Table of Contents

In [1]:
```
%%javascript
/*******************************************************************************
****************
Known Mathjax Issue with Chrome - a rounding issue adds a border to the right o
f mathjax markup
https://github.com/mathjax/MathJax/issues/1300
A quick hack to fix this based on stackoverflow discussions:
http://stackoverflow.com/questions/34277967/chrome-rendering-mathjax-equations-
with-a-trailing-vertical-line
*******************************************************************************
***************/

$('.math>span').css("border-left-color","transparent")
```

In [2]:
```
%reload_ext autoreload
%autoreload 2
```

MIDS - w261 Machine Learning At Scale

Course Lead: Dr James G. Shanahan (**email** Jimi via James.Shanahan *AT* gmail.com)

Assignment - HW3

Name: Nilesh Bhoyat **Class:** MIDS w261 Summer Group 2
Email: *Your UC Berkeley Email Goes Here*@iSchool.Berkeley.edu
StudentId 26302327 **End of StudentId**
Week: 3

NOTE: please replace `1234567` with your student id above

Due Time: HW is due the Tuesday of the following week by 8AM (West coast time). I.e., Tuesday, Jan 31, 2017 in the case of this homework.

Table of Contents

1 Instructions

Back to Table of Contents

MIDS UC Berkeley, Machine Learning at Scale DATSCIW261 ASSIGNMENT #3

Version 2017-26-1

IMPORTANT

This homework can be completed locally on your computer

=== INSTRUCTIONS for SUBMISSIONS ===

Follow the instructions for submissions carefully.

NEW: Going forward, each student will have a `HW-<user>` repository for all assignments.

Click this link to enable you to create a github repo within the MIDS261 Classroom:
https://classroom.github.com/assignment-invitations/3b1d6c8e58351209f9dd865537111ff8
(https://classroom.github.com/assignment-invitations/3b1d6c8e58351209f9dd865537111ff8)
and follow the instructions to create a HW repo.

Push the following to your HW github repo into the master branch:

- Your local HW3 directory. Your repo file structure should look like this:

```
HW-<user>
    --HW3
       |__MIDS-W261-HW-03-<Student_id>.ipnb
       |__MIDS-W261-HW-03-<Student_id>.pdf
       |__some other hw3 file
    --HW4
       |__MIDS-W261-HW-04-<Student_id>.ipnb
       |__MIDS-W261-HW-04-<Student_id>.pdf
       |__some other hw4 file
    etc..
```

HW3.0.

1. How do you merge two sorted lists/arrays of records of the form [key, value]?
2. Where is this used in Hadoop MapReduce? [Hint within the shuffle]
3. What is a combiner function in the context of Hadoop?
4. Give an example where it can be used and justify why it should be used in the context of this problem.
5. What is the Hadoop shuffle?

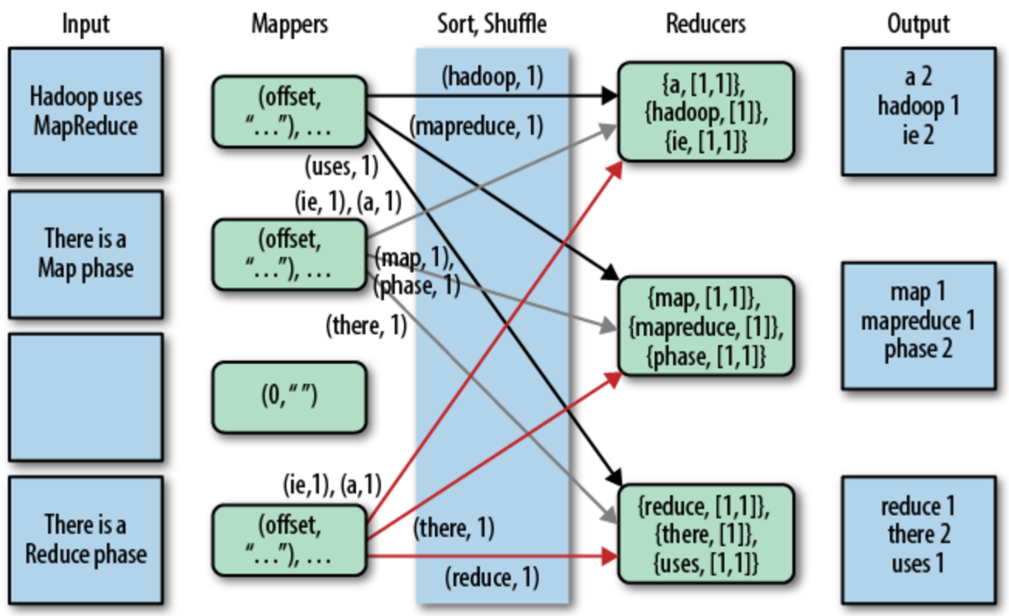
How do you merge two sorted lists/arrays of records of the form [key, value]?

```
Merge sort is  divide and conquer algorithm. It is a very efficient sort algorithm.
The algorithm gets is named from the fact that it divides the collection in half, r
ecursively sorts each half, and then merges the two sorted halves back together. Ea
ch half of the collection is repeatedly halved until there is only one object in th
e half, at which point it is sorted by definition. As each sorted half is merged, t
he algorithm compares the objects to determine where to place each sub set.

for records in format [key,value], operations happen on keys.
Pseudo Code


Union of keys from both dictionaries.
Loop for each key in union:
  if key is in dict1 and not in dict 2 then
   add key records in result dict
  if key is in dict2 and not in dict 1 then
  add key record in result dict
  if key is present in both dicts then
  union of result with same key
```

In [3]:
```python
from collections import defaultdict
dict1 = {'bookA': 1, 'bookB': 2, 'bookC': 3}
dict2 = {'bookC': 2, 'bookD': 4, 'bookE': 5}

def union_collections(d1, d2):
    union = {}

    for key in set(d1.keys()).union(d2.keys()):
        if key in d1 and key not in d2:
            union[key] = d1[key]

        if key in d2 and key not in d1:
            union[key] = d2[key]

        if key in d1 and key in d2:
            union[key] = (d1[key] ,d2[key])

    return union
union_collections(dict1, dict2)
```

Out[3]:
```
{'bookA': 1, 'bookB': 2, 'bookC': (3, 2), 'bookD': 4, 'bookE': 5}
```

Where is this used in Hadoop MapReduce? [Hint within the shuffle]



```
Answer:
As shown in figure, key-value pairs are merge-sort is used in shuffle-sort phase.
Hadoop sorts the key-value pairs by key and it “shuffles” all pairs with the same k
ey to the same Reducer. There are several possible techniques that can be used to d
ecide which reducer gets which range of keys.
```

What is a combiner function in the context of Hadoop?

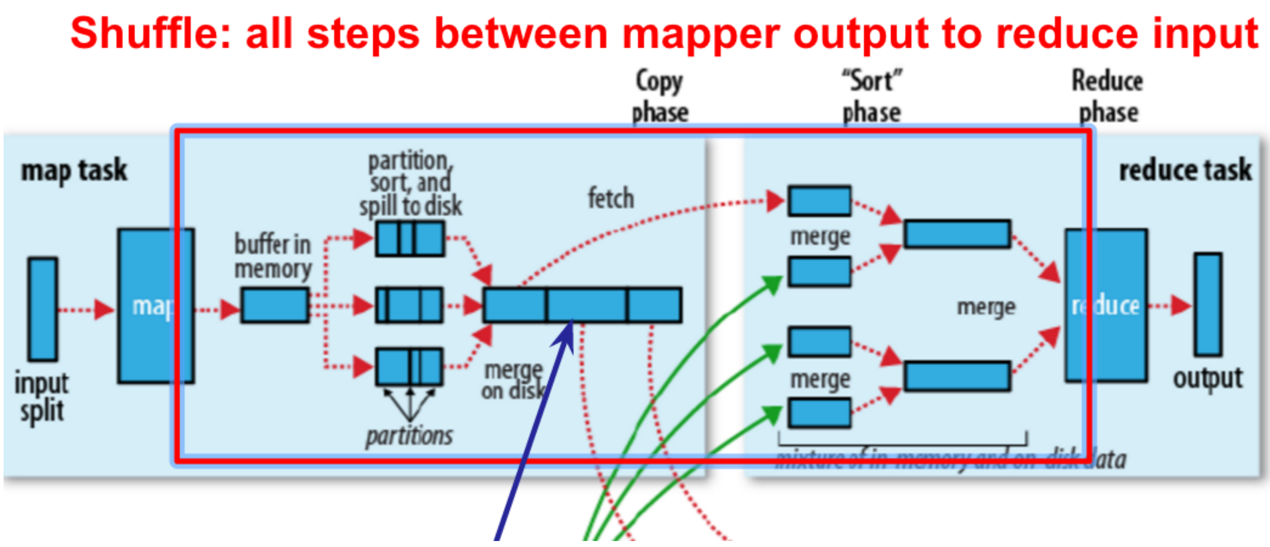
```
Mappers produce a lot of intermediate data that must be sent over the network to be
shuffled, sorted, and reduced. Because networking is a physical resource, large amo
unts of transmitted data can lead to job delays and memory bottlenecks (e.g., there
is too much data for the reducer to hold into memory). Combiners are the primary me
chanism to solve this problem, and are essentially intermediate reducers that are a
ssociated with the mapper output. Combiners reduce network traffic by performing a
mapper-local reduction of the data before forwarding it on to the appropriate reduc
er.
```

Give an example where it can be used and justify why it should be used in the context of this problem.

```
 Answer

 Consider an example of wordcount for large corpora. When mulitple mappers are prod
ucing word count as below

 mapper 1 output
 (a,10) (the,10) (tent,1) (the,20)(a,20)
 Mapper 2 output

 (a,30) (the,10)

 Intended Sum Reduce is
 (a,60) (the,40)(tent,1)

 Each mapper is emitting extra work for the reducer, namely in the duplication of t
he different keys coming from each mapper. Combiner can can reduce such duplicate k
ey by aggreating records before and there by reduce network traffic. This will also
reduce overall shuffle phase time.
```

What is the Hadoop shuffle?

```
Answer
     The process by which the hadoop performs the sort—and transfers the map output
s to the reducers as inputs.
     If reducer has to take all output of mapper in as-is format i.e. (word,1) then
it would be very slow process.
     Shuffle does this
     Shuffle is all of this!:
    1. partition, sort, combine - Partitions records and does partial sort for each
partition.
    2. mergesort
    3. Send to reducer
    4. Merge sort
    5. Stream to reducer
```



HW3.1 consumer complaints dataset: Use Counters to do EDA (exploratory data analysis and to monitor progress)

Counters are lightweight objects in Hadoop that allow you to keep track of system progress in both the map and reduce stages of processing. By default, Hadoop defines a number of standard counters in "groups"; these show up in the jobtracker webapp, giving you information such as "Map input records", "Map output records", etc.

While processing information/data using MapReduce job, it is a challenge to monitor the progress of parallel threads running across nodes of distributed clusters. Moreover, it is also complicated to distinguish between the data that has been processed and the data which is yet to be processed. The MapReduce Framework offers a provision of user-defined Counters, which can be effectively utilized to monitor the progress of data across nodes of distributed clusters.

Use the Consumer Complaints Dataset provide here to complete this question:

```
https://www.dropbox.com/s/vbalm3yva2rr86m/Consumer_Complaints.csv?dl=0
```

The consumer complaints dataset consists of diverse consumer complaints, which have been reported across the United States regarding various types of loans. The dataset consists of records of the form:

Complaint ID,Product,Sub-product,Issue,Sub-issue,State,ZIP code,Submitted via,Date received,Date sent to company,Company,Company response,Timely response?,Consumer disputed?

Here's is the first few lines of the of the Consumer Complaints Dataset:

```
Complaint ID,Product,Sub-product,Issue,Sub-issue,State,ZIP code,Submitted via,Date
received,Date sent to company,Company,Company response,Timely response?,Consumer di
sputed?
1114245,Debt collection,Medical,Disclosure verification of debt,Not given enough in
fo to verify debt,FL,32219,Web,11/13/2014,11/13/2014,"Choice Recovery, Inc.",Closed
with explanation,Yes,
1114488,Debt collection,Medical,Disclosure verification of debt,Right to dispute no
tice not received,TX,75006,Web,11/13/2014,11/13/2014,"Expert Global Solutions, Inc.
",In progress,Yes,
1114255,Bank account or service,Checking account,Deposits and withdrawals,,NY,11102
,Web,11/13/2014,11/13/2014,"FNIS (Fidelity National Information Services, Inc.)",In
progress,Yes,
1115106,Debt collection,"Other (phone, health club, etc.)",Communication tactics,Fr
equent or repeated calls,GA,31721,Web,11/13/2014,11/13/2014,"Expert Global Solution
s, Inc.",In progress,Yes,
```

User-defined Counters

Now, let's use Hadoop Counters to identify the number of complaints pertaining to debt collection, mortgage and other categories (all other categories get lumped into this one) in the consumer complaints dataset. Basically produce the distribution of the Product column in this dataset using counters (limited to 3 counters here).

Hadoop offers Job Tracker, an UI tool to determine the status and statistics of all jobs. Using the job tracker UI, developers can view the Counters that have been created. Screenshot your job tracker UI as your job completes and include it here. Make sure that your user defined counters are visible.

