-04-10 03:40:25,361 [main] INFO org.apache.pig.newplan.logical.rules.ColumnPruneVisitor - Columns pruned for data: $0, $2, $4
2017-04-10 03:40:25,364 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MRCompiler - File concatenation threshold: 100 optimistic? false
2017-04-10 03:40:25,365 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MultiQueryOptimizer - MR plan size before optimization: 1
2017-04-10 03:40:25,365 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MultiQueryOptimizer - MR plan size after optimization: 1
```

Figure 5.7: Pig script of analysis 2

After successful execution of the below figure is show in the command shell.

```

edureka@localhost:~
File Edit View Search Terminal Help
2017-04-10 03:40:29,367 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - detailed location: M: data[20,7],data[-1,-1],b[22,4],a[21,4] C: R:
2017-04-10 03:40:29,425 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 0% complete
2017-04-10 03:40:39,084 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 50% complete
2017-04-10 03:40:44,634 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 100% complete
2017-04-10 03:40:44,634 [main] INFO org.apache.pig.tools.pigstats.SimplePigStats - Script Statistics:

HadoopVersion PigVersion UserId StartedAt FinishedAt Features
2.2.0 0.12.1-SNAPSHOT edureka 2017-04-10 03:40:25 2017-04-10 03:40:44 FILTER

Success!

Job Stats (time in seconds):
JobId Maps Reduces MaxMapTime MinMapTime AvgMapTime MedianMapTime MaxReduceTime MinReduceTime AvgReduceTimeM
edianReductime Alias Feature Outputs
job_1491759183721_0003 1 0 2 2 2 2 n/a n/a n/a n/a a,b,data MAP_ONLY /
useCase2,

Input(s):
Successfully read 49590 records (2942925 bytes) from: "/movie.txt"

Output(s):
Successfully stored 1477 records (40336 bytes) in: "/useCase2"

Counters:
Total records written : 1477
Total bytes written : 40336
Spillable Memory Manager spill count : 0
Total bags proactively spilled: 0
Total records proactively spilled: 0

Job DAG:

```

Figure 5.8: Successful execution of Pig script of analysis 2

Then we need to check output into the HDFS. Below figure shows the output of the analysis 2.

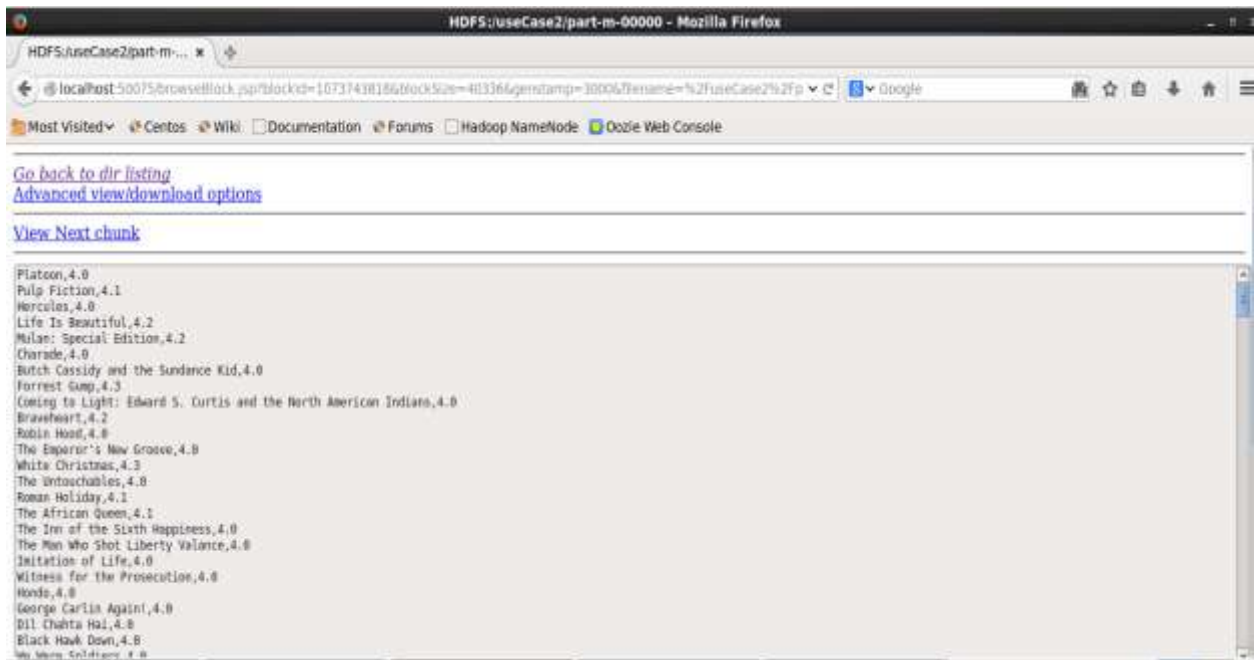


Figure 5.9: Output of the analysis 2 in HDFS

5.2.3 Analysis 3: Finding movies having rating between 3 and 4

To finding the movies which rating is between 3 and 4 we need to write a script that is illustrate in figure 5.10



Figure 5.10: Pig script of analysis 3

Below figure show the successful execution of pig script of analysis 3.

