BIG DATA AND HADOOP

Project Report

Submitted by

RUTANSHU JHAVERI 15BCE2016

VENIKA ANAND 15BCE0998

UJJWAL KHANNA 15BCE0658

PAUL MATHAI 15BCE0981

RAHUL RAGHUNADHAN 15BCE0987

Course Code: **CSE4001**

Course Title: **Parallel and Distributed Computing**

Under the guidance of

Prof. MANOOV R
Associate Professor (Senior), SCSE

VIT University, Vellore-632014

Department of ….

School of Computer Science

And Engineering

NOVEMBER 2017

TABLE OF CONTENTS

ABSTRACT

The term ‘Big Data’ describes innovative techniques and technologies to capture, store, distribute, manage and analyze petabyte- or larger-sized datasets with high-speed and completely different structures. Big Data is structured, unstructured or semi-structured, leading to incapability of typical knowledge management ways. Data is generated from varied completely different sources and may arrive within the system at varied rates. So as to process these massive amounts of information in a cheap and economical means, parallelism is employed. Big Data could be a data whose scale, diversity, and quality need new design, techniques, algorithms, and analytics to manage it and extract price and hidden data from it. Hadoop is that the core platform for structuring Big Data, and solves the matter of constructing it helpful for analytics functions. Hadoop is Associate in nursing open supply package project that permits the distributed process of Big Datasets across clusters of artifact servers. It’s designed to rescale from one server to thousands of machines, with an awfully high degree of fault tolerance.

INTRODUCTION

Big Data could be a term that refers to information sets or mixtures of information sets whose size (volume), complexness (variability), and rate of growth (velocity) build them tough to be captured, managed, processed or analyzed by standard technologies and tools, like relative databases and desktop statistics or mental image packages, among the time necessary to form them helpful, whereas the dimensions accustomed confirm whether or not a selected information set is taken into account huge information isn't firmly outlined and continues to vary over time, most analysts and practitioners presently sit down with information sets from 30-50 terabytes to multiple petabytes as Big Data. It is rotten into 3 layers, as well as Infrastructure Layer, Computing Layer, and Application Layer from prime to bottom.

3 Vs of Big Data

Volume of data: Volume denotes to amount of data. Volume of information deposited in enterprise depositories have grown from megabytes and gigabytes to petabytes.

Variety of data: Different types of data and sources of data. Data variety exploded from structured and legacy information kept in enterprise repositories to unstructured, semi structured, audio, video, XML etc.

Velocity of data: Velocity denotes to the speed of data processing. For time-sensitive procedures such as catching fraud, big data must be used as it streams into your enterprise in order to enhance its value.

Hadoop

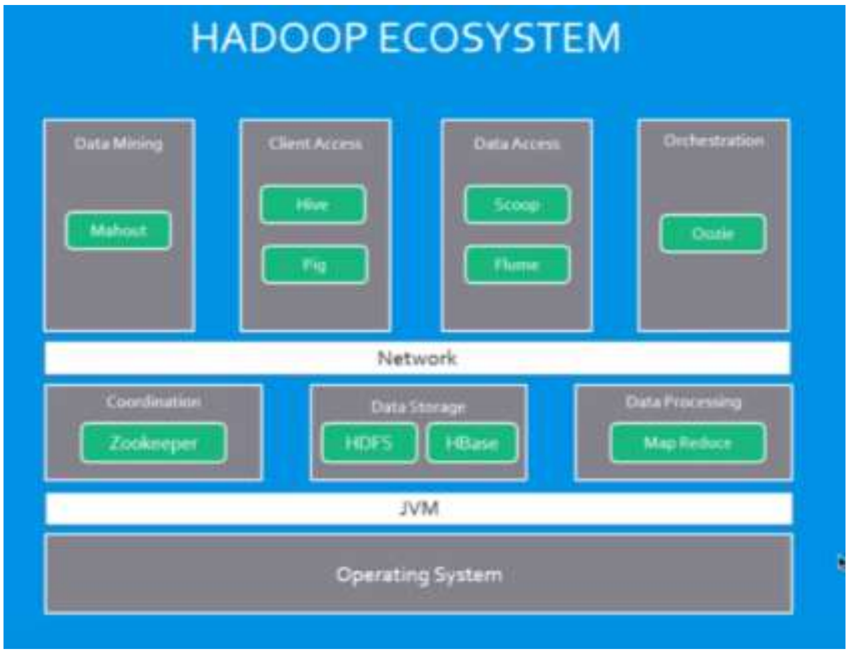
Hadoop is a Programming framework used to support the handling of vast data sets in a distributed computing environment. Hadoop was established by Google's MapReduce that is a software framework where an application breakdown into various parts. The Current Appache Hadoop ecosystem comprises of the Hadoop Kernel, MapReduce, HDFS & numbers of several components for eg. Apache Hive, Base and Zookeeper. HDFS and MapReduce are explained in following points.

PROJECT BACKGROUND

HADOOP ECOSYSTEM

The Hadoop system includes each official Apache open supply comes and a good vary of business tools and solutions. a number of the known open supply examples embrace Spark, Hive, Pig, Oozie and Sqoop. Industrial Hadoop offerings square measure even additional various and embrace platforms and prepacked distributions from vendors like Cloudera, Hortonworks, and MapR, and a range of tools for specific Hadoop development, production, and maintenance tasks.

Most of the solutions offered within the Hadoop system square measure meant to supplement one or 2 of Hadoop's four core components (HDFS, MapReduce, YARN, and Common). However, the commercially offered framework solutions give additional comprehensive practicality. The sections below give look into a number of the more distinguished elements of the Hadoop system, beginning with the Apache comes.



Apache open source Hadoop ecosystem elements

The Apache Hadoop project actively supports multiple comes supposed to increase Hadoop's capabilities and create it easier to use. There square measure many top-ranking comes to form development tools likewise as for managing Hadoop knowledge flow and process. Several industrial third-party solutions rest on the technologies developed inside the Apache Hadoop system.

Spark, Pig, and Hive square measure 3 of the known Apache Hadoop comes. Every employed to form applications to method Hadoop knowledge. whereas there square measure plenty of articles and discussions regarding whether or not Spark, Hive or Pig is healthier, in follow several organizations don't solely use one one as a result of every is optimized for specific functions.

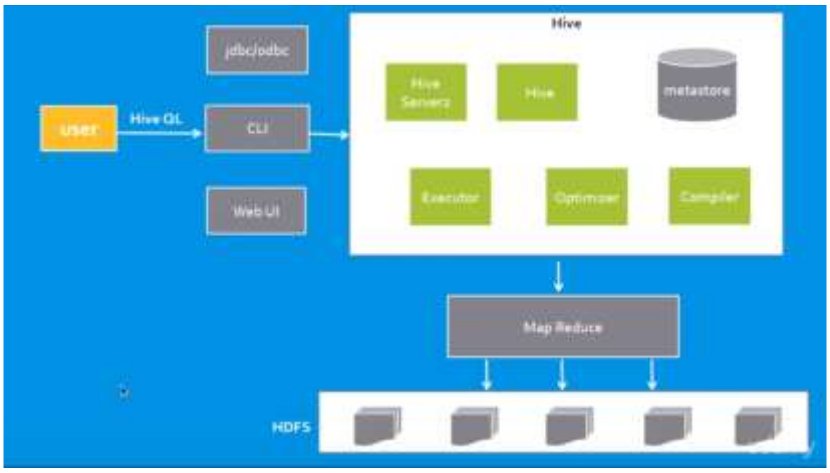
Spark

Spark is both a programming model and a computing model. It provides a gateway to in-memory computing for Hadoop, which is a big reason for its popularity and wide adoption. Spark provides an alternative to MapReduce that enables workloads to execute in memory, instead of on disk. Spark accesses data from HDFS but bypasses the MapReduce processing framework, and thus eliminates the resource-intensive disk operations that MapReduce requires. By using in-memory computing, Spark workloads typically run between 10 and 100 times faster compared to disk execution.

Spark can be used independently of Hadoop. However, it is used most commonly with Hadoop as an alternative to MapReduce for data processing. Spark can easily coexist with MapReduce and with other ecosystem components that perform other tasks.

Spark is also popular because it supports SQL, which helps overcome a shortcoming in core Hadoop technology. The Spark programming environment works interactively with Scala, Python, and R shells. It has been used for data extract/transform/load (ETL) operations, stream processing, machine learning development and with the Apache GraphX API for graph computation and display. Spark can run on a variety of Hadoop and non-Hadoop clusters, including Amazon S3.

Hive



Hive is information deposition software system that addresses however information is structured and queried in distributed Hadoop clusters. Hive is additionally a preferred development surroundings that's wont to write queries for information within the Hadoop surroundings. It provides tools for ETL operations and brings some SQL-like capabilities to the surroundings. Hive could be a declarative language that's wont to develop applications for the Hadoop surroundings, but it doesn't support time period queries.

- ❖ HCatalog – Helps processing tools scan and write information on the grid. It supports MapReduce and Pig.
- ❖ WebHCat – allows you to use associate degree HTTP/REST interface to run MapReduce, Yarn, Pig, and Hive jobs.
- ❖ HiveQL – Hive's source language meant as some way for SQL developers to simply add Hadoop. it's like SQL and helps each structure and question information in distributed Hadoop

Hive also allows MapReduce-compatible mapping and reduction software to perform more sophisticated functions. However, Hive does not allow row-level updates or support for real-time queries, and it is not intended for OLTP workloads. Many consider Hive to be much more effective for processing structured data than unstructured data, for which Pig is considered advantageous.

Pig

Pig may be a procedural language for developing data processing applications for giant knowledge sets within the Hadoop surroundings. Pig is another to Java programming for MapReduce, and mechanically generates MapReduce functions. Pig includes Pig Latin, that may be a scripting language. Pig interprets Pig Latin scripts into MapReduce, which might then run on YARN and method knowledge within the HDFS cluster. Pig is fashionable as a result of it automates a number of the quality in MapReduce development.

Pig is often used for complicated use cases that need multiple knowledge operations. it's a lot of of a process language than a question language. Pig helps develop applications that mixture and type knowledge and supports multiple inputs and exports. it's extremely customizable, as a result of users will write their own functions victimisation their most well-liked scripting language. Ruby, Python and even Java square measure all supported. Thus, Pig has been a preferred possibility for developers that square measure aware of those languages however not with MapReduce. However, SQL developers might realize Hive easier to be told.

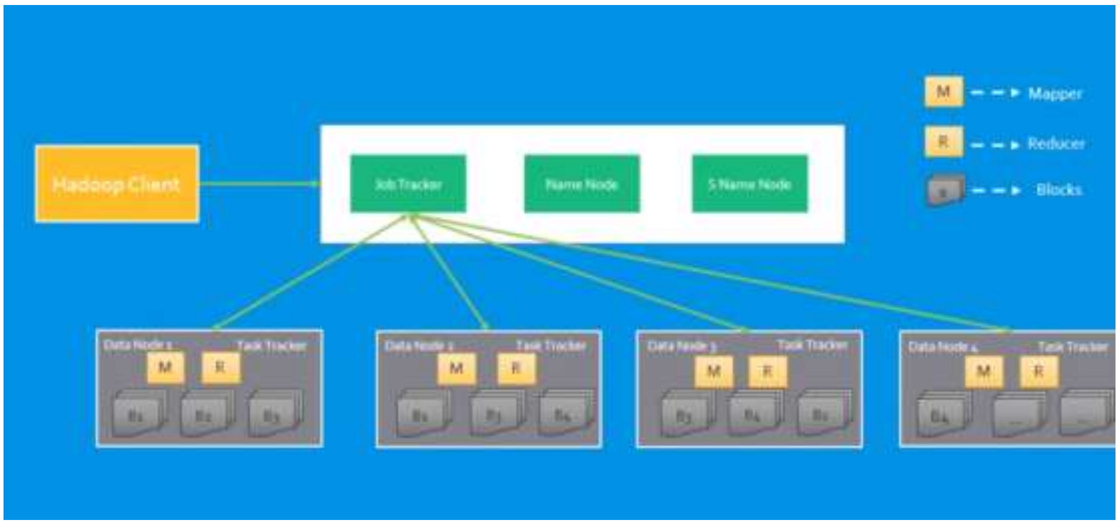
HBase

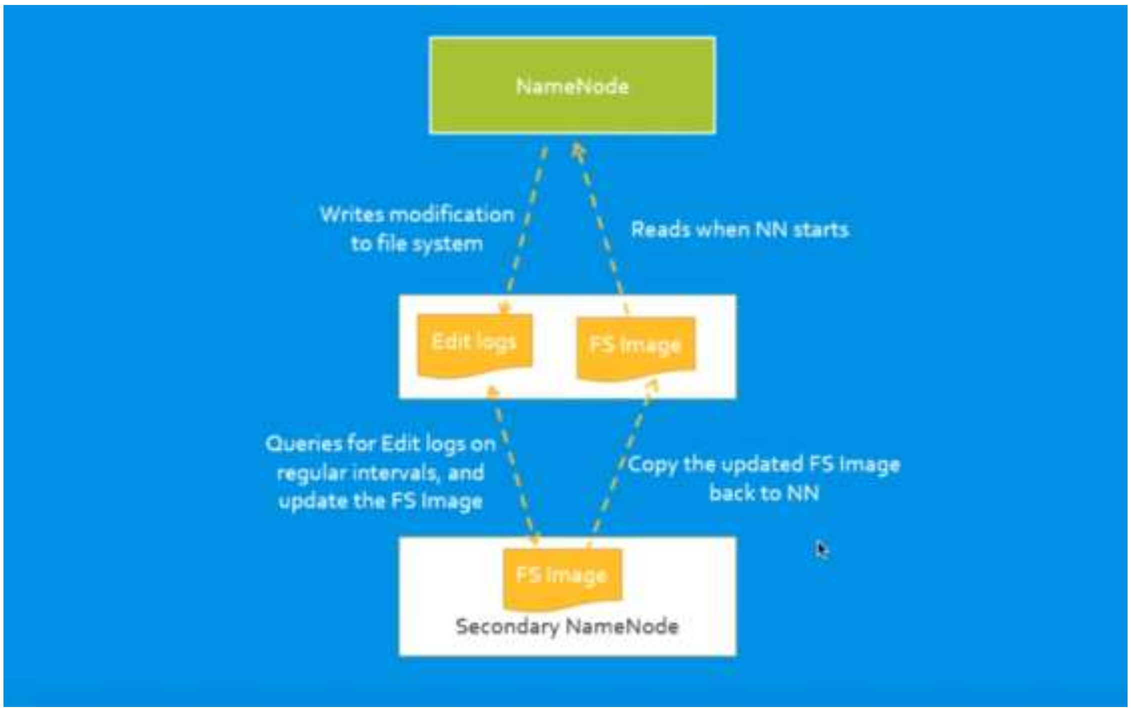
It absolutely was designed to store structured information in tables that would have billions of rows and countless columns. it's been deployed to power historical searches through giant information sets, particularly once the specified information is contained at intervals an oversized quantity of unimportant or orthogonal information (also referred to as thin information sets). it's additionally associate degree underlying technology behind many giant electronic communication applications, as well as Facebook's.