In [4]:
```
# Put the data into HDFS
!wget 'https://www.dropbox.com/s/vbalm3yva2rr86m/Consumer_Complaints.csv'
```

```
--2017-05-30 05:53:50--  https://www.dropbox.com/s/vbalm3yva2rr86m/Consumer_Co
mplaints.csv
Resolving www.dropbox.com... 162.125.4.1
Connecting to www.dropbox.com|162.125.4.1|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://dl.dropboxusercontent.com/content_link/bySujatrsT8oJft8X7tWO
kDd81kv2gyO533Cjc3drrnM5QzbM5ET3cDIZ62ValFT/file [following]
--2017-05-30 05:53:57--  https://dl.dropboxusercontent.com/content_link/bySuja
trsT8oJft8X7tWOkDd81kv2gyO533Cjc3drrnM5QzbM5ET3cDIZ62ValFT/file
Resolving dl.dropboxusercontent.com... 162.125.4.6
Connecting to dl.dropboxusercontent.com|162.125.4.6|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 50906486 (49M) [text/csv]
Saving to: `Consumer_Complaints.csv'

100%[======================================>] 50,906,486  1.23M/s   in 92s

2017-05-30 05:55:32 (541 KB/s) - `Consumer_Complaints.csv' saved [50906486/509
06486]
```

In [5]:
```
# Create HDFS directories
!hdfs dfs -mkdir -p /user/nibhoyar
!hdfs dfs -put Consumer_Complaints.csv /user/nibhoyar
!hdfs dfs -rm Consumer_Complaints.csv
!hdfs dfs -copyFromLocal Consumer_Complaints.csv
!hdfs dfs -rm -r hw3.1-output
```

```
put: `/user/nibhoyar/Consumer_Complaints.csv': File exists
Deleted Consumer_Complaints.csv
Deleted hw3.1-output
```

In [6]:
```
%%writefile complaintCountsMapper.py
#!/usr/bin/env python
# START STUDENT CODE HW31MAPPER
import sys
separator = ','
for line in (sys.stdin):
        fields = line.split(separator)
        if 'Complaint ID' != fields[0] :

            # we have a real record, so do some mapping
            counter_name = None
            if (fields[1].lower() == 'debt collection' or \
                fields[1].lower() == 'mortgage'):
                counter_name = fields[1].strip().lower()
            else:
                counter_name = 'other'
            # update the counter
            sys.stderr.write("reporter:counter:Category Counters,{0},1\n".forma
t(counter_name))


# END STUDENT CODE HW31MAPPER
```

```
Overwriting complaintCountsMapper.py
```

In [7]:
```
!chomd a+x complaintCountsMapper.py
```

```
/bin/sh: chomd: command not found
```

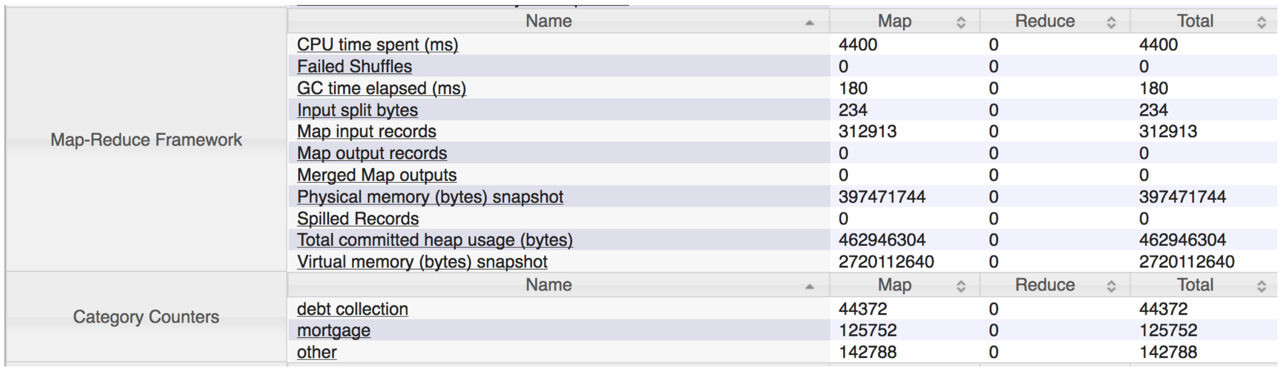
In [8]:
```
%%writefile complaintCountsReducer.py
#!/usr/bin/env python
# START STUDENT CODE HW31REDUCER

# END STUDENT CODE HW31REDUCER
```

```
Overwriting complaintCountsReducer.py
```

In [9]:
```
# Hadoop command
# START STUDENT CODE HW31HADOOP
!hdfs dfs -rm -r hw3.1-output
!hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming.jar \
    -D mapred.reduce.tasks=0 \
    -files complaintCountsMapper.py\
    -mapper complaintCountsMapper.py \
    -reducer org.apache.hadoop.mapred.lib.IdentityReducer\
    -input Consumer_Complaints.csv \
    -output  hw3.1-output
# END STUDENT CODE HW31HADOOP
```

```
rm: `hw3.1-output': No such file or directory
17/05/30 05:56:00 INFO Configuration.deprecation: mapred.reduce.tasks is depre
cated. Instead, use mapreduce.job.reduces
packageJobJar: [] [/usr/jars/hadoop-streaming-2.6.0-cdh5.7.0.jar] /tmp/streamj
ob1644182036203141886.jar tmpDir=null
17/05/30 05:56:02 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0
.0:8032
17/05/30 05:56:03 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0
.0:8032
17/05/30 05:56:06 INFO mapred.FileInputFormat: Total input paths to process :
1
17/05/30 05:56:07 INFO mapreduce.JobSubmitter: number of splits:2
17/05/30 05:56:08 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_
1496033164706_0086
17/05/30 05:56:09 INFO impl.YarnClientImpl: Submitted application application_
1496033164706_0086
17/05/30 05:56:09 INFO mapreduce.Job: The url to track the job: http://quickst
art.cloudera:8088/proxy/application_1496033164706_0086/
17/05/30 05:56:09 INFO mapreduce.Job: Running job: job_1496033164706_0086
17/05/30 05:56:25 INFO mapreduce.Job: Job job_1496033164706_0086 running in ub
er mode : false
17/05/30 05:56:25 INFO mapreduce.Job:  map 0% reduce 0%
17/05/30 05:56:36 INFO mapreduce.Job:  map 50% reduce 0%
17/05/30 05:56:37 INFO mapreduce.Job:  map 100% reduce 0%
17/05/30 05:56:37 INFO mapreduce.Job: Job job_1496033164706_0086 completed suc
cessfully
17/05/30 05:56:38 INFO mapreduce.Job: Counters: 33
        File System Counters
                FILE: Number of bytes read=0
                FILE: Number of bytes written=232896
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=50910816
                HDFS: Number of bytes written=0
                HDFS: Number of read operations=10
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=4
        Job Counters
                Launched map tasks=2
                Data-local map tasks=2
                Total time spent by all maps in occupied slots (ms)=18446
                Total time spent by all reduces in occupied slots (ms)=0
                Total time spent by all map tasks (ms)=18446
                Total vcore-seconds taken by all map tasks=18446
                Total megabyte-seconds taken by all map tasks=18888704
        Map-Reduce Framework
                Map input records=312913
                Map output records=0
                Input split bytes=234
                Spilled Records=0
                Failed Shuffles=0
                Merged Map outputs=0
                GC time elapsed (ms)=139
                CPU time spent (ms)=4320
                Physical memory (bytes) snapshot=392908800
                Virtual memory (bytes) snapshot=2711154688
                Total committed heap usage (bytes)=367525888
        Category Counters
                debt collection=44372
                mortgage=125752
                other=142788
        File Input Format Counters
                Bytes Read=50910582
        File Output Format Counters
                Bytes Written=0
17/05/30 05:56:38 INFO streaming.StreamJob: Output directory: hw3.1-output
```

output



	Name	Map	Reduce	Total
Map-Reduce Framework	CPU time spent (ms)	4400	0	4400
	Failed Shuffles	0	0	0
	GC time elapsed (ms)	180	0	180
	Input split bytes	234	0	234
	Map input records	312913	0	312913
	Map output records	0	0	0
	Merged Map outputs	0	0	0
	Physical memory (bytes) snapshot	397471744	0	397471744
	Spilled Records	0	0	0
	Total committed heap usage (bytes)	462946304	0	462946304
	Virtual memory (bytes) snapshot	2720112640	0	2720112640
Category Counters	Name	Map	Reduce	Total
	debt collection	44372	0	44372
	mortgage	125752	0	125752
	other	142788	0	142788

HW 3.2 Analyze the performance of your Mappers, Combiners and Reducers using Counters

For this brief study the Input file will be one record (the next line only):
`foo foo quux labs foo bar quux`

3.2.A
Perform a word count analysis of this single record dataset using a Mapper and Reducer based WordCount (i.e., no combiners are used here) using user defined Counters to count up how many times the mapper and reducer are called. What is the value of your user defined Mapper Counter, and Reducer Counter after completing this word count job. The answer should be 1 and 4 respectively. Please explain.

3.2.B
Please use mulitple mappers and reducers for these jobs (at least 2 mappers and 2 reducers). Perform a word count analysis of the Issue column of the Consumer Complaints Dataset using a Mapper and Reducer based WordCount (i.e., no combiners used anywhere) using user defined Counters to count up how many time the mapper and reducer are called. What is the value of your user defined Mapper Counter, and Reducer Counter after completing your word count job.

3.2.C
Perform a word count analysis of the Issue column of the Consumer Complaints Dataset using a Mapper, Reducer, and standalone combiner (i.e., not an in-memory combiner) based WordCount using user defined Counters to count up how many time the mapper, combiner, reducer are called. What is the value of your user defined Mapper Counter, and Reducer Counter after completing your word count job.

Using a single reducer:

- What are the top 50 most frequent terms in your word count analysis?
- Present the top 50 terms and their frequency and their relative frequency. If there are ties please sort the tokens in alphanumeric/string order.
- Present bottom 10 tokens (least frequent items).

NOTE: You can use: `WORD_RE = re.compile(r"[\w']+")` to tokenize.

3.2.A SOLUTION

In [10]:
```
%%writefile mapper3.2.A.py
#!/usr/bin/env python
# START STUDENT CODE HW32AMAPPER
import sys
import re

sys.stderr.write("reporter:counter:Mapper Counters,Calls,1\n")
WORD_RE = re.compile(r"[\w']+")
for line in sys.stdin:
    for word  in [s.lower() for s in WORD_RE.findall(line)]:
        print '%s\t%s' % (word, 1)

# END STUDENT CODE HW32AMAPPER
```

```
Overwriting mapper3.2.A.py
```

In [11]:
```
%%writefile reducer3.2.A.py
#!/usr/bin/env python
# START STUDENT CODE HW32AREDUCER
import sys

cur_key = None
cur_count = 0
sys.stderr.write("reporter:counter:Reducer Counters,Calls,1\n")
for line in sys.stdin:
    key, value = line.split()
    if key == cur_key:
        cur_count += int(value)
    else:
        if cur_key:
            print '%s\t%s' % (cur_key, cur_count)
        cur_key = key
        cur_count = int(value)

print '%s\t%s' % (cur_key, cur_count)

# END STUDENT CODE HW32AREDUCER
```

```
Overwriting reducer3.2.A.py
```

In [12]:
```
!echo "foo foo quux labs foo bar quux"|python mapper3.2.A.py|python reducer3.2.
A.py|sort -k2,2n
```

```
reporter:counter:Reducer Counters,Calls,1
reporter:counter:Mapper Counters,Calls,1
bar     1
foo     1
labs    1
quux    1
quux    1
foo     2
```

In [13]:
```
!chmod a+x mapper3.2.A.py
!chmod a+x reducer3.2.A.py
```

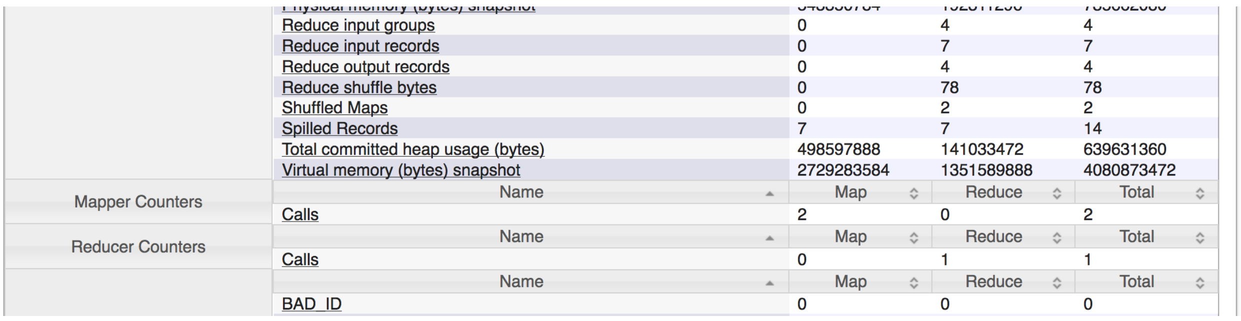
In [14]:
```
# Hadoop command
# START STUDENT CODE HW32AHADOOP

!echo "foo foo quux labs foo bar quux" >foofoo.txt
!hdfs dfs -copyFromLocal foofoo.txt
!hdfs dfs -rm -r hw3.2.A-output


!hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming.jar \
    -D mapred.output.key.comparator.class=org.apache.hadoop.mapred.lib.KeyField
BasedComparator \
    -D stream.num.map.output.key.fields=2 \
    -D stream.map.output.field.separator="\t" \
    -D mapreduce.partition.keycomparator.options="-k1,1nr -k2,2" \
    -files mapper3.2.A.py,reducer3.2.A.py\
    -mapper mapper3.2.A.py \
    -reducer reducer3.2.A.py\
    -input foofoo.txt \
    -output  hw3.2.A-output

# END STUDENT CODE HW32AHADOOP
```

```
copyFromLocal: `foofoo.txt': File exists
Deleted hw3.2.A-output
17/05/30 05:56:48 INFO Configuration.deprecation: mapred.output.key.comparator
.class is deprecated. Instead, use mapreduce.job.output.key.comparator.class
packageJobJar: [] [/usr/jars/hadoop-streaming-2.6.0-cdh5.7.0.jar] /tmp/streamj
ob3826147749252105465.jar tmpDir=null
17/05/30 05:56:49 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0
.0:8032
17/05/30 05:56:50 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0
.0:8032
17/05/30 05:56:51 INFO mapred.FileInputFormat: Total input paths to process :
1
17/05/30 05:56:51 INFO mapreduce.JobSubmitter: number of splits:2
17/05/30 05:56:51 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_
1496033164706_0087
17/05/30 05:56:52 INFO impl.YarnClientImpl: Submitted application application_
1496033164706_0087
17/05/30 05:56:52 INFO mapreduce.Job: The url to track the job: http://quickst
art.cloudera:8088/proxy/application_1496033164706_0087/
17/05/30 05:56:52 INFO mapreduce.Job: Running job: job_1496033164706_0087
17/05/30 05:57:03 INFO mapreduce.Job: Job job_1496033164706_0087 running in ub
er mode : false
17/05/30 05:57:03 INFO mapreduce.Job:  map 0% reduce 0%
17/05/30 05:57:11 INFO mapreduce.Job:  map 50% reduce 0%
17/05/30 05:57:13 INFO mapreduce.Job:  map 100% reduce 0%
17/05/30 05:57:20 INFO mapreduce.Job:  map 100% reduce 100%
17/05/30 05:57:20 INFO mapreduce.Job: Job job_1496033164706_0087 completed suc
cessfully
17/05/30 05:57:20 INFO mapreduce.Job: Counters: 51
        File System Counters
                FILE: Number of bytes read=72
                FILE: Number of bytes written=353330
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=255
                HDFS: Number of bytes written=26
                HDFS: Number of read operations=9
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
        Job Counters
                Launched map tasks=2
                Launched reduce tasks=1
                Data-local map tasks=2
                Total time spent by all maps in occupied slots (ms)=12831
                Total time spent by all reduces in occupied slots (ms)=4650
                Total time spent by all map tasks (ms)=12831
                Total time spent by all reduce tasks (ms)=4650
                Total vcore-seconds taken by all map tasks=12831
                Total vcore-seconds taken by all reduce tasks=4650
                Total megabyte-seconds taken by all map tasks=13138944
                Total megabyte-seconds taken by all reduce tasks=4761600
        Map-Reduce Framework
                Map input records=1
                Map output records=7
                Map output bytes=52
                Map output materialized bytes=78
                Input split bytes=208
                Combine input records=0
                Combine output records=0
                Reduce input groups=4
                Reduce shuffle bytes=78
                Reduce input records=7
                Reduce output records=4
                Spilled Records=14
                Shuffled Maps =2
                Failed Shuffles=0
                Merged Map outputs=2
                GC time elapsed (ms)=123
                CPU time spent (ms)=2640
```

