Introduction to Data Science

Lecture 09; June 1st, 2015

Ernst Henle
ErnstHe@UW.edu
Skype: ernst.predixion

Agenda

- Social Interactions / Announcements
  - Continue the LinkedIn group for the next 2 quarters
- Review EAV and Sparse Matrices
- Quiz 09a Sparse Matrices and EAV
- A (brief) introduction to Python for Data Science by Matt Danielson
- Break
- Hadoop HDFS
- Quiz 09b Hadoop
- Hadoop Labs
- Break
- Hadoop Labs ccontinued

Social Interactions and Announcements

- Continue the LinkedIn group for the next 2 quarters
- Next week is the last class of the 1st quarter
  - My goals for this quarter:
    - Breadth (Relational Algebra, NoSQL, Hadoop, MapReduce, Data Preparation, Data flow, Machine Learning, Graph Data, Guest Lectures)
    - Feedback! Feedback for every homework and every quiz

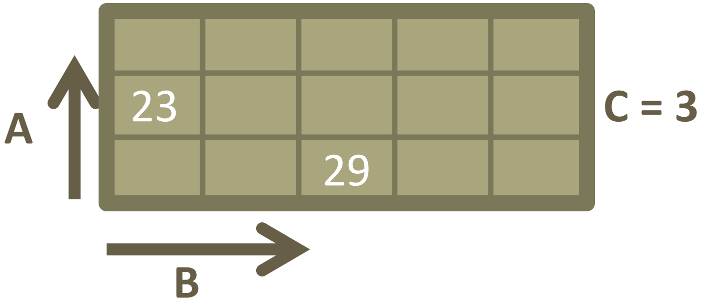
      Challenging Lectures

      Encourage interactions among students
- After-hours with Matt Danielson at Rock Bottom. One Block from Class: 1333 Fifth Ave. Seattle, WA98101

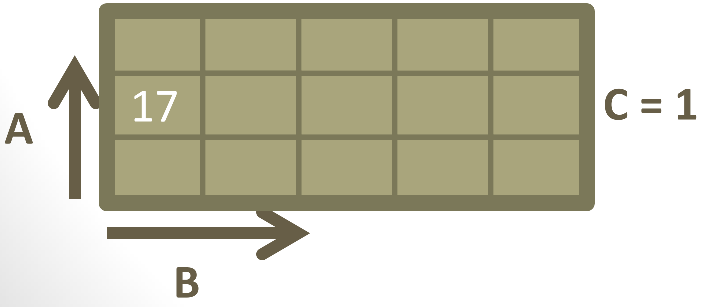
Sparse Matrices and EAV Review

Sparse Matrices: Review (1)

- Data: Real estate survey of single-family houses in downtown Seattle.   Cell values are number (**N**) of houses found for sale.
  - **A**: Area in 1000’s of square feet
  - **B**: Number of Bathrooms
  - **C**: Cost in $100,000.-
- Task: Create sparse matrices

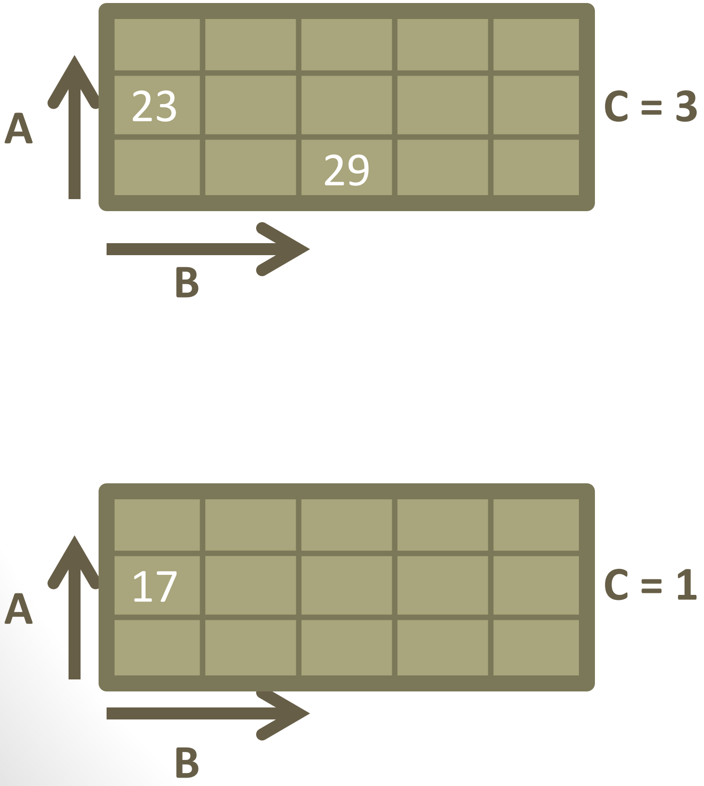


23				
		29		



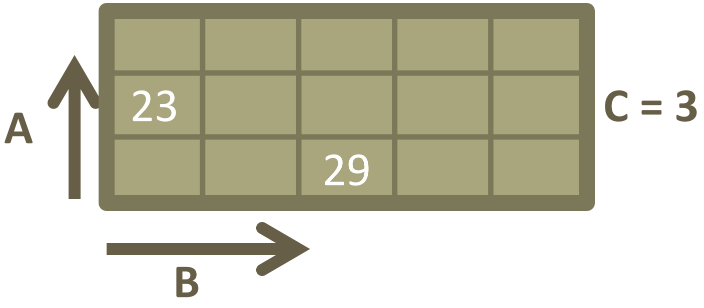
17				

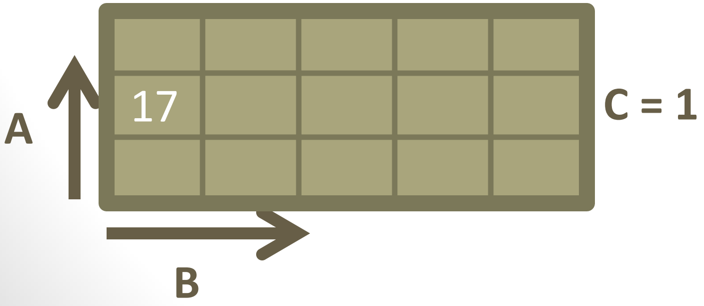
Sparse Matrices: Review (2)



A	B	C	N
2	1	3	23
1	3	3	29
2	1	1	17

Sparse Matrices: Review (3)

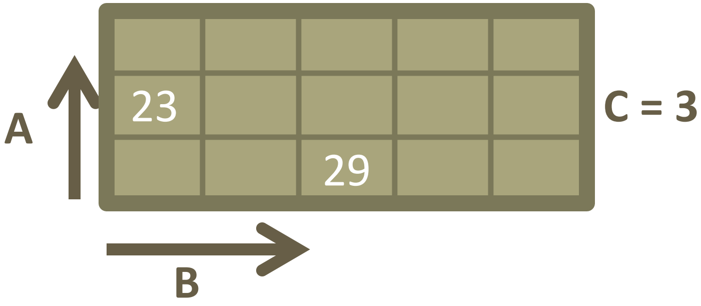


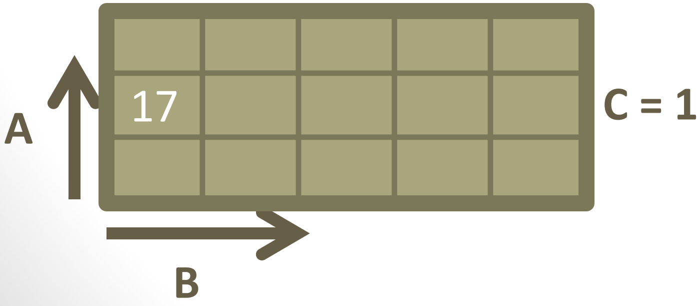


A	B	C	N
2	1	3	23
1	3	3	29
2	1	1	17

R	C	M
1	A	2
1	B	1
1	C	3
1	N	23
2	A	1
2	B	3
2	C	3
2	N	29
3	A	2
3	B	1
3	C	1
3	N	17

Sparse Matrices: Review (4)





A	B	C	N	CA
2	1	3	23	1.5
1	3	3	29	3
2	1	1	17	0.5

R	C	M
1	A	2
1	B	1
1	C	3
1	N	23
2	A	1
2	B	3
2	C	3
2	N	29
3	A	2
3	B	1
3	C	1
3	N	17
1	CA	1.5
2	CA	3
3	CA	0.5

Sparse Matrices: Review (5)

See: **MatrixAlgebraUpdated.sql**

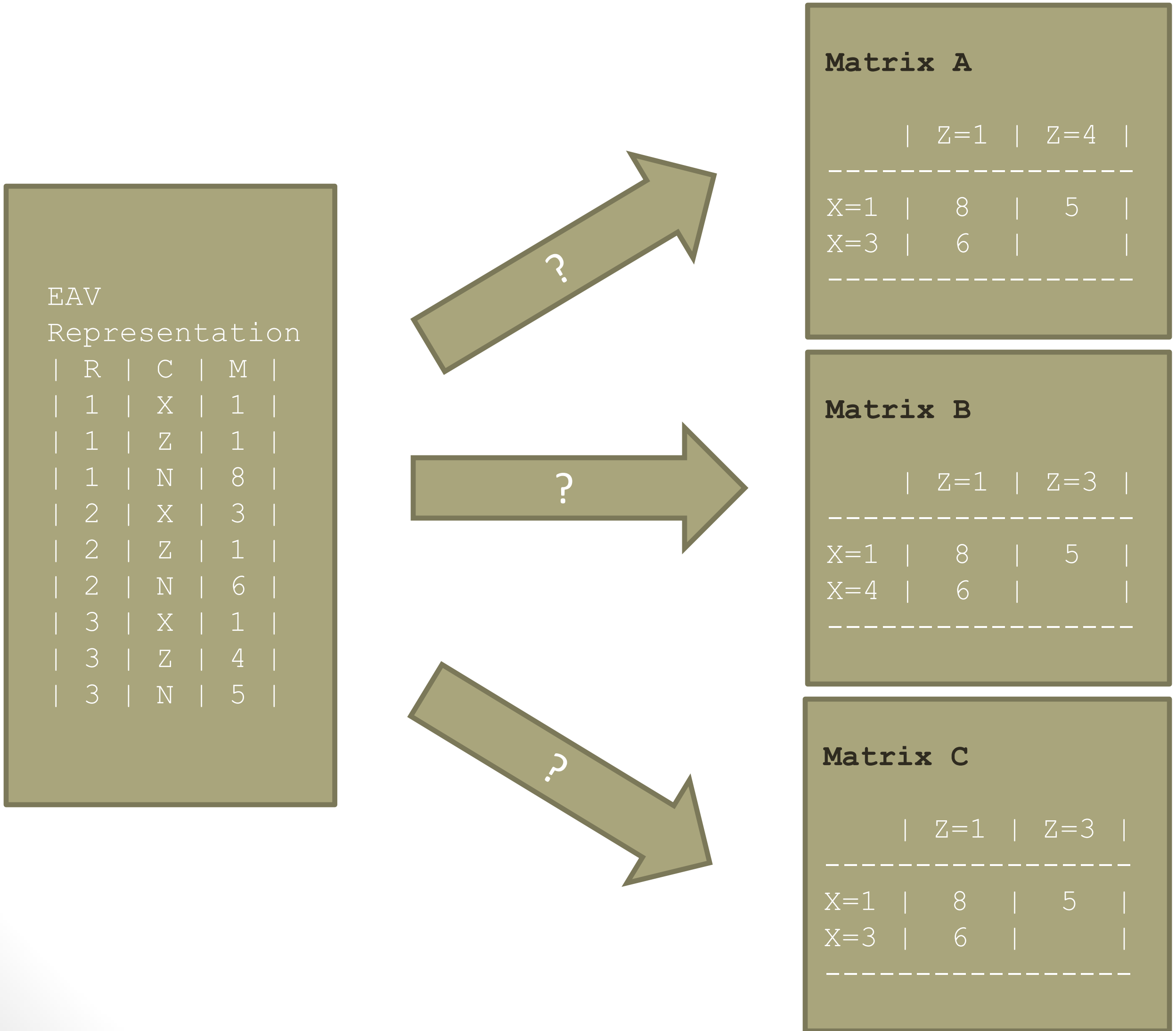
```
--Exercise 5;  Write SQL to multiply Matrix1 by 7
SELECT RowID, ColumnID, 7*Value AS Value FROM Matrix1
```

```
--Exercise 6;  Write SQL to transpose Matrix2
SELECT ColumnID AS RowID, RowID AS ColumnID, Value FROM
Matrix1
```

```
--Exercise 7;  Add two Matrices  (Add Matrix1 to Matrix3)
SELECT Matrix1.RowID AS RowID, Matrix1.ColumnID AS ColumnID,
(ISNULL(Matrix3.Value, 0) + ISNULL(Matrix1.Value, 0)) AS Value
   FROM Matrix1 FULL OUTER JOIN Matrix3 ON (Matrix1.ColumnID =
Matrix3.ColumnID) AND (Matrix1.RowID = Matrix3.RowID)
```

Sparse Matrices: Review (6)

- Exercise



EAV Representation

R	C	M
1	X	1
1	Z	1
1	N	8
2	X	3
2	Z	1
2	N	6
3	X	1
3	Z	4
3	N	5

?

