به نام خدا

دانشکده مهندسی کامپیوتر

گزارش تمرین سوم

درس: رایانش ابری

دانشجو: فرشید نوشی – ۹۸۳۱۰۶۸

بخش اول

به این خاطر که سیستم استفاده شده در این تمرین از معماری ARM برای پردازنده استفاده کرده بود و پاسخ تدریسیاران، از یک مخزن دیگر Hadoop برروی سیستم بالا آورده شد که در ادامه مراحل اجرای آن آمده‌اند.

پس از گرفتن کدهای مربوط به docker-compose با استفاده از دستور docker-compose up -d اقدام به بالا آوردن Hadoop در سیستم میکنیم:

```
farshid @ MacBook-Pro: /Volumes/Farshid_SSD/Projects/University/Cloud Computing/Project 3/docker-hadoop
(venv) $ docker-compose up -d
[+] Running 6/6
 ✔ Container namenode          Started
 ✔ Container historyserver     Started
 ✔ Container nodemanager       Started
 ✔ Container resourcemanager   Started
 ✔ Container datanode          Started
 ✔ Container postgres13        Started
```

سیستم نیز پس از دانلود ایمیج‌های مربوط و فایل‌های لازم Hadoop را اجرا میکند.

تصویر زیر شامل کانتینرهای ایجاد شده است. البته در بالا نیز آن‌هایی که ایجاد شدند آورده شده‌اند، همانطور که میبینیم در چند سطر اول کانتینرهای مربوط به این تمرین هستند که ایجاد شده اند و بالا آمده‌اند:

```
farshid @ MacBook-Pro: /Volumes/Farshid_SSD/Projects/University/Cloud Computing/Project 3/docker-hadoop                                                                  master [+!]
(venv) $ docker ps
CONTAINER ID   IMAGE                                                     COMMAND                  CREATED          STATUS                  PORTS                                            NAMES
26ca37c71c12   2edf9c994f19                                              "/kube-vpnkit-forwar…"   11 minutes ago   Up 11 minutes                                                            k8s_vpnkit-controller_vpnkit-controller_kube-s
ystem_4388ff71-01c6-4d87-87ee-372dda652bd7_94
b7ac0eb8334e   wxwmatt/hadoop-historyserver:2.1.1-hadoop3.3.1-java8      "/entrypoint.sh /run…"   2 hours ago      Up 2 hours (healthy)    0.0.0.0:8188->8188/tcp                           historyserver
da093b0a38f2   wxwmatt/hadoop-namenode:2.1.1-hadoop3.3.1-java8           "/entrypoint.sh /run…"   2 hours ago      Up 2 hours (healthy)    0.0.0.0:9000->9000/tcp, 0.0.0.0:9870->9870/tcp   namenode
0178674e5088   wxwmatt/hadoop-datanode:2.1.1-hadoop3.3.1-java8           "/entrypoint.sh /run…"   2 hours ago      Up 2 hours (healthy)    0.0.0.0:9864->9864/tcp                           datanode
126dc6e53903   wxwmatt/hadoop-resourcemanager:2.1.1-hadoop3.3.1-java8    "/entrypoint.sh /run…"   2 hours ago      Up 2 hours (healthy)    0.0.0.0:8088->8088/tcp                           resourcemanager
d82624025e13   wxwmatt/hadoop-nodemanager:2.1.1-hadoop3.3.1-java8        "/entrypoint.sh /run…"   2 hours ago      Up 2 hours (healthy)    0.0.0.0:8042->8042/tcp                           nodemanager
440f49974c2b   postgres:latest                                           "docker-entrypoint.s…"   2 hours ago      Up 2 hours              0.0.0.0:5432->5432/tcp                           postgres13
44a07b16fb62   farshidnooshi/url-mining                                  "python main.py"         7 hours ago      Up 7 hours                                                               k8s_url-mining-container_url-mining-deployment
-55f6496867-mmr6j_default_fc0a5f6a-3725-41ae-881f-8665b4bc151d_3
9dfc67bd6fa8   farshidnooshi/url-mining                                  "python main.py"         7 hours ago      Up 7 hours                                                               k8s_url-mining-container_url-mining-deployment
-55f6496867-jg9rr_default_24ff2bec-aed4-41ca-ad4f-e0ff1e21454e_3
1ebde96c9ea1   redis                                                     "docker-entrypoint.s…"   7 hours ago      Up 7 hours                                                               k8s_redis-container_redis-deployment-7cd46c6ff
-mtsf5_default_c23194b7-a9ba-4fc7-a762-eabc0899618c_3
9e3378c072ed   farshidnooshi/url-mining                                  "python main.py"         7 hours ago      Up 7 hours                                                               k8s_url-mining_url-mining_default_6c310d78-3ae
2-4844-a0ce-50dda4bcb95a_3
43a06320b3f0   registry.k8s.io/pause:3.8                                 "/pause"                 7 hours ago      Up 7 hours                                                               k8s_POD_redis-deployment-7cd46c6ff-mtsf5_defau
lt_c23194b7-a9ba-4fc7-a762-eabc0899618c_3
2a20705c3317   b19406328e70                                              "/coredns -conf /etc…"   7 hours ago      Up 7 hours                                                               k8s_coredns_coredns-565d847f94-ctshp_kube-syst
em_b3820285-b09e-4faa-87a7-e5322784dc9a_3
77900dc253b3   registry.k8s.io/pause:3.8                                 "/pause"                 7 hours ago      Up 7 hours                                                               k8s_POD_coredns-565d847f94-ctshp_kube-system_b
```

