Environment preparation

Navigate to the following link for details of how to create your cluster:
https://github.com/aws-support-bigdata-cpt-vls/2021/blob/main/Day%201/EMR-EC2%20Setup/EMR%20-EC2%20Setup.pdf

Copy this yaml file and use it as template in the above steps:
https://raw.githubusercontent.com/aws-support-bigdata-cpt-vls/2021/main/Day%201/EMR-EC2%20Setup/EMR-EC2-CFN-template.yaml

sh-4.2$ sudo su hadoop

[hadoop@ip-10-10-10-35 bin]$ cd

[hadoop@ip-10-10-10-35 ~]$ hadoop fs -mkdir /user/hadoop/data/

[hadoop@ip-10-10-10-35 ~]$ mkdir /mnt1/folder

[hadoop@ip-10-10-10-35 ~]$ cd /mnt1/folder

[hadoop@ip-10-10-10-35 folder]$ aws s3 cp s3://nyc-tlc/trip\ data/yellow_tripdata_2020-12.csv /mnt1/folder/yellow_tripdata_2020-12.csv

download: s3://nyc-tlc/trip data/yellow_tripdata_2020-12.csv to ./yellow_tripdata_2020-12.csv

[hadoop@ip-10-10-10-35 folder]$ hadoop fs -put yellow_tripdata_2020-12.csv /user/hadoop/data/

[hadoop@ip-10-10-10-35 folder]$ hadoop fs -ls /user/hadoop/data/
Found 1 items
-rw-r--r-- 1 hadoop hdfsadmingroup 134481400 2021-06-24 14:41 /user/hadoop/data/yellow_tripdata_2020-12.csv

Please Note:
Please do your best to attempt Challenge 1 and 2. Should you have time, attempt solving challenge 3 as well. Please share all the commands used and the output of the commands for the scenarios below. Even if you cannot finish everything, please submit whatever you are able to complete!

Challenge 1 :-

a) How do you check for the replication factor of the file? */user/hadoop/data/yellow_tripdata_2020-12.csv*

b) How do you set replication factor of the file below to 2?

/user/hadoop/data/yellow_tripdata_2020-12.csv

c) Increase replication factor of the below file to 3. Does the command run successfully? Give reasons for your answer.
/user/hadoop/data/yellow_tripdata_2020-12.csv

d) Check the replication factor of the file again, and share the command used as well as the output?
/user/hadoop/data/yellow_tripdata_2020-12.csv

e) Run the command below and explain why we have "Mis-replicated blocks "

hdfs fsck /user/hadoop/data/

f) How do you check the default replication factor configured on your cluster?

g) How do you check the default block size configured on your cluster?

Challenge 2 :-

Run the below script on the cluster using either spark-submit or pyspark interactive interpreter.

```
from pyspark import SparkConf, SparkContext
from pyspark.sql import SparkSession
spark = SparkSession.builder.config(conf=SparkConf()).getOrCreate()
df = spark.read.format("csv").option("header", "true").load("hdfs:///user/hadoop/data/*.csv")
results = df.groupBy("VendorID").count()
results.show()
```

a) Modify the script above and write the output in HDFS in parquet format. Submit your modified script as the answer.

b) After reading the dataframe:

df = spark.read.format("csv").option("header", "true").load("hdfs:///user/hadoop/data/*.csv")

How many partitions are created? Please share the command to check for the partitions, as well as the partition count.

c) How do you change the number of partitions?

d) How do you write a single file into HDFS?

e) Print the schema of the results dataframe above?

f) Write your output data into HDFS so that your folder structure will look like vendorid="somevalue", that is hive style partitions (key=value)?

Share your output schema as well as your your output folder structure in HDFS?

Example folder structure

```
$ hadoop fs -ls /user/hadoop/output/
drwxr-xr-x   - hadoop hdfsadmingroup          0 2021-06-24 10:43 /user/hadoop/output/VendorID=somevalue1
```

Challenge 3:-

You are working as a big data developer for a startup which sells groceries through their ecommerce site.
The manager wants to understand which product sells most in real time in order to manage their grocery stock better and plan for specials. To this end you are tasked to design a real time pyspark socket streaming application to collect information about product sales in real time.

For this use case, the online website will be represented by a producer python application (Producer.py) which runs on an EC2 instance (webserver for the online shopping site). This producer randomly generates products which

represents what customers are buying. The pyspark spark streaming App (Consumer.py) is to run on an EMR cluster hosted on AWS cloud where you will collect the product sales information and perform some analysis.

The manager is not very knowledgeable about IT and has the following questions to determine how feasible his idea is. As the big data specialist in the team, you are expected to answer the questions. Please can you provide answers to the following questions?

Note: The spark app must use port 8000

a) How would you determine the IP addresses of the ec2 host and the EMR cluster after logging to both via terminal? Particularly, what command will you run and what will you look for in the output?

b) What is the ec2-server (producer) instance ip address and what is the EMR cluster master node IP address?

c) After determining the IPs, you test to see if the ec2 is reachable from the host, what command will you use and what will you check from the output to ensure that the the ec2 server is reachable?

d) You notice that the ec2 server is not reachable from the EMR master node. A colleague suggests you need to update the security group rules. Will you edit the inbound or outbound rules or both? why?

e) What inbound/outbound rules will you configure on the ec2 producer and on EMR cluster. If inbound rule, indicate the Protocol, Port range and Source for each rule you will add.

f) Run the producer using the script below:

```
# Producer.py
import socket
import time
import json
import random
import datetime
def ProductInfoProducer():
   item=random.choice(['Apple', 'Orange', 'Mango', 'Guava', 'Pear','Grape', 'Lemon', 'Banana'])
   return f'{item}\n'
HOST = '10.10.20.158'  #ec2  server producer
PORT = 8000 #port number
with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
   s.bind((HOST, PORT))
   s.listen()
   conn, addr = s.accept()
   with conn:
      print('Connected by', addr)
      while True:
         data = ProductInfoProducer()
         print(data)
         conn.send(data.encode('utf-8'))
s.close()
```

g) After creating the file, Producer.py, you run it on ec2 server, how will you check from the EMR cluster that the ec2 producer is listening on port 8000?

h) Next, you create the spark streaming application (Consumer.py) in the home directory of the master node of the EMR cluster using the following code.

```
# Consumer.py
from pyspark.sql import SparkSession
from pyspark.sql.functions import explode
from pyspark.sql.functions import split

spark = SparkSession \
    .builder \
    .appName("ProductCounter") \
    .getOrCreate()

lines = spark \
    .readStream \
    .format("socket") \
    .option("host", "10.10.20.158") \
    .option("port", 8000) \
    .load()

# Extract product information from streams
products = lines.select(
   explode(
       split(lines.value, " ")
   ).alias("product")
)

# count products bought by customers
productCount = products.groupBy("product").count()

query = productCount \
    .writeStream \
    .outputMode("complete") \
    .format("console") \
   .start()

query.awaitTermination()
```

you may need to edit the host ip to match your environment.

i) Which command will you use to submit the app in client mode?

j) Which command will you use to submit in cluster mode?

k) To see the output on the console, which mode will you use?

l) After submitting the spark application, can you indicate the application id? How did you get it?

m) Which is the product with the highest purchase frequency in the second batch. Can you paste the output of the second batch here?