?

?

Matrix A

	Z=1	Z=4
X=1	8	5
X=3	6	

Matrix B

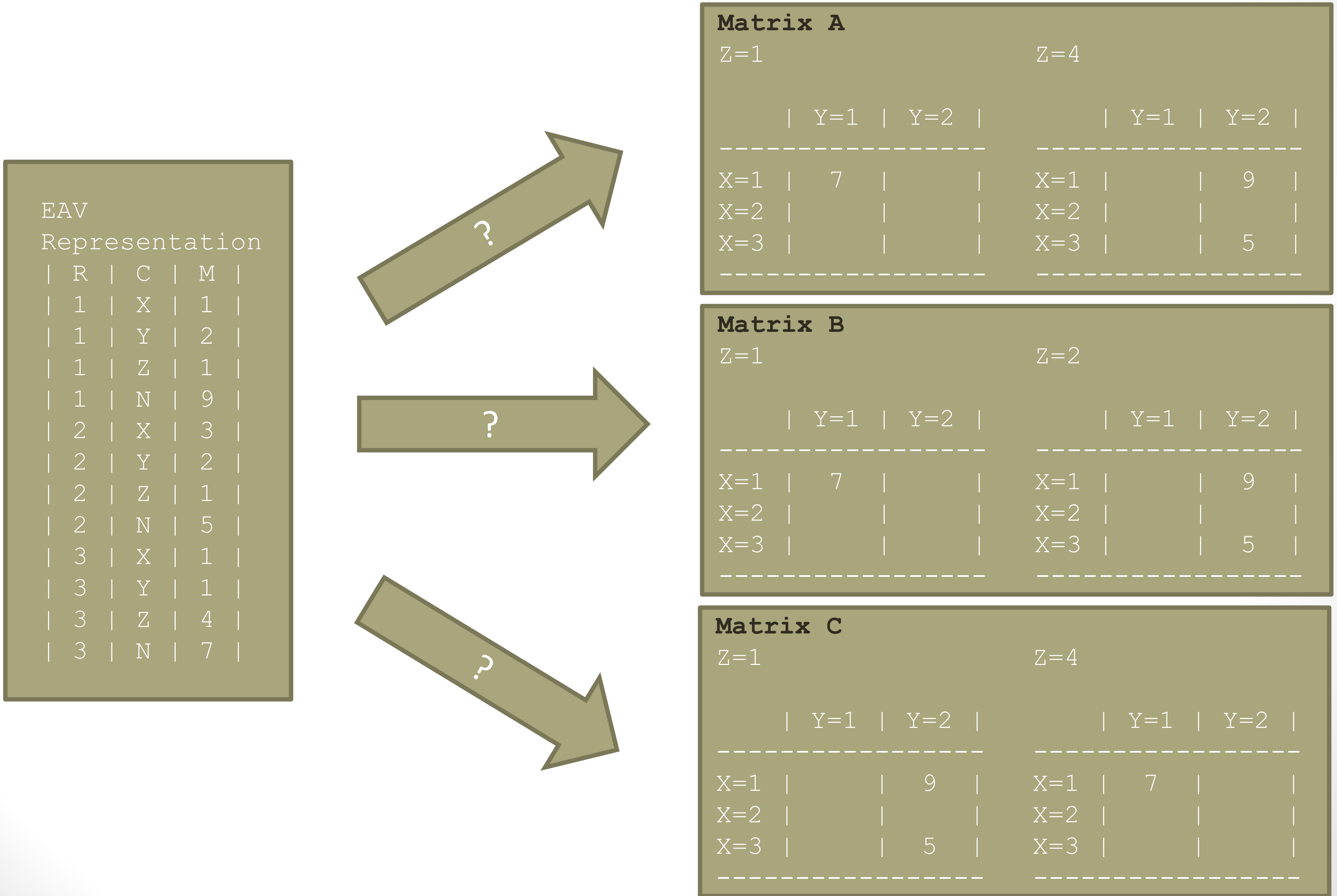
	Z=1	Z=3
X=1	8	5
X=4	6	

Matrix C

	Z=1	Z=3
X=1	8	5
X=3	6	

Sparse Matrices: Review (7)

- Exercise



```
EAV
Representation
| R | C | M |
| 1 | X | 1 |
| 1 | Y | 2 |
| 1 | Z | 1 |
| 1 | N | 9 |
| 2 | X | 3 |
| 2 | Y | 2 |
| 2 | Z | 1 |
| 2 | N | 5 |
| 3 | X | 1 |
| 3 | Y | 1 |
| 3 | Z | 4 |
| 3 | N | 7 |
```

```
Matrix A
Z=1                  Z=4

    | Y=1 | Y=2 |        | Y=1 | Y=2 |
-----------------    -----------------
X=1 |  7  |     |    X=1 |     |  9  |
X=2 |     |     |    X=2 |     |     |
X=3 |     |     |    X=3 |     |  5  |
-----------------    -----------------
```

```
Matrix B
Z=1                  Z=2

    | Y=1 | Y=2 |        | Y=1 | Y=2 |
-----------------    -----------------
X=1 |  7  |     |    X=1 |     |  9  |
X=2 |     |     |    X=2 |     |     |
X=3 |     |     |    X=3 |     |  5  |
-----------------    -----------------
```

```
Matrix C
Z=1                  Z=4

    | Y=1 | Y=2 |        | Y=1 | Y=2 |
-----------------    -----------------
X=1 |     |  9  |    X=1 |  7  |     |
X=2 |     |     |    X=2 |     |     |
X=3 |     |  5  |    X=3 |     |     |
-----------------    -----------------
```

Sparse Matrices and EAV Review

Quiz 09a

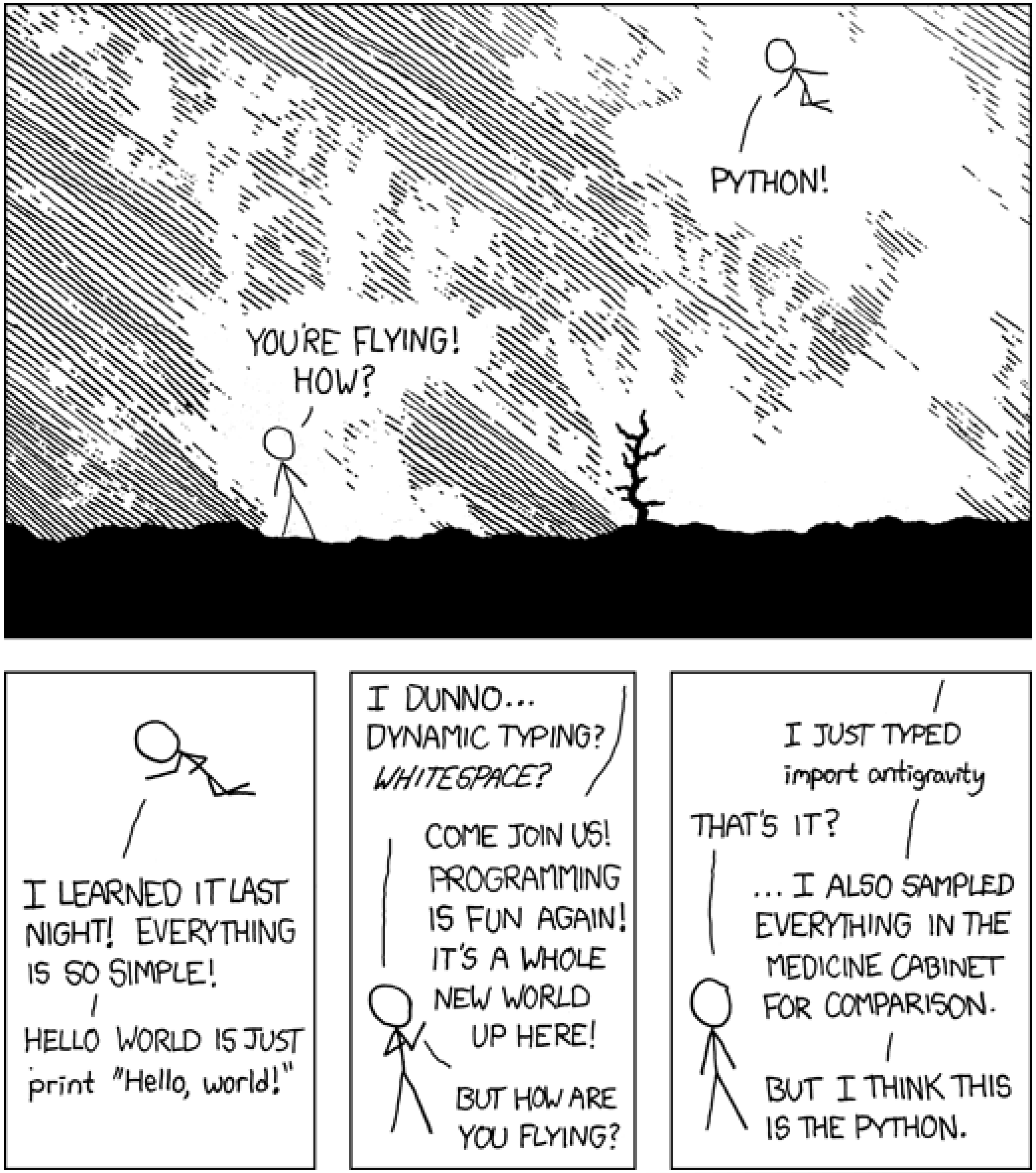
- EAV and Sparse Matrices
- https://catalyst.uw.edu/webq/survey/ernsthe/272210
- You need to view the projected slide to answer questions 1 and 2
- The tables in questions 6 and 7 are also depicted on these two slides:
  - Sparse Matrices: Review (6)
  - Sparse Matrices: Review (7)

Guest Lecture

A (brief) introduction to Python for Data Science

by Matt Danielson

Break


PYTHON!
YOU'RE FLYING! HOW?
I LEARNED IT LAST NIGHT! EVERYTHING IS SO SIMPLE!
HELLO WORLD IS JUST
print "Hello, world!"
I DUNNO... DYNAMIC TYPING? WHITESPACE?
COME JOIN US! PROGRAMMING IS FUN AGAIN! IT'S A WHOLE NEW WORLD UP HERE!
BUT HOW ARE YOU FLYING?
I JUST TYPED import antigravity
THAT'S IT?
... I ALSO SAMPLED EVERYTHING IN THE MEDICINE CABINET FOR COMPARISON.
BUT I THINK THIS IS THE PYTHON.

Hadoop

Hadoop Intro (1)

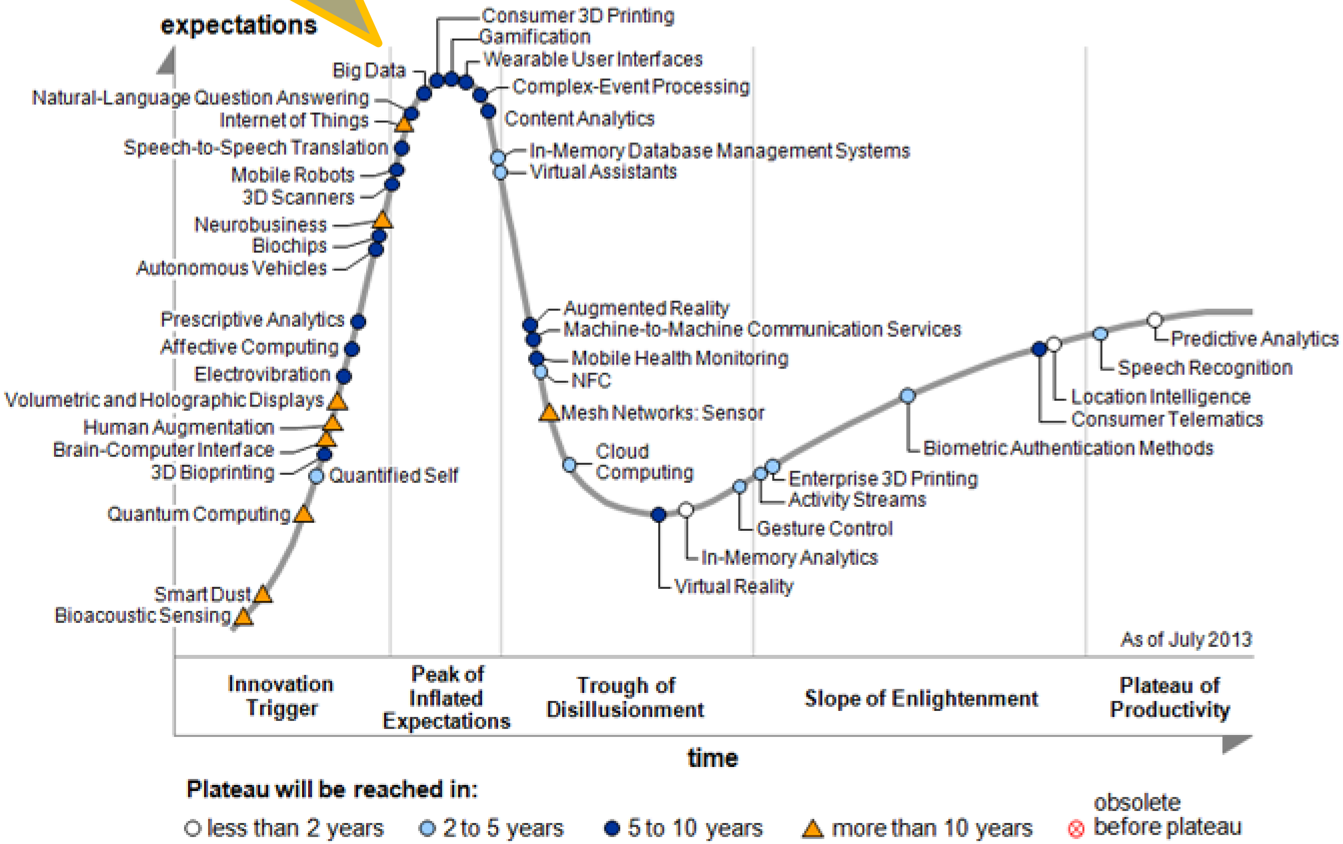
- Hadoop was created in 2008 at Yahoo by Doug Cutting and Mike Cafarella. Their work was based on two papers by Google:
  - **The Google File System**
  - **MapReduce: Simplied Data Processing on Large Clusters**