در یک خوشه Hadoop، هر Container نقش خاصی را ایفا می کند تا پردازش توزیع شده داده‌ها را فعال کند.

۱. NameNode:

- NameNode مسئول مدیریت سیستم فایل Hadoop (HDFS) است.

- Meta Dataهای مرتبط با فایل ها و دایرکتوری ها را در HDFS، مانند مکان ها و مجوزهای آنها، بررسی می کند.

- توزیع داده ها در میان DataNode ها را در خوشه مدیریت میکند.

- File System Name space NameNode را حفظ می کند و درخواست های کاربر مربوط به File Operation را مدیریت می کند.

۲. DataNode:

- DataNodeها بلوک های داده واقعی فایل ها را در HDFS ذخیره و مدیریت می کنند.

- آنها داده ها را از کلاینت ها یا دیگر DataNode ها دریافت کرده و روی دیسک local ذخیره می کنند.

- DataNode ها برای اطمینان از Fault Tolerance با ایجاد چندین نسخه از Data Block و توزیع آنها در گره های مختلف در خوشه، Data Replication را انجام می دهد.

- آنها با NameNode ارتباط برقرار می کنند تا وضعیت بلوک‌هایی را که ذخیره می کنند گزارش کنند همچنین instruction هایی را برای Data Replication یا حذف مدیریت می کنند.

۳. NodeManager:

- NodeManager ها مسئول مدیریت و نظارت بر گره های تکی در خوشه هستند.

- آنها بر تخصیص منابع (CPU، حافظه و غیره) در هر گره برای اجرای کانتینرهای برنامه نظارت می کنند.

- NodeManager برای درخواست منابع و گزارش وضعیت و سلامت گره با ResourceManager ارتباط برقرار می کند.

- آنها اجرای کانتینرهای برنامه را در گره های مربوطه خودشان را راه اندازی و مدیریت می کنند.

۴. ResourceManager:

- ResourceManager مرجع مرکزی است که تخصیص منابع را در کلاستر مدیریت می‌کند.

- منابع موجود در کلاستر مانند CPU و حافظه را ردیابی می کند.

- ResourceManager درخواست های منابع را از برنامه ها دریافت می کند (کارهای MapReduce، مشاغل Spark و ...) و نحوه تخصیص منابع به آنها را تعیین می کند.

- سلامت و وضعیت NodeManagerها را نظارت می کند و زمان بندی وظایف به گره ها را بر اساس منابع موجود انجام می دهد.

۵. HistoryManager:

- HistoryManager مسئول جمع آوری و مدیریت گزارش های تاریخی و اطلاعات مربوط به کارهای تکمیل شده MapReduce است.

- گزارش های history کارها را ذخیره می کند که حاوی اطلاعات دقیق در مورد اجرای هر کار، از جمله مسیرهای ورودی و خروجی، جزئیات پیکربندی، شمارنده ها و غیره است.

- HistoryManager مدیران و کاربران را قادر می‌سازد تا اجرای گذشته کارها را در خوشه تجزیه و تحلیل و بررسی کنند.

در ادامه نیز دستور JPS و نمونه خروجی webui در لوکال هاست آورده شده‌اند.

Decommissioning Nodes	0
Entering Maintenance Nodes	0
Total Datanode Volume Failures	0 (0 B)
Number of Under-Replicated Blocks	7
Number of Blocks Pending Deletion (including replicas)	0
Block Deletion Start Time	Sat Jun 10 23:15:36 +0330 2023
Last Checkpoint Time	Sat Jun 10 23:15:34 +0330 2023
Enabled Erasure Coding Policies	RS-6-3-1024k

