



Big Data Programming Using Hadoop Workshop

February 2015

Dr.Thanachart Numnonda
IMC Institute
thanachart@imcinstitute.com

Modify from Original Version by Danairat T.
Certified Java Programmer, TOGAF – Silver
danairat@gmail.com

Hands-On: Running Hadoop

Running this lab using Cloudera Live



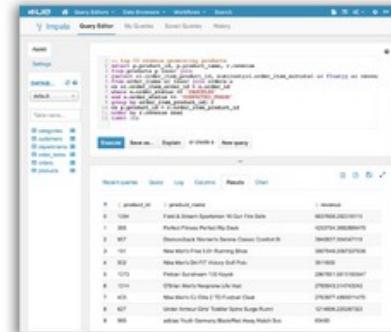
PRODUCTS & SERVICES TRAINING SOLUTIONS CUSTOMERS PARTNERS RESOURCES BLOGS

Cloudera Live

Try Apache Hadoop Now

Cloudera Live is the fastest and easiest way to get started with Apache Hadoop and it now includes self-guided, interactive demos and tutorials. With a one-button deployment option, you can spin up a four-node cluster of CDH, Cloudera's open source Hadoop platform, within minutes. This free, cloud-based Hadoop environment lets you:

- Learn the basics of Hadoop (and CDH) through pre-loaded, hands-on tutorials
- Plan your Hadoop project using your own datasets
- Explore the latest features in CDH
- Extend the capabilities of Hadoop and CDH through familiar partner tools, including Tableau and Zoomdata



Rank	product_id	product_name	category	brand
1	1000	Ford A Crown Spectre W/Our Fire Side	REDFIRE	REDFIRE
2	1001	BMW M3 E92 2010	BMW	BMW
3	1007	Diamondback Women's Reserve Classic Commute B	DIAMONDBK	DIAMONDBK
4	101	New Balance Fresh 2.0 Running Shoe	NEWBALANCE	NEWBALANCE
5	102	New Balance Drift Fit Walk/Golf Polo	NEWBALANCE	NEWBALANCE
6	1023	Pinkie Squeezies 100% Wool	PINKIESQUEEZIES	PINKIESQUEEZIES
7	1024	New Balance Men's Fresh 2.0 Running Trail	NEWBALANCE	NEWBALANCE
8	1025	New Balance Men's Fresh 2.0 FuelFit Cross	NEWBALANCE	NEWBALANCE
9	1027	Under Armour Girls' Rival Stripe Surge Runn	UNDERARMOUR	UNDERARMOUR
10	1030	adidas Youth Germany Black/Red Away Match Bus	ADIDAS	ADIDAS

Cloudera Live FAQ

Read-Only Demo

Quickstart VM

cloudera® LIVE

Using Cloudera VM

Contact Sales: 866-843-7207

Search

cloudera PRODUCTS & SERVICES TRAINING SOLUTIONS CUSTOMERS PARTNERS RESOURCES BLOGS

Downloads > QuickStart VMs | Cloudera Manager | CDH | Connectors | CDH4 Components

QuickStart VMs for CDH 5.2.x

A Single-Node Hadoop Cluster and Examples for Easy Learning!

Start testing Hadoop with Cloudera's QuickStart VMs. The QuickStart VMs contain a single-node Apache Hadoop cluster, complete with example data, queries, scripts, and Cloudera Manager to manage your cluster.

The VMs run CentOS 6.2 and are available for VMware, VirtualBox, and KVM.

All require a 64-bit host OS.

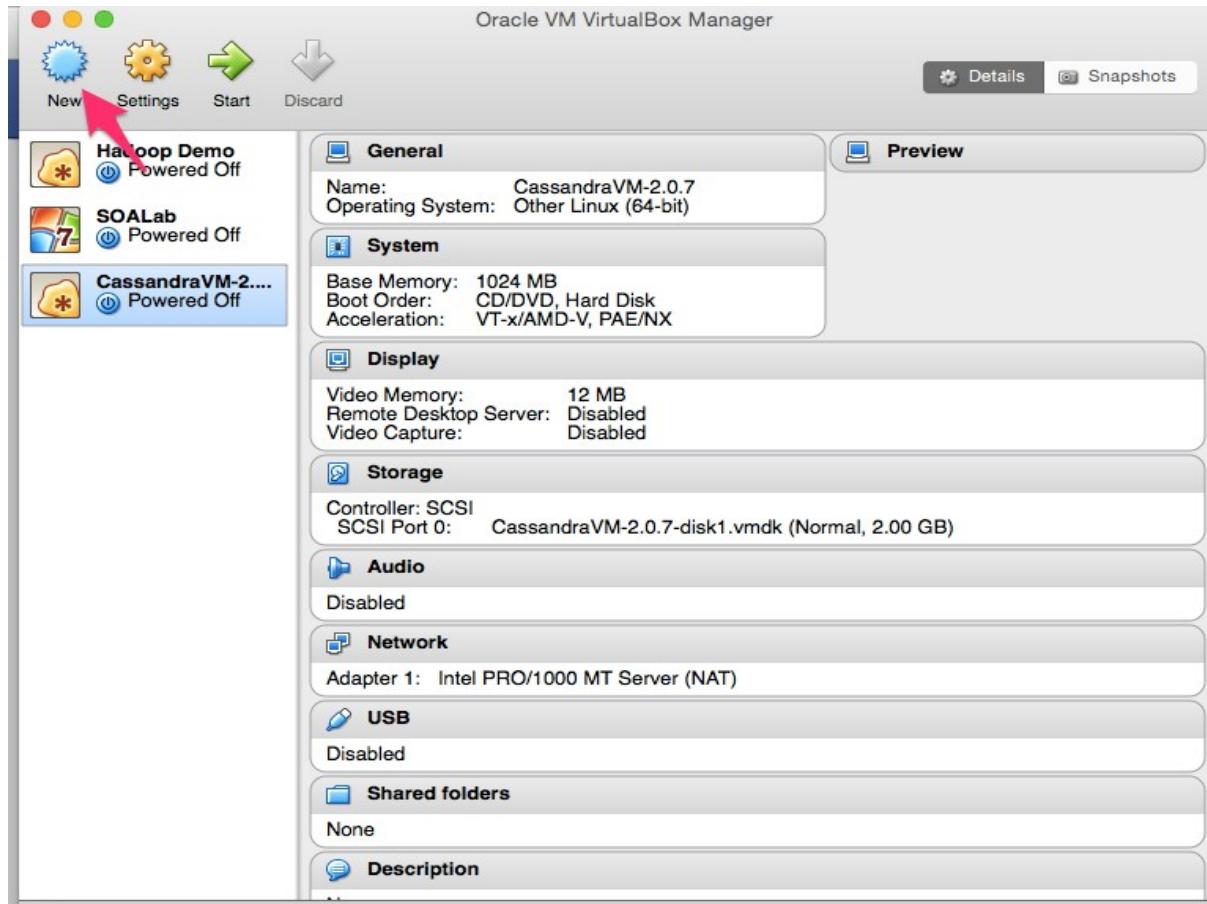
Version:

Please Note: Cloudera QuickStart VMs are for demo purposes only and are not to be used as a starting point for clusters.



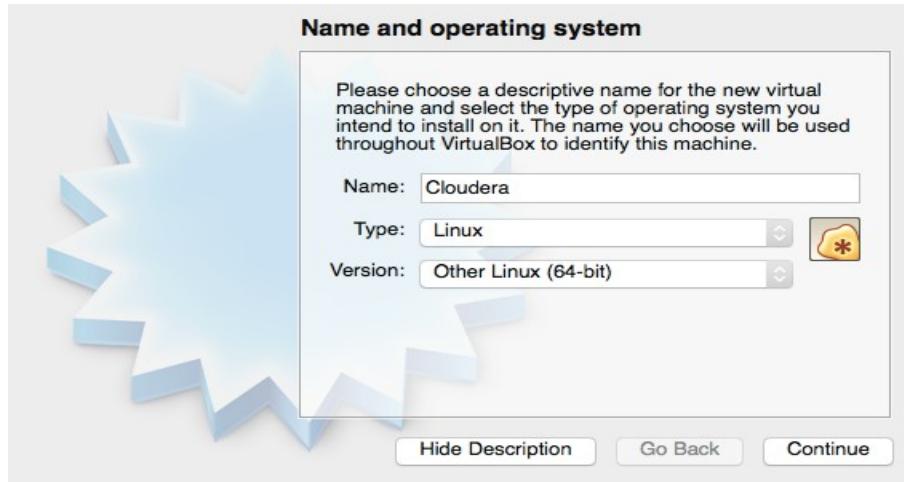
Starting Cloudera VM

Start VirtualBox and Select New

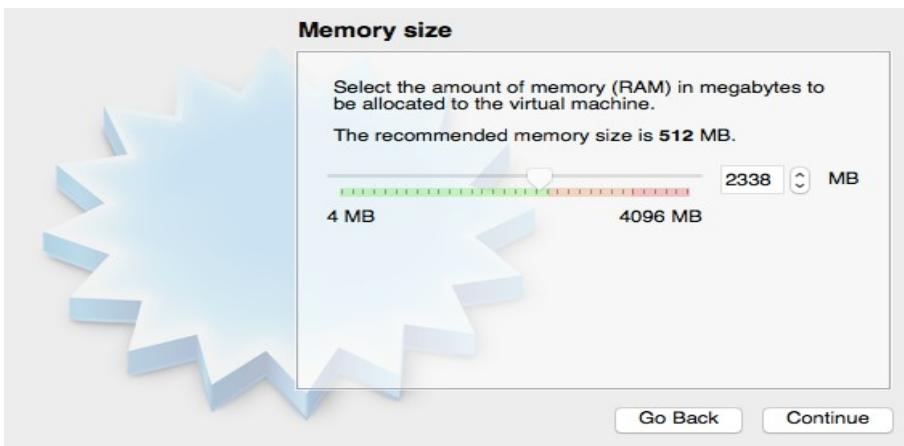


Starting Cloudera VM (cont)

Name the image as Cloudera and select OS as Linux 64 bit



Then select memory size

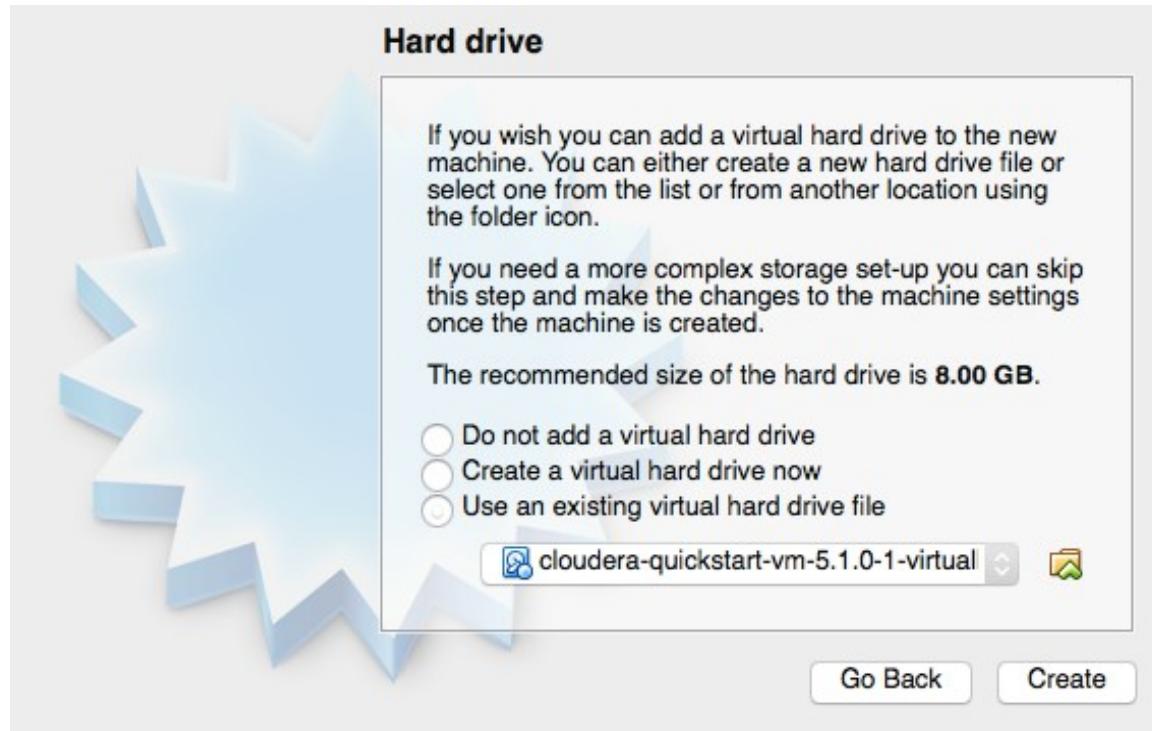


Starting Cloudera VM (cont)