Hadoop Intro (2)

- Hadoop is
  - for big data
  - a framework for storing and processing large datasets (> 100 GB)
  - Apache open source software
  - inefficient and awkward for small datasets (< 10 GB)

Hadoop Intro (3)

Big Data on the BI Hype Curve



Hadoop Intro (4)

- Hadoop’s most important two components
  1. HDFS for storage (Hadoop Distributed File System)
  2. MapReduce for processing (Supported programming pattern for Hadoop tasks)
- Other required Hadoop components
  1. YARN (Resource management of MapReduce tasks)
  2. Common (Basic Hadoop libraries)

Hadoop Intro (5)

- Commodity Hardware (typical: 2 processors, 4 GB)
- Scalable (Vary number of nodes)
- Distributed processing to increase processing power
- Data replication to make storage more secure
- Avoid Data roundtrips
- Write Once Read Many and Update as rarely as possible! (http://en.wikipedia.org/wiki/Write_once_read_many)
- Streaming Access Pattern (Sequential Access of large chunks)
- Large block size (64 MB) enables fast sequential read. Data are streamed off the storage device while maintaining the maximum read rate. (Compare to common 4 KB block)

Hadoop Intro (6)

- Hadoop daemons:
  - Master Services:
    - NameNode: Manages HDFS
    - Secondary NameNode (Helps NameNode)
    - JobTracker: Manages TaskTrackers and MapReduce for a cluster
  - Slave Services:
    - DataNode: Executes MapReduce
    - TaskTracker: Manages MapReduce for a DataNode

LET'S SOLVE THIS PROBLEM BY USING THE BIG DATA NONE OF US HAVE THE SLIGHTEST IDEA WHAT TO DO WITH

Hadoop Intro (7)

- Ecosystem
  - Hive: Data warehouse on Hadoop
  - HiveQL SQL-like language for Hive
  - Impala: SQL-like query engine (Cloudera)
  - Sqoop: SQL to Hadoop. Transfers data between Hadoop and a relation database.
  - Oozie: Workflow scheduler for Hadoop. Scheduling is arranged like a DFD with DAG constraints
  - Flume: Collects and aggregates log files in Hadoop
  - Pig: Dataflow language and programming tool for MapReduce programs on Hadoop
  - Pig Latin: Language for Pig.
  - Mahout: Machine Learning algorithms that utilize the MapReduce pattern and run on Hadoop
  - Hue: Web-based interactive file browser

HDFS: How it works

- See: HowHDFSworksByManeeshVarshney.pdf
- The file is available in the resources website for Lecture 08.

Quiz 09b

- Quiz 09b Hadoop
- https://catalyst.uw.edu/webq/survey/ernsthe/272331
- You need to view the projected slide to answer questions 1 and 2

HDFS Lab

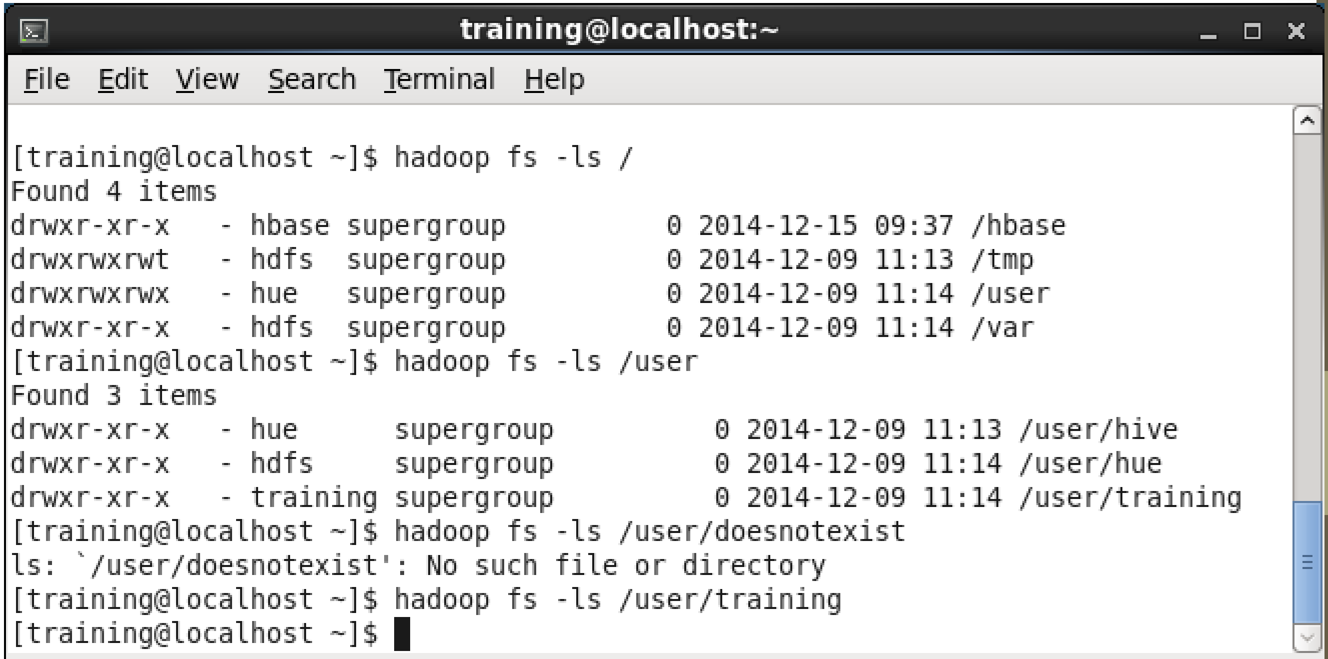
- Prerequisite: Lecture 05 Assignment item 4.

© 2014 Ted Goff www.tedgoff.com

"So you want to hire me as a Data Scientist for Intelligent Virtualized Deep Machine Learning Real-time Big Data in the Cloud for Social Networks? Ok, but if you also want Hadoop, increase my salary by 50%."

HDFS Lab (1)

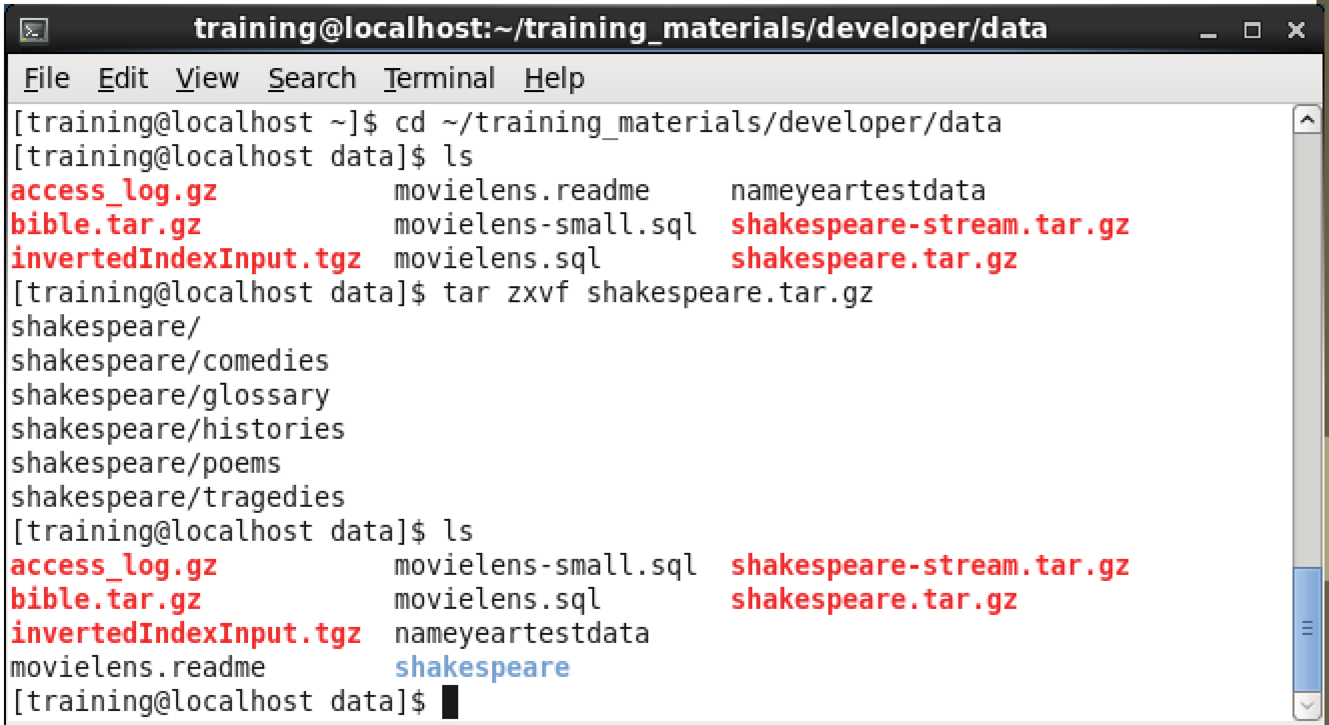
- Enter these commands into the console to list some folders and files inside HDFS:
  - $ **hadoop fs –ls /**
  - $ **hadoop fs –ls /user**
  - $ **hadoop fs –ls /user/doesnotexist**
  - $ **hadoop fs –ls /user/training**
- “fs” stands for file system (HDFS). “-ls” is an argument to list HDFS files.



```
[training@localhost ~]$ hadoop fs -ls /
Found 4 items
drwxr-xr-x   - hbase supergroup          0 2014-12-15 09:37 /hbase
drwxrwxrwt   - hdfs  supergroup          0 2014-12-09 11:13 /tmp
drwxrwxrwx   - hue   supergroup          0 2014-12-09 11:14 /user
drwxr-xr-x   - hdfs  supergroup          0 2014-12-09 11:14 /var
[training@localhost ~]$ hadoop fs -ls /user
Found 3 items
drwxr-xr-x   - hue      supergroup          0 2014-12-09 11:13 /user/hive
drwxr-xr-x   - hdfs     supergroup          0 2014-12-09 11:14 /user/hue
drwxr-xr-x   - training supergroup          0 2014-12-09 11:14 /user/training
[training@localhost ~]$ hadoop fs -ls /user/doesnotexist
ls: `/user/doesnotexist': No such file or directory
[training@localhost ~]$ hadoop fs -ls /user/training
[training@localhost ~]$ 
```

HDFS Lab (2)

- Enter these commands into the console to find and expand the shakespeare tar on the Linux OS :
  - $ **cd ~/training_materials/developer/data**
  - $ **ls**
  - $ **tar zxvf shakespeare.tar.gz**
  - $ **ls**

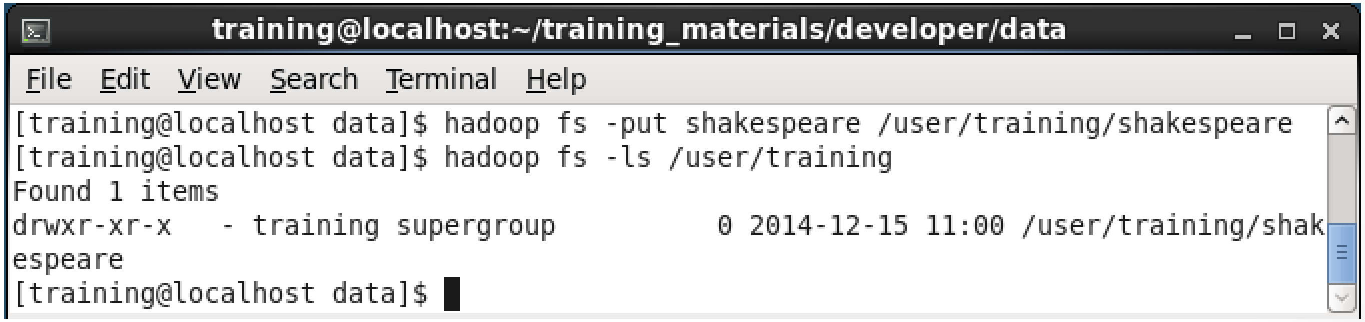


training@localhost:~/training_materials/developer/data
File Edit View Search Terminal Help

```
[training@localhost ~]$ cd ~/training_materials/developer/data
[training@localhost data]$ ls
access_log.gz            movielens.readme     nameyeartestdata
bible.tar.gz             movielens-small.sql  shakespeare-stream.tar.gz
invertedIndexInput.tgz   movielens.sql        shakespeare.tar.gz
[training@localhost data]$ tar zxvf shakespeare.tar.gz
shakespeare/
shakespeare/comedies
shakespeare/glossary
shakespeare/histories
shakespeare/poems
shakespeare/tragedies
[training@localhost data]$ ls
access_log.gz            movielens-small.sql  shakespeare-stream.tar.gz
bible.tar.gz             movielens.sql        shakespeare.tar.gz
invertedIndexInput.tgz   nameyeartestdata
movielens.readme         shakespeare
[training@localhost data]$ 
```