NameNode Journal Status

Current transaction ID: 50

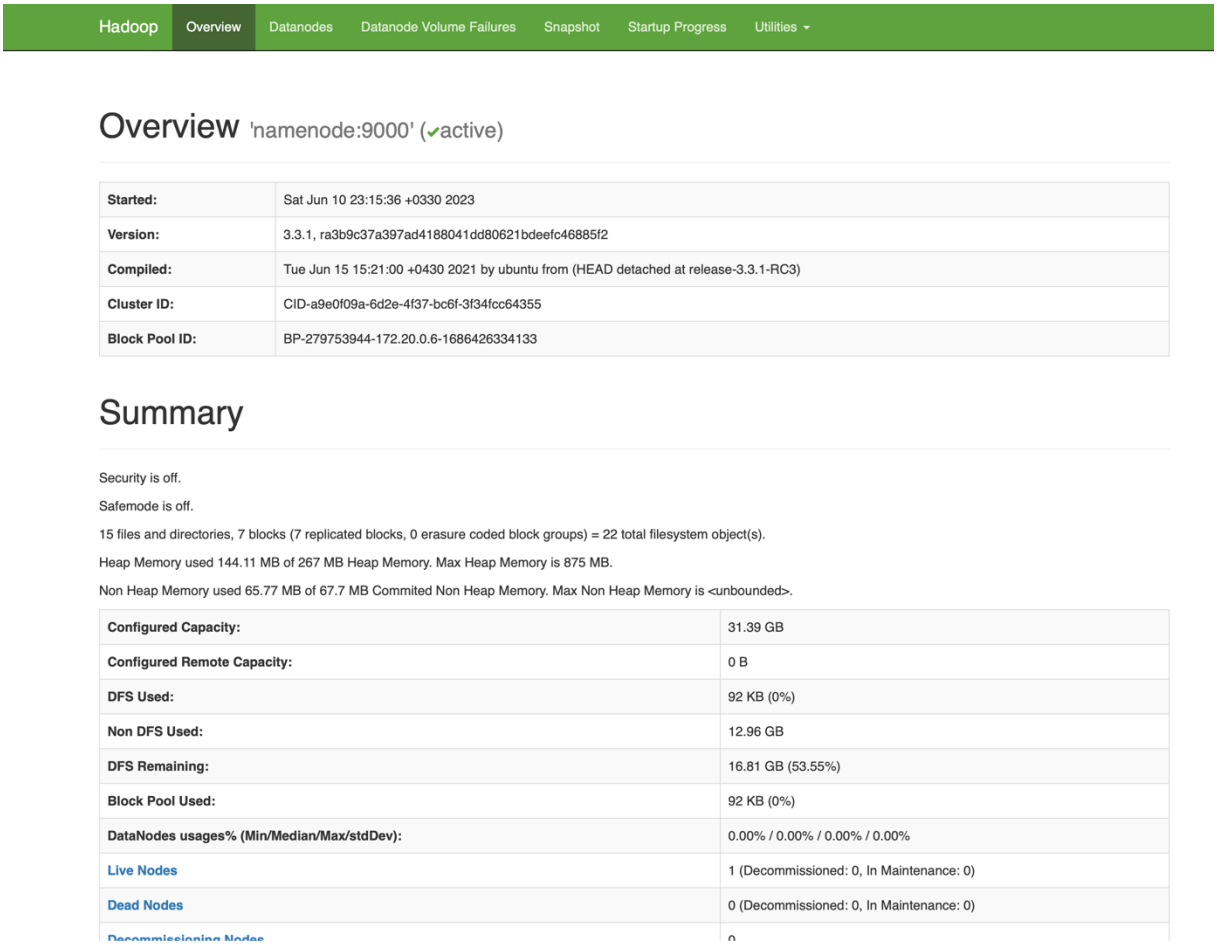
Journal Manager	State
FileJournalManager(root=/hadoop/dfs/name)	EditLogFileOutputStream(/hadoop/dfs/name/current/edits_inprogress_0000000000000000001)

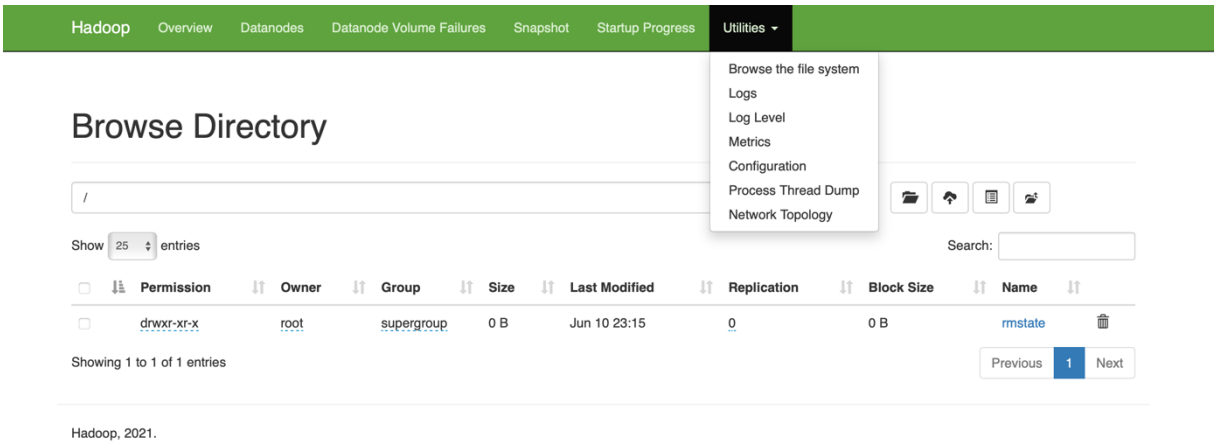
NameNode Storage

Storage Directory	Type	State
/hadoop/dfs/name	IMAGE_AND_EDITS	Active

DFS Storage Types

Storage Type	Configured Capacity	Capacity Used	Capacity Remaining	Block Pool Used	Nodes In Service
DISK	31.39 GB	92 KB (0%)	16.81 GB (53.55%)	92 KB	1