HBase isn't a on-line database and wasn't designed to support transactional and different period applications. it's accessible through a Java API and has ODBC and JDBC drivers. HBase doesn't support SQL queries, but there area unit many SQL support tools out there from the Apache project and from code vendors. as an example, Hive may be wont to run SQL-like queries in HBase.

Oozie

Oozie is that the work flow computer hardware that was developed as a part of the Apache Hadoop project. It manages however workflows begin and execute, and additionally controls the execution path. Oozie may be a server-based Java internet application that uses work flow definitions written in hPDL, that is associate degree XML method Definition Language like JBOSS JBPM jPDL. Oozie solely supports specific work flow varieties, therefore different work schedulers area unit unremarkably used rather than or additionally to Oozie in Hadoop environments.


Hadoop Client
Job Tracker
Name Node
S Name Node
M Mapper
R Reducer
Blocks
Data Node 1
Task Tracker
M
R
B1
B2
B3
Data Node 2
Task Tracker
M
R
B1
B3
B4
Data Node 3
Task Tracker
M
R
B3
B4
B1
Data Node 4
Task Tracker
M
R
B4
...
...


NameNode
Writes modification to file system
Reads when NN starts
Edit logs
FS Image
Queries for Edit logs on regular intervals, and update the FS Image
Copy the updated FS Image back to NN
FS Image
Secondary NameNode

HADOOP 1.X

- Node limitations(supports up to 4,000 node/cluster)
- Job tracker bottleneck
- Only has one NameNode for managing HDFS
- Map and Reduce slots are static
- Only job to run is MapReduce

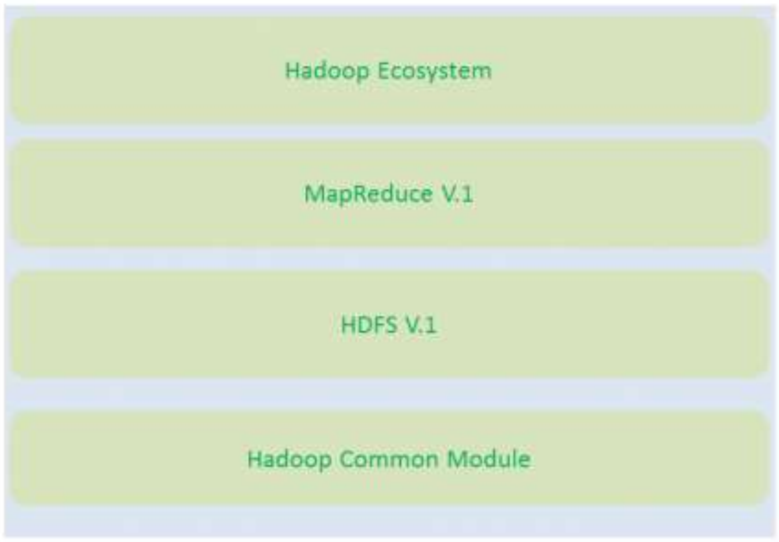
Hadoop 1.x Architecture

Apache Hadoop one.x or earlier versions area unit exploitation the subsequent Hadoop design. It is a Hadoop one.x High-level design. We are going to discuss in-detailed Low-level design in returning sections.

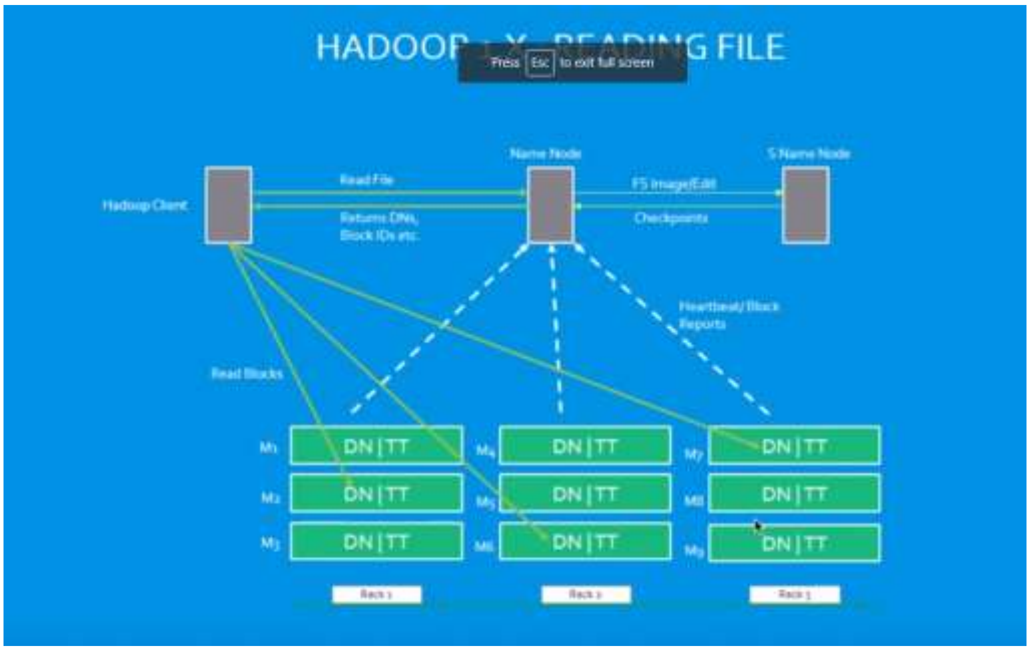
- Hadoop Common Module could be a Hadoop Base API (A Jar file) for all Hadoop elements. All different elements works on high of this module.

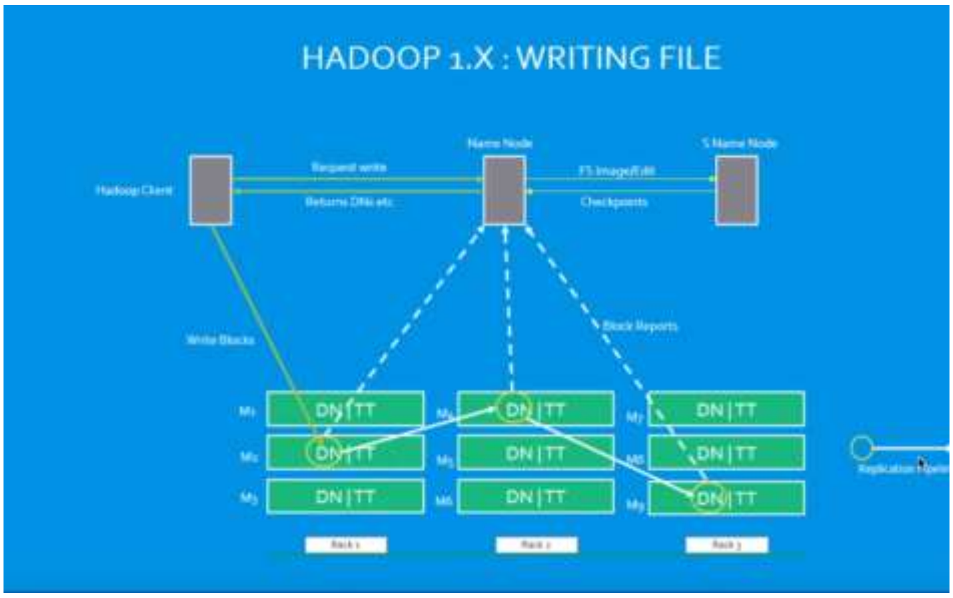
- HDFS stands for Hadoop Distributed filing system. It's additionally understand as HDFS V1 because it is a component of Hadoop one.x. It's used as a Distributed Storage System in Hadoop design.

- MapReduce could be a execution or Distributed processing Module. It's designed by following Google's MapReduce algorithmic program. It's additionally understand as "MR V1" or "Classic MapReduce" because it is a component of Hadoop one.x. Remaining all Hadoop scheme elements work on high of those 2 major components: HDFS and MapReduce. We are going to discuss all Hadoop scheme elements in-detail in my returning posts.



Hadoop V.1.x Components


HADOOP 1.X : READING FILE
Press Esc to exit full screen
Name Node
S Name Node
Hadoop Client
Read File
Returns DNs, Block IDs etc.
FS Image/Edit
Checkpoints
Heartbeat/ Block Reports
Read Blocks
M1
M2
M3
M4
M5
M6
M7
M8
M9
DN | TT
Rack 1
Rack 2
Rack 3

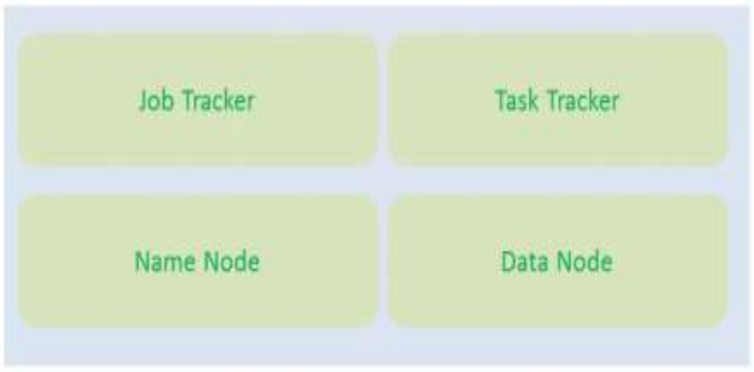

HADOOP 1.X : WRITING FILE
Name Node
S Name Node
Hadoop Client
Request write
Returns DNs etc.
FS Image/Edit
Checkpoints
Block Reports
Write Blocks
M1
M2
M3
M4
M5
M6
M7
M8
M9
DN | TT
Rack 1
Rack 2
Rack 3
Replication Pipeline

Hadoop 1.x - Major Components

Hadoop 1.x Major parts parts are: HDFS and MapReduce. They're additionally apprehend as "Two Pillars" of Hadoop one.x.

HDFS:

- HDFS may be a Hadoop Distributed FileSystem, wherever our BigData is hold on victimisation goods Hardware. it's designed to figure with giant DataSets with default block size is 64MB (We will amendment it as per our Project requirements).
- HDFS element is once more divided into 2 sub-components:
- Name Node
- Name Node is placed in Master Node. It accustomed store Meta information regarding information Nodes like "How several blocks area unit hold on in information Nodes, that information Nodes have information, Slave Node Details, information Nodes locations, timestamps etc" .
- Data Node
- Data Nodes area unit places in Slave Nodes. It's accustomed store our Application Actual information. It stores information in information Slots of size 64MB by default.



HDFS and MR Components

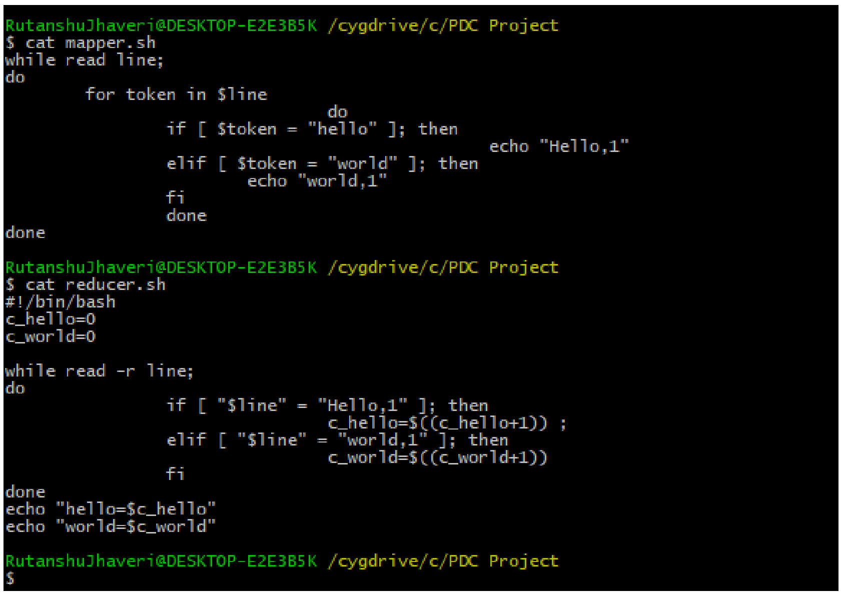
MapReduce:

MapReduce may be a Distributed processing or execution Programming Model. Like HDFS, MapReduce part additionally uses

- Commodity Hardware to method "High Volume of form of knowledge at High rate Rate" in a very reliable and fault-tolerant manner.
- MapReduce part is once more divided into 2 sub-components:
- Job hunter

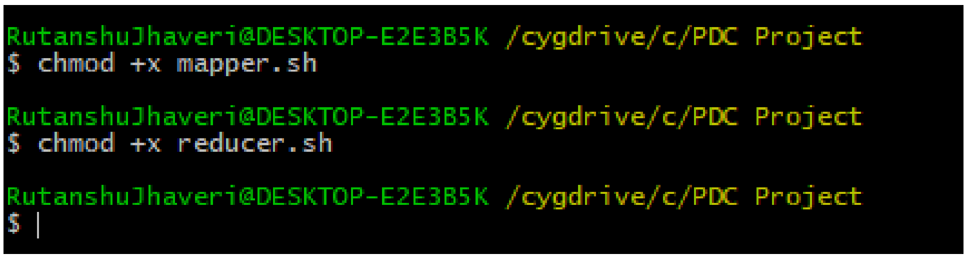
  Job hunter is employed to assign MapReduce Tasks to Task Trackers within the Cluster of Nodes. Occasionally, it reassigns same tasks to alternative Task Trackers as previous Task Trackers ar failing or closedown eventualities. Job hunter maintains all the Task Trackers standing like Up/running, Failed, Recovered etc.
- Task hunter
- Task hunter executes the Tasks that are assigned by Job hunter and sends the standing of these tasks to Job hunter.
- We can deliberate these four sub-component's responsibilities and the way they move one another to perform a "Client Application Tasks" intimately in next section.
- How Hadoop one.x Major elements Works
- Hadoop 1.x elements follow this design to move one another and to figure parallel in a very dependable and fault-tolerant manner.