HDFS Lab (3)

- Enter these commands into the console to place the shakespeare directory into HDFS under training and then verify that the directory is in HDFS:
  - $ **hadoop fs -put shakespeare /user/training/shakespeare**
  - $ **hadoop fs -ls /user/training**

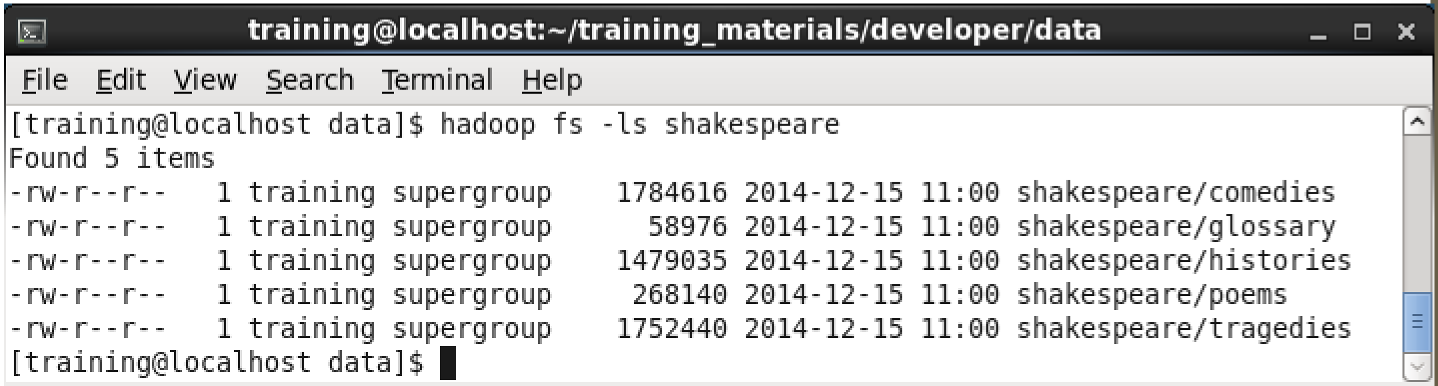

training@localhost:~/training_materials/developer/data
File Edit View Search Terminal Help

```
[training@localhost data]$ hadoop fs -put shakespeare /user/training/shakespeare
[training@localhost data]$ hadoop fs -ls /user/training
Found 1 items
drwxr-xr-x   - training supergroup          0 2014-12-15 11:00 /user/training/shak
espeare
[training@localhost data]$ 
```

For more HDFS fs commands see:
http://hadoop.apache.org/docs/r2.7.0/hadoop-project-dist/hadoop-common/FileSystemShell.html

HDFS Lab (4)

- Enter this command into the console to list the 5 files in the shakespeare directory of HDFS:
  - $ **hadoop fs -ls /user/training/shakespeare**

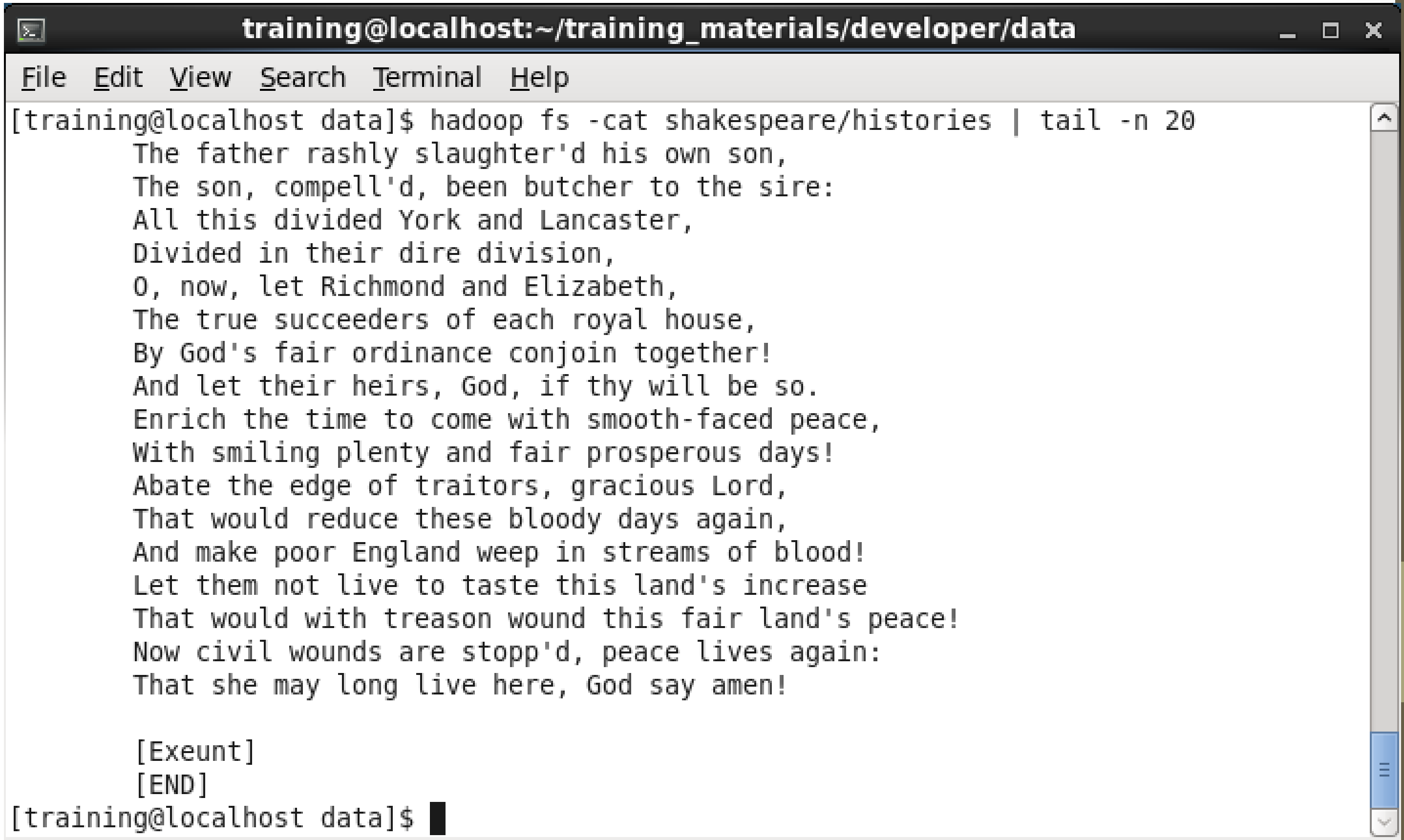


training@localhost:~/training_materials/developer/data
File Edit View Search Terminal Help

```
[training@localhost data]$ hadoop fs -ls shakespeare
Found 5 items
-rw-r--r--   1 training supergroup    1784616 2014-12-15 11:00 shakespeare/comedies
-rw-r--r--   1 training supergroup      58976 2014-12-15 11:00 shakespeare/glossary
-rw-r--r--   1 training supergroup    1479035 2014-12-15 11:00 shakespeare/histories
-rw-r--r--   1 training supergroup     268140 2014-12-15 11:00 shakespeare/poems
-rw-r--r--   1 training supergroup    1752440 2014-12-15 11:00 shakespeare/tragedies
[training@localhost data]$
```

HDFS Lab (5)

- Enter this command into the console to read the last 20 lines of the histories file in HDFS:
  - $ **hadoop fs -cat shakespeare/histories | tail -n 20**



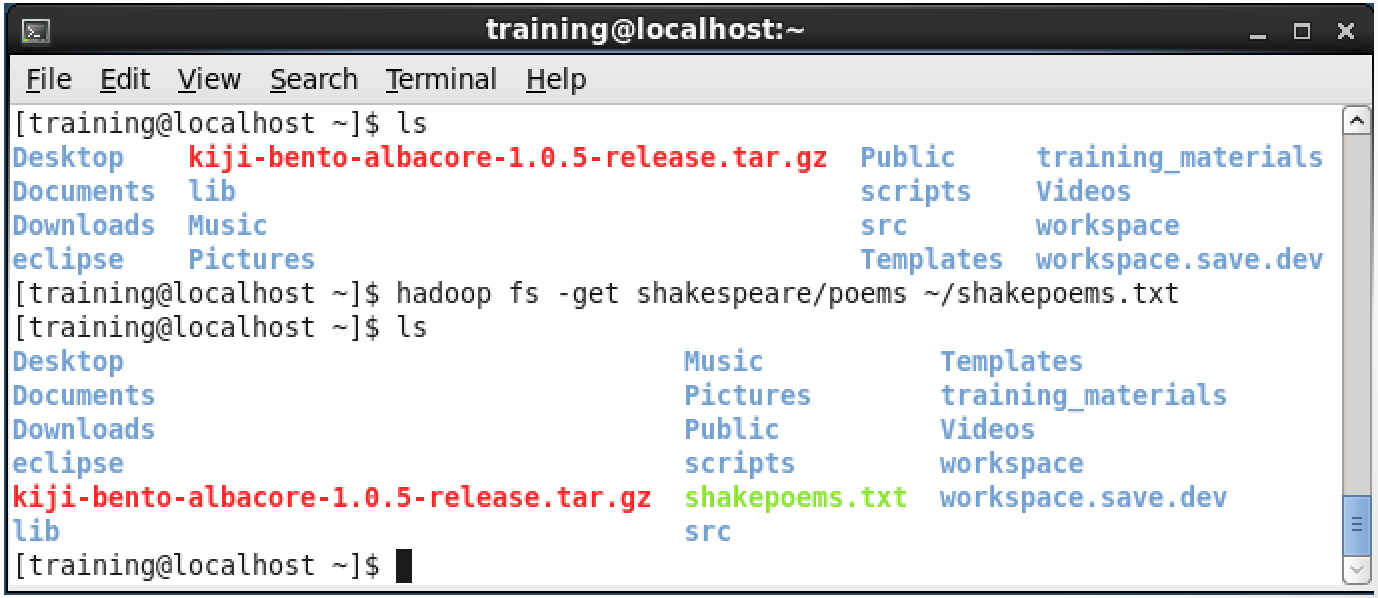
training@localhost:~/training_materials/developer/data
File Edit View Search Terminal Help

```
[training@localhost data]$ hadoop fs -cat shakespeare/histories | tail -n 20
        The father rashly slaughter'd his own son,
        The son, compell'd, been butcher to the sire:
        All this divided York and Lancaster,
        Divided in their dire division,
        O, now, let Richmond and Elizabeth,
        The true succeeders of each royal house,
        By God's fair ordinance conjoin together!
        And let their heirs, God, if thy will be so.
        Enrich the time to come with smooth-faced peace,
        With smiling plenty and fair prosperous days!
        Abate the edge of traitors, gracious Lord,
        That would reduce these bloody days again,
        And make poor England weep in streams of blood!
        Let them not live to taste this land's increase
        That would with treason wound this fair land's peace!
        Now civil wounds are stopp'd, peace lives again:
        That she may long live here, God say amen!

        [Exeunt]
        [END]
[training@localhost data]$
```