```

edureka@localhost:~
File Edit View Search Terminal Help
2017-04-10 04:07:49,623 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 0% complete
2017-04-10 04:07:59,351 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 50% complete
2017-04-10 04:08:05,066 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 100% complete
2017-04-10 04:08:05,068 [main] INFO org.apache.pig.tools.pigstats.SimplePigStats - Script Statistics:

HadoopVersion PigVersion UserId StartedAt FinishedAt Features
2.2.0 0.12.1-SNAPSHOT edureka 2017-04-10 04:07:45 2017-04-10 04:08:05 FILTER

Success!

Job Stats (time in seconds):
JobId Maps Reduces MaxMapTime MinMapTime AvgMapTime MedianMapTime MaxReduceTime MinReduceTime AvgReduceTime MedianReduceTime Alias F
easure Outputs
job_1491759183721_0006 1 0 2 2 2 2 n/a n/a n/a n/a a,b,data MAP_ONLY /useCase3,

Input(s):
Successfully read 49590 records (2942925 bytes) from: "/movie.txt"

Output(s):
Successfully stored 8222 records (206345 bytes) in: "/useCase3"

Counters:
Total records written : 8222
Total bytes written : 206345
Spillable Memory Manager spill count : 0
Total bags proactively spilled: 0
Total records proactively spilled: 0

Job DAG:
job_1491759183721_0006

```

Figure 5.11 Successful execution of Pig script of analysis 3

Below figure show the output of the analysis 3



Figure 5.12 Output of the pig script of the analysis 3

5.2.3 Analysis 4: Finding List of Year and Number of Movies Each Year

To find list of year from the data set and the number of movies each year we need to write a pig script. As data set is already loaded into pig we don't need to load data again into the pig. Below figure show the script of the use analysis 4.



```
edureka@localhost:~  
File Edit View Search Terminal Help  
grunt> data = load '/movie.txt' using PigStorage(',') as (id:int,name:chararray,year:int,rating:float,duration:int);  
grunt> a = foreach data generate year;  
grunt> b = group a by year;  
grunt> c = foreach b generate group as year, COUNT($1);  
grunt> store c into '/useCase4' using PigStorage('t');  
2017-04-10 04:27:35,176 [main] INFO org.apache.pig.tools.pigstats.ScriptState - Pig features used in the script: GROUP BY  
2017-04-10 04:27:35,177 [main] INFO org.apache.pig.newplan.logical.optimizer.LogicalPlanOptimizer - {RULES_ENABLED=[AddForEach, ColumnMapKeyPrune, DuplicateForEachColumnRewrite, GroupByConstParallelSetter, ImplicitSplitInserter, LimitOptimizer, LoadTypeCastInserter, MergeFilter, MergeForEach, NewPartitionFilterOptimizer, PartitionFilterOptimizer, PushDownForEachFlatten, PushUpFilter, SplitFilter, StreamTypeCastInserter], RULES_DISABLED=[FilterLogicExpressionsSimplifier]}  
2017-04-10 04:27:35,178 [main] INFO org.apache.pig.newplan.logical.rules.ColumnPruneVisitor - Columns pruned for data: $0, $1, $3, $4  
2017-04-10 04:27:35,189 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MRCompiler - File concatenation threshold: 100 optimistic? false  
2017-04-10 04:27:35,195 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.CombineOptimizer - Choosing to move algebraic foreach to combiner  
2017-04-10 04:27:35,207 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MultiQueryOptimizer - MR plan size before optimization: 1  
2017-04-10 04:27:35,207 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MultiQueryOptimizer - MR plan size after optimization: 1  
2017-04-10 04:27:35,231 [main] INFO org.apache.hadoop.yarn.client.RMProxy - Connecting to ResourceManager at /0.0.0.0:8032  
2017-04-10 04:27:35,233 [main] INFO org.apache.pig.tools.pigstats.ScriptState - Pig script settings are added to the job  
2017-04-10 04:27:35,236 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.JobControlCompiler - mapred.job.reduce.markreset.buffer.percent is not set, set to default 0.0
```

Figure 5.13: Pig script of analysis 4

After running pig script a successful execution message will appear in the command shell.

Figure 5.14 illustrate the successful execution message.



