Hadoop Streaming – Join and Aggregate

1st Map Reduce

myMapper_join.py

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

# input comes from STDIN (standard input)
for line in sys.stdin:
        line = line.strip()
        split = line.split('|')
        if split[1].startswith('Customer'): #Mapper1, Customer
                if split[5] == 'AFRICA':
                        print "%d\t%s\t%s"% (int(split[0]),split[4],'Customer')
        else:                                   #Mapper2, lineorder
                if split[11] == '6':
                print "%s\t%s\t%s"% (int(split[2]),split[9],'Lineorder')
```

myReducer_join.py

```
#!/usr/bin/python

import sys
import itertools

currentKey = None
valsLineorder = []
valsCustomer = []
lineordervalue = None
customervalue = None
cnation = None
extendedprice = None

# input comes from STDIN
for line in sys.stdin:

        split = line.strip().split('\t')    # 'Q11 \t Val1 \t Val2 \t Val3'
        key = split[0]
        line_value = '\t'.join(split[1:])

        if currentKey == key:  # Same key
                if line_value.endswith('Customer'):
                        customervalue = line_value.strip().split('\t')
                        cnation = customervalue[0]
                        valsCustomer.append(cnation)
                if line_value.endswith('Lineorder'):
                        lineordervalue = line_value.strip().split('\t')
                        extendedprice = lineordervalue[0]
                        valsLineorder.append(extendedprice)
```

```
        else:
                if currentKey:
                        lenLineorder = len(valsLineorder)
                        lenCustomer = len(valsCustomer)
                        if (lenLineorder*lenCustomer > 0):
                                for i in valsCustomer:
                                        for j in valsLineorder:
                                                print '%s\t%s' %(i,j)
                currentKey = key
                valsLineorder = []
                valsCustomer = []
                cnation = None
                extendedprice = None
                if line_value.endswith('Customer'):
                        customervalue = line_value.strip().split('\t')
                        cnation = customervalue[0]
                        valsCustomer.append(cnation)
                elif line_value.endswith('Lineorder'):
                        lineordervalue = line_value.strip().split('\t')
                        extendedprice = lineordervalue[0]
                        valsLineorder.append(extendedprice)
lenLineorder = len(valsLineorder)
lenCustomer = len(valsCustomer)
if (lenLineorder*lenCustomer > 0):
        for i in valsCustomer:
                for j in valsLineorder:
                        print '%s\t%s' %(i,j)
```

2nd Map Reduce

myMapper_agg.py

```
#!/usr/bin/python

import sys

for line in sys.stdin:
    line = line.strip()
    vals = line.split("\t")
    print "%s\t%d" % (vals[0], int(vals[1]))  # 123 456
```

myReducer_agg.py

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

curr_id = None
id = None
valsExtendedPrice = []
# 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')     # [1, 5]
    # grab the key
    id = ln[0]   # current received key is cnation
    if curr_id == id:
        valsExtendedPrice.append(int(ln[1]))
    else:
        if curr_id: # output the count, single key completed
            # NOTE: Change this to '%s\t%d' if your key is a string
            print '%s\t%d' % (curr_id, max(valsExtendedPrice))  # print 1\t2 if you saw two 1s
        curr_id = id   # Reset the current key to the new key (e.g., 6)
        valsExtendedPrice = []
        valsExtendedPrice.append(int(ln[1]))

# output the last key
if curr_id == id:
    print '%s\t%d' % (curr_id, max(valsExtendedPrice))
```

Command (1st Map Reduce)

hadoop jar hadoop-streaming-2.6.4.jar -input /data/joinLineorderCustomer/ -mapper myMapper_join.py -file ../myMapper_join.py -reducer myReducer_join.py -file ../myReducer_join.py -output /data/LCoutput

```
                Data-local map tasks=7
                Total time spent by all maps in occupied slots (ms)=88303
                Total time spent by all reduces in occupied slots (ms)=18382
                Total time spent by all map tasks (ms)=88303
                Total time spent by all reduce tasks (ms)=18382
                Total vcore-milliseconds taken by all map tasks=88303
                Total vcore-milliseconds taken by all reduce tasks=18382
                Total megabyte-milliseconds taken by all map tasks=90422272
                Total megabyte-milliseconds taken by all reduce tasks=18823168
        Map-Reduce Framework
                Map input records=6031215
                Map output records=550970
                Map output bytes=12945599
                Map output materialized bytes=14047575
                Input split bytes=677
                Combine input records=0
                Combine output records=0
                Reduce input groups=22009
                Reduce shuffle bytes=14047575
                Reduce input records=550970
                Reduce output records=107884
                Spilled Records=1101940
                Shuffled Maps =6
                Failed Shuffles=0
                Merged Map outputs=6
                GC time elapsed (ms)=557
                CPU time spent (ms)=12910
                Physical memory (bytes) snapshot=1783775232
                Virtual memory (bytes) snapshot=14892220416
                Total committed heap usage (bytes)=1265631232
        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=597166431
        File Output Format Counters
                Bytes Written=1755787
20/11/19 06:31:29 INFO streaming.StreamJob: Output directory: /data/LCoutput
```

Command (2nd Map Reduce)

hadoop jar hadoop-streaming-2.6.4.jar -input /data/LCoutput -mapper myMapper_agg.py -file ../myMapper_agg.py -reducer myReducer_agg.py -file ../myReducer_agg.py -output /data/LCoutput2

```
                Data-local map tasks=2
                Total time spent by all maps in occupied slots (ms)=8719
                Total time spent by all reduces in occupied slots (ms)=2735
                Total time spent by all map tasks (ms)=8719
                Total time spent by all reduce tasks (ms)=2735
                Total vcore-milliseconds taken by all map tasks=8719
                Total vcore-milliseconds taken by all reduce tasks=2735
                Total megabyte-milliseconds taken by all map tasks=8928256
                Total megabyte-milliseconds taken by all reduce tasks=2800640
        Map-Reduce Framework
                Map input records=107884
                Map output records=107884
                Map output bytes=1755787
                Map output materialized bytes=1971567
                Input split bytes=194
                Combine input records=0
                Combine output records=0
                Reduce input groups=5
                Reduce shuffle bytes=1971567
                Reduce input records=107884
                Reduce output records=5
                Spilled Records=215768
                Shuffled Maps =2
                Failed Shuffles=0
                Merged Map outputs=2
                GC time elapsed (ms)=261
                CPU time spent (ms)=3200
                Physical memory (bytes) snapshot=693125120
                Virtual memory (bytes) snapshot=6387449856
                Total committed heap usage (bytes)=495976448
        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=1759883
        File Output Format Counters
                Bytes Written=87
20/11/19 06:36:59 INFO streaming.StreamJob: Output directory: /data/LCoutput2
```

Ouput:

hadoop fs -cat /data/LCoutput2/part-00000

```
[ec2-user@ip-172-31-77-124 hadoop-2.6.4]$ hadoop fs -cat /data/LCoutput2/part-00000
ALGERIA 10314850
ETHIOPIA        10384900
KENYA   10364850
MOROCCO 10464950
MOZAMBIQUE      10244850
```