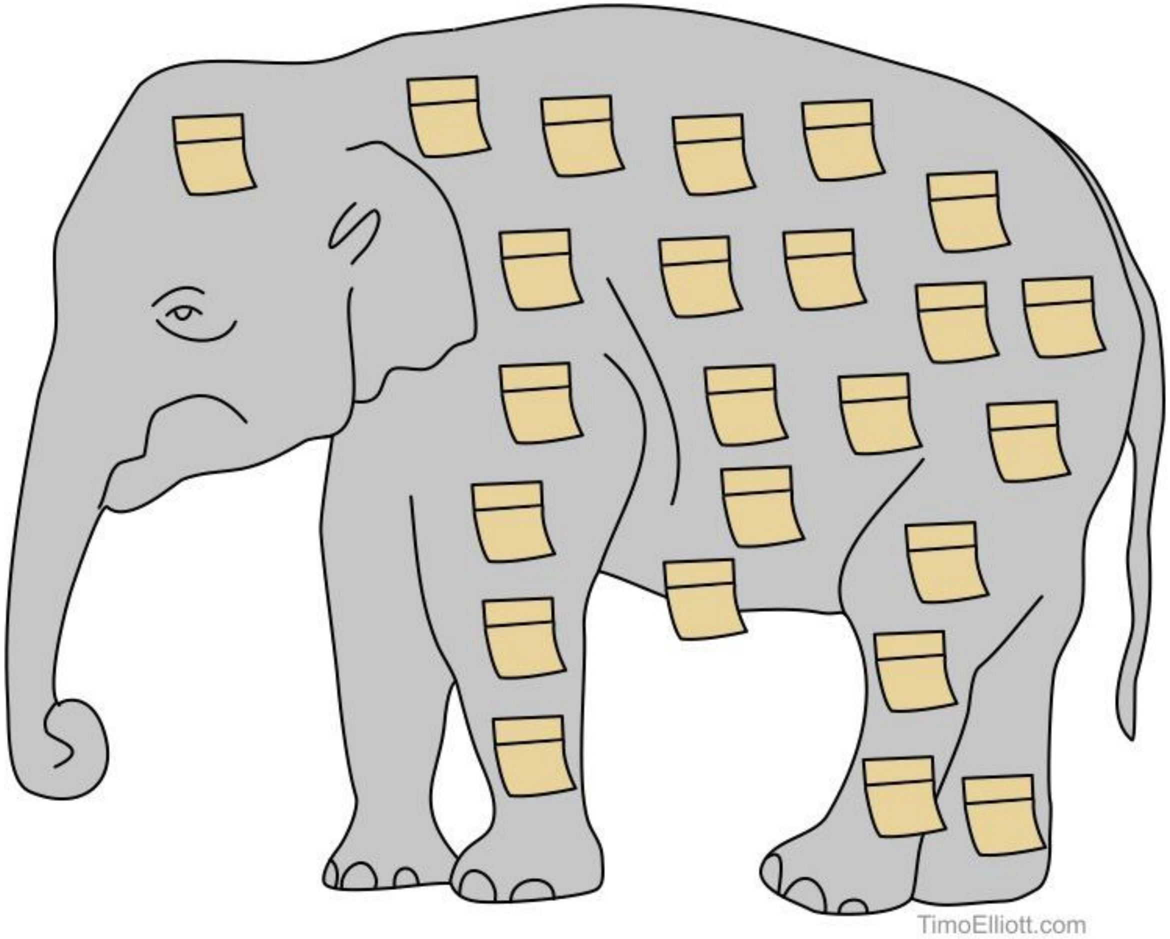
HDFS Lab (6)

- Close your terminal. Open a new terminal.
- Enter the following commands to retrieve files from HDFS and list them in the Linux OS:
  - `$ ls`
  - `$ hadoop fs -get shakespeare/poems ~/shakepoems.txt`
  - `$ ls`



```
[training@localhost ~]$ ls
Desktop    kiji-bento-albacore-1.0.5-release.tar.gz  Public     training_materials
Documents  lib                                       scripts    Videos
Downloads  Music                                     src        workspace
eclipse    Pictures                                  Templates  workspace.save.dev
[training@localhost ~]$ hadoop fs -get shakespeare/poems ~/shakepoems.txt
[training@localhost ~]$ ls
Desktop                                   Music           Templates
Documents                                 Pictures        training_materials
Downloads                                 Public          Videos
eclipse                                   scripts         workspace
kiji-bento-albacore-1.0.5-release.tar.gz  shakepoems.txt  workspace.save.dev
lib                                       src
[training@localhost ~]$ 
```

Break



An early Hadoop prototype...

Sqoop Lab

- **Sq**l + Had**oop** → **Sqoop**
- Use Sqoop to share between a RDBMS and Hadoop

"Finance here - we're not sure about this Hadoop thing... Could you just dump it all into Excel for us?"
TimoElliott.com

Sqoop Lab (1)

- Use this Sqoop command to familiarize yourself with Sqoop:

  -
    ```
    $ sqoop help
    ```

- Use these sqoop commands to list mysql databases on localhost and then tables in one of those databases:

  -
    ```
    $ sqoop list-databases --connect jdbc:mysql://localhost --username training --password training
    ```

  -
    ```
    $ sqoop list-tables --connect jdbc:mysql://localhost/movielens --username training --password training
    ```

Sqoop Lab (2)

- Use this Sqoop command to import data from a mysql table to an HDFS table. The fields of a table record (row) will be separated by tabs in the corresponding record of the HDFS file:

  -
    ```
    $ sqoop import --connect jdbc:mysql://localhost/movielens --username training --password training --fields-terminated-by '\t' --table movie
    ```

- Note that Sqoop constructs some SQL statements for MySQL and that it converts the SQL statement into a MapReduce job that only has a Map component:

  -
    ```
    Test SQL:  SELECT t.* FROM `movie` AS t LIMIT 1
    ```

  -
    ```
    SELECT MIN(`id`), MAX(`id`) FROM `movie`
    ```

- Use this HDFS command to list the newly imported file(s) and then view the last part of one of the files:

  -
    ```
    $ hadoop fs -ls movie
    ```

  -
    ```
    $ hadoop fs -tail movie/part-m-00000
    ```

To run this command a second time you will need to avoid a naming collision:

```
sqoop import --connect jdbc:mysql://localhost/movielens --username training --password training --fields-terminated-by '\t' --table movie --target-dir /movie2
```

Sqoop Lab (3)

- Use this Sqoop command to import data from a mysql table to an HDFS table. The fields of a table record (row) will be separated by tabs in the corresponding record of the HDFS file:

  -
    ```
    $ sqoop import --connect jdbc:mysql://localhost/movielens --username training --password training --fields-terminated-by '\t' --table movierating
    ```

- Use these HDFS commands to list the newly imported file(s) and then view the last part of one of the files:

  -
    ```
    $ hadoop fs -ls movierating
    ```

  -
    ```
    $ hadoop fs -tail movierating/part-m-00000
    ```

To run this command a second time you will need to avoid a naming collision:

sqoop import --connect jdbc:mysql://localhost/movielens --username training --password training --fields-terminated-by '\t' --table movierating --target-dir /movierating2

Hive Lab

TimoElliott.com

"It does look similar—but this one is powered by Hadoop"

Hive Lab (1)

- Check that we have movie and movierating in HDFS
  - $ `hadoop fs -cat movie/part-m-00000 | head`
  - $ `hadoop fs -cat movierating/part-m-00000 | head`

- Start Hive
  - $ `hive`

- Tell Hive that the HDFS files, movie and movierating, are tables:
  - **hive> CREATE EXTERNAL TABLE movie (id INT, name STRING, year INT) ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t' LOCATION '/user/training/movie';**
  - **hive> CREATE EXTERNAL TABLE movierating (userid INT, movieid INT, rating INT) ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t' LOCATION '/user/training/movierating';**

To run this command a second time you will need to avoid a naming collision:
hive> DROP TABLE **movie**
hive> DROP TABLE **movie**rating

Hive Lab (2)

- Ask Hive to show tables and provide metadata on these tables
  - `hive> SHOW TABLES;`
  - `hive> DESCRIBE movie;`
  - `hive> DESCRIBE movierating;`

- Use Hive to find information from your tables
  - **hive> SELECT * FROM movie WHERE year < 1925;**
  - **hive> SELECT * FROM movie WHERE year < 1925 AND year != 0 ORDER BY name;**
  - **hive> SELECT * FROM movierating WHERE userid=149;**

Hive Lab (3)

- hive> CREATE TABLE USERRATING (userid INT, numratings INT, avgrating FLOAT);
- hive> insert overwrite table userrating SELECT userid,COUNT(userid),AVG(rating) FROM movierating GROUP BY userid;

- hive> SELECT AVG(rating) FROM movierating WHERE userid=149;
- hive> SELECT userid, COUNT(userid),AVG(rating) FROM movierating where userid < 10 GROUP BY userid;

Hive Lab (4)

- Query with Join
  - hive> select movieid,rating,name from movierating join movie on movierating.movieid=movie.id where userid=149;
- hive> QUIT;

- Take-home Exercise
  - What is oldest movie in the database that has a top rating?

Hue and Impala

- Use Impala to execute “HiveQL”
- Open Firefox
- Start Hue by clicking on the Hue link in the favorites
- Select Impala and Query Tab
- Enter Query into Query text box for Impala:
  - **select movieid,rating,name from movierating join movie on movierating.movieid=movie.id where userid=149;**

