PROTOTYPE IMPLEMENTATION OF VIRTUAL DATACENTER USING HADOOP FRAMEWORK FOR DISTRIBUTED AND PARALLEL BIG DATA PROCESSING

BY
A K M MAHBUBUL HOSSEN
102-25-157

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

Supervised By

Dr Syed Akhter Hossain
Professor and Head
Department of Computer Science and Engineering
Daffodil International University

Co-Supervised By

A B M Moniruzzaman
Lecturer
Department of Computer Science and Engineering
Daffodil International University

DAFFODIL INTERNATIONAL UNIVERSITY
DHAKA, BANGLADESH
APRIL 2015

APPROVAL

This Project titled **"Prototype Implementation of Virtual Datacenter using Hadoop Framework for Distributed and Parallel Big Data Processing",** submitted by A K M MahbubulHossen to the Department of Computer Science and Engineering, Daffodil International University, has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of M.Sc. in Computer Science and Engineering (MSc) and approved as to its style and contents. The presentation has been held on 21st April 2105.

BOARD OF EXAMINERS

_____

Dr Syed Akhter Hossain — **Chairman**
Professor and Head
Department of Computer Science and Engineering
Faculty of Science & Information Technology
DaffodilInternationalUniversity

_____

DrSheakRashedHaiderNoori — **Internal Examiner**
Asstt. Professor
Department of Computer Science and Engineering
Faculty of Science & Information Technology
DaffodilInternationalUniversity

_____

A.H.M. Saiful Islam — **Internal Examiner**
Asstt. Professor
Department of Computer Science and Engineering
Faculty of Science & Information Technology
DaffodilInternationalUniversity

_____

Dr Muhammad Shorif Uddin — **External Examiner**
Professor and Chairperson
Department of Computer Science and Engineering
JahangirnagarUniversity

DECLARATION

I hereby declare that, this project has been done by me under the supervision of**Dr Syed Akhter Hossain, Professor and Head, Department of CSE,** co-supervised by **A B M Moniruzzaman, Lecturer, Department of CSE,**Daffodil International University. I 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:

_____

Dr Syed Akhter Hossain
Professor and Head
Department of CSE
Daffodil International University

Co-Supervised by:

_____

A B M Moniruzzaman
Lecturer
Department of CSE
Daffodil International University

Submitted by:

_____

A K M MahbubulHossen
ID: 102-25-157
Department of CSE
DaffodilInternationalUniversity

ACKNOWLEDGEMENT

First I express my heartiest thanks and gratefulness to almighty Allah for His divine blessing makes me possible to complete the final year project successfully.

I really grateful and wish my profound my indebtedness to**Dr. Prof. Syed Akhter Hossain**, Professor and Head,Department of CSE, Daffodil International University, Dhaka. Deep Knowledge & keen interest of mysupervisor's in the field of "*Computer Science*" to carry out this project. His endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice,reading many inferior draft and correcting them at all stage have made it possible to complete this project.

I would like to express my heartiest gratitude to **A B M Moniruzzaman**, Lecturer, Department of CSE, for his kind help to finish my project and also to other faculty member and the staff of CSE department of Daffodil International University.

I would like to thank my entire course mate in Daffodil International University, who took part in this discuss while completing the course work.

Finally, I must acknowledge with due respect the constant support and patients of my parents and colleagues.

ABSTRACT

The aim of this project is to implement a prototype of a virtual datacenter using distributed and parallel computing technology. Here the main concept is to reduce datacenter implementation cost using commodity hardware and provide high performance. This project will explain the benefit of using distributed and parallel computing architecture. This report will also help us to explain about some new technology and framework which are open sourced and we can easily utilize those technologies for our complex data analysis which resembling structured, semi structured and non-structured data.

Here a high level virtual datacenter design has been provided with implementation steps. Some processing comparison between distributed and parallel computing with single computing system also provided to explain the value of this project.

To develop this project the most essential component was Hardware Virtualization, Operating System and Hadoop Framework.

After implementation of this project, the system is tested in different stages and it works successfully as a prototype.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

Introduction

1.1Background

The digitization of everything is creating a data explosion near us. Data is rapidly increasing day by day. Specially unstructured data is like bursting every year comparing to semi structured and structured data. Massive use of e-mails, social networking, official documents, scan copy, video calls, mobile data, video and audio data, medical equipment data, log files, system generated data are some good example of extensive data growth.

Growth rate of structured and unstructured data has been shown in figure 1.1

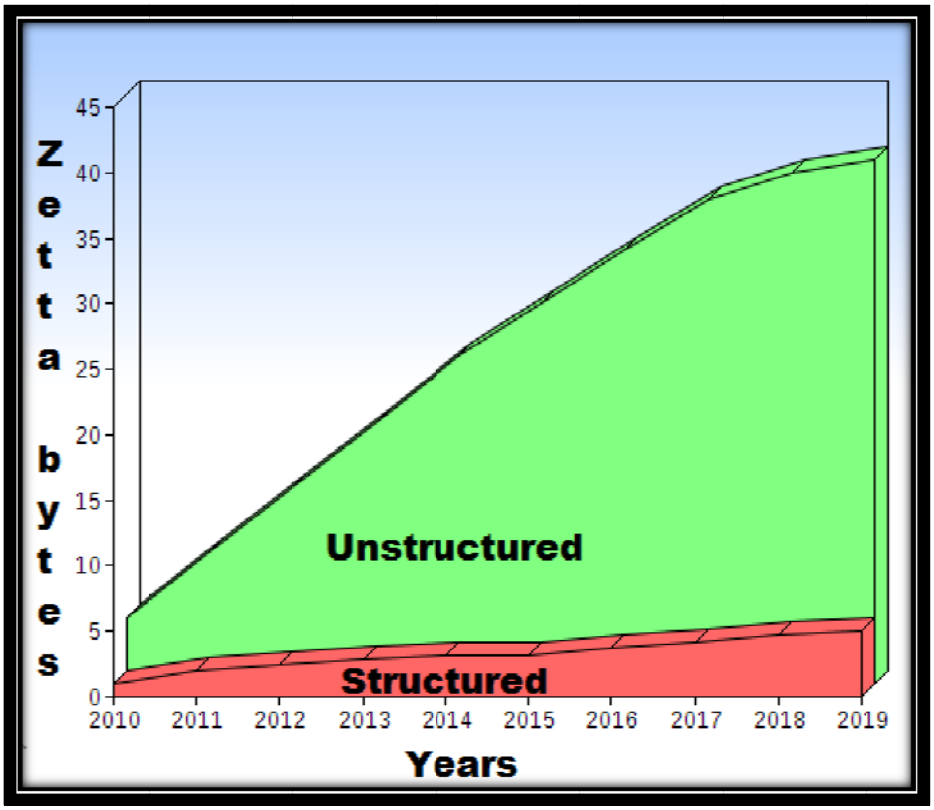


Figure 1.1: Data growth per year (Structured and Unstructured data)[1]

Which type of data growing much faster is shown here in figure 1.2

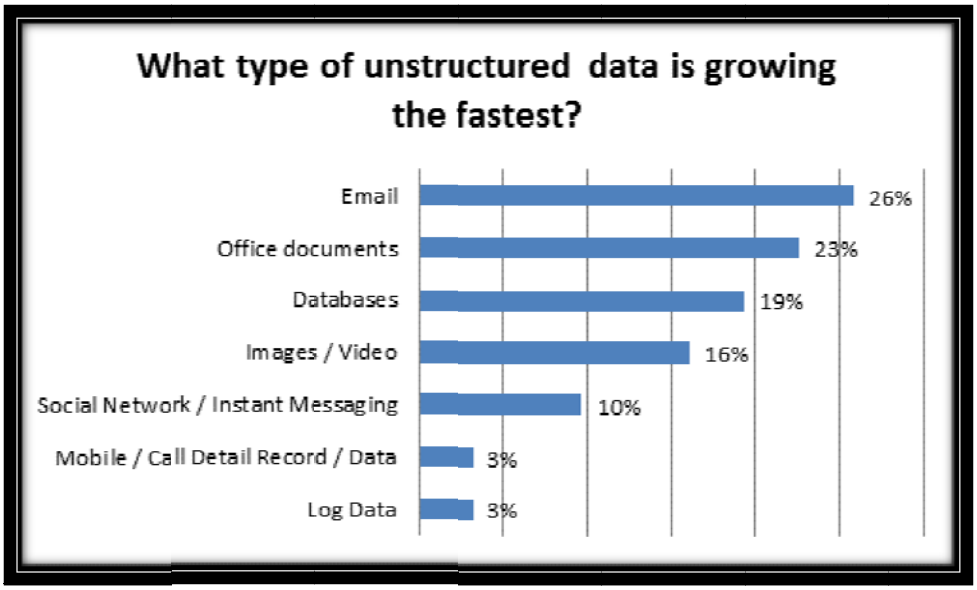


Figure 1.2: Which type of unstructured data growing fast?[2]

Allowing to this data growth people also trying to handle and process this huge amount of data by constructing huge datacenter and using high-end powerful computing system, which consumes more space, more electricity, more man power and more money. Again it's difficult to handle structured and unstructured data at a time which also huge cost involved. If we consider this costing for small and semi large organizations it's like waste of money without any benefit. In this paper it has been presented that how can we handle this huge amount of data efficiently and cost effectively by implementing a virtual datacenter for distributed and parallel computing.

1.2Feasibility Study

While assigning this projectsome new technologies about distributed and parallel processing has been analyzed to get a perfect idea to design a virtual datacenter using commodity hardware which will be cost effective and better performance considering small and semi-large organizations.In this project a virtual datacenter has been design assuming an infrastructure which have two remote sites interconnected through internet. It can be implement on virtual environment to simulate this project and perform some data processing query to prove the feasibility according to the technical concept of this project.

1.3Goals and Scopes

The objective of this project is to design and establish a virtual datacenter using commodity hardware to perform distributed and parallel processing which will save the costing, space requirement, electricity, man power as well as maintaining better computing performance.