Hadoop
Overview
Datanodes
Datanode Volume Failures
Snapshot
Startup Progress
Utilities
Overview 'namenode:9000' (✔active)
Started: Sat Jun 10 23:15:36 +0330 2023
Version: 3.3.1, ra3b9c37a397ad4188041dd80621bdeefc46885f2
Compiled: Tue Jun 15 15:21:00 +0430 2021 by ubuntu from (HEAD detached at release-3.3.1-RC3)
Cluster ID: CID-a9e0f09a-6d2e-4f37-bc6f-3f34fcc64355
Block Pool ID: BP-279753944-172.20.0.6-1686426334133
Summary
Security is off.
Safemode is off.
15 files and directories, 7 blocks (7 replicated blocks, 0 erasure coded block groups) = 22 total filesystem object(s).
Heap Memory used 144.11 MB of 267 MB Heap Memory. Max Heap Memory is 875 MB.
Non Heap Memory used 65.77 MB of 67.7 MB Commited Non Heap Memory. Max Non Heap Memory is <unbounded>.
Configured Capacity: 31.39 GB
Configured Remote Capacity: 0 B
DFS Used: 92 KB (0%)
Non DFS Used: 12.96 GB
DFS Remaining: 16.81 GB (53.55%)
Block Pool Used: 92 KB (0%)
DataNodes usages% (Min/Median/Max/stdDev): 0.00% / 0.00% / 0.00% / 0.00%
Live Nodes 1 (Decommissioned: 0, In Maintenance: 0)
Dead Nodes 0 (Decommissioned: 0, In Maintenance: 0)


Hadoop
Overview
Datanodes
Datanode Volume Failures
Snapshot
Startup Progress
Utilities
Browse the file system
Logs
Log Level
Metrics
Configuration
Process Thread Dump
Network Topology
Browse Directory
/
Show 25 entries
Search:
Permission
Owner
Group
Size
Last Modified
Replication
Block Size
Name
drwxr-xr-x
root
supergroup
0 B
Jun 10 23:15
0
0 B
rmstate
Showing 1 to 1 of 1 entries
Previous
1
Next
Hadoop, 2021.

```
farshid @ MacBook-Pro: /Volumes/Farshid_SSD/Projects/University/Cloud Computing/Project 3
(venv) $ docker exec historyserver jps
362 ApplicationHistoryServer
1661 Jps
------- ---------- ------------------------------------------------------------------------
farshid @ MacBook-Pro: /Volumes/Farshid_SSD/Projects/University/Cloud Computing/Project 3
(venv) $ docker exec namenode jps
1764 Jps
406 NameNode
------- ---------- ------------------------------------------------------------------------
farshid @ MacBook-Pro: /Volumes/Farshid_SSD/Projects/University/Cloud Computing/Project 3
(venv) $ docker exec datanode jps
358 DataNode
1724 Jps
------- ---------- ------------------------------------------------------------------------
farshid @ MacBook-Pro: /Volumes/Farshid_SSD/Projects/University/Cloud Computing/Project 3
(venv) $ docker exec resourcemanager jps
1851 Jps
351 ResourceManager
------- ---------- ------------------------------------------------------------------------
farshid @ MacBook-Pro: /Volumes/Farshid_SSD/Projects/University/Cloud Computing/Project 3
(venv) $ docker exec nodemanager jps
355 NodeManager
1736 Jps
```

- 'docker exec historyserver jps': خروجی نشان می دهد که ApplicationHistoryServer در حال اجرا است و Jps نیز فهرست شده است که ابزار Java Process Status است که برای فهرست کردن Java Process استفاده می شود.

- 'docker exec namenode jps': خروجی نشان می دهد که فرآیند NameNode در حال اجرا است و Jps نیز فهرست شده است.

- 'docker exec datanode jps': خروجی نشان می دهد که فرآیند DataNode در حال اجرا است و Jps نیز فهرست شده است.

- `docker exec resourcemanager jps`: خروجی تأیید می کند که فرآیند ResourceManager در حال اجرا است و Jps نیز فهرست شده است.

- `docker exec nodemanager jps`: خروجی نشان می دهد که فرآیند NodeManager در حال اجرا است و Jps نیز لیست شده است.

این پاسخ ها نشان می دهد که کانتینرها به درستی کار می کنند.