In [15]:
```
#HDFS Output
!hdfs dfs -ls hw3.2.A-output
!hdfs dfs -cat hw3.2.A-output/part-0000*
```

```
Found 2 items
-rw-r--r--   1 root supergroup          0 2017-05-30 05:57 hw3.2.A-output/_SUC
CESS
-rw-r--r--   1 root supergroup         26 2017-05-30 05:57 hw3.2.A-output/part
-00000
quux    2
labs    1
foo     3
bar     1
```

INSERT SCREENSHOT OF JOB TRACKER UI COUNTERS



	Name	Map	Reduce	Total
	Physical memory (bytes) snapshot	[illegible]	[illegible]	[illegible]
	Reduce input groups	0	4	4
	Reduce input records	0	7	7
	Reduce output records	0	4	4
	Reduce shuffle bytes	0	78	78
	Shuffled Maps	0	2	2
	Spilled Records	7	7	14
	Total committed heap usage (bytes)	498597888	141033472	639631360
	Virtual memory (bytes) snapshot	2729283584	1351589888	4080873472
Mapper Counters	Name	Map	Reduce	Total
	Calls	2	0	2
Reducer Counters	Name	Map	Reduce	Total
	Calls	0	1	1
	Name	Map	Reduce	Total
	BAD_ID	0	0	0

3.2.A EXPLANATION

With default setting , MapReduce selected to partition records into two maps so mapper is called 2 times. And then results are sent to single reducer.

3.2.B SOLUTION

In [16]:
```
%%writefile mapper3.2.B.py
#!/usr/bin/env python
# START STUDENT CODE HW32BMAPPER
from __future__ import division
import math
import os
import sys
import re

separator = ','
sys.stderr.write("reporter:counter:Mapper Counters,Calls,1\n")
WORD_RE = re.compile(r"[\w']+")
numReducers = int(os.environ.get('NUM_PARTITIONS', '4'))


def makeKey(word,n):
  divisor = 26/n
  return int(math.ceil((ord(word[0])-96)/divisor))

#loop through each records
for line in (sys.stdin):
#get 3rd column
        fields = line.split(separator)
        if 'Complaint ID' != fields[0] :

            # we have a real record, so do some mapping
            counter_name = None
            for word  in [s.lower() for s in WORD_RE.findall(fields[3])]:
                key = makeKey(word,numReducers)
                print '%s\t%s\t%s' % (key,word, 1)


# END STUDENT CODE HW32BMAPPER
```

```
Overwriting mapper3.2.B.py
```

In [17]:
```
%%writefile reducer3.2.B.py
#!/usr/bin/env python
# START STUDENT CODE HW32BREDUCER
import sys

cur_key = None
cur_count = 0
sys.stderr.write("reporter:counter:Reducer Counters,Calls,1\n")
for line in sys.stdin:
    partkey,key, value = line.split()
    if key == cur_key:
        cur_count += int(value)
    else:
        if cur_key:
            print '%s\t%s' % (cur_key, cur_count)
        cur_key = key
        cur_count = int(value)

print '%s\t%s' % (cur_key, cur_count)
# END STUDENT CODE HW32BREDUCER
```

```
Overwriting reducer3.2.B.py
```

In [18]:
```
!chmod a+x mapper3.2.B.py
!chmod a+x reducer3.2.B.py
```

In [19]:
```
#unit test
!head -10 Consumer_Complaints.csv|python mapper3.2.B.py|sort -k1,1|python reduc
er3.2.B.py
```

```
reporter:counter:Reducer Counters,Calls,1
reporter:counter:Mapper Counters,Calls,1
and     1
attempts        1
club    1
collect 1
cont'd  1
credit  1
debt    3
deposits        1
disclosure      2
false   1
health  1
incorrect       1
information     1
lease   2
loan    2
managing        2
not     1
of      2
on      1
or      3
owed    1
report  1
representation  1
statements      1
the     2
verification    2
withdrawals     1
```

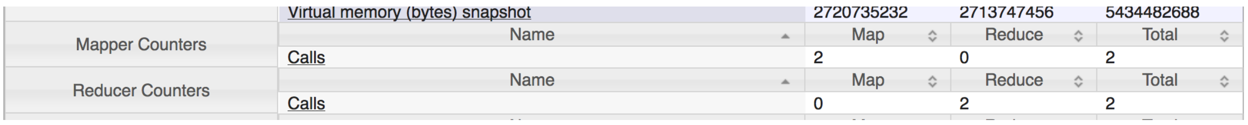
In [20]:
```
# Hadoop command
# START STUDENT CODE HW32BHADOOP
!hdfs dfs -rm -r hw3.2.B-output
!hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming.jar \
    -D mapreduce.job.maps=2 \
    -D mapreduce.job.reduces=2\
    -D stream.num.map.output.key.fields=2 \
    -D stream.map.output.field.separator="\t" \
    -D mapreduce.partition.keypartitioner.options=-k1,1 \
    -D mapred.output.key.comparator.class=org.apache.hadoop.mapred.lib.KeyField
BasedComparator \
    -D mapreduce.partition.keycomparator.options="-k1,1nr -k2,2" \
    -files mapper3.2.B.py,reducer3.2.B.py\
    -mapper mapper3.2.B.py \
    -reducer reducer3.2.B.py\
    -input  Consumer_Complaints.csv \
    -cmdenv NUM_PARTITIONS=2\
    -output  hw3.2.B-output

# END STUDENT CODE HW32BHADOOP
```

```
Deleted hw3.2.B-output
17/05/30 05:57:37 INFO Configuration.deprecation: mapred.output.key.comparator
.class is deprecated. Instead, use mapreduce.job.output.key.comparator.class
packageJobJar: [] [/usr/jars/hadoop-streaming-2.6.0-cdh5.7.0.jar] /tmp/streamj
ob4182344244284091905.jar tmpDir=null
17/05/30 05:57:38 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0
.0:8032
17/05/30 05:57:39 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0
.0:8032
17/05/30 05:57:41 INFO mapred.FileInputFormat: Total input paths to process :
1
17/05/30 05:57:41 INFO mapreduce.JobSubmitter: number of splits:2
17/05/30 05:57:42 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_
1496033164706_0088
17/05/30 05:57:42 INFO impl.YarnClientImpl: Submitted application application_
1496033164706_0088
17/05/30 05:57:42 INFO mapreduce.Job: The url to track the job: http://quickst
art.cloudera:8088/proxy/application_1496033164706_0088/
17/05/30 05:57:42 INFO mapreduce.Job: Running job: job_1496033164706_0088
17/05/30 05:57:50 INFO mapreduce.Job: Job job_1496033164706_0088 running in ub
er mode : false
17/05/30 05:57:50 INFO mapreduce.Job:  map 0% reduce 0%
17/05/30 05:58:06 INFO mapreduce.Job:  map 33% reduce 0%
17/05/30 05:58:07 INFO mapreduce.Job:  map 67% reduce 0%
17/05/30 05:58:08 INFO mapreduce.Job:  map 100% reduce 0%
17/05/30 05:58:20 INFO mapreduce.Job:  map 100% reduce 100%
17/05/30 05:58:21 INFO mapreduce.Job: Job job_1496033164706_0088 completed suc
cessfully
17/05/30 05:58:21 INFO mapreduce.Job: Counters: 51
        File System Counters
                FILE: Number of bytes read=14174923
                FILE: Number of bytes written=28821512
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=50910816
                HDFS: Number of bytes written=2091
                HDFS: Number of read operations=12
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=4
        Job Counters
                Launched map tasks=2
                Launched reduce tasks=2
                Data-local map tasks=2
                Total time spent by all maps in occupied slots (ms)=29807
                Total time spent by all reduces in occupied slots (ms)=17766
                Total time spent by all map tasks (ms)=29807
                Total time spent by all reduce tasks (ms)=17766
                Total vcore-seconds taken by all map tasks=29807
                Total vcore-seconds taken by all reduce tasks=17766
                Total megabyte-seconds taken by all map tasks=30522368
                Total megabyte-seconds taken by all reduce tasks=18192384
        Map-Reduce Framework
                Map input records=312913
                Map output records=980482
                Map output bytes=12213947
                Map output materialized bytes=14174935
                Input split bytes=234
                Combine input records=0
                Combine output records=0
                Reduce input groups=169
                Reduce shuffle bytes=14174935
                Reduce input records=980482
                Reduce output records=169
                Spilled Records=1960964
                Shuffled Maps =4
                Failed Shuffles=0
                Merged Map outputs=4
                GC time elapsed (ms)=432
                CPU time spent (ms)=15310
```

In [21]:
```
# 3.2.B OUTPUT/ANSWER
!hdfs dfs -ls hw3.2.B-output
!echo "________Output_________"
!hdfs dfs -cat hw3.2.B-output/part-0000*
```

```
Found 3 items
-rw-r--r--   1 root supergroup          0 2017-05-30 05:58 hw3.2.B-output/_SUC
CESS
-rw-r--r--   1 root supergroup        841 2017-05-30 05:58 hw3.2.B-output/part
-00000
-rw-r--r--   1 root supergroup       1250 2017-05-30 05:58 hw3.2.B-output/part
-00001
________Output_________
opening 16205
other   7886
out     1242
pay     3821
payment 92
plans   350
practices       1003
privacy 240
problems        9484
rate    3431
receiving       3226
report  34903
reporting       6559
rewards 1002
scam    566
score   4357
servicing       36767
sharing 2832
shopping        672
statements      2508
stop    131
taking  3747
transaction     1485
underwriting    2774
using   2422
when    4095
with    1944
withdrawals     10555
a       3503
account 20681
acct    163
an      2505
and     16448
applied 139
apr     3431
arbitration     168
available       274
bankruptcy      222
being   5663
billing 8158
by      5663
can't   1999
cash    240
caused  5663
changes 350
charges 131
checks  75
closing 2795
company's       4858
cont'd  11848
convenience     75
credit  55251
debt    19309
delay   243
delinquent      1061
deposits        10555
determination   1490
did     139
disclosure      5214
disputes        6938
expect  807
false   2508
```

INSERT SCREENSHOT OF JOB TRACKER UI COUNTERS



	Name	Map	Reduce	Total
	Virtual memory (bytes) snapshot	2720735232	2713747456	5434482688
Mapper Counters	Name	Map	Reduce	Total
	Calls	2	0	2
Reducer Counters	Name	Map	Reduce	Total
	Calls	0	2	2

3.2.C SOLUTION

In [22]:

```
%%writefile mapper3.2.C.py
#!/usr/bin/env python
# START STUDENT CODE HW32CMAPPER
from __future__ import division
import math
import os
import sys
import re

separator = ','
sys.stderr.write("reporter:counter:Mapper Counters,Calls,1\n")
WORD_RE = re.compile(r"[\w']+")
#numReducers = int(os.environ.get('NUM_PARTITIONS', '4'))

total = 0

def makeKey(word,n):
  divisor = 26/n
  return int(math.ceil((ord(word[0])-96)/divisor))

#loop through each records
for line in (sys.stdin):
#get 3rd column
        fields = line.split(separator)
        if 'Complaint ID' != fields[0] :

            # we have a real record, so do some mapping
            counter_name = None
            for word  in [s.lower() for s in WORD_RE.findall(fields[3])]:
                #key = makeKey(word,numReducers)
                print '%s\t%s' % (word, 1)
                total = total + 1
print '%s\t%s' % ("*total", total)



# END STUDENT CODE HW32CMAPPER
```

```
Overwriting mapper3.2.C.py
```

In [23]:
```
%%writefile combiner3.2.C.py
#!/usr/bin/env python
# START STUDENT CODE HW32CCOMBINER
import sys

cur_key = None
cur_count = 0
sys.stderr.write("reporter:counter:Combiner Counters,Calls,1\n")
for line in sys.stdin:
    key, value = line.split()
    if key == cur_key:
        cur_count += int(value)
    else:
        if cur_key:
            print '%s\t%s' % (cur_key, cur_count)
        cur_key = key
        cur_count = int(value)

print '%s\t%s' % (cur_key, cur_count)
# END STUDENT CODE HW32CCOMBINER
```

```
Overwriting combiner3.2.C.py
```

In [24]:
```
%%writefile reducer3.2.C.py
#!/usr/bin/env python
# START STUDENT CODE HW32CREDUCER
import sys

cur_key = None
cur_count = 0
sys.stderr.write("reporter:counter:Reducer Counters,Calls,1\n")
for line in sys.stdin:
    key, value = line.split()
    if key == cur_key:
        cur_count += int(value)
    else:
        if cur_key:
            print '%s\t%s' % (cur_key, cur_count)
        cur_key = key
        cur_count = int(value)

print '%s\t%s' % (cur_key, cur_count)
# END STUDENT CODE HW32CREDUCER
```

```
Overwriting reducer3.2.C.py
```

In [25]:
```
!chmod a+x mapper3.2.C.py
!chmod a+x reducer3.2.C.py
!chmod a+x combiner3.2.C.py
```

In [26]:
```
#unit Testing
!head -10 Consumer_Complaints.csv|python mapper3.2.C.py|sort -k1,1|python combi
ner3.2.C.py|python reducer3.2.C.py
```

```
reporter:counter:Reducer Counters,Calls,1
reporter:counter:Combiner Counters,Calls,1
reporter:counter:Mapper Counters,Calls,1
*total  38
and     1
attempts        1
club    1
collect 1
cont'd  1
credit  1
debt    3
deposits        1
disclosure      2
false   1
health  1
incorrect       1
information     1
lease   2
loan    2
managing        2
not     1
of      2
on      1
or      3
owed    1
report  1
representation  1
statements      1
the     2
verification    2
withdrawals     1
```