Select >> Use an existing virtual hard drive file

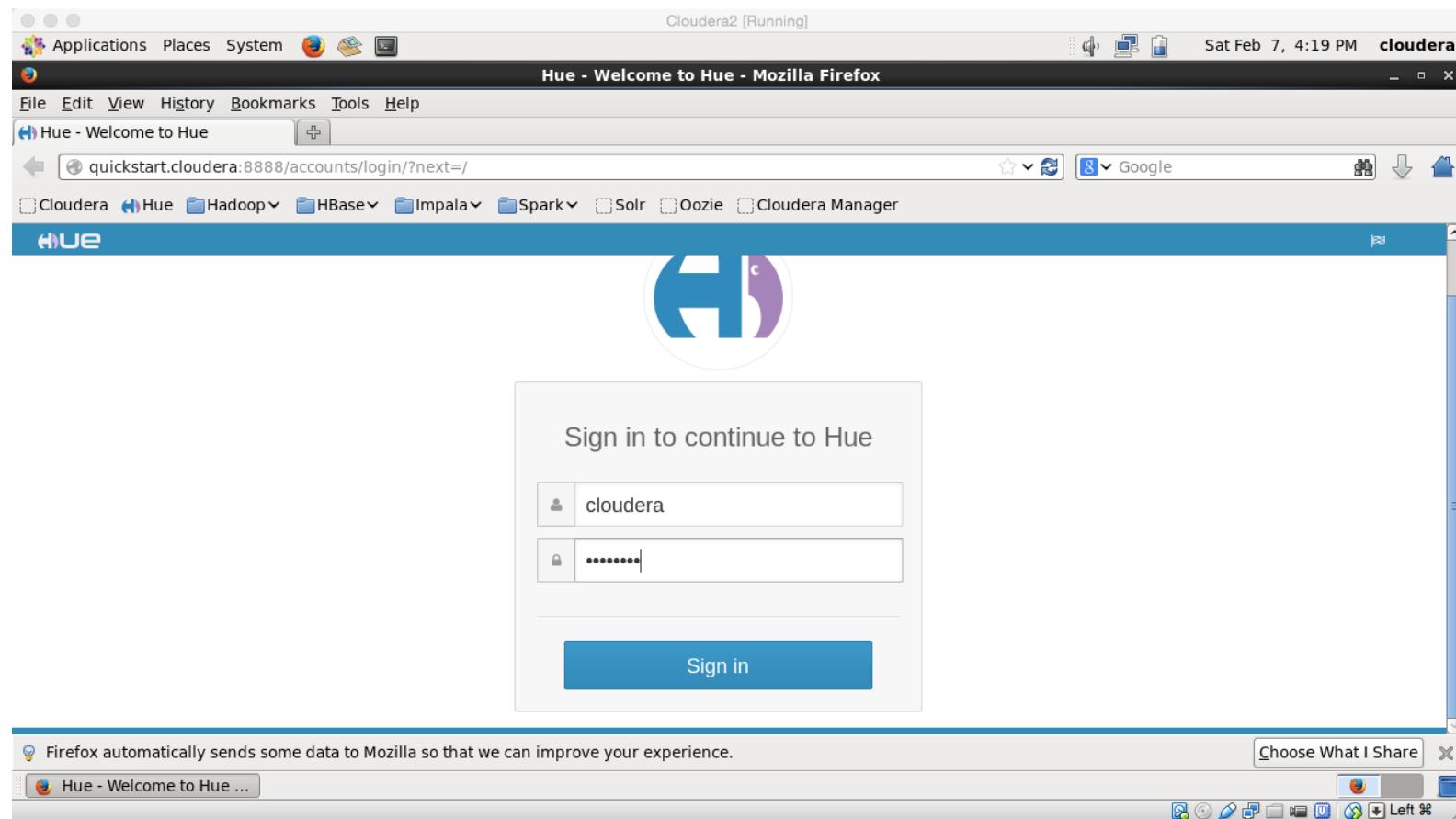
Locate to file cloudera-quickstart-vm-5.1.0-1-virtualbox-disk1.vmdk

Then click Start



Sign in to Hue

Username: cloudera; Password: cloudera



Starting Hue on Cloudera

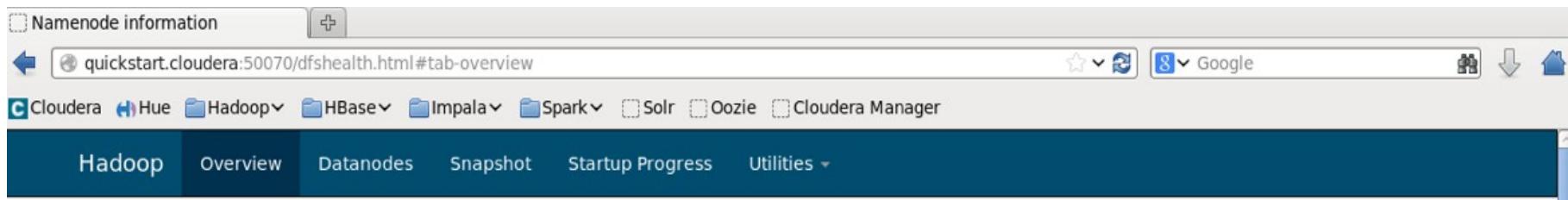
The screenshot shows the Hue File Browser interface. At the top, there's a header bar with tabs for Cloudera, Hue, Hadoop, HBase, Impala, Spark, Solr, Oozie, and Cloudera Manager. Below that is a navigation bar with links for Home, Query Editors, Data Browsers, Workflows, Search, File Browser (which is selected), Job Browser, and cloudera. The main area is titled "File Browser" and shows the path "/user/cloudera". There are buttons for Search, Rename, Move, Copy, Change permissions, Download, Move to trash, Upload, and New. The file listing table has columns for Name, Size, User, Group, Permissions, and Date. Two entries are listed: a folder named '.' owned by cloudera and a folder named '..' owned by hdfs.

Name	Size	User	Group	Permissions	Date
.		cloudera	cloudera	drwxr-xr-x	October 21, 2014 02:20 AM
..		hdfs	supergroup	drwxr-xr-x	October 20, 2014 11:03 PM

Viewing HDFS

The screenshot shows the Hue - File Browser interface. A vertical sidebar on the left contains links for HDFS NameNode, HDFS Secondary NameNode, HDFS DataNode, YARN ResourceManager, and YARN NodeManager. Red arrows point from the top of the slide towards these links. The main content area displays a file listing for the directory /user/cloudera. The table has columns for Name, Size, User, Group, Permissions, and Date. Two entries are listed: a folder named '.' and a file named 'hdfs'.

Name	Size	User	Group	Permissions	Date
.		cloudera	cloudera	drwxr-xr-x	October 21, 2014 02:20 AM
..		hdfs	supergroup	drwxr-xr-x	October 20, 2014 11:03 PM



The screenshot shows the Hadoop Overview page of the Cloudera Manager interface. The top navigation bar includes links for Cloudera, Hue, Hadoop, HBase, Impala, Spark, Solr, Oozie, and Cloudera Manager. The Hadoop tab is currently selected. The main content area displays the cluster overview, including the cluster name 'quickstart.cloudera:8020' (active), and a table with the following information:

Started:	Tue Oct 21 23:07:24 PDT 2014
Version:	2.3.0-cdh5.1.0, r8e266e052e423af592871e2dfe09d54c03f6a0e8
Compiled:	2014-07-12T13:49Z by jenkins from (no branch)
Cluster ID:	CID-829b265e-8a66-46c1-9914-1eef5c000c10
Block Pool ID:	BP-1205966836-127.0.0.1-1406828172945

Overview 'quickstart.cloudera:8020' (active)

Started:	Tue Oct 21 23:07:24 PDT 2014
Version:	2.3.0-cdh5.1.0, r8e266e052e423af592871e2dfe09d54c03f6a0e8
Compiled:	2014-07-12T13:49Z by jenkins from (no branch)
Cluster ID:	CID-829b265e-8a66-46c1-9914-1eef5c000c10
Block Pool ID:	BP-1205966836-127.0.0.1-1406828172945

Hands-On: Importing/Exporting Data to HDFS

Importing Data to Hadoop

Download War and Peace Full Text

www.gutenberg.org/ebooks/2600

The screenshot shows a web browser window with the title "War and Peace by graf Leo Tolstoy". On the left, there is a placeholder for a book cover with the text "No cover available". In the center, there is a table titled "Download This eBook" listing five download options:

Format	Size	Actions
Read this book online: HTML	3.6 MB	
EPUB (no images)	1.3 MB	
Kindle (no images)	5.1 MB	
Plain Text UTF-8	3.1 MB	
More Files...		

```
$hadoop fs -mkdir input
```

```
$hadoop fs -mkdir output
```

```
$hadoop fs -copyFromLocal Downloads/pg2600.txt input
```

Review file in Hadoop HDFS

List HDFS File

```
[cloudera@localhost ~]$ hadoop fs -ls input
Found 1 items
-rw-r--r-- 3 cloudera cloudera      15 2014-06-16 19:07 input/input_test.txt
[cloudera@localhost ~]$ █
```

Read HDFS File

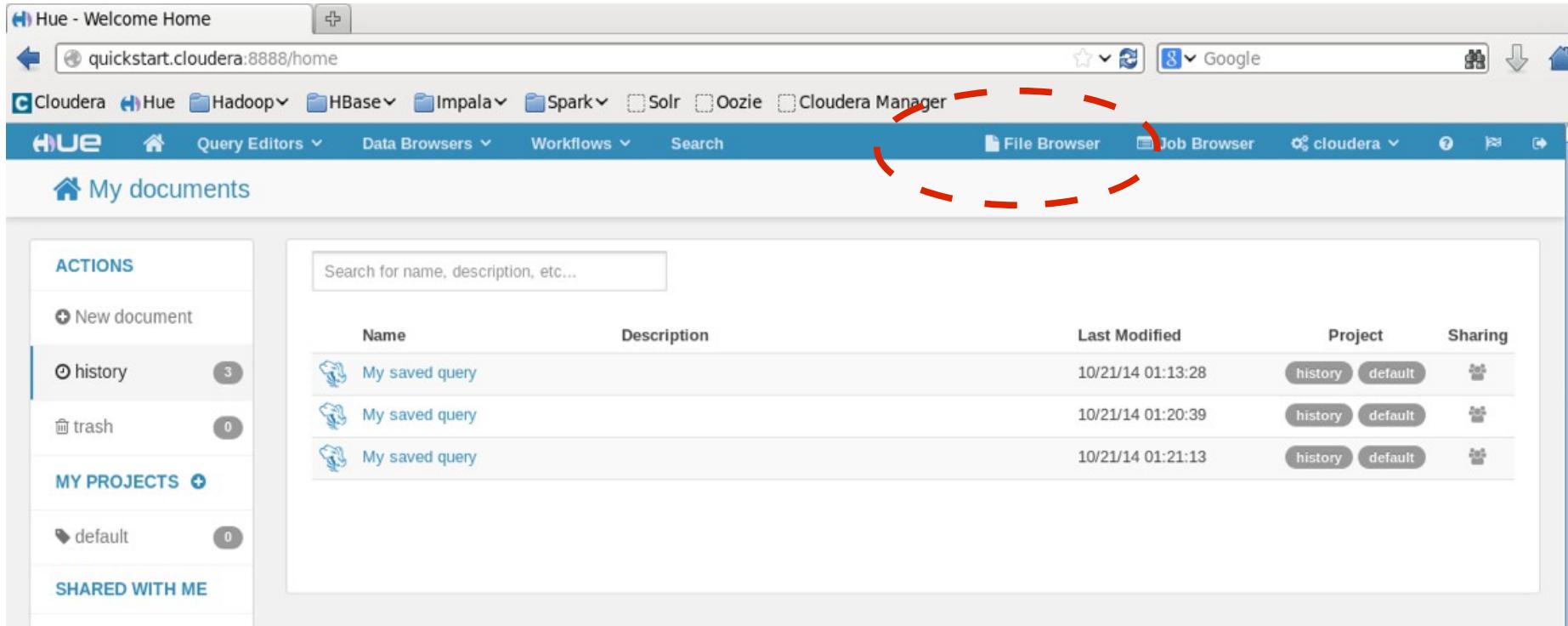
```
[hdadmin@localhost bin]$ hadoop fs -cat input/pg2600.txt
```

Retrieve HDFS File to Local File System

```
[hdadmin@localhost bin]$ hadoop fs -copyToLocal input/pg2600.txt tmp/file.txt
```

Please see also http://hadoop.apache.org/docs/r1.0.4/commands_manual.html

Review file in Hadoop HDFS using File Browse



The screenshot shows the Hue interface for managing Hadoop HDFS. The top navigation bar includes links for Cloudera, Hue, Hadoop, HBase, Impala, Spark, Solr, Oozie, and Cloudera Manager. The main menu bar has options for HUE, Home, Query Editors, Data Browsers, Workflows, Search, File Browser (which is highlighted with a red dashed circle), Job Browser, cloudera, Help, and Logout. The left sidebar under 'ACTIONS' shows 'New document', 'history' (with 3 items), and 'trash'. Under 'MY PROJECTS', there is a 'default' project (with 0 items). Under 'SHARED WITH ME', there are no items. The central content area displays a search bar and a table of saved queries:

Name	Description	Last Modified	Project	Sharing
My saved query		10/21/14 01:13:28	history default	
My saved query		10/21/14 01:20:39	history default	
My saved query		10/21/14 01:21:13	history default	

Review file in Hadoop HDFS using Hue

The screenshot shows the Hue interface with the 'File Browser' tab selected. The top navigation bar includes links for Cloudera, Hue, Hadoop, HBase, Impala, Spark, Solr, Oozie, and Cloudera Manager. Below the navigation is a toolbar with actions: Rename, Move, Copy, Change permissions, Download, Move to trash, Upload, and New. The main area displays a file listing for the '/user/cloudera' directory. The table has columns for Name, Size, User, Group, Permissions, and Date. The data is as follows:

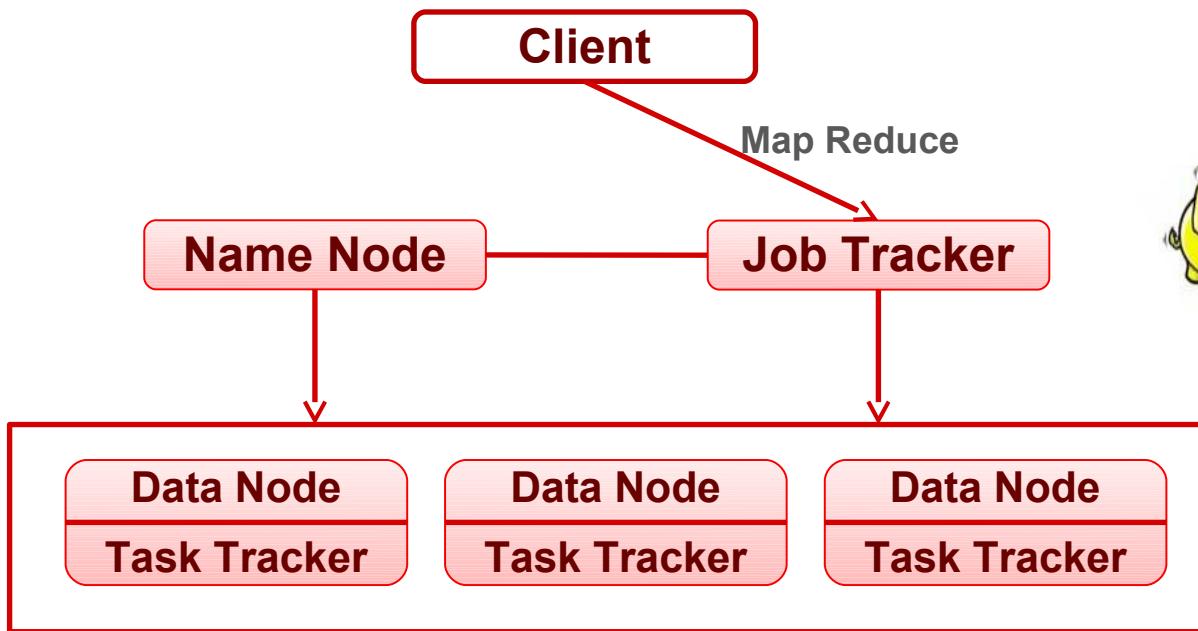
Name	Size	User	Group	Permissions	Date
.		cloudera	cloudera	drwxr-xr-x	October 21, 2014 02:20 AM
..		hdfs	supergroup	drwxr-xr-x	October 20, 2014 11:03 PM
input		cloudera	cloudera	drwxr-xr-x	October 20, 2014 11:12 PM
output		cloudera	cloudera	drwxr-xr-x	October 21, 2014 12:10 AM

Hadoop Port Numbers

	Daemon	Default Port	Configuration Parameter in conf/*-site.xml
HDFS	Namenode	50070	dfs.http.address
	Datanodes	50075	dfs.datanode.http.address
	Secondarynamenode	50090	dfs.secondary.http.address
MR	JobTracker	50030	mapred.job.tracker.http.address
	Tasktrackers	50060	mapred.task.tracker.http.address

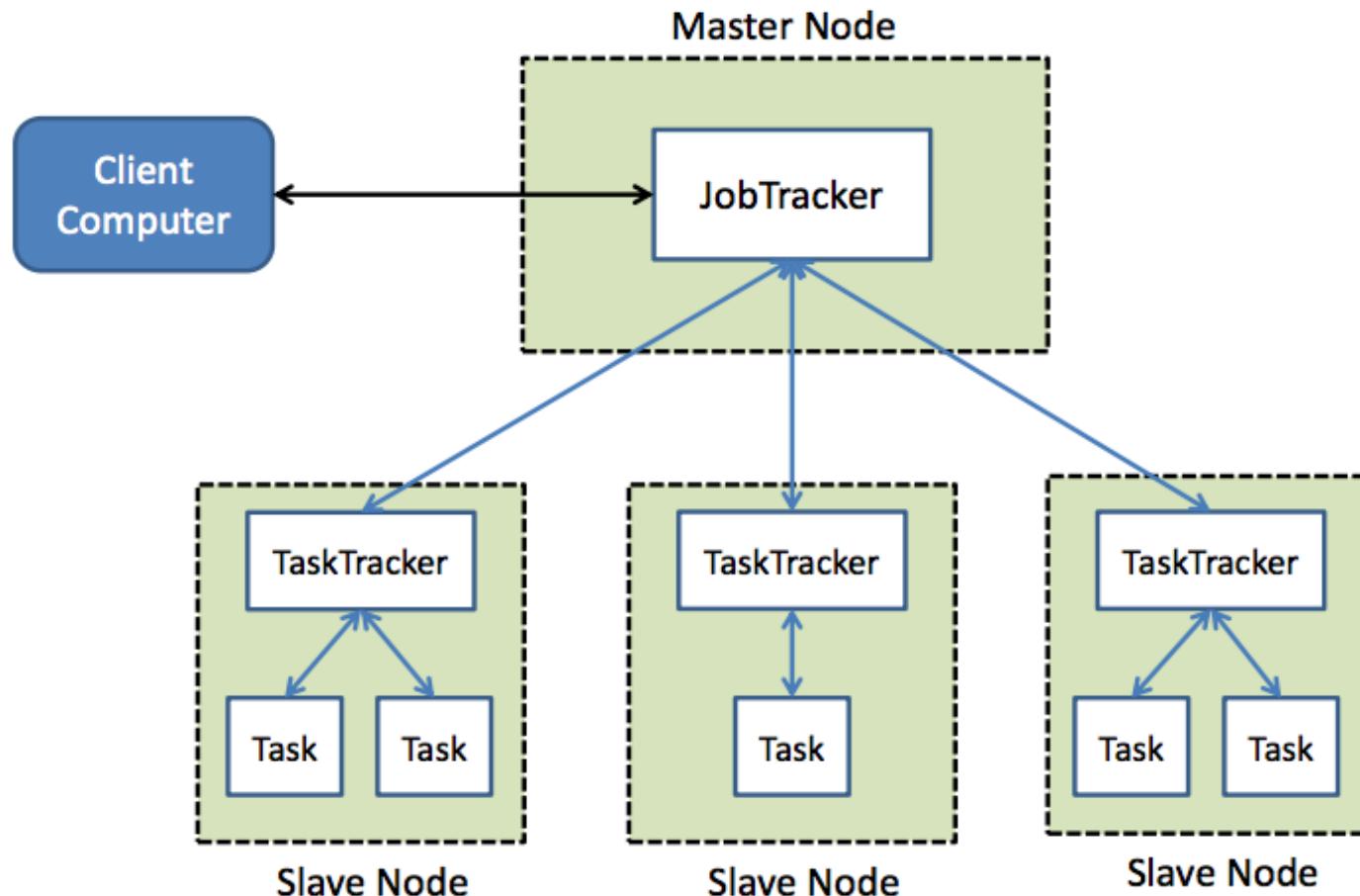
Removing data from HDFS using Shell Command

```
hdadmin@localhost detach]$ hadoop fs -rm input/pg2600.txt
Deleted hdfs://localhost:54310/input/pg2600.txt
hdadmin@localhost detach]$
```



Lecture: Understanding Map Reduce Processing

High Level Architecture of MapReduce



Before MapReduce...

- Large scale data processing was difficult!
 - Managing hundreds or thousands of processors
 - Managing parallelization and distribution
 - I/O Scheduling
 - Status and monitoring
 - Fault/crash tolerance
- MapReduce provides all of these, easily!

Source: <http://labs.google.com/papers/mapreduce-osdi04-slides/index-auto-0002.html>

MapReduce Overview

- What is it?
 - Programming model used by Google
 - A combination of the Map and Reduce models with an associated implementation
 - Used for processing and generating large data sets

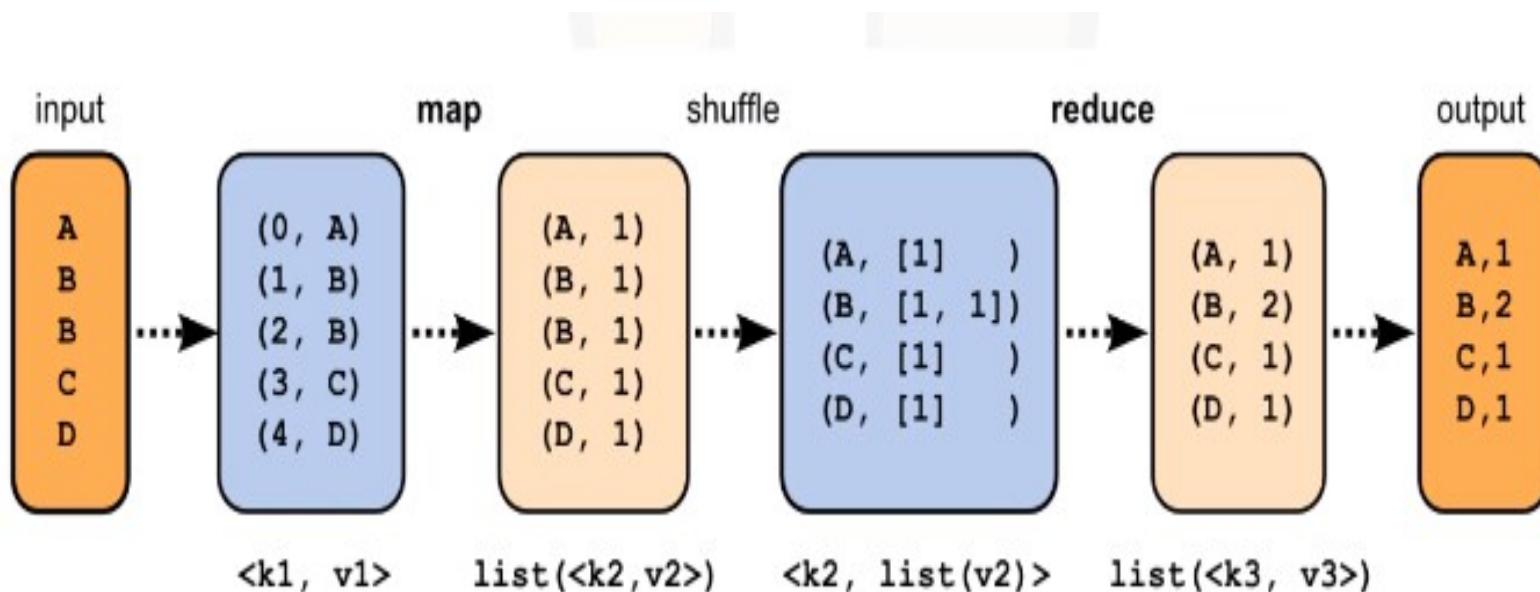
Source: <http://labs.google.com/papers/mapreduce-osdi04-slides/index-auto-0002.html>

MapReduce Overview

- How does it solve our previously mentioned problems?
 - MapReduce is highly scalable and can be used across many computers.
 - Many small machines can be used to process jobs that normally could not be processed by a large machine.

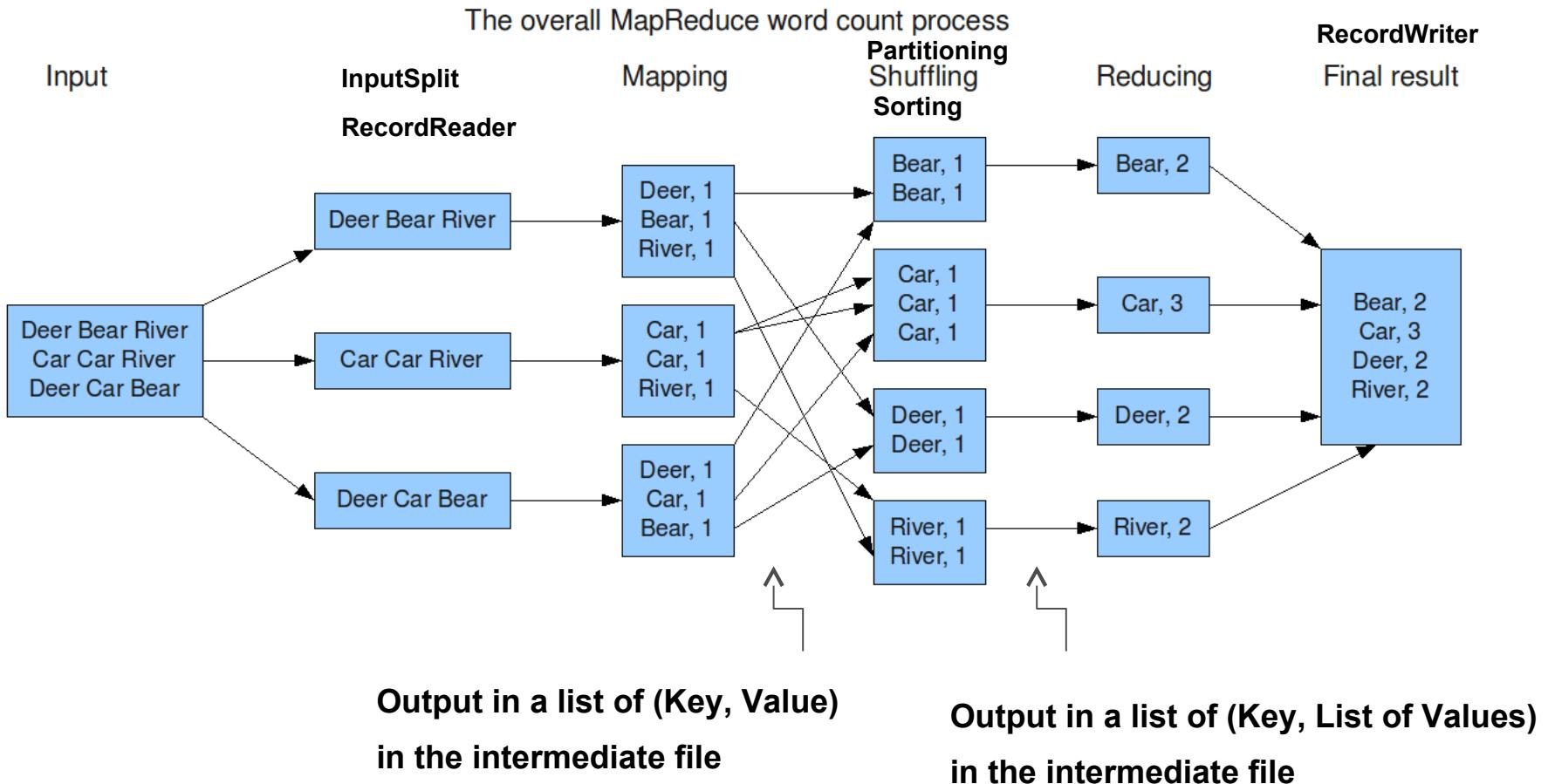
Source: <http://labs.google.com/papers/mapreduce-osdi04-slides/index-auto-0002.html>