بخش دوم

با استفاده از دستور زیر فایل dataset.csv به namenode کپی شد:

```
farshid @ MacBook-Pro : /Volumes/Farshid_SSD/Projects/University/Cloud Computing/Project 3
(venv) $ docker cp dataset.csv namenode:/dataset.csv

                        Successfully copied 103MB to namenode:/dataset.csv
```

بخش سوم

در این بخش ابتدا به مانند تصویر زیر dataset.csv را به پوشه خواسته شده با استفاده از HDFS CLI میبریم.

برای اینکار ابتدا یک ترمینال از namenode گرفتیم و در ادامه نیز دستورات را با استفاده از hdfs dfs میزنیم:

```
root@da093b0a38f2:~# hdfs dfs -mkdir /user
root@da093b0a38f2:~# hdfs dfs -mkdir /user/root
root@da093b0a38f2:~# hdfs dfs -mkdir /user/root/input
root@da093b0a38f2:~# ls
data  jars  res
root@da093b0a38f2:~# cd ..
root@da093b0a38f2:/# ls
KEYS  app  bin  boot  dataset.csv  dev  entrypoint.sh  etc  hadoop  hadoop-data  home  lib  media  mnt  opt  proc  root  run  run.sh  sbin  srv  sys  tmp  usr  var
root@da093b0a38f2:/# hdfs dfs -put dataset.csv /user/root/input/dataset.csv
root@da093b0a38f2:/# exit
exit
```

در ادامه نیز با دستور docker cp dataset.csv namenode/dataset.csv فایل دیتاست را به جایی که میخواهیم میبریم. در ادامه نیز با دستور زیر فایل اول map reduce را اجرا میکنیم:

hadoop jar /opt/hadoop-3.3.1/share/hadoop/tools/lib/hadoop-streaming-3.3.1.jar -files count_candidates.py -mapper "python3 count_candidates.py mapper" -reducer "python3 count_candidates.py reducer" -input /user/root/input/dataset.csv -output /user/root/output/program1

docker-compose.yml | count_candidates.py | out.txt | dataset.csv | count_tweets_by_state.py | count_tweets_by_state_geo.py

```python
#!/usr/bin python3

from mrjob.job import MRJob
import csv

new *
class CountCandidates(MRJob):

    new *
    def mapper(self, _, line):
        reader = csv.reader([line])
        fields = next(reader)
        if len(fields) >= 7 and fields[0] != 'created_at':
            tweet = fields[2]
            likes = float(fields[3])
            retweets = float(fields[4])
            source = fields[5]
            candidates = ["Both Candidate", 'Donald Trump', 'Joe Biden']

            for candidate in candidates:
                if candidate.lower() in tweet.lower():
                    yield (candidate, (1, likes, retweets, source))

    new *
    def reducer(self, key, values):
        count = 0
        total_likes = 0
        total_retweets = 0
        source_counts = {'Twitter Web App': 0, 'Twitter for iPhone': 0, 'Twitter for Android': 0}

        for value in values:
            count += value[0]
            total_likes += value[1]
            total_retweets += value[2]
            if value[3] in source_counts:
                source_counts[value[3]] += 1
        yield (key, (total_likes, total_retweets, *source_counts.values()))

if __name__ == '__main__':
    CountCandidates.run()
```

```
"Both Candidate"        [125.0, 42.0, 22, 14, 15]
"Donald Trump"  [19077.0, 5729.0, 956, 404, 336]
"Joe Biden"     [38038.0, 17265.0, 2034, 1095, 784]
```

برای فایل‌های دیگر نیز به همین ترتیب:

docker-compose.yml | count_candidates.py | dataset.csv | count_tweets_by_state.py | count_tweets_by_state_geo.py

```python
import csv


class TweetsByState(MRJob):

    def mapper(self, _, line):
        reader = csv.reader([line])
        fields = next(reader)
        if len(fields) >= 18 and fields[0] != 'created_at':
            state = fields[18]
            tweet_time = fields[0]
            tweet = fields[2]
            possible_states = ['New York', 'Texas', 'California', 'Florida']
            if 'Joe Biden' in tweet or 'Donald Trump' in tweet:
                hour = int(tweet_time[11:13])
                is_state_in_possible_states = any(state.lower() == possible_state.lower() for possible_state in possible_states)
                if hour >= 9 and hour <= 17 and is_state_in_possible_states:
                    yield (state, (1, 'both' if 'Joe Biden' in tweet and 'Donald Trump' in tweet else 'biden' if 'Joe Biden' in tweet else 'trump'))

    def reducer(self, key, values):
        total_tweets = 0
        both_tweets = 0
        biden_tweets = 0
        trump_tweets = 0

        for value in values:
            total_tweets += 1
            if value[1] == 'both':
                both_tweets += 1
            elif value[1] == 'biden':
                biden_tweets += 1
            elif value[1] == 'trump':
                trump_tweets += 1

        both_percentage = both_tweets / total_tweets
        biden_percentage = biden_tweets / total_tweets
        trump_percentage = trump_tweets / total_tweets
        output = f"{both_percentage:.4f}, {biden_percentage:.4f}, {trump_percentage:.4f}, {total_tweets}"
        yield (key, output)

if __name__ == '__main__':
    TweetsByState.run()
```

```
"California"    "0.0897, 0.5641, 0.3462, 78"
"Florida"       "0.0316, 0.7579, 0.2105, 95"
"New York"      "0.0568, 0.7386, 0.2045, 88"
"Texas" "0.0175, 0.8246, 0.1579, 57"
```