In [27]:

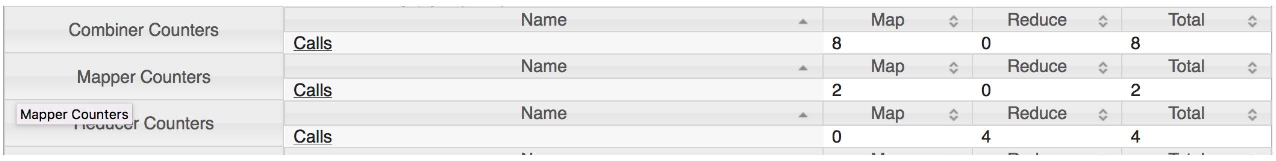
```
# Hadoop command
# START STUDENT CODE HW32CHADOOP
!hdfs dfs -rm -r hw3.2.C-output
!hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming.jar \
    -files mapper3.2.C.py,reducer3.2.C.py,combiner3.2.C.py\
    -mapper mapper3.2.C.py \
    -reducer reducer3.2.C.py\
    -combiner combiner3.2.C.py\
    -input  Consumer_Complaints.csv \
    -output  hw3.2.C-output \
    -numReduceTasks 4

# END STUDENT CODE HW32CHADOOP
```

```
Deleted hw3.2.C-output
packageJobJar: [] [/usr/jars/hadoop-streaming-2.6.0-cdh5.7.0.jar] /tmp/streamj
ob4293544407680665915.jar tmpDir=null
17/05/30 05:58:35 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0
.0:8032
17/05/30 05:58:35 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0
.0:8032
17/05/30 05:58:37 INFO mapred.FileInputFormat: Total input paths to process :
1
17/05/30 05:58:37 INFO mapreduce.JobSubmitter: number of splits:2
17/05/30 05:58:38 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_
1496033164706_0089
17/05/30 05:58:38 INFO impl.YarnClientImpl: Submitted application application_
1496033164706_0089
17/05/30 05:58:38 INFO mapreduce.Job: The url to track the job: http://quickst
art.cloudera:8088/proxy/application_1496033164706_0089/
17/05/30 05:58:38 INFO mapreduce.Job: Running job: job_1496033164706_0089
17/05/30 05:58:46 INFO mapreduce.Job: Job job_1496033164706_0089 running in ub
er mode : false
17/05/30 05:58:46 INFO mapreduce.Job:  map 0% reduce 0%
17/05/30 05:58:59 INFO mapreduce.Job:  map 50% reduce 0%
17/05/30 05:59:00 INFO mapreduce.Job:  map 100% reduce 0%
17/05/30 05:59:10 INFO mapreduce.Job:  map 100% reduce 25%
17/05/30 05:59:13 INFO mapreduce.Job:  map 100% reduce 50%
17/05/30 05:59:14 INFO mapreduce.Job:  map 100% reduce 75%
17/05/30 05:59:15 INFO mapreduce.Job:  map 100% reduce 100%
17/05/30 05:59:15 INFO mapreduce.Job: Job job_1496033164706_0089 completed suc
cessfully
17/05/30 05:59:15 INFO mapreduce.Job: Counters: 52
        File System Counters
                FILE: Number of bytes read=4488
                FILE: Number of bytes written=715820
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=50910816
                HDFS: Number of bytes written=2105
                HDFS: Number of read operations=18
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=8
        Job Counters
                Launched map tasks=2
                Launched reduce tasks=4
                Data-local map tasks=2
                Total time spent by all maps in occupied slots (ms)=20694
                Total time spent by all reduces in occupied slots (ms)=40757
                Total time spent by all map tasks (ms)=20694
                Total time spent by all reduce tasks (ms)=40757
                Total vcore-seconds taken by all map tasks=20694
                Total vcore-seconds taken by all reduce tasks=40757
                Total megabyte-seconds taken by all map tasks=21190656
                Total megabyte-seconds taken by all reduce tasks=41735168
        Map-Reduce Framework
                Map input records=312913
                Map output records=980484
                Map output bytes=9272529
                Map output materialized bytes=4512
                Input split bytes=234
                Combine input records=980484
                Combine output records=315
                Reduce input groups=170
                Reduce shuffle bytes=4512
                Reduce input records=315
                Reduce output records=170
                Spilled Records=630
                Shuffled Maps =8
                Failed Shuffles=0
                Merged Map outputs=8
                GC time elapsed (ms)=444
                CPU time spent (ms)=11680
```

In [28]:
```
# 3.2.C OUTPUT/ANSWER
!hdfs dfs -ls hw3.2.C-output
!echo "________Output_________"
!hdfs dfs -cat hw3.2.C-output/part-0000*
```

```
Found 5 items
-rw-r--r--   1 root supergroup          0 2017-05-30 05:59 hw3.2.C-output/_SUC
CESS
-rw-r--r--   1 root supergroup        452 2017-05-30 05:59 hw3.2.C-output/part
-00000
-rw-r--r--   1 root supergroup        600 2017-05-30 05:59 hw3.2.C-output/part
-00001
-rw-r--r--   1 root supergroup        517 2017-05-30 05:59 hw3.2.C-output/part
-00002
-rw-r--r--   1 root supergroup        536 2017-05-30 05:59 hw3.2.C-output/part
-00003
________Output_________
a       3503
account 20681
acct    163
applied 139
available       274
by      5663
can't   1999
cash    240
caused  5663
checks  75
closing 2795
company's       4858
cont'd  11848
debt    19309
delinquent      1061
disputes        6938
for     929
i       925
incorrect       29133
issuance        640
issue   1098
making  3226
of      10885
on      29069
or      22533
owed    11848
payoff  1155
processing      243
repay   1647
sale    139
service 1518
the     6248
to      8401
transfer        597
unable  8178
verification    5214
was     274
workout 350
wrong   169
your    3844
advance 240
amount  98
apply   118
atm     2422
bank    202
cancelling      2795
card    4405
collect 11848
communication   6920
costs   4350
credited        92
dealing 1944
decision        2774
didn't  925
disclosures     64
fee     3198
funds   5663
got     4357
```

INSERT SCREENSHOT OF JOB TRACKER UI COUNTERS



	Name	Map	Reduce	Total
Combiner Counters	Calls	8	0	8
Mapper Counters	Calls	2	0	2
Reducer Counters	Calls	0	4	4

In [29]:
```
%%writefile frequencies_mapper3.2.C.py
#!/usr/bin/env python
# START STUDENT CODE HW32CFREQMAPPER
from __future__ import division
import math
import os
import sys
import re

separator = ','
sys.stderr.write("reporter:counter:Mapper Counters,Calls,1\n")
WORD_RE = re.compile(r"[\w']+")

#loop through each records
for line in sys.stdin:
    print line.strip()



# END STUDENT CODE HW32CFREQMAPPER
```

```
Overwriting frequencies_mapper3.2.C.py
```

In [30]:
```
%%writefile frequencies_reducer3.2.C.py
#!/usr/bin/env python
# START STUDENT CODE HW32CFREQREDUCER
import sys

# Initialize variables
total = 0
cur_key = None
cur_count = 0
sys.stderr.write("reporter:counter:Reducer Counters,Calls,1\n")
for line in sys.stdin:

    fields = line.replace('\n','').split('\t')
    count = fields[1]
    word = fields[0]
    try:
        count = int(count)
    except ValueError:
        continue
    if word == '*total':
        total =  total + float(count)
    else:
        print '%s\t%s\t%2.3f' % (word, count, float(count)/total)
        #print "{0:20}\t{1:10}\t{2}\n".format(word, count, float(count)/total)
# END STUDENT CODE HW32CFREQREDUCER
```

```
Overwriting frequencies_reducer3.2.C.py
```

In [31]:
```
# Hadoop command
# START STUDENT CODE HW32CFREQHADOOP
!hdfs dfs -rm -r hw3.2.D-output
!hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming.jar \
    -D stream.num.map.output.key.fields=4 \
    -D mapred.output.key.comparator.class=org.apache.hadoop.mapred.lib.KeyField
BasedComparator \
    -D mapreduce.partition.keycomparator.options="-k2,2nr -k1,1" \
    -D mapreduce.job.reduces=1 \
    -files frequencies_mapper3.2.C.py,frequencies_reducer3.2.C.py \
    -mapper frequencies_mapper3.2.C.py \
    -reducer frequencies_reducer3.2.C.py \
    -input hw3.2.C-output \
    -output hw3.2.D-output \
    -partitioner org.apache.hadoop.mapred.lib.KeyFieldBasedPartitioner
# END STUDENT CODE HW32CFREQHADOOP
```

```
Deleted hw3.2.D-output
17/05/30 05:59:27 INFO Configuration.deprecation: mapred.output.key.comparator
.class is deprecated. Instead, use mapreduce.job.output.key.comparator.class
packageJobJar: [] [/usr/jars/hadoop-streaming-2.6.0-cdh5.7.0.jar] /tmp/streamj
ob8276092118841390770.jar tmpDir=null
17/05/30 05:59:28 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0
.0:8032
17/05/30 05:59:28 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0
.0:8032
17/05/30 05:59:29 INFO mapred.FileInputFormat: Total input paths to process :
4
17/05/30 05:59:29 INFO mapreduce.JobSubmitter: number of splits:4
17/05/30 05:59:29 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_
1496033164706_0090
17/05/30 05:59:30 INFO impl.YarnClientImpl: Submitted application application_
1496033164706_0090
17/05/30 05:59:30 INFO mapreduce.Job: The url to track the job: http://quickst
art.cloudera:8088/proxy/application_1496033164706_0090/
17/05/30 05:59:30 INFO mapreduce.Job: Running job: job_1496033164706_0090
17/05/30 05:59:38 INFO mapreduce.Job: Job job_1496033164706_0090 running in ub
er mode : false
17/05/30 05:59:38 INFO mapreduce.Job:  map 0% reduce 0%
17/05/30 05:59:47 INFO mapreduce.Job:  map 25% reduce 0%
17/05/30 05:59:50 INFO mapreduce.Job:  map 50% reduce 0%
17/05/30 05:59:52 INFO mapreduce.Job:  map 75% reduce 0%
17/05/30 05:59:53 INFO mapreduce.Job:  map 100% reduce 0%
17/05/30 05:59:55 INFO mapreduce.Job:  map 100% reduce 100%
17/05/30 05:59:56 INFO mapreduce.Job: Job job_1496033164706_0090 completed suc
cessfully
17/05/30 05:59:56 INFO mapreduce.Job: Counters: 51
        File System Counters
                FILE: Number of bytes read=2621
                FILE: Number of bytes written=594745
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=2581
                HDFS: Number of bytes written=3105
                HDFS: Number of read operations=15
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
        Job Counters
                Launched map tasks=4
                Launched reduce tasks=1
                Data-local map tasks=4
                Total time spent by all maps in occupied slots (ms)=37939
                Total time spent by all reduces in occupied slots (ms)=5807
                Total time spent by all map tasks (ms)=37939
                Total time spent by all reduce tasks (ms)=5807
                Total vcore-seconds taken by all map tasks=37939
                Total vcore-seconds taken by all reduce tasks=5807
                Total megabyte-seconds taken by all map tasks=38849536
                Total megabyte-seconds taken by all reduce tasks=5946368
        Map-Reduce Framework
                Map input records=170
                Map output records=170
                Map output bytes=2275
                Map output materialized bytes=2639
                Input split bytes=476
                Combine input records=0
                Combine output records=0
                Reduce input groups=170
                Reduce shuffle bytes=2639
                Reduce input records=170
                Reduce output records=169
                Spilled Records=340
                Shuffled Maps =4
                Failed Shuffles=0
                Merged Map outputs=4
                GC time elapsed (ms)=437
```

In [32]:
```
# 3.2.C OUTPUT/ANSWER
!hdfs dfs -ls hw3.2.D-output
!echo "________Output_________"

!hdfs dfs -cat hw3.2.D-output/part-0000*
```

```
Found 2 items
-rw-r--r--   1 root supergroup          0 2017-05-30 05:59 hw3.2.D-output/_SUC
CESS
-rw-r--r--   1 root supergroup       3105 2017-05-30 05:59 hw3.2.D-output/part
-00000
________Output_________
loan    119630  0.122
modification    70487   0.072
credit  55251   0.056
servicing       36767   0.037
report  34903   0.036
incorrect       29133   0.030
information     29069   0.030
on      29069   0.030
or      22533   0.023
account 20681   0.021
debt    19309   0.020
and     16448   0.017
opening 16205   0.017
club    12545   0.013
health  12545   0.013
not     12353   0.013
attempts        11848   0.012
collect 11848   0.012
cont'd  11848   0.012
owed    11848   0.012
of      10885   0.011
my      10731   0.011
deposits        10555   0.011
withdrawals     10555   0.011
problems        9484    0.010
application     8868    0.009
to      8401    0.009
unable  8178    0.008
billing 8158    0.008
other   7886    0.008
disputes        6938    0.007
communication   6920    0.007
tactics 6920    0.007
reporting       6559    0.007
lease   6337    0.006
the     6248    0.006
being   5663    0.006
by      5663    0.006
caused  5663    0.006
funds   5663    0.006
low     5663    0.006
process 5505    0.006
disclosure      5214    0.005
verification    5214    0.005
managing        5006    0.005
company's       4858    0.005
investigation   4858    0.005
identity        4729    0.005
card    4405    0.004
get     4357    0.004
score   4357    0.004
costs   4350    0.004
settlement      4350    0.004
improper        4309    0.004
interest        4238    0.004
protection      4139    0.004
when    4095    0.004
repaying        3844    0.004
your    3844    0.004
fraud   3842    0.004
are     3821    0.004
pay     3821    0.004
you     3821    0.004
taking  3747    0.004
```

What are the top 50 most frequent terms in your word count analysis?

In [33]:
```
!hdfs dfs -cat hw3.2.D-output/part-0000* | head -50
```

```
loan    119630  0.122
modification    70487   0.072
credit  55251   0.056
servicing       36767   0.037
report  34903   0.036
incorrect       29133   0.030
information     29069   0.030
on      29069   0.030
or      22533   0.023
account 20681   0.021
debt    19309   0.020
and     16448   0.017
opening 16205   0.017
club    12545   0.013
health  12545   0.013
not     12353   0.013
attempts        11848   0.012
collect 11848   0.012
cont'd  11848   0.012
owed    11848   0.012
of      10885   0.011
my      10731   0.011
deposits        10555   0.011
withdrawals     10555   0.011
problems        9484    0.010
application     8868    0.009
to      8401    0.009
unable  8178    0.008
billing 8158    0.008
other   7886    0.008
disputes        6938    0.007
communication   6920    0.007
tactics 6920    0.007
reporting       6559    0.007
lease   6337    0.006
the     6248    0.006
being   5663    0.006
by      5663    0.006
caused  5663    0.006
funds   5663    0.006
low     5663    0.006
process 5505    0.006
disclosure      5214    0.005
verification    5214    0.005
managing        5006    0.005
company's       4858    0.005
investigation   4858    0.005
identity        4729    0.005
card    4405    0.004
get     4357    0.004
```

Present the top 50 terms and their frequency and their relative frequency. If there are ties please sort the tokens in alphanumeric/string order.