MapReduce Framework

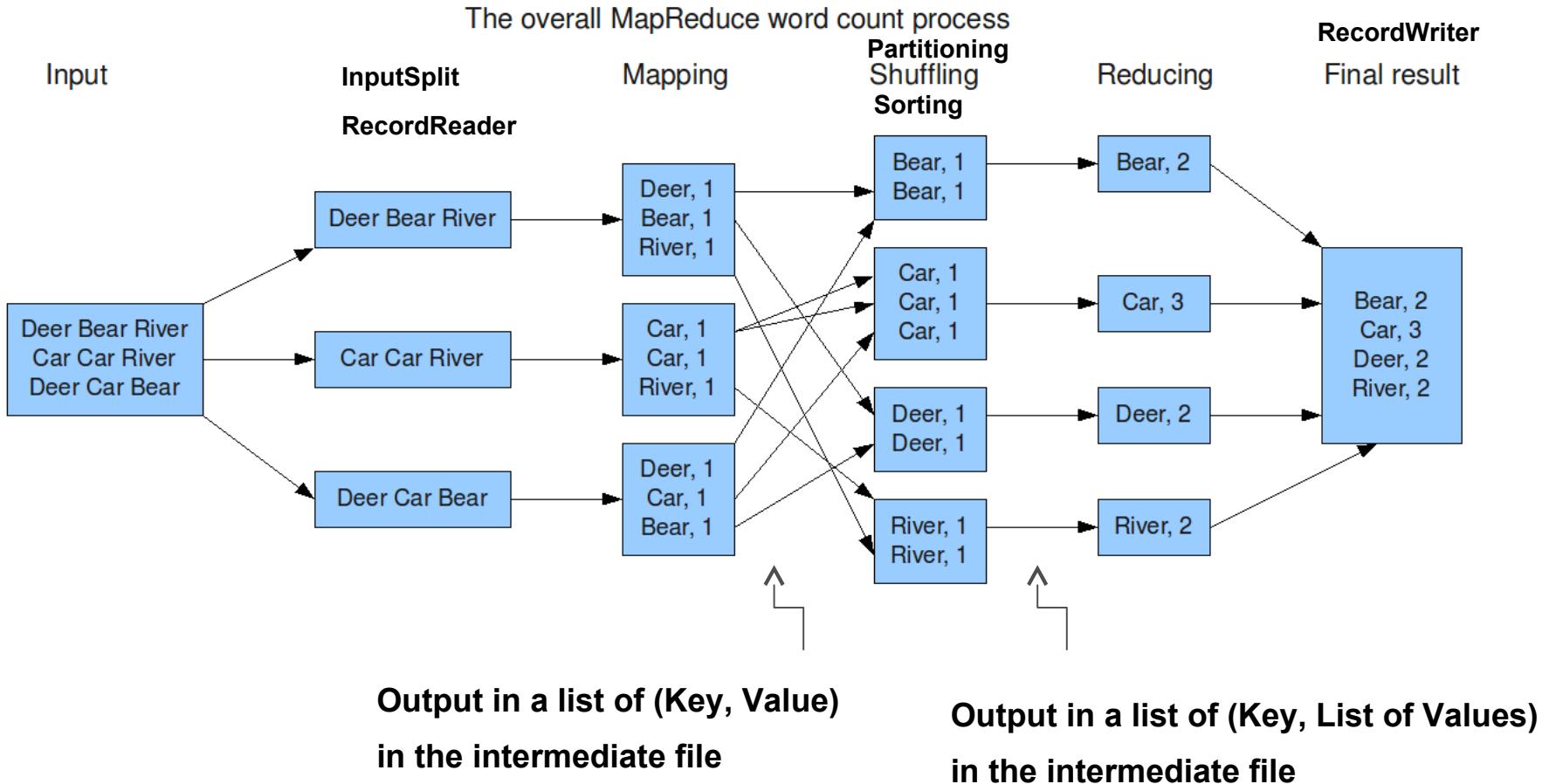


Source: www.bigdatauniversity.com

How does the MapReduce work?



How does the MapReduce work?



Map Abstraction

- Inputs a key/value pair
 - Key is a reference to the input value
 - Value is the data set on which to operate
- Evaluation
 - Function defined by user
 - Applies to every value in value input
 - Might need to parse input
- Produces a new list of key/value pairs
 - Can be different type from input pair

Source: <http://labs.google.com/papers/mapreduce-osdi04-slides/index-auto-0002.html>

Reduce Abstraction

- Starts with intermediate Key / Value pairs
 - Ends with finalized Key / Value pairs
-
- Starting pairs are sorted by key
 - Iterator supplies the values for a given key to the Reduce function.

Source: <http://labs.google.com/papers/mapreduce-osdi04-slides/index-auto-0002.html>

Reduce Abstraction

- Typically a function that:
 - Starts with a large number of key/value pairs
 - One key/value for each word in all files being grep'd (including multiple entries for the same word)
 - Ends with very few key/value pairs
 - One key/value for each unique word across all the files with the number of instances summed into this entry
- Broken up so a given worker works with input of the same key.

Source: <http://labs.google.com/papers/mapreduce-osdi04-slides/index-auto-0002.html>

How Map and Reduce Work Together

- Map returns information
- Reduces accepts information
- Reduce applies a user defined function to reduce the amount of data

Other Applications

- Yahoo!
 - Webmap application uses Hadoop to create a database of information on all known webpages
- Facebook
 - Hive data center uses Hadoop to provide business statistics to application developers and advertisers
- Rackspace
 - Analyzes sever log files and usage data using Hadoop

MapReduce Framework

map: (K1 , V1) -> list(K2 , V2)

reduce: (K2 , list(V2)) -> list(K3 , V3)

MapReduce Processing – The Data flow

1. **InputFormat, InputSplits, RecordReader**
2. **Mapper - your focus is here**
3. **Partition, Shuffle & Sort**
4. **Reducer - your focus is here**
5. **OutputFormat, RecordWriter**

InputFormat

InputFormat	Description	Key	Value
TextInputFormat	Default format; reads lines of text files	The byte offset of the line	The line contents
KeyValueInputFormat	Parses lines into key, val pairs	Everything up to the first tab character	The remainder of the line
SequenceFileInputFormat	A Hadoop-specific high-performance binary format	user-defined	user-defined

InputSplit

An InputSplit describes a unit of work that comprises a single *map task*.

InputSplit presents a byte-oriented view of the input.

You can control this value by setting the mapred.min.split.size parameter in core-site.xml, or by overriding the parameter in the JobConf object used to submit a particular MapReduce job.

RecordReader

RecordReader reads <key, value> pairs from an InputSplit.

Typically the RecordReader converts the byte-oriented view of the input, provided by the InputSplit, and presents a record-oriented to the Mapper

Mapper

Mapper: The Mapper performs the user-defined logic to the input a key, value and emits (key, value) pair(s) which are forwarded to the Reducers.

Partition, Shuffle & Sort

After the first map tasks have completed, the nodes may still be performing several more map tasks each. But they also begin exchanging the intermediate outputs from the map tasks to where they are required by the reducers.

Partitioner controls the partitioning of map-outputs to assign to reduce task . the total number of partitions is the same as the number of reduce tasks for the job

The set of intermediate keys on a single node is automatically sorted by internal Hadoop before they are presented to the Reducer

This process of moving map outputs to the reducers is known as shuffling.

Reducer

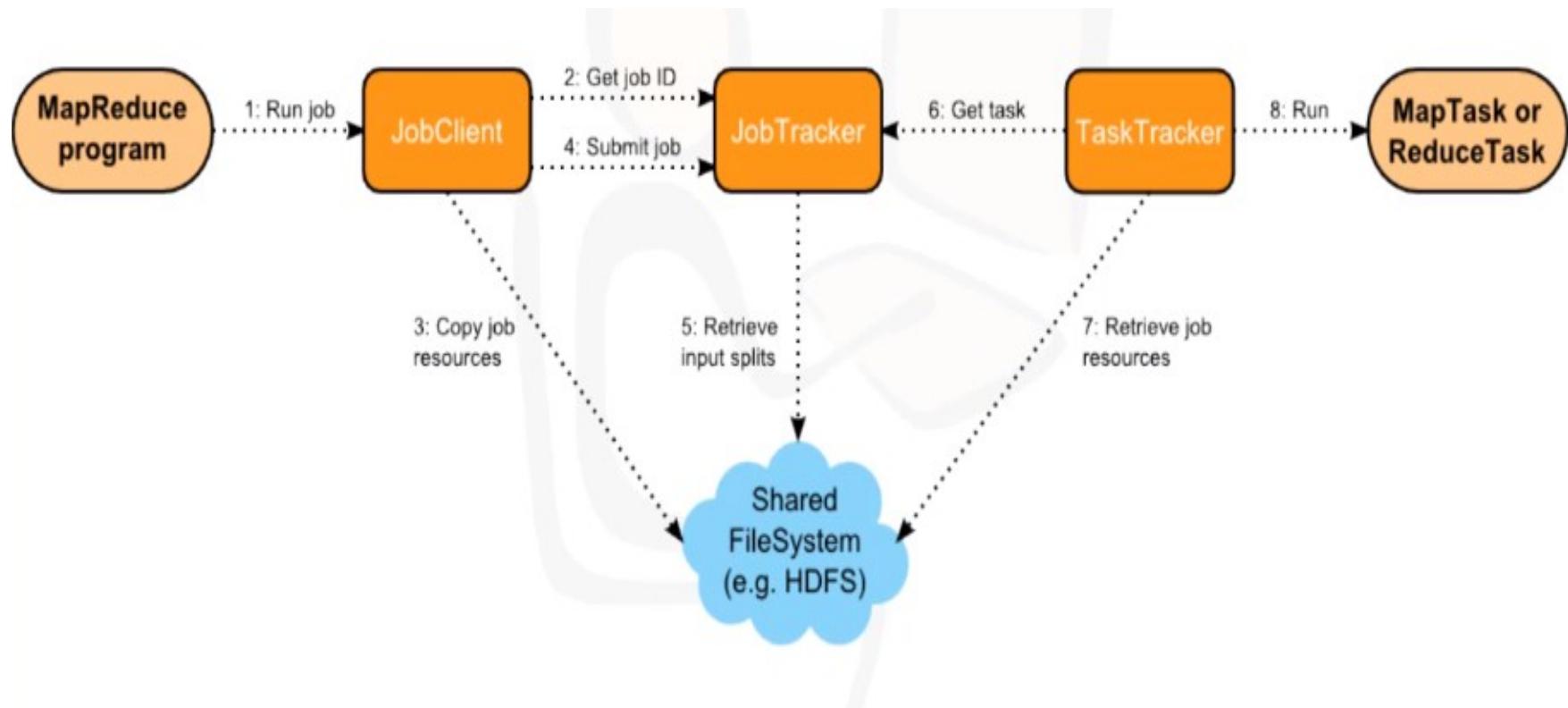
This is an instance of user-provided code that performs read each key, iterator of values in the partition assigned. The *OutputCollector* object in Reducer phase has a method named `collect()` which will collect a (key, value) output.

OutputFormat, Record Writer

OutputFormat governs the writing format in *OutputCollector* and **RecordWriter** writes output into HDFS.

TextOutputFormat	Default; writes lines in "key \t value" form
SequenceFileOutputFormat	Writes binary files suitable for reading into subsequent MapReduce jobs
NullOutputFormat	generates no output files

Submitting a MapReduce job



Source: www.bigdatauniversity.com

Hands-On: Writing your own Map Reduce Program

Wordcount (HelloWord in Hadoop)

```
1. package org.myorg;
2.
3. import java.io.IOException;
4. import java.util.*;
5.
6. import org.apache.hadoop.fs.Path;
7. import org.apache.hadoop.conf.*;
8. import org.apache.hadoop.io.*;
9. import org.apache.hadoop.mapred.*;
10. import org.apache.hadoop.util.*;

11.
12. public class WordCount {
13.
14.     public static class Map extends MapReduceBase implements Mapper<LongWritable, Text, Text,
15.     IntWritable> {
16.         private final static IntWritable one = new IntWritable(1);
17.         private Text word = new Text();
18.
19.         public void map(LongWritable key, Text value, OutputCollector<Text, IntWritable> output,
20. Reporter reporter) throws IOException {
21.             String line = value.toString();
22.             StringTokenizer tokenizer = new StringTokenizer(line);
23.             while (tokenizer.hasMoreTokens()) {
24.                 word.set(tokenizer.nextToken());
25.                 output.collect(word, one);
26.             }
27.         }
28.     }
29. }
```

Wordcount (HelloWord in Hadoop)

