Azat Dovgeldiyev

CSC 555

Phase 2 Queries (Take-home final)

Part 1: Querying

Implement the following query:

```
select c_nation, MAX(lo_extendedprice)
from customer, lineorder
where lo_custkey = c_custkey
  and c_region = 'AFRICA'
  and lo_discount = 6
group by c_nation;
```

using MapReduce with HadoopStreaming and Pig (2 different solutions). In Hadoop streaming, this will require a total of 2 passes (one for join and another one for GROUP BY).

Hadoop Streaming

phaseMap.py

```
  GNU nano 2.9.8                    phaseMap.py

#!/usr/bin/python

import sys
c_dict = {}
f = open('customer.tbl', 'r')
exclude = map(lambda x:x.split('|'), f.readlines())

for line in exclude:
    c_custkey = line[0]
    c_nation = line[4]
    c_region = line[5]
    if c_region == 'AFRICA':
        c_dict[c_custkey]=c_nation
f.close

for line in sys.stdin:
    line = line.strip()
    vals = line.split('|')
    lo_custkey = vals[2]
                          [ Read 23 lines ]
```

phaseReduce.py

```
  GNU nano 2.9.8                    phaseReduce.py

#!/usr/bin/python
#from __future__ import division

import sys

cur_name = None
cur_price = []
name = None

for line in sys.stdin:
    line = line.strip()
    ln = line.split('\t')

    name = ln[0]
    value = int(ln[1])

    if cur_name == name:
        cur_price.append(value)
        #cur_count += 1.0
    else:
        if cur_name:
            print "%s\t%f" % (cur_name,max(cur_price))
        cur_name = name
        cur_price = [value]
        #cur_count = 1.0
if cur_name == name:
    print "%s\t%f" % (cur_name, max(cur_price))
```

hadoop jar hadoop-streaming-2.6.4.jar -input /user/ec2-user/phase2/lineorder.tbl -output /data/phaseHadoop -mapper phaseMap.py -reducer phaseReduce.py -file phaseReduce.py -file phaseMap.py -file customer.tbl

```
        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=594329385
        File Output Format Counters
                Bytes Written=122
20/11/16 21:26:12 INFO streaming.StreamJob: Output directory: /data/phaseHadoop
[ec2-user@ip-172-31-74-226 hadoop-2.6.4]$ hadoop fs -ls /data/phaseHadoop
Found 2 items
-rw-r--r--   2 ec2-user supergroup          0 2020-11-16 21:26 /data/phaseHadoop
/_SUCCESS
-rw-r--r--   2 ec2-user supergroup        122 2020-11-16 21:26 /data/phaseHadoop
/part-00000
[ec2-user@ip-172-31-74-226 hadoop-2.6.4]$ hadoop fs -cat /data/phaseHadoop/part-
00000
ALGERIA 10314850.000000
ETHIOPIA        10384900.000000
KENYA   10364850.000000
MOROCCO 10464950.000000
MOZAMBIQUE      10244850.000000
```

Pig

customer = LOAD '/user/ec2-user/phase2/customer.tbl' USING PigStorage ('|') AS (c_custkey:int, c_name:chararray, c_address:chararray, c_city: chararray, c_nation:chararray, c_region:chararray, c_phone:chararray, c_mktsegment:chararray);

lod = LOAD '/user/ec2-user/phase2/lineorder.tbl' USING PigStorage('|') AS(lo_orderkey:int,lo_linenumber:int, lo_custkey:int, lo_partkey:int, lo_suppkey:int, lo_orderdate:int, lo_orderpriority:chararray, lo_shippriority:chararray, lo_quantity:int, lo_extendedprice:int, lo_ordertotalprice:int, lo_discount:int, lo_revenue:int,lo_supplycost:int, lo_tax:int, lo_commitdate:int, lo_shipmode:chararray);

c_region_AFRICA = FILTER customer by c_region == 'AFRICA';
lodiscount_6 = FILTER lod BY lo_discount == 6;
joinTable = JOIN c_region_AFRICA BY c_custkey, lodiscount_6 BY lo_orderkey;
nation_group = Group joinTable by c_nation;
data = FOREACH nation_group GENERATE group, joinTable.c_nation, MAX(joinTable.lo_extendedprice) as maxPrice;
dump data;

```
(KENYA,9284667)
(ALGERIA,9676608)
(MOROCCO,8638270)
(ETHIOPIA,9494050)
(MOZAMBIQUE,8914095)
2020-11-23 18:31:31,310 [main] INFO  org.apache.pig.Main - Pig script completed
in 1 minute, 12 seconds and 582 milliseconds (72582 ms)
[ec2-user@ip-172-31-74-226 pig-0.15.0]$
```