In [34]:
```
!hdfs dfs -cat hw3.2.D-output/part-0000* | sort -k2,2nr |head -50
```

```
loan    119630  0.122
modification    70487   0.072
credit  55251   0.056
servicing       36767   0.037
report  34903   0.036
incorrect       29133   0.030
information     29069   0.030
on      29069   0.030
or      22533   0.023
account 20681   0.021
debt    19309   0.020
and     16448   0.017
opening 16205   0.017
club    12545   0.013
health  12545   0.013
not     12353   0.013
attempts        11848   0.012
collect 11848   0.012
cont'd  11848   0.012
owed    11848   0.012
of      10885   0.011
my      10731   0.011
deposits        10555   0.011
withdrawals     10555   0.011
problems        9484    0.010
application     8868    0.009
to      8401    0.009
unable  8178    0.008
billing 8158    0.008
other   7886    0.008
disputes        6938    0.007
communication   6920    0.007
tactics 6920    0.007
reporting       6559    0.007
lease   6337    0.006
the     6248    0.006
being   5663    0.006
by      5663    0.006
caused  5663    0.006
funds   5663    0.006
low     5663    0.006
process 5505    0.006
disclosure      5214    0.005
verification    5214    0.005
managing        5006    0.005
company's       4858    0.005
investigation   4858    0.005
identity        4729    0.005
card    4405    0.004
get     4357    0.004
```

Present bottom 10 tokens (least frequent items).

In [35]:
```
!hdfs dfs -tail  hw3.2.D-output/part-00000 > hw3.2.D.txt
!tail -10 hw3.2.D.txt
```

```
apply	118	0.000
amount	98	0.000
credited	92	0.000
payment	92	0.000
checks	75	0.000
convenience	75	0.000
amt	71	0.000
day	71	0.000
disclosures	64	0.000
missing	64	0.000
```

3.2.1

Using **2 reducers**: What are the top **50 most frequent terms** in your word count analysis?

Present the top 50 terms and their frequency and their relative frequency. Present the top 50 terms and their frequency and their relative frequency. If there are ties please sort the tokens in alphanumeric/string order. Present bottom 10 tokens (least frequent items). Please **use a combiner.**

START STUDENT CODE HW321 (INSERT CELLS BELOW AS NEEDED)

In [36]:
```
%%writefile frequencies_mapper3.2.1.py
#!/usr/bin/env python
from __future__ import division
import math
import os
import sys
import re


count = 0


separator = ','
#create partition key
def makeKeyn(word):
  if ord(word[0]) in range(ord('a'), ord('m')):
     return 'A'
  else:
     return 'B'
#regex for word extraction
WORD_RE = re.compile(r"[\w']+")
for line in sys.stdin:
    fields = line.split(separator)
    if 'Complaint ID' != fields[0]:
        for word  in [s.lower() for s in WORD_RE.findall(fields[3])]:
    # prepend a key based on the number of reducers
            key = makeKeyn(word)
            count = count + 1
            print key,"\t",word,"\t",1
print 'A',"\t","*total","\t",count #to get total in all combiners
print 'B',"\t","*total","\t",count
```

```
Overwriting frequencies_mapper3.2.1.py
```

In [37]:
```
%%writefile frequencies_reducer3.2.1.py
#!/usr/bin/env python
import sys
import os
# Initialize variables
total = 0
cur_key = None
cur_count = 0
sys.stderr.write("reporter:counter:Reducer Counters,Calls,1\n")
dictcounts = {}
#totalrecs = int(os.environ.get('TOTAL_RECS', '980482'))
for line in sys.stdin:

    fields = line.replace('\n','').split('\t')
    count = fields[2]
    word = fields[1]
    try:
        count = int(count)
    except ValueError:
        continue
    if word == '*total': #not required in multireducers
        total =  total + int(count)
    else:
        x = dictcounts.get(word,None)
        if x != None:
            dictcounts[word]+=count
        else:
            dictcounts[word]=count
for key in dictcounts:

        print '%s\t%d\t%2.3f' %(key,dictcounts[key] ,float(dictcounts[key])/tot
al)

        #print '%s,%s\t%s\t%2.3f' % (fields[0],word, count, float(count)/total)
```

```
Overwriting frequencies_reducer3.2.1.py
```

In [38]:
```
%%writefile frequencies_combine3.2.1.py
#!/usr/bin/env python
import sys
import os
# Initialize variables
total = 0
cur_key = ("key1","key2")
cur_count = 0
sys.stderr.write("reporter:counter:combiner Counters,Calls,1\n")

for line in sys.stdin:

        partkey,key1, value = line.split()
        partkey = partkey
        key = (partkey,key1)
        if key1 == cur_key[1]:
            cur_count += int(value)
        else:
            if cur_key!= ("key1","key2"):
                print '%s\t%s\t%d' % (cur_key[0],cur_key[1], cur_count)
            cur_key = key
            cur_count = int(value)

print '%s\t%s\t%d' % (cur_key[0],cur_key[1], cur_count)
```

```
Overwriting frequencies_combine3.2.1.py
```

In [39]:
```
!chmod a+x frequencies_reducer3.2.1.py
!chmod a+x frequencies_combine3.2.1.py
!chmod a+x frequencies_mapper3.2.1.py
```

END STUDENT CODE HW321

In [40]:
```
#start 3.2.1

!hdfs dfs -rm -r hw3.2.1-output
!hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming.jar \
    -D stream.num.map.output.key.fields=2 \
    -D mapreduce.partition.keypartitioner.options=-k1,1 \
    -D mapreduce.job.output.key.comparator.class=org.apache.hadoop.mapred.lib.K
eyFieldBasedComparator \
    -D mapreduce.partition.keycomparator.options="-k1,1 -k2,2" \
    -D mapreduce.job.reduces=2 \
    -files frequencies_mapper3.2.1.py,frequencies_reducer3.2.1.py,frequencies_c
ombine3.2.1.py \
    -mapper frequencies_mapper3.2.1.py\
    -reducer frequencies_reducer3.2.1.py\
    -combiner frequencies_combine3.2.1.py\
    -input Consumer_Complaints.csv \
    -output hw3.2.1-output \
    -partitioner org.apache.hadoop.mapred.lib.KeyFieldBasedPartitioner
```

```
Deleted hw3.2.1-output
packageJobJar: [] [/usr/jars/hadoop-streaming-2.6.0-cdh5.7.0.jar] /tmp/streamj
ob5705506576480295317.jar tmpDir=null
17/05/30 06:00:21 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0
.0:8032
17/05/30 06:00:21 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0
.0:8032
17/05/30 06:00:22 INFO mapred.FileInputFormat: Total input paths to process :
1
17/05/30 06:00:22 INFO mapreduce.JobSubmitter: number of splits:2
17/05/30 06:00:23 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_
1496033164706_0091
17/05/30 06:00:23 INFO impl.YarnClientImpl: Submitted application application_
1496033164706_0091
17/05/30 06:00:23 INFO mapreduce.Job: The url to track the job: http://quickst
art.cloudera:8088/proxy/application_1496033164706_0091/
17/05/30 06:00:23 INFO mapreduce.Job: Running job: job_1496033164706_0091
17/05/30 06:00:31 INFO mapreduce.Job: Job job_1496033164706_0091 running in ub
er mode : false
17/05/30 06:00:31 INFO mapreduce.Job:  map 0% reduce 0%
17/05/30 06:00:46 INFO mapreduce.Job:  map 28% reduce 0%
17/05/30 06:00:47 INFO mapreduce.Job:  map 56% reduce 0%
17/05/30 06:00:49 INFO mapreduce.Job:  map 62% reduce 0%
17/05/30 06:00:50 INFO mapreduce.Job:  map 100% reduce 0%
17/05/30 06:01:02 INFO mapreduce.Job:  map 100% reduce 50%
17/05/30 06:01:03 INFO mapreduce.Job:  map 100% reduce 100%
17/05/30 06:01:04 INFO mapreduce.Job: Job job_1496033164706_0091 completed suc
cessfully
17/05/30 06:01:04 INFO mapreduce.Job: Counters: 51
        File System Counters
                FILE: Number of bytes read=5142
                FILE: Number of bytes written=485598
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=50910816
                HDFS: Number of bytes written=3105
                HDFS: Number of read operations=12
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=4
        Job Counters
                Launched map tasks=2
                Launched reduce tasks=2
                Data-local map tasks=2
                Total time spent by all maps in occupied slots (ms)=32337
                Total time spent by all reduces in occupied slots (ms)=19546
                Total time spent by all map tasks (ms)=32337
                Total time spent by all reduce tasks (ms)=19546
                Total vcore-seconds taken by all map tasks=32337
                Total vcore-seconds taken by all reduce tasks=19546
                Total megabyte-seconds taken by all map tasks=33113088
                Total megabyte-seconds taken by all reduce tasks=20015104
        Map-Reduce Framework
                Map input records=312913
                Map output records=980486
                Map output bytes=13194501
                Map output materialized bytes=5154
                Input split bytes=234
                Combine input records=980486
                Combine output records=317
                Reduce input groups=2
                Reduce shuffle bytes=5154
                Reduce input records=317
                Reduce output records=169
                Spilled Records=634
                Shuffled Maps =4
                Failed Shuffles=0
                Merged Map outputs=4
                GC time elapsed (ms)=344
                CPU time spent (ms)=20030
```

In [41]:
```
!hdfs dfs -ls hw3.2.1-output
!hdfs dfs -cat hw3.2.1-output/part-0000* | sort -k2,2nr > hw3.2.1.txt
```

```
Found 3 items
-rw-r--r--   1 root supergroup          0 2017-05-30 06:01 hw3.2.1-output/_SUC
CESS
-rw-r--r--   1 root supergroup       1392 2017-05-30 06:01 hw3.2.1-output/part
-00000
-rw-r--r--   1 root supergroup       1713 2017-05-30 06:01 hw3.2.1-output/part
-00001
```

In [42]:
```
!head -50 hw3.2.1.txt
```

```
loan    119630  0.122
modification    70487   0.072
credit  55251   0.056
servicing       36767   0.037
report  34903   0.036
incorrect       29133   0.030
information     29069   0.030
on      29069   0.030
or      22533   0.023
account 20681   0.021
debt    19309   0.020
and     16448   0.017
opening 16205   0.017
club    12545   0.013
health  12545   0.013
not     12353   0.013
attempts        11848   0.012
collect 11848   0.012
cont'd  11848   0.012
owed    11848   0.012
of      10885   0.011
my      10731   0.011
deposits        10555   0.011
withdrawals     10555   0.011
problems        9484    0.010
application     8868    0.009
to      8401    0.009
unable  8178    0.008
billing 8158    0.008
other   7886    0.008
disputes        6938    0.007
communication   6920    0.007
tactics 6920    0.007
reporting       6559    0.007
lease   6337    0.006
the     6248    0.006
being   5663    0.006
by      5663    0.006
caused  5663    0.006
funds   5663    0.006
low     5663    0.006
process 5505    0.006
disclosure      5214    0.005
verification    5214    0.005
managing        5006    0.005
company's       4858    0.005
investigation   4858    0.005
identity        4729    0.005
card    4405    0.004
get     4357    0.004
```

HW3.3. Shopping Cart Analysis

Product Recommendations: The action or practice of selling additional products or services to existing customers is called cross-selling. Giving product recommendation is one of the examples of cross-selling that are frequently used by online retailers. One simple method to give product recommendations is to recommend products that are frequently browsed together by the customers.

For this homework use the online browsing behavior dataset located at:

```
https://www.dropbox.com/s/zlfyiwa70poqg74/ProductPurchaseData.txt?dl=0
```

Each line in this dataset represents a browsing session of a customer. On each line, each string of 8 characters represents the id of an item browsed during that session. The items are separated by spaces.

Here are the first few lines of the ProductPurchaseData FRO11987 ELE17451 ELE89019 SNA90258 GRO99222 GRO99222 GRO12298 FRO12685 ELE91550 SNA11465 ELE26917 ELE52966 FRO90334 SNA30755 ELE17451 FRO84225 SNA80192 ELE17451 GRO73461 DAI22896 SNA99873 FRO86643 ELE17451 ELE37798 FRO86643 GRO56989 ELE23393 SNA11465 ELE17451 SNA69641 FRO86643 FRO78087 SNA11465 GRO39357 ELE28573 ELE11375 DAI54444

Do some exploratory data analysis of this dataset guided by the following questions:.

How many unique items are available from this supplier?

Using a single reducer: Report your findings such as number of unique products; largest basket; report the top 50 most frequently purchased items, their frequency, and their relative frequency (break ties by sorting the products alphabetical order) etc. using Hadoop Map-Reduce.

