L03a: Filter, Join & SQL

ANLY 502: Massive Data Fundamentals

Simson Garfinkel & Ghaleb Abdulla

January 25, 2016

Snowzilla Recap!

Outline for today's class

Administrivia

PS01 — Redux

Student Presentations

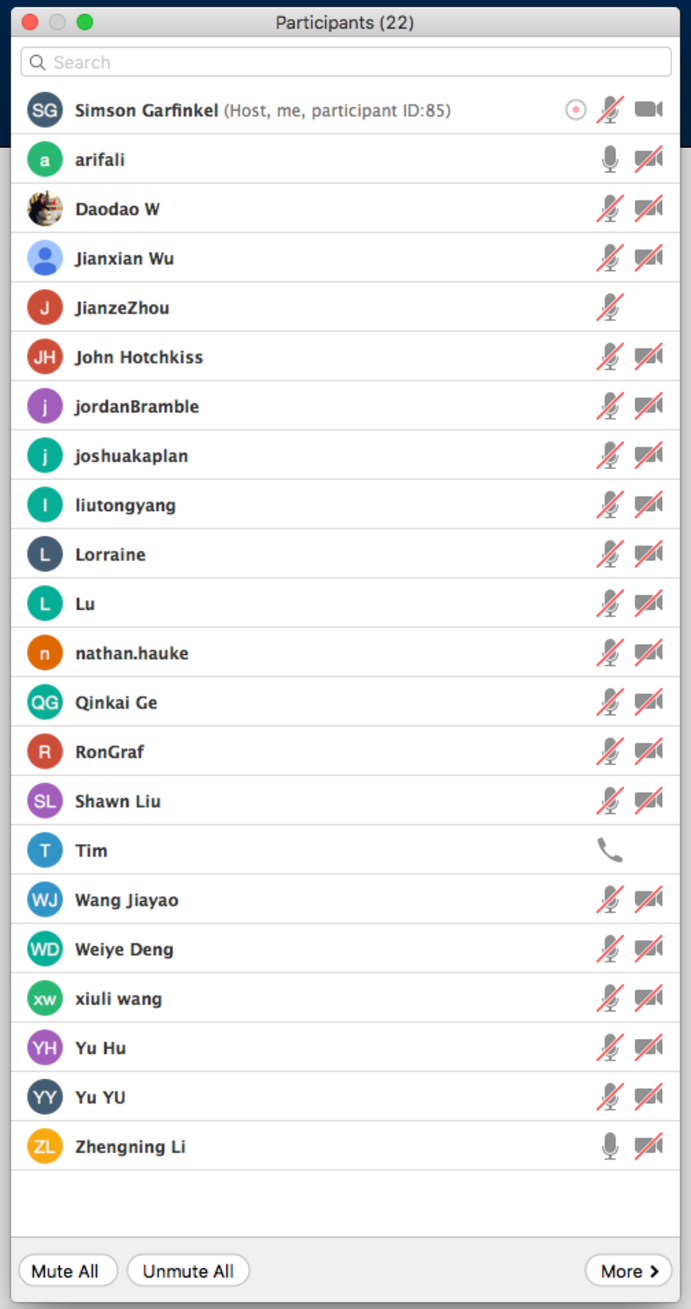
Filter, Top 10, and Join with Map Reduce

Filter, Order & Join with SQL

PS02 — Preview

Review of RAM errors paper

Thanks for attending L02



Administrivia 2 — GitHub Repo

Git hub was down!

- https://github.com/simsong/ANLY502

Backup repository at bitbucket

- https://bitbucket.org/simson_garfinkel/anly502

Two services avoids single point of failure.

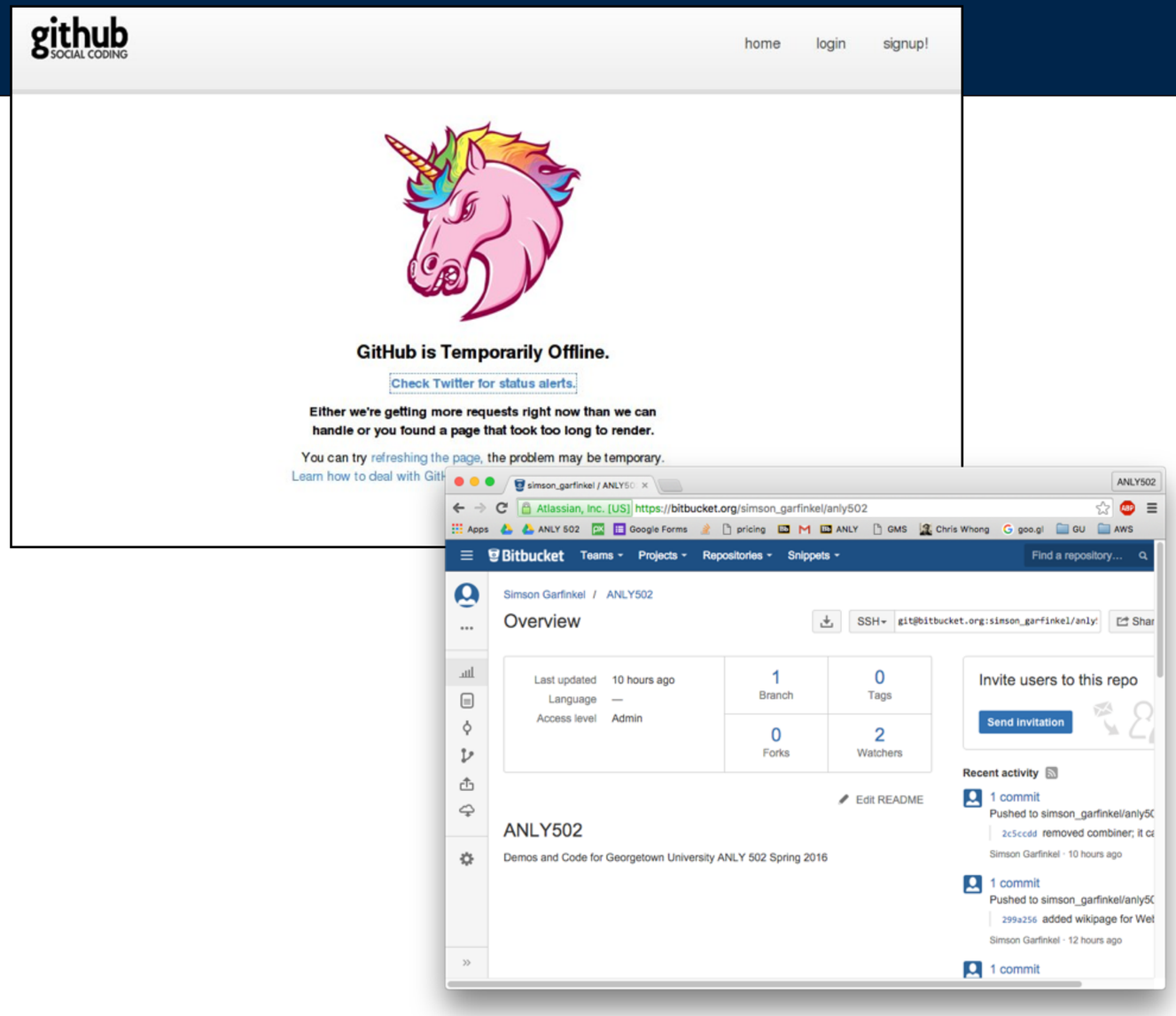
- I need to push to each.

ANLY502 git repository will have:

- Sample code for problem sets
- Sample code from class slides

Git commands:

- git clone
- git stash; git pull; git stash apply



PS01

PS01

You needed to be able to:

- Run VMWare
- Run Cloudera Quickstart VM

Part 1 — WordCount in Java

- Put files in HDFS
- Java program provided. You had to compile & run Java MapReduce Program
- Report the Top 10 and their frequency count (to minimize grading). *You were expected to use "sort & head"*

Part 2 — WordCount using Streaming API

- Python program provided. You had to run with command line.
- Report Top 10 and their frquency.

Part 3 — WordCount using mrjob.

- Had to install mrjob.
- -r local ; -r hadoop.

BIG TEACHER GOOFS:

- Neglected to specify format for submission.
- Neglected to tell students to put their names in all submitted source-code.
- Bad input data:
  - Included invalid file
  - Included original dist file EMACS backup.
- Spelled "Shakespeare" wrong.

PS01 — Data Quality Issues

Three Issues:

- _MACOS directory
- .txt
- t8.shakespear.txt~

These files resulted in different results.

- Inconsistent implementations.

```
[cloudera@quickstart L03]$ hdfs dfs -ls Shakespear
Found 40 items
-rw-r--r--   1 cloudera cloudera     113409 2015-11-29 16:09 Shakespear/.txt
-rw-r--r--   1 cloudera cloudera      16149 2015-11-29 16:13 Shakespear/A LOVER'S COMPLAINT.txt
-rw-r--r--   1 cloudera cloudera     109476 2015-11-29 16:12 Shakespear/A MIDSUMMER NIGHT'S DREAM.txt
-rw-r--r--   1 cloudera cloudera     147270 2015-11-29 16:09 Shakespear/ALLS WELL THAT ENDS WELL.txt
-rw-r--r--   1 cloudera cloudera       1333 2015-11-29 16:09 Shakespear/AS YOU LIKE IT.txt
-rw-r--r--   1 cloudera cloudera     177660 2015-11-29 16:09 Shakespear/CYMBELINE.txt
-rw-r--r--   1 cloudera cloudera     159769 2015-11-29 16:12 Shakespear/KING HENRY THE EIGHTH.txt
-rw-r--r--   1 cloudera cloudera     133614 2015-11-29 16:12 Shakespear/KING JOHN.txt
-rw-r--r--   1 cloudera cloudera     194754 2015-11-29 16:12 Shakespear/KING RICHARD III.txt
-rw-r--r--   1 cloudera cloudera     144486 2015-11-29 16:12 Shakespear/KING RICHARD THE SECOND.txt
-rw-r--r--   1 cloudera cloudera     141839 2015-11-29 16:12 Shakespear/LOVE'S LABOUR'S LOST.txt
-rw-r--r--   1 cloudera cloudera     139071 2015-11-29 16:12 Shakespear/MEASURE FOR MEASURE.txt
-rw-r--r--   1 cloudera cloudera     132142 2015-11-29 16:12 Shakespear/MUCH ADO ABOUT NOTHING.txt
-rw-r--r--   1 cloudera cloudera     168709 2015-11-29 16:11 Shakespear/SECOND PART OF KING HENRY IV.txt
-rw-r--r--   1 cloudera cloudera     116104 2015-11-29 16:09 Shakespear/THE COMEDY OF ERRORS.txt
-rw-r--r--   1 cloudera cloudera     147910 2015-11-29 16:12 Shakespear/THE FIRST PART OF HENRY THE SIXTH.txt
-rw-r--r--   1 cloudera cloudera     154042 2015-11-29 16:11 Shakespear/THE FIRST PART OF KING HENRY THE FOURTH.txt
-rw-r--r--   1 cloudera cloudera     173712 2015-11-29 16:12 Shakespear/THE HISTORY OF TROILUS AND CRESSIDA.txt
-rw-r--r--   1 cloudera cloudera     168961 2015-11-29 16:12 Shakespear/THE LIFE OF KING HENRY THE FIFTH.txt
-rw-r--r--   1 cloudera cloudera     123494 2015-11-29 16:12 Shakespear/THE LIFE OF TIMON OF ATHENS.txt
-rw-r--r--   1 cloudera cloudera     132955 2015-11-29 16:12 Shakespear/THE MERCHANT OF VENICE.txt
-rw-r--r--   1 cloudera cloudera     142208 2015-11-29 16:12 Shakespear/THE MERRY WIVES OF WINDSOR.txt
-rw-r--r--   1 cloudera cloudera     165505 2015-11-29 16:12 Shakespear/THE SECOND PART OF KING HENRY THE SIXTH.txt
-rw-r--r--   1 cloudera cloudera     103489 2015-11-29 16:09 Shakespear/THE SONNETS.txt
-rw-r--r--   1 cloudera cloudera     137033 2015-11-29 16:12 Shakespear/THE TAMING OF THE SHREW.txt
-rw-r--r--   1 cloudera cloudera     110718 2015-11-29 16:12 Shakespear/THE TEMPEST.txt
-rw-r--r--   1 cloudera cloudera     161623 2015-11-29 16:12 Shakespear/THE THIRD PART OF KING HENRY THE SIXTH.txt
-rw-r--r--   1 cloudera cloudera     168911 2015-11-29 16:09 Shakespear/THE TRAGEDY OF ANTONY AND CLEOPATRA.txt
-rw-r--r--   1 cloudera cloudera     181884 2015-11-29 16:09 Shakespear/THE TRAGEDY OF CORIOLANUS.txt
-rw-r--r--   1 cloudera cloudera     196760 2015-11-29 16:10 Shakespear/THE TRAGEDY OF HAMLET, PRINCE OF DENMARK.txt
-rw-r--r--   1 cloudera cloudera     129407 2015-11-29 16:12 Shakespear/THE TRAGEDY OF JULIUS CAESAR.txt
-rw-r--r--   1 cloudera cloudera     172756 2015-11-29 16:12 Shakespear/THE TRAGEDY OF KING LEAR.txt
-rw-r--r--   1 cloudera cloudera     116773 2015-11-29 16:12 Shakespear/THE TRAGEDY OF MACBETH.txt
-rw-r--r--   1 cloudera cloudera     176013 2015-11-29 16:12 Shakespear/THE TRAGEDY OF OTHELLO, MOOR OF VENICE.txt
-rw-r--r--   1 cloudera cloudera     156112 2015-11-29 16:12 Shakespear/THE TRAGEDY OF ROMEO AND JULIET.txt
-rw-r--r--   1 cloudera cloudera     134326 2015-11-29 16:12 Shakespear/THE TRAGEDY OF TITUS ANDRONICUS.txt
-rw-r--r--   1 cloudera cloudera     111572 2015-11-29 16:12 Shakespear/THE TWO GENTLEMEN OF VERONA.txt
-rw-r--r--   1 cloudera cloudera     159147 2015-11-29 16:12 Shakespear/THE WINTER'S TALE.txt
-rw-r--r--   1 cloudera cloudera     127183 2015-11-29 16:12 Shakespear/TWELFTH NIGHT; OR, WHAT YOU WILL.txt
-rw-r--r--   1 cloudera cloudera    5458199 2009-03-10 11:13 Shakespear/t8.shakespeare.txt~
[cloudera@quickstart L03]$
```