This project established on a limited virtual environment because of resource availability and technical complexity. But it can be implement in any organization who already have commodity hardware which are not using or not fully utilized.

1.4Some Technical Concepts

In this project report there are some technical term has been used which are very important to understand the project clearly. Keeping it on mind some technical concept has been described here before start explaining the project work.

1.4.1 Big Data

Big data is an emerging term todescribe any voluminous amount of structured, semi-structured and unstructured data. Although big data doesn't refer to any specific quantity, the term is often used when speaking about petabytes and exabytes and more than that amount of data. [3]

Data sets grow in size because they are increasingly rapidly being gathered by cheap and numerous information-sensing mobile devices, remote sensing, software logs, video cameras, microphones, radio-frequency identification (RFID) readers, and wireless sensor networks equipment.The world's technological per-capita capacity to store information has roughly doubled every 40 months since the 1980s;as of 2012, every day 2.5 exabytes (2.5×1018) of data were created;The challenge for large enterprises is determining who should own big data initiatives that straddle the entire organization. [4]. A table of data magnitude orders has been shown in below figure 1.3.

Multiples of bytes						
Decimal			Binary			
Value	Metric		Value	JEDEC	IEC	
1000	kB	kilobyte	1024	KB kilobyte	KiB	kibibyte
1000^2	MB	megabyte	1024^2	MB megabyte	MiB	mebibyte
1000^3	GB	gigabyte	1024^3	GB gigabyte	GiB	gibibyte
1000^4	TB	terabyte	1024^4	– –	TiB	tebibyte
1000^5	PB	petabyte	1024^5	– –	PiB	pebibyte
1000^6	EB	exabyte	1024^6	– –	EiB	exbibyte
1000^7	ZB	zettabyte	1024^7	– –	ZiB	zebibyte
1000^8	YB	yottabyte	1024^8	– –	YiB	yobibyte

Figure1.3: Orders of magnitude of data [5]

1.4.2 Hardware Virtualization

Hardware virtualization or platform virtualization refers to virtual machine creation which are acts like a real computer with an operating system. On these virtual machines software execution is separated from the underlying hardware resources. For example, a computer that is running Microsoft Windows may host a machine virtually that appears like a computer with the other types of operating system; Ubuntu-based software can be run on the virtual machine. In hardware virtualization, the host machine is the genuine machine on which the virtualization takes place, and the virtual machine is the guest machine. The words host andguest are used to distinguish the software that runs on the physical machine from the software that runs on the virtual machine. The software or firmware that creates a virtual machine on the host hardware is called a hypervisor or Virtual Machine Manager. [6]

1.4.3 Hadoop

The Apache Hadoop software library is a framework that allows for the distributed processing of large data sets across clusters of computers using simple programming models. Hadoop is designed to scale up from single servers to thousands of servers, each involving local computation and storage. Rather than rely on hardware to deliver high-availability, the library itself is designed to detect and handle failures at the application layer, so delivering a highly available service on top of a cluster of computers, each of which may be prone to failures.Basically, it's a way of storing

enormous data sets across distributed clusters of servers and then running "distributed" analysis applications in each cluster.It's designed to be robust, in that our Big Data applications will continue to run even when individual servers — or clusters — fail. And it's also designed to be efficient, because it doesn't require applications to shuttle huge volumes of data across your network.

Hadoop has two main parts - a data processing framework (MapReduce) and a distributed filesystem for data storage (HDFS).

The distributed filesystem HDFS is like the bucket of the Hadoop system. We can dump in our data and it sits there all nice and cozy until we want to do something with it, whether that's running an analysis on it within Hadoop or capturing and exporting a set of data to another tool and performing the analysis there.

The data processing framework is the tool used to work with the data itself. By default, this is the Java-based system known as MapReduce. In a "normal" relational database, data is found and analyzed using queries, based on the industry-standard Structured Query Language (SQL). Non-relational databases use queries, too; they're just not constrained to use only SQL, but can use other query languages to pull information out of data stores. Hence, the term NoSQL. But Hadoop is not really a database. It stores data and we can pull data out of it, but there are no queries involved - SQL or otherwise. Hadoop is more of a data warehousing system - so it needs a system like MapReduce to actually process the data. MapReduce runs as a series of jobs, with each job essentially a separate Java application that goes out into the data and starts pulling out information as needed. [7]

Hadoop is composed of four core components—Hadoop Common, Hadoop Distributed File System (HDFS), MapReduce and YARN.

Hadoop Common: A module containing the utilities that support the other Hadoop components.

HDFS: A file system that provides reliable data storage and access across all the nodes in a Hadoop cluster. It links together the file systems on many local nodes to create a single file system.

MapReduce: A framework for writing applications that process large amounts of structured and unstructured data in parallel across a cluster of thousands of machines, in a reliable, fault-tolerant manner.

Yet Another Resource Negotiator (YARN): The next-generation MapReduce, which assigns CPU, memory and storage to applications running on a Hadoop cluster. It enables application frameworks other than MapReduce to run on Hadoop, opening up a wealth of possibilities.

1.4.4 Master/NameNode and Slave/DataNode

The **NameNode** is the centerpiece of an HDFS file system. It keeps the directory tree of all files in the file system, and tracks where across the cluster the file data is kept. It does not store the data of these files itself.Client applications talk to the NameNode whenever they wish to locate a file, or when they want to add/copy/move/delete a file. The NameNode responds the successful requests by returning a list of relevant DataNode servers where the data lives. [8]

A **DataNode** stores data in the [HadoopFileSystem]. A functional filesystem has more than one DataNode, with data replicated across them.On startup, a DataNode connects to the NameNode; spinning until that service comes up. It then responds to requests from the NameNode for filesystem operations.Client applications can talk directly to a DataNode, once the NameNode has provided the location of the data. Similarly, MapReduce operations farmed out to TaskTrackerinstances near a DataNode, talk directly to the DataNode to access the files. TaskTracker instances can, indeed should, be deployed on the same servers that host DataNode instances, so that MapReduce operations are performed close to the data. [9]

An architecture of Hadoop framework has been provide in figure 1.4 where itself explains how NameNode and DataNodes are connect with each other.

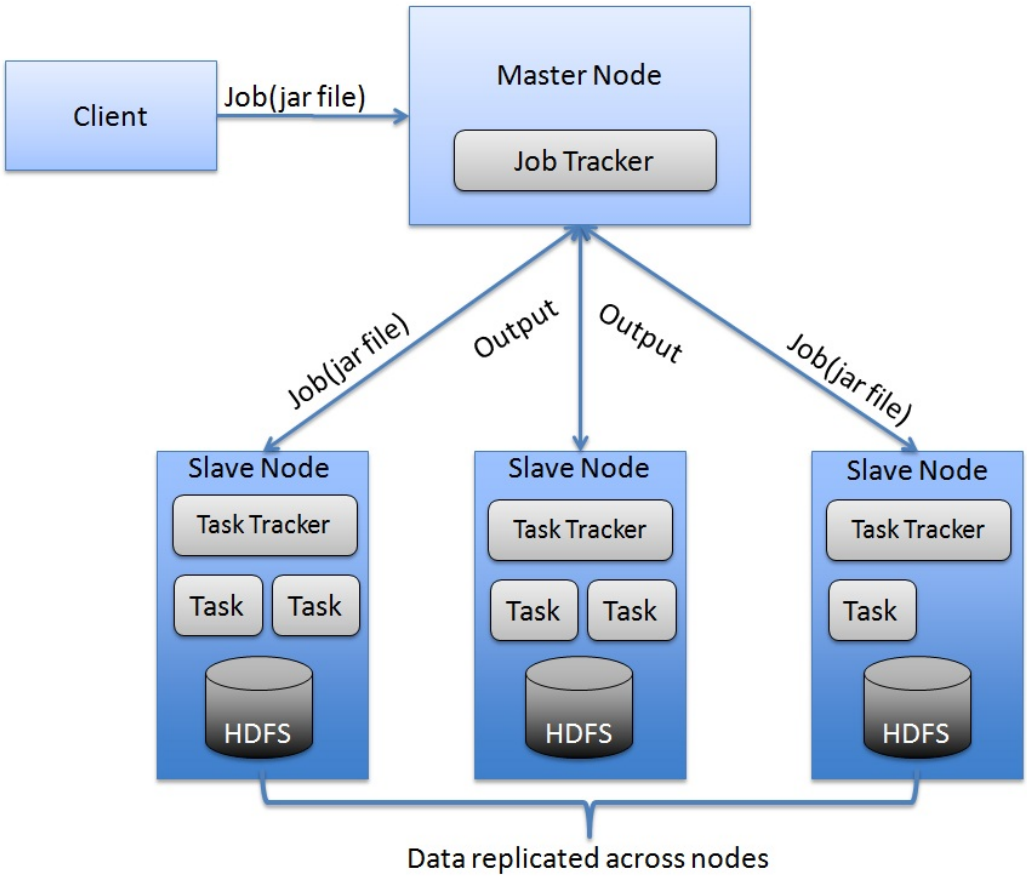


Figure 1.4: Architecture of Hadoop Framework. [10]

CHAPTER 2

Infrastructure Design

If we consider a small organization (profitable or non-profitable) we can find so many desktop and low-end servers which are utilize only 5% to 10%. So we can find such hardware and use them to design a Virtual Datacenter of distributed and parallel processing architecture, where we can process big amount of research data like structured, unstructured and semi-structured data. For this project we need Hypervisor software, Linux Operating System and Hadoop framework.

2.1Infrastructure Design

Let's assume a small organization have two remote site and they have 20 workstations in each site. Now design a virtual datacenter using these 40 workstations and additional two high-end server. User can use these workstations to perform their usual purpose and in background we can install hypervisor to create another virtual machine on each physical machine. Combining those virtual machine (Slave/DataNode) and two high-end machine (Master/NameNode) with Hadoop cluster and design avirtual datacenter infrastructure with distributed and parallel processing power.

