Name of the student:	Tanmay Prashant Rane	**Roll No.**	8031
Practical Numb er: 1	1	**Date of Practical:**	
Rele vant CO's: ITC802.2	**At the end of the course students will be able to use tools like hado op and NoSQL to solv e big data relate d problems.**		
Sign here to indicate that you hav e read all the rele vant material provide d befor e attempting this practical			**Sign:**

Practical grading using Rubrics

Indicator	**Very Poor**	**Poor**	**Average**	**Good**	**Excellent**
Timeline (2)	More than a session late (0)	NA (0.5)	NA(1)	NA (1.5)	Early or on time (2)
Completeness (3)	N/A	N/A	Not Completed (1)	Partially Completed (2)	Completed(3)
Legibility (3)	N/A	N/A	poor(1)	Good(2)	Very Good (3)
Postlab (2)	N/A	N/A	N/A	Partially Correct(1)	All correct answers (2)

Total Marks (10)	**Sign of instructor**

Practical

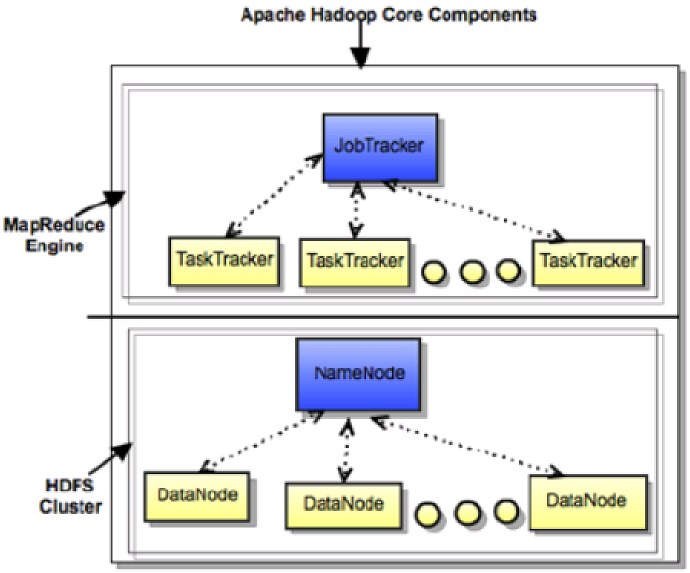
Course title: Big Data Analytics
Course term: 2019-2020

Problem Statement: Study of hado op installation, eclipse plugin configuration for hadoop and hado op ecosystem.

Theory:Write about components of hadoop

Apache Hadoop is a framework that allows for the distributed processing of large data sets across clusters of commodity computers using a simple programming model. It is designed to scale up from single servers to thousands of machines, each providing computation and storage. Rather than rely on hardware to deliver high-availability, the framework itself is designed to detect and handle failures at the application layer, thus delivering a highly-available service on top of a cluster of computers, each of which may be prone to failures.

HDFS (storage) and **MapReduce** (processing) are the two core components of Apache Hadoop. The most important aspect of Hadoop is that both HDFS and MapReduce are designed with each other in mind and each are co-deployed such that there is a single cluster and thus pro¬vides the ability to move computation to the data not the other way around. Thus, the storage system is not physically separate from a processing system.



Hadoop Distributed File System (HDFS)

HDFS is a distributed file system that provides high-throughput access to data. It provides a limited interface for managing the file system to allow it to scale and provide high throughput. HDFS creates multiple replicas of each data block and distributes them on computers throughout a cluster to enable reliable and rapid access.

The main components of HDFS are as described below:

NameNode is the master of the system. It maintains the name system (directories and files) and manages the blocks which are present on the DataNodes.

DataNodes are the slaves which are deployed on each machine and provide the actual stor¬age. They are responsible for serving read and write requests for the clients.

Secondary NameNode is responsible for performing periodic checkpoints. In the event of NameNode failure, you can restart the NameNode using the checkpoint.

MapReduce

MapReduce is a framework for performing distributed data processing using the MapReduce programming paradigm. In the MapReduce paradigm, each job has a user-defined map phase (which is a parallel, share-nothing processing of input; followed by a user-defined reduce phase where the output of the map phase is aggregated). Typically, HDFS is the storage system for both input and output of the MapReduce jobs.

The main components of MapReduce are as described below:

JobTracker is the master of the system which manages the jobs and resources in the clus¬ter (TaskTrackers). The JobTracker tries to schedule each map as close to the actual data being processed i.e. on the TaskTracker which is running on the same DataNode as the underlying block.

TaskTrackers are the slaves which are deployed on each machine. They are responsible for running the map and reduce tasks as instructed by the JobTracker.

JobHistoryServer is a daemon that serves historical information about completed applications. Typically, JobHistory server can be co-deployed with Job¬Tracker, but we recommend to run it as a separate daemon.

Explain the step by step procedure of hado op installation.