Different results with “-r local” and “-r hadoop”

With .txt & t8.shakespear.txt~:

EXPERIMENT!

```
"Top10"     [55224, "the"]
"Top10"     [53413, "and"]
"Top10"     [41362, "i"]
"Top10"     [38348, "to"]
"Top10"     [36296, "of"]
"Top10"     [29206, "a"]
"Top10"     [27264, "you"]
"Top10"     [24960, "my"]
"Top10"     [22230, "that"]
"Top10"     [21923, "in"]
```

-r local

```
"Top10"     [54659, "the"]
"Top10"     [52844, "and"]
"Top10"     [40820, "i"]
"Top10"     [38006, "to"]
"Top10"     [35912, "of"]
"Top10"     [28849, "a"]
"Top10"     [26861, "you"]
"Top10"     [24737, "my"]
"Top10"     [21935, "that"]
"Top10"     [21658, "in"]
```

-r hadoop

Remove the ".txt" file

$ hdfs dfs -rm hdfs:///user/cloudera/Shakespear/.txt

$ rm Shakespear/.txt

Fewer words

```
"Top10"     [54659, "the"]
"Top10"     [52844, "and"]
"Top10"     [40820, "i"]
"Top10"     [38006, "to"]
"Top10"     [35912, "of"]
"Top10"     [28849, "a"]
"Top10"     [26861, "you"]
"Top10"     [24737, "my"]
"Top10"     [21935, "that"]
"Top10"     [21658, "in"]
```

-r local

No change

```
"Top10"     [54659, "the"]
"Top10"     [52844, "and"]
"Top10"     [40820, "i"]
"Top10"     [38006, "to"]
"Top10"     [35912, "of"]
"Top10"     [28849, "a"]
"Top10"     [26861, "you"]
"Top10"     [24737, "my"]
"Top10"     [21935, "that"]
"Top10"     [21658, "in"]
```

-r hadoop

Conclusion: -r hadoop ignores the file ".txt"

Remove the t8.shakespear.txt~ file

```
$ hdfs dfs -rm  hdfs:///user/cloudera/Shakespear/t8.shakespeare.txt~
$ rm Shakespear/t8.shakespeare.txt~
```

Fewer words

```
"Top10"     [27016, "the"]
"Top10"     [26116, "and"]
"Top10"     [20139, "i"]
"Top10"     [18808, "to"]
"Top10"     [17739, "of"]
"Top10"     [14236, "a"]
"Top10"     [13212, "you"]
"Top10"     [12257, "my"]
"Top10"     [10814, "that"]
"Top10"     [10691, "in"]
```

Local

Fewer words

```
"Top10"     [27016, "the"]
"Top10"     [26116, "and"]
"Top10"     [20139, "i"]
"Top10"     [18808, "to"]
"Top10"     [17739, "of"]
"Top10"     [14236, "a"]
"Top10"     [13212, "you"]
"Top10"     [12257, "my"]
"Top10"     [10814, "that"]
"Top10"     [10691, "in"]
```

Hadoop

Conclusion: *.txt~ and *.txt files are processed on both

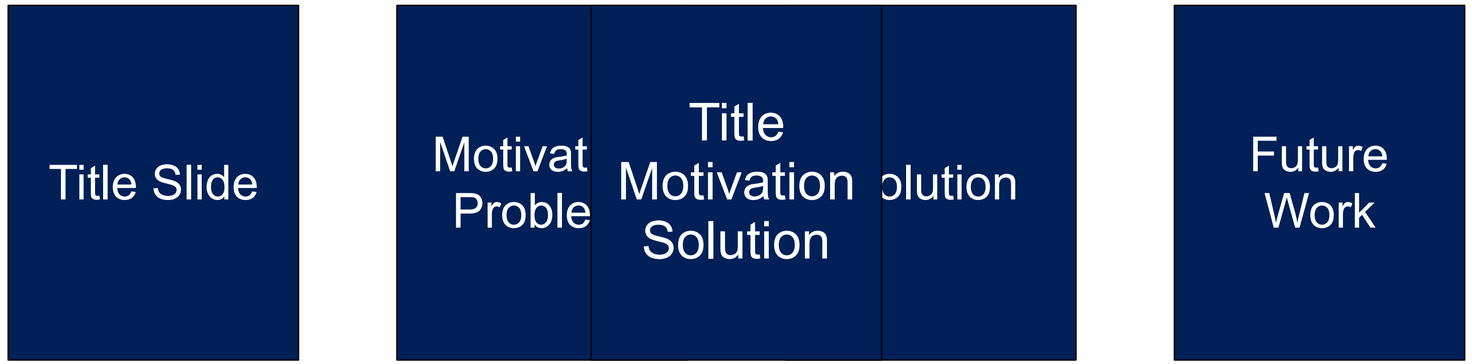
Student Presentations

Student presentations

Purpose of a five minute presentation:

- Convey ***one idea*** about a topic.
- Invite people to learn more.
- Make connections for future work.

Slides:



This week, we will enforce the 5 minute rule

L02 — Student Presentations

Yu Hu	Paper	MapReduce: Simplified Data Processing on Large Clusters
Lu Wang	Paper	GraphChi: Large-Scale Graph Computation on Just a PC
Qinkai Ge	Program	Mahout Samsara

MapReduce Design Patterns

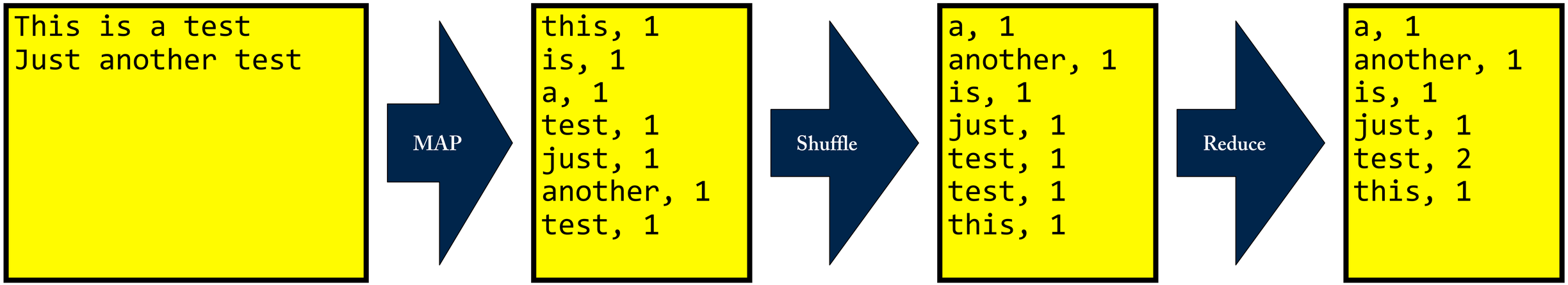
Simple MapReduce tricks with mrjob

Basic pattern — Map/Reduce is really Map/Shuffle/Reduce”

Basic word count idea.

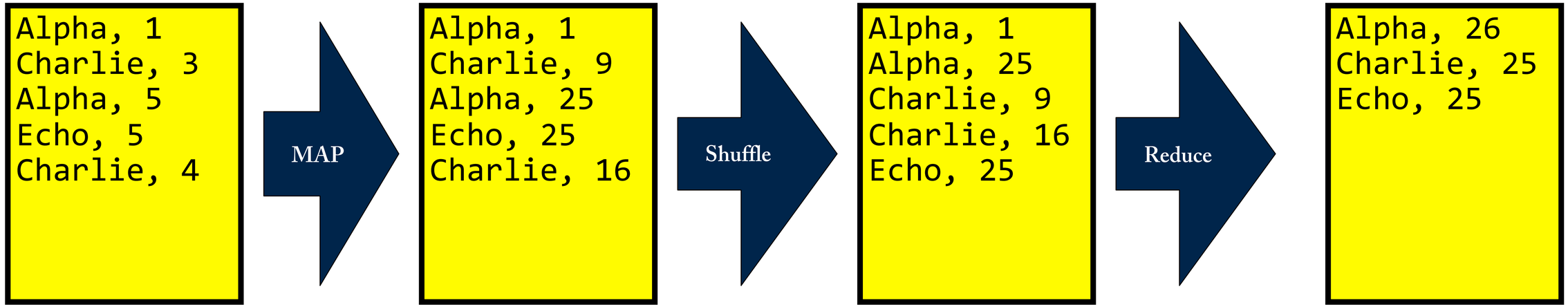
Input:

```
This is a test
Just another test
```




```
def mapper(self, _, line):
    for word in line.strip().split():
        word = filter(str.isalpha,word.lower())
        yield word,1

def reducer(self, key, values):
    yield key, sum(values)
```