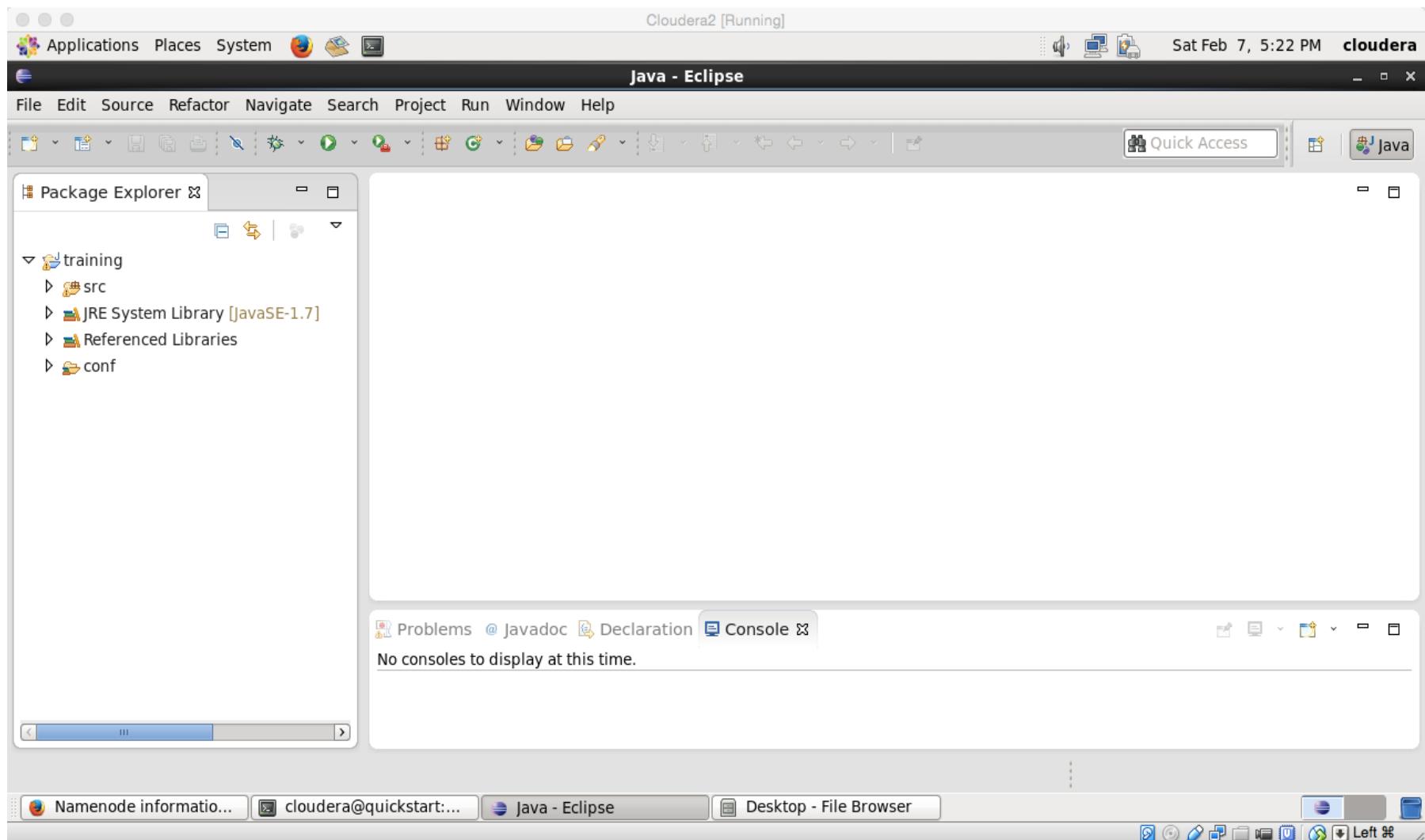
```
27.  
28. public static class Reduce extends MapReduceBase implements Reducer<Text, IntWritable, Text,  
IntWritable> {  
29.     public void reduce(Text key, Iterator<IntWritable> values, OutputCollector<Text, IntWritable>  
output, Reporter reporter) throws IOException {  
30.         int sum = 0;  
31.         while (values.hasNext()) {  
32.             sum += values.next().get();  
33.         }  
34.         output.collect(key, new IntWritable(sum));  
35.     }  
36. }  
37.
```

Wordcount (HelloWord in Hadoop)

```
38. public static void main(String[] args) throws Exception {  
39.     JobConf conf = new JobConf(WordCount.class);  
40.     conf.setJobName("wordcount");  
41.  
42.     conf.setOutputKeyClass(Text.class);  
43.     conf.setOutputValueClass(IntWritable.class);  
44.  
45.     conf.setMapperClass(Map.class);  
46.  
47.     conf.setReducerClass(Reduce.class);  
48.  
49.     conf.setInputFormat(TextInputFormat.class);  
50.     conf.setOutputFormat(TextOutputFormat.class);  
51.  
52.     FileInputFormat.setInputPaths(conf, new Path(args[1]));  
53.     FileOutputFormat.setOutputPath(conf, new Path(args[2]));  
54.  
55.     JobClient.runJob(conf);  
56. }  
57. }  
58. }  
59.
```

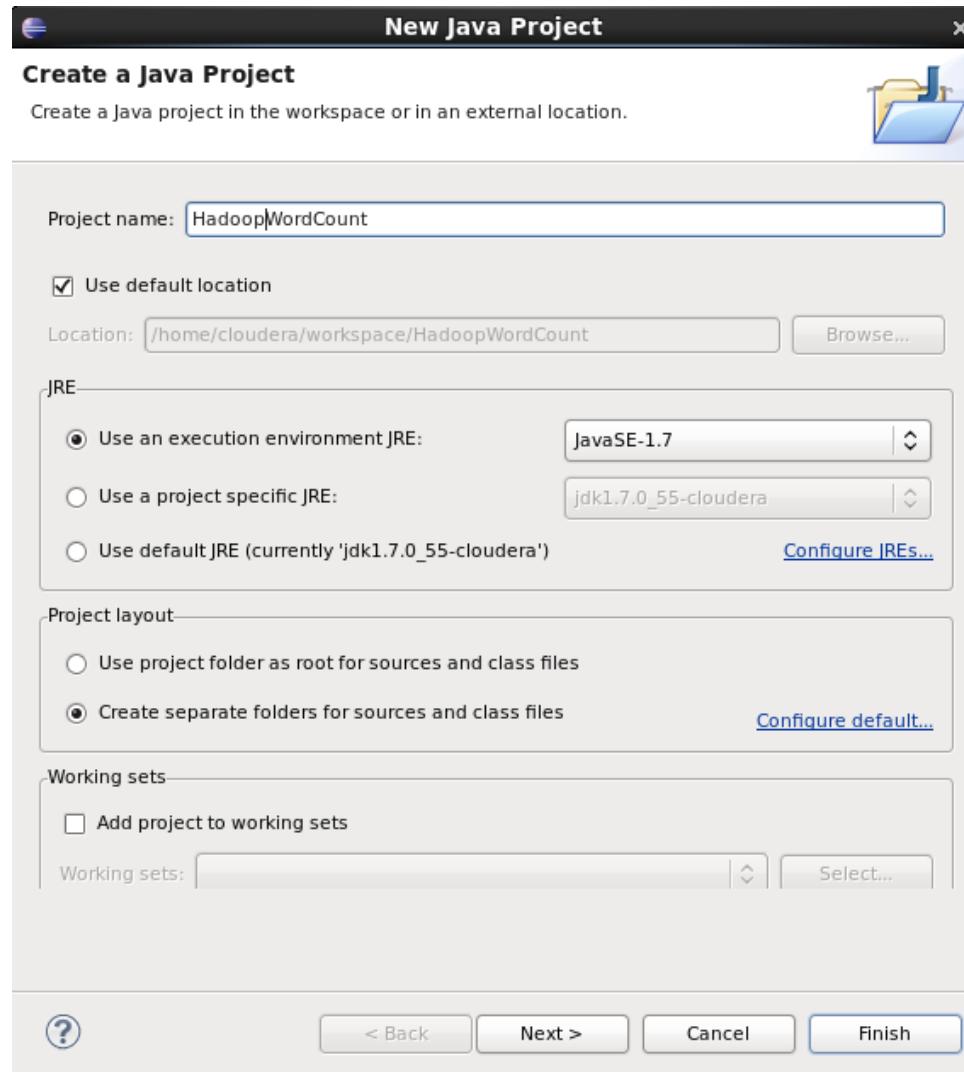
Hands-On: Writing Map/Reduce Program on Eclipse

Starting Eclipse in Cloudera VM



Create a Java Project

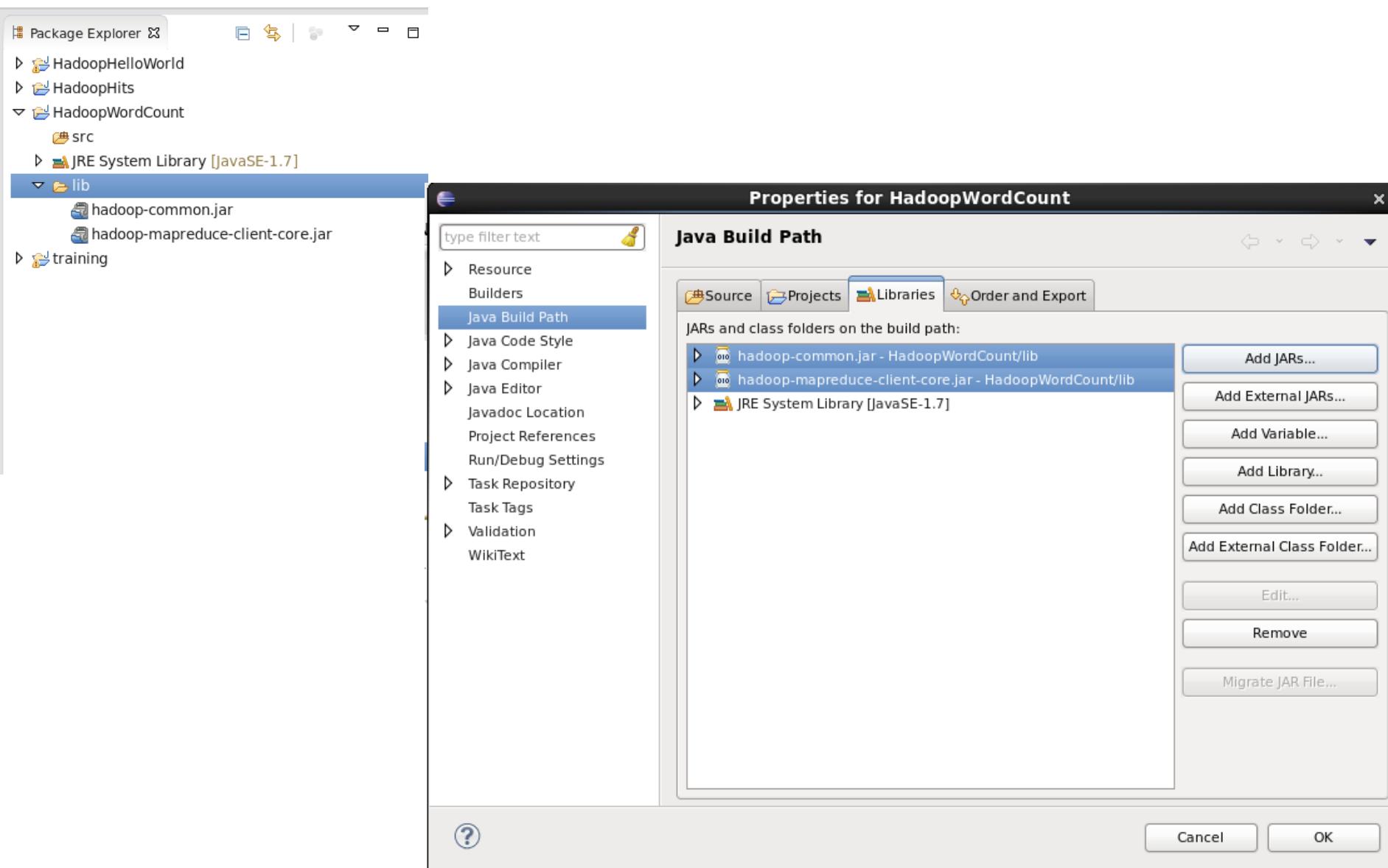
Let's name it HadoopWordCount



Add dependencies to the project

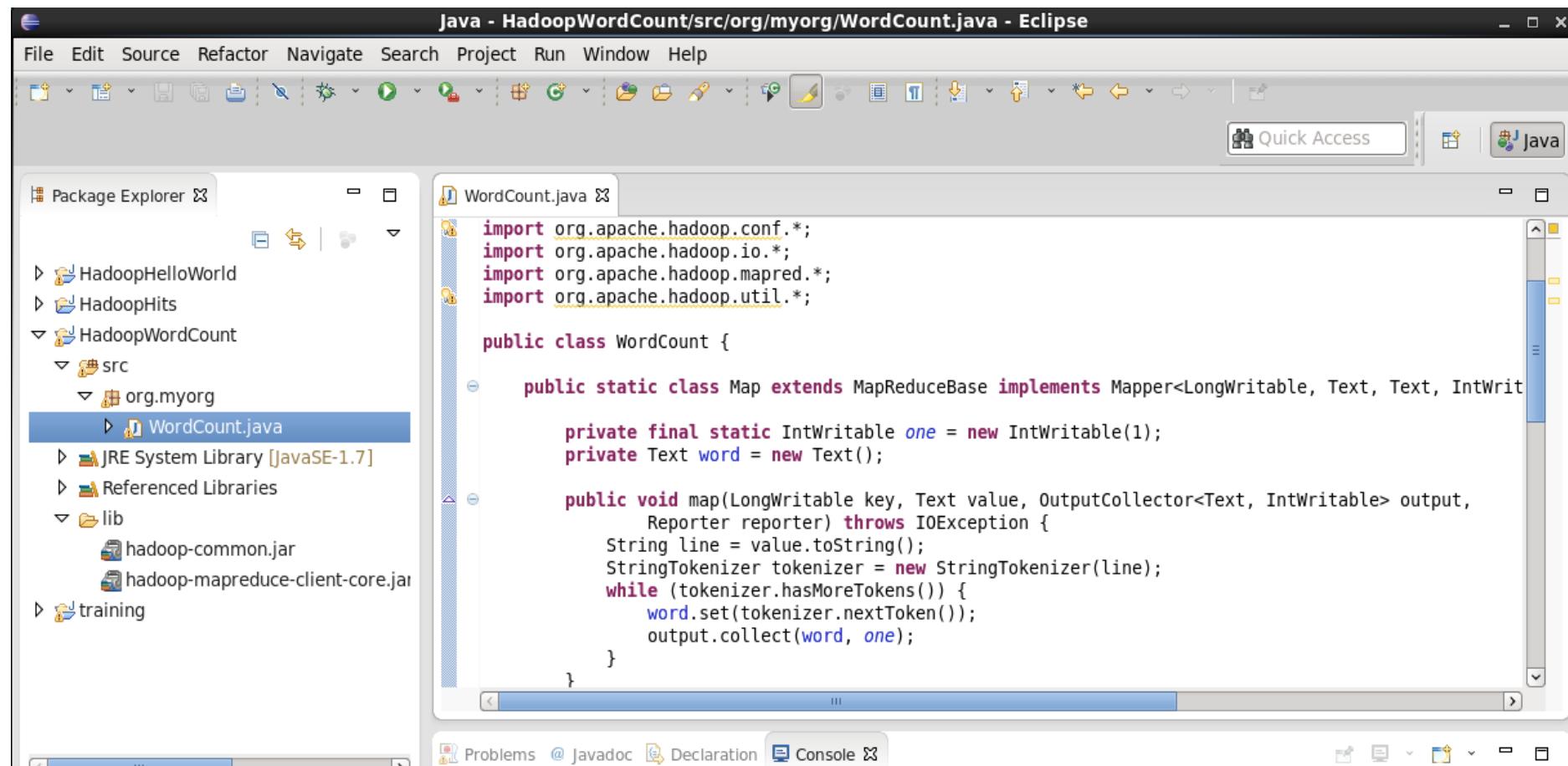
- Add the following two JARs to your build path
- [hadoop-common.jar](#) and [hadoop-mapreduce-client-core.jar](#). Both can be founded at [/usr/lib/hadoop/client](#)
- By perform the following steps
 - Add a folder named [lib](#) to the project
 - Copy the mentioned JARs in this folder
 - Right-click on the project name >> select **Build Path** >> then **Configure Build Path**
 - Click on Add Jars, select these two JARs from the [lib](#) folder