MapReduce (0)

MapReduce (1)

- MapReduce is a pattern (programming model) designed to work in Hadoop
- MapReduce works with HDFS
- The two primary ideas behind MapReduce
  - Send the program to the data as opposed to the data to the program
  - Make use of distributed data where each chunk of data has its own CPUs
- When a MapReduce job is started, then Hadoop sends Map and Reduce jobs to the data nodes. Hadoop manages all the details of data passing among data nodes.
  - The Map portion of the computation occurs on the individual data nodes where the data reside.
  - The reduce portion of the computation occurs on some of the data nodes (not necessarily where the data reside). In some cases part of the reduce operation can occur on the data node where the data reside.

MapReduce (2)

- MapReduce uses three stages:
  - Map
  - Shuffle
  - Reduce
- Then, why don’t we call MapReduce: MapShuffleReduce?
  - Because: In most cases, the programmer need only be concerned with Map and Reduce. The Shuffle and Sorting is taken care of by Hadoop.
  - To achieve scalable programs, Hadoop takes care off:
    - Creating tasks on the various data nodes
    - Tracking Tasks
    - Moving data among data nodes
    - File I/O, networking, process synchronization, recovery from failures, re-running jobs, splitting input, Moving Key-Value-Pairs from Map to Reduce
    - Outputs results

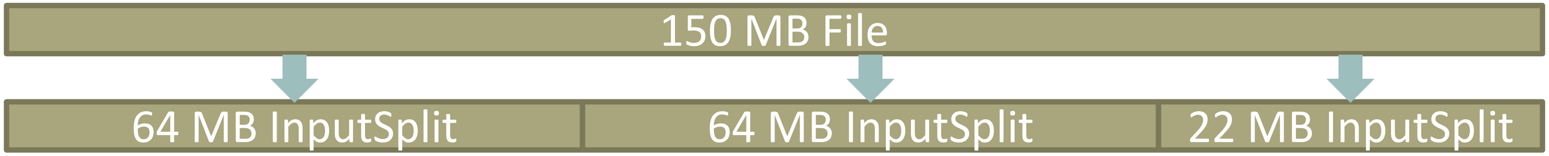
MapReduce (3)

- Parallelization Rules in MapReduce
  - Map tasks occur in parallel independent of each other
  - Reducers can start before Mappers are done. But, reducers cannot complete before Mapping and Shuffle & Sort have completed.
  - Reduce tasks occur in parallel independent of each other

MapReduce (4)

150 MB File

MapReduce (5)



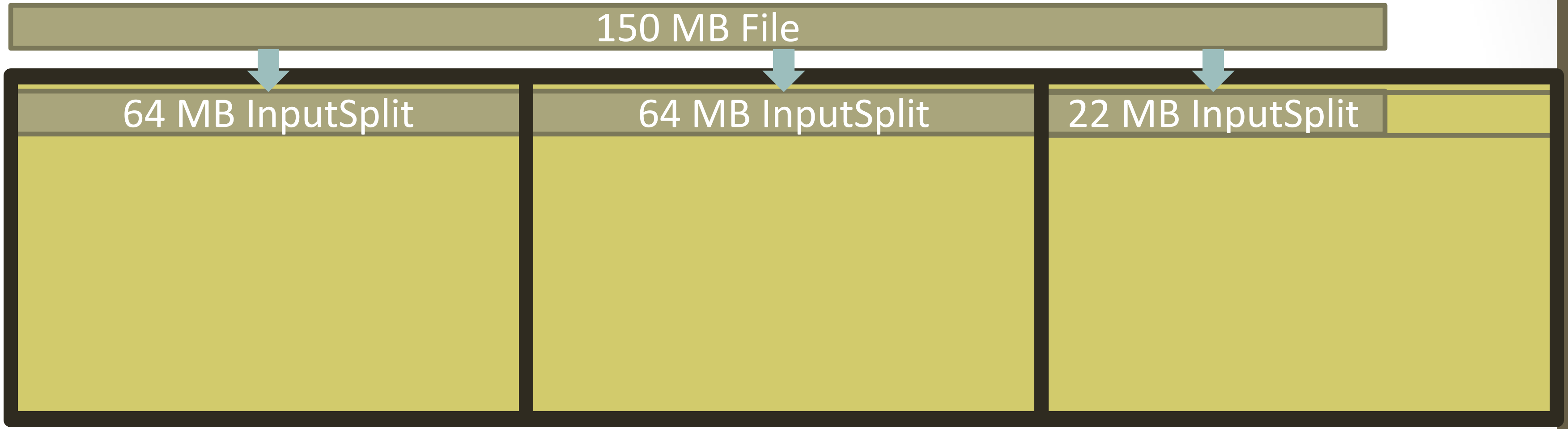
150 MB File

64 MB InputSplit

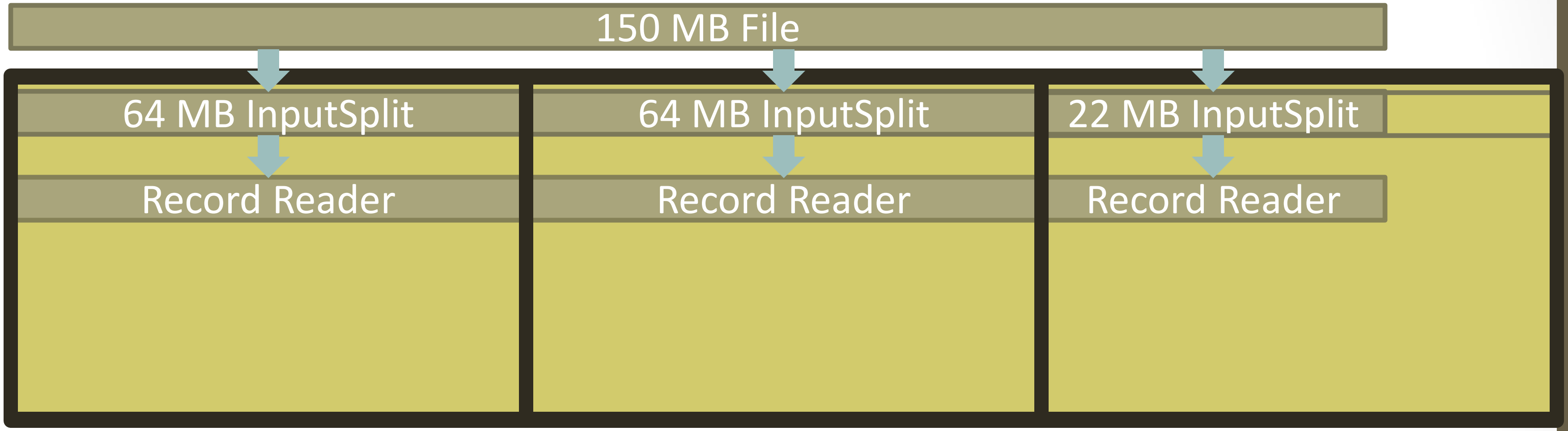
64 MB InputSplit

22 MB InputSplit

MapReduce (6)


150 MB File
64 MB InputSplit
64 MB InputSplit
22 MB InputSplit

MapReduce (7)



150 MB File

64 MB InputSplit

Record Reader

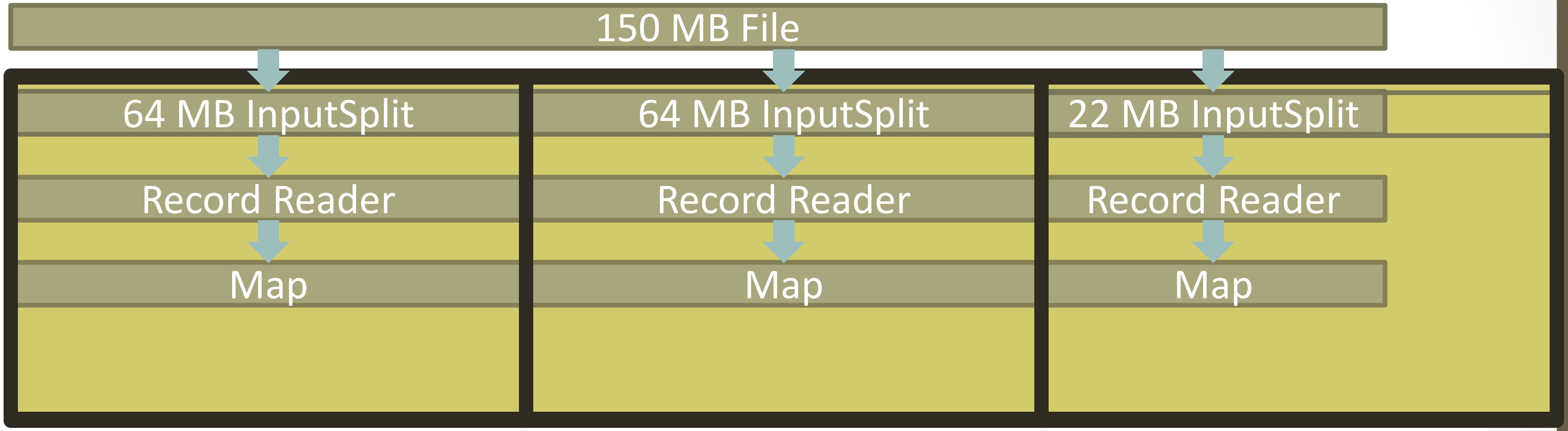
64 MB InputSplit

Record Reader

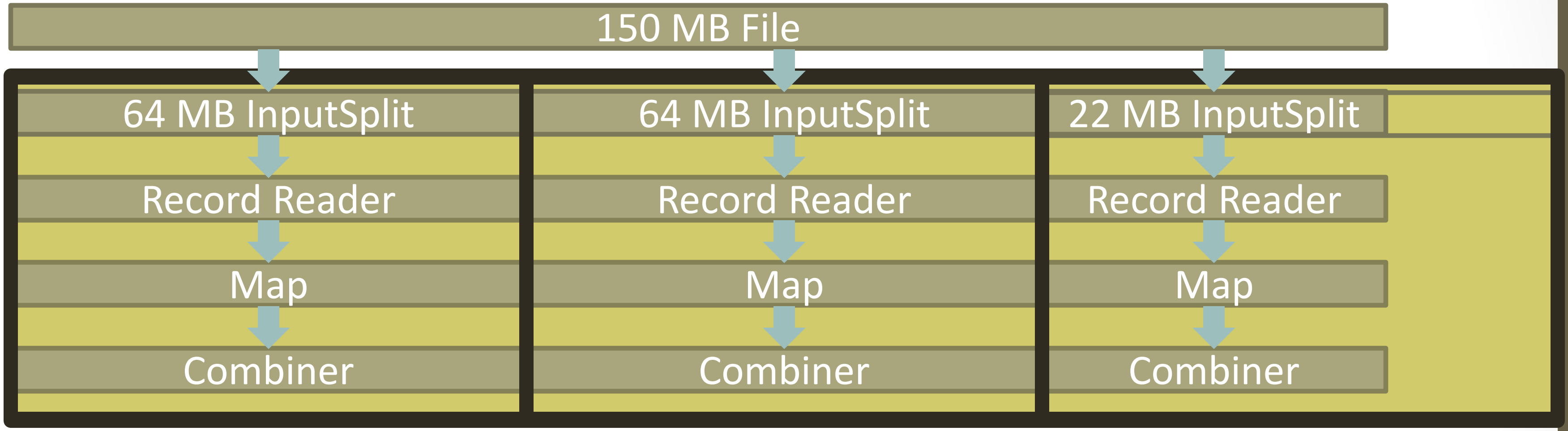
22 MB InputSplit

Record Reader

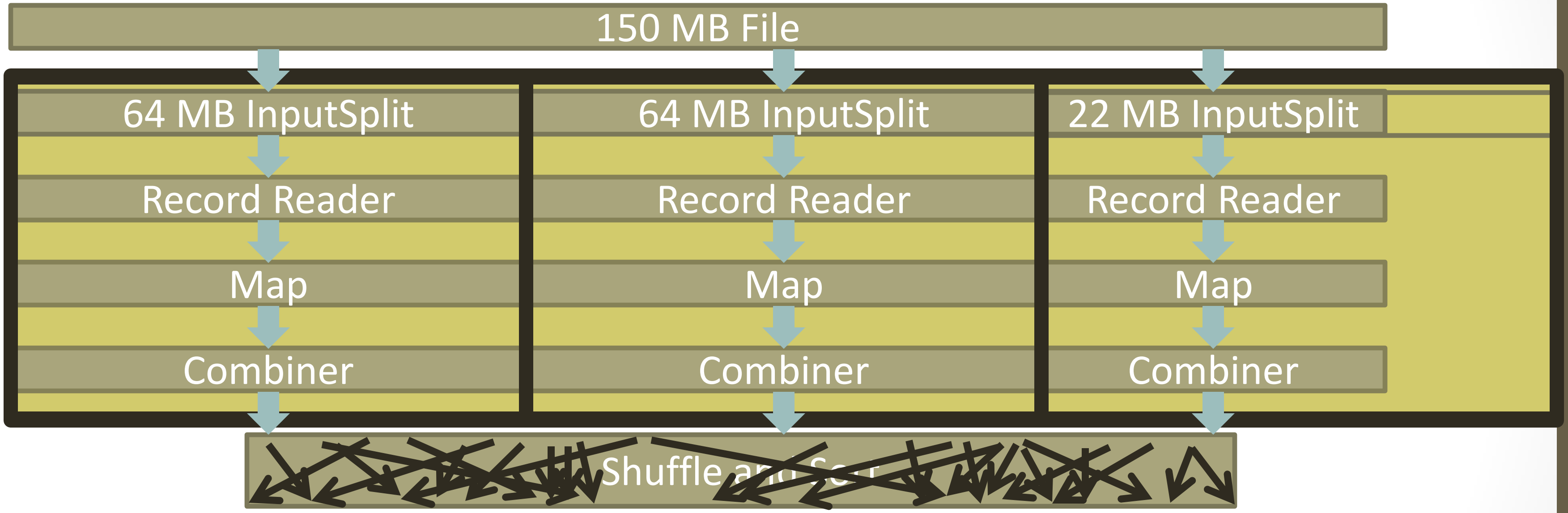
MapReduce (8)


150 MB File
64 MB InputSplit
Record Reader
Map
64 MB InputSplit
Record Reader
Map
22 MB InputSplit
Record Reader
Map

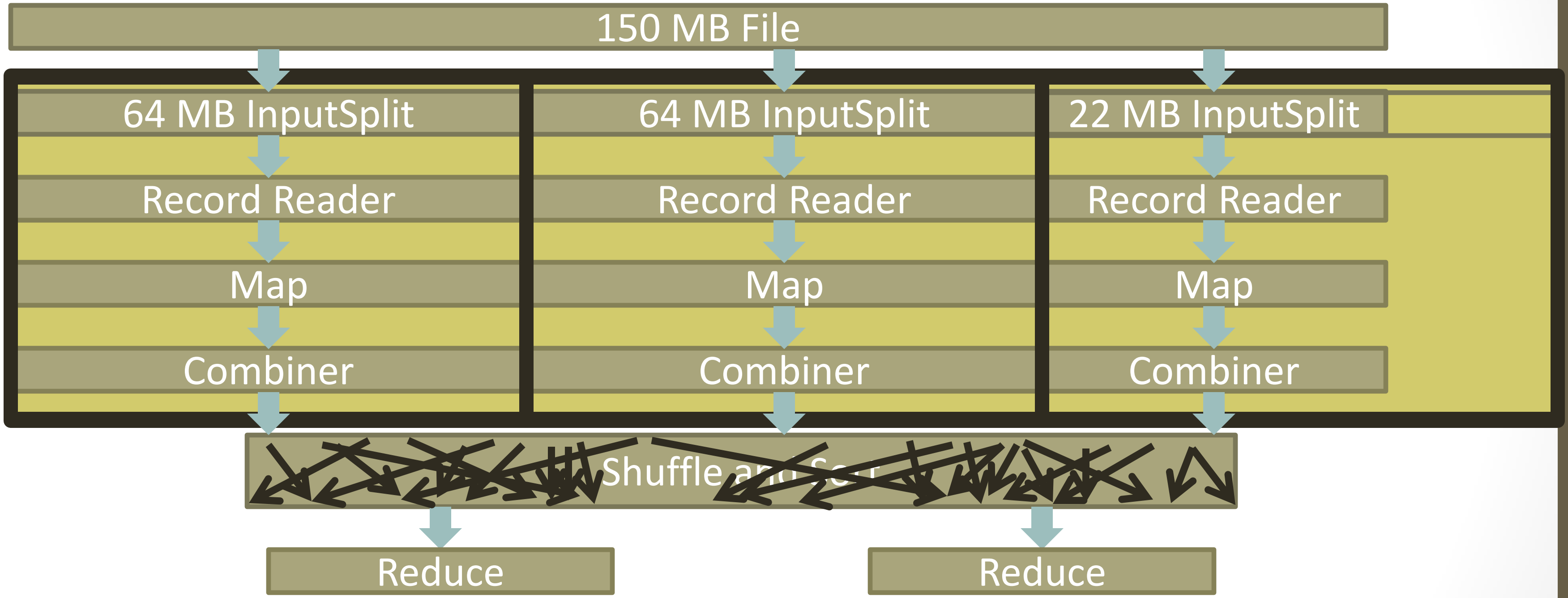
MapReduce (9)


150 MB File