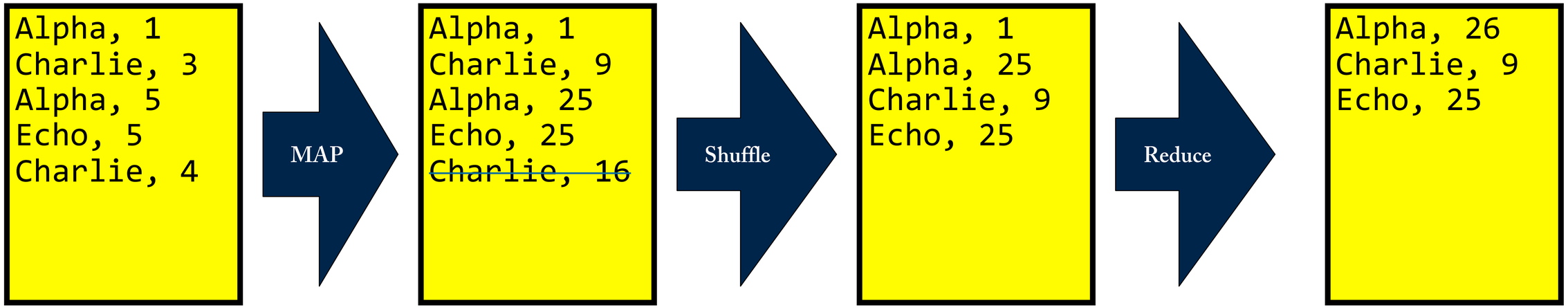
Example: sum of squares




```
class SumOfSquares(MRJob):
    def mapper(self, _, line):
        (label,value) = line.strip().split(",")
        value = int(value)
        yield label,value**2

    def reducer(self, label, values):
        yield label, sum(values)
```

Pattern #1: Filtering — Remove values in the Map or Reduce phase

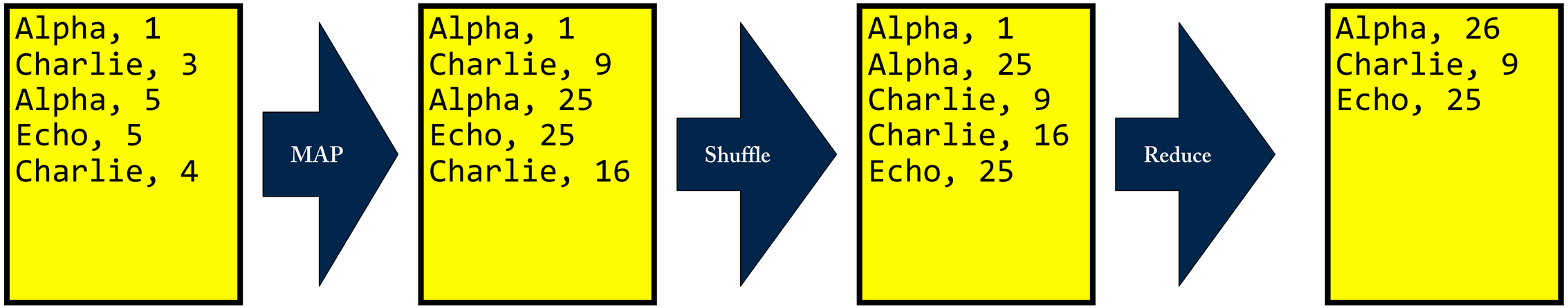
Example: filter even sums of squares — filtering in the Mapper




```
class SumOfSquares(MRJob):
    def mapper(self, _, line):
        (label,value) = line.strip().split(",")
        value = int(value)
        if value % 2 == 1:
            yield label,value**2

    def reducer(self, label, values):
        yield label, sum(values)
```

Filtering — Remove values in the Map or Reduce phase

Example: filter even sums of squares — filtering in the Reducer




```
def isodd(x):
    return x%2==1
```

```
class SumOfSquares(MRJob):
    def mapper(self, _, line):
        (label,value) = line.strip().split(",")
        value = int(value)
        yield label,value**2

    def reducer(self, label, values):
        yield label, sum(filter(isodd,values))
```

Filtering in the Mapper is more efficient:

- **Less I/O**
- **Less CPU**

“Shed work early.”

Two approaches to filtering in python: procedural and functional

```
#!/usr/bin/env python

numbers = [1,2,3,4,5,6,7,8]

# print even numbers

# Procedural:
even_numbers = []
for i in numbers:
    if i%2==0:
        even_numbers.append(i)
print("Even: %s" % even_numbers)
```

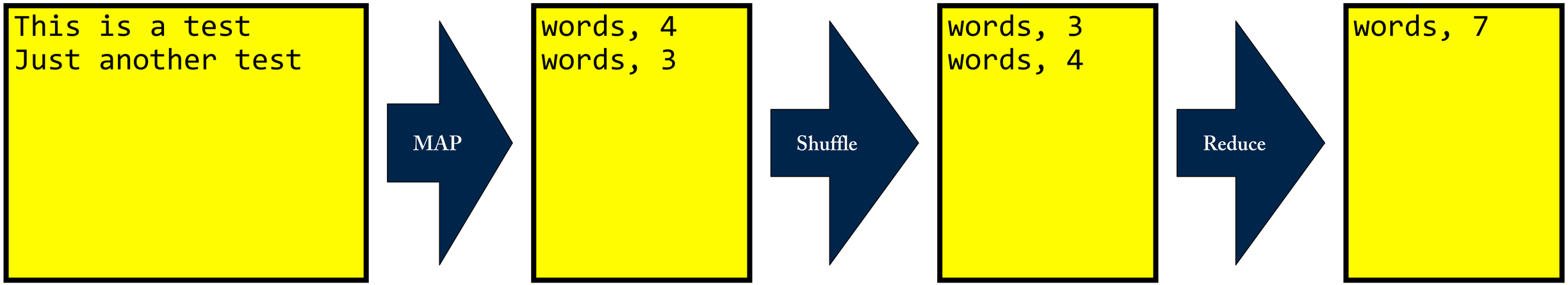
```
# Functional:
def isEven(x):
    return x%2==0
even_numbers = filter(isEven,numbers)
print("Even: %s" % even_numbers)


# With Lambda:
even_numbers = filter(lambda x:x%2==0,numbers)
print("Even: %s" % even_numbers)
```

Pattern #2: Do multiple operations in parallel
Traditional Unix word count — print the number of words

Basic word count idea.

Input:

```
This is a test
Just another test
```




```
def mapper(self, _, line):
      yield "words",len(line.split())

def reducer(self, key, values):
      yield key, sum(values)
```

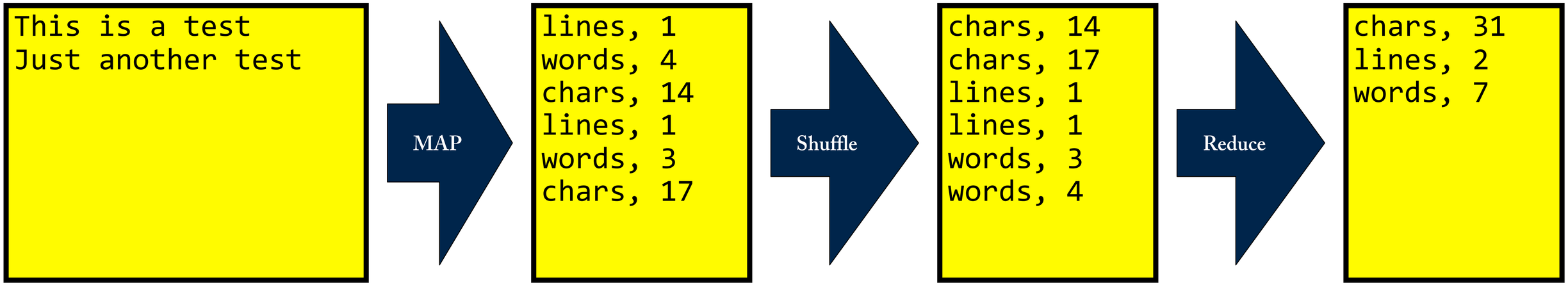
Goal: print # lines, words, chars

Pattern #2: Do multiple operations in parallel
Traditional Unix word count — print the number of words

Basic word count idea.

Input:

```
This is a test
Just another test
```

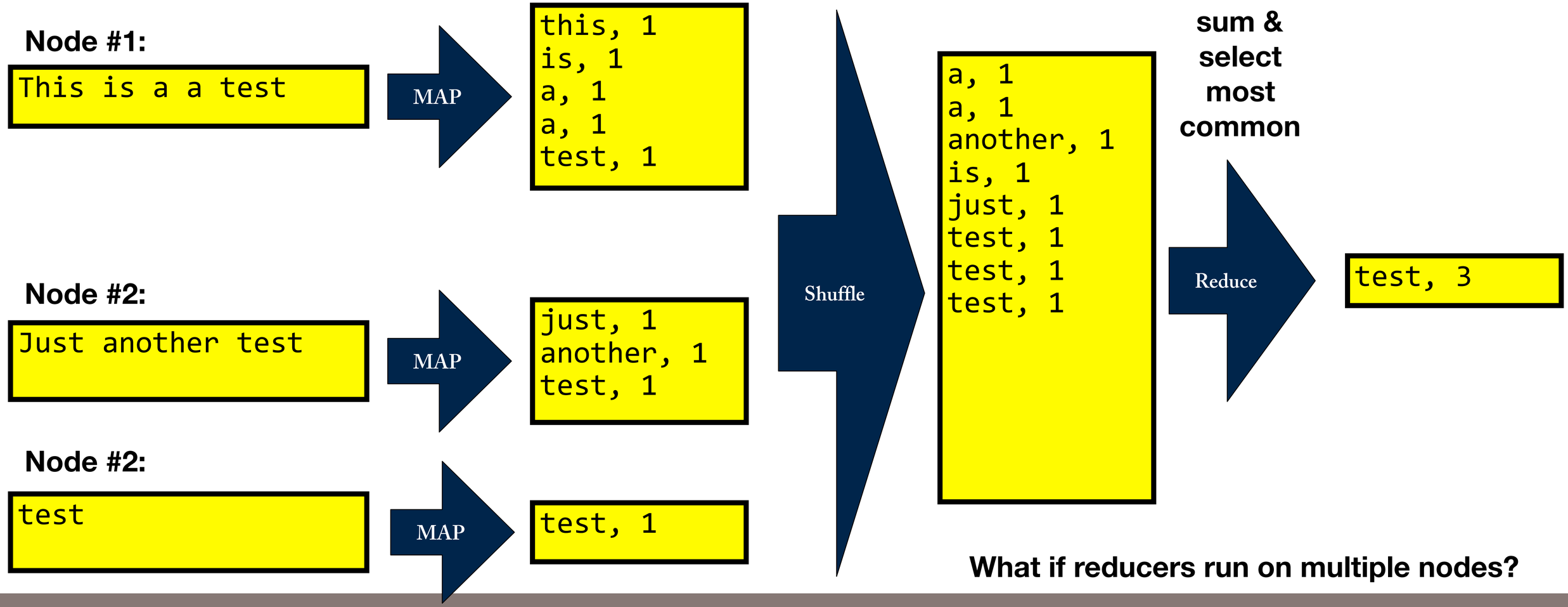



```
def mapper(self, _, line):
    yield "lines", 1
    yield "words",len(line.split())
    yield "chars", len(line)

def reducer(self, key, values):
    yield key, sum(values)
```