Implementation Example:-



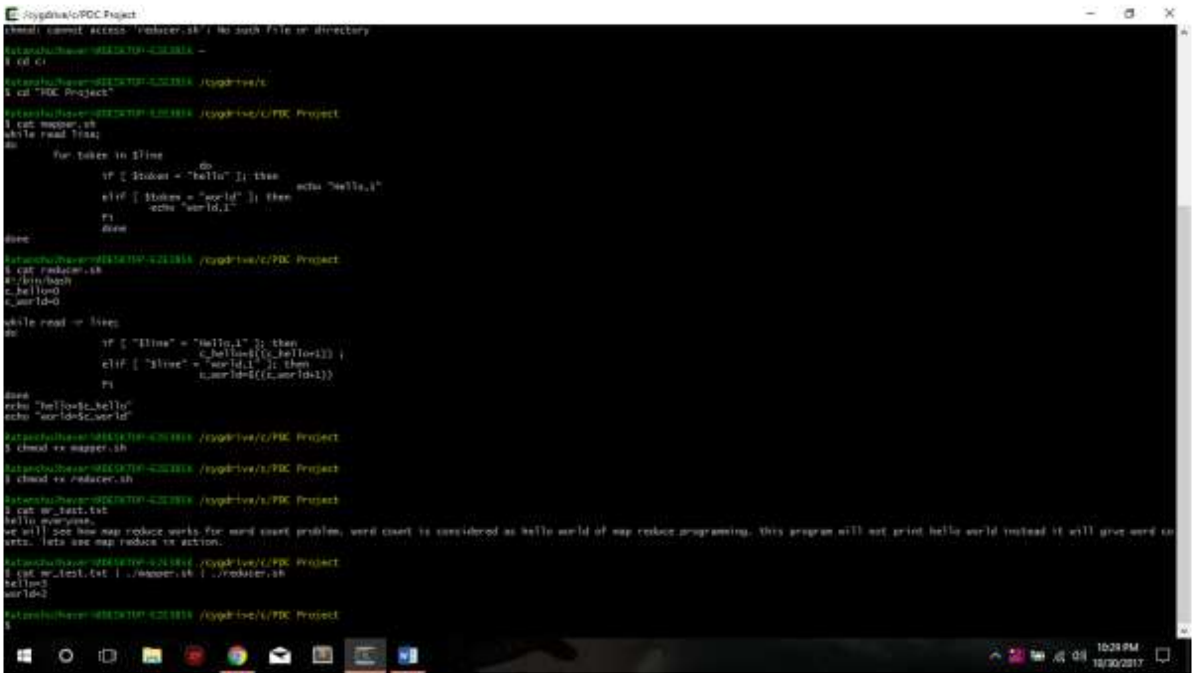
```
RutanshuJhaveri@DESKTOP-E2E3B5K /cygdrive/c/PDC Project
$ cat mapper.sh
while read line;
do
        for token in $line
                                do
                if [ $token = "hello" ]; then
                                                echo "Hello,1"
                elif [ $token = "world" ]; then
                        echo "world,1"
                fi
                done
done

RutanshuJhaveri@DESKTOP-E2E3B5K /cygdrive/c/PDC Project
$ cat reducer.sh
#!/bin/bash
c_hello=0
c_world=0

while read -r line;
do
                if [ "$line" = "Hello,1" ]; then
                                c_hello=$((c_hello+1)) ;
                elif [ "$line" = "world,1" ]; then
                                c_world=$((c_world+1))
                fi
done
echo "hello=$c_hello"
echo "world=$c_world"

RutanshuJhaveri@DESKTOP-E2E3B5K /cygdrive/c/PDC Project
$
```

Mapper and reducer shell scripts to count the words "hello" and "world"



```
RutanshuJhaveri@DESKTOP-E2E3B5K /cygdrive/c/PDC Project
$ chmod +x mapper.sh

RutanshuJhaveri@DESKTOP-E2E3B5K /cygdrive/c/PDC Project
$ chmod +x reducer.sh

RutanshuJhaveri@DESKTOP-E2E3B5K /cygdrive/c/PDC Project
$ |
```

Giving permission to file of shell scripts



```
chmod: cannot access 'reducer.sh': No such file or directory

RutanshuJhaveri@DESKTOP-E2E3B5K ~
$ cd c:

RutanshuJhaveri@DESKTOP-E2E3B5K /cygdrive/c
$ cd "PDC Project"

RutanshuJhaveri@DESKTOP-E2E3B5K /cygdrive/c/PDC Project
$ cat mapper.sh
while read line;
do
        for token in $line
                                do
                if [ $token = "hello" ]; then
                                                echo "hello,1"
                elif [ $token = "world" ]; then
                        echo "world,1"
                fi
                done
done

RutanshuJhaveri@DESKTOP-E2E3B5K /cygdrive/c/PDC Project
$ cat reducer.sh
#!/bin/bash
c_hello=0
c_world=0

while read -r line;
do
                if [ "$line" = "hello,1" ]; then
                                c_hello=$((c_hello+1)) ;
                elif [ "$line" = "world,1" ]; then
                                c_world=$((c_world+1))
                fi
done
echo "hello=$c_hello"
echo "world=$c_world"

RutanshuJhaveri@DESKTOP-E2E3B5K /cygdrive/c/PDC Project
$ chmod +x mapper.sh

RutanshuJhaveri@DESKTOP-E2E3B5K /cygdrive/c/PDC Project
$ chmod +x reducer.sh

RutanshuJhaveri@DESKTOP-E2E3B5K /cygdrive/c/PDC Project
$ cat mr_test.txt
hello everyone,
we will see how map reduce works for word count problem. word count is considered as hello world of map reduce programming. this program will not print hello world instead it will give word co
unts. lets see map reduce in action.

RutanshuJhaveri@DESKTOP-E2E3B5K /cygdrive/c/PDC Project
$ cat mr_test.txt | ./mapper.sh | ./reducer.sh
hello=3
world=2

RutanshuJhaveri@DESKTOP-E2E3B5K /cygdrive/c/PDC Project
$
```

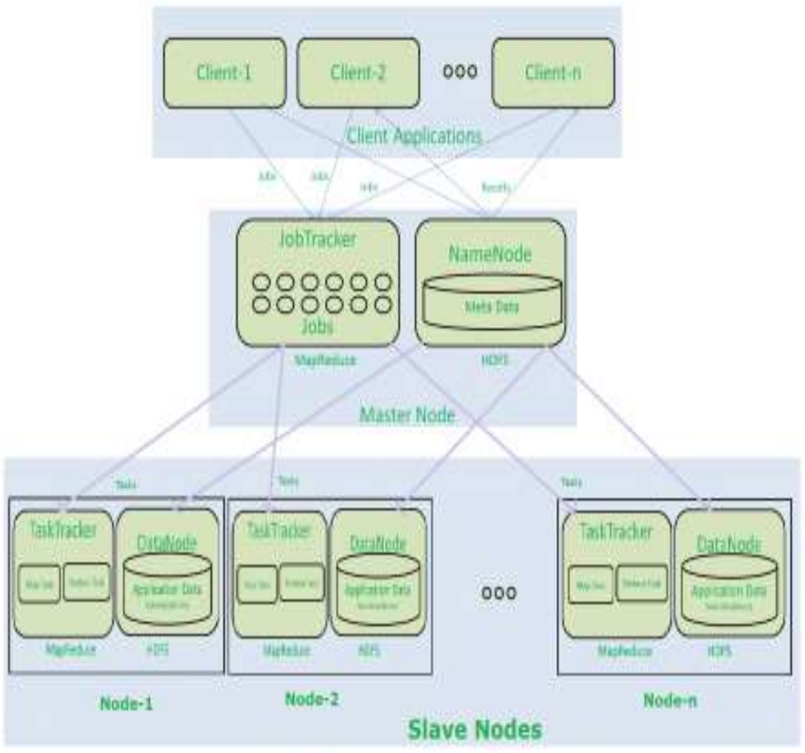
<u>Project is on Virtual Box</u>

We can show the code pig and hive files.

Therefore, there is no directory to show, The final project would be shown in video of Review 3.

- https://github.com/rutanshuj/PDCProject

Hadoop 1.x - Components In-detail Architecture



Hadoop 1.x Architecture

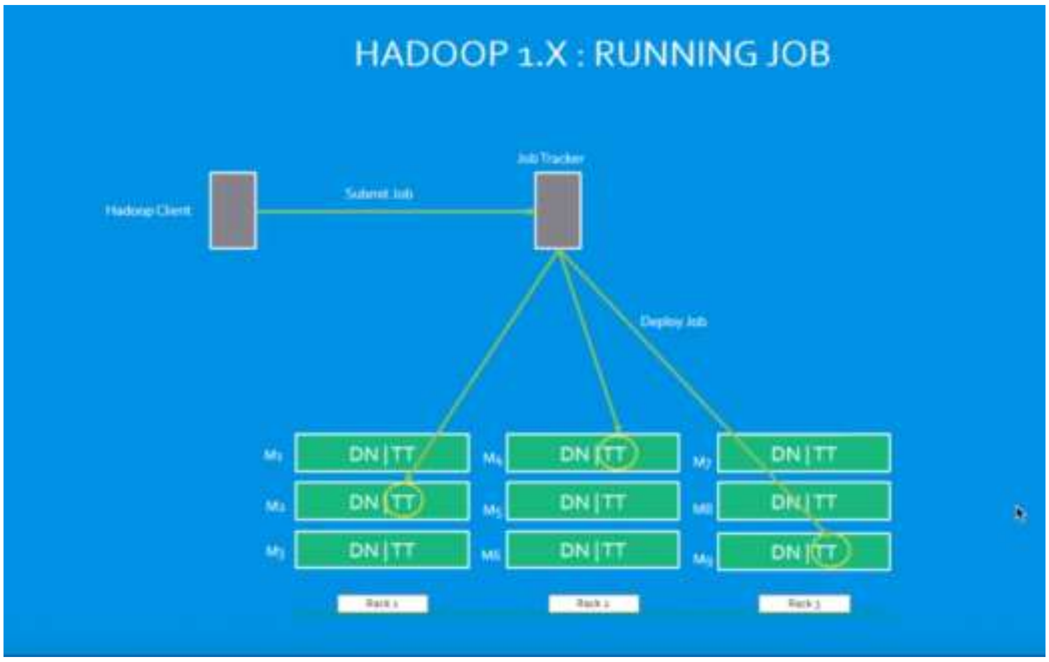
<u>Hadoop 1.x Architecture Description</u>

- Clients submit their work to Hadoop System.
- When Hadoop System receives a consumer Request, initial it's received by a Master Node.
- Master Node's MapReduce half "Job Tracker" is in charge of receiving shopper Work and divides into manageable freelance Tasks and assign them to Task Trackers.
- Slave Node's MapReduce half "Task Tracker" receives those Tasks from "Job Tracker" and perform those tasks by exploitation MapReduce components.
- Once all Task Trackers finished their job, Job hunter takes those results and combines them into effect.
- Finally Hadoop System will send that effect to the consumer.
- How Store and figure Operations add Hadoop
- All these Master Node and Slave Nodes square measure organized into a Network of clusters. each Cluster is over again divided into Racks. each rack contains a bunch of Nodes (Commodity Computer).

- When Hadoop system receives “Store” operation like storing large DataSets into HDFS, it stores that data into 3 altogether completely different Nodes (As we've got a bent to piece Replication issue = 3 by default). this whole data is not hold on in one single node. large record is split into manageable and purposeful Blocks and distributed into altogether completely different nodes with 3 copies.
- If Hadoop system receives any “Compute” operation, it's going to sit down with near-by nodes to retrieve those blocks of information. whereas Reading data or Computing if one or further nodes get failing, then it's going to automatically pick-up performing those tasks by approaching any near-by and on the market node.
- That’s why Hadoop system provides very on the market and fault tolerant Big Data Solutions.

NOTE:-

- ❖ Hadoop 1.x Architecture has lot of limitations and drawbacks. So that Hadoop Community has evaluated and redesigned this Architecture into Hadoop 2.x Architecture.
- ❖ Hadoop 2.x Architecture is completely different and resolved all Hadoop 1.x Architecture’s limitations and drawbacks.



HADOOP 2.X

Introduction

Apache Hadoop 2.0 represents a generational shift in the architecture of Apache Hadoop. With YARN, Apache Hadoop is recast as a significantly more powerful platform – one that takes Hadoop beyond merely batch applications to taking its position as a 'data operating system' where HDFS is the file system and YARN is the operating system.