Add dependencies to the project



Writing a source code

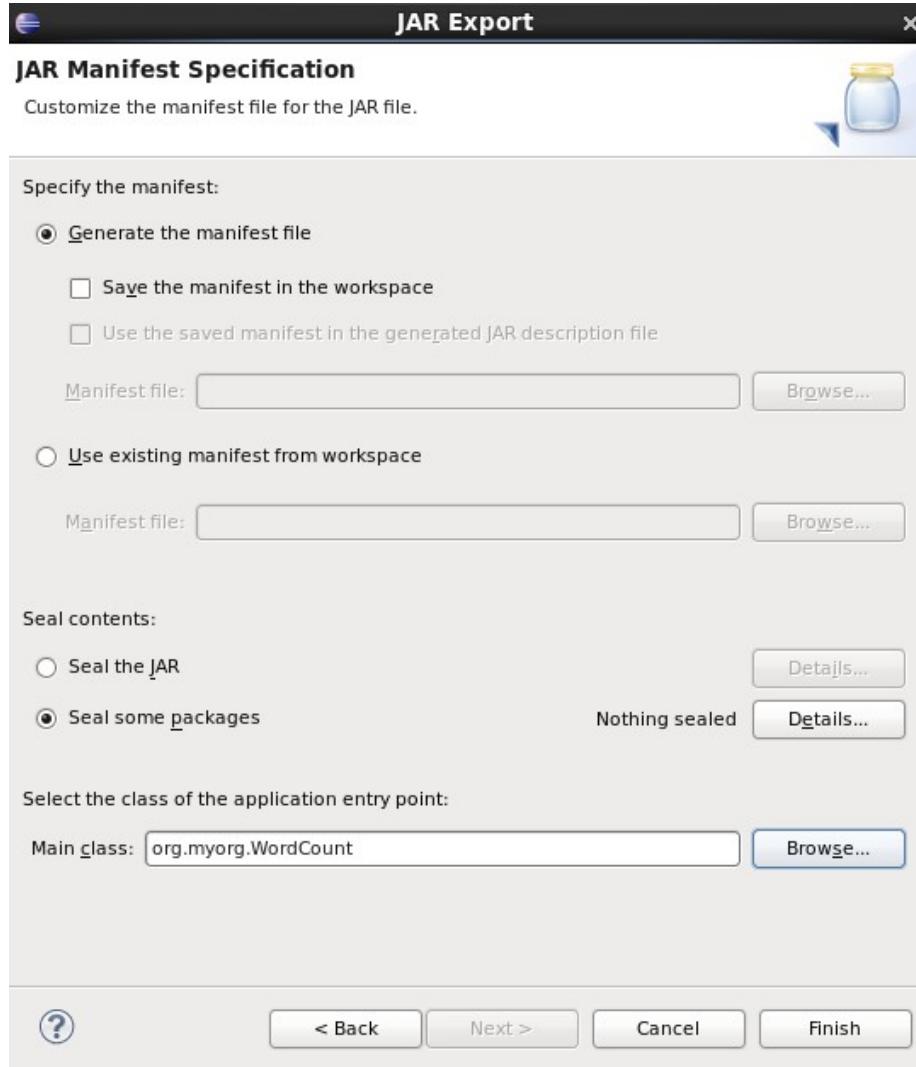
- Right click the project, the select **New >> Package**
- Name the package as org.myorg
- Right click at org.myorg, the select **New >> Class**
- Name the package as WordCount
- Writing a source code as shown in previous slides



Building a Jar file

- Right click the project, then select **Export**
- Select **Java** and then **JAR** file
- Provide the JAR name, as [wordcount.jar](#)
- Leave the **JAR package options** as default
- In the **JAR Manifest Specification** section, in the bottom, specify the **Main class**
- In this case, select WordCount
- Click on **Finish**
- The JAR file will be build and will be located at cloudera/workspace

Note: you may need to re-size the dialog font size by select
Windows >> Preferences >> Appearance >> Colors and Fonts



Hands-On: Running Map Reduce and Deploying to Hadoop Runtime Environment

Running Map Reduce Program

```
[cloudera@quickstart ~]$ cd workspace/  
[cloudera@quickstart workspace]$ hadoop jar wordcount.jar org.myorg.WordCount input/* output/wordcount_output  
15/02/08 10:30:31 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032  
15/02/08 10:30:32 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032  
15/02/08 10:30:33 WARN mapreduce.JobSubmitter: Hadoop command-line option parsing not performed. Implement the Tool interface with ToolRunner to remedy this.  
15/02/08 10:30:33 INFO mapred.FileInputFormat: Total input paths to process : 1  
15/02/08 10:30:34 INFO mapreduce.JobSubmitter: number of splits:2  
15/02/08 10:30:34 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1423408479621_0009  
15/02/08 10:30:35 INFO impl.YarnClientImpl: Submitted application application_1423408479621_0009  
15/02/08 10:30:35 INFO mapreduce.Job: The url to track the job: http://quickstart.cloudera:8088/proxy/application_142  
15/02/08 10:30:35 INFO mapreduce.Job: Running job: job_1423408479621_0009  
15/02/08 10:30:52 INFO mapreduce.Job: Job job_1423408479621_0009 running in uber mode : false  
15/02/08 10:30:52 INFO mapreduce.Job: map 0% reduce 0%  
15/02/08 10:31:22 INFO mapreduce.Job: map 58% reduce 0%  
15/02/08 10:31:25 INFO mapreduce.Job: map 100% reduce 0%  
15/02/08 10:31:52 INFO mapreduce.Job: map 100% reduce 100%  
15/02/08 10:31:53 INFO mapreduce.Job: Job job_1423408479621_0009 completed successfully  
15/02/08 10:31:53 INFO mapreduce.Job: Counters: 49
```

Reviewing MapReduce Job in Hue

The screenshot shows the Hue Job Browser interface. At the top, there's a header bar with the title "Hue - Job Browser" and a search bar containing "quickstart.cloudera:8888/jobbrowser/". Below the header is a navigation bar with links to Cloudera, Hue, Hadoop, HBase, Impala, Spark, Solr, Oozie, and Cloudera Manager. The main content area is titled "Job Browser" and features a search bar with "Username: cloudera" and a "Search for text" input field. To the right of the search bar are buttons for "Succeeded", "Running" (which is highlighted in orange), "Failed", and "Killed". Below this is a table listing the running job. The table has columns: Logs, ID, Name, Status, User, Maps, Reduces, Queue, Priority, Duration, and Submitted. The single entry in the table is "1413756276038_0001 wordcount" with the status "RUNNING" and other details like "User: cloudera", "Maps: 5%", "Reduces: 5%", "Queue: root.cloudera", "Priority: N/A", "Duration: 55s", and "Submitted: 10/19/14 16:03:41". At the bottom of the table, it says "Showing 1 to 1 of 1 entries" and has navigation buttons for "Previous", "1", and "Next".

Logs	ID	Name	Status	User	Maps	Reduces	Queue	Priority	Duration	Submitted
	1413756276038_0001	wordcount	RUNNING	cloudera	5%	5%	root.cloudera	N/A	55s	10/19/14 16:03:41

Reviewing MapReduce Job in Hue

The screenshot shows the Hue Job Browser interface for a completed MapReduce job. The job ID is 1413756276038_0001, run by user cloudera, and has a status of SUCCEEDED. The interface displays recent tasks, including three MAP tasks and one REDUCE task.

LOGS	Logs
SUCCESSFUL	task_1413756276038_0001_m_000000
SUCCESSFUL	task_1413756276038_0001_m_000001
SUCCESSFUL	task_1413756276038_0001_r_000000

Recent Tasks

Logs	Type
task_1413756276038_0001_m_000000	MAP
task_1413756276038_0001_m_000001	MAP
task_1413756276038_0001_r_000000	REDUCE

Reviewing MapReduce Output Result

The screenshot shows the Hue File Browser interface. The top navigation bar includes links for Home, Query Editors, Data Browsers, Workflows, Search, File Browser, Job Browser, cloudera, and various system icons. The main area is titled "File Browser" and displays the contents of the "/user/cloudera" directory. The directory structure is as follows:

- .. (Size: 0, User: cloudera, Group: cloudera, Permissions: drwxr-xr-x, Date: October 19, 2014 03:20 PM)
- .Trash (Size: 0, User: cloudera, Group: cloudera, Permissions: drwxr-xr-x, Date: October 19, 2014 03:50 PM)
- input (Size: 0, User: cloudera, Group: cloudera, Permissions: drwxr-xr-x, Date: October 19, 2014 03:51 PM)
- output (Size: 0, User: cloudera, Group: cloudera, Permissions: drwxr-xr-x, Date: October 19, 2014 04:04 PM)

Reviewing MapReduce Output Result

The screenshot shows the Hue File Browser interface. The top navigation bar includes links for Home, Query Editors, Data Browsers, Workflows, Search, File Browser, Job Browser, cloudera, and various system icons. The main title is "File Browser". The current path is displayed as "/ user / cloudera / output / wordcount_output". A trash icon is also present in the top right.

The main content area displays a table of files in the "wordcount_output" directory:

Name	Size	User	Group	Permissions	Date
.		cloudera	cloudera	drwxr-xr-x	February 08, 2015 10:31 AM
..		cloudera	cloudera	drwxr-xr-x	February 08, 2015 10:30 AM
_SUCCESS	0 bytes	cloudera	cloudera	-rw-r--r--	February 08, 2015 10:31 AM
part-00000	456.9 KB	cloudera	cloudera	-rw-r--r--	February 08, 2015 10:31 AM

Reviewing MapReduce Output Result

The screenshot shows the Hue File Browser interface. The left sidebar has a 'INFO' section with 'Last modified' (Feb. 8, 2015, 10:31 a.m.), 'User' (cloudera), and 'Group'. The main area shows a list of words and their counts from a file named 'part-00000' located at /user/cloudera/output/wordcount_output. The counts are:

Word	Count
"About	6
"According	1
"Adele	1
"Adieu,	2
"Adjutant!"	1
"Admirable!"	1
"Adorable!	1
"Adored	1
"Afraid	3
"After	7
"Again!"	1

Hands-On: Running Map Reduce using Oozie workflow

Using Hue: select WorkFlow >> Editor

The screenshot shows the Hue interface for managing Oozie workflows. The browser title bar reads "Hue - Oozie Editor/Dashboard - Workflows - Mozilla Firefox". The address bar shows the URL "quickstart.cloudera:8888/oozie/list_workflows/". The top navigation bar includes links for File, Edit, View, History, Bookmarks, Tools, Help, Public - Dropbox, and Hue - Oozie Editor/Dashboard. Below the navigation bar is a toolbar with icons for Cloudera Manager, Hue, Hadoop, HBase, Impala, Spark, Solr, Oozie, and Cloudera Manager. The main menu bar has sections for HUE, Home, Query Editors, Data Browsers, Workflows, Search, File Browser, Job Browser, cloudera, Help, and Logout.

The main content area is titled "Workflow Manager". It features a search bar with placeholder text "Search for name, description, etc...". Below the search bar are buttons for "Submit", "Schedule", "Copy", "Export", "Move to trash", "Create", "Import", and "Trash". A table header row includes columns for Name, Description, Last Modified, Steps, Status, and Owner, each with a sorting arrow icon. A message "No data available" is displayed below the table. At the bottom, it says "Showing 0 to 0 of 0 entries" and includes "Previous" and "Next" navigation buttons.

Create a new workflow

- Click Create button; the following screen will be displayed
- Name the workflow as WordCountWorkflow

The screenshot shows the Hue Oozie Editor interface. The top navigation bar includes links for HUE, Home, Query Editors, Data Browsers, Workflows (which is currently selected), and Search. To the right are links for File Browser, Job Browser, cloudera, and help.