docker-compose.yml | count_candidates.py | dataset.csv | count_tweets_by_state.py | count_tweets_by_state_geo.py

```python
class CountTweetsByStateGeo(MRJob):

    new *
    def mapper(self, _, line):
        reader = csv.reader([line])
        fields = next(reader)
        if len(fields) >= 7 and fields[0] != 'created_at':
            tweet = fields[2]
            created_at = fields[0]
            lat = float(fields[13]) if fields[13] else 0
            long = float(fields[14]) if fields[14] else 0

            if (45.0153 <= lat <= 79.7624) and (32.5121 <= long <= 124.6509):
                state = "California"
            elif (40.4772 <= lat <= 45.0153) and (-79.7624 <= long <= -71.7517):
                state = "New York"
            else:
                return

            if "Donald Trump" in tweet or "Joe Biden" in tweet:
                yield state, (1 if "Donald Trump" in tweet else 0, 1 if "Joe Biden" in tweet else 0)

    new *
    def reducer(self, key, values):
        total_tweets = 0
        both_tweets = 0
        total_trump = 0
        total_biden = 0

        for trump, biden in values:
            total_tweets += 1
            total_trump += trump
            total_biden += biden
            if trump and biden:
                both_tweets += 1

        both_percentage = both_tweets / total_tweets
        biden_percentage = total_biden / total_tweets
        trump_percentage = total_trump / total_tweets
        output = f"{both_percentage:.8f}, {biden_percentage:.4f}, {trump_percentage:.4f}, {total_tweets}"
        yield (key, output)

if __name__ == '__main__':
    CountTweetsByStateGeo.run()
```

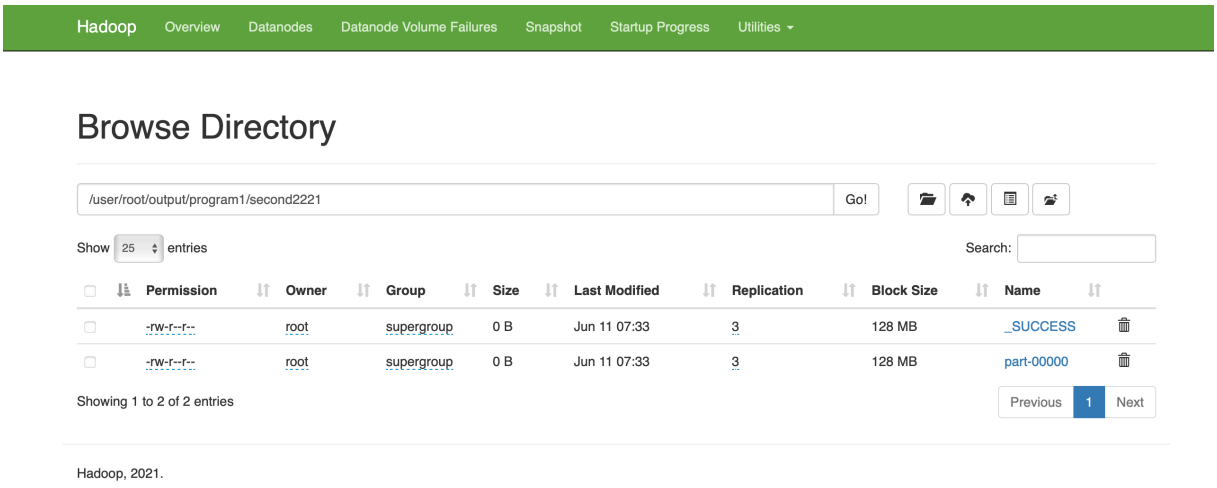
```
"California"    "0.00000000, 0.9091, 0.0909, 11"
"New York"      "0.03956835, 0.6871, 0.3525, 278"
```