START STUDENT CODE HW33 (INSERT CELLS BELOW AS NEEDED)

In [43]:
```
#get data first
!wget "https://www.dropbox.com/s/zlfyiwa70poqg74/ProductPurchaseData.txt?dl=0"
!mv ProductPurchaseData.txt?dl=0 ProductPurchaseData.txt
```

```
--2017-05-30 06:01:12--  https://www.dropbox.com/s/zlfyiwa70poqg74/ProductPurc
haseData.txt?dl=0
Resolving www.dropbox.com... 162.125.4.1
Connecting to www.dropbox.com|162.125.4.1|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://dl.dropboxusercontent.com/content_link/YnjBvBDFNoHAIgv7R3QKC
7QvPDOfghJjID5D4DL2UM0PKueav0KljFOdzVxrdSlZ/file [following]
--2017-05-30 06:01:13--  https://dl.dropboxusercontent.com/content_link/YnjBvB
DFNoHAIgv7R3QKC7QvPDOfghJjID5D4DL2UM0PKueav0KljFOdzVxrdSlZ/file
Resolving dl.dropboxusercontent.com... 162.125.4.6
Connecting to dl.dropboxusercontent.com|162.125.4.6|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3458517 (3.3M) [text/plain]
Saving to: `ProductPurchaseData.txt?dl=0'

100%[======================================>] 3,458,517   2.46M/s   in 1.3s

2017-05-30 06:01:15 (2.46 MB/s) - `ProductPurchaseData.txt?dl=0' saved [345851
7/3458517]
```

In [44]:
```
!hdfs dfs -copyFromLocal ProductPurchaseData.txt
!hdfs dfs -rm -r hw3.3-output
```

```
copyFromLocal: `ProductPurchaseData.txt': File exists
Deleted hw3.3-output
```

In [45]:
```
%%writefile mapper_33.py
#!/usr/bin/python
## mapper.py


import sys

# Increment mapper counter
sys.stderr.write("reporter:counter:Mapper Counters,Calls,1\n")

# Initialsys.stderr.write(ize variables
total = 0
basket_size = 0
largest_basket_size = 0

for line in sys.stdin:
    total = 0
    basket_size = 0
    # Split our line into products
    for product in line.replace('\n','').split():
        print '%s\t%s' % (product, 1)
        #print generateLongCountToken(product)
        basket_size += 1
        total += 1



    print '%s\t%s' % ('*largest_basket', total)
    #basket_size = 0
```

```
Overwriting mapper_33.py
```

In [46]:
```
!chmod a+x mapper_33.py
```

In [47]:
```
%%writefile reducer33.py
#!/usr/bin/env python
# START STUDENT CODE HW32AREDUCER
import sys
from collections import OrderedDict

cur_key = None
cur_count = 0
dictcounts = {}
largest = []
sys.stderr.write("reporter:counter:Reducer Counters,Calls,1\n")
for line in sys.stdin:
    key, value = line.split()

    if key != '*largest_basket':
        if key == cur_key:
            cur_count += int(value)
        else:
            if cur_key:
                dictcounts[cur_key] = cur_count
            #print '%s\t%s' % (cur_key, cur_count)
            cur_key = key
            cur_count = int(value)
    else:
        if key == '*largest_basket':
            largest.append(int(value))
print "***********************Output*****************************"
print "Maximum length of Bucket %d"%(max(largest))

print "Total No of Unique products %d"% len(dictcounts.keys())
totals = sum(dictcounts.values())
dictcounts =OrderedDict(sorted(dictcounts.items(), key=lambda t: t[1], reverse=
True))
count  = 0
print "*****Top 50 Products*************"
for key in dictcounts:
    if count <= 49:
        print '%s\t%d\t%2.3f' %(key,dictcounts[key] ,float(dictcounts[key])/tot
als )
    count += 1
```

```
Overwriting reducer33.py
```

In [48]:
```
!hdfs dfs -cat ProductPurchaseData.txt |python mapper_33.py|sort -k1,1 |python
reducer33.py
```

```
reporter:counter:Mapper Counters,Calls,1
reporter:counter:Reducer Counters,Calls,1
************************Output*****************************
Maximum length of Bucket 37
Total No of Unique products 12591
*****Top 50 Products*************
DAI62779        6667    0.018
FRO40251        3881    0.010
ELE17451        3875    0.010
GRO73461        3602    0.009
SNA80324        3044    0.008
ELE32164        2851    0.007
DAI75645        2736    0.007
SNA45677        2455    0.006
FRO31317        2330    0.006
DAI85309        2293    0.006
ELE26917        2292    0.006
FRO80039        2233    0.006
GRO21487        2115    0.006
SNA99873        2083    0.005
GRO59710        2004    0.005
GRO71621        1920    0.005
FRO85978        1918    0.005
GRO30386        1840    0.005
ELE74009        1816    0.005
GRO56726        1784    0.005
DAI63921        1773    0.005
GRO46854        1756    0.005
ELE66600        1713    0.004
DAI83733        1712    0.004
FRO32293        1702    0.004
ELE66810        1697    0.004
SNA55762        1646    0.004
DAI22177        1627    0.004
FRO78087        1531    0.004
ELE99737        1516    0.004
ELE34057        1489    0.004
GRO94758        1489    0.004
FRO35904        1436    0.004
FRO53271        1420    0.004
SNA93860        1407    0.004
SNA90094        1390    0.004
GRO38814        1352    0.004
ELE56788        1345    0.004
GRO61133        1321    0.003
ELE74482        1316    0.003
DAI88807        1316    0.003
ELE59935        1311    0.003
SNA96271        1295    0.003
DAI43223        1290    0.003
ELE91337        1289    0.003
GRO15017        1275    0.003
DAI31081        1261    0.003
GRO81087        1220    0.003
DAI22896        1219    0.003
GRO85051        1214    0.003
```

In [49]:
```
!hdfs dfs -rm -r hw3.3-output

!hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming.jar \
    -D mapred.reduce.tasks=1 \
    -D mapred.output.key.comparator.class=org.apache.hadoop.mapred.lib.KeyField
BasedComparator \
    -D stream.num.map.output.key.fields=2 \
    -D stream.map.output.field.separator="\t" \
    -D mapreduce.partition.keycomparator.options="-k1,1" \
    -files mapper_33.py,reducer33.py\
    -mapper mapper_33.py \
    -reducer reducer33.py\
    -input ProductPurchaseData.txt \
    -output  hw3.3-output
```

```
rm: `hw3.3-output': No such file or directory
17/05/30 06:01:36 INFO Configuration.deprecation: mapred.reduce.tasks is depre
cated. Instead, use mapreduce.job.reduces
17/05/30 06:01:36 INFO Configuration.deprecation: mapred.output.key.comparator
.class is deprecated. Instead, use mapreduce.job.output.key.comparator.class
packageJobJar: [] [/usr/jars/hadoop-streaming-2.6.0-cdh5.7.0.jar] /tmp/streamj
ob2739617999185734490.jar tmpDir=null
17/05/30 06:01:37 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0
.0:8032
17/05/30 06:01:37 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0
.0:8032
17/05/30 06:01:38 INFO mapred.FileInputFormat: Total input paths to process :
1
17/05/30 06:01:39 INFO mapreduce.JobSubmitter: number of splits:2
17/05/30 06:01:39 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_
1496033164706_0092
17/05/30 06:01:39 INFO impl.YarnClientImpl: Submitted application application_
1496033164706_0092
17/05/30 06:01:39 INFO mapreduce.Job: The url to track the job: http://quickst
art.cloudera:8088/proxy/application_1496033164706_0092/
17/05/30 06:01:39 INFO mapreduce.Job: Running job: job_1496033164706_0092
17/05/30 06:01:50 INFO mapreduce.Job: Job job_1496033164706_0092 running in ub
er mode : false
17/05/30 06:01:50 INFO mapreduce.Job:  map 0% reduce 0%
17/05/30 06:02:07 INFO mapreduce.Job:  map 50% reduce 0%
17/05/30 06:02:08 INFO mapreduce.Job:  map 100% reduce 0%
17/05/30 06:02:17 INFO mapreduce.Job:  map 100% reduce 100%
17/05/30 06:02:18 INFO mapreduce.Job: Job job_1496033164706_0092 completed suc
cessfully
17/05/30 06:02:18 INFO mapreduce.Job: Counters: 51
        File System Counters
                FILE: Number of bytes read=6005167
                FILE: Number of bytes written=12363340
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=3462847
                HDFS: Number of bytes written=1160
                HDFS: Number of read operations=9
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
        Job Counters
                Launched map tasks=2
                Launched reduce tasks=1
                Data-local map tasks=2
                Total time spent by all maps in occupied slots (ms)=28944
                Total time spent by all reduces in occupied slots (ms)=7936
                Total time spent by all map tasks (ms)=28944
                Total time spent by all reduce tasks (ms)=7936
                Total vcore-seconds taken by all map tasks=28944
                Total vcore-seconds taken by all reduce tasks=7936
                Total megabyte-seconds taken by all map tasks=29638656
                Total megabyte-seconds taken by all reduce tasks=8126464
        Map-Reduce Framework
                Map input records=31101
                Map output records=411925
                Map output bytes=5181311
                Map output materialized bytes=6005173
                Input split bytes=234
                Combine input records=0
                Combine output records=0
                Reduce input groups=41578
                Reduce shuffle bytes=6005173
                Reduce input records=411925
                Reduce output records=54
                Spilled Records=823850
                Shuffled Maps =2
                Failed Shuffles=0
                Merged Map outputs=2
```

In [50]:
```
!hdfs dfs -ls hw3.3-output
!hdfs dfs -cat hw3.3-output/part-0000*
```

```
Found 2 items
-rw-r--r--   1 root supergroup          0 2017-05-30 06:02 hw3.3-output/_SUCCE
SS
-rw-r--r--   1 root supergroup       1160 2017-05-30 06:02 hw3.3-output/part-0
0000
***********************Output****************************
Maximum length of Bucket 37
Total No of Unique products 12591
*****Top 50 Products*************
DAI62779        6667    0.018
FRO40251        3881    0.010
ELE17451        3875    0.010
GRO73461        3602    0.009
SNA80324        3044    0.008
ELE32164        2851    0.007
DAI75645        2736    0.007
SNA45677        2455    0.006
FRO31317        2330    0.006
DAI85309        2293    0.006
ELE26917        2292    0.006
FRO80039        2233    0.006
GRO21487        2115    0.006
SNA99873        2083    0.005
GRO59710        2004    0.005
GRO71621        1920    0.005
FRO85978        1918    0.005
GRO30386        1840    0.005
ELE74009        1816    0.005
GRO56726        1784    0.005
DAI63921        1773    0.005
GRO46854        1756    0.005
ELE66600        1713    0.004
DAI83733        1712    0.004
FRO32293        1702    0.004
ELE66810        1697    0.004
SNA55762        1646    0.004
DAI22177        1627    0.004
FRO78087        1531    0.004
ELE99737        1516    0.004
ELE34057        1489    0.004
GRO94758        1489    0.004
FRO35904        1436    0.004
FRO53271        1420    0.004
SNA93860        1407    0.004
SNA90094        1390    0.004
GRO38814        1352    0.004
ELE56788        1345    0.004
GRO61133        1321    0.003
ELE74482        1316    0.003
DAI88807        1316    0.003
ELE59935        1311    0.003
SNA96271        1295    0.003
DAI43223        1290    0.003
ELE91337        1289    0.003
GRO15017        1275    0.003
DAI31081        1261    0.003
GRO81087        1220    0.003
DAI22896        1219    0.003
GRO85051        1214    0.003
```

END STUDENT CODE HW33

HW3.3.1 OPTIONAL

Using 2 reducers: Report your findings such as number of unique products; largest basket; report the top 50 most frequently purchased items, their frequency, and their relative frequency (break ties by sorting the products alphabetical order) etc. using Hadoop Map-Reduce.

START STUDENT CODE HW331 (INSERT CELLS BELOW AS NEEDED)

END STUDENT CODE HW331

HW3.4. (Computationally prohibitive but then again Hadoop can handle this) Pairs

Suppose we want to recommend new products to the customer based on the products they have already browsed on the online website. Write a map-reduce program to find products which are frequently browsed together. Fix the support count (cooccurence count) to s = 100 (i.e. product pairs need to occur together at least 100 times to be considered frequent) and find pairs of items (sometimes referred to itemsets of size 2 in association rule mining) that have a support count of 100 or more.

List the top 50 product pairs with corresponding support count (aka frequency), and relative frequency or support (number of records where they coccur, the number of records where they coccur/the number of baskets in the dataset) in decreasing order of support for frequent (100>count) itemsets of size 2.

Use the Pairs pattern (lecture 3) to extract these frequent itemsets of size 2. Free free to use combiners if they bring value. Instrument your code with counters for count the number of times your mapper, combiner and reducers are called.

Please output records of the following form for the top 50 pairs (itemsets of size 2):

```
item1, item2, support count, support
```

Fix the ordering of the pairs lexicographically (left to right), and break ties in support (between pairs, if any exist) by taking the first ones in lexicographically increasing order.

Report the compute time for the Pairs job. Describe the computational setup used (E.g., single computer; dual core; linux, number of mappers, number of reducers) Instrument your mapper, combiner, and reducer to count how many times each is called using Counters and report these counts.

START STUDENT CODE HW34 (INSERT CELLS BELOW AS NEEDED)

In [51]:
```
%%writefile mapper_34.py
#!/usr/bin/python
## mapper.py

import sys
from itertools import  combinations

# Increment mapper counter
sys.stderr.write("reporter:counter:Mapper Counters,Calls,1\n")

# Initialize variables
total = 0

# Our input comes from STDIN (standard input)
for line in sys.stdin:
    # Split our line into products
    products = line.replace('\n','').split()

    # Get all combinations of products:
    #  - Use a set to remove duplicate products
    #  - Combinations finds tuples of length 2 with no repeats
    for pair in combinations(sorted(set(products)), 2):
                print '%s\t%s\t%s' % (pair[0], pair[1], 1)

    total += 1
# Print total words
print '%s\t%s\t%s' % ('*total', '*total', total)
```

```
Overwriting mapper_34.py
```

In [52]:
```
!chmod a+x mapper_34.py
```

In [53]:
```
#unit test
!hdfs dfs -cat ProductPurchaseData.txt |head -1|python mapper_34.py|sort -k1,1
```

```
reporter:counter:Mapper Counters,Calls,1
cat: Unable to write to output stream.
*total  *total  1
ELE17451        ELE89019        1
ELE17451        FRO11987        1
ELE17451        GRO99222        1
ELE17451        SNA90258        1
ELE89019        FRO11987        1
ELE89019        GRO99222        1
ELE89019        SNA90258        1
FRO11987        GRO99222        1
FRO11987        SNA90258        1
GRO99222        SNA90258        1
```

In [54]:
```
%%writefile combiner34.py
#!/usr/bin/env python
# START STUDENT CODE HW32CCOMBINER
import sys

cur_key = None
cur_count = 0
mydict = {}
sys.stderr.write("reporter:counter:Combiner Counters,Calls,1\n")
for line in sys.stdin:
    key1,key2, value = line.split()
    key = (key1,key2)
    if key == cur_key:
        cur_count += int(value)
    else:
        if cur_key and cur_count >=100:
            print '%s\t%s\t%s' % (cur_key[0],cur_key[1], cur_count)
        cur_key = key
        cur_count = int(value)

print '%s\t%s\t%s'% (cur_key[0],cur_key[1], cur_count)
```

```
Overwriting combiner34.py
```