The main area is titled "Properties" under "NEW WORKFLOW". On the left, there's a sidebar with "Properties" selected. The main form has fields for "Name" (containing "WordCountWokflow") and "Description". Below the form is an "advanced" link. At the bottom are "Save" and "Back" buttons.

Select a Java job for the workflow

- From the Oozie editor, drag **Java** and drop between start and end

The screenshot shows the HUE Oozie Editor interface. The top navigation bar includes links for Home, Query Editors, Data Browsers, Workflows, Search, File Browser, Job Browser, and cloudera. The main area is titled "Oozie Editor" and has tabs for Workflows, Coordinators, and Bundles. On the left, there's a sidebar with sections for Properties, Workspace, ADVANCED (Import action, Kill node), History, and ACTIONS (Submit). The central workspace shows a workflow graph with a "start" node at the top, a "Java" action node in the middle, and an "end" node at the bottom. A "Properties" panel on the right lists various action types: MapReduce, Streaming, Java, Pig, Hive, Swoop, Shell, Ssh, DistCp, Fs, Email, Sub-workflow, and Generic.

Edit the Java Job

- Assign the following value
 - Name: WordCount
 - Jar name: wordcount.jar (select ... choose upload from local machine)
 - Main Class: org.myorg.WordCount
 - Arguments: input/* output/wordcount_output2

Edit Node: WordCount



Jar name	wordcount.jar	...
Main class	org.myorg.WordCount	
Arguments	input/* output/wordcount_output2	
Java options		
Capture output	<input type="checkbox"/>	
Prepare	Add delete	Add mkdir

Submit the workflow

- Click Done, follow by Save
- Then click submit

The screenshot shows the Hue Oozie Editor interface. The top navigation bar includes links for Hue, Home, Query Editors, Data Browsers, Workflows, Search, File Browser, Job Browser, Cloudera, and Help. The main area is titled "WordCount" and contains a "java" action. On the left, a sidebar titled "ACTIONS" has a "Submit" button highlighted with a red arrow. At the bottom, there are "Save" and "Back" buttons.

ACTIONS

- Submit

WordCount

java

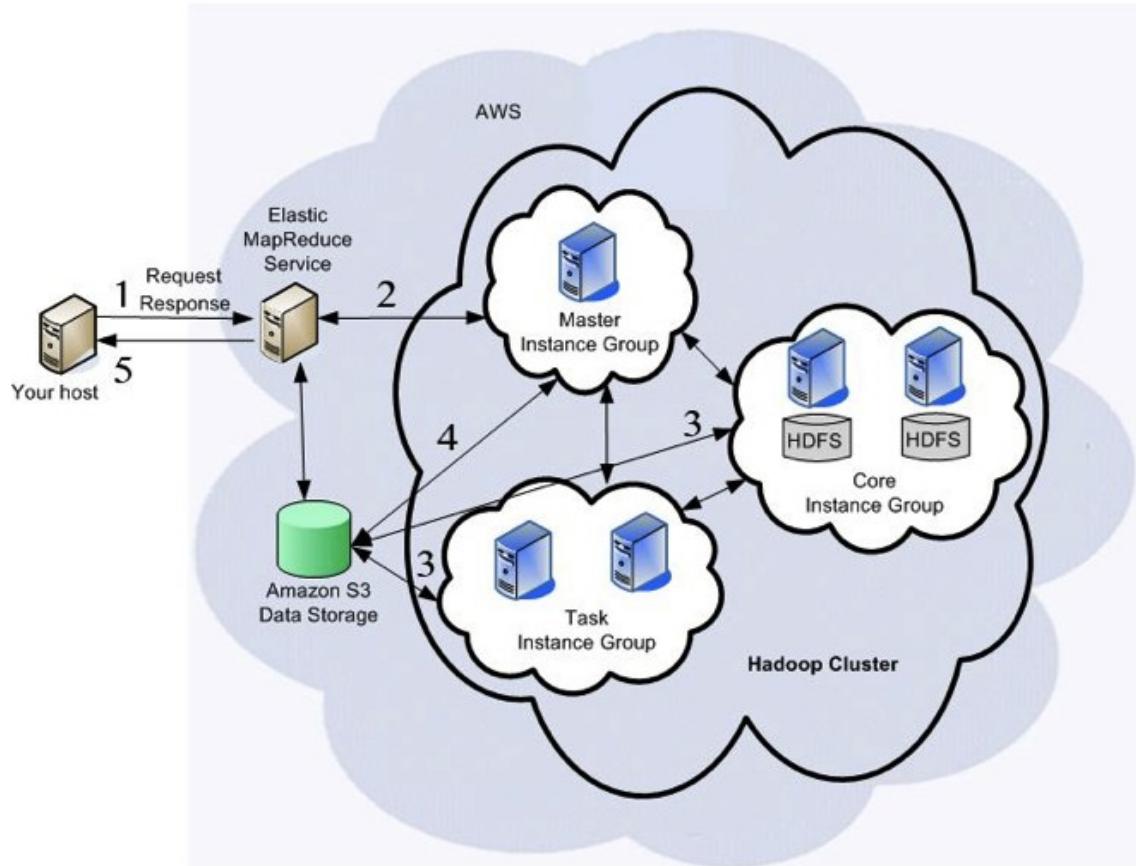
end

Save Back

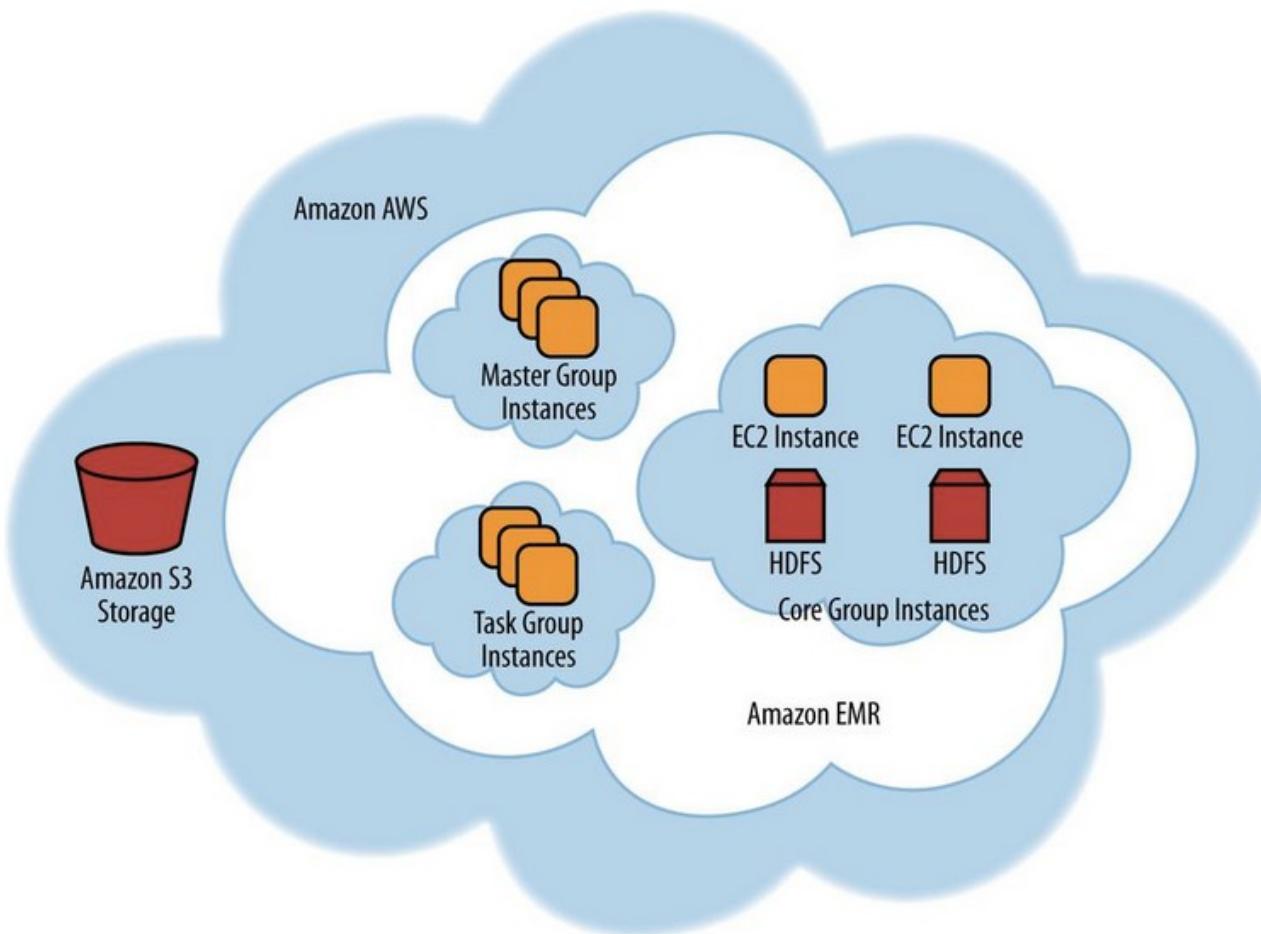


Hands-On: Running WordCount.jar on Amazon EMR

Architecture Overview of Amazon EMR



Amazon EMR Cluster



Creating an AWS account

The screenshot shows the official AWS website homepage. At the top, the Amazon Web Services logo is on the left, followed by a "Sign Up" button, "My Account / Console" dropdown, and "English" dropdown. Below the header is a navigation bar with "AWS Products & Solutions" dropdown, search bar, "Entire Site" dropdown, and "Developer" and "Support" dropdowns. The main content area features three stylized cloud icons. To the left, text promotes the "AWS Free Tier" and "Includes Windows". Below this, it says "Receive 750 hours per month of Amazon EC2 Microsoft Windows Server Micro instance usage." A blue link "Try Windows Server on AWS »" is present. To the right, the Windows Server 2012 logo is shown next to the text "Windows Server 2012". At the bottom, there are two buttons: "Get Started for Free »" and "Launch virtual machines and apps in minutes."

AWS Free Tier
Includes Windows

Receive 750 hours per month of Amazon EC2 Microsoft Windows Server Micro instance usage.

Try Windows Server on AWS »

Get Started for Free » Launch virtual machines and apps in minutes.

Windows Server 2012

Signing up for the necessary services

- Simple Storage Service (S3)
- Elastic Compute Cloud (EC2)
- Elastic MapReduce (EMR)

Caution! This costs real money!

Creating Amazon S3 bucket

Services | Edit

Welcome to Amazon Simple Storage Service

Amazon S3 is storage for the Internet. It is designed to make web-scale computing easier for developers.

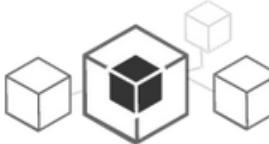
Amazon S3 provides a simple web services interface that can be used to store and retrieve any amount of data, at any time, from anywhere on the web. It gives any developer access to the same highly scalable, reliable, secure, fast, inexpensive infrastructure that Amazon uses to run its own global network of web sites. The service aims to maximize benefits of scale and to pass those benefits on to developers.

You can read, write, and delete objects ranging in size from 1 byte to 5 terabytes each. The number of objects you can store is unlimited. Each object is stored in a bucket with a unique key that you assign.

Get started by simply creating a bucket and uploading a test object, for example a photo or .txt file.

[Create Bucket](#)

S3 at a glance

Create	Add	Manage
		
Create a bucket in one of several	Upload objects to your bucket. Amazon	Manage your data with Amazon S3's

Additional Information

[Getting Started Guide](#)

[Documentation](#)

[All S3 Resources](#)

[Forums](#)

Create access key using Security Credentials in the AWS Management Console

Access Credentials

There are three types of access credentials used to authenticate your requests to AWS services: (a) access keys, (b) X.509 certificates, and (c) key pairs. Each access credential type is explained below.

 Access Keys  X.509 Certificates  Key Pairs

Use access keys to make secure REST or Query protocol requests to any AWS service API. We create one for you when your account is created — see your access key below.

Your Access Keys

Created	Access Key ID	Secret Access Key	Status
July 14, 2013	[REDACTED]	Show	Active (Make Inactive)

[Create a new Access Key](#)

For your protection, you should never share your secret access keys with anyone. In addition, industry best practice recommends frequent key rotation.

 [Learn more about Access Keys](#)

Access Credentials

There are three types of access credentials used to authenticate your requests to AWS services: (a) access keys, (b) X.509 certificates, and (c) key pairs. Each access credential type is explained below.

 **Access Keys**  **X.509 Certificates**  **Key Pairs**

Use access keys to make secure REST or Query protocol requests to any AWS service API. We create one for you when your account is created — see your access key below.

Your Access Keys

Created	Access Key ID	Secret Access Key	Status
July 14, 2013	[REDACTED]	Show	Active (Make Inactive)

[Create a new Access Key](#)

For your protection, you should never share your secret access keys with anyone. In addition, industry best practice recommends frequent key rotation.