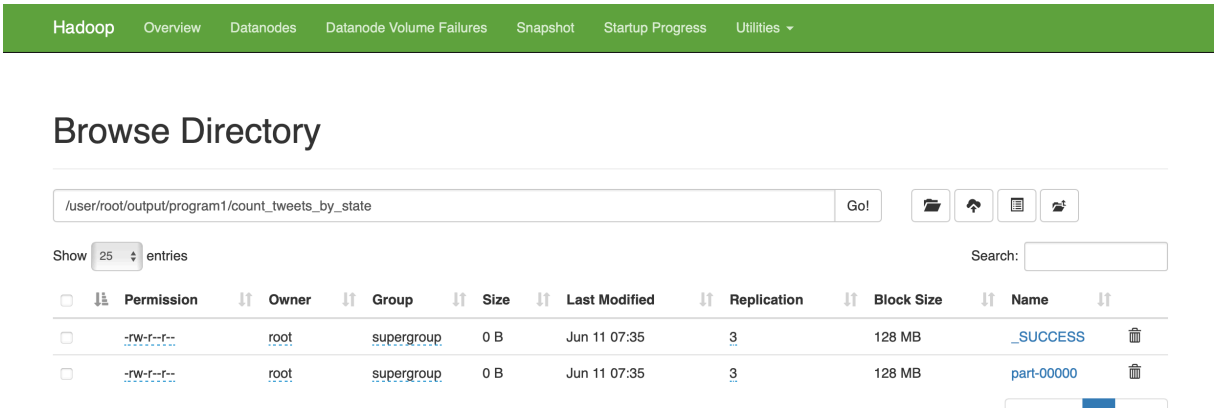
نمونه خروجی برروی داکر:

```
root@da093b0a38f2:~/code# hadoop jar /opt/hadoop-3.3.1/share/hadoop/tools/lib/hadoop-streaming-3.3.1.jar -files count_candidates.py -mapper "python3 count_candidates.py" -reducer "python3 count_candidates.py" -input /user/root/input
/dataset.csv -output /user/root/output/program1/second2112
packageJobJar: [/tmp/hadoop-unjar3780988147869633798/] [] /tmp/streamjob1719364802219628915.jar tmpDir=null
2023-06-11 03:58:13,616 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at resourcemanager/172.20.0.2:8032
2023-06-11 03:58:13,684 INFO client.AHSProxy: Connecting to Application History server at historyserver/172.20.0.4:10200
2023-06-11 03:58:13,701 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at resourcemanager/172.20.0.2:8032
2023-06-11 03:58:13,701 INFO client.AHSProxy: Connecting to Application History server at historyserver/172.20.0.4:10200
2023-06-11 03:58:13,848 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/root/.staging/job_1686426344531_0009
2023-06-11 03:58:14,091 INFO mapred.FileInputFormat: Total input files to process : 1
2023-06-11 03:58:14,124 INFO mapreduce.JobSubmitter: number of splits:2
2023-06-11 03:58:14,210 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1686426344531_0009
2023-06-11 03:58:14,210 INFO mapreduce.JobSubmitter: Executing with tokens: []
2023-06-11 03:58:14,304 INFO conf.Configuration: resource-types.xml not found
2023-06-11 03:58:14,304 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2023-06-11 03:58:14,561 INFO impl.YarnClientImpl: Submitted application application_1686426344531_0009
2023-06-11 03:58:14,613 INFO mapreduce.Job: The url to track the job: http://resourcemanager:8088/proxy/application_1686426344531_0009/
2023-06-11 03:58:14,614 INFO mapreduce.Job: Running job: job_1686426344531_0009
2023-06-11 03:58:18,715 INFO mapreduce.Job: Job job_1686426344531_0009 running in uber mode : false
2023-06-11 03:58:18,722 INFO mapreduce.Job:  map 0% reduce 0%
2023-06-11 03:58:25,859 INFO mapreduce.Job:  map 50% reduce 0%
2023-06-11 03:58:26,864 INFO mapreduce.Job:  map 100% reduce 0%
2023-06-11 03:58:29,885 INFO mapreduce.Job:  map 100% reduce 100%
2023-06-11 03:58:29,913 INFO mapreduce.Job: Job job_1686426344531_0009 completed successfully
2023-06-11 03:58:30,070 INFO mapreduce.Job: Counters: 54
        File System Counters
                FILE: Number of bytes read=198
                FILE: Number of bytes written=838766
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=103282660
                HDFS: Number of bytes written=0
                HDFS: Number of read operations=11
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
                HDFS: Number of bytes read erasure-coded=0
        Job Counters
                Launched map tasks=2
oject 3 > code > count_candidates.py                6:30 (1179 chars, 37 line breaks)  LF  UTF-8  4 spaces  Project 3  Dracula (Material)  Python 3.10 (Project 3)
```

