**Commands Used**

**Vagrant file**

Vagrant.configure("2") do |config|

# Common settings for all VMs

config.vm.box = "hashicorp/bionic64"

# Define VM 1 (node01)

config.vm.define "vm1" do |vm1|

vm1.vm.hostname = "node01"

vm1.vm.network "private\_network", ip: "192.168.50.4"

# Provisioning for node01

vm1.vm.provision "shell", inline: <<-SHELL

sudo apt-get update

sudo apt-get install -y openjdk-11-jdk wget

SHELL

end

# Define VM 2 (node02)

config.vm.define "vm2" do |vm2|

vm2.vm.hostname = "node02"

vm2.vm.network "private\_network", ip: "192.168.50.5"

# Provisioning for node02

vm2.vm.provision "shell", inline: <<-SHELL

sudo apt-get update

sudo apt-get install -y openjdk-11-jdk wget

SHELL

end

End

* Vagrant up
* Vagrant ssh vm1
* ssh-keygen -t rsa
* ssh-copy-id vagrant@node01
* ssh-copy-id vagrant@node02
* wget https://downloads.apache.org/hadoop/common/hadoop-3.4.1/hadoop-3.4.1.tar.gz
* tar -xvzf hadoop-3.4.1.tar.gz
* sudo mv hadoop-3.4.1 /usr/local/hadoop
* sudo nano ~/.bashrc

**add these in .bashrc file**

**export HADOOP\_HOME=/usr/local/hadoop**

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

**export JAVA\_HOME=$(readlink -f /usr/bin/java | sed "s:/bin/java::")**

**export JAVA\_HOME=/usr/lib/jvm/java-11-openjdk-amd64**

* source ~/.bashrc
* cd /usr/local/hadoop/etc/Hadoop
* sudo nano core-site.xml

**<configuration>**

**<property>**

**<name>fs.defaultFS</name>**

**<value>hdfs://192.168.50.4:9000</value>**

**</property>**

**<property>**

**<name>dfs.webhdfs.address</name>**

**<value>hdfs://192.168.50.4:9870</value> <!-- or use 9870 for newer versions -->**

**</property>**

</configuration>

* sudo nano hdfs-site.xml

**<configuration>**

**<property>**

**<name>dfs.replication</name>**

**<value>2</value>**

**</property>**

**<property>**

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

**<value>/usr/local/hadoop/hdfs/namenode</value>**

**</property>**

**<property>**

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

**<value>/usr/local/hadoop/hdfs/datanode</value>**

**</property>**

**<property>**

**<name>dfs.namenode.rpc-address</name>**

**<value>192.168.50.4:9000</value>**

**</property>**

**</configuration>**

* sudo nano mapred-site.xml

**<configuration>**

**<property>**

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

**<value>yarn</value>**

**</property>**

**</configuration>**

* sudo nano yarn-site.xml

**<configuration>**

**<property>**

**<name>yarn.resourcemanager.hostname</name>**

**<value>node01</value>**

**</property>**

**<property>**

**<name>yarn.nodemanager.aux-services</name>**

**<value>mapreduce\_shuffle</value>**

**</property>**

**</configuration>**

* sudo nano workers

**node01**

**node02**

* hdfs namenode -format
* start-dfs.sh
* start-yarn.sh
* wget https://downloads.apache.org/spark/spark-3.5.3/spark-3.5.3-bin-hadoop3.tgz
* tar -xvzf spark-3.5.3-bin-hadoop3.tgz
* sudo mv spark-3.5.3-bin-hadoop3 /usr/local/spark
* sudo nano ~/.bashrc

**export SPARK\_HOME=/usr/local/spark**

**export PATH=$PATH:$SPARK\_HOME/bin:$SPARK\_HOME/sbin**

* source ~/.bashrc
* $SPARK\_HOME/sbin/start-master.sh
* $SPARK\_HOME/sbin/start-slave.sh spark://node01:7077
* Jps
* hdfs dfsadmin -report
* $HADOOP\_HOME/sbin/hadoop-daemon.sh start datanode
* cd /opt/spark/jars
* wget <https://repo1.maven.org/maven2/net/snowflake/snowflake-jdbc/3.17.0/snowflake-jdbc-3.17.0.jar>
* wget <https://repo1.maven.org/maven2/net/snowflake/spark-snowflake_2.12/3.0.0/spark-snowflake_2.12-3.0.0.jar>
* wget [https://search.maven.org/remotecontent?filepath=org/apache/spark/spark-sql\_2.12/3.5.3/spark-sql\_2.12-3.5.3.jar -O spark-sql\_2.12-3.5.3.jar](https://search.maven.org/remotecontent?filepath=org/apache/spark/spark-sql_2.12/3.5.3/spark-sql_2.12-3.5.3.jar%20-O%20spark-sql_2.12-3.5.3.jar)
* sudo nano SnowflakeIntegration.py

**from pyspark.sql import SparkSession**

**# Create a Spark session**

**spark = SparkSession.builder \**

**.appName("SnowflakeIntegration") \**

**.config("spark.jars", "/opt/spark/jars/snowflake-jdbc-3.17.0.jar,/opt/spark/jars/spark-snowflake\_2.12-3.0.0.jar") \**

**.getOrCreate()**

**# Set up Snowflake options**

**snowflake\_options = {**

**"sfURL": "xnsgsxr-av28936.snowflakecomputing.com",**

**"sfDatabase": "BIGDATA",**

**"sfSchema": "SPARK\_SNOWFLAKE",**

**"sfWarehouse": "WAREHOUSE",**

**"sfRole": "ACCOUNTADMIN",**

**"sfUser": "C00313480",**

**"sfPassword": "Password@123",**

**"sfSSL": "true"**

**}**

**try:**

**df = spark.read \**

**.format("snowflake") \**

**.options(\*\*snowflake\_options) \**

**.option("dbtable", "BIGDATA.SPARK\_SNOWFLAKE.SOCIAL\_MEDIA\_DATA") \**

**.load()**

**df.show()**

**print("Data retrieved successfully from Snowflake!")**

**hdfs\_path = "hdfs://192.168.50.4:9000/path/in/hdfs/"**

**df.write \**

**.format("csv") \**

**.mode("overwrite") \**

**.save(hdfs\_path)**

**print("Data successfully written to HDFS.")**

**except Exception as e:**

**print(f"An error occurred: {e}")**

**# Step 5: Stop the Spark session**

**spark.stop()**

* spark-submit --master spark://node01:7077 --conf spark.sql.shuffle.partitions=4 --jars /opt/spark/jars/snowflake-jdbc-3.17.0.jar,/opt/spark/jars/spark-snowflake\_2.12-3.0.0.jar,/usr/local/spark/jars/spark-sql\_2.12-3.5.3.jar /opt/spark/jars/SnowflakeIntegration.py
* sudo nano readfromhdfs.py

**from pyspark.sql import SparkSession**

**# Initialize Spark Session**

**spark = SparkSession.builder \**

**.appName("Read CSV from HDFS") \**

**.getOrCreate()**

**# Read the CSV file from HDFS**

**hdfs\_path = "hdfs://node01:9000/path/in/hdfs/part-00000-1111cef1-3c6d-4cc8-9710-827396165aa1-c000.csv"**

**csv\_df = spark.read.csv(hdfs\_path, header=True, inferSchema=True)**

**# Show the DataFrame (print the content)**

**csv\_df.show()**

* spark-submit /opt/spark/jars/readfromhdfs.py
* hdfs dfs -ls /path/in/hdfs
* hdfs dfs -rm /path/in/hdfs/filename