First of all place a high-end server in each site which is called as cluster Master/NameNode and other 20 workstation will work as Slave/DataNode.Hadoop framework will be install on Master/NameNode. Need to install Hypervisor software in each workstation to create a virtual server and add the virtual server to the Hadoop cluster as Slave/DataNode.Same task need to complete on both site and finally configure Hadoop cluster services and other related services to Master and Slave nodes.

Please find the below infrastructure design in figure 2.1 where two remote site is connected with each other through privet network. Each site have LAN connection and all the workstations are connected to LAN. A high-end server has been placed to each site connected to LAN.

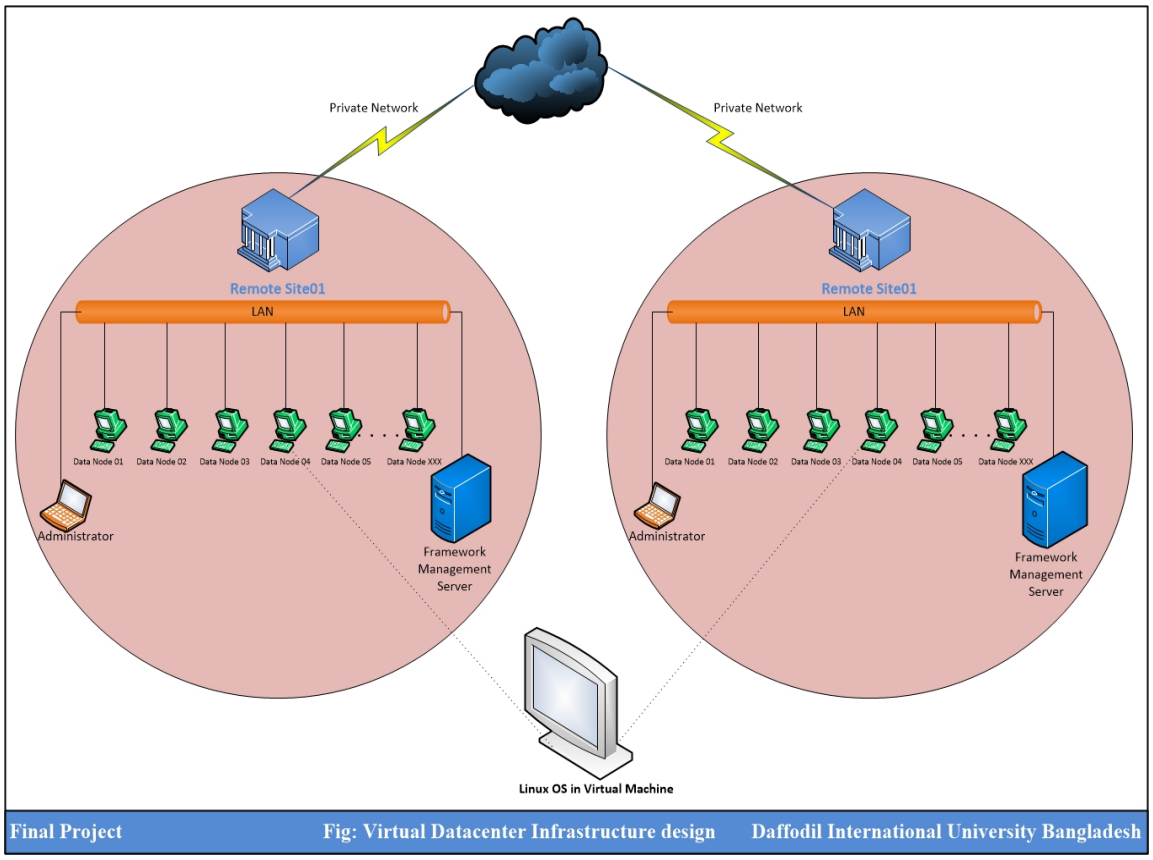


Figure 2.1: Infrastructure design of a Virtual Datacenter

2.2 Installation Steps Summery

To simulate theinfrastructure 64 bit Red Hat Linux6 has been used as operating system on Master and Slave servers in this project. Cloudera Manager CDH5.2.0 distribution has been used for Hadoop cluster framework and VMware has been used as hypervisor to prepare this virtual datacenter as a simulation. Please find the installation steps and command as below.

Prepare Master / NameNode

- Install Red Hat Enterprise Linux 64 bit operating system
- Configure IP
- Disable firewall
- Disable selinux
- Configure hosts file
- Configure NTP
- Configure SSH (Password less authentication)
- Install Java and other perquisite
- Install Postgresql Database

- Install Cloudera Manager CDH5.2.0 (Hadoop Framework)
- Login to Cloudera Manager console
- Configure cluster in Cloudera Manager
- Configure services as required

Prepare Slave / Data Node

- Install Red Hat Enterprise Linux 64 bit operating system
- Configure IP
- Disable firewall
- Disable selinux
- Configure hosts file
- Configure NTP
- Configure SSH (Password less authentication)
- Install Java and other perquisite
- Login to Cloudera Manager console
- Configure data node to the cluster
- Configure services as required

2.3Step by Step Installation

I. Configure IP Address (Master and Slave)

setup

vim /etc/sysconfig/network-scripts/ifcfg-eth0

ONBOOT=yes

service network restart

II. Stop Firewall (Master and Slave)

chkconfig --list iptables

chkconfig --level 0123456 iptables off

serviceiptables stop

serviceiptables status

III. Disableselinux (Master and Slave)

sestatus

vim /etc/sysconfig/selinux

SELINUX=disabled

Init 0

IV. Configure Hostname

vim /etc/sysconfig/network

NETWORKING=yes

HOSTNAME= XXXXXXXX

GATEWAY= XXX.XXX.XXX.XXX

vim /etc/hosts

IP<XXX.XXX.XXX.XXX> FQDN<XXXX.XXXXX.XXX>

V. Configure NTP Service (Master and Slave)

cd /etc

catntp.conf |more

vintp.conf |more

cd /etc/init.d

service ntpd status

service ntpd start

ntpq -p

chkconfig --list --level 345 |grepntp

chkconfigntpd on --level 345

chkconfig --list --level 345 |grepntpd

VI. Configure SSH for Password less login (Master and Slave)

login to slave01:

ssh-keygen -t rsa

cd /root/.ssh

ls

Login to master01:

ssh-keygen -t rsa

cd /root/.ssh

ls

cat id_rsa.pub >>authorized_keys

scpauthorized_keys root@slave01:/root/.ssh/

login to slave01:

cd /root/.ssh

ls

cat id_rsa.pub >>authorized_keys

scpauthorized_keys root@master01:/root/.ssh/

Test:

ssh slave01 (able to login without password)

ssh master02 (able to login without password)

VII. Install Postgresql8.4 Database (Master Server)

Create folder and copy jre and postgresql software to that folder:

chmod 777 jre-7u1-linux-x64.tar.gz

tar -zxvf jre-7u1-linux-x64.tar.gz

cd /

java -version

host to export java path: export PATH=$PATH:/soft/jre1.7.0_01/bin/

chmod 777 postgresql84-8.4.22-1PGDG.rhel6.x86_64.rpm

chmod 777 postgresql84-libs-8.4.22-1PGDG.rhel6.x86_64.rpm

chmod 777 postgresql84-server-8.4.22-1PGDG.rhel6.x86_64.rpm

rpm -Uvh postgresql84-libs-8.4.22-1PGDG.rhel6.x86_64.rpm

rpm -Uvh postgresql84-8.4.22-1PGDG.rhel6.x86_64.rpm

rpm -Uvh postgresql84-server-8.4.22-1PGDG.rhel6.x86_64.rpm

VIII. Check services and logs after installation (Master Server)

service postgresql-8.4 status

service postgresql-8.4 initdb

service postgresql-8.4 start

chkconfig --level 345 postgresql-8.4 on

chkconfig --list |greppostgre*

IX. Configure yum.conf file if proxy used for internet (Master and Slave)

vim /etc/yum.conf

proxy=http://XXX.XXX.XXX.XXX:XXXX

X. DownloadCloudera manager (Master Server)

cd /root

wget http://archive.cloudera.com/cm5/installer/latest/cloudera-manager-installer.bin

ls–l

XI. Install Cloudera Manager (Master Server)

Cd /root

ls -l

-rw-r--r-- 1 root root 501703 Jul 29 10:51 cloudera-manager-installer.bin

chmod 777 cloudera-manager-installer.bin

./cloudera-manager-installer.bin

Here from figure 2.2 to figure 2.5is showing the installation steps of cloudera. We just need to click Next or Yes to continue the installation process.