64 MB InputSplit
Record Reader
Map
Combiner
64 MB InputSplit
Record Reader
Map
Combiner
22 MB InputSplit
Record Reader
Map
Combiner

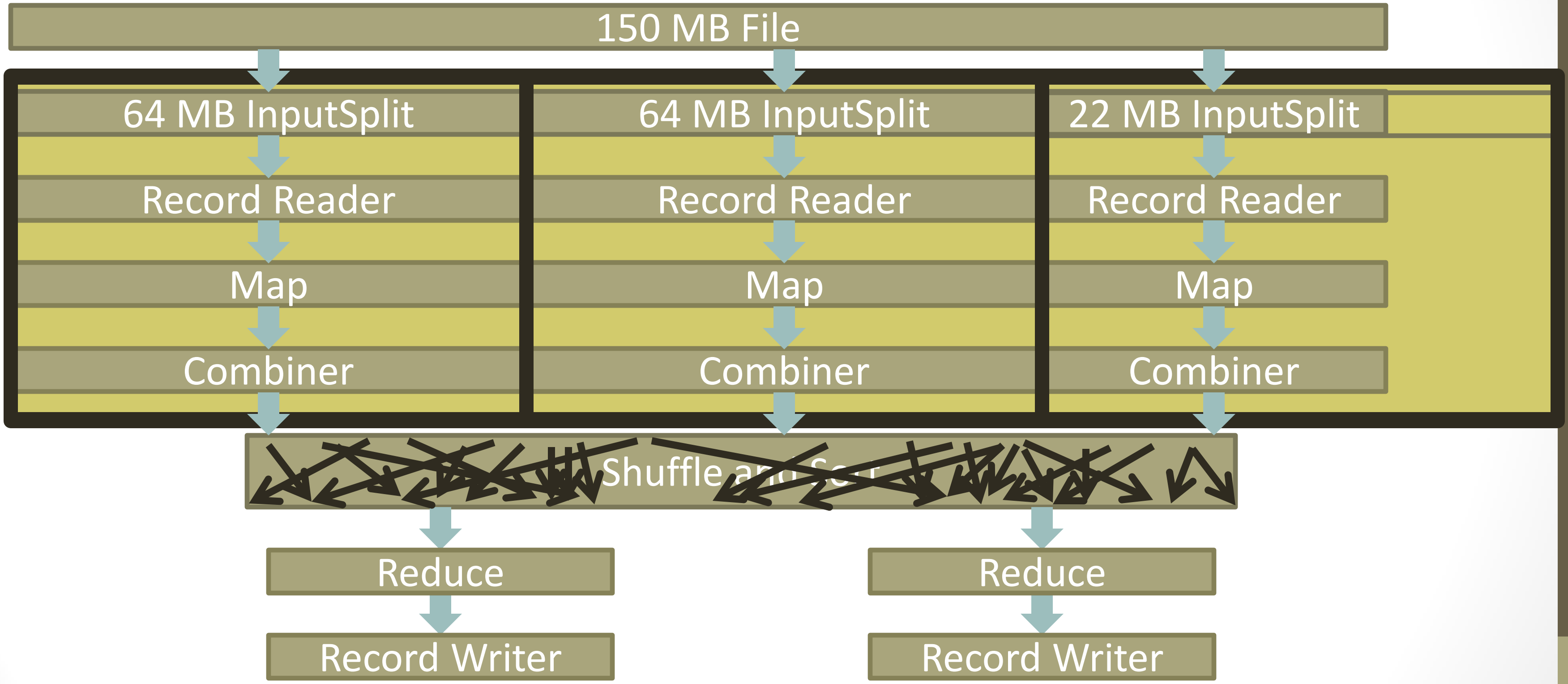
MapReduce (10)



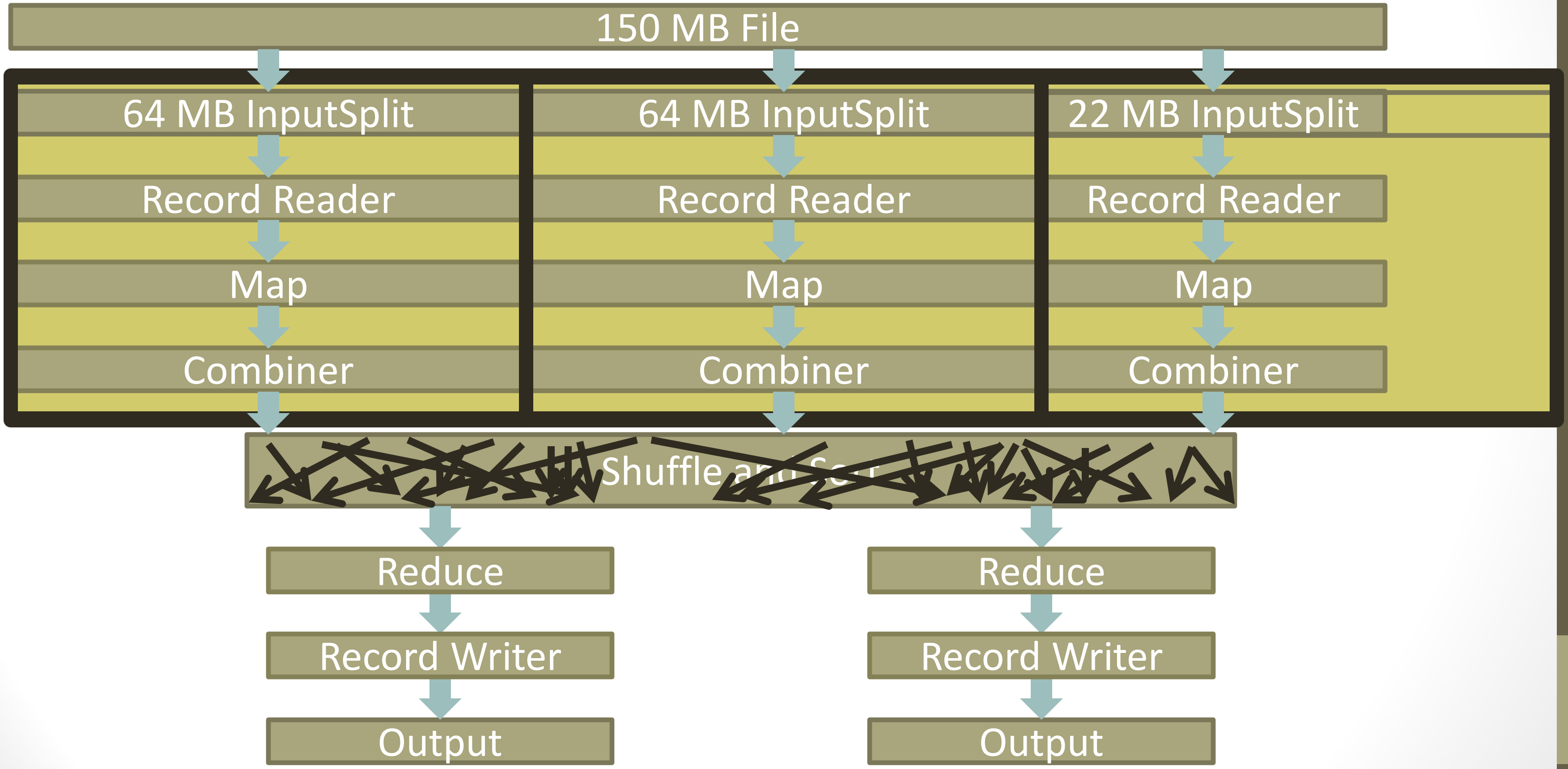
MapReduce (11)



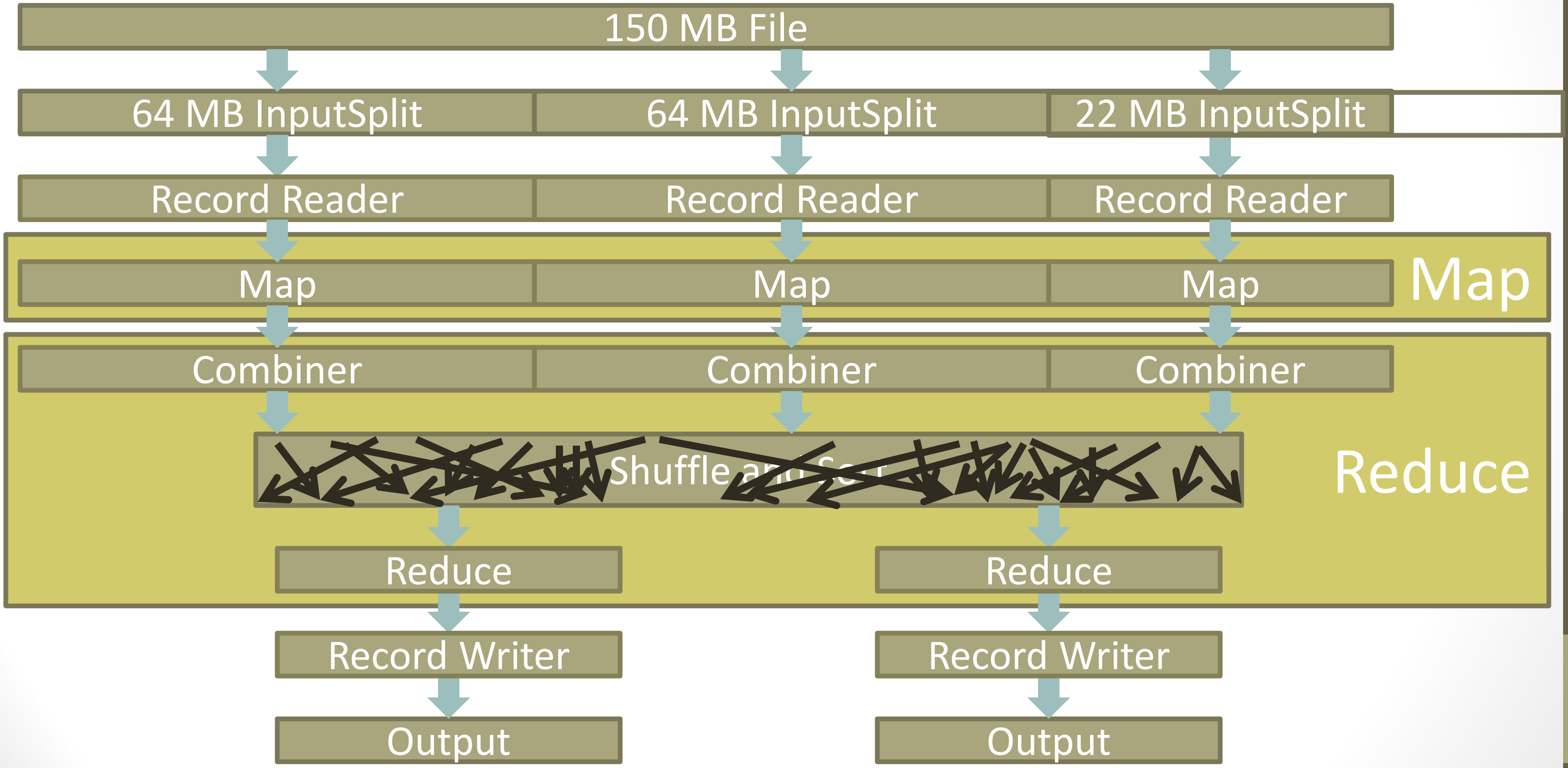
MapReduce (12)



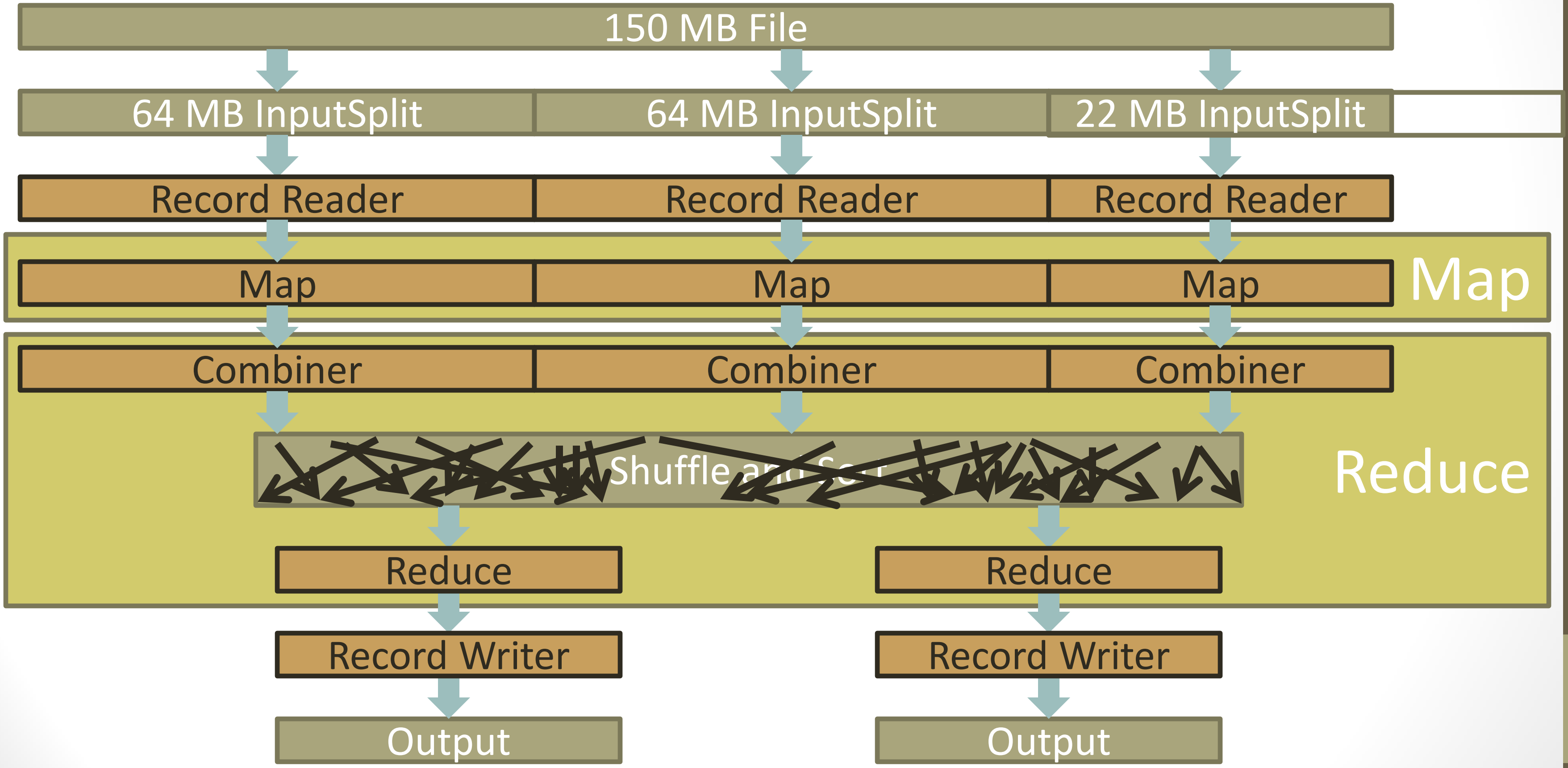
MapReduce (13)



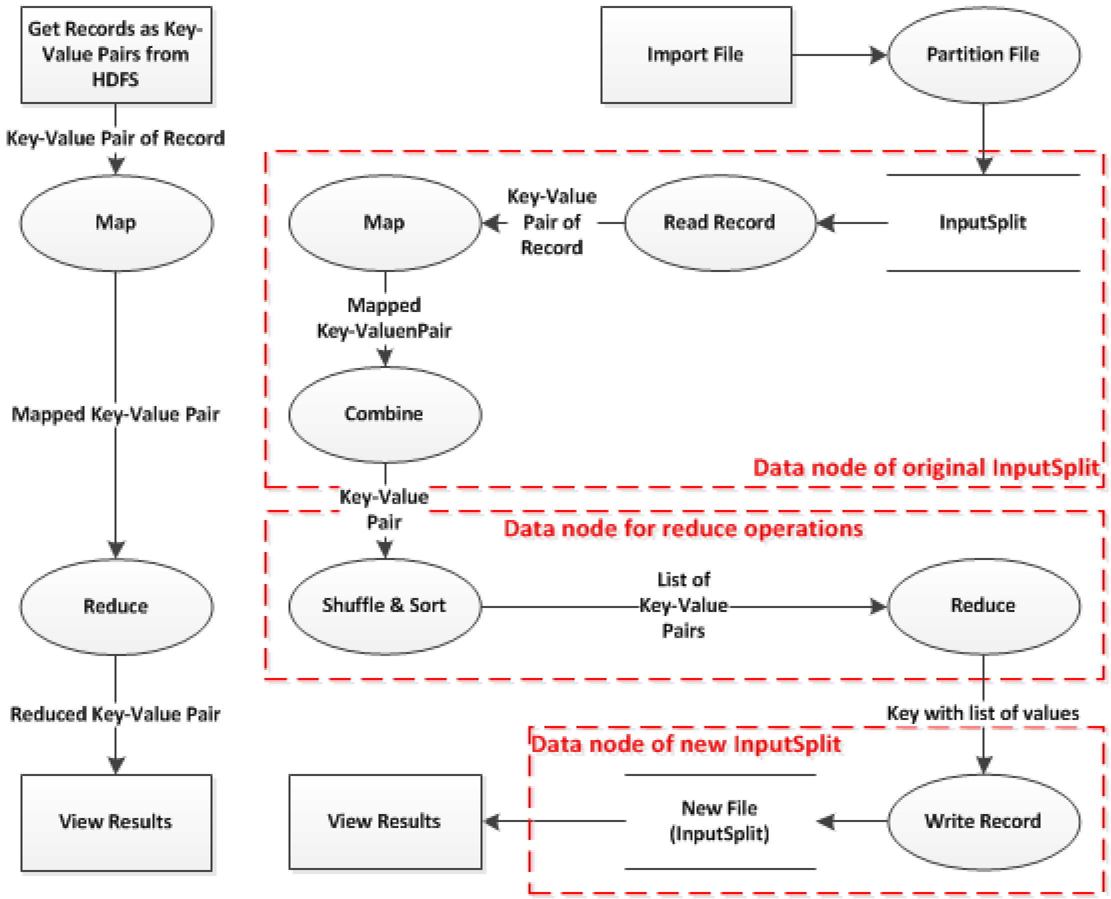
MapReduce(14)



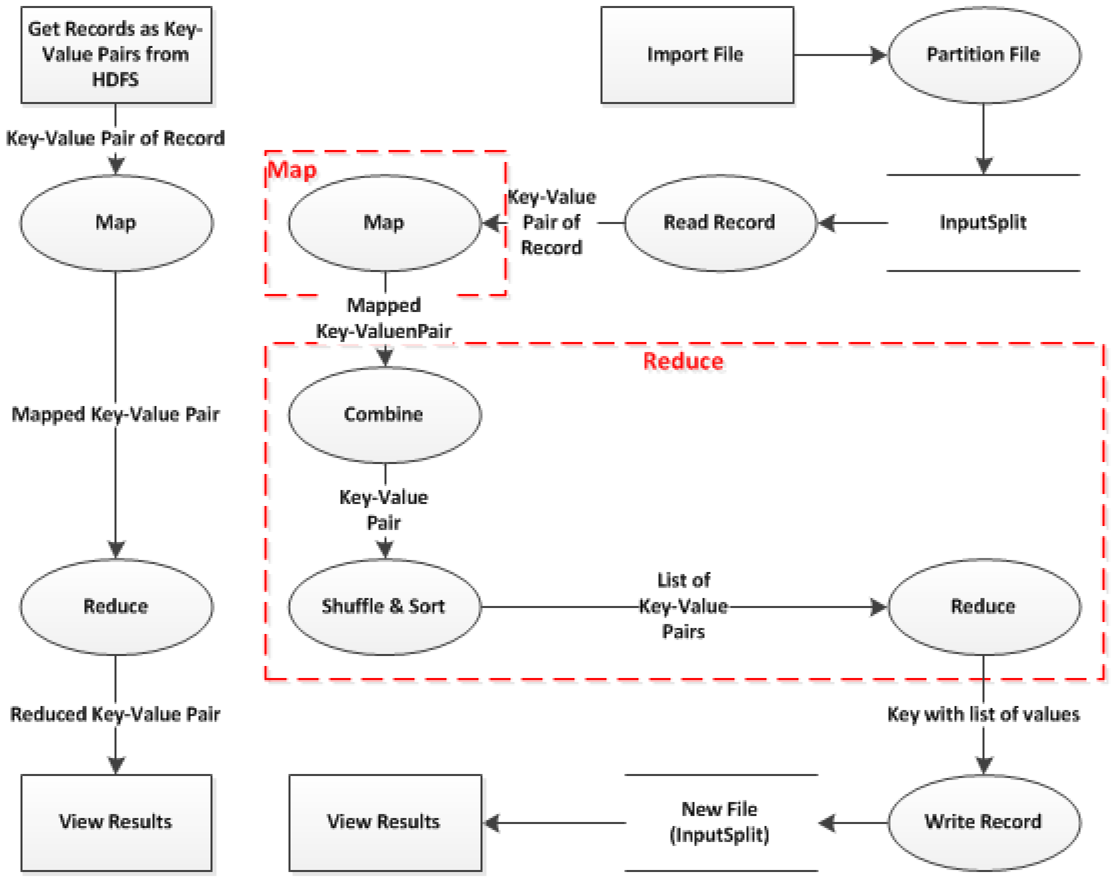
MapReduce (15)



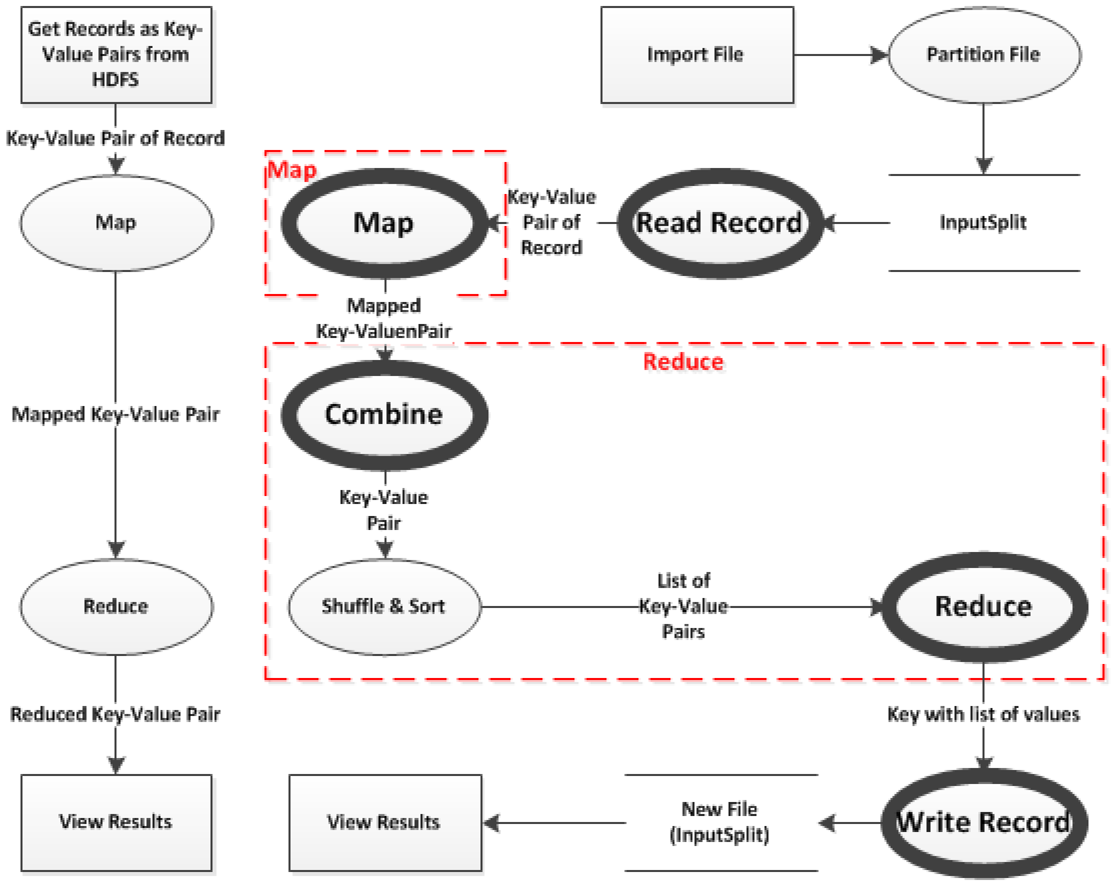
MapReduce (16)



MapReduce (17)



MapReduce (18)



MapReduce (19)

- Record Read
  - Record → (K1,V1)
  - to be or not to be → (0,“to be or not to be”)
- Map
  - (K1,V1) → list(K2,V2)
  - (0,“to be or not to be”) → [(to,1),(be,1),(or,1),(not,1),(to,1),(be,1)]
- Shuffle and Sort
  - list(K2,V2) → (K2,list(V2))
  - [(to,1),(be,1),(or,1),(not,1),(to,1),(be,1)] → (to,[1,1]),(be,[1,1]),(or,[1]),(not,[1])
- Reduce
  - (K2,list(V2)) → list(K3,V3)
  - (to,[1,1]),(be,[1,1]),(or,[1]),(not,[1]) → [(to,2),(be,2),(or,1),(not,1)]
- Record Write
  - (K3,V3) → Records
  - [(to,2),(be,2),(or,1),(not,1)] →

to,2
be,2
or,1
not,1

MapReduce Lab (1)

- Open Eclipse by clicking on the and navigate to WordCount to see the java files that are part of :
  - wordcount\src\solution\
    - WordCount.java
    - WordMapper.java
    - SumReducer.java

- Study these files

- In the terminal:

```
$ cd ~/workspace/wordcount/src
$ ls
$ ls solution
$ hadoop classpath
$ javac -classpath `hadoop classpath` solution/*.java
```

Note that the apostrophe in this statement is the not the straight or slanted quote (' or ‘) but rather this character (`):    javac -classpath `hadoop classpath` solution/*.java

MapReduce Lab (2)

- In the terminal:

```
$ jar cvf wc.jar solution/*.class
$ hadoop jar wc.jar solution.WordCount shakespeare wordcounts
$ hadoop fs -ls wordcounts
$ hadoop fs -cat wordcounts/part-r-00000 | less
$ hadoop jar wc.jar solution.WordCount shakespeare/poems pwords
$ hadoop fs -rm -r wordcounts pwords
```

MapReduce Lab (3)

- Open Firefox
- Start Hue
- Start File Browser in Hue
  - Find wordcounts/part-r-00000
  - View Contents of part-r-00000

C'mon, Hadoop — it's bedtime!

Hadoop

Assignment

- Preview the SPARQL Lecture and the Graph Data Lecture (previews were posted in Resource site under Lecture 7)
- Look through the quizzes for Lecture 10. They are posted in the preview section of Lecture 09.
- There is nothing to submit!

Introduction to Data Science