Azat Dovgeldiyev

CSC 555

Assignment 4

Problem 1.

a) How long will it take for the reducer to write the job output on a 5-node Hadoop cluster? (ignoring the cost of Map processing, but counting replication cost in the output writing).

Dividing 78 by 5 worker nodes will give us 15.6, which is 16 mins (1*16). Since the replication factor is set to 3, 16 *3 = **48 mins** to write the job.

b) How long will it take for reducer(s) to write the job output to 10 Hadoop worker nodes? (Assume that data is distributed evenly and replication factor is set to 1)

78 dividing by 10 is 7.8 round up to 8, and 8 iteration times by 1 minute is 8 minutes. Since the replication factor is set to 1, then it will take 8*1 =**8 mins to write the job.**

c) How long will it take for reducer(s) to write the job output to 10 Hadoop worker nodes? (Assume that data is distributed evenly and replication factor is set to 3)

Replication factor is set to 3, and 78 dividing by 10 is 7.8 round up to 8, and 8 iteration times by 1 minute is 8 minutes. 8*3 = **24 mins** to write a job.

d) How long will it take for reducer(s) to write the job output to 100 Hadoop worker nodes? (Assume that data is distributed evenly and replication factor is set to 1)

Without any failure we can assume that 100 worker nodes (78/100 = 0.78 round up to 1) gives 1 iteration, and times by 1 min, 1*1, gives 1 min. And 1-minute times by 1 replication factor gives us **1min** to write a job.

e) How long will it take for reducer(s) to write the job output to 100 Hadoop worker nodes? (Assume that data is distributed evenly and replication factor is set to 3)

Without any failure we can assume that 100 worker nodes (78/100 = 0.78 round up to 1) gives 1 iteration, and times by 1 min, 1*1, gives 1 min. And 1-minute times by 3 replication factor gives us **3 mins** to write a job.

Problem 2.

a) The distribution and output of these keys across 3 reducers using the default partitioner (% 3)
Partitioning for 3 reducers
R1: where mod 3 = 0
R2: mod 3=1
R3: mod 3 = 2

R1 = 6, 12, 15, 27, 51, 99
R2 = 1, 4, 16, 25, 52, 58
R3 = 8, 11, 14, 26, 50, 89, 92

b) Custom partitioning
R1: where less than x
R2: where greater than or equal to x and less than y
R3: greater than or equal to y

R1: 1, 4, 6, 8, 11, 12, 14,15, 25, 26, 27
R2: 50, 51, 52, 58
R3: 89, 92, 99

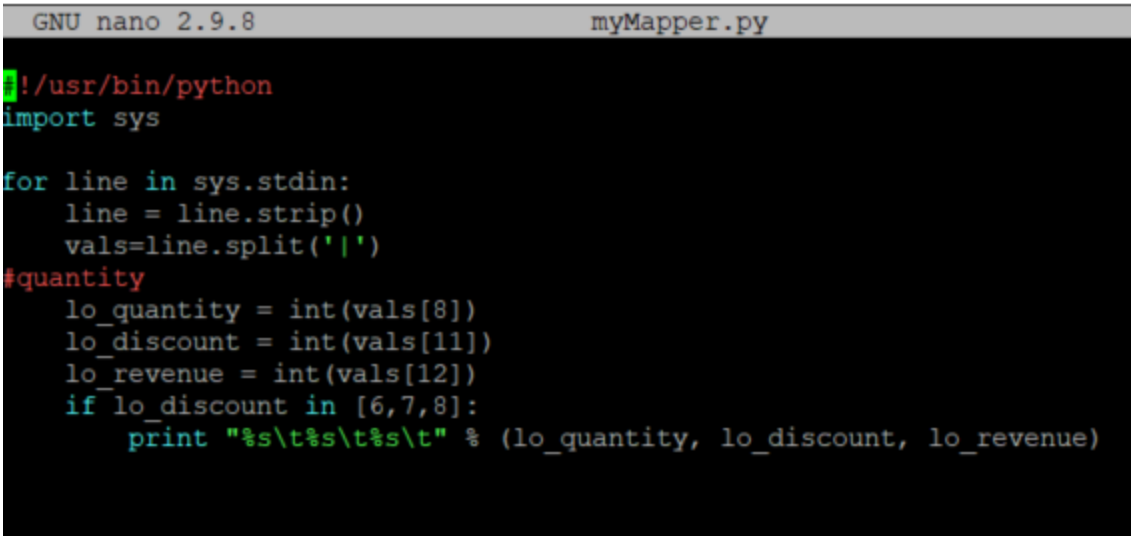
c) Custom partitioning may not equally sort the keys/values and user might be cramped for space on one partition but have plenty of free space on another, I think that is the downside of custom partitioning.