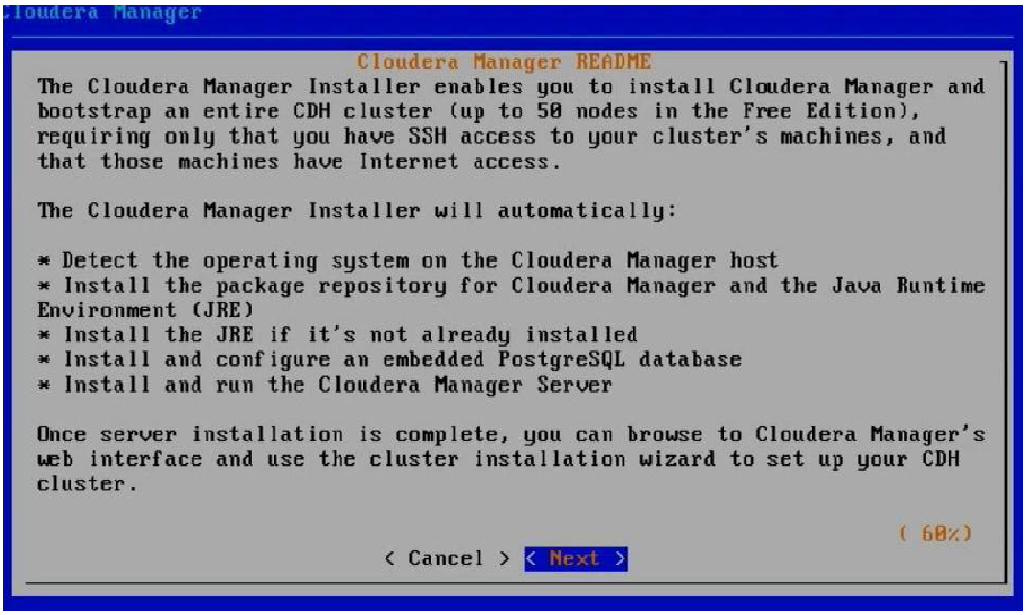


Figure 2.2: Cloudera Manager Installation screen

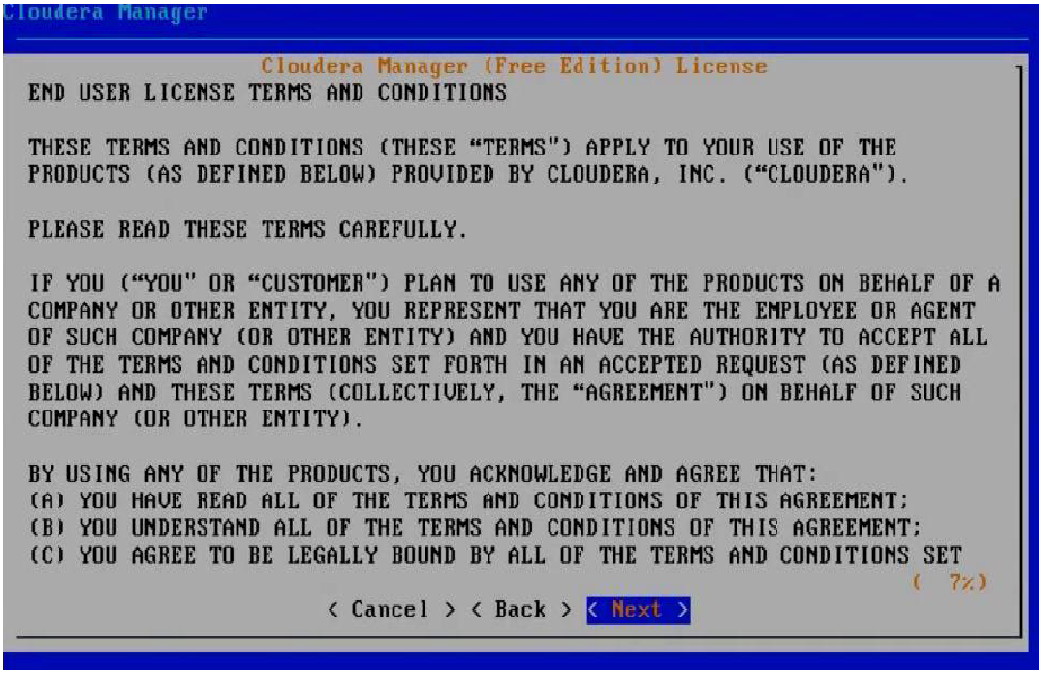


Figure 2.3: Cloudera Manager Licensing screen

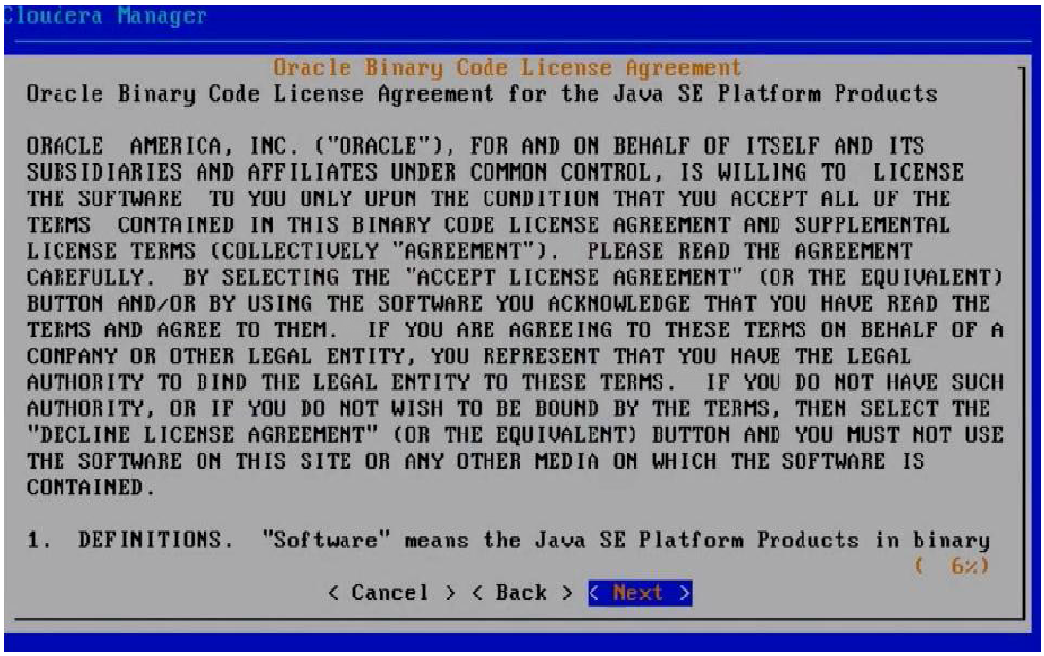


Figure 2.4: Java SE Licensing screen

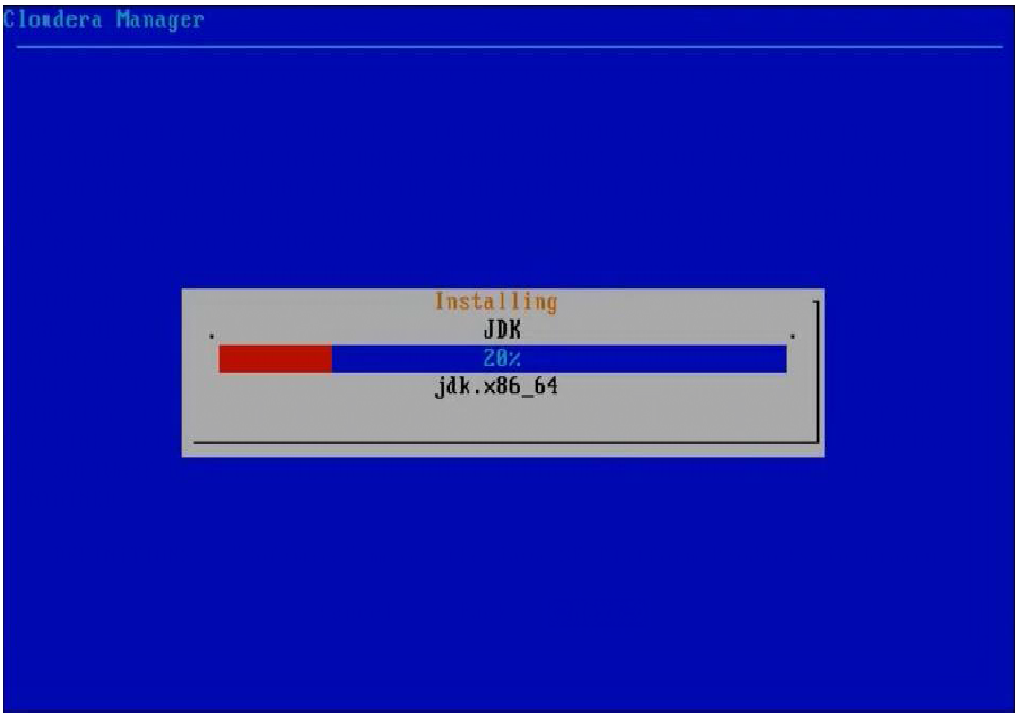


Figure 2.5: JDK Installation screen

After this step cloudera manager will be installed on the Master server and go the following link (http://XXX.XXX.XXX.XXX <IP>:7180/) to install and configure Hadoop cluster.

XII. Check services after installation (Master Server)

cd /var/log/cloudera-manager-installer

less 6.start-scm-server.log (check OK)

less 5.start-embedded-db.log (check OK)

chkconfig --level 345 cloudera-scm-server-db on

chkconfig --level 345 cloudera-scm-server on

XIII. Yum configuration (All Slave Servers)

cd /etc/yum.repos.d/

vim server_rhel6.repo

```
[local_mirror_server_rhel6]
name=patch_mirror_server_rhel6
baseurl=ftp://XXX.XXX.XXX.XXX/repo/rhel-x86_64-server-6/getPackage
enabled=1
gpgcheck=0
```

---

yum clean all

yumrepolist

yum list all

XIV. Install dependent services (All Slave Servers)

yum install cyrus-sasl-gssapi

yum install redhat-lsb

yum install portmap

servicerpcbind status

servicerpcbind start

XV. Configure Hadoop Cluster (Cloudera Manager Console)

Go the following link (http://XXX.XXX.XXX.XXX <IP>:7180/) to install and configure Hadoop cluster that is showing in the figure 2.6 Default User ID and Password is admin, admin

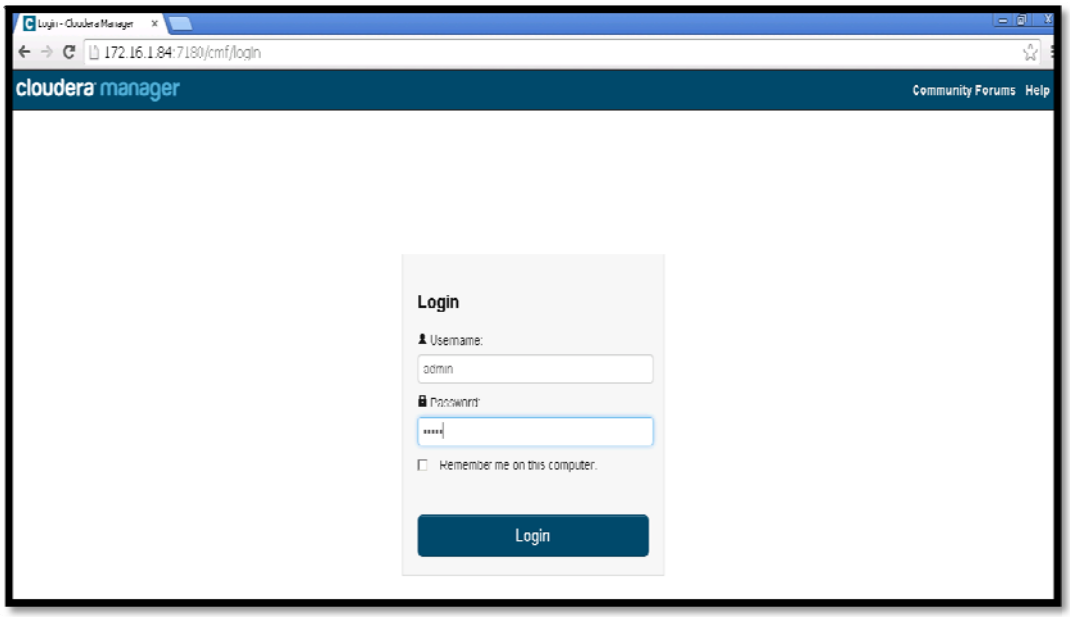


Figure 2.6: Cloudera Manager console to login

Figure 2.7 showing to click on just free installation to choose the Cloudera Manager version.