In [55]:
```
!chmod a+x combiner34.py
```

In [56]:
```
%%writefile reducer34.py
#!/usr/bin/env python
# START STUDENT CODE HW32AREDUCER
import sys
from collections import OrderedDict

cur_key = None
cur_count = 0
dictcounts = {}
largest = []
sys.stderr.write("reporter:counter:Reducer Counters,Calls,1\n")
for line in sys.stdin:
    key1,key2, value = line.split()
    key = (key1,key2)
    if key1 != '*total':
        if key == cur_key:
            cur_count += int(value)
        else:
            if cur_key:
                dictcounts[cur_key] = cur_count
            #print '%s\t%s' % (cur_key, cur_count)
            cur_key = key
            cur_count = int(value)
    else:
        if key1 == '*total':
            largest.append(int(value))

totals = sum(largest)
dictcounts =OrderedDict(sorted(dictcounts.items(), key=lambda t: t[1], reverse=
True))
count  = 0
print "*****Top 50 Products*************"
for key in dictcounts:
    if count <= 49:
        print '%s\t%s\t%d\t%2.3f' %(key[0],key[1],dictcounts[key] ,float(dictco
unts[key])/totals )
    count += 1
```

```
Overwriting reducer34.py
```

In [57]:
```
!chmod a+x reducer34.py
```

In [58]:
```
#hadoop call
!hdfs dfs -rm -r hw3.4-output
!time hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming.jar \
    -D stream.num.map.output.key.fields=4 \
    -D mapreduce.job.reduces=1 \
    -D mapreduce.job.output.key.comparator.class=org.apache.hadoop.mapred.lib.K
eyFieldBasedComparator \
    -D mapreduce.partition.keycomparator.options="-k1,1 -k2,2" \
    -files mapper_34.py,combiner34.py,reducer34.py\
    -mapper mapper_34.py\
    -reducer reducer34.py\
    -input  ProductPurchaseData.txt \
    -output  hw3.4-output \
Deleted hw3.4-output
packageJobJar: [] [/usr/jars/hadoop-streaming-2.6.0-cdh5.7.0.jar] /tmp/streamj
ob7922580423188743914.jar tmpDir=null
17/05/30 06:02:42 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0
.0:8032
17/05/30 06:02:42 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0
.0:8032
17/05/30 06:02:44 INFO mapred.FileInputFormat: Total input paths to process :
1
17/05/30 06:02:44 INFO mapreduce.JobSubmitter: number of splits:2
17/05/30 06:02:44 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_
1496033164706_0093
17/05/30 06:02:45 INFO impl.YarnClientImpl: Submitted application application_
1496033164706_0093
17/05/30 06:02:45 INFO mapreduce.Job: The url to track the job: http://quickst
art.cloudera:8088/proxy/application_1496033164706_0093/
17/05/30 06:02:45 INFO mapreduce.Job: Running job: job_1496033164706_0093
17/05/30 06:03:02 INFO mapreduce.Job: Job job_1496033164706_0093 running in ub
er mode : false
17/05/30 06:03:02 INFO mapreduce.Job:  map 0% reduce 0%
17/05/30 06:03:20 INFO mapreduce.Job:  map 20% reduce 0%
17/05/30 06:03:22 INFO mapreduce.Job:  map 38% reduce 0%
17/05/30 06:03:23 INFO mapreduce.Job:  map 51% reduce 0%
17/05/30 06:03:25 INFO mapreduce.Job:  map 59% reduce 0%
17/05/30 06:03:28 INFO mapreduce.Job:  map 67% reduce 0%
17/05/30 06:03:34 INFO mapreduce.Job:  map 83% reduce 0%
17/05/30 06:03:36 INFO mapreduce.Job:  map 100% reduce 0%
17/05/30 06:03:48 INFO mapreduce.Job:  map 100% reduce 76%
17/05/30 06:03:51 INFO mapreduce.Job:  map 100% reduce 88%
17/05/30 06:03:54 INFO mapreduce.Job:  map 100% reduce 100%
17/05/30 06:04:12 INFO mapreduce.Job: Job job_1496033164706_0093 completed suc
cessfully
17/05/30 06:04:12 INFO mapreduce.Job: Counters: 51
        File System Counters
                FILE: Number of bytes read=58282374
                FILE: Number of bytes written=116918585
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=3462847
                HDFS: Number of bytes written=1442
                HDFS: Number of read operations=9
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
        Job Counters
                Launched map tasks=2
                Launched reduce tasks=1
                Data-local map tasks=2
                Total time spent by all maps in occupied slots (ms)=61869
                Total time spent by all reduces in occupied slots (ms)=34671
                Total time spent by all map tasks (ms)=61869
                Total time spent by all reduce tasks (ms)=34671
                Total vcore-seconds taken by all map tasks=61869
                Total vcore-seconds taken by all reduce tasks=34671
                Total megabyte-seconds taken by all map tasks=63353856
                Total megabyte-seconds taken by all reduce tasks=35503104
        Map-Reduce Framework
                Map input records=31101
                Map output records=2534016
                Map output bytes=53214336
                Map output materialized bytes=58282380
                Input split bytes=234
                Combine input records=0
                Combine output records=0
                Reduce input groups=877097
                Reduce shuffle bytes=58282380
                Reduce input records=2534016
                Reduce output records=51
                Spilled Records=5068032
                Shuffled Maps =2
```

In [59]:
```
#print output
!hdfs dfs -ls hw3.4-output
!hdfs dfs -cat hw3.4-output/part-0000*
```

```
Found 2 items
-rw-r--r--   1 root supergroup          0 2017-05-30 06:04 hw3.4-output/_SUCCE
SS
-rw-r--r--   1 root supergroup       1442 2017-05-30 06:04 hw3.4-output/part-0
0000
*****Top 50 Products*************
DAI62779        ELE17451        1592    0.051
FRO40251        SNA80324        1412    0.045
DAI75645        FRO40251        1254    0.040
FRO40251        GRO85051        1213    0.039
DAI62779        GRO73461        1139    0.037
DAI75645        SNA80324        1130    0.036
DAI62779        FRO40251        1070    0.034
DAI62779        SNA80324        923     0.030
DAI62779        DAI85309        918     0.030
ELE32164        GRO59710        911     0.029
FRO40251        GRO73461        882     0.028
DAI62779        DAI75645        882     0.028
DAI62779        ELE92920        877     0.028
FRO40251        FRO92469        835     0.027
DAI62779        ELE32164        832     0.027
DAI75645        GRO73461        712     0.023
DAI43223        ELE32164        711     0.023
DAI62779        GRO30386        709     0.023
ELE17451        FRO40251        697     0.022
DAI85309        ELE99737        659     0.021
DAI62779        ELE26917        650     0.021
GRO21487        GRO73461        631     0.020
DAI62779        SNA45677        604     0.019
ELE17451        SNA80324        597     0.019
DAI62779        GRO71621        595     0.019
DAI62779        SNA55762        593     0.019
DAI62779        DAI83733        586     0.019
ELE17451        GRO73461        580     0.019
GRO73461        SNA80324        562     0.018
DAI62779        GRO59710        561     0.018
DAI62779        FRO80039        550     0.018
DAI75645        ELE17451        547     0.018
DAI62779        SNA93860        537     0.017
DAI55148        DAI62779        526     0.017
DAI43223        GRO59710        512     0.016
ELE17451        ELE32164        511     0.016
DAI62779        SNA18336        506     0.016
ELE32164        GRO73461        486     0.016
DAI85309        ELE17451        482     0.015
DAI62779        FRO78087        482     0.015
DAI62779        GRO94758        479     0.015
DAI62779        GRO21487        471     0.015
GRO85051        SNA80324        471     0.015
ELE17451        GRO30386        468     0.015
FRO85978        SNA95666        463     0.015
DAI62779        FRO19221        462     0.015
DAI62779        GRO46854        461     0.015
DAI43223        DAI62779        459     0.015
ELE92920        SNA18336        455     0.015
DAI88079        FRO40251        446     0.014
```

END STUDENT CODE HW34

HW3.5: Stripes

Repeat 3.4 using the stripes design pattern for finding cooccuring pairs.

Report the compute times for stripes job versus the Pairs job. Describe the computational setup used (E.g., single computer; dual core; linux, number of mappers, number of reducers)

Instrument your mapper, combiner, and reducer to count how many times each is called using Counters and report these counts. Discuss the differences in these counts between the Pairs and Stripes jobs

START STUDENT CODE HW35 (INSERT CELLS BELOW AS NEEDED)

In [60]:
```
%%writefile mapper_35.py
#!/usr/bin/python
## mapper.py

import sys
from itertools import  combinations
#import collections

# Increment mapper counter
sys.stderr.write("reporter:counter:Mapper Counters,Calls,1\n")

# Initialize variables
total = 0

# Our input comes from STDIN (standard input)
for line in sys.stdin:
    # Split our line into products
    products = line.replace('\n','').split()

    # Get all combinations of products:
    #  - Use a set to remove duplicate products
    #  - Combinations finds tuples of length 2 with no repeats

    for i, term in enumerate(products):
                # Create a new stripe for each term
                stripe = {}

                for j, token in enumerate(products):
                    # Don't count the term's co-occurrence with itself
                    if i != j:
                        x = stripe.get(token,None)
                        if x == None:
                            stripe[token] = 1
                        else:
                            stripe[token] += 1

                # Emit the term and the stripe
                print '%s\t%s' % (term, stripe)
# Increment total number of baskets
    total += 1
stripe = {}
stripe['*total'] = total
print '%s\t%s' % ('*total', stripe)
```

```
Overwriting mapper_35.py
```

In [61]:
```
%%writefile reducer35.py
#!/usr/bin/env python
# START STUDENT CODE HW32AREDUCER
import sys
from collections import OrderedDict
#from collections import collections
#import collections
prev_key = None
cur_count = 0
prev_stripe = {}
largest = []
dictcounts = {}
sys.stderr.write("reporter:counter:Reducer Counters,Calls,1\n")

for line in sys.stdin:

    fields = line.replace('\n','').split('\t')
    key = fields[0]

    stripe = eval(fields[1])


    if prev_key == key:
        # We need to move through the dictionary and update counts
        for item in stripe:
            if item in prev_stripe:
                prev_stripe[item] += stripe[item]
            else:
                prev_stripe[item] = stripe[item]

    else:
        if len(prev_stripe) > 0:
            # We are at a new pair, need to print previous pair sum
            #print '%s\t%s' % (prev_key, prev_stripe)
            for word in prev_stripe:
                dictcounts[(prev_key,word)] = prev_stripe[word]
        prev_stripe = stripe
        prev_key = key

# Output the last line
if prev_stripe == stripe:
    for word in prev_stripe:
        dictcounts[(prev_key,word)] = prev_stripe[word]
totals = dictcounts[('*total','*total')]
dictcounts =OrderedDict(sorted(dictcounts.items(), key=lambda t: t[1], reverse=
True))
count  = 0
print "*****Top 50 Products*************"
for key in dictcounts:
    if count <= 100 and count%2 == 0:
        print '%s\t%s\t%d\t%2.3f' %(key[0],key[1],dictcounts[key] ,float(dictco
unts[key])/totals )
    count += 1
```

```
Overwriting reducer35.py
```

In [62]:
```
!chmod a+x mapper_35.py
!chmod a+x reducer35.py
```

In [63]:
```
!hdfs dfs -cat ProductPurchaseData.txt |head -10|python mapper_35.py|sort -k1,1
|python reducer35.py
```

```
reporter:counter:Mapper Counters,Calls,1
reporter:counter:Reducer Counters,Calls,1
cat: Unable to write to output stream.
*****Top 50 Products*************
*total  *total  10      1.000
ELE17451        SNA80192        5       0.500
SNA69641        ELE17451        3       0.300
ELE17451        SNA69641        3       0.300
GRO73461        DAI22896        3       0.300
GRO73461        ELE17451        3       0.300
ELE17451        SNA11465        3       0.300
ELE17451        GRO73461        3       0.300
SNA90258        ELE17451        3       0.300
DAI22896        GRO73461        3       0.300
ELE17451        FRO86643        3       0.300
SNA11465        FRO86643        2       0.200
FRO78087        ELE11375        2       0.200
SNA69641        FRO78087        2       0.200
ELE17451        SNA85662        2       0.200
SNA80192        FRO18919        2       0.200
GRO73461        SNA99873        2       0.200
FRO81176        GRO94758        2       0.200
DAI91535        GRO94758        2       0.200
GRO56989        ELE37798        2       0.200
ELE17451        ELE37798        2       0.200
GRO99222        SNA80192        2       0.200
ELE28573        SNA69641        2       0.200
DAI22896        SNA80192        2       0.200
SNA69641        ELE28573        2       0.200
DAI22177        ELE17451        2       0.200
DAI91535        FRO81176        2       0.200
GRO94758        ELE17451        2       0.200
ELE23393        ELE17451        2       0.200
SNA85662        ELE17451        2       0.200
ELE17451        FRO81176        2       0.200
SNA80192        GRO94758        2       0.200
DAI91535        SNA80192        2       0.200
FRO86643        SNA11465        2       0.200
DAI22177        SNA85662        2       0.200
ELE28573        ELE11375        2       0.200
SNA80192        DAI91535        2       0.200
ELE28573        ELE17451        2       0.200
GRO94758        SNA80192        2       0.200
ELE11375        SNA69641        2       0.200
SNA85662        SNA80192        2       0.200
GRO94758        FRO81176        2       0.200
SNA80192        DAI22896        2       0.200
DAI22896        SNA99873        2       0.200
ELE66810        SNA80192        1       0.100
GRO73461        DAI91535        1       0.100
ELE59935        FRO18919        1       0.100
GRO39357        DAI54444        1       0.100
FRO84225        FRO90334        1       0.100
ELE17451        ELE89019        1       0.100
GRO75578        ELE17451        1       0.100
```