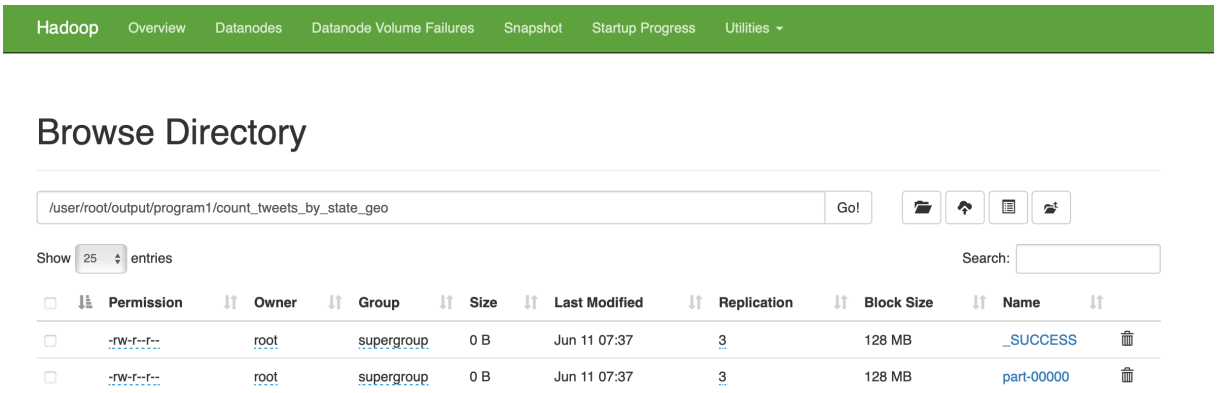
```
                Map input records=200000
                Map output records=6
                Map output bytes=265
                Map output materialized bytes=267
                Input split bytes=200
                Combine input records=0
                Combine output records=0
                Reduce input groups=3
                Reduce shuffle bytes=267
                Reduce input records=6
                Reduce output records=0
                Spilled Records=12
                Shuffled Maps =2
                Failed Shuffles=0
                Merged Map outputs=2
                GC time elapsed (ms)=160
                CPU time spent (ms)=1860
                Physical memory (bytes) snapshot=735088640
                Virtual memory (bytes) snapshot=18039099392
                Total committed heap usage (bytes)=661127168
                Peak Map Physical memory (bytes)=277241856
                Peak Map Virtual memory (bytes)=4901027840
                Peak Reduce Physical memory (bytes)=183246848
                Peak Reduce Virtual memory (bytes)=8243359744
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=103282460
        File Output Format Counters
```

نمونه خروجی موفق و ذخیره سالم در web ui:





```
        Total megabyte-milliseconds taken by all reduce tasks=13803520
Map-Reduce Framework
        Map input records=200000
        Map output records=4
        Map output bytes=180
        Map output materialized bytes=162
        Input split bytes=200
        Combine input records=0
        Combine output records=0
        Reduce input groups=2
        Reduce shuffle bytes=162
        Reduce input records=4
        Reduce output records=0
        Spilled Records=8
        Shuffled Maps =2
        Failed Shuffles=0
        Merged Map outputs=2
        GC time elapsed (ms)=108
        CPU time spent (ms)=1700
        Physical memory (bytes) snapshot=729288512
        Virtual memory (bytes) snapshot=18033336320
        Total committed heap usage (bytes)=663748608
        Peak Map Physical memory (bytes)=273850368
        Peak Map Virtual memory (bytes)=4895739904
        Peak Reduce Physical memory (bytes)=181747712
        Peak Reduce Virtual memory (bytes)=8243171328
Shuffle Errors
        BAD_ID=0
        CONNECTION=0
        IO_ERROR=0
        WRONG_LENGTH=0
        WRONG_MAP=0
        WRONG_REDUCE=0
File Input Format Counters
        Bytes Read=183282460
```



در بالا خروجی نمونه از تمامی سه تا تسک در ترمینال و فایل سیستم آورده شده است.

خروجی نهایی پس از اجراها:

Summary

Security is off.

Safemode is off.

171 files and directories, 86 blocks (86 replicated blocks, 0 erasure coded block groups) = 257 total filesystem object(s).

Heap Memory used 229.9 MB of 321 MB Heap Memory. Max Heap Memory is 875 MB.

Non Heap Memory used 76.51 MB of 78.64 MB Commited Non Heap Memory. Max Non Heap Memory is <unbounded>.

Configured Capacity:	31.39 GB
Configured Remote Capacity:	0 B
DFS Used:	107.74 MB (0.34%)
Non DFS Used:	15.91 GB
DFS Remaining:	13.75 GB (43.82%)
Block Pool Used:	107.74 MB (0.34%)
DataNodes usages% (Min/Median/Max/stdDev):	0.34% / 0.34% / 0.34% / 0.00%
Live Nodes	1 (Decommissioned: 0, In Maintenance: 0)
Dead Nodes	0 (Decommissioned: 0, In Maintenance: 0)
Decommissioning Nodes	0
Entering Maintenance Nodes	0
Total Datanode Volume Failures	0 (0 B)
Number of Under-Replicated Blocks	86
Number of Blocks Pending Deletion (including replicas)	0
Block Deletion Start Time	Sat Jun 10 23:15:36 +0330 2023

NameNode Journal Status

Current transaction ID: 1654

Journal Manager	State
FileJournalManager(root=/hadoop/dfs/name)	EditLogFileOutputStream(/hadoop/dfs/name/current/edits_inprogress_0000000000000000001)

NameNode Storage

Storage Directory	Type	State
/hadoop/dfs/name	IMAGE_AND_EDITS	Active

DFS Storage Types

Storage Type	Configured Capacity	Capacity Used	Capacity Remaining	Block Pool Used	Nodes In Service
DISK	31.39 GB	107.74 MB (0.34%)	13.75 GB (43.82%)	107.74 MB	1

Hadoop, 2021.