1) Create a hadoop user by using the following command

```
sudo adduser hadoop
```

2) Add hadoop user to sudoers by logging into root account

```
# sudo adduser hadoop sudo
```

3) Re-login as hadoop user

```
su hadoop
```

4) Configure SSH

a) Switch to sudo mode on hadoop user

```
su hadoop
```

b) Generate keys using ssh keygen

```
ssh-keygen -t rsa -P “”
```

c) Move the keys to $HOME/.ssh/authorized_keys

```
cat $HOME/.ssh/id_rsa >> $HOME/.ssh/authorized_keys
```

d) try ssh

```
ssh localhost
```

e) Download hadoop version

```
wget “https://downloads.apache.org/hadoop/common/hadoop-2.7.7/hadoop-2.7.7.tar.gz”
```

f) Extract the tar file

```
sudo tar -xvzf hadoop-2.7.7.tar.gz
```

g) Rename the extracted folder

```
sudo mv -r hadoop-2.7.7 hadoop
```

h) Give folder rights to hadoop user

```
sudo chown -R hadoop
```

5) Configure Hadoop

a) Modify ~/.bashrc file

add the following lines

```
#Set HADOOP_HOME
export HADOOP_HOME=<Installation Directory of Hadoop>
#Set JAVA_HOME
export JAVA_HOME=<Installation Directory of Java>
# Add bin/ directory of Hadoop to PATH
export PATH=$PATH:$HADOOP_HOME/bin
```

b) Save

```
. ~/.bashrc
```

c) Set JAVA_HOME inside $HADOOP_HOME/etc/hadoop/hadoop-env.sh

change export JAVA_HOME = ${JAVA_HOME}

to export JAVA_HOME = /usr/lib/jvm/default-java

d) There are two parameters in $HADOOP_HOME/etc/hadoop/core-site.xml which need to be set-

1. 'hadoop.tmp.dir' - Used to specify a directory which will be used by Hadoop to store its data files.

.

2. 'fs.default.name' - This specifies the default file system
To set these parameters, open core-site.xml

```
sudo gedit $HADOOP_HOME/etc/hadoop/core-site.xml
Copy below line in between tags <configuration></configuration>
<property><name>hadoop.tmp.dir</name>
<value>/app/hadoop/tmp</value>
<description>Parent directory for other temporary directories.</description>
</property><property><name>fs.defaultFS </name>
<value>hdfs://localhost:54310</value>
<description>The name of the default file system. </description></property>
```

e) Navigate to directory $HADOOP_HOME/etc/Hadoop

f) Create directory(<value></value> mentioned in core-site.xml

```
sudo mkdir -p /app/hadoop/tmp
```

g) Grant permission to the directory

```
sudo chown -R hadoop /app/hadoop/tmp
sudo chmod 750 /app/hadoop/tmp
```

h) Map Reduce Configuration

```
sudo gedit /etc/profile.d/hadoop.sh
//add this line
export HADOOP_HOME=/home/tanmay/Downloads/hadoop
//run this file
sudo chmod =x /etc/profile.d/hadoop.sh
//exit terminal and restart again.
```

i) Type echo $HADOOP_HOME. To verify the path

```
echo $HADOOP_HOME
```

j) Copy Files

```
sudo cp $HADOOP_HOME/etc/hadoop/mapred-site.xml.template $HADOOP_HOME/etc/hadoop/mapred-site.xml
```

k) Open mapred-site.xml

```
sudo gedit $HADOOP_HOME/etc/hadoop/mapred-site.xml
//add belows liines of setting in between tags <configuration> and </configuration>
<property>
<name>mapreduce.jobtracker.address</name>
<value>localhost:54311</value>
<description>MapReduce job tracker runs at this host and port.
</description>
</property>
```

l) Open $HADOOP_HOME/etc/hadoop/hdfs-site.xml

```
<property>
<name>dfs.replication</name>
<value>1</value>
<description>Default block replication.</description>
</property>
<property>
<name>dfs.datanode.data.dir</name>
<value>/home/hduser_/hdfs</value>
</property>
```

m) sudo mkdir -p /home/hadoop/hdfs
sudo chmod 750 /home/hadoop/hdfs

n) Format HDFS before starting Hadoop
HADOOP_HOME/sbin/start-dfs.sh

o) $HADOOP_HOME/sbin/start-yarn.sh

Explain the step by step procedure of installing and configuring eclipse plugin for hadoop.

a) Download hadoop installation by following URL:

```
-->wget “https://downloads.apache.org/hadoop/common/hadoop-2.7.7/hadoop-2.7.7.tar.gz”
-->tar -xvzf hadoop2.7.7.tar.gz
```

b) Download the eclipse 2020-03 version from eclipse website and install it giving it appropriate java path.

c) Download the eclipse plugin using the following link

-->https://drive.google.com/file/d/0B-ur4R5mlgGLaW9Cd2libWVfZFU/view

d) Put the jar file in plugins folder in eclipse installation directory(/home/tanmay/eclipse/jee-2020-03/eclipse/plugins)(in my case)