In [64]:
```
#hadoop call
!hdfs dfs -rm -r hw3.5-output
!time hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming.jar \
    -D stream.num.map.output.key.fields=4 \
    -D mapreduce.job.reduces=1 \
    -D mapreduce.job.output.key.comparator.class=org.apache.hadoop.mapred.lib.K
eyFieldBasedComparator \
    -D mapreduce.partition.keycomparator.options="-k1,1 -k2,2" \
    -files mapper_35.py,combiner34.py,reducer35.py\
    -mapper mapper_35.py\
    -reducer reducer35.py\
    -input  ProductPurchaseData.txt \
    -output  hw3.5-output \
    -cmdenv PATH=/opt/anaconda/bin:$PATH
```

```
Deleted hw3.5-output
packageJobJar: [] [/usr/jars/hadoop-streaming-2.6.0-cdh5.7.0.jar] /tmp/streamj
ob6862122422670722550.jar tmpDir=null
17/05/30 06:04:40 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0
.0:8032
17/05/30 06:04:40 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0
.0:8032
17/05/30 06:04:43 INFO mapred.FileInputFormat: Total input paths to process :
1
17/05/30 06:04:43 INFO mapreduce.JobSubmitter: number of splits:2
17/05/30 06:04:44 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_
1496033164706_0094
17/05/30 06:04:44 INFO impl.YarnClientImpl: Submitted application application_
1496033164706_0094
17/05/30 06:04:44 INFO mapreduce.Job: The url to track the job: http://quickst
art.cloudera:8088/proxy/application_1496033164706_0094/
17/05/30 06:04:44 INFO mapreduce.Job: Running job: job_1496033164706_0094
17/05/30 06:04:54 INFO mapreduce.Job: Job job_1496033164706_0094 running in ub
er mode : false
17/05/30 06:04:54 INFO mapreduce.Job:  map 0% reduce 0%
17/05/30 06:05:13 INFO mapreduce.Job:  map 58% reduce 0%
17/05/30 06:05:16 INFO mapreduce.Job:  map 67% reduce 0%
17/05/30 06:05:22 INFO mapreduce.Job:  map 83% reduce 0%
17/05/30 06:05:23 INFO mapreduce.Job:  map 100% reduce 0%
17/05/30 06:05:36 INFO mapreduce.Job:  map 100% reduce 71%
17/05/30 06:05:40 INFO mapreduce.Job:  map 100% reduce 75%
17/05/30 06:05:43 INFO mapreduce.Job:  map 100% reduce 78%
17/05/30 06:05:46 INFO mapreduce.Job:  map 100% reduce 82%
17/05/30 06:05:49 INFO mapreduce.Job:  map 100% reduce 86%
17/05/30 06:05:52 INFO mapreduce.Job:  map 100% reduce 89%
17/05/30 06:05:55 INFO mapreduce.Job:  map 100% reduce 92%
17/05/30 06:05:58 INFO mapreduce.Job:  map 100% reduce 96%
17/05/30 06:06:01 INFO mapreduce.Job:  map 100% reduce 99%
17/05/30 06:06:04 INFO mapreduce.Job:  map 100% reduce 100%
17/05/30 06:06:19 INFO mapreduce.Job: Job job_1496033164706_0094 completed suc
cessfully
17/05/30 06:06:19 INFO mapreduce.Job: Counters: 51
        File System Counters
                FILE: Number of bytes read=81828248
                FILE: Number of bytes written=164010753
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=3462847
                HDFS: Number of bytes written=1468
                HDFS: Number of read operations=9
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
        Job Counters
                Launched map tasks=2
                Launched reduce tasks=1
                Data-local map tasks=2
                Total time spent by all maps in occupied slots (ms)=51492
                Total time spent by all reduces in occupied slots (ms)=52635
                Total time spent by all map tasks (ms)=51492
                Total time spent by all reduce tasks (ms)=52635
                Total vcore-seconds taken by all map tasks=51492
                Total vcore-seconds taken by all reduce tasks=52635
                Total megabyte-seconds taken by all map tasks=52727808
                Total megabyte-seconds taken by all reduce tasks=53898240
        Map-Reduce Framework
                Map input records=31101
                Map output records=380826
                Map output bytes=80638408
                Map output materialized bytes=81828254
                Input split bytes=234
                Combine input records=0
                Combine output records=0
                Reduce input groups=377574
                Reduce shuffle bytes=81828254
```

In [65]:
```
#print output
!hdfs dfs -ls hw3.5-output
!hdfs dfs -cat hw3.5-output/part-0000*
```

```
Found 2 items
-rw-r--r--   1 root supergroup          0 2017-05-30 06:06 hw3.5-output/_SUCCE
SS
-rw-r--r--   1 root supergroup       1468 2017-05-30 06:06 hw3.5-output/part-0
0000
*****Top 50 Products*************
*total  *total  31101   1.000
ELE17451        DAI62779        1592    0.051
FRO40251        SNA80324        1412    0.045
FRO40251        DAI75645        1254    0.040
FRO40251        GRO85051        1213    0.039
GRO73461        DAI62779        1139    0.037
DAI75645        SNA80324        1130    0.036
FRO40251        DAI62779        1070    0.034
SNA80324        DAI62779        923     0.030
DAI85309        DAI62779        918     0.030
GRO59710        ELE32164        911     0.029
FRO40251        GRO73461        882     0.028
DAI62779        DAI75645        882     0.028
ELE92920        DAI62779        877     0.028
FRO92469        FRO40251        835     0.027
DAI62779        ELE32164        832     0.027
DAI75645        GRO73461        712     0.023
DAI43223        ELE32164        711     0.023
GRO30386        DAI62779        709     0.023
FRO40251        ELE17451        697     0.022
ELE99737        DAI85309        659     0.021
ELE26917        DAI62779        650     0.021
GRO73461        GRO21487        631     0.020
DAI62779        SNA45677        604     0.019
SNA80324        ELE17451        597     0.019
GRO71621        DAI62779        595     0.019
DAI62779        SNA55762        593     0.019
DAI62779        DAI83733        586     0.019
GRO73461        ELE17451        580     0.019
SNA80324        GRO73461        562     0.018
GRO59710        DAI62779        561     0.018
FRO80039        DAI62779        550     0.018
ELE17451        DAI75645        547     0.018
DAI62779        SNA93860        537     0.017
DAI55148        DAI62779        526     0.017
GRO59710        DAI43223        512     0.016
ELE17451        ELE32164        511     0.016
SNA18336        DAI62779        506     0.016
GRO73461        ELE32164        486     0.016
DAI62779        FRO78087        482     0.015
FRO78087        DAI62779        482     0.015
GRO94758        DAI62779        479     0.015
DAI62779        GRO21487        471     0.015
GRO21487        DAI62779        471     0.015
ELE17451        GRO30386        468     0.015
FRO85978        SNA95666        463     0.015
DAI62779        FRO19221        462     0.015
DAI62779        GRO46854        461     0.015
DAI62779        DAI43223        459     0.015
SNA18336        ELE92920        455     0.015
FRO40251        DAI88079        446     0.014
```

In [66]:
```
!cat /proc/cpuinfo | grep processor | wc -l
```

```
2
```

In [67]:
```
!cat /proc/meminfo
```

```
MemTotal:        5068464 kB
MemFree:         1719108 kB
MemAvailable:    1947436 kB
Buffers:           77720 kB
Cached:           477608 kB
SwapCached:          808 kB
Active:          2839068 kB
Inactive:         369844 kB
Active(anon):    2597472 kB
Inactive(anon):   204776 kB
Active(file):     241596 kB
Inactive(file):   165068 kB
Unevictable:           0 kB
Mlocked:               0 kB
SwapTotal:       1048572 kB
SwapFree:        1040120 kB
Dirty:               764 kB
Writeback:             0 kB
AnonPages:       2652740 kB
Mapped:           124824 kB
Shmem:            148664 kB
Slab:              87636 kB
SReclaimable:      61824 kB
SUnreclaim:        25812 kB
KernelStack:       15392 kB
PageTables:        12984 kB
NFS_Unstable:          0 kB
Bounce:                0 kB
WritebackTmp:          0 kB
CommitLimit:     3582804 kB
Committed_AS:    6061972 kB
VmallocTotal:   34359738367 kB
VmallocUsed:           0 kB
VmallocChunk:          0 kB
AnonHugePages:         0 kB
ShmemHugePages:        0 kB
ShmemPmdMapped:        0 kB
HugePages_Total:       0
HugePages_Free:        0
HugePages_Rsvd:        0
HugePages_Surp:        0
Hugepagesize:       2048 kB
DirectMap4k:       24576 kB
DirectMap2M:     4169728 kB
DirectMap1G:     3145728 kB
```

```
Answer
```

System Setup

Single Computer , docker Container, 2 Cores and 5GB RAM.

How many times is each mapper and reducer called?

Mapper 2 Reducer 1

Total time

With Pairs

```
real    1m6.364s
user    0m4.720s
sys    0m1.240s

    Launched map tasks=2
    Launched reduce tasks=1
    Data-local map tasks=2
```

Total time spent by all maps in occupied slots (ms)=39445

```
    Total time spent by all reduces in occupied slots (ms)=27979
    Total time spent by all map tasks (ms)=39445
    Total time spent by all reduce tasks (ms)=27979
    Total vcore-seconds taken by all map tasks=39445
    Total vcore-seconds taken by all reduce tasks=27979
    Total megabyte-seconds taken by all map tasks=40391680
    Total megabyte-seconds taken by all reduce tasks=28650496
```

With Stripes

```
real    1m11.730s
user    0m5.310s
sys    0m1.310s

Launched map tasks=2
    Launched reduce tasks=1
    Data-local map tasks=2
```

Total time spent by all maps in occupied slots (ms)=26607

```
    Total time spent by all reduces in occupied slots (ms)=39085
    Total time spent by all map tasks (ms)=26607
    Total time spent by all reduce tasks (ms)=39085
    Total vcore-seconds taken by all map tasks=26607
    Total vcore-seconds taken by all reduce tasks=39085
    Total megabyte-seconds taken by all map tasks=27245568
    Total megabyte-seconds taken by all reduce tasks=40023040
```

As expected Mappers took much less time with Stripes compared to pairs as expected but in reducers Pairs took bit longer. This could be due to all unpacking we have to do with stripes to calculate final count.

END STUDENT CODE HW35

OPTIONAL

QUESTIONS BELOW THIS LINE ARE OPTIONAL

HW3.6 Computing Relative Frequencies on 100K WikiPedia pages (93Meg)

Dataset description For this assignment you will explore a set of 100,000 Wikipedia documents:

https://www.dropbox.com/s/n5lfbnztclo93ej/wikitext_100k.txt?dl=0 (https://www.dropbox.com/s/n5lfbnztclo93ej/wikitext_100k.txt?dl=0) s3://cs9223/wikitext_100k.txt, or https://s3.amazonaws.com/cs9223/wikitext_100k.txt (https://s3.amazonaws.com/cs9223/wikitext_100k.txt) Each line in this file consists of the plain text extracted from a Wikipedia document.

Task Compute the relative frequencies of each word that occurs in the documents in wikitext_100k.txt and output the top 100 word pairs sorted by decreasing order of relative frequency.

Recall that the relative frequency (RF) of word B given word A is defined as follows:

f(B|A) = Count(A, B) / Count (A) = Count(A, B) / sum_B'(Count (A, B')

where count(A,B) is the number of times A and B co-occur within a window of two words (co-occurrence window size of two) in a document and count(A) the number of times A occurs with anything else. Intuitively, given a document collection, the relative frequency captures the proportion of time the word B appears in the same document as A. (See Section 3.3, in Data-Intensive Text Processing with MapReduce).

In the async lecture you learned different approaches to do this, and in this assignment, you will implement them:

a. Write a mapreduce program which uses the Stripes approach and writes its output in a file named rfstripes.txt

b. Write a mapreduce program which uses the Pairs approach and writes its output in a file named rfpairs.txt

c. Compare the performance of the two approaches and output the relative performance to a file named rfcomp.txt. Compute the relative performance as follows: (running time for Pairs/ running time for Stripes). Also include an analysis comparing the communication costs for the two approaches. Instrument your mapper and reduces for counters where necessary to aid with your analysis.

NOTE: please limit your analysis to the top 100 word pairs sorted by decreasing order of relative frequency for each word (tokens with all alphabetical letters).

Please include markdown cell named rf.txt that describes the following:

the input/output format in each Hadoop task, i.e., the keys for the mappers and reducers the Hadoop cluster settings you used, i.e., number of mappers and reducers the running time for each approach: pairs and stripes

You can write your program using Python or MrJob (with Hadoop streaming) and you should run it on AWS. It is a good idea to develop and test your program on a local machine before deploying on AWS. Remember your notebook, needs to have all the commands you used to run each Mapreduce job (i.e., pairs and stripes) -- include the Hadoop streaming commands you used to run your jobs.

In addition the All the following files should be compressed in one ZIP file and submitted. The ZIP file should contain:

A. The result files: rfstripes.txt, rfpairs.txt, rfcomp.txt

Prior to working with Hadoop, the corpus should first be preprocessed as follows: perform tokenization (whitespace and all non-alphabetic characters) and stopword removal using standard tools from the Lucene search engine. All tokens should then be replaced with unique integers for a more efficient encoding.

== Preliminary information for the remaing HW problems===

Much of this homework beyond this point will focus on the Apriori algorithm for frequent itemset mining and the additional step for extracting association rules from these frequent itemsets. Please acquaint yourself with the background information (below) before approaching the remaining assignments.

=== Apriori background information ===

Some background material for the Apriori algorithm is located at:

HW3.7 Apriori Algorithm

What is the Apriori algorithm? Describe an example use in your domain of expertise and what kind of . Define confidence and lift.

NOTE: For the remaining homework use the online browsing behavior dataset located at (same dataset as used above):

```
https://www.dropbox.com/s/zlfyiwa70poqg74/ProductPurchaseData.txt?dl=0
```

Each line in this dataset represents a browsing session of a customer. On each line, each string of 8 characters represents the id of an item browsed during that session. The items are separated by spaces.

Here are the first few lines of the ProductPurchaseData FRO11987 ELE17451 ELE89019 SNA90258 GRO99222 GRO99222 GRO12298 FRO12685 ELE91550 SNA11465 ELE26917 ELE52966 FRO90334 SNA30755 ELE17451 FRO84225 SNA80192 ELE17451 GRO73461 DAI22896 SNA99873 FRO86643 ELE17451 ELE37798 FRO86643 GRO56989 ELE23393 SNA11465 ELE17451 SNA69641 FRO86643 FRO78087 SNA11465 GRO39357 ELE28573 ELE11375 DAI54444

HW3.8. Shopping Cart Analysis

Product Recommendations: The action or practice of selling additional products or services to existing customers is called cross-selling. Giving product recommendation is one of the examples of cross-selling that are frequently used by online retailers. One simple method to give product recommendations is to recommend products that are frequently browsed together by the customers.

Suppose we want to recommend new products to the customer based on the products they have already browsed on the online website. Write a program using the A-priori algorithm to find products which are frequently browsed together. Fix the support to s = 100 (i.e. product sets need to occur together at least 100 times to be considered frequent) and find itemsets of size 2 and 3.

Then extract association rules from these frequent items.

A rule is of the form:

(item1, item5) ⇒ item2.

List the top 10 discovered rules in descreasing order of confidence in the following format

(item1, item5) ⇒ item2, supportCount ,support, confidence

HW3.8.1

Benchmark your results using the pyFIM implementation of the Apriori algorithm (Apriori - Association Rule Induction / Frequent Item Set Mining implemented by Christian Borgelt). You can download pyFIM from here:

http://www.borgelt.net/pyfim.html (http://www.borgelt.net/pyfim.html)

Comment on the results from both implementations (your Hadoop MapReduce of apriori versus pyFIM) in terms of results and execution times.

END OF HOMEWORK