YARN is a re-architecture of Hadoop that allows multiple applications to run on the same platform. With YARN, applications run "in" Hadoop, instead of "on" Hadoop. The fundamental idea of YARN is to split up the two major responsibilities of the JobTracker and TaskTracker into separate entities

- Support up to to 10,000 node per cluster
- Supports multiple NameNodes for managing HDFS
- Introduction of YARN for efficient cluster utilization.
- MRv1 Backward and forward compatible
- Any apps can be integrated with Hadoop
- Beyond MapReduce

Hadoop 2.x Architecture

- Apache Hadoop two.x or later versions ar victimization the subsequent Hadoop design. it's a Hadoop two.x High-level design. we'll discuss in-detailed Low-level design in returning sections.
- Hadoop Common Module could be a Hadoop Base API (A Jar file) for all Hadoop parts. All alternative parts works on high of this module.
- HDFS stands for Hadoop Distributed filing system. it's additionally apprehend as HDFS V2 because it is a component of Hadoop two.x with some increased options. it's used as a Distributed Storage System in Hadoop design.
- YARN stands for yet one more Resource communicator. it's new part in Hadoop two.x design. it's additionally apprehend as "MR V2".

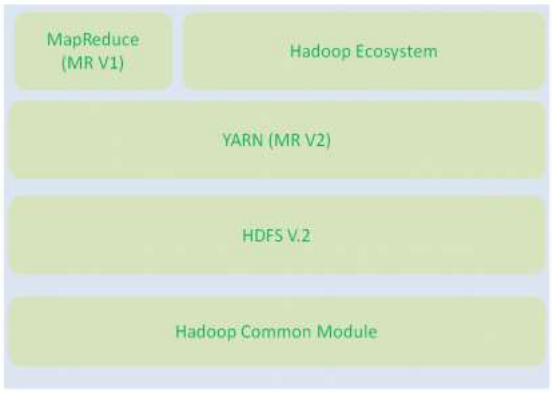
- MapReduce could be a instruction execution or Distributed processing Module. it's additionally apprehend as “MR V1” because it is a component of Hadoop one.x with some updated options.
- Remaining all Hadoop scheme parts work on high of those 3 major components: HDFS, YARN and MapReduce. we'll discuss all Hadoop scheme parts in-detail in my returning posts.

When compared to Hadoop one.x, Hadoop 2.x design is intended fully completely different. It’s supplementary one new component: YARN and additionally updated HDFS and MapReduce component’s Responsibilities.When compared to Hadoop 1.x, Hadoop 2.x Architecture is designed completely different. It has added one new component: YARN and also updated HDFS and MapReduce component’s Responsibilities.

Hadoop 2.x Major Components

Hadoop 2.x has the subsequent 3 Major Components:
• HDFS
• YARN
• MapReduce
These 3 also are called 3 Pillars of Hadoop a pair of. Here key element amendment is YARN. it's very game dynamicalelement in BigData Hadoop System.

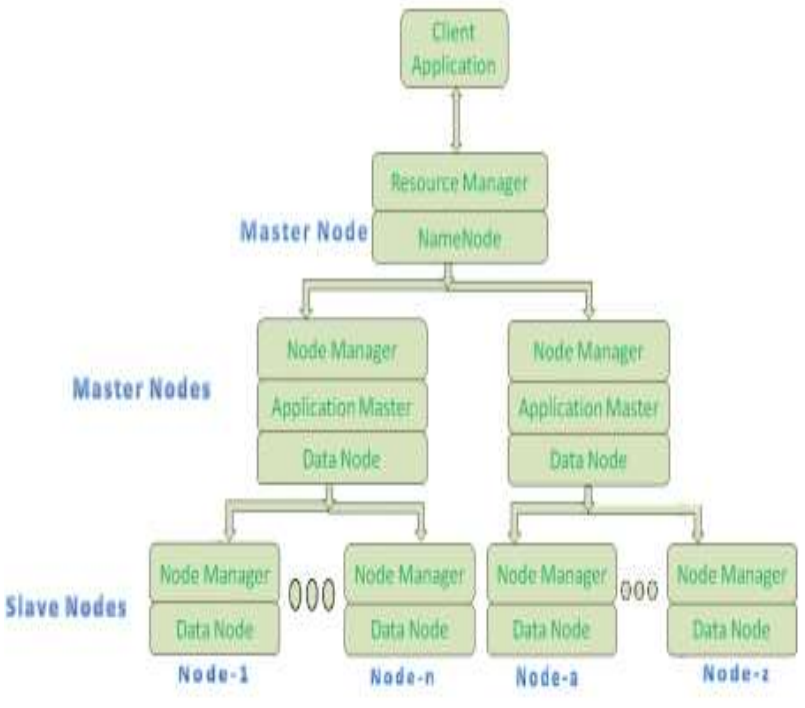
How Hadoop a pair of.x Major elements Works



Hadoop V.2.x Components

Hadoop 2.x elements follow this design to move one another and to figure parallel in an exceedingly reliable, extremely obtainable and fault-tolerant manner.

Hadoop 2.x Components High-Level Architecture



Hadoop 2.x High-Level Architecture

All Master Nodes and Slave Nodes contains each MapReduce and HDFS elements.

- One Master Node has 2 components:
- Resource Manager(YARN or MapReduce v2)
- HDFS

HDFS element is additionally is aware of as NameNode. Its NameNode is employed to store Meta information. In Hadoop a pair of. X, some additional Nodes acts as Master Nodes as shown within the higher than diagram. Every this second level Master Node has three components:

- ❖ Node Manager
- ❖ Application Master
- ❖ Data Node

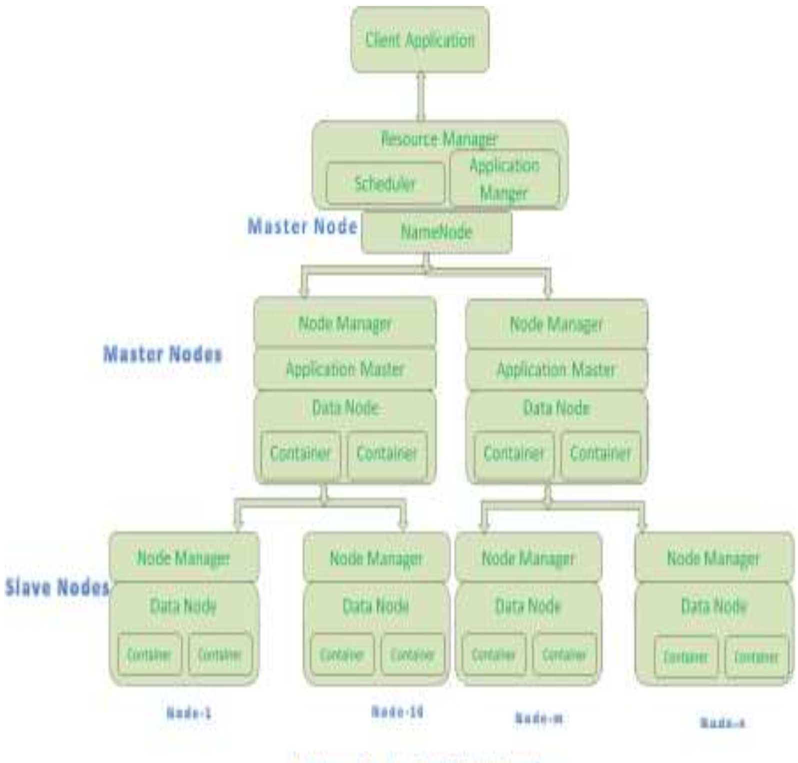
Each this second level Master Node {again|once additional} contains one or more Slave Nodes as shown within the higher than diagram.

These Slave Nodes have 2 components:

1. Node Manager
2. HDFS

HDFS element is additionally is aware of as information Node. Its information Node element is employed to store actual our application huge information. These nodes doesn't contain Application Master Element.

Hadoop 2.x Components In-detail Architecture



Hadoop 2.x In-Detail Architecture

Hadoop 2.x Architecture Description

Resource Manager:

- Resource Manager could be a Per-Cluster Level part.
- Resource Manager is once more divided into 2 components:

1. Scheduler
2. Application Manager

❖ Resource Manager's hardware is :

1. Responsible to schedule needed resources to Applications (that is Per-Application Master).
2. It will solely programming.
3. It will care regarding watching or following of these Applications.

Application Master:

- Application Master could be a per-application level part. it's accountable for:
- Managing allotted Application Life cycle.
- It interacts with each Resource Manager's hardware and Node Manager
- It interacts with hardware to accumulate needed resources.
- It interacts with Node Manager to execute allotted tasks and monitor those task's standing.

Node Manager:

- Node Manager could be a Per-Node Level part.

It is accountable for:

- Managing the life-cycle of the instrumentality.
- Monitoring every Container's Resources utilization.

Container:

Each Master Node or Slave Node contains set of Containers. During this diagram, Main Node's Name Node isn't showing the Containers. However, it conjointly contains a collection of Containers.

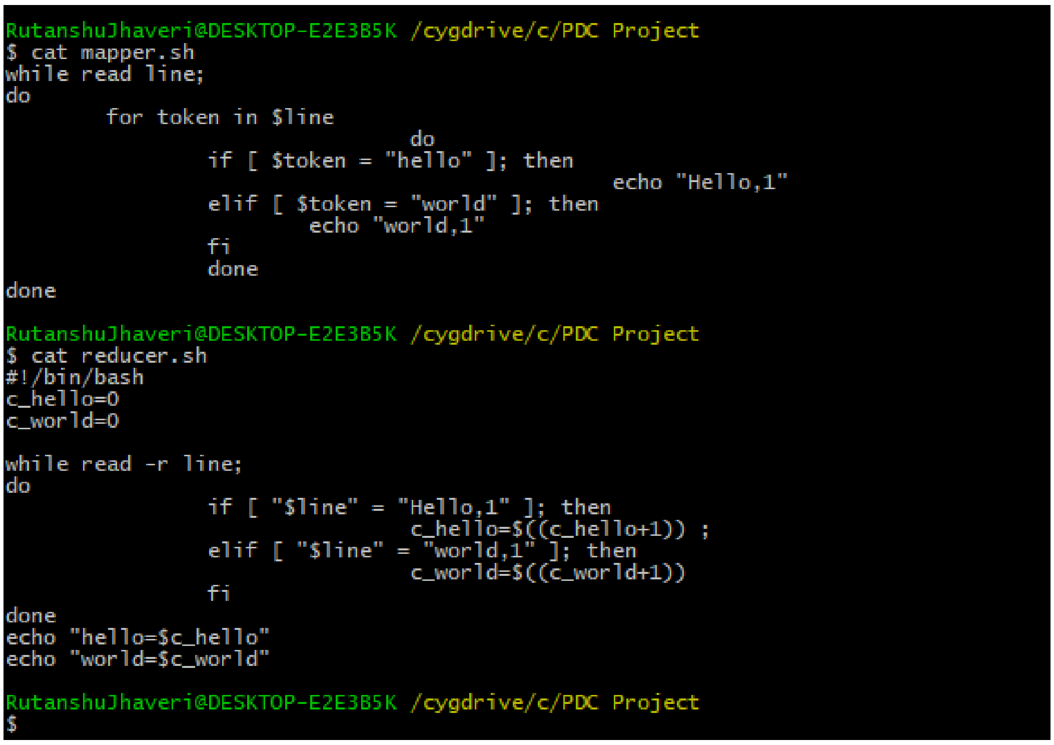
Container could be a portion of Memory in HDFS (Either Name Node or information Node).

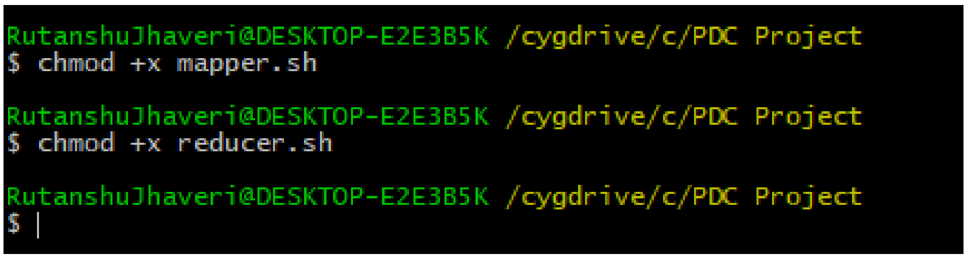
In Hadoop two.x, instrumentality is comparable to information Slots in Hadoop one.x. We'll see the foremost variations between these 2 Components: Slots Vs Containers in my returning posts.

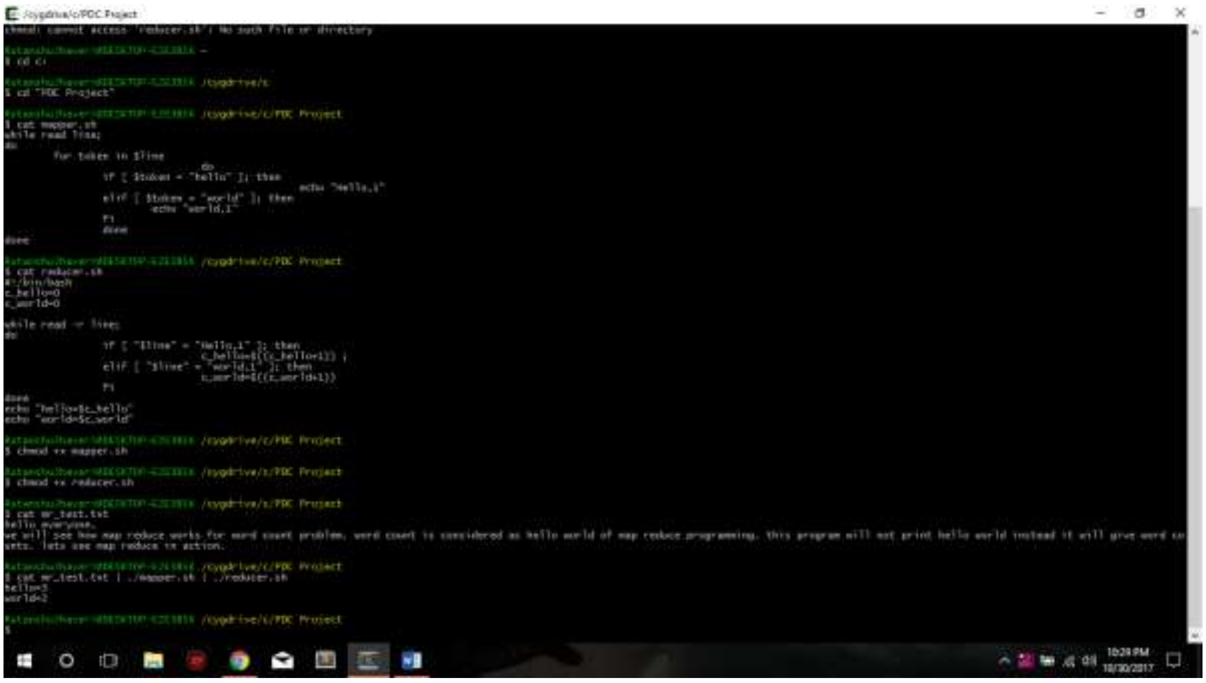
NOTE:-

- Resource Manager is Per-Cluster element where as Application Master is per-application component.
- Both Hadoop 1.x and Hadoop 2.x Architectures follow Master-Slave Architecture Model.