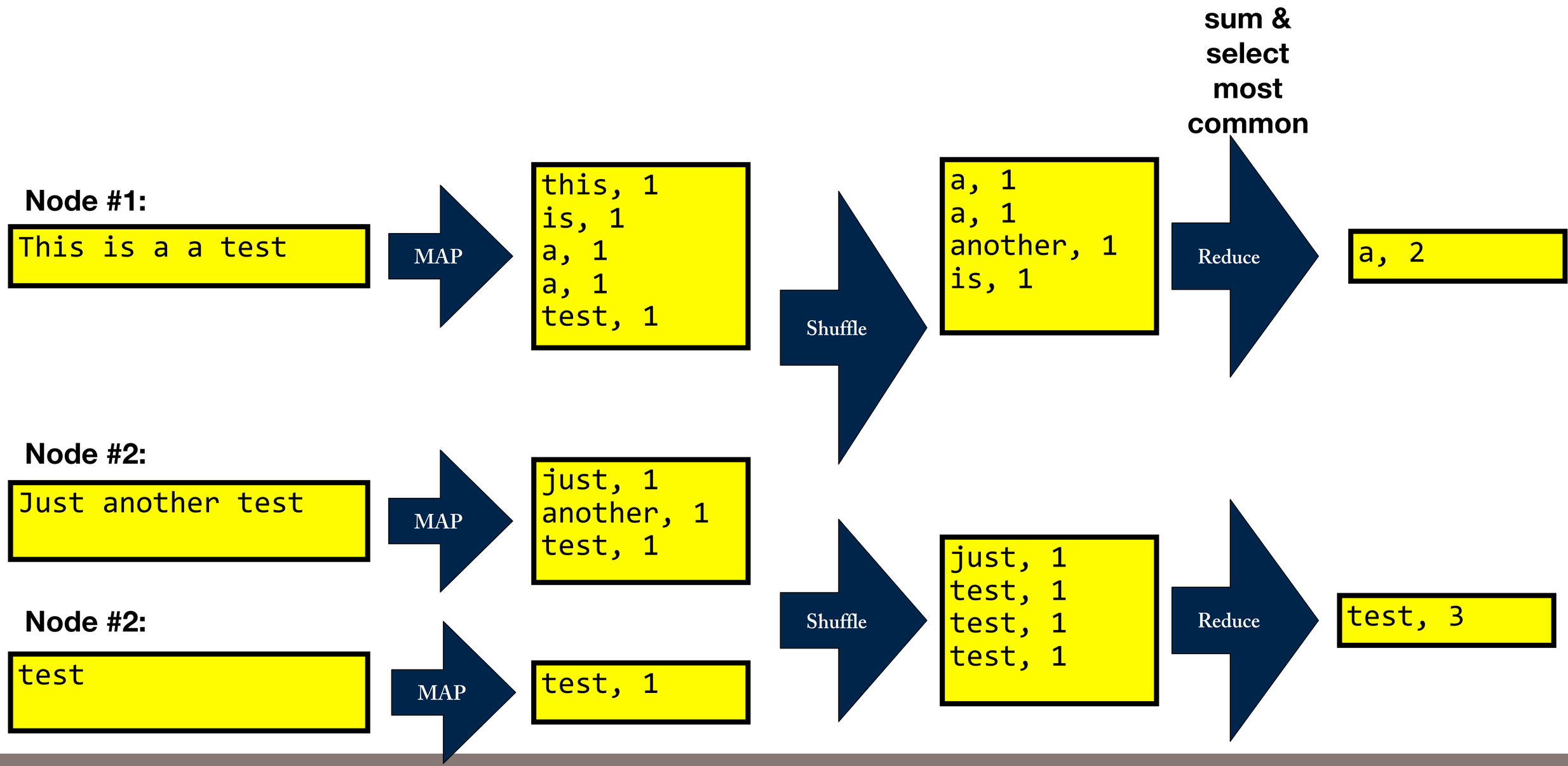
Pattern #3: Top-10
To compute the Top-10 requires reviewing *all the data!*

Basic idea: State between keys must be maintained in the *reducer*

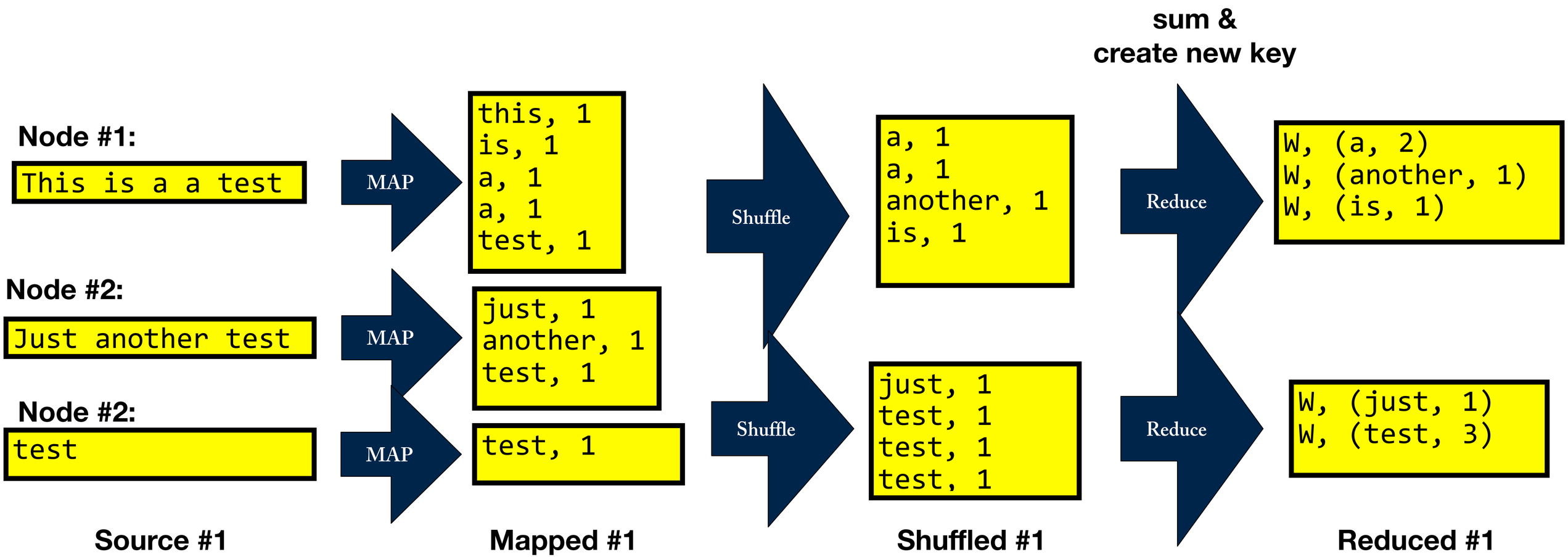
Example: Find most common word:



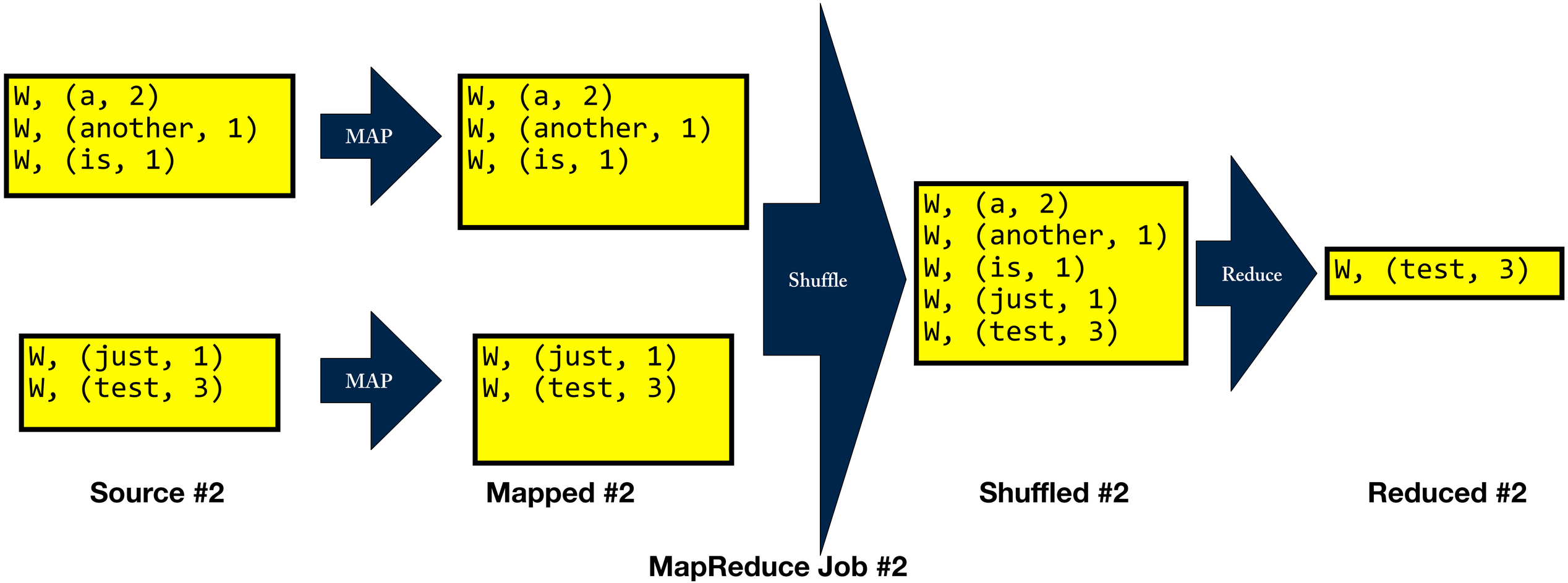
Map/Shuffle/Reduce may use multiple reducers if there are multiple keys. Default Partitioner Guarantee: all of the keys go to the same reducer.



Solution: Perform the sum & selection separately.
Use a new key to route all (k,v) pairs to the same reducer.



If there is only one key, there can be only one reducer.



Here is a *correct* mapReduce WordCount topN implementation

#1

#2

```
...
TOPN=10
class WordCountTopN(MRJob):
    def mapper(self, _, line):
        for word in line.strip().lower().split():
            yield filter(str.isalpha,word),1

    def reducer(self, word, counts):
        yield word, sum(counts)

    def topN_mapper(self,word,count):
        yield "Top"+str(TOPN), (count,word)

    def topN_reducer(self,_,countsAndWords):
       for countAndWord in heapq.nlargest(TOPN,countsAndWords):
            yield _,countAndWord

    def steps(self):
        return [
            MRStep(mapper=self.mapper,
                   reducer=self.reducer),

            MRStep(mapper=self.topN_mapper,
                   reducer=self.topN_reducer) ]
...
```

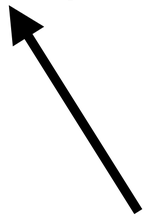
Features:
- Multiple mappers on different nodes
- heapq.nlargest() for efficient topN
- Nicely labels output with operation.

Key point:
To calculate a Max/Min/TopN, the node must see ALL THE DATA

You can also use mapper_init() and mapper_final() and perform your own topN algorithm.