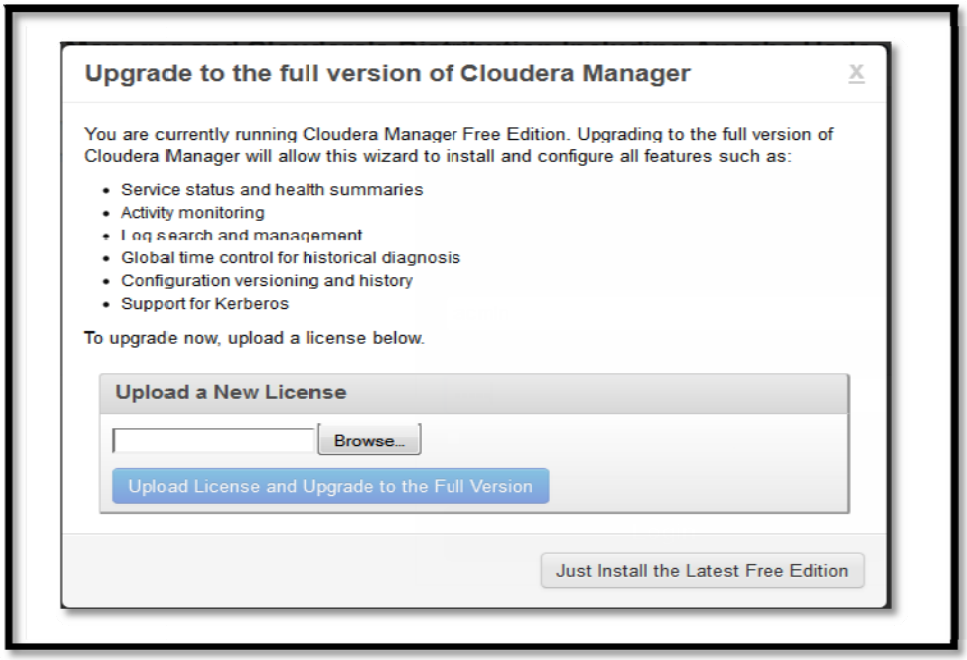


Figure 2.7: Cloudera Manager version selection

Figure 2.8 and figure 2.9 showing the how to add host to create a cluster on Cloudera manager. Add hostnames as defined in on /etc/hosts file of all servers.

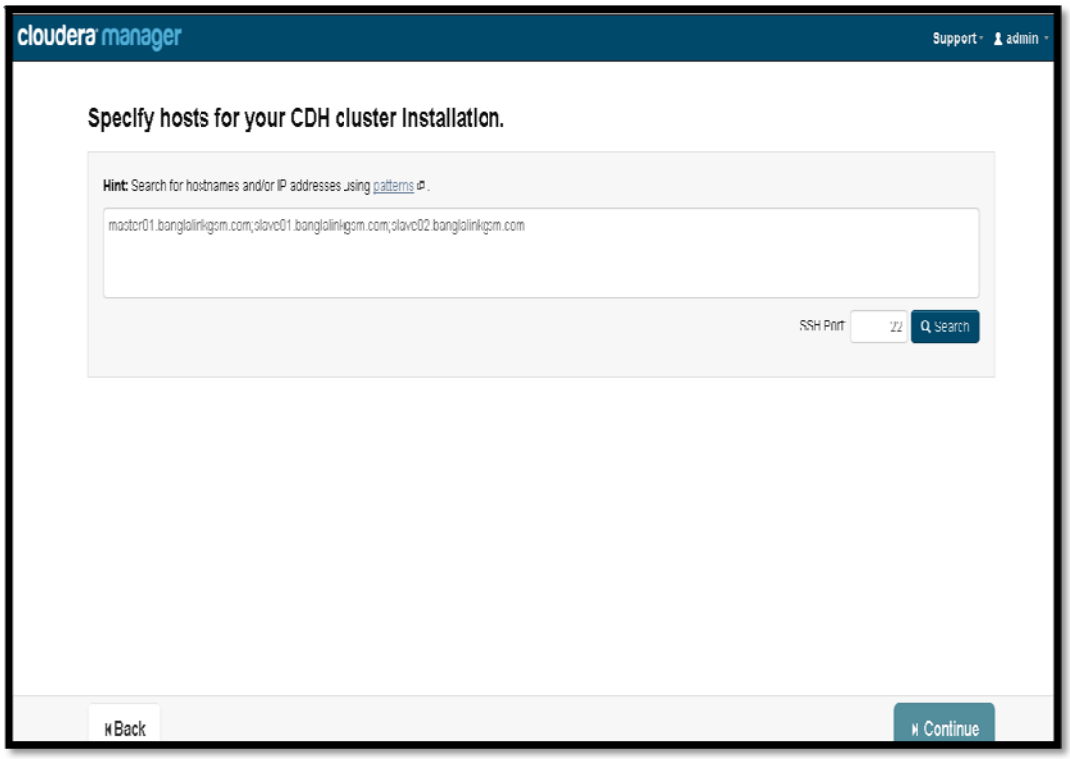


Figure 2.8: Cluster Installation

Figure 2.9 showing the hosts availability, now select all the available hosts by checking the text box against it.

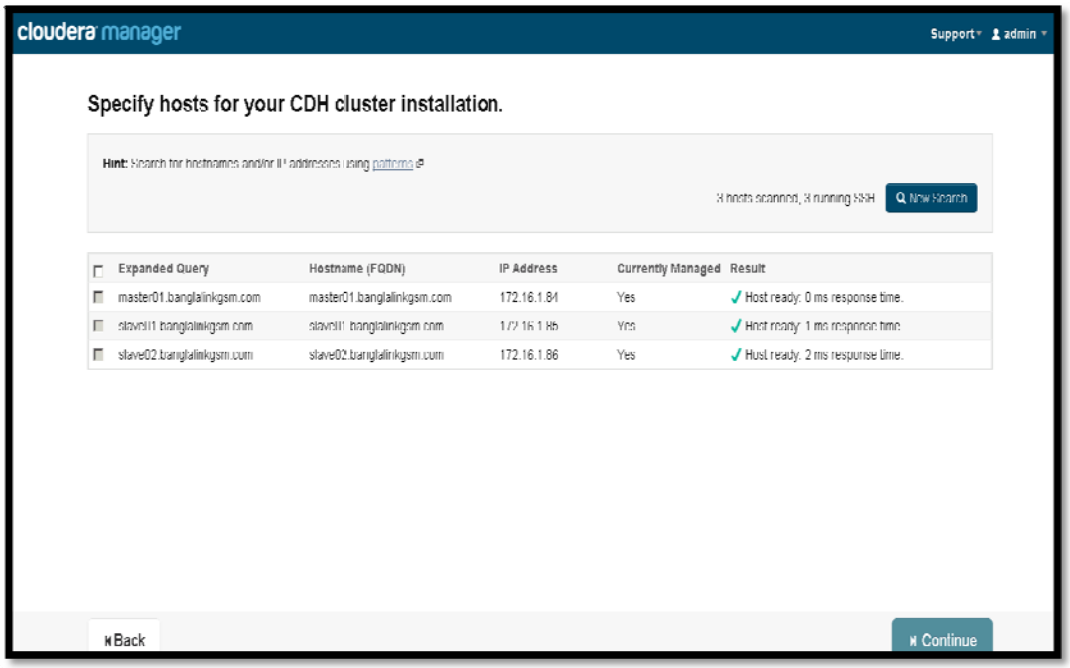


Figure 2.9: Specify host to install cluster

Figure 2.10 shows the CDH version selection where we need to click on CDH4 cloudera manager latest version of Hadoop.