Problem 3.

myMapper.py

```
#!/usr/bin/python
import sys

for line in sys.stdin:
   line = line.strip()
   vals=line.split('|')
#quantity
   lo_quantity = int(vals[8])
   lo_discount = int(vals[11])
   lo_revenue = int(vals[12])
   if lo_discount in [6,7,8]:
      print "%s\t%s\t%s\t" % (lo_quantity, lo_discount, lo_revenue)
```



In reducer we parse string coming from mapper as Key-value pair, and keep them in dictionary like {quantity:[revenue,..]}

myReducer.py

```
#!/usr/bin/python

import sys

quant_rev = {}

for line in sys.stdin:
   line = line.strip()
   lo_quantity, lo_discount,  lo_revenue = line.split('\t')
   if lo_quantity in quant_rev:
      quant_rev[lo_quantity].append(int(lo_revenue))
   else:
      quant_rev[lo_quantity] = []
      quant_rev[lo_quantity].append(int(lo_revenue))
for lo_quantity in quant_rev.keys():
   ave_rev = sum(quant_rev[lo_quantity]) / len(quant_rev[lo_quantity])
   print '%s\t%s' % (lo_quantity, ave_rev)
```

```
#!/usr/bin/python

import sys

quant_rev = {}
#partitioner
for line in sys.stdin:
    line = line.strip()
    lo_quantity, lo_discount,  lo_revenue = line.split('\t')
    if lo_quantity in quant_rev:
        quant_rev[lo_quantity].append(int(lo_revenue))
    else:
        quant_rev[lo_quantity] = []
        quant_rev[lo_quantity].append(int(lo_revenue))

#reducer
for lo_quantity in quant_rev.keys():
    ave_rev = sum(quant_rev[lo_quantity]) / len(quant_rev[lo_quantity])
    print '%s\t%s' % (lo_quantity, ave_rev)
```

```
[ec2-user@ip-172-31-48-241 hadoop-2.6.4]$ cat lineorder.tbl | python myMapper.py
 | sort -n | python myReducer.py
42      5858283
48      6699379
43      6000194
49      6837399
24      3345892
25      3485642
26      3634084
27      3768283
20      2789022
21      2925041
22      3071083
23      3202331
46      6434834
47      6555244
44      6135599
45      6271572
28      3905890
29      4048916
40      5577892
41      5720788
1       139452
3       418738
2       278553
5       696104
4       557755
7       977779
6       837641
```

hadoop jar hadoop-streaming-2.6.4.jar -input /user/ec2-user/ssbm -output /data/assignment3 -mapper myMapper.py -reducer myReducer.py -file myReducer.py -file myMapper.py

```
[ec2-user@ip-172-31-48-241 hadoop-2.6.4]$ hadoop jar hadoop-streaming-2.6.4.jar
-input /user/ec2-user/ssbm -output /data/assignment3 -mapper myMapper.py -reduce
r myReducer.py -file myReducer.py -file myMapper.py
```

```
20/10/18 20:38:44 INFO streaming.StreamJob: Output directory: /data/assignment3
[ec2-user@ip-172-31-48-241 hadoop-2.6.4]$ hadoop fs -cat /data/assignment3/part-
00000
42      5858283
48      6699379
43      6000194
49      6837399
24      3345892
25      3485642
26      3634084
27      3768283
20      2789022
21      2925041
22      3071083
23      3202331
46      6434834
47      6555244
44      6135599
45      6271572
28      3905890
29      4048916
40      5577892
41      5720788
1       139452
3       418738
2       278553
5       696104
4       557755
7       977779
6       837641
9       1254213
8       1115301
39      5439082
38      5299180
11      1533974
10      1394592
13      1813827
12      1672518
15      2090607
14      1950263
17      2371535
16      2230922
19      2653429
18      2516921
31      4324416
30      4188699
37      5164615
36      5026968
35      4880643
34      4739361
33      4602964
32      4471392
50      6966732
```