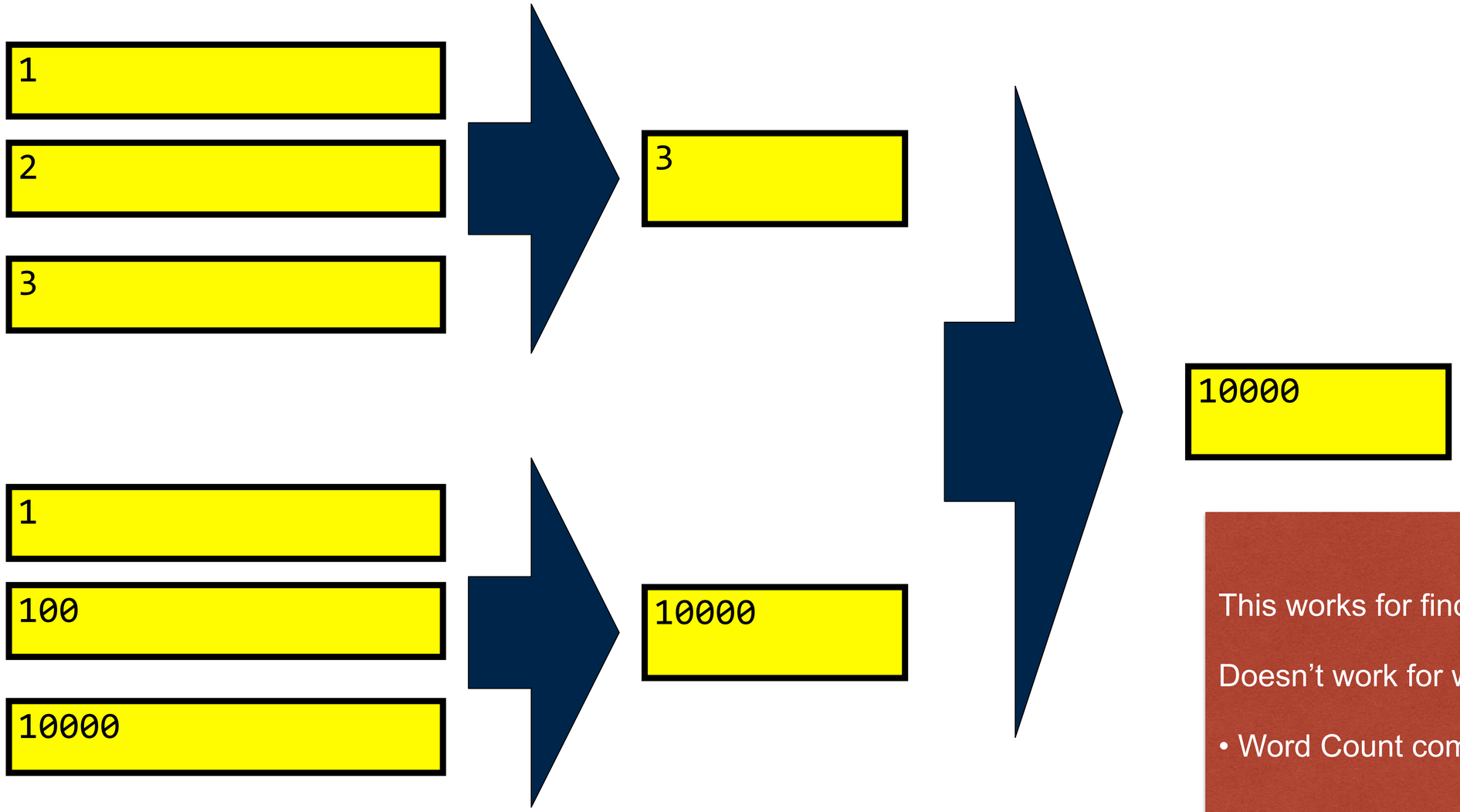
TopN output

```
"Top10" [20, "the"]
"Top10" [15, "to"]
"Top10" [15, "of"]
"Top10" [12, "and"]
"Top10" [7, "that"]
"Top10" [5, "sleep"]
"Top10" [5, "a"]
"Top10" [4, "we"]
"Top10" [4, "be"]
"Top10" [3, "with"]
```



Tab Character
(HadoopStreaming Delimiter)
Better than comma!

You can do a distributed MIN/MAX/TopN, if values from nodes aren't combined



This works for finding the topN records.

Doesn't work for wordCount:

- Word Count combines records!

Counters:
You can increment them anywhere; they are reported when the job ends.

Typical uses:

- Count improperly formatted input lines
- Count missing values
- Global statistics

Code:

```
self.increment_counter( GROUP ,  COUNTER_NAME,  Amount)
```

—*e.g.*

```
self.increment_counter("warn", "missing gross pay", 1)
```

Pattern #4: Joins

We might have data from two tables (data frames) to combine

ID, E#, Observation

```
100, 17, Yellow
101, 35, Red
102, 53, Purple
103, 29, Green
...
993243, 549003, Clear
```

E#, Name

```
17, Chlorine
29, Copper
35, Bromine
53, Iodine
...
549003, Adamantine
```

MAP

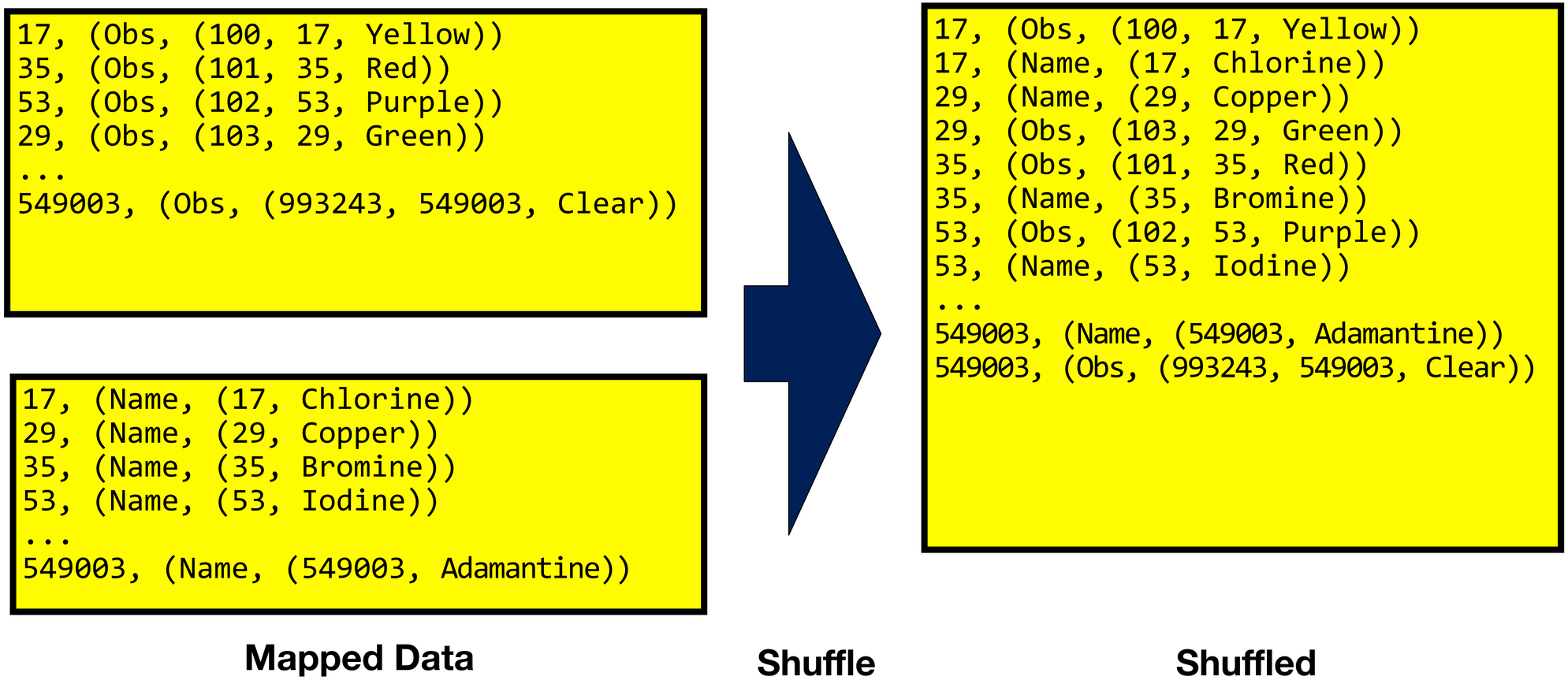
```
17, (Obs, (100, 17, Yellow))
35, (Obs, (101, 35, Red))
53, (Obs, (102, 53, Purple))
29, (Obs, (103, 29, Green))
...
549003, (Obs, (993243, 549003, Clear))
```

```
17, (Name, (17, Chlorine))
29, (Name, (29, Copper))
35, (Name, (35, Bromine))
53, (Name, (53, Iodine))
...
549003, (Name, (549003, Adamantine))
```

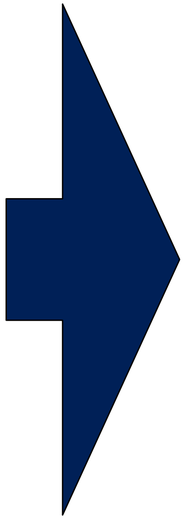
Mapped Data

```
100, (17, Chlorine, Yellow)
101, (35, Bromine, Red)
102, (53, Iodine, Purple)
103, (29, Copper, Green)
...
993243, (549003, Adamantine, Clear)
```

Final Data


17, (Obs, (100, 17, Yellow))
35, (Obs, (101, 35, Red))
53, (Obs, (102, 53, Purple))
29, (Obs, (103, 29, Green))
...
549003, (Obs, (993243, 549003, Clear))
17, (Name, (17, Chlorine))
29, (Name, (29, Copper))
35, (Name, (35, Bromine))
53, (Name, (53, Iodine))
...
549003, (Name, (549003, Adamantine))
Mapped Data
Shuffle
17, (Obs, (100, 17, Yellow))
17, (Name, (17, Chlorine))
29, (Name, (29, Copper))
29, (Obs, (103, 29, Green))
35, (Obs, (101, 35, Red))
35, (Name, (35, Bromine))
53, (Obs, (102, 53, Purple))
53, (Name, (53, Iodine))
...
549003, (Name, (549003, Adamantine))
549003, (Obs, (993243, 549003, Clear))
Shuffled

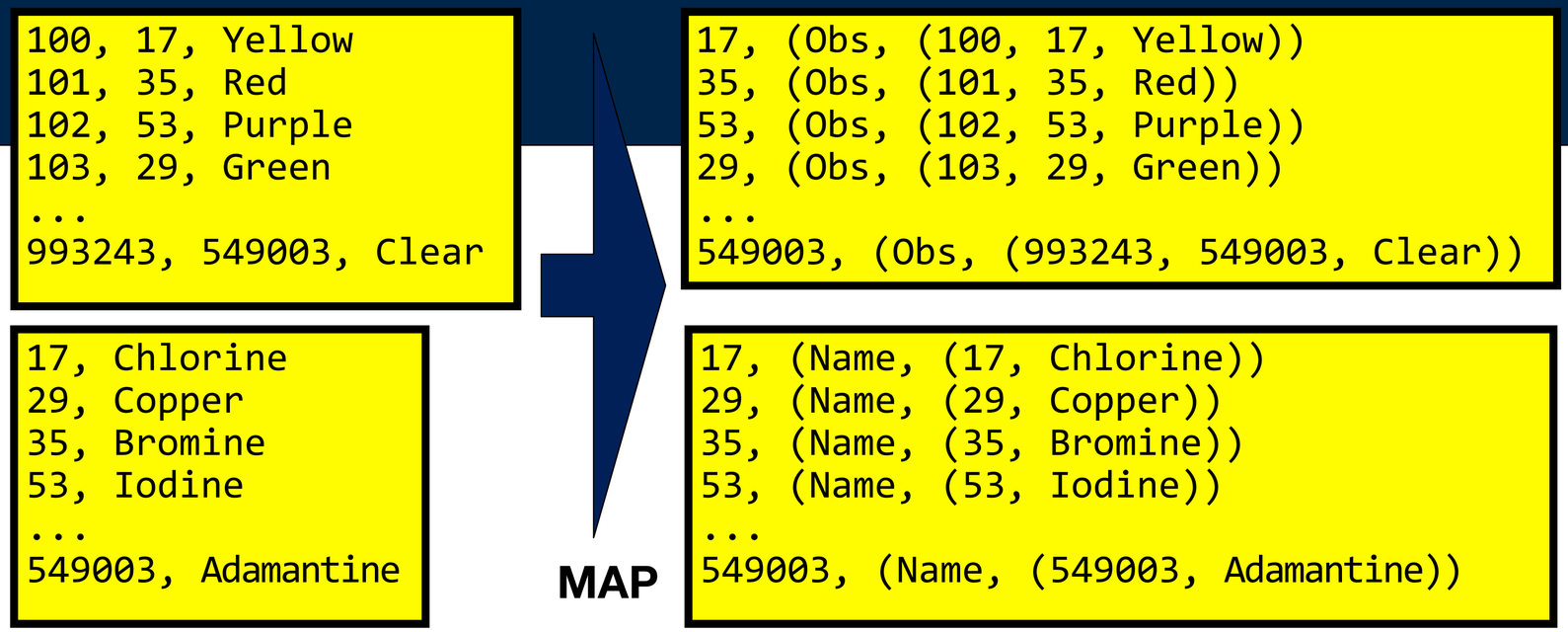
```
17, (Obs, (100, 17, Yellow))
17, (Name, (17, Chlorine))
29, (Name, (29, Copper))
29, (Obs, (103, 29, Green))
35, (Obs, (101, 35, Red))
35, (Name, (35, Bromine))
53, (Obs, (102, 53, Purple))
53, (Name, (53, Iodine))
...
549003, (Name, (549003, Adamantine))
549003, (Obs, (993243, 549003, Clear))
```



```
100, (17, Chlorine, Yellow)
101, (35, Bromine, Red)
102, (53, Iodine, Purple)
103, (29, Copper, Green)
...
993243, (549003, Adamantine, Clear)
```