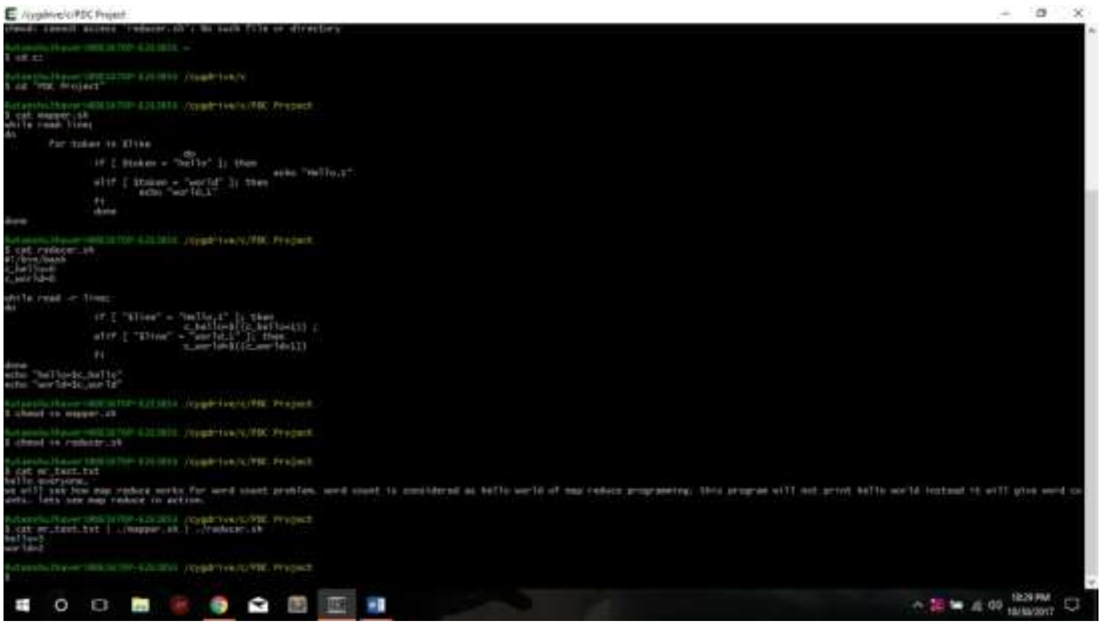
Implementation Example:-


```
RutanshuJhaveri@DESKTOP-E2E3B5K /cygdrive/c/PDC Project
$ cat mapper.sh
while read line;
do
        for token in $line
                                do
                if [ $token = "hello" ]; then
                                                echo "Hello,1"
                elif [ $token = "world" ]; then
                        echo "world,1"
                fi
                done
done

RutanshuJhaveri@DESKTOP-E2E3B5K /cygdrive/c/PDC Project
$ cat reducer.sh
#!/bin/bash
c_hello=0
c_world=0

while read -r line;
do
                if [ "$line" = "Hello,1" ]; then
                                c_hello=$((c_hello+1)) ;
                elif [ "$line" = "world,1" ]; then
                                c_world=$((c_world+1))
                fi
done
echo "hello=$c_hello"
echo "world=$c_world"

RutanshuJhaveri@DESKTOP-E2E3B5K /cygdrive/c/PDC Project
$
```
Mapper and reducer shell scripts to count the words “hello” and “world”


```
RutanshuJhaveri@DESKTOP-E2E3B5K /cygdrive/c/PDC Project
$ chmod +x mapper.sh

RutanshuJhaveri@DESKTOP-E2E3B5K /cygdrive/c/PDC Project
$ chmod +x reducer.sh

RutanshuJhaveri@DESKTOP-E2E3B5K /cygdrive/c/PDC Project
$ |
```
Giving permission to file of shell scripts



```
chmod: cannot access 'reducer.sh': No such file or directory

RutanshuJhaveri@DESKTOP-[illegible] ~
$ cd c:

RutanshuJhaveri@DESKTOP-[illegible] /cygdrive/c
$ cd "PDC Project"

RutanshuJhaveri@DESKTOP-[illegible] /cygdrive/c/PDC Project
$ cat mapper.sh
while read line;
do
        for token in $line
                                do
                if [ $token = "hello" ]; then
                                                echo "hello,1"
                elif [ $token = "world" ]; then
                        echo "world,1"
                fi
                done
done

RutanshuJhaveri@DESKTOP-[illegible] /cygdrive/c/PDC Project
$ cat reducer.sh
#!/bin/bash
c_hello=0
c_world=0

while read -r line;
do
                if [ "$line" = "hello,1" ]; then
                                c_hello=$((c_hello+1)) ;
                elif [ "$line" = "world,1" ]; then
                                c_world=$((c_world+1))
                fi
done
echo "hello=$c_hello"
echo "world=$c_world"

RutanshuJhaveri@DESKTOP-[illegible] /cygdrive/c/PDC Project
$ chmod +x mapper.sh

RutanshuJhaveri@DESKTOP-[illegible] /cygdrive/c/PDC Project
$ chmod +x reducer.sh

RutanshuJhaveri@DESKTOP-[illegible] /cygdrive/c/PDC Project
$ cat mr_test.txt
hello everyone,
we will see how map reduce works for word count problem, word count is considered as hello world of map reduce programming. this program will not print hello world instead it will give word co
unts. lets see map reduce in action.

RutanshuJhaveri@DESKTOP-[illegible] /cygdrive/c/PDC Project
$ cat mr_test.txt | ./mapper.sh | ./reducer.sh
hello=3
world=2

RutanshuJhaveri@DESKTOP-[illegible] /cygdrive/c/PDC Project
$
```



```
chmod: cannot access 'reducer.sh': No such file or directory

RutanshuJhaveri@DESKTOP-[illegible] ~
$ cd c:

RutanshuJhaveri@DESKTOP-[illegible] /cygdrive/c
$ cd "PDC Project"

RutanshuJhaveri@DESKTOP-[illegible] /cygdrive/c/PDC Project
$ cat mapper.sh
while read line;
do
        for token in $line
                                do
                if [ $token = "hello" ]; then
                                                echo "hello,1"
                elif [ $token = "world" ]; then
                        echo "world,1"
                fi
                done
done

RutanshuJhaveri@DESKTOP-[illegible] /cygdrive/c/PDC Project
$ cat reducer.sh
#!/bin/bash
c_hello=0
c_world=0

while read -r line;
do
                if [ "$line" = "hello,1" ]; then
                                c_hello=$((c_hello+1)) ;
                elif [ "$line" = "world,1" ]; then
                                c_world=$((c_world+1))
                fi
done
echo "hello=$c_hello"
echo "world=$c_world"

RutanshuJhaveri@DESKTOP-[illegible] /cygdrive/c/PDC Project
$ chmod +x mapper.sh

RutanshuJhaveri@DESKTOP-[illegible] /cygdrive/c/PDC Project
$ chmod +x reducer.sh

RutanshuJhaveri@DESKTOP-[illegible] /cygdrive/c/PDC Project
$ cat mr_test.txt
hello everyone,
we will see how map reduce works for word count problem, word count is considered as hello world of map reduce programming. this program will not print hello world instead it will give word co
unts. lets see map reduce in action.

RutanshuJhaveri@DESKTOP-[illegible] /cygdrive/c/PDC Project
$ cat mr_test.txt | ./mapper.sh | ./reducer.sh
hello=3
world=2

RutanshuJhaveri@DESKTOP-[illegible] /cygdrive/c/PDC Project
$
```

Project is on Virtual Box

We can show the code pig and hive files.

Therefore, there is no directory to show, The final project would be shown in video of Review 3.

https://github.com/rutanshuj/PDCProject

HDFS File Read Workflow

Now let’s perceive complete finish to finish HDFS information scan operation. As shown within the higher than figure the info scan operation in HDFS is distributed, the shopper reads the info parallelly from datanodes, the steps by step clarification of information scan cycle is:

i) Shopper opens the file it desires to scan by vocation open () on the FileSystem object, that for HDFS is AN instance of DistributedFileSystem.

ii) DistributedFileSystem calls the namenode victimization RPC to work out the locations of the blocks for the primary few blocks within the file. for every block, the namenode returns the addresses of the datanodes that have a duplicate of that block and datanode square measure sorted in line with their proximity to the shopper.

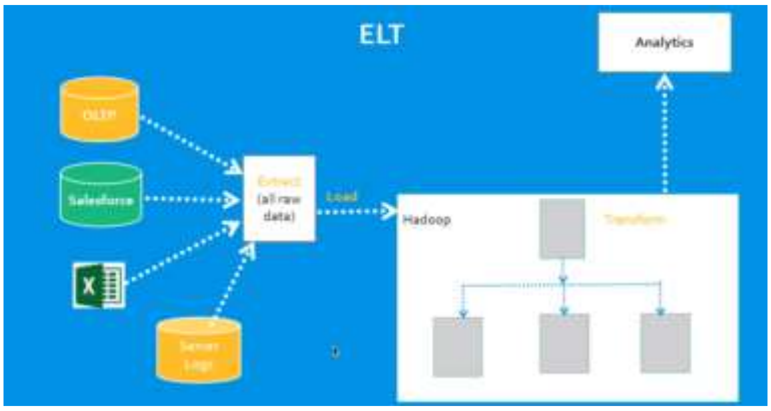
iii) Distributed Filesystem returns a FSDataInputStream to the shopper for it to scan information from. FSDataInputStream, thus, wraps the DFSInputStream that manages the datanode and name node I/O. shopper calls read () on the stream. DFSInputStream that has keep the datanode addresses then connects to the nearest datanode for the primary block within the file.

iv) Information is streamed from the datanode back to the shopper, as a result shopper will decision read () repeatedly on the stream. Once the block ends, DFSInputStream can shut the association to the data node then finds the simplest data node for succeeding block.

v) If the DFSInputStream encounters a blunder whereas communication with a datanode, it'll attempt succeeding nearest one for that block. It’ll conjointly bear in mind datanodes that have failing so it doesn’t needlessly rehear them for later blocks. The DFSInputStream conjointly verifies checksums for the info transferred to that from the datanode. If it finds a corrupt block, it reports this to the namenode before theDFSInputStream makes an attempt to scan a duplicate of the block from another datanode.

vi) once the shopper has finished reading the info, it calls close() on the stre

ELT and ETL



These 2 definitions of ETL area unit what create ELT somewhat confusing? ELT may be a completely different means of staring at the tool approach to information movement. Rather than reworking the information before it's written, ELT leverages the target system to try and do the transformation. The information is traced to the target so remodeled in situ.

ETL, on the opposite hand, is intended employing a pipeline approach. Whereas information is flowing from the supply to the target, a metamorphosis engine (something distinctive to the tool) takes care of any information changes.

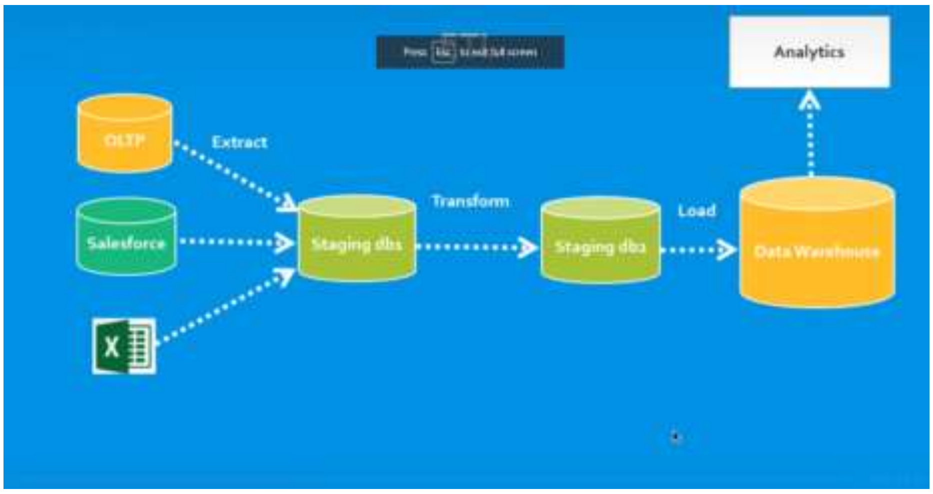
Which is best depends on priorities. All things being equal, it's higher to possess fewer moving elements. ELT has no transformation engine – the work is finished by the target system, which is already there and possibly being employed for alternative development work. On the opposite hand, the ETL approach will offer drastically higher performance in bound situations.

The coaching and development prices of ETL ought to be weighed against the requirement for higher performance. (Additionally, if you don't have a target system powerful enough for ELT, ETL could also be additional economical.

The specifics of ELT development can vary counting on the platform. as an example, Hadoop clusters work by breaking a tangle into smaller chunks, then distributing those chunks across an oversized variety of machines for process.

The problem is solved quicker as a result of it's being worn out parallel. this needs careful style to create certain that the act of rending the matter will be refrained from poignant the solution. Some issues will be simply split, others are going to be a lot of tougher.

ELT is a wonderful military science tool for loading a knowledge warehouse. It needs a strong system in situ because the target, however additional and additional warehouses area unit being designed with such systems in mind to fulfill ever-growing analytic wants. like any tool, knowing once to use it's a minimum of as vital as knowing the way to use it. Ironsides will offer strategic direction and/or technical support in information integration and management. Contact North American nation nowadays to debate that choices suit your surroundings best.



