Task – 1 : Setup a single-node Hadoop cluster and practice basic HDFS commands.

- Install Oracle V-Box → Link: https://www.virtualbox.org/wiki/Downloads
- Install Linux OS in V-Box. Ubuntu → https://releases.ubuntu.com/18.04/
- Install JAVA:
  - sudo apt-get update
  - sudo apt install default-jdk
  - Check: java -version
- Download Hadoop.
  - wget https://archive.apache.org/dist/hadoop/core/hadoop-2.7.3/hadoop-2.7.3.tar.gz
  - tar -xvf hadoop-2.7.3.tar.gz
- Add the Hadoop and Java paths in the bash file (.bashrc).
  - **Command**: vi .bashrc or sudo nano .bashrc

```
#Hadoop Environmnet
#Hadoop Related Options
export HADOOP_HOME=/home/ubuntu/hadoop-2.7.3
export HADOOP_CONF_DIR=$HADOOP_HOME/etc/hadoop
export HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export YARN_HOME=$HADOOP_HOME
export PATH=$PATH:$HADOOP_HOME/sbin:$HADOOP_HOME/bin

# Set Java home
export JAVA_HOME=/home/ubuntu/jdk1.8.0_361
export PATH=$JAVA_HOME/bin:$PATH
```

  - Save the bash file and close.
  - Apply the changes: **Command**: source .bashrc
  - Check whether Hadoop properly installed or not. **Command**: hadoop version

```
ubuntu@ubuntu:~$ hadoop version
Hadoop 2.7.3
Subversion https://git-wip-us.apache.org/repos/asf/hadoop.git -r baa91f7c6bc9cb
92be5982de4719c1c8af91ccff
Compiled by root on 2016-08-18T01:41Z
Compiled with protoc 2.5.0
From source with checksum 2e4ce5f957ea4db193bce3734ff29ff4
This command was run using /home/ubuntu/hadoop-2.7.3/share/hadoop/common/hadoop
-common-2.7.3.jar
ubuntu@ubuntu:~$
```

- Edit the Hadoop Configuration files.
  - All the Hadoop configuration files are located in hadoop-2.7.3/etc/hadoop directory.
  - **Command**: cd hadoop-2.7.3/etc/hadoop/
  - **Command**: ls

```
ubuntu@ubuntu:~$ cd hadoop-2.7.3/etc/hadoop
ubuntu@ubuntu:~/hadoop-2.7.3/etc/hadoop$ ls
capacity-scheduler.xml      kms-env.sh
configuration.xsl           kms-log4j.properties
container-executor.cfg      kms-site.xml
core-site.xml               log4j.properties
hadoop-env.cmd              mapred-env.cmd
hadoop-env.sh               mapred-env.sh
hadoop-metrics2.properties  mapred-queues.xml.template
hadoop-metrics.properties   mapred-site.xml
hadoop-policy.xml           mapred-site.xml.template
hadoop.sh                   slaves
hdfs-site.xml               ssl-client.xml.example
httpfs-env.sh               ssl-server.xml.example
httpfs-log4j.properties     yarn-env.cmd
httpfs-signature.secret     yarn-env.sh
httpfs-site.xml             yarn-site.xml
kms-acls.xml
```

- Open **core-site.xml** and edit the property mentioned below inside configuration tag.
    - core-site.xml informs Hadoop daemon where NameNode runs in the cluster. It contains configuration settings of Hadoop core such as I/O settings that are common to HDFS & MapReduce.
    - **Command**: vi core-site.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<configuration>
<property>
<name>fs.default.name</name>
<value>hdfs://localhost:9000</value>
</property>
</configuration>
```

- Edit **hdfs-site.xml** and edit the property mentioned below inside configuration tag:
    - hdfs-site.xml contains configuration settings of HDFS daemons (i.e. NameNode, DataNode, Secondary NameNode). It also includes the replication factor and block size of HDFS.
    - dfs.permissions is the property which can be used to enable/disable HDFS ACL (simple permission). With dfs.permission set to false any user can create/delete files/directories anywhere in HDFS. With this property set to false, You can access all other Hadoop services such as Hive, Pig etc.
    - **Command**: vi hdfs-site.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<configuration>
<property>
<name>dfs.replication</name>
<value>1</value>
</property>
<property>
<name>dfs.permission</name>
<value>false</value>
</property>
</configuration>
```