Shuffled **Reduce** **Reduced**

Join map code in mrjob




```
def mapper(self, _, line):
    fields = line.split(", ")
    if len(fields)==3:
        self.increment_counter("Info","Obs Count",1)
        yield fields[1], ("Obs", fields)
    elif len(fields)==2:
        self.increment_counter("Info","Name Count",1)
        yield fields[0], ("Name",fields)
    else:
        self.increment_counter("Warn","Invalid Data",1)
```

Counters for
“situational awareness”

Join reduce code in mrjob

```
17, (Obs, (100, 17, Yellow))
35, (Obs, (101, 35, Red))
53, (Obs, (102, 53, Purple))
29, (Obs, (103, 29, Green))
...
549003, (Obs, (993243, 549003, Clear))
```

```
17, (Name, (17, Chlorine))
29, (Name, (29, Copper))
35, (Name, (35, Bromine))
53, (Name, (53, Iodine))
...
549003, (Name, (549003, Adamantine))
```

```
100, (17, Chlorine, Yellow)
101, (35, Bromine, Red)
102, (53, Iodine, Purple)
103, (29, Copper, Green)
...
993243, (549003, Adamantine, Clear)
```

SORT_VALUES=True

```
def reducer(self, key, values):
    name = None
    for v in values:
        if len(v)!=2:
            self.increment_counter("Warn","Invalid Join",1)
            continue
        if v[0]=='Name':
            name = v[1]
            continue
        if v[0]=='Obs':
            obs = v[1]
            if name:
                assert key==name[0]
                assert key==obs[1]
                yield obs[0],(obs[1],name[1],obs[2])
            else:
                self.increment_counter("Warn","Obs without Name")
                yield obs[0],(obs[1],"n/a",obs[2])
```

Runtime Error Checking

NOTE: THIS CODE ASSUMES THAT VALUES ARE SORTED.

SEE SORT_VALUES=True

SQL

What you need to know

SQL — Structured Query Language

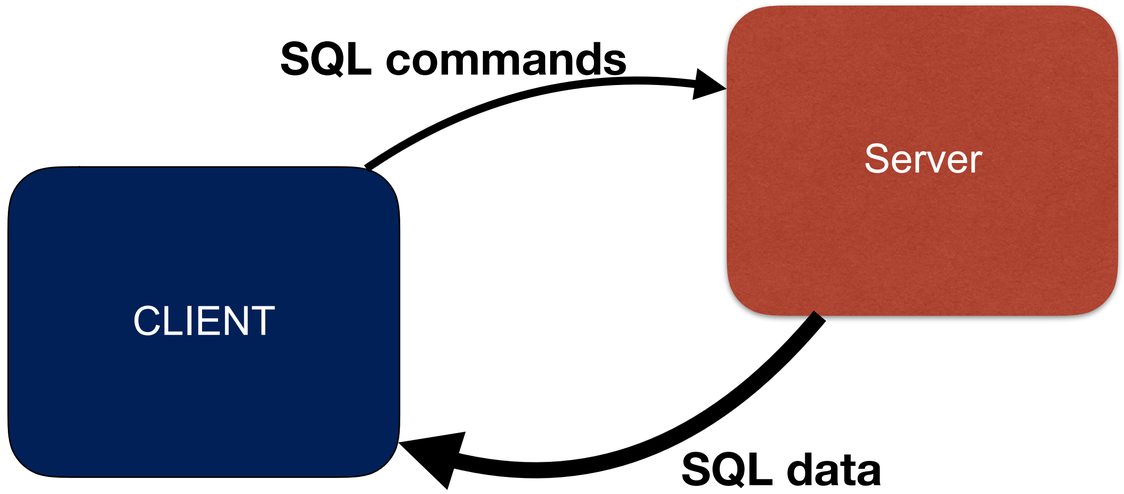
SQL is a language for communicating with data bases.

Server:

- A remote databsae
- A local databse on the same computer
- An embedded database (sqlite3)

Client:

- Your program!
- Excel / Access



Typical SQL servers:

- Sqlite3
- MySQL
- PostgreSQL
- Oracle SQL Server
- IBM Informix
- Microsoft SQL Server

SQL — Advantages and Disadvantages

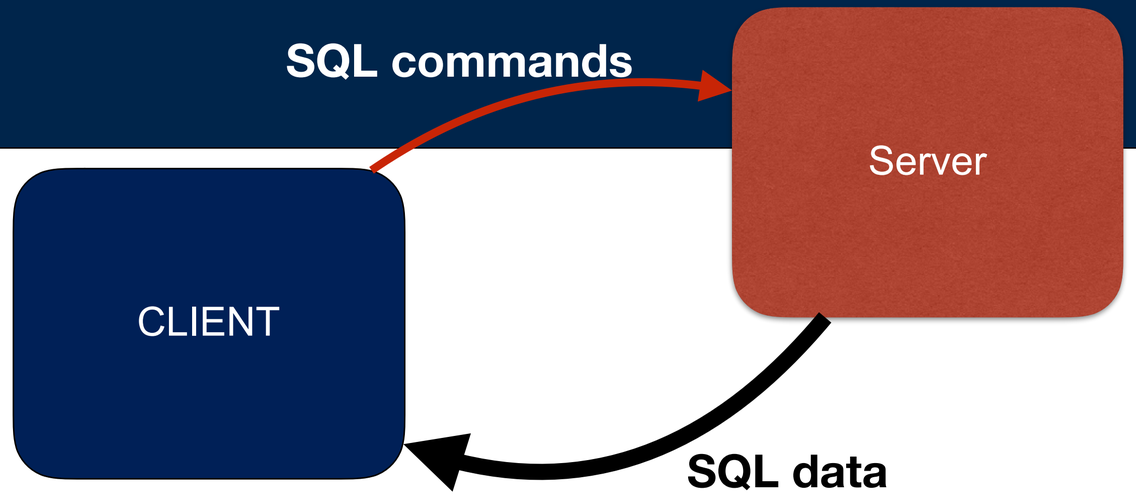
Advantages:

- Standard — implemented by many different vendors & systems
- Declarative — describe what you want, DB figures out how to do it
- Optimizer — DB figures fastest way to do it.

Disadvantages:

- Another language to learn.
- Overhead of converting data to text
- Data model does not fit well with graph data (e.g. trees, social networks, etc.)
- SQLs are all a little different.

SQL Terminology



Server — Where the data are kept.

Client — Issues commands / gets data

Data are arranged *tables.*

Tables have *rows* and *columns*:

ID	EID	Color
100	17	Yellow
101	35	Red
102	53	Purple
103	29	Green

Observations

EID	Name
17	Chlorine
29	Copper
35	Bromine
53	Iodine

Names

A *database* is a collection of tables.

A *database server* can have multiple *databases.*

SQL Statements you need to know:
SELECT *expression[,expression ...]* FROM *table* [WHERE *expression*];

```
sqlite> .mode column
sqlite> .header on
sqlite> select * from observations;
id          eid         color
----------  ----------  ----------
100         17          Yellow
101         35          Red
102         53          Purple
103         29          Green

sqlite> select id,eid from observations;
id          eid
----------  ----------
100         17
101         35
102         53
103         29

sqlite> select * from observations where id>101;
id          eid         color
----------  ----------  ----------
102         53          Purple
103         29          Green
sqlite>
```

ID	EID	Color
100	17	Yellow
101	35	Red
102	53	Purple
103	29	Green

Observations

EID	Name
17	Chlorine
29	Copper
35	Bromine
53	Iodine

Names

SQL Statements you need to know:
SELECT ... [ORDER BY *expression*] [LIMIT [start,] len];

```
sqlite> select * from observations order by eid;
id          eid         color
----------  ----------  ----------
100         17          Yellow
103         29          Green
101         35          Red
102         53          Purple
sqlite> select * from observations order by color;
id          eid         color
----------  ----------  ----------
103         29          Green
102         53          Purple
101         35          Red
100         17          Yellow
sqlite>

sqlite> select * from observations limit 1;
id          eid         color
----------  ----------  ----------
100         17          Yellow
sqlite> select * from observations limit 2,1;
id          eid         color
----------  ----------  ----------
102         53          Purple
sqlite> select * from observations limit 2,1;
```

ID	EID	Color
100	17	Yellow
101	35	Red
102	53	Purple
103	29	Green

Observations

EID	Name
17	Chlorine
29	Copper
35	Bromine
53	Iodine

Names

SQL Statements you need to know: SELECT ... JOIN ...;

Inner Join:

```
sqlite> SELECT OBS.id, OBS.eid, N.name, OBS.color FROM observations \
        AS OBS join names as N ON OBS.eid = N.eid;
id          eid         name        color
----------  ----------  ----------  ----------
100         17          Chlorine    Yellow
101         35          Bromine     Red
102         53          Iodine      Purple
103         29          Copper      Green
sqlite>
```

Natural Join:

```
sqlite> select * from observations natural join names;
id          eid         color       name
----------  ----------  ----------  ----------
100         17          Yellow      Chlorine
101         35          Red         Bromine
102         53          Purple      Iodine
103         29          Green       Copper
```

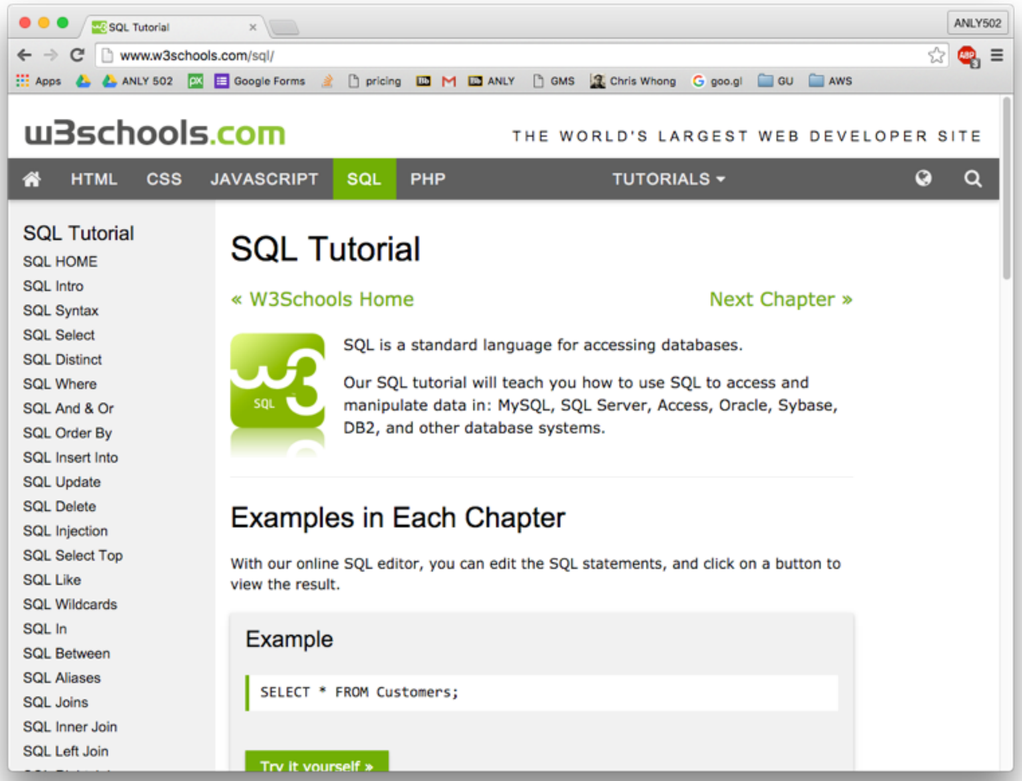
ID	EID	Color
100	17	Yellow
101	35	Red
102	53	Purple
103	29	Green