```
edureka@localhost:~  
File Edit View Search Terminal Help  
2017-04-10 04:27:39,937 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - detailed locations: R: data[0,7],a[-1,-1],c[11,4],b[10,4] C: c[11,4],b[10,4] R: c[11,4]  
2017-04-10 04:27:39,962 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 0% complete  
2017-04-10 04:27:39,918 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 100% complete  
2017-04-10 04:27:39,407 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 100% complete  
2017-04-10 04:27:39,407 [main] INFO org.apache.pig.tools.pigstats.SimplePigStats - Script Statistics:  
HadoopVersion PigVersion UserId StartedAt FinishedAt Features  
2.2.0 0.12.1-SNAPSHOT edureka 2017-04-10 04:27:35 2017-04-10 04:27:39 GROUP BY  
Success!  
Job Stats (time in seconds):  
JobId Maps Reduces MapTime MinMapTime AvgMapTime MedianMapTime MaxReduceTime MinReduceTime AvgReduceTime MedianReduceTime Alias F  
e9Tarn OUTPUT3  
job_1491759183721_0007 1 1 3 3 3 3 3 3 3 3 a,b,c,data GROUP_BY,COMBINER /useCase4  
Input(s):  
Successfully read 49399 records (2942025 bytes) from: "/movie.txt"  
Output(s):  
Successfully stored 101 records (841 bytes) in: "/useCase4"  
Counters:  
Total records written: 101  
Total bytes written: 841  
Spillable Memory Manager spill count: 0  
Total bags proactively spilled: 0  
Total records proactively spilled: 0  
Job Info:  
job_1491759183721_0007
```

Figure 5.14: Successful execution of Pig script of analysis 4

Then we need check the output in the HDFS. Below figure show the output of the pig script of analysis 4 which contains all the list of year and number of movies each. Below figure show the output of the Pig script of the analysis 4.

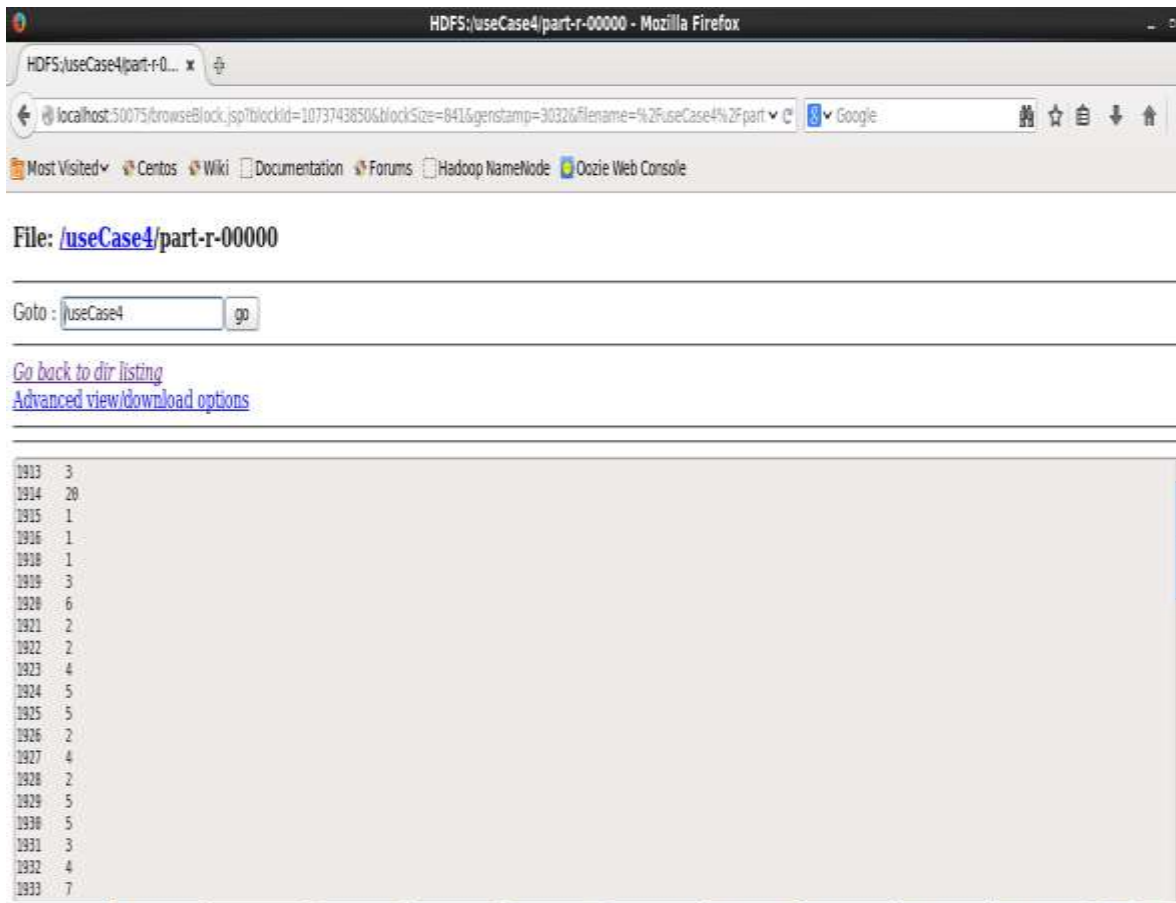


Figure 5.15: Output of the pig script of the analysis 4

In this chapter we show the data set description and analyze our test data set using Pig. We chose Pig over MapReduce because if we would chose MapReduce it would be so tough for us to get the desire result of each analysis. By using Pig it saves our development time. We also show four analysis and also show each analysis implementation with their screenshot.

CHAPTER 6

CONCLCUSION AND FUTURE SCOPE

6.1 Discussion and Conclusion

By the grace of almighty Allah we just completed our project and documentation. After the long session of studying about big data, Hadoop, Hadoop components and being comfortable of all the ecosystem of Hadoop, designing and last implementation we point that instead of completion. Completing a project on big data analytics was a big challenge for us and we complete our project successfully.

6.2 Scope for Future Development

In our project we use pseudo distributed mode in cluster set up. We will work in future will multi-mode cluster set up. We use a CSV file as our test data set, in future we will work with critical data set format like JSON format.

REFERENCES

[1] Deepthi Yaramala, “HEALTH CARE DATA ANALYTICS USING HADOOP”, <http://www-rohan.sdsu.edu/~eckberg/HadoopHealth/Health.html>, [Last Accessed on 3rd April 2017 at 9:05am].