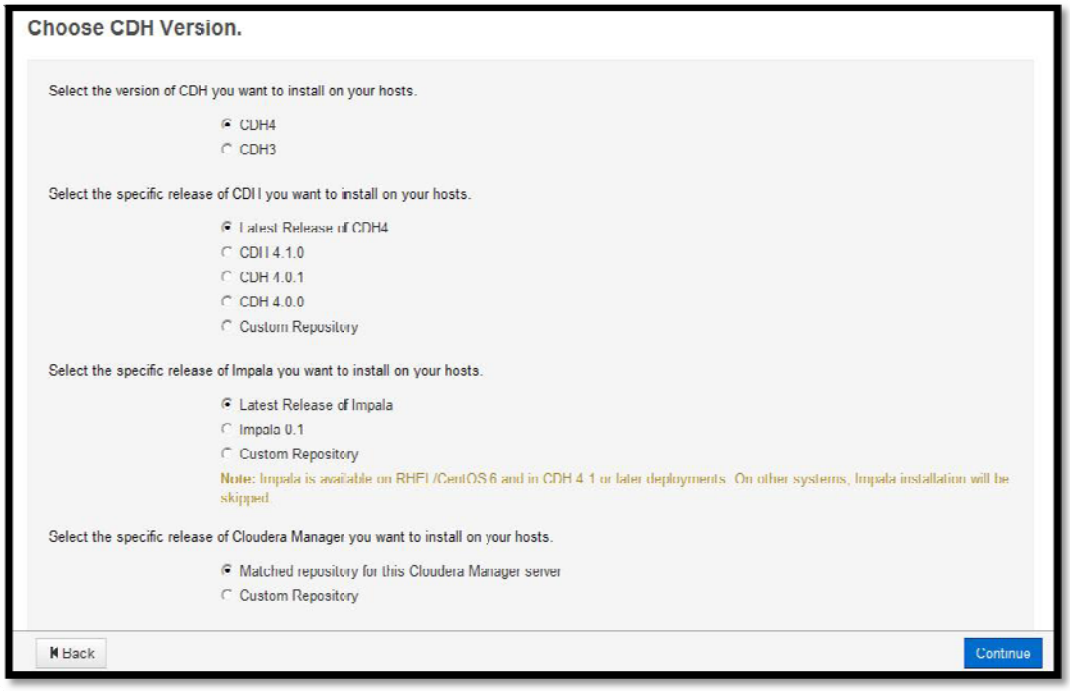


Figure 2.10: CDH version selection

Figure 2.11 shows the installation credential option where we need to click on start installation button to start the installation process on selected three nodes.

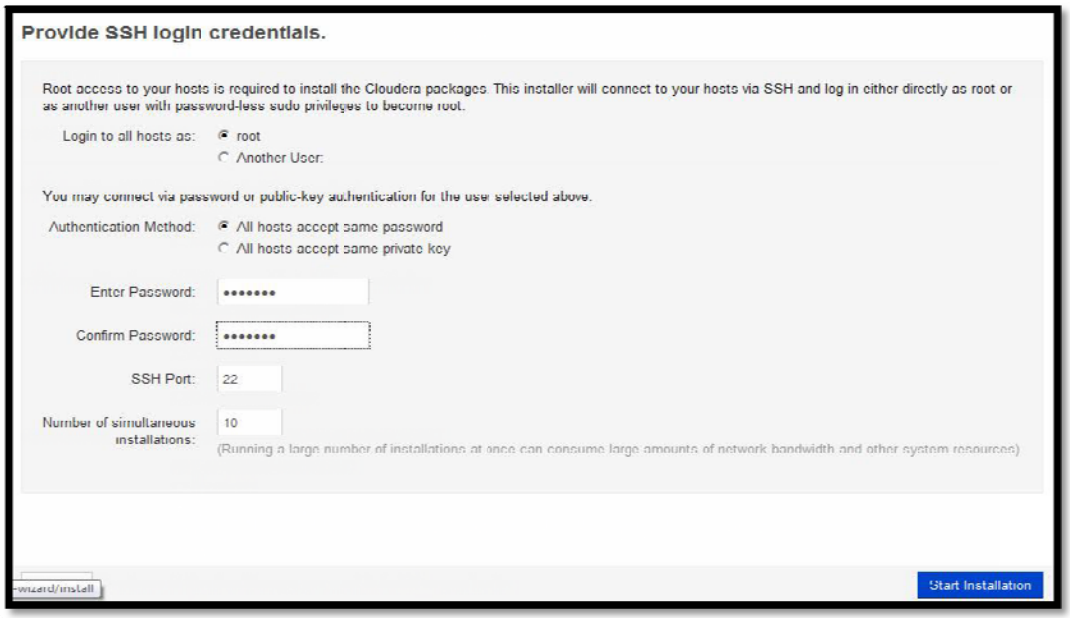


Figure 2.11: CDH installation credential option

Figure 2.12 shows this process is successfully completed click on continue button and that will run host inspector to check the host for correctness. After the inspector is finished click on continue

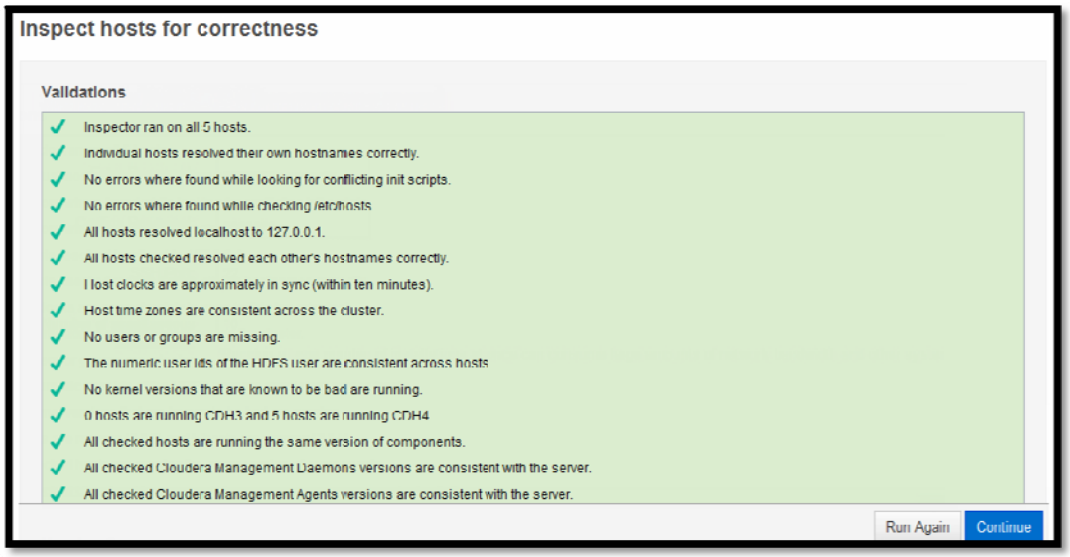


Figure 2.12: Inspect hosts for correctness

Figure 2.13 showing service we want to install on each host forCDH4

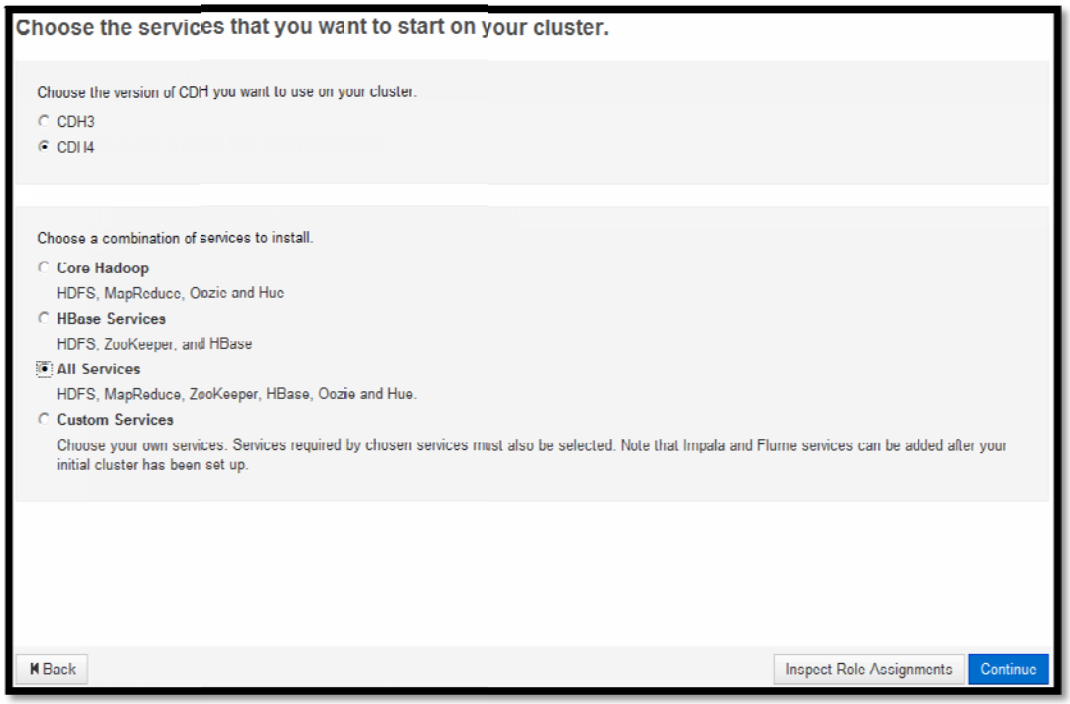


Figure 2.13: Services selection option

Figure 2.14 showing the cluster service installing steps where Cloudera manager is installing services on the nodes.

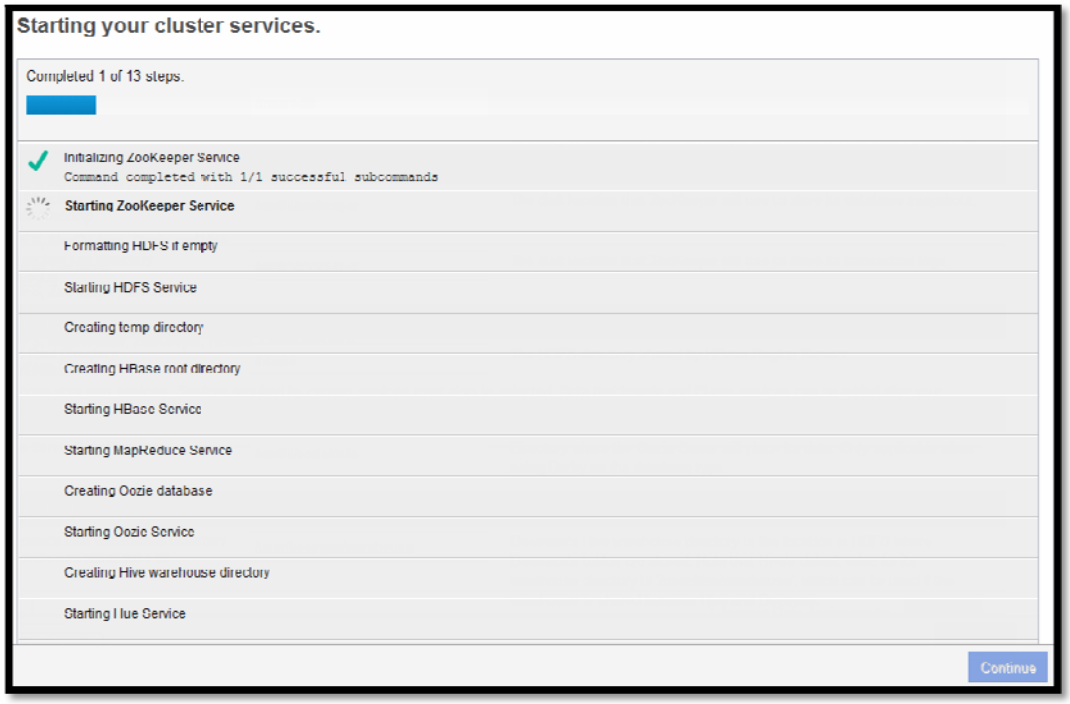


Figure 2.14: Cluster Service Installation

Figure 2.15 shows the Cloudera Manager Dashboard login in to the console through default id and password i.e. admin, admin respectively. Below screen shot shows various services of Hadoop running on the three nodes and managed by cloudera manager.