Observations

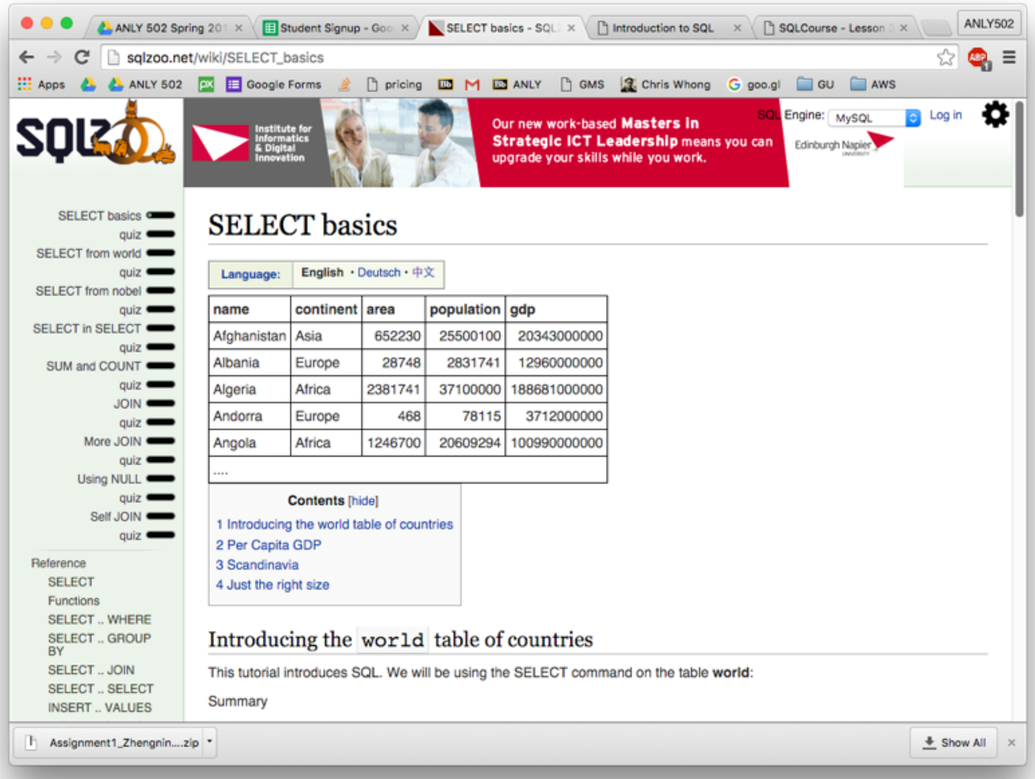
EID	Name
17	Chlorine
29	Copper
35	Bromine
53	Iodine

Names

SQL Tutorial Sites



http://www.w3schools.com
SQL eval in another window



http://sqlzoo.net
SQL eval in same window

PS02

PS02 — Due February 5th , 2016

Part 1 — Short answer about clock time to process input files.

- Key fact you need to know: large text files can be split between different nodes. Gzip-compressed files can't be split.

Part 2 — Practice Map/Reduce jobs.

- Code is provided to parse Apache log files.
- Compressed Apache log files are provided.
- Your job:
  - *Try the code that reports MIN & MAX date for each log file.*
  - *Print # of URLs served each day.*
  - *Filter "Special:" out of report.*
  - *Report # of times each Wiki Page is accessed.*
  - *Display top-10 with a two-step MRJOB.*

Part 3 — Joins

- *Join Wiki logfiles with Geolocation Data*
- *Report # of hits per country.*
- *Report Top-10 countries (requires a three-step MRJOB)*

py.test — add tests to your code!

Install with: **sudo yum install pytest**

Code to include:

```
# Import pytest if we have it available
try:
    import pytest
except ImportError as e:
    pass
```

Example of test:

```
# Tests
# test with pytest
demo_line1 = '172.16.0.3 - - [25/Sep/2002:14:04:19 +0200] "GET /hello.html HTTP/1.1" 401 - "" "Moz
illa/5.0 (X11; U; Linux i686; en-US; rv:1.1) Gecko/20020827"'
def test_weblog():
    obj = Weblog(demo_line1)
    assert obj.ipaddr=="172.16.0.3"
    assert obj.timestamp==parser.parse("25-Sep-2002 14:14:19 +0200")
    assert obj.request=="GET /hello.html HTTP/1.1"
    assert obj.result==401
    assert obj.agent=="Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.1) Gecko/20020827"
```

```
File  Edit  View  Search  Terminal  Help
================================================== 1 failed in 0.03 seconds ==================================================
[cloudera@quickstart PS02]$ py.test weblog.py
================================================== test session starts ==================================================
platform linux2 -- Python 2.6.6 -- pytest-2.3.5
collected 1 items

weblog.py F

======================================================= FAILURES ========================================================
_____________________________________________________ test_weblog _____________________________________________________

    def test_weblog():
        obj = Weblog(demo_line1)
>       assert obj.ip=="172.16.0.3"
E       AttributeError: 'Weblog' object has no attribute 'ip'

weblog.py:42: AttributeError
================================================== 1 failed in 0.03 seconds ==================================================
[cloudera@quickstart PS02]$ py.test weblog.py
================================================== test session starts ==================================================
platform linux2 -- Python 2.6.6 -- pytest-2.3.5
collected 1 items

weblog.py F

======================================================= FAILURES ========================================================
_____________________________________________________ test_weblog _____________________________________________________

    def test_weblog():
        obj = Weblog(demo_line1)
        assert obj.ipaddr=="172.16.0.3"
>       assert obj.timestamp==parser.parse("25-Sep-2002 14:14:19 +0200")
E       assert '25/Sep/2002:14:04:19 +0200' == datetime.datetime(2002, 9, 25, 14, 14, 19, tzinfo=tzoffset(None, 7200))
E        +  where '25/Sep/2002:14:04:19 +0200' = <weblog.Weblog object at 0x1570c88>.timestamp
E        +  and   datetime.datetime(2002, 9, 25, 14, 14, 19, tzinfo=tzoffset(None, 7200)) = <function parse at 0x15f9f50>('25-
Sep-2002 14:14:19 +0200')
E        +    where <function parse at 0x15f9f50> = parser.parse

weblog.py:43: AssertionError
================================================== 1 failed in 0.03 seconds ==================================================
[cloudera@quickstart PS02]$
```

Tips for writing more reliable code

Use assert statements.

Use counters to track how many records are processed & ignored.

You descriptive variable names.

Write comments that explain what the code is supposed to do.

DRAM Errors in the Wild: A Large-Scale Field Study

Bianca Schroeder
Dept. of Computer Science
University of Toronto
Toronto, Canada
bianca@cs.toronto.edu

Eduardo Pinheiro
Google Inc.
Mountain View, CA

Wolf-Dietrich Weber
Google Inc.
Mountain View, CA

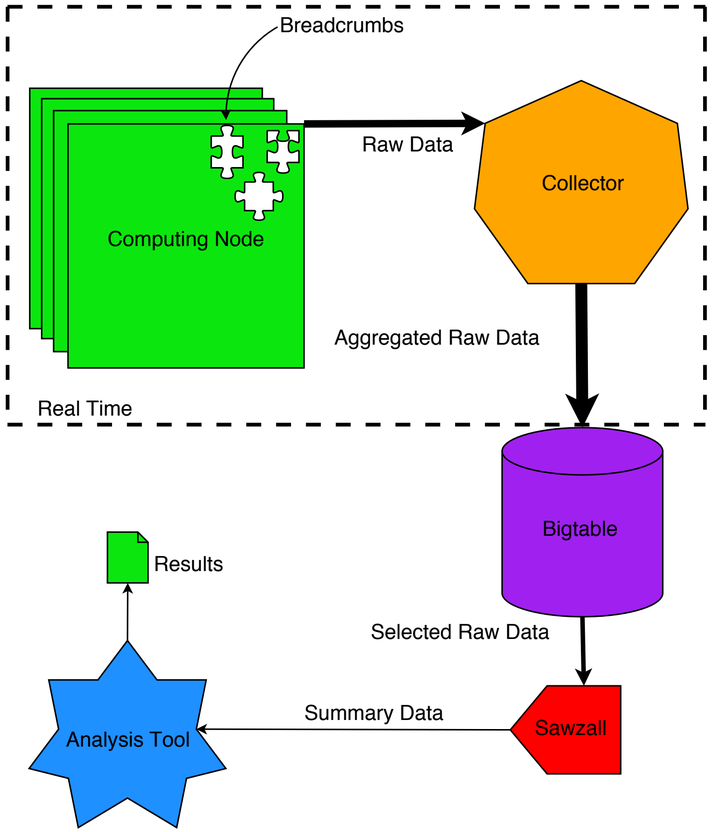


Figure 1: Collection, storage, and analysis architecture.