- Edit the **mapred-site.xml** file and edit the property mentioned below inside configuration tag:
  - mapred-site.xml contains configuration settings of MapReduce application like number of JVM that can run in parallel, the size of the mapper and the reducer process, CPU cores available for a process, etc.
  - In some cases, mapred-site.xml file is not available. So, we have to create the mapred-site.xml file using mapred-site.xml template.
  - **Command**: cp mapred-site.xml.template mapred-site.xml
  - **Command**: vi mapred-site.xml.

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<configuration>
<property>
<name>mapreduce.framework.name</name>
<value>yarn</value>
</property>
</configuration>
```

- Edit ***yarn-site.xml*** and edit the property mentioned below inside configuration tag:
  - *yarn-site.xml* contains configuration settings of ResourceManager and NodeManager like application memory management size, the operation needed on program & algorithm, etc.
  - You can even check out the details of Big Data with the Azure Data Engineering Certification in Hyderabad.
  - ***Command***: vi yarn-site.xml

```
<?xml version="1.0">
<configuration>
<property>
<name>yarn.nodemanager.aux-services</name>
<value>mapreduce_shuffle</value>
</property>
<property>
<name>yarn.nodemanager.auxservices.mapreduce.shuffle.class</name>
<value>org.apache.hadoop.mapred.ShuffleHandler</value>
</property>
</configuration>
```

- Edit ***hadoop-env.sh*** and add the Java Path as mentioned below:
  - *hadoop-env.sh* contains the environment variables that are used in the script to run Hadoop like Java home path, etc.
  - ***Command***: vi hadoop–env.sh

```
#This is java path
export JAVA_PATH=/home/ubuntu/jdk1.8.0_361
```

- *Go to Hadoop home directory and format the NameNode.*
  - **Command***: cd*
  - **Command***: cd hadoop-2.7.3*
  - **Command***: bin/hadoop namenode -format*

- *This formats the HDFS via NameNode. This command is only executed for the first time. Formatting the file system means initializing the directory specified by the dfs.name.dir variable.*
- *Never format, up and running Hadoop filesystem. You will lose all your data stored in the HDFS.*

- *Once the NameNode is formatted, go to hadoop-2.7.3/sbin directory and start all the daemons.*
  - Command: *cd hadoop-2.7.3/sbin*
  - *Either you can start all daemons with a single command or do it individually.*
  - **Command:** *./start-all.sh*
  - *The above command is a combination of* **start-dfs.sh, start-yarn.sh** *&* **mr-jobhistory-daemon.sh**
- *Start NameNode:*
  - *The NameNode is the centerpiece of an HDFS file system. It keeps the directory tree of all files stored in the HDFS and tracks all the file stored across the cluster.*
  - ***Command****: ./hadoop-daemon.sh start namenode*
- *Start DataNode:*
  - *On startup, a DataNode connects to the Namenode and it responds to the requests from the Namenode for different operations.*
  - ***Command****: ./hadoop-daemon.sh start datanode*
- ***Start ResourceManager:***
  - *ResourceManager is the master that arbitrates all the available cluster resources and thus helps in managing the distributed applications running on the YARN system. Its work is to manage each NodeManagers and the each application's ApplicationMaster.*
  - **Command**: *./yarn-daemon.sh start resourcemanager*
- ***Start NodeManager:***
  - *The NodeManager in each machine framework is the agent which is responsible for managing containers, monitoring their resource usage and reporting the same to the ResourceManager.*
  - **Command**: *./yarn-daemon.sh start nodemanager*
- ***Start JobHistoryServer:***
  - *JobHistoryServer is responsible for servicing all job history related requests from client.*
  - **Command*****:*** *./mr-jobhistory-daemon.sh start historyserver*
- *To check that all the Hadoop services are up and running, run the below command.*
  - **Command**: *jps*
- *Now open the Mozilla browser and go to* ***localhost****:****50070/dfshealth.html*** *to check the NameNode interface.*