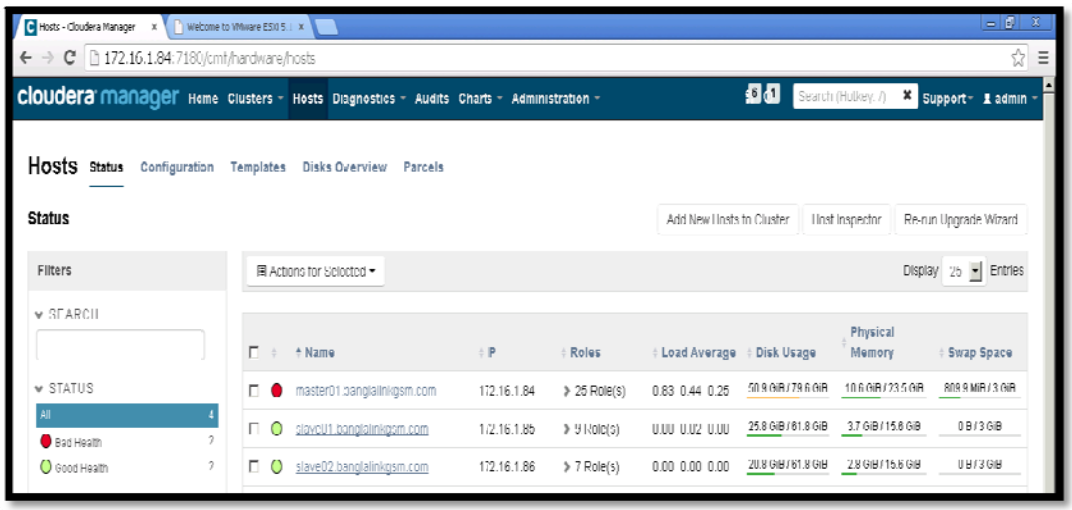


Figure 2.15: Cloudera Manager Dashboard

CHAPTER 3

Distributed Computing VS Single Computing

Here we tried to demonstrate a performance comparison by executing some queries between distributed parallel computing system and traditional single computing system. For the simulation of the infrastructure Hadoop cluster has been used for distributed parallel processing and Oracle 11g is used for traditional single processing system. We prepare three virtual host for Hadoop cluster and a high-end hardware for Oracle 11g.

In according to benchmarking standard for Logical database design, Scaling and Database population, Query selection to execute and compare the performance by following an standard benchmarking guideline document from "TPC BENCHMARK ™ DS" by-Transaction Processing Performance Council (TPC), Version 1.3.0, publish on November 2014.[11] [12]

3.1 Hardware Details

Oracle Database Server configuration:

System Model: HP Compaq 6000 Pro MT PC
System Type: x64-based PC
Processor: Intel(R) Core 2 Quad CPU @ 2.66GHz
OS: Red Hat Enterprise Linux 64 bit
RAM: 24 GB

Hadoop Cluster Server configuration:

Name Node / Master Server
System Type: VMware
Processor: 4 vCPU
OS: Red Hat Enterprise Linux 64 bit
RAM: 16 GB

Data Node / Slave Server
System Type: VMware
Processor: 1 vCPU
OS: Red Hat Enterprise Linux 64 bit
RAM: 8 GB

3.2 Performance Comparison

With the help of TPC guideline we design Database on both systems. Generate same set of data and table on each system. We uses TPC-DS provided Data Generator tools to generate benchmarking standard dataset.

Here figure 3.1 and figure 3.2 shows the same dataset on a table "**store_sales**" on both of the database (Hadoop and Oracle). Some queries will be execute on the table "**store_sales**" to compare performance of distributed parallel processing and single processing system.

```
[master01.banglalinkgsm.com:21000] > select
Query: select count(*) from store_sales
+----------+
| count(*) |
+----------+
| 27504814 |
+----------+
```

Figure 3.1: Hadoop Dataset

```
21:32:59 SQL> select count(*) from store_sales;

  COUNT(*)
----------
  27504814
```

Figure 3.2: Oracle dataset

Query Example One:

This query is collected from TPC benchmarking document query number Q3. This query report the total extended sales price per item brand of a specific manufacturer for all sales in a specific month of the year. Figure 3.3 and figure 3.4 shows the time consumed to run the query on both of the systems.

```
| 1999  | 5004001  | edu packscholar #1     | 23502.77 |
| 1999  | 9006009  | corpmaxi #9            | 23170.36 |
| 1999  | 4004001  | edu packedu pack #1    | 22654.02 |
| 1999  | 3004002  | edu packexporti #2     | 20383.52 |
| 1999  | 10008003 | namelessunivamalg #3   | 20245.31 |
| 1999  | 1001001  | amalgamalg #1          | 19220.68 |
| 1999  | 10009004 | maxiunivamalg #4       | 16249.42 |
| 1999  | 4003001  | exportiedu pack #1     | 13867.48 |
| 1999  | 1003001  | exportiamalg #1        | 10974.34 |
| 1999  | 1004002  | edu packamalg #2       | 10924.04 |
| 2000  | 5003001  | exportischolar #1      | 96501.86 |
| 2000  | 2004001  | importoedu pack #2     | 95701.84 |
+-------+----------+------------------------+----------
Returned 100 row(s) in 6.05s
[master01.banglalinkgsm.com:21000] >
```

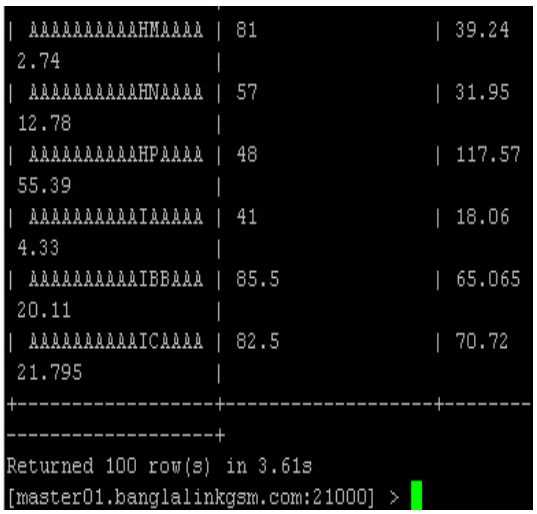
Figure 3.3: Q3 executed on Hadoop Cluster

```
    D_YEAR   BRAND_ID BRAND
---------- ---------- --------------------
  SUM_AGG
----------
      1999    1003001 exportiamalg #1
  10974.34

      1999    1004002 edu packamalg #2
  10924.04

      2000    5003001 exportischolar #1
  96501.86


99 rows selected.

Elapsed: 00:00:10.39
22:03:58 SQL>
```

Figure 3.4: Q3 executed on OracleDB

Query Example Two:

This query is collected from TPC benchmarking document query number Q7. This query report to compute the average quantity, list price, discount, and sales price for promotional items sold in stores where the promotion is not offered by mail or a special event. Restrict the results to a specific gender, marital and educational status. Figure 3.5 and figure 3.6 shows the elapsed time to process same query on same data.



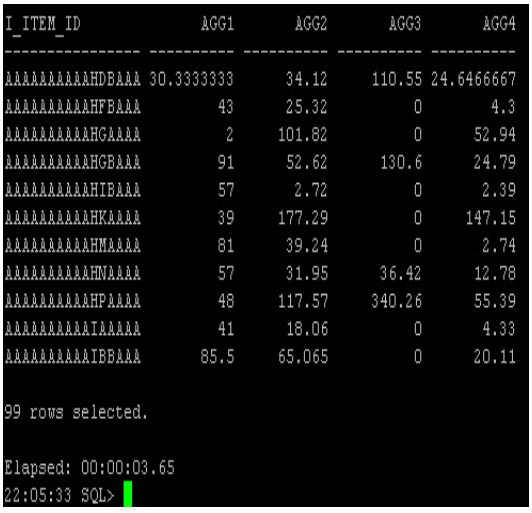
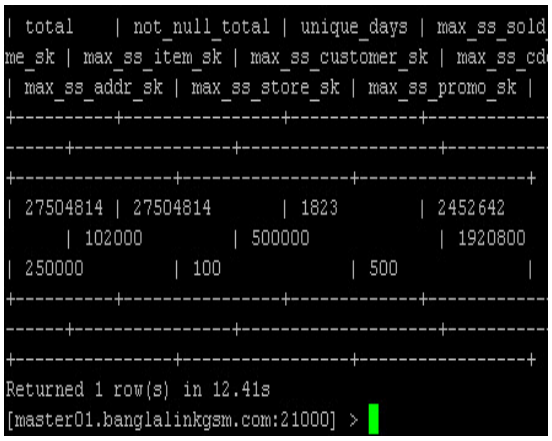


Figure 3.5: Q7 executed on Hadoop Cluster Figure 3.6: Q7 executed on Oracle DB

Query Example Three:

This query is collected from TPC benchmarking document which will stress the database system by accessing all table and will apply aggregation for in-depth analysis of the database. This type of query used to stress the database to test the performance. Figure 3.7 and figure 3.8 shows the stress test timing of the database.