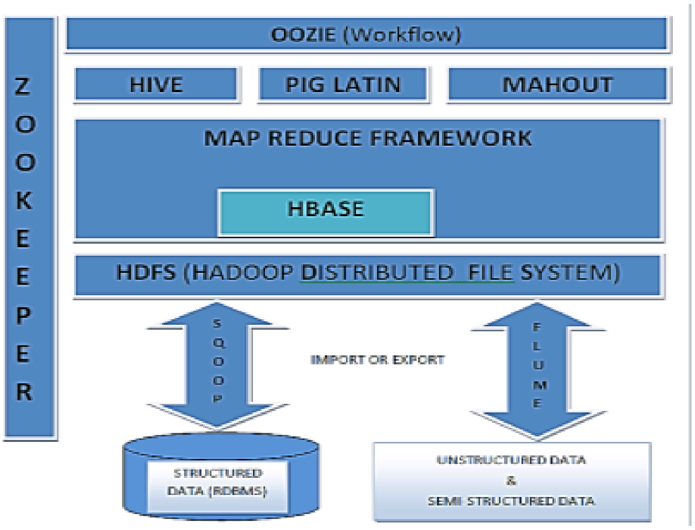
e) Start eclipse you can find the map-reduce perspective in perspective list. Click on it and the perspective will be successfully set.

f) Then you can successfully create a map-reduce project in add new project section, while creating the project click on configure hadoop installation directory and given the path of file that was extracted in step (a)

g) After this project can be run successfully

PostLab:Explain hadoop ecosystem in detail.

Hadoop is a framework which deals with Big Data but unlike any other frame work it's not a simple framework, it has its own family for processing different thing which is tied up in one umbrella called as Hadoop Ecosystem.



HDFS (Hadoop Distributed File System)

HDFS is a main component of Hadoop and a technique to store the data in distributed manner in order to compute fast. HDFS saves data in a block of 64MB(default) or 128 MB in size which is logical splitting of data in a Datanode (physical storage of data) in Hadoop cluster(formation of several Datanode which is a collection commodity hardware connected through single network). All information about data splits in data node known as metadata is captured in Namenode which is again a part of HDFS.

MapReduce Framework

It is another main component of Hadoop and a method of programming in a distributed data stored in a HDFS. We can write Map reduce program by using any language like JAVA, C++ PIPEs, PYTHON, RUBY etc. By name only Map Reduce gives its functionality Map will do mapping of logic into data (distributed in HDFS) and once computation is over reducer will collect the result of Map to generate final output result of MapReduce. MapReduce Program can be applied to any type of data whether Structured or Unstructured stored in HDFS. Example - word count using MapReduce.

HBASE

Hadoop Database or HBASE is a non-relational (NoSQL) database that runs on top of HDFS. HBASE was created for large table which have billions of rows and millions of columns with fault tolerance capability and horizontal scalability and based on Google Big Table. Hadoop can perform only batch processing, and data will be accessed only in a sequential manner for random access of huge data HBASE is used.

Hive

Many programmers and analyst are more comfortable with Structured Query Language than Java or any other programming language for which Hive is created by Facebook and later donated to Apache foundation. Hive mainly deals with structured data which is stored in HDFS with a Query Language similar to SQL and known as HQL (Hive Query Language). Hive also run Map reduce program in a backend to process data in HDFS but here programmer has not worry about that backend MapReduce job it will look similar to SQL and result will be displayed on console.

Pig

Similar to HIVE, PIG also deals with structured data using PIG LATIN language. PIG was originally developed at Yahoo to answer similar need to HIVE. It is an alternative provided to programmer who loves scripting and don't want to use Java/Python or SQL to process data. A Pig Latin program is made up of a series of operations, or transformations, that are applied to the input data which runs MapReduce program in backend to produce output

Mahout

Mahout is an open source machine learning library from Apache written in java. The algorithms it implements fall under the broad umbrella of machine learning or collective intelligence. This can mean many things, but at the moment for Mahout it means primarily recommender engines (collaborative filtering), clustering, and classification. Mahout aims to be the machine learning tool of choice when the collection of data to be processed is very large, perhaps far too large for a single machine. In its current incarnation, these scalable machine learning implementations in Mahout are written in Java, and some portions are built upon Apache's Hadoop distributed computation project.

Oozie

It is a workflow scheduler system to manage hadoop jobs. It is a server-based Workflow Engine specialized in running workflow jobs with actions that run Hadoop MapReduce and Pig jobs. Oozie is implemented as a Java Web-Application that runs in a Java Servlet-Container. Hadoop basically deals with bigdata and when some programmer wants to run many job in a sequential manner like output of job A will be input to Job B and similarly output of job B is input to job C and final output will be output of job C. To automate this sequence we need a workflow and to execute same we need engine for which OOZIE is used.

Zookeeper

Writing distributed applications is difficult because of partial failure may occur between nodes to overcome this Apache Zookeper has been developed by maintaining an open-source server which enables highly reliable distributed coordination. ZooKeeper is a centralized service for maintaining configuration information, naming, providing distributed synchronization, and providing group services. In case of any partial failure clients can connect to any node and be assured that they will receive the correct, up-to-date information.