Table 1: Memory errors per year:

Platf.	Tech.	Per machine				
		CE Incid. (%)	CE Rate Mean	CE Rate C.V.	CE Median Affct.	UE Incid. (%)
A	DDR1	45.4	19,509	3.5	611	0.17
B	DDR1	46.2	23,243	3.4	366	–
C	DDR1	22.3	27,500	17.7	100	2.15
D	DDR2	12.3	20,501	19.0	63	1.21
E	FBD	–	–	–	–	0.27
F	DDR2	26.9	48,621	16.1	25	4.15
Overall	–	32.2	22,696	14.0	277	1.29

Platf.	Tech.	Per DIMM				
		CE Incid. (%)	CE Rate Mean	CE Rate C.V.	CE Median Affct.	UE Incid. (%)
A	DDR1	21.2	4530	6.7	167	0.05
B	DDR1	19.6	4086	7.4	76	–
C	DDR1	3.7	3351	46.5	59	0.28
D	DDR2	2.8	3918	42.4	45	0.25
E	FBD	–	–	–	–	0.08
F	DDR2	2.9	3408	51.9	15	0.39
Overall	–	8.2	3751	36.3	64	0.22

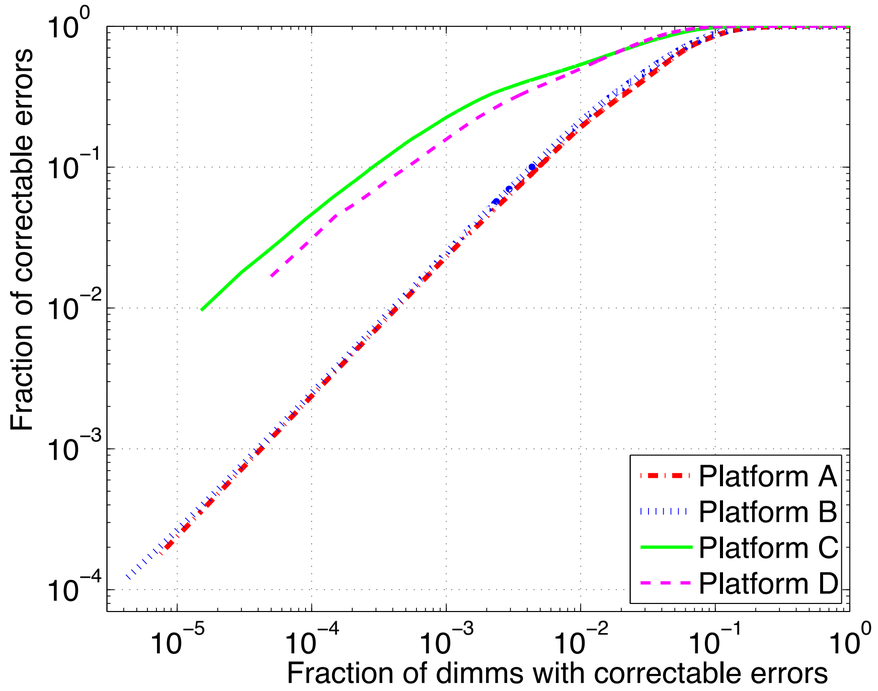


Figure 2: The distribution of correctable errors over DIMMs: *The graph plots the fraction Y of all errors in a platform that is made up by the fraction X of DIMMs with the largest number of errors.*

Table 2: Errors per DIMM by DIMM type/manufacturer

Pf	Mfg	GB	Incid. CE (%)	Incid. UE (%)	Mean CE rate	C.V. CE	CEs/ GB
A	1	1	20.6	0.03	4242	6.9	4242
	1	2	19.7	0.07	4487	5.9	2244
	2	1	6.6		1496	11.9	1469
	3	1	27.1	0.04	5821	6.2	5821
	4	1	5.3	0.03	1128	13.8	1128
B	1	1	20.3	–	3980	7.5	3980
		2	18.4	–	5098	6.8	2549
	2	1	7.9	–	1841	11.0	1841
		2	18.1	–	2835	8.9	1418
C	1	1	3.6	0.21	2516	69.7	2516
	4	1	2.6	0.43	2461	57.2	2461
	5	2	4.7	0.22	10226	12.0	5113
D	6	2	2.7	0.24	3666	39.4	1833
		4	5.7	0.24	12999	23.0	3250
E	1	2	–	0	–	–	–
		4	–	0.13	–	–	–
	2	2	–	0.05	–	–	–
		4	–	0.27	–	–	–
	5	2	–	0.06	–	–	–
		4	–	0.14	–	–	–
F	1	2	2.8	0.20	2213	53.0	1107
		4	4.0	1.09	4714	42.8	1179

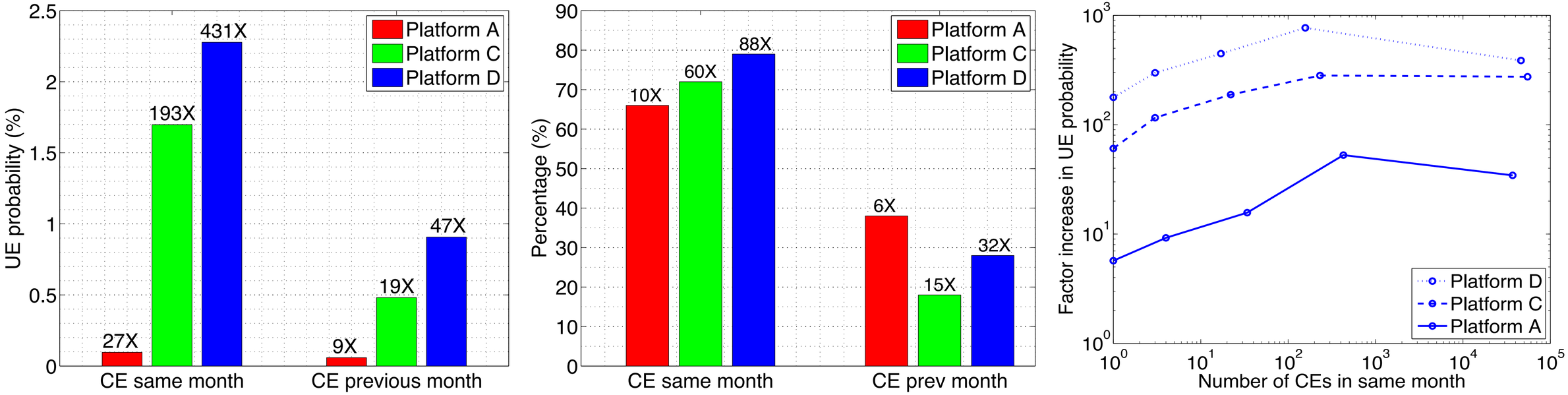
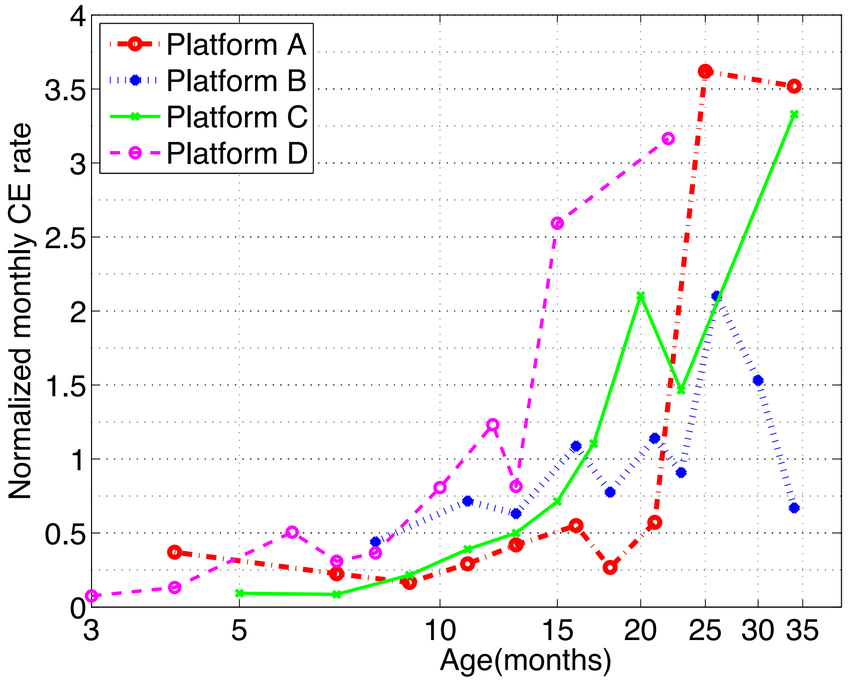


Figure 4: Correlations between correctable and uncorrectable errors in the same DIMM: *The left graph shows the UE probability in a month depending on whether there were CEs in the same month or in the previous month. The numbers on top of the bars give the increase in UE probability compared to a month without CEs (three left-most bars) and the case where there were no CEs in the previous month (three right-most bars). The middle graph shows how often a UE was preceded by a CE in the same/previous month. The right graph shows the factor increase in the probability of observing an UE as a function of the number of CEs in the same month.*



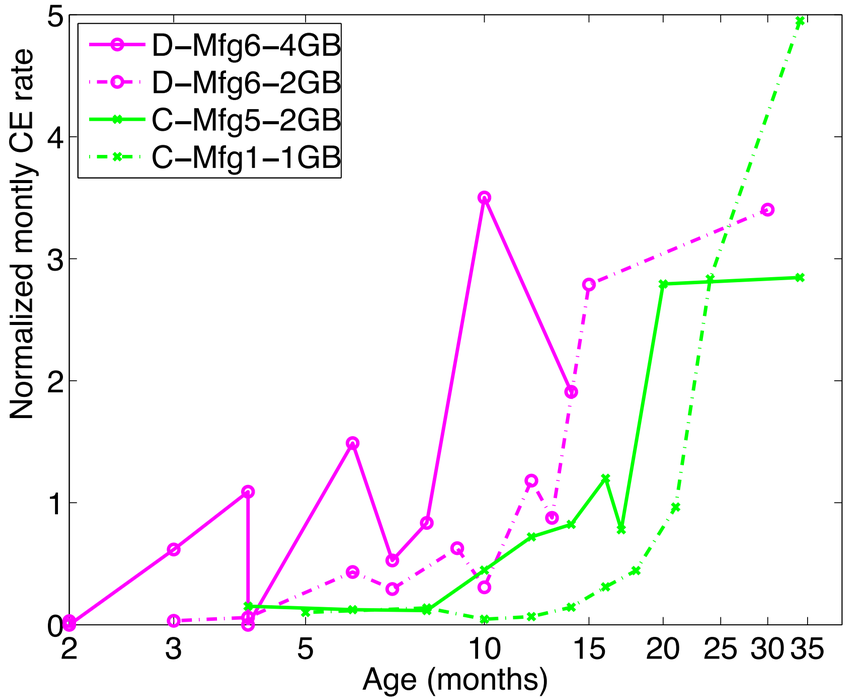


Figure 10: The effect of age: *The normalized monthly rate of experiencing a CE as a function of age by platform (left) and for four common DIMM configurations (right). We consider only DIMMs manufactured after July 2005, to exclude very old platforms (due to a rapidly decreasing population).*

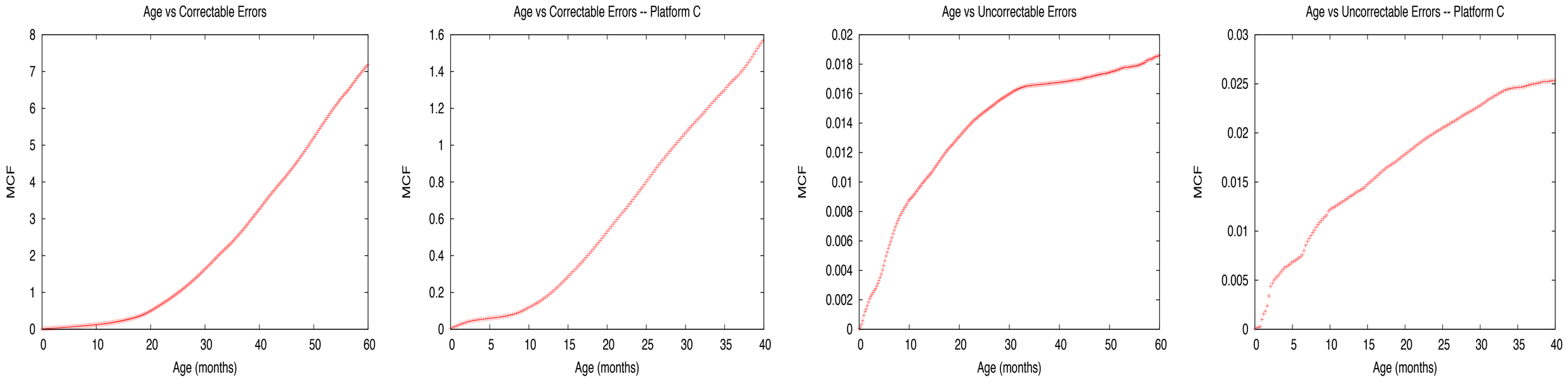


Figure 11: The effect of age: *The two graphs on the left show the mean cumulative function for CEs for all DIMMs in production in January 2007 until November 2008, and for Platform C , respectively. The two graphs on the right show for the same two populations the mean cumulative function for UEs.*