Problem 4.

```
hbase(main):005:0> put 'employees','ID1','private:address','243 N. Wabash Av.'
0 row(s) in 0.0090 seconds

hbase(main):006:0> scan 'employees'
ROW                                   COLUMN+CELL
 ID1                                  column=private:address, timestamp=1603057819362, value=243 N. Wabash Av.
 ID1                                  column=private:ssn, timestamp=1603057713165, value=111-222-334
 ID2                                  column=private:ssn, timestamp=1603057747950, value=222-333-445
 ID3                                  column=private:address, timestamp=1603057785538, value=123 State St.
3 row(s) in 0.0540 seconds
```

2 new columns to private

```
hbase(main):007:0> put 'employees','ID4','private:name','Marc'
0 row(s) in 0.0050 seconds

hbase(main):008:0> put 'employees','ID4','private:age','25'
0 row(s) in 0.0150 seconds

hbase(main):009:0> put 'employees','ID2','private:name','Lily'
0 row(s) in 0.0130 seconds

hbase(main):010:0> put 'employees','ID5','private:age','23'
0 row(s) in 0.0150 seconds
```

1 new column to public family

```
hbase(main):011:0> put 'employees','ID3','public:ssn','111-222-334'
0 row(s) in 0.0110 seconds

hbase(main):012:0> put 'employees','ID1','public:address','123 State St.'
0 row(s) in 0.0100 seconds

hbase(main):013:0> scan 'employees'
ROW                                   COLUMN+CELL
 ID1                                  column=private:address, timestamp=1603057819362, value=243 N. Wabash Av.
 ID1                                  column=private:ssn, timestamp=1603057713165, value=111-222-334
 ID1                                  column=public:address, timestamp=1603058524539, value=123 State St.
 ID2                                  column=private:name, timestamp=1603058358078, value=Lily
 ID2                                  column=private:ssn, timestamp=1603057747950, value=222-333-445
```

create a brand new family with at least 3 columns.

disable 'employees'
alter 'employees','code'
enable 'employees'

```
hbase(main):017:0> put 'employees','ID1','code:name','Adam'
0 row(s) in 0.0070 seconds

hbase(main):018:0> put 'employees','ID3','code:name','Janet'
0 row(s) in 0.0150 seconds

hbase(main):019:0> put 'employees','ID5','code:name','Paul'
0 row(s) in 0.0100 seconds

hbase(main):020:0> put 'employees','ID2','code:phone','111-222-333'
0 row(s) in 0.0080 seconds

hbase(main):021:0> put 'employees','ID4','code:phone','222-333-444'
0 row(s) in 0.0170 seconds

hbase(main):022:0> put 'employees','ID6','code:phone','333-444-555'
0 row(s) in 0.0160 seconds

hbase(main):023:0> scan 'employees'
ROW                             COLUMN+CELL
 ID1                            column=code:name, timestamp=1603058962620, value=Adam
 ID1                            column=private:address, timestamp=1603057819362, value=243 N. Wabash Av.
 ID1                            column=private:ssn, timestamp=1603057713165, value=111-222-334
 ID1                            column=public:address, timestamp=1603058524539, value=123 State St.
 ID2                            column=code:phone, timestamp=1603059075624, value=111-222-333
 ID2                            column=private:name, timestamp=1603058358078, value=Lily
 ID2                            column=private:ssn, timestamp=1603057747950, value=222-333-445
 ID3                            column=code:name, timestamp=1603059003911, value=Janet
 ID3                            column=private:address, timestamp=1603057785538, value=123 State St.
 ID3                            column=public:ssn, timestamp=1603058486847, value=111-222-334
 ID4                            column=code:phone, timestamp=1603059121287, value=222-333-444
 ID4                            column=private:age, timestamp=1603058322107, value=25
 ID4                            column=private:name, timestamp=1603058287460, value=Marc
 ID5                            column=code:name, timestamp=1603059034678, value=Paul
 ID5                            column=private:age, timestamp=1603058393579, value=23
 ID6                            column=code:phone, timestamp=1603059137112, value=333-444-555
6 row(s) in 0.0520 seconds
```