Part 2: Clustering

Centers.txt initial points

```
1 4 28 6 13 7 32 88 75 36 102
2 53 30 84 35 3 46 52 14 18 40
3 112 40 338 36 60 103 41 58 23 66
4 10 8 54 17 14 55 11 203 114 64
5 14 16 5 90 43 98 13 39 51 115
```

kmeansMap.py

```
#!/usr/bin/python

import sys
import math

# if given centers.txt with -file centers.txt at command line. must be changed for other inputs
fd = open('centers.txt', 'r')
centers = fd.readlines()
fd.close()

keys = [] # create keys list
for i in range(len(centers)):
    center = centers[i].strip().split()
    center_pts = [int(x) for x in center[1:]] # a list of int points
    keys.append(center[0]) # add center key to list
    centers[i] = center_pts

for line in sys.stdin:
    line = line.strip().split()
    points = [int(x) for x in line] # a list of int points

    p_min = 10000
    key = None
    # compute the closest center using euclidean distance in 3-dimensional space
    for i in range(len(centers)):
        # find the euclidean distance from next center point
        tmp_key = keys[i]
        tmp_min = math.sqrt(sum([(a - b) ** 2 for a, b in zip(points, centers[i][1:])]))
        if tmp_min <  p_min: # if center point is closer to point, update
            p_min = tmp_min
```

```
        key = tmp_key
    print '%s\t%d\t%d\t%d\t%d\t%d\t%d\t%d\t%d\t%d\t%d' % (key, points[0], points[1], points[2],points[3],points[4],points[5],points[6],points[7],points[8],points[9])
```

```
#!/usr/bin/python

import sys
import math

# if given centers.txt with -file centers.txt at command line. must be changed for other inputs
fd = open('centers.txt', 'r')
centers = fd.readlines()
fd.close()

keys = [] # create keys list
for i in range(len(centers)):
    center = centers[i].strip().split()
    center_pts = [int(x) for x in center[1:]] # a list of int points
    keys.append(center[0]) # add center key to list
    centers[i] = center_pts

for line in sys.stdin:
    line = line.strip().split()
    points = [int(x) for x in line] # a list of int points

    p_min = 10000
    key = None
    # compute the closest center using euclidean distance in 3-dimensional space
    for i in range(len(centers)):
        # find the euclidean distance from next center point
        tmp_key = keys[i]
        tmp_min = math.sqrt(sum([(a - b) ** 2 for a, b in zip(points, centers[i][1:])]))
        if tmp_min <  p_min: # if center point is closer to point, update
            p_min = tmp_min
            key = tmp_key
    print '%s\t%d\t%d\t%d\t%d\t%d\t%d\t%d\t%d\t%d\t%d' % (key, points[0], points[1], points[2],points[3],points[4],points[5],points[6],points[7],points[8],points[9])
```

kReduce.py

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

curr_key = None
key = None
center = None
total = 0

# The input comes from standard input (line by line)
for line in sys.stdin:
    line = line.strip()
    # parse the line and split it by '\t'
    ln = line.split('\t')
    # grab the key
    key = ln[0]
    points = [int(x) for x in ln[1:]] # get int point values
```

```
    #print("points" + points)
    if curr_key == key:
        # running totals for calculating new centers
        center = [a + b for a, b in zip(points, center)]
        total += 1
    else:
        if curr_key:
            # calculate the floor division value for the new center point
            center = [y // total for y in center]
            #print("center" + center)
            print "%s\t%d\t%d\t%d\t%d\t%d\t%d\t%d\t%d\t%d\t%d" % (curr_key, center[0],
center[1], center[2], center[3],center[4],center[5],center[6],center[7],center[8],center[9])
        curr_key = key
        center = points
        total += 1

# output the last key
if curr_key == key:
    if curr_key:
        # calculate the floor division value for the new center point
        center = [y // total for y in center]
        print "%s\t%d\t%d\t%d\t%d\t%d\t%d\t%d\t%d\t%d\t%d" % (curr_key, center[0],
center[1], center[2], center[3],center[4],center[5],center[6],center[7],center[8],center[9])
```

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

curr_key = None
key = None
center = None
total = 0

# The input comes from standard input (line by line)
for line in sys.stdin:
    line = line.strip()
    # parse the line and split it by '\t'
    ln = line.split('\t')
    # grab the key
    key = ln[0]
    points = [int(x) for x in ln[1:]] # get int point values
    #print("points" + points)
    if curr_key == key:
        # running totals for calculating new centers
        center = [a + b for a, b in zip(points, center)]
        total += 1
    else:
        if curr_key:
            # calculate the floor division value for the new center point
            center = [y // total for y in center]
            #print("center" + center)
            print "%s\t%d\t%d\t%d\t%d\t%d\t%d\t%d\t%d\t%d" % (curr_key, center[0], center[1], center[2], center[3],center[4],center[5],center[6],center[7],center[8],center[9])
        curr_key = key
        center = points
        total += 1

# output the last key
if curr_key == key:
    if curr_key:
        # calculate the floor division value for the new center point
        center = [y // total for y in center]
        print "%s\t%d\t%d\t%d\t%d\t%d\t%d\t%d\t%d\t%d" % (curr_key, center[0], center[1], center[2], center[3],center[4],center[5],center[6],center[7],center[8],center[9])
```

