**Step :**

1. OS
2. JAVA Installation. <https://phoenixnap.com/kb/how-to-install-java-centos-8>

$sudo yum install java-1.8.0-openjdk-devel

Set JAVA HOME :

$nano /etc/profile

export JAVA\_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.312.b07-2.el8\_5.x86\_64/jre

export PATH=$JAVA\_HOME/bin:$PATH

$source /etc/profile

1. Download Hadoop File : wget <https://archive.apache.org/dist/hadoop/core/hadoop-2.7.2/hadoop-2.7.2.tar.gz>
2. Extract and Locate

$tar –xf hadoop-2.7.2.tar.gz

$mv hadoop-2.7.2 /usr/local/hadoop

1. Configure Classpath for Global Variable Access

===

$sudo nano .bashrc

export JAVA\_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.312.b07-2.el8\_5.x86\_64/jre

export HADOOP\_HOME=/usr/local/hadoop

export PATH=$PATH:$HADOOP\_HOME/bin

export PATH=$PATH:$HADOOP\_HOME/sbin

export HADOOP\_MAPRED\_HOME=$HADOOP\_HOME

export HADOOP\_COMMON\_HOME=$HADOOP\_HOME

export HADOOP\_HDFS\_HOME=$HADOOP\_HOME

export YARN\_HOME=$HADOOP\_HOME

export HADOOP\_COMMON\_LIB\_NATIVE\_DIR=$HADOOP\_HOME/lib/native

export HADOOP\_OPTS="-Djava.library.path=$HADOOP\_HOME/lib"

export PDSH\_RCMD\_TYPE=ssh

export HADOOP\_CONF\_DIR=/usr/local/hadoop/etc/hadoop

export HADOOP\_OPTS="$HADOOP\_OPTS -Djava.library.path=$HADOOP\_HOME/lib/native"

===

$exec bash

$sudo chown -R centos:centos /usr/local/hadoop

1. Update Hadoop Configuration :[/usr/local/hadoop/etc/hadoop]
   1. hadoop-env.sh

$vi hadoop-env.sh

export JAVA\_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.312.b07-2.el8\_5.x86\_64/jre

export HADOOP\_LOG\_DIR=/var/log/hadoop/

2. Create log dir

$chown

$ sudo chown -R centos:centos /var/log/hadoop/

3. Create Password-less communication

$ssh-keygen

4. $vi /usr/local/hadoop/etc/hadoop/masters

5. $vi /usr/local/hadoop/etc/hadoop/slaves

7. Update core-site.xml

$vi /usr/local/hadoop/etc/hadoop/core-site.xml

<property>

<name>fs.defaultFS</name>

<value>hdfs://node1:9000</value>

</property>

$mkdir -p /usr/local/hadoop/data/hdfs/namenode

1. Update Hdfs-site.xml

$vi /usr/local/hadoop/etc/hadoop/hdfs-site.xml

<property>

<name>dfs.replication</name>

<value>3</value>

</property>

<property>

<name>dfs.namenode.name.dir</name>

<value>file:///usr/local/hadoop/data/hdfs/namenode</value>

</property>

<property>

<name>dfs.datanode.name.dir</name>

<value>file:///usr/local/hadoop/data/hdfs/datanode</value>

</property>

$mkdir -p /usr/local/hadoop/data/hdfs/datanode

1. Update yarn-site.xml

$vi /usr/local/hadoop/etc/hadoop/yarn-site.xml

<!-- Site specific YARN configuration properties -->

I

1. Update Mapred-site.xml

$cp /usr/local/hadoop/etc/hadoop/mapred-site.xml.template /usr/local/hadoop/etc/hadoop/mapred-site.xml

$usr/local/hadoop/etc/hadoop/mapred-site.xml

<property>

<name>mapreduce.jobtracker.address</name>

<value>node1:54311</value>

</property>

<property>

<name>mapreduce.framework.name</name>

<value>yarn</value>

</property>

$sudo chown -R centos:centos $HADOOP\_HOME

1. Format Namenode : $hdfs namenode -format
2. Start cluster : $start-all.sh
3. Check status : $jps

=========

WordCount

$cat >WordCount.java

package org.myorg;

import java.io.IOException;

import java.util.\*;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.conf.\*;

import org.apache.hadoop.io.\*;

import org.apache.hadoop.mapred.\*;

import org.apache.hadoop.util.\*;

public class WordCount {

  public static class Map extends MapReduceBase implements Mapper<LongWritable, Text, Text, IntWritable> {

    private final static IntWritable one = new IntWritable(1);

    private Text word = new Text();

    public void map(LongWritable key, Text value, OutputCollector<Text, IntWritable> output, Reporter reporter) throws IOException {