Different Vendors:

Amazon Elastic MapReduce (EMR)

It is Associate in Nursing Amazon internet Services (AWS) tool for giant processing and analysis. Amazon EMR offers the expandable low-configuration service as a neater different to running in-house cluster computing.

Amazon EMR is predicated on Apache Hadoop, a Java-based programming framework that supports the process of huge knowledge sets in a very distributed computing setting. MapReduce may be a package framework that permits developers to write down programs that method large amounts of unstructured knowledge in parallel across a distributed cluster of processors or complete computers. it had been developed at Google for categorization sites and replaced their original categorization algorithms and heuristics in 2004.

Amazon EMR processes huge knowledge across a Hadoop cluster of virtual servers on Amazon Elastic reckon Cloud (EC2) and Amazon straightforward Storage Service (S3). The elastic in EMR's name refers to its dynamic resizing ability, that permits it to build or cut back resource use betting on the demand at any given time.

Cloud Era

Cloudera opposition. may be a United States-based package company that has Apache Hadoop-based package, support and services, and coaching to business customers.

Cloudera's ASCII text file Apache Hadoop distribution, CDH (Cloudera Distribution together with Apache Hadoop), targets enterprise-class deployments of that technology. Cloudera says that quite five hundredth of its engineering output is given upstream to the assorted Apache-licensed open supply comes (Apache Hive, Apache Avro, Apache HBase, and then on) that mix to create the Hadoop platform. Cloudera is additionally a sponsor of the Apache package Foundation.

IBM InfoSphere

InfoSphere DataStage may be a powerful knowledge integration tool.

It was noninheritable by IBM in 2005 and has become a neighborhood of IBM info Server Platform. It uses a shopper/server style wherever jobs square measure created and administered via a Windows client against central repository on a server.

The IBM InfoSphere DataStage is capable of desegregation knowledge on demand across multiple and high volumes of information sources and target applications employing a high performance parallel framework. InfoSphere DataStage additionally facilitates extended information management and enterprise property. it's 3 levels of correspondence that are:

Pipeline correspondence, knowledge correspondence, part correspondence

Teradata

Teradata Corporation may be a supplier of database-related product and services. The corporate was fashioned in 1979 in Brentwood, California, as a collaboration between researchers at Caltech and Citibank's advanced technology cluster.[2] the corporate was noninheritable by NCR Corporation in 1991, Associate in Nursingd later on spun-off once more as an freelance public company on October one, 2007.

The company produces a electronic database management system of a similar name, that it markets as an information warehouse. Teradata offers 3 main services to its customers: cloud and hardware-based knowledge repositing, business analytics, and consulting services.

the company launched Teradata everyplace, that permits users to submit queries against public and personal databases. The service uses massively data processing across each its physical knowledge warehouse and cloud storage, together with managed environments like Amazon internet Services, Microsoft Azure, VMware, and Teradata's Managed Cloud and IntelliFlex Teradata offers customers each hybrid cloud and multi-cloud storage.

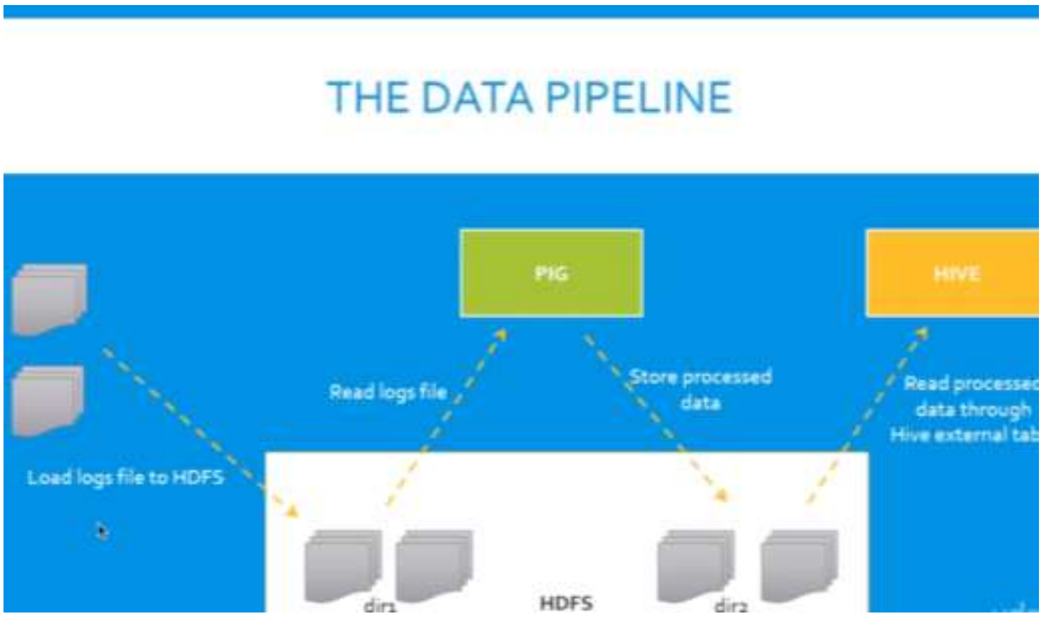
DATA PIPELINE DESIGN

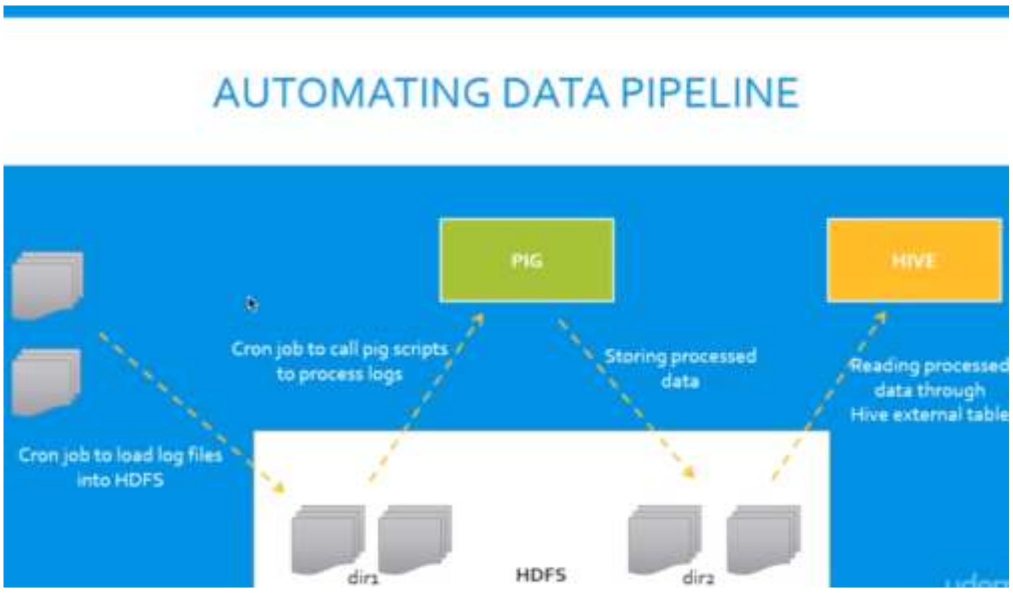
Our project is mainly based on Data Pipeline Design.

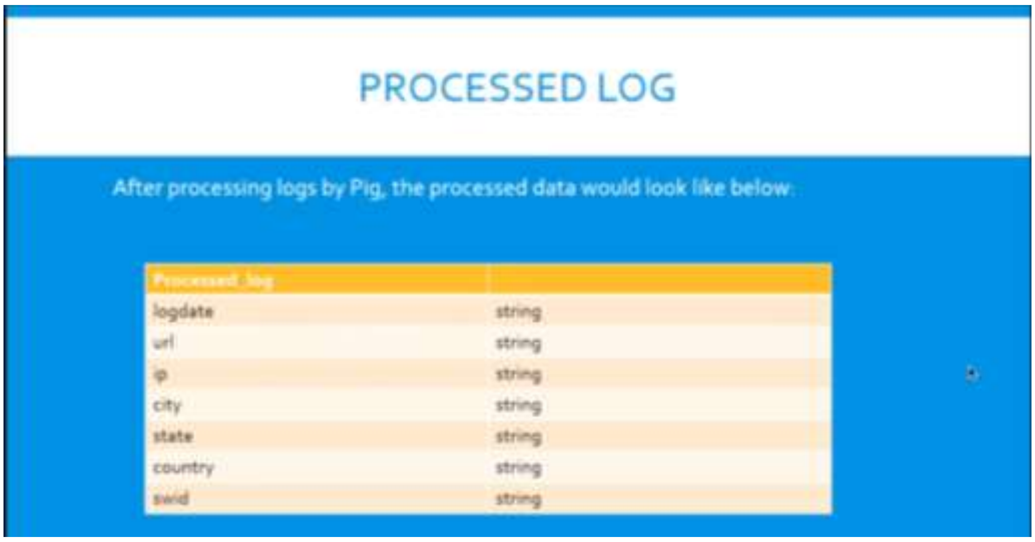
Introduction

A data pipeline is the process of structuring, processing and transforming data in stages regardless of what the source data form may be.

Data pipeline is an automated process that executes at regular interval to ingest, cleanse, transform and aggregates incoming feed of data to generate the output data set in the format that is suitable for downstream processing, with no manual intervention.




AUTOMATING DATA PIPELINE
PIG
HIVE
Cron job to call pig scripts to process logs
Storing processed data
Reading processed data through Hive external table
Cron job to load log files into HDFS
dir1
HDFS
dir2



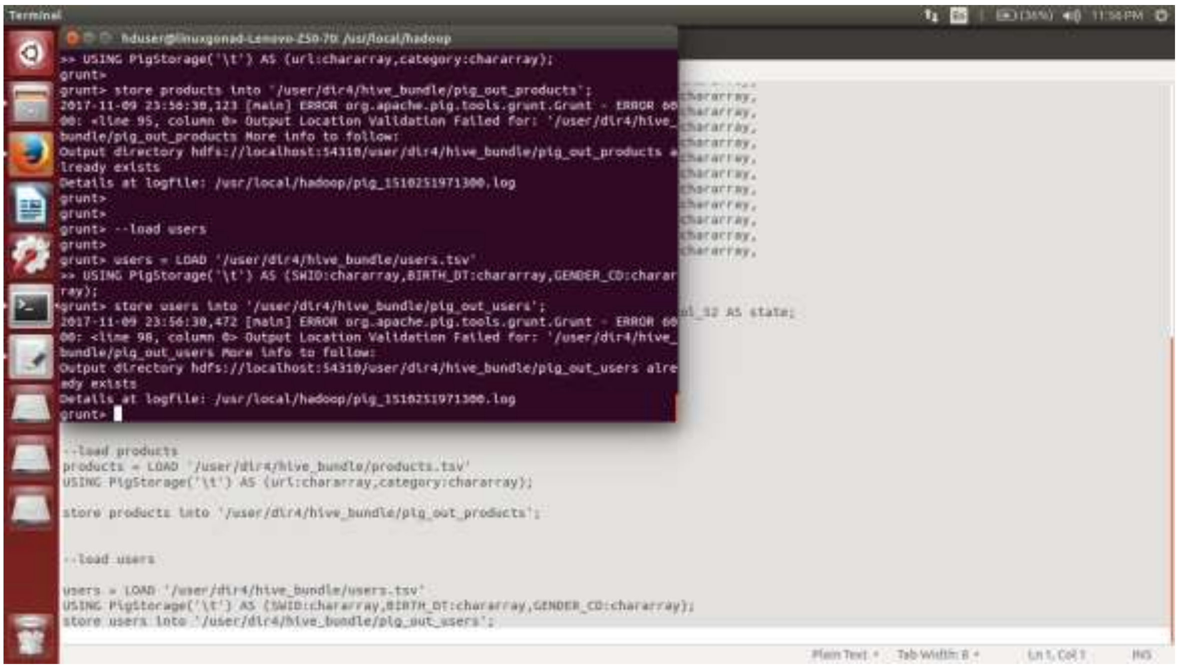
PROCESSED LOG

After processing logs by Pig, the processed data would look like below

Processed_log	
logdate	string
url	string
ip	string
city	string
state	string
country	string
swid	string