hadoop jar hadoop-streaming-2.6.4.jar -input /user/ec2-user/numbers/Numbers.txt -file centers.txt -mapper kmeansMap.py -file kmeansMap.py -reducer kmeansRed.py -file kmeansRed.py -output /data/kMeans111

```
        File Input Format Counters
                Bytes Read=13695730
        File Output Format Counters
                Bytes Written=170
20/11/25 03:33:37 INFO streaming.StreamJob: Output directory: /data/kMeans111
[ec2-user@ip-172-31-74-226 hadoop-2.6.4]$ cat Numbers.txt | python kmeansMap.py | sort | python kmeansRed.py
1       603     114     292     381     266     899     346     368     624     543
2       0       0       0       0       0       0       0       0       0       0
3       549     686     533     552     550     548     505     511     528     538
4       106     47      101     101     100     106     168     139     106     113
5       53      13      72      58      61      51      32      53      67      56
```

Remove original centers and replace with newly generated
rm centers.txt
hadoop fs -get /data/kmeans111/part-00000 centers.txt

Second iteration:

hadoop jar hadoop-streaming-2.6.4.jar -input /user/ec2-user/numbers/Numbers.txt -file centers.txt -mapper kmeansMap.py -file kmeansMap.py -reducer kmeansRed.py -file kmeansRed.py -output /data/kMeans8

```
20/11/25 04:03:30 INFO streaming.StreamJob: Output directory: /data/kMeans8
[ec2-user@ip-172-31-74-226 hadoop-2.6.4]$ hadoop fs -ls /data/kMeans8/
Found 2 items
-rw-r--r--   2 ec2-user supergroup          0 2020-11-25 04:03 /data/kMeans8/_SUCCESS
-rw-r--r--   2 ec2-user supergroup        128 2020-11-25 04:03 /data/kMeans8/part-00000
[ec2-user@ip-172-31-74-226 hadoop-2.6.4]$ hadoop fs -cat /data/kMeans8/part-00000
1       124     233     256     220     409     264     261     324     304     293
3       270     246     241     249     211     239     240     227     232     231
4       2       3       3       3       2       4       4       3       3       6
5       0       0       0       0       0       0       0       0       0       0
```

Third iteration:

hadoop jar hadoop-streaming-2.6.4.jar -input /user/ec2-user/numbers/Numbers.txt -file centers.txt -mapper kmeansMap.py -file kmeansMap.py -reducer kmeansRed.py -file kmeansRed.py -output /data/kMeans333

```
                Bytes Written=105
20/11/25 04:08:59 INFO streaming.StreamJob: Output directory: /data/kMeans333
[ec2-user@ip-172-31-74-226 hadoop-2.6.4]$ hadoop fs -ls /data/kMeans333/
Found 2 items
-rw-r--r--   2 ec2-user supergroup          0 2020-11-25 04:08 /data/kMeans333/_SUCCESS
-rw-r--r--   2 ec2-user supergroup        105 2020-11-25 04:08 /data/kMeans333/part-00000
[ec2-user@ip-172-31-74-226 hadoop-2.6.4]$ hadoop fs -cat /data/kMeans333/part-00000
1       286     299     278     409     304     301     338     326     321     292
3       144     137     148     80      135     136     117     123     126     141
4       0       0       0       0       0       0       0       0       0       0
[ec2-user@ip-172-31-74-226 hadoop-2.6.4]$
```

Fourth iteration:

hadoop jar hadoop-streaming-2.6.4.jar -input /user/ec2-user/numbers/Numbers.txt -file centers.txt -mapper kmeansMap.py -file kmeansMap.py -reducer kmeansRed.py -file kmeansRed.py -output /data/kMeans444

```
                Bytes Written=96
20/11/25 04:11:37 INFO streaming.StreamJob: Output directory: /data/kMeans444
[ec2-user@ip-172-31-74-226 hadoop-2.6.4]$ hadoop fs -cat /data/kMeans444/part-00000
1       303     301     314     303     303     307     306     305     303     292
3       28      30      18      28      28      24      25      25      28      37
4       0       0       0       0       0       0       0       0       0       0
[ec2-user@ip-172-31-74-226 hadoop-2.6.4]$
```

After second iteration I got zero values and the last iteration yielded 2 centers only.