      String line = value.toString();

      StringTokenizer tokenizer = new StringTokenizer(line);

      while (tokenizer.hasMoreTokens()) {

        word.set(tokenizer.nextToken());

        output.collect(word, one);

      }

    }

  }

  public static class Reduce extends MapReduceBase implements Reducer<Text, IntWritable, Text, IntWritable> {

    public void reduce(Text key, Iterator<IntWritable> values, OutputCollector<Text, IntWritable> output, Reporter reporter) throws IOException {

      int sum = 0;

      while (values.hasNext()) {

        sum += values.next().get();

      }

      output.collect(key, new IntWritable(sum));

    }

  }

  public static void main(String[] args) throws Exception {

    JobConf conf = new JobConf(WordCount.class);

    conf.setJobName("wordcount");

mkdir wordcount\_classes

    conf.setOutputKeyClass(Text.class);

    conf.setOutputValueClass(IntWritable.class);

    conf.setMapperClass(Map.class);

    conf.setCombinerClass(Reduce.class);

    conf.setReducerClass(Reduce.class);

    conf.setInputFormat(TextInputFormat.class);

    conf.setOutputFormat(TextOutputFormat.class);

    FileInputFormat.setInputPaths(conf, new Path(args[0]));

    FileOutputFormat.setOutputPath(conf, new Path(args[1]));

    JobClient.runJob(conf);

  }

}

// ctrl + D to save program

$mkdir wordcount\_classes

$javac -cp $(hadoop classpath) -d wordcount\_classes/ WordCount.java

$jar -cvf wordcount.jar -C wordcount\_classes/ .

$hdfs dfs -mkdir -p /map/wordcount/input /map/wordcount/output

$cat > testdata.txt

$hd$hadoop jar wordcount.jar  org.myorg.WordCount /map/wordcount/input/testdata.txt /map/wordcount//output/map\_output

$hdfs dfs -ls /map/wordcount//output/map\_output

==========

Hive Installation

<https://archive.apache.org/dist/hive/hive-2.0.0/apache-hive-2.0.0-bin.tar.gz>

$tar -xf apache-hive-2.0.0-bin.tar.gz

$sudo mv apache-hive-2.0.0-bin /usr/local/hive

$ vi .bashrc

export HIVE\_HOME=/usr/local/hive

export PATH=$PATH:$HIVE\_HOME/bin

$bash

$cd $HIVE\_HOME/conf

$cp hive-env.sh.template hive-env.sh

$vi $HIVE\_HOME/conf/hive-env.sh

export HADOOP\_HOME=/usr/local/hadoop

$hdfs dfs -mkdir -p /user/hive/warehouse

$hive

$hdfs dfs -mkdir -p /tmp

$hdfs dfs -chmod -R 777 /tmp

$hive

$mv metastore\_db metastore\_db.tmp

$schematool -initSchema -dbType derby

$hive

$craete database hivetest;

$

-====

$create table doc(text string) row format delimited fields terminated by '\n' stored as textfile;

On HDFS

$hdfs dfs -put /hadoop-root-namenode-node1.hadoop.user-pandhale.svc.cluster.local.out

On hive :

$load data inpath '/hadoop-root-namenode-node1.hadoop.user-pandhale.svc.cluster.local.out' overwrite into table doc;

$select word , COUNT(\*) FROM DOC LATERAL VIEW explode(split(text,' '))lTable as word GROUP BY word;

=======

Spark

$wget <https://archive.apache.org/dist/spark/spark-2.4.3/spark-2.4.3-bin-hadoop2.7.tgz>

$tar xvf spark-2.4.3-bin-hadoop2.7.tgz

$mv spark-2.4.3-bin-hadoop2.7/ /usr/local/spark

$vi .bashrc

export PATH=/usr/local/spark/bin:$PATH

$source /etc/profile

$cd /usr/local/spark/conf/

$cp spark-env.sh.template spark-env.sh

$vi spark-env.sh

export SPARK\_MASTER\_HOST=node1

export JAVA\_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.312.b07-2.el8\_5.x86\_64/jre

$cp slaves.template slaves

$vi slaves

Node1

$bash

$/usr/local/spark/sbin/start-all.sh

$spark-shell