Implementation is as follows:-

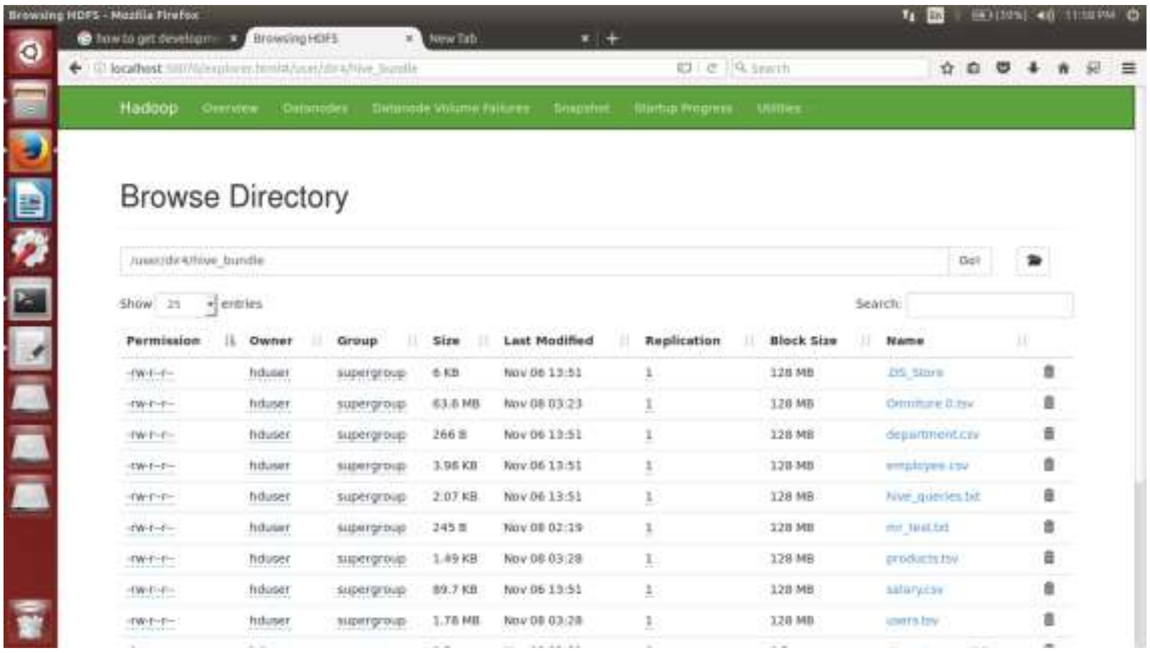
Pig Script ran first on the Omniture tsv file



```
>> USING PigStorage('\t') AS (url:chararray,category:chararray);
grunt>
grunt> store products into '/user/dir4/hive_bundle/pig_out_products';
2017-11-09 23:56:30,123 [main] ERROR org.apache.pig.tools.grunt.Grunt - ERROR 60
00: <line 95, column 0> Output Location Validation Failed for: '/user/dir4/hive_
bundle/pig_out_products More info to follow:
Output directory hdfs://localhost:54310/user/dir4/hive_bundle/pig_out_products a
lready exists
Details at logfile: /usr/local/hadoop/pig_1510251971300.log
grunt>
grunt>
grunt> --load users
grunt>
grunt> users = LOAD '/user/dir4/hive_bundle/users.tsv'
>> USING PigStorage('\t') AS (SWID:chararray,BIRTH_DT:chararray,GENDER_CD:charar
ray);
grunt> store users into '/user/dir4/hive_bundle/pig_out_users';
2017-11-09 23:56:30,472 [main] ERROR org.apache.pig.tools.grunt.Grunt - ERROR 60
00: <line 98, column 0> Output Location Validation Failed for: '/user/dir4/hive_
bundle/pig_out_users More info to follow:
Output directory hdfs://localhost:54310/user/dir4/hive_bundle/pig_out_users alre
ady exists
Details at logfile: /usr/local/hadoop/pig_1510251971300.log
grunt>
```

```
--load products
products = LOAD '/user/dir4/hive_bundle/products.tsv'
USING PigStorage('\t') AS (url:chararray,category:chararray);

store products into '/user/dir4/hive_bundle/pig_out_products';


--load users

users = LOAD '/user/dir4/hive_bundle/users.tsv'
USING PigStorage('\t') AS (SWID:chararray,BIRTH_DT:chararray,GENDER_CD:chararray);
store users into '/user/dir4/hive_bundle/pig_out_users';
```

Web UI during the pipeline showcasing the logs that are obtained after Pig data extraction



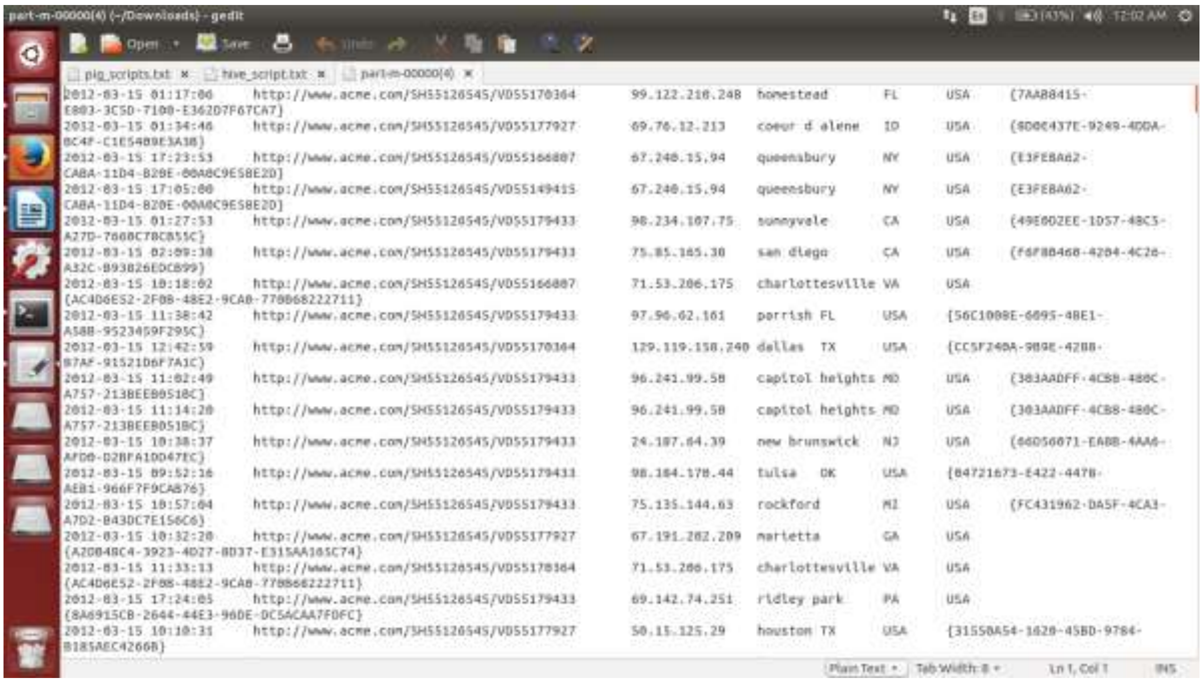
Browse Directory

/user/dir4/hive_bundle

Show 25 entries

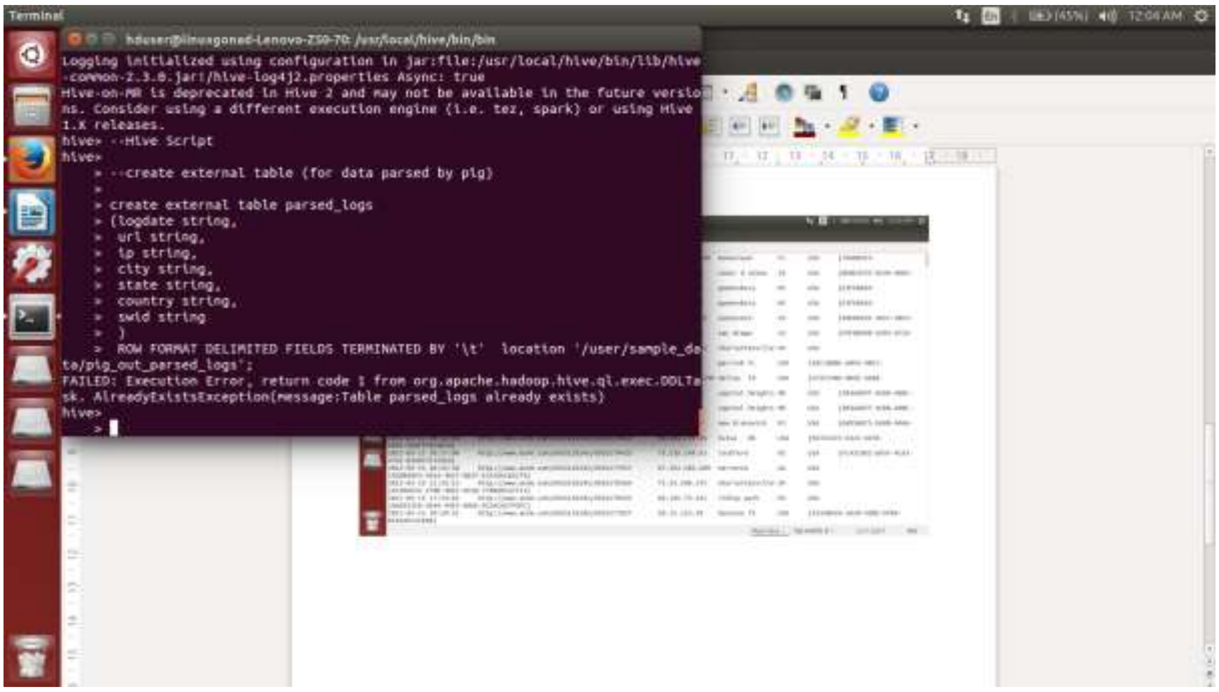
Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rw-r--r--	hduser	supergroup	6 KB	Nov 06 13:51	1	128 MB	.DS_Store
-rw-r--r--	hduser	supergroup	63.6 MB	Nov 08 03:23	1	128 MB	Omniture.0.tsv
-rw-r--r--	hduser	supergroup	266 B	Nov 06 13:51	1	128 MB	department.csv
-rw-r--r--	hduser	supergroup	3.96 KB	Nov 06 13:51	1	128 MB	employee.csv
-rw-r--r--	hduser	supergroup	2.07 KB	Nov 06 13:51	1	128 MB	hive_queries.txt
-rw-r--r--	hduser	supergroup	245 B	Nov 08 02:19	1	128 MB	mr_test.txt
-rw-r--r--	hduser	supergroup	1.49 KB	Nov 08 03:28	1	128 MB	products.tsv
-rw-r--r--	hduser	supergroup	89.7 KB	Nov 06 13:51	1	128 MB	salary.csv
-rw-r--r--	hduser	supergroup	1.78 MB	Nov 08 03:28	1	128 MB	users.tsv

pig_parsed_logs



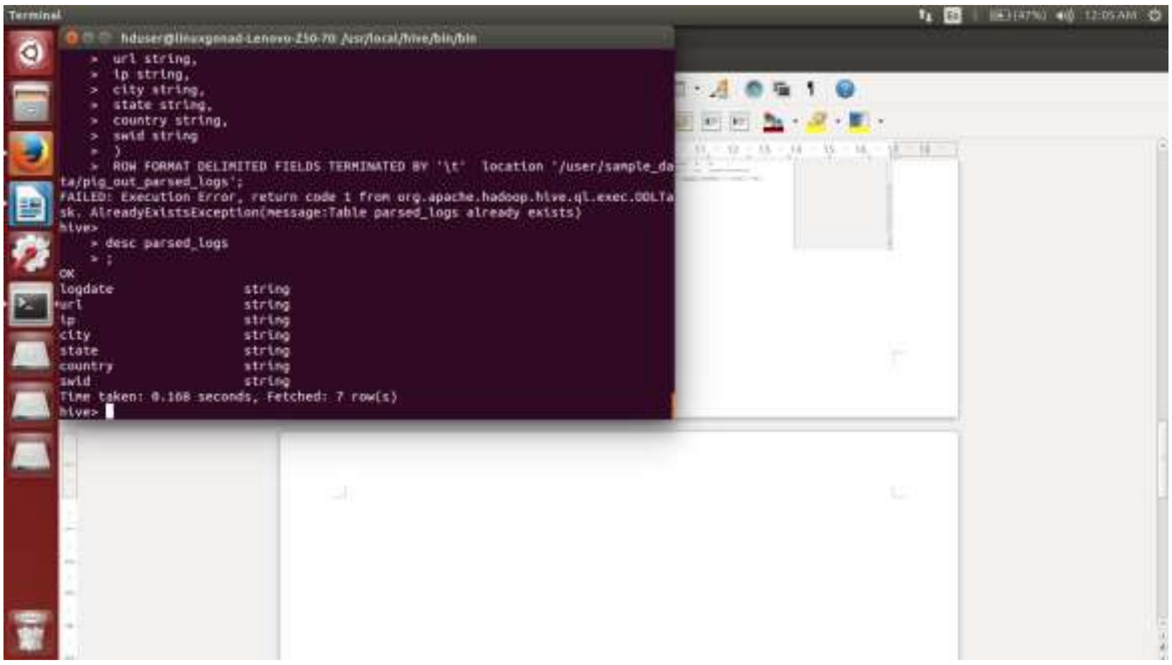
```
2012-03-15 01:17:06     http://www.acme.com/SH55126545/VD55170364       99.122.210.248  homestead       FL      USA     {7AAB8415-
E803-3C5D-7100-E362D7F67CA7}
2012-03-15 01:34:46     http://www.acme.com/SH55126545/VD55177927       69.76.12.213    coeur d alene   ID      USA     {9D0C437E-9249-4DDA-
BC4F-C1E5409E3A38}
2012-03-15 17:23:53     http://www.acme.com/SH55126545/VD55166807       67.240.15.94    queensbury      NY      USA     {E3FEBA62-
CABA-11D4-820E-00A0C9E58E2D}
2012-03-15 17:05:00     http://www.acme.com/SH55126545/VD55149415       67.240.15.94    queensbury      NY      USA     {E3FEBA62-
CABA-11D4-820E-00A0C9E58E2D}
2012-03-15 01:27:53     http://www.acme.com/SH55126545/VD55179433       98.234.107.75   sunnyvale       CA      USA     {49E0D2EE-1D57-48C5-
A27D-7660C78CB55C}
2012-03-15 02:09:38     http://www.acme.com/SH55126545/VD55179433       75.85.165.38    san diego       CA      USA     {f6F80468-4204-4C26-
A32C-B93826EDCB99}
2012-03-15 10:18:02     http://www.acme.com/SH55126545/VD55166807       71.53.206.175   charlottesville VA      USA
{AC4D6E52-2F08-48E2-9CA0-770068222711}
2012-03-15 11:38:42     http://www.acme.com/SH55126545/VD55179433       97.96.62.161    parrish FL      USA     {56C1008E-6695-48E1-
A588-9523459F295C}
2012-03-15 12:42:59     http://www.acme.com/SH55126545/VD55170364       129.119.158.240 dallas  TX      USA     {CC5F240A-989E-4288-
B7AF-91521D6F7A1C}
2012-03-15 11:02:49     http://www.acme.com/SH55126545/VD55179433       96.241.99.58    capitol heights MD      USA     {383AADFF-4CB8-480C-
A757-213BEEB0510C}
2012-03-15 11:14:20     http://www.acme.com/SH55126545/VD55179433       96.241.99.58    capitol heights MD      USA     {303AADFF-4CB8-480C-
A757-213BEEB051BC}
2012-03-15 10:38:37     http://www.acme.com/SH55126545/VD55179433       24.187.64.39    new brunswick   NJ      USA     {66D56071-EA8B-4AA6-
AFD0-D2BFA1DD47EC}
2012-03-15 09:52:16     http://www.acme.com/SH55126545/VD55179433       98.184.178.44   tulsa   OK      USA     {84721673-E422-4478-
AEB1-966F7F9CA876}
2012-03-15 10:57:04     http://www.acme.com/SH55126545/VD55179433       75.135.144.63   rockford        MI      USA     {FC431962-DA5F-4CA3-
A7D2-843DC7E156C6}
2012-03-15 10:32:20     http://www.acme.com/SH55126545/VD55177927       67.191.202.209  marietta        GA      USA
{A2D048C4-3923-4D27-8D37-E315AA165C74}
2012-03-15 11:33:13     http://www.acme.com/SH55126545/VD55170364       71.53.206.175   charlottesville VA      USA
{AC4D6E52-2F08-48E2-9CA0-770068222711}
2012-03-15 17:24:05     http://www.acme.com/SH55126545/VD55179433       69.142.74.251   ridley park     PA      USA
{8A6915CB-2644-44E3-96DE-DC5ACAA7F0FC}
2012-03-15 10:10:31     http://www.acme.com/SH55126545/VD55177927       50.15.125.29    houston TX      USA     {31550A54-1620-45BD-9784-
B185AEC4266B}
```