[2] EY’s Big Data: changing the way business compete and operate, <http://www.ey.com/gl/en/services/advisory/ey-big-data-big-opportunities-big-challenges>, [Last Accessed on 23th January 2017 at 7:30 pm].

[3] Apache Pig-Architecture, https://www.tutorialspoint.com/apache_pig/apache_pig_architecture.htm, [Last Accessed on 10th February 2017 at 10:37 am].

[4] Sing, N., Garg, N. & Mittal, V. (2013). Big Data-insight, motivation and challenges. International Journal of Scientific & Engineering Research, 4(12).

[5] Introduction, https://en.wikipedia.org/wiki/Big_data, [Last Accessed on 17th February 2017 at 6:45 pm].

[6] Apache Hadoop, https://en.wikipedia.org/wiki/Apache_Hadoop#History, [Last Accessed on 7th March 2017 at 9:30am].

[7] Virtualbox download, <https://www.virtualbox.org/wiki/Downloads>, [Last Accessed on 30th January 2017 at 4:47 pm].

[8] Virtual Machine installation, https://www.cloudera.com/downloads/quickstart_vms/5-8.html, [Last Accessed on 26th January 2017 at 10:30pm].

[9] Alice Macfarlan, “Big Data and Evaluation – Use and Implication” http://www.betterevaluation.org/en/blog/big_data_in_evaluation, [Last Accessed on 29th March 2017 at 6:12 pm].

[10] Characteristics of Big Data, <https://www.edureka.co/blog/the-hype-behind-big-data/>, [Last Accessed on 07th February 2017 at 10:30 pm].

[4] Sing, N., Garg, N. & Mittal, V. (2013). Big Data-insight, motivation and challenges. International Journal of Scientific & Engineering Research, 4(12).

[12]Map Reduce, https://www.edureka.co/blog/apache-hadoop-hdfs-architecture/?utm_source=blog&utm_medium=left-menu&utm_campaign=hdfs-tutorial, [Last accessed on 01st May 2017 at 3:00pm]

[13] Pig, https://www.edureka.co/blog/pig-tutorial/?utm_source=blog&utm_medium=left-menu&utm_campaign=apache-hadoop-hdfs-architecture, [Last accessed on 15th March 2017 at 4:30]

APPENDICES

Appendix A: Project Reflection

The purpose of this appendix is to provide an introduction to **Project Reflection**. The group research was project was a challenging and enjoyable experience typical of the course as a whole. We have had little exposure to group work at university so it was a nice change to be part of an effective and dynamic team. I think we complemented one another quite well both in bringing together interdisciplinary perspectives and in balancing the work at hand.

The experience taught us that planning and crafting responses take a longer time in team than on your own. The extensive effort required was ultimately a good thing. When working alone, you can end up with a result that is identical to your initial plans. In our group, we were constantly developing and refining one another's ideas. It was fascinating just how productive our group meetings were. The time seemed to fly and yet we always got a lot done and managed to help another along the way towards the endpoint of having a substantive policy.

PLAGIARISM REPORT