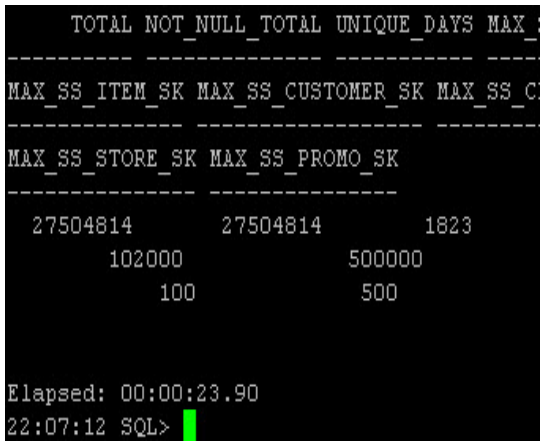


Figure 3.7: Query executed on Hadoop Cluster Figure 3.8: Query executed on Oracle

3.3 Statement

After analyzing above given examples and query execution time, it has been observed that Hadoop cluster provide better performance than Oracle because of distributed and parallel processing architecture. There are some reason why Hadoop is processing data faster than Oracle which are analyzed and discussed on next chapter.

CHAPTER 4

Performance and Benefit Analysis

4.1 Performance Analysis

According to the Performance Comparison chapter it can be undertake that distributed and parallel processing perform faster than single processing system. In this report it's been tried to explain technical reason why distributed and parallel computing system process big data faster than traditional single computing system.

Distributed Filesystem: In Hadoop filesystem data splits into 64MB block size to store and in traditionalfilesystem data splitsinto 4KB block size to store it.So, when any execution command process same amount of data Hadoop can process those data from a few blocks whereas any traditional system have to process same amount of data from a huge number of blocks. So processing huge amount of data is faster in distributed file system.[13] [14]

Parallel Processing: In Hadoop cluster jobs are also split and distribute among all of the hosts in cluster. It can execute all the jobs into different hosts and combined the result in a single point after execution. It doesn't require applications to shuttle huge volumes of data across your network. Whereas Oracle needs to run jobs in a single machine and shuttle huge volumes of data across your network. So processing huge amount of data is faster in parallel processing system.

4.2 Benefit Analysis

Main benefit of this project is cost efficiency. This project designed based on commodity hardware support and hardware virtualization concept which definitely save the huge costing to setup a virtual datacenter instead of setup a physical datacenter infrastructure. As it is a virtual datacenter it can reduce usages of electricity and can consume small space. Because of distributed and parallel processing it's ensure faster and redundant processing system as well. Again it's an open sourced framework which anyone can change the coding by themselves to configure it according to their infrastructure.

CHAPTER 5
Conclusion

Extensive data growth and verity of data is a big problem to store data, confirm redundancy and processing of big data. In this project it has been shown that how can we implement a virtual datacenter using commodity hardware to store data, ensure redundancy and perform distributed and parallel processing of different types of data from a single solution. There are lots of solution which can provide data store, redundancy, distributed and parallel processing separately where huge costing is involve to integrate those options. Now a day every organization wants to have their datacenter which is too costly to implement and maintain.

In this project a virtual datacenter prototype has been implement which will help any organization to implement datacenter using their existing commodity hardware to process huge amount of data. This will save huge amount of costing and will provide high performance.

Because of resource constraint and limited technical facility the simulation has been implemented in a small virtual lab and small amount of data. To get better performance and the beauty of the framework it's recommended to increase the host in cluster as much as we can and try to process huge amount and different types of data (at least more than 10TB of data).

APPENDIX

Query Example one (Hadoop):

```
select
dt.d_year,
item.i_brand_idbrand_id,
item.i_brand brand,
sum(ss_ext_sales_price) sum_agg
from
date_dimdt,
store_sales,
item
where
dt.d_date_sk = store_sales.ss_sold_date_sk
andstore_sales.ss_item_sk = item.i_item_sk
anditem.i_manufact_id = 436
anddt.d_moy = 12
 -- partition key filters
and (ss_sold_date_sk between 2451149 and 2451179
orss_sold_date_sk between 2451514 and 2451544
orss_sold_date_sk between 2451880 and 2451910
orss_sold_date_sk between 2452245 and 2452275
orss_sold_date_sk between 2452610 and 2452640)
group by
dt.d_year,
item.i_brand,
item.i_brand_id
order by
dt.d_year,
sum_aggdesc,
brand_id
limit 100;
```

Query Example one (Oracle):

```
select * from (
select
 dt.d_year,
 item.i_brand_id brand_id,
 item.i_brand brand,
 sum(ss_ext_sales_price) sum_agg
from
 date_dim dt,
 store_sales,
 item
where
 dt.d_date_sk = store_sales.ss_sold_date_sk
 and store_sales.ss_item_sk = item.i_item_sk
 and item.i_manufact_id = 436
 and dt.d_moy = 12
```

```
  -- partition key filters
  and (ss_sold_date_sk between 2451149 and 2451179
    or ss_sold_date_sk between 2451514 and 2451544
    or ss_sold_date_sk between 2451880 and 2451910
    or ss_sold_date_sk between 2452245 and 2452275
    or ss_sold_date_sk between 2452610 and 2452640)
group by
  dt.d_year,
  item.i_brand,
  item.i_brand_id
order by
  dt.d_year,
  sum_agg desc,
  brand_id)
where rownum<100;
```

Query Example Two (Hadoop):

```
select
i_item_id,
avg(ss_quantity) agg1,
avg(ss_list_price) agg2,
avg(ss_coupon_amt) agg3,
avg(ss_sales_price) agg4
from
store_sales,
customer_demographics,
date_dim,
item,
promotion
where
ss_sold_date_sk = d_date_sk
andss_item_sk = i_item_sk
andss_cdemo_sk = cd_demo_sk
andss_promo_sk = p_promo_sk
andcd_gender = 'F'
andcd_marital_status = 'W'
andcd_education_status = 'Primary'
and (p_channel_email = 'N'
orp_channel_event = 'N')
andd_year = 1998
andss_sold_date_sk between 2450815 and 2451179 -- partition key filter
group by
i_item_id
order by
i_item_id
limit 100;
```

Query Example Two (Oracle):

```
select * from (
select
  i_item_id,
  avg(ss_quantity) agg1,
  avg(ss_list_price) agg2,
  avg(ss_coupon_amt) agg3,
  avg(ss_sales_price) agg4
from
  store_sales,
  customer_demographics,
  date_dim,
  item,
  promotion
where
  ss_sold_date_sk = d_date_sk
  and ss_item_sk = i_item_sk
  and ss_cdemo_sk = cd_demo_sk
  and ss_promo_sk = p_promo_sk
  and cd_gender = 'F'
  and cd_marital_status = 'W'
  and cd_education_status = 'Primary'
  and (p_channel_email = 'N'
    or p_channel_event = 'N')
  and d_year = 1998
  and ss_sold_date_sk between 2450815 and 2451179 -- partition key filter
group by
  i_item_id
order by
  i_item_id)
where rownum<100;
```

Query Example Three (Hadoop):

```
select
count(*) as total,
count(ss_sold_date_sk) as not_null_total,
count(distinct ss_sold_date_sk) as unique_days,
max(ss_sold_date_sk) as max_ss_sold_date_sk,
max(ss_sold_time_sk) as max_ss_sold_time_sk,
max(ss_item_sk) as max_ss_item_sk,
max(ss_customer_sk) as max_ss_customer_sk,
max(ss_cdemo_sk) as max_ss_cdemo_sk,
max(ss_hdemo_sk) as max_ss_hdemo_sk,
max(ss_addr_sk) as max_ss_addr_sk,
max(ss_store_sk) as max_ss_store_sk,
max(ss_promo_sk) as max_ss_promo_sk
fromstore_sales;
```

Query Example Three (Oracle):

```
select
  count(*) as total,
  count(ss_sold_date_sk) as not_null_total,
  count(distinct ss_sold_date_sk) as unique_days,
  max(ss_sold_date_sk) as max_ss_sold_date_sk,
  max(ss_sold_time_sk) as max_ss_sold_time_sk,
  max(ss_item_sk) as max_ss_item_sk,
  max(ss_customer_sk) as max_ss_customer_sk,
  max(ss_cdemo_sk) as max_ss_cdemo_sk,
  max(ss_hdemo_sk) as max_ss_hdemo_sk,
  max(ss_addr_sk) as max_ss_addr_sk,
  max(ss_store_sk) as max_ss_store_sk,
  max(ss_promo_sk) as max_ss_promo_sk
from store_sales;
```

REFERENCES

[1] http://www.slashroot.in/what-is-object-storage

[2] http://blogs.informatica.com/perspectives/2013/04/01/is-the-data-explosion-impacting-you-how-do-you-compare-to-your-peers/#fbid=ZjEbHf6Ofbc

[3] http://searchcloudcomputing.techtarget.com/definition/big-data-Big-Data

[4] http://en.wikipedia.org/wiki/Big_data

[5] http://en.wikipedia.org/wiki/Exabyte

[6] http://en.wikipedia.org/wiki/Virtualization

[7] http://readwrite.com/2013/05/23/hadoop-what-it-is-and-how-it-works

[8] http://wiki.apache.org/hadoop/NameNode

[9] https://wiki.apache.org/hadoop/DataNode

[10] http://blog.raremile.com/hadoop-demystified/

[11] "TPC BENCHMARK ™ DS" by-Transaction Processing Performance Council (TPC), Version 1.3.0, publish on November 2014.

[12] The NoSQL Technical Comparison Report Cassandra (DataStax), MongoDB, and Couchbase Server. By Altoros September 2014

[13] Hadoop Operations –by Eric Sammer, Published by O'Reilly Media, 2012

[14] Hadoop The Definitive Guide –by Tom White, Published by O'Reilly Media, 2012