Hive script ran over data stored after pig partitioning



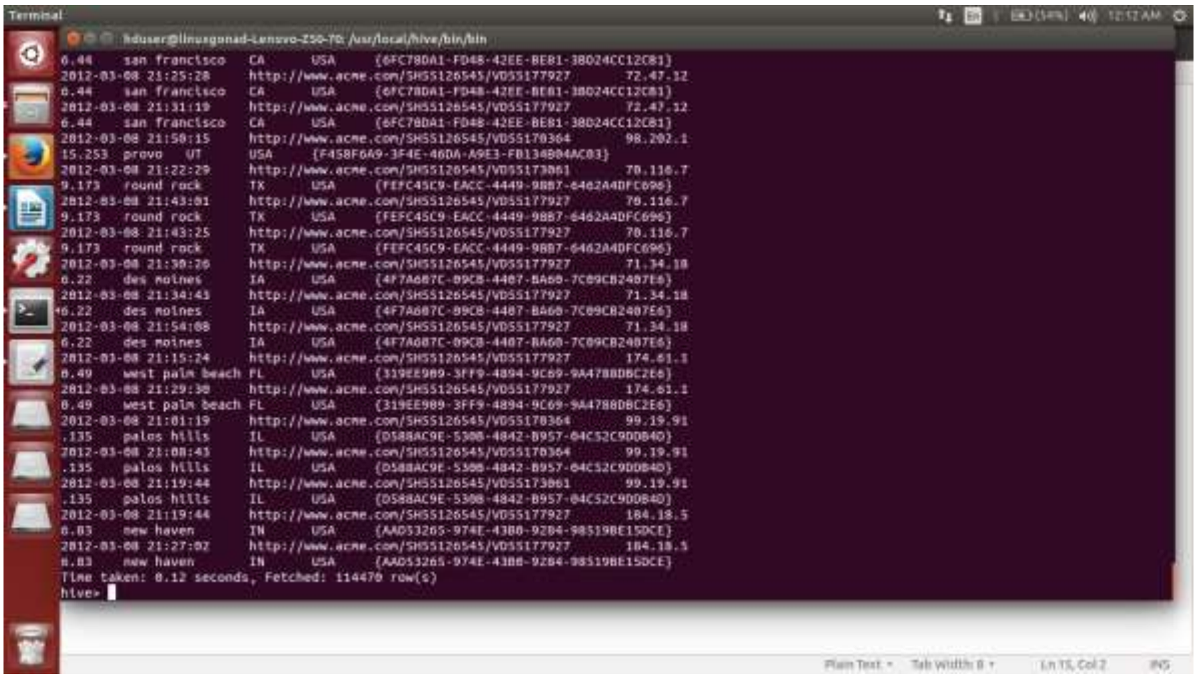
```
hduser@linuxgonad-Lenovo-Z50-70: /usr/local/hive/bin/bin
Logging initialized using configuration in jar:file:/usr/local/hive/bin/lib/hive
-common-2.3.0.jar!/hive-log4j2.properties Async: true
Hive-on-MR is deprecated in Hive 2 and may not be available in the future versio
ns. Consider using a different execution engine (i.e. tez, spark) or using Hive
1.X releases.
hive> --Hive Script
hive>
    > --create external table (for data parsed by pig)
    >
    > create external table parsed_logs
    > (logdate string,
    >  url string,
    >  ip string,
    >  city string,
    >  state string,
    >  country string,
    >  swid string
    >  )
    >  ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t'  location '/user/sample_da
ta/pig_out_parsed_logs';
FAILED: Execution Error, return code 1 from org.apache.hadoop.hive.ql.exec.DDLTa
sk. AlreadyExistsException(message:Table parsed_logs already exists)
hive>
    >
```

desc parsed_logs



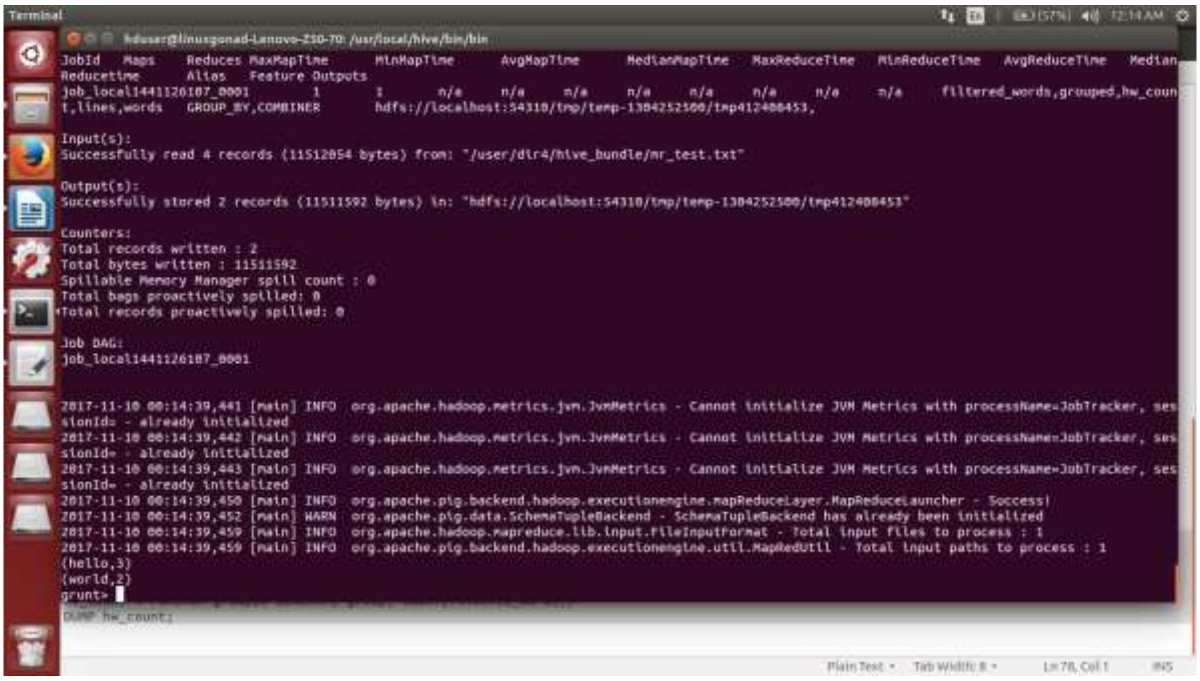
```
hduser@linuxgonad-Lenovo-Z50-70: /usr/local/hive/bin/bin
    >  url string,
    >  ip string,
    >  city string,
    >  state string,
    >  country string,
    >  swid string
    >  )
    >  ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t'  location '/user/sample_da
ta/pig_out_parsed_logs';
FAILED: Execution Error, return code 1 from org.apache.hadoop.hive.ql.exec.DDLTa
sk. AlreadyExistsException(message:Table parsed_logs already exists)
hive>
    > desc parsed_logs
    > ;
OK
logdate                 string
url                     string
ip                      string
city                    string
state                   string
country                 string
swid                    string
Time taken: 0.168 seconds, Fetched: 7 row(s)
hive>
```

Finally end of hive script returning all the records in a structured format along with schema



```
hduser@linusgonad-Lenovo-Z50-70: /usr/local/hive/bin/bin
6.44    san francisco   CA      USA     {6FC78DA1-FD48-42EE-BE81-38D24CC12C81}
2012-03-08 21:25:28     http://www.acme.com/SH55126545/VD55177927       72.47.12
6.44    san francisco   CA      USA     {6FC78DA1-FD48-42EE-BE81-38D24CC12C81}
2012-03-08 21:31:19     http://www.acme.com/SH55126545/VD55177927       72.47.12
6.44    san francisco   CA      USA     {6FC78DA1-FD48-42EE-BE81-38D24CC12C81}
2012-03-08 21:50:15     http://www.acme.com/SH55126545/VD55178364       98.202.1
15.253  provo   UT      USA     {F458F6A9-3F4E-46DA-A9E3-FB134B04AC83}
2012-03-08 21:22:29     http://www.acme.com/SH55126545/VD55173061       70.116.7
9.173   round rock      TX      USA     {FEFC45C9-EACC-4449-9BB7-6462A4DFC696}
2012-03-08 21:43:01     http://www.acme.com/SH55126545/VD55177927       70.116.7
9.173   round rock      TX      USA     {FEFC45C9-EACC-4449-9BB7-6462A4DFC696}
2012-03-08 21:43:25     http://www.acme.com/SH55126545/VD55177927       70.116.7
9.173   round rock      TX      USA     {FEFC45C9-EACC-4449-9BB7-6462A4DFC696}
2012-03-08 21:30:26     http://www.acme.com/SH55126545/VD55177927       71.34.18
6.22    des moines      IA      USA     {4F7A687C-09CB-4407-BA60-7C09CB2407E6}
2012-03-08 21:34:43     http://www.acme.com/SH55126545/VD55177927       71.34.18
6.22    des moines      IA      USA     {4F7A687C-09CB-4407-BA60-7C09CB2407E6}
2012-03-08 21:54:08     http://www.acme.com/SH55126545/VD55177927       71.34.18
6.22    des moines      IA      USA     {4F7A687C-09CB-4407-BA60-7C09CB2407E6}
2012-03-08 21:15:24     http://www.acme.com/SH55126545/VD55177927       174.61.1
0.49    west palm beach FL      USA     {319EE989-3FF9-4894-9C69-9A4788DBC2E6}
2012-03-08 21:29:30     http://www.acme.com/SH55126545/VD55177927       174.61.1
0.49    west palm beach FL      USA     {319EE989-3FF9-4894-9C69-9A4788DBC2E6}
2012-03-08 21:01:19     http://www.acme.com/SH55126545/VD55178364       99.19.91
.135    palos hills     IL      USA     {D588AC9E-5308-4842-B957-04C52C9DDB4D}
2012-03-08 21:08:43     http://www.acme.com/SH55126545/VD55178364       99.19.91
.135    palos hills     IL      USA     {D588AC9E-5308-4842-B957-04C52C9DDB4D}
2012-03-08 21:19:44     http://www.acme.com/SH55126545/VD55173061       99.19.91
.135    palos hills     IL      USA     {D588AC9E-5308-4842-B957-04C52C9DDB4D}
2012-03-08 21:19:44     http://www.acme.com/SH55126545/VD55177927       184.18.5
0.83    new haven       IN      USA     {AAD53265-974E-4380-9284-98519BE15DCE}
2012-03-08 21:27:02     http://www.acme.com/SH55126545/VD55177927       184.18.5
0.83    new haven       IN      USA     {AAD53265-974E-4380-9284-98519BE15DCE}
Time taken: 0.12 seconds, Fetched: 114470 row(s)
hive>
```

Hello world count Pig program



```
hduser@linusgonad-Lenovo-Z50-70: /usr/local/hive/bin/bin
JobId   Maps    Reduces MaxMapTime      MinMapTime      AvgMapTime      MedianMapTime   MaxReduceTime   MinReduceTime   AvgReduceTime   Median
Reducetime      Alias   Feature Outputs
job_local1441126187_0001        1       1       n/a     n/a     n/a     n/a     n/a     n/a     n/a     n/a     filtered_words,grouped,hw_coun
t,lines,words   GROUP_BY,COMBINER       hdfs://localhost:54310/tmp/temp-1304252500/tmp412408453,

Input(s):
Successfully read 4 records (11512054 bytes) from: "/user/dir4/hive_bundle/mr_test.txt"

Output(s):
Successfully stored 2 records (11511592 bytes) in: "hdfs://localhost:54310/tmp/temp-1304252500/tmp412408453"

Counters:
Total records written : 2
Total bytes written : 11511592
Spillable Memory Manager spill count : 0
Total bags proactively spilled: 0
Total records proactively spilled: 0

Job DAG:
job_local1441126187_0001


2017-11-10 00:14:39,441 [main] INFO  org.apache.hadoop.metrics.jvm.JvmMetrics - Cannot initialize JVM Metrics with processName=JobTracker, ses
sionId= - already initialized
2017-11-10 00:14:39,442 [main] INFO  org.apache.hadoop.metrics.jvm.JvmMetrics - Cannot initialize JVM Metrics with processName=JobTracker, ses
sionId= - already initialized
2017-11-10 00:14:39,443 [main] INFO  org.apache.hadoop.metrics.jvm.JvmMetrics - Cannot initialize JVM Metrics with processName=JobTracker, ses
sionId= - already initialized
2017-11-10 00:14:39,450 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Success!
2017-11-10 00:14:39,452 [main] WARN  org.apache.pig.data.SchemaTupleBackend - SchemaTupleBackend has already been initialized
2017-11-10 00:14:39,459 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input files to process : 1
2017-11-10 00:14:39,459 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process : 1
(hello,3)
(world,2)
grunt>
DUMP hw_count;
```