 [Learn more about Access Keys](#)

Creating a cluster in EMR

Services ▾ Edit ▾ IMC Institute ▾ Oregon ▾ Help ▾

Elastic MapReduce ▾ Cluster List EMR Help

Create cluster View details Clone Terminate

Filter:		All clusters	Filter clusters ...	9 clusters (all loaded)			
		Name	ID	Status	Creation time (UTC+7) ▾	Elapsed time	Normalized instance hours
<input type="checkbox"/>	▶	WordCount Demo	j-1CIQF5578M3EE	Terminated User request	2014-04-22 10:57	8 minutes	12
<input type="checkbox"/>	▶	Word count	j-2B9L4MS0BL5YO	Terminated All steps completed	2014-04-21 15:50	5 minutes	20
<input type="checkbox"/>	▶	WordCount IMC Demo	j-R3CGBLS7FL0	Terminated	2014-04-10 11:54	5 minutes	5
<input type="checkbox"/>	▶	CloudFront HTTP Log Analyzer (Custom JAR)	j-L222ZNOHV535	Terminated All steps completed	2014-04-09 16:21	17 minutes	20
<input type="checkbox"/>	▶	Word count	j-2BKCWQBXCE6MW	Terminated User request	2014-03-18 13:36	49 minutes	28

Services ▾ Edit ▾ IMC Institute ▾ Oregon ▾ Help ▾

Elastic MapReduce ▾ Create Cluster EMR Help

Cluster Configuration

Configure sample application

Cluster name IMC Institute Demo Cluster

Termination protection Yes No Prevents accidental termination of the cluster: to shut down the cluster, you must turn off termination protection. [Learn more](#)

Logging Enabled Copy the cluster's log files automatically to S3. [Learn more](#)

Log folder S3 location
s3:// 
s3://<bucket-name>/<folder>/

Debugging Enabled Index logs to enable console debugging functionality (requires logging). [Learn more](#)

Choose configure sample application

Configure Sample Application X

i Select a sample application to auto-populate the Create Cluster page

Select sample application ▼

Output location

Logging Enabled

[Cancel](#) Ok

Select create cluster

The screenshot shows the AWS Elastic MapReduce Cluster Details page. At the top, there's a navigation bar with 'Services' and 'Edit' dropdowns, and a search bar on the right. Below that is a breadcrumb trail: 'Elastic MapReduce' > 'Cluster List' > 'Cluster Details'. A row of buttons includes 'Add step', 'Resize', 'Clone', and 'Terminate'. The main content area displays a cluster named 'Word count' which is currently 'Starting'. It provides details like Master public DNS (---), Tags (--- View All / Edit), and a summary table.

Summary	Configuration Details	Security/Network
ID: j-2NR0M10Q07214 Creation date: 2014-06-05 08:31 (UTC+7) Elapsed time: 1 second Auto-terminate: Yes Termination On Change protection:	AMI version: 2.4.2 Hadoop Amazon 1.0.3 distribution: Applications: --- Log URI: ---	Availability --- zone: Subnet ID: subnet-cf510ca7 Key name: --- EC2 role: --- Visible to all None Change users:

Hardware

Master: Provisioning 1 m1.small
Core: Provisioning 2 m1.small
Task: ---

▶ Monitoring

Services ▾ | Edit ▾ | IMC Institute ▾ | Oregon ▾ | Help ▾

Elastic MapReduce ▾ | Cluster List | EMR Help

Create cluster | View details | Clone | Terminate

Filter: All clusters ▾ | Filter clusters ... | 10 clusters (all loaded)

	Name	ID	Status	Creation time (UTC+7) ▾	Elapsed time	Normalized instance hours
<input type="checkbox"/>	Word count	j-2NR0M10Q07214	Starting	2014-06-05 08:31	1 minute	0

View Result from the S3 bucket

All Buckets / imcbucket / wordcount / output / 2013-07-14

	Name	Storage Class	Size	Last Modified
<input type="checkbox"/>	_SUCCESS	Standard	0 bytes	Sun Jul 14 19:01:25 GMT+700 2013
<input type="checkbox"/>	part-00000	Standard	97.3 KB	Sun Jul 14 19:01:13 GMT+700 2013
<input type="checkbox"/>	part-00001	Standard	98.6 KB	Sun Jul 14 19:01:11 GMT+700 2013
<input type="checkbox"/>	part-00002	Standard	97.1 KB	Sun Jul 14 19:01:25 GMT+700 2013

Hands-On: Working with a csv data

A sample CSV data

- The input data is access logs with the following form

Date, Requesting-IP-Address

- We will write a map reduce program to count the number of hits to the website per country.

```
2013-09-01,211.139.190.234
2013-09-01,114.221.140.56
2013-09-01,211.155.113.221
2013-09-01,211.155.113.221
2013-09-01,221.4.143.9
2013-09-01,121.12.149.106
2013-09-01,114.221.140.56
2013-09-01,121.12.149.106
2013-09-01,211.139.190.234
2013-09-01,221.4.143.9
2013-09-01,211.155.113.221
2013-09-01,221.4.143.9
2013-09-01,211.139.190.234
2013-09-01,61.171.134.87
2013-09-01,218.249.47.126
2013-09-01,61.171.134.87
2013-09-01,106.120.176.93
2013-09-01,221.4.143.9
2013-09-01,118.194.193.18
2013-09-01,221.4.143.9
2013-09-01,221.4.143.9
2013-09-01,221.4.143.9
```

HitsByCountryMapper.java

```
package learning.bigdata.mapreduce;

import java.io.IOException;

import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;

public class HitsByCountryMapper extends Mapper<LongWritable, Text, Text, IntWritable> {

    private final static String[] COUNTRIES = { "India", "UK", "US", "China" };
    private Text outputKey = new Text();
    private IntWritable outputValue = new IntWritable();

    @Override
    protected void setup(Context context) throws IOException, InterruptedException {
        super.setup(context);
    }

    @Override
    public void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException {

        try {
            String valueString = value.toString();

            // Split the value string to get Date and ipAddress
            String[] row = valueString.split(",");

```

HitsByCountryMapper.java

```
// row[0]= Date and row[1]=ipAddress
String ipAddress = row[1];

// Get the country name to which the ipAddress belongs
String countryName = getCountryNameFromIpAddress(ipAddress);
outputKey.set(countryName);
outputValue.set(1);
context.write(outputKey, outputValue);

} catch (ArrayIndexOutOfBoundsException ex) {
    context.getCounter("Custom counters", "MAPPER_EXCEPTION_COUNTER").increment(1);
    ex.printStackTrace();
}
}

private static String getCountryNameFromIpAddress(String ipAddress) {

    if (ipAddress != null && !ipAddress.isEmpty()) {

        int randomIndex = Math.abs(ipAddress.hashCode()) % COUNTRIES.length;
        return COUNTRIES[randomIndex];
    }

    return null;
}
```

HitsByCountryReducer.java

```
package learning.bigdata.mapreduce;

import java.io.IOException;
import java.util.Iterator;

import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;

public class HitsByCountryReducer extends Reducer<Text, IntWritable, Text, IntWritable> {

    private Text outputKey = new Text();
    private IntWritable outputValue = new IntWritable();
    private int count = 0;

    protected void reduce(Text key, Iterable<IntWritable> values, Context context) throws IOException,
        InterruptedException {

        count = 0;
        Iterator<IntWritable> iterator = values.iterator();
        while (iterator.hasNext()) {
            IntWritable value = iterator.next();
            count += value.get();
        }
        outputKey.set(key);
        outputValue.set(count);
        context.write(outputKey, outputValue);
    }
}
```

HitsByCountry.java

```
package learning.bigdata.main;

import learning.bigdata.mapreduce.HitsByCountryMapper;
import learning.bigdata.mapreduce.HitsByCountryReducer;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;
import org.apache.hadoop.util.Tool;
import org.apache.hadoop.util.ToolRunner;

public class HitsByCountry extends Configured implements Tool {

    private static final String JOB_NAME = "Calculating hits by country";

    public static void main(String[] args) throws Exception {

        if (args.length < 2) {
            System.out.println("Usage: HitsByCountry <comma separated input directories> <output dir>");
            System.exit(-1);
        }
        int result = ToolRunner.run(new HitsByCountry(), args);
        System.exit(result);
    }
}
```

HitsByCountry.java

```
@Override
public int run(String[] args) throws Exception {

    try {
        Configuration conf = getConf();
        Job job = Job.getInstance(conf);

        job.setJarByClass(HitsByCountry.class);
        job.setJobName(JOB_NAME);

        job.setMapperClass(HitsByCountryMapper.class);
        job.setMapOutputKeyClass(Text.class);
        job.setMapOutputValueClass(IntWritable.class);

        job.setReducerClass(HitsByCountryReducer.class);
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(IntWritable.class);

        job.setInputFormatClass(TextInputFormat.class);
        job.setOutputFormatClass(TextOutputFormat.class);

        FileInputFormat.setInputPaths(job, args[0]);
        FileOutputFormat.setOutputPath(job, new Path(args[1]));

        boolean success = job.waitForCompletion(true);
        return success ? 0 : 1;
    } catch (Exception e) {
        e.printStackTrace();
        return 1;
    }
}
```

Exercise: Write a Map Reduce program to count number of vowels in an input file.

Lecture: Developing Complex Hadoop MapReduce Applications

Choosing appropriate Hadoop data types

- Hadoop uses the *Writable* interface based classes as the data types for the MapReduce computations.
- Choosing the appropriate *Writable* data types for your input, intermediate, and output data can have a large effect on the performance and the programmability of your MapReduce programs.
- In order to be used as a *value* data type, a data type must implement the *org.apache.hadoop.io.Writable* interface.
- In order to be used as a *key* data type, a data type must implement the *org.apache.hadoop.io.WritableComparable<T>* interface

Examples

Specify the data types for the input (key: LongWritable, value: Text) and output (key: Text, value: IntWritable) key-value pairs of your mapper using the generic-type variables.

```
public class SampleMapper extends Mapper<LongWritable, Text, Text,  
IntWritable> {  
  
    public void map(LongWritable key, Text value,  
        Context context) ... {  
    .....  
    }  
}
```

Specify the data types for the input (key: Text, value: IntWritable) and output (key: Text, value: IntWritable) key-value pairs of your reducer using the generic-type variables. The reducer's input key-value pair data types should match the mapper's output key-value pairs.

```
public class Reduce extends Reducer<Text, IntWritable, Text,  
IntWritable> {  
  
    public void reduce(Text key,  
        Iterable<IntWritable> values, Context context) {  
    .....  
    }  
}
```

Hadoop built-in data types

- *Text*: This stores a UTF8 text
- *BytesWritable*: This stores a sequence of bytes
- *VIntWritable* and *VLongWritable*: These store variable length integer and long values
- *NullWritable*: This is a zero-length Writable type that can be used when you don't want to use a key or value type

Hadoop built-in data types

- The following Hadoop build-in collection data types can only be used as *value* types.
 - *ArrayWritable*: This stores an array of values belonging to a *Writable* type.
 - *TwoDArrayWritable*: This stores a matrix of values belonging to the same *Writable* type.
 - *MapWritable*: This stores a map of key-value pairs. Keys and values should be of the *Writable* data types.
 - *SortedMapWritable*: This stores a sorted map of key-value pairs. Keys should implement the *WritableComparable* interface.

Implementing a custom Hadoop Writable data type

- we can easily write a custom *Writable* data type by implementing the *org.apache.hadoop.io.Writable* interface
- The *Writable* interface-based types can be used as *value* types in Hadoop MapReduce computations.

Examples

Write a new LogWritable class implementing the org.apache.hadoop.io.Writable interface.

```
public class LogWritable implements Writable{  
  
    private Text userIP, timestamp, request;  
    private IntWritable responseSize, status;  
  
    public LogWritable() {  
        this.userIP = new Text();  
        this.timestamp= new Text();  
        this.request = new Text();  
        this.responseSize = new IntWritable();  
        this.status = new IntWritable();  
    }  
    public void readFields(DataInput in) throws IOException {  
        userIP.readFields(in);  
        timestamp.readFields(in);  
        request.readFields(in);  
    }  
}
```

Examples

```
    responseSize.readFields(in);
    status.readFields(in);
}

public void write(DataOutput out) throws IOException {
    userIP.write(out);
    timestamp.write(out);
    request.write(out);
    responseSize.write(out);
    status.write(out);
}

..... // getters and setters for the fields
}
```

Examples

Use the new `LogWritable` type as a value type in your MapReduce computation. In the following example, we use the `LogWritable` type as the Map output value type.

```
public class LogProcessorMap extends Mapper<LongWritable,  
Text, Text, LogWritable> {  
...  
}  
  
public class LogProcessorReduce extends Reducer<Text,  
LogWritable, Text, IntWritable> {  
  
    public void reduce(Text key,  
    Iterable<LogWritable> values, Context context) {  
        .... }  
}
```

Choosing a suitable Hadoop InputFormat for your input data format

- Hadoop supports processing of many different formats and types of data through *InputFormat*.
- The *InputFormat* of a Hadoop MapReduce computation generates the key-value pair inputs for the mappers by parsing the input data.
- *InputFormat* also performs the splitting of the input data into logical partitions

***InputFormat* that Hadoop provide**

- *TextInputFormat*: This is used for plain text files.
TextInputFormat generates a key-value record for each line of the input text files.
- *NLineInputFormat*: This is used for plain text files.
NlineInputFormat splits the input files into logical splits of fixed number of lines.
- *SequenceFileInputFormat*: For Hadoop Sequence file input data
- *DBInputFormat*: This supports reading the input data for MapReduce computation from a SQL table.

Implementing new input data formats

- Hadoop enables us to implement and specify custom *InputFormat* implementations for our MapReduce computations.
- A *InputFormat* implementation should extend the `org.apache.hadoop.mapreduce.InputFormat<K, V>` abstract class
- overriding the *createRecordReader()* and *getSplits()* methods.

Formatting the results of MapReduce computations – using Hadoop OutputFormats

- it is important to store the result of a MapReduce computation in a format that can be consumed efficiently by the target application
- We can use Hadoop *OutputFormat* interface to define the data storage format
- A *OutputFormat* prepares the output location and provides a *RecordWriter* implementation to perform the actual serialization and storage of the data.
- Hadoop uses the `org.apache.hadoop.mapreduce.lib.output.TextOutputFormat<K,V>` as the default *OutputFormat*

Hands-On: Analytics Using MapReduce

Three Analytic MapReduce Examples

- 1. Simple analytics using MapReduce**
- 2. Performing Group-By using MapReduce**
- 3. Calculating frequency distributions and sorting using MapReduce**

Preparing Example Data

NASA weblog dataset available from

<http://ita.ee.lbl.gov/html/contrib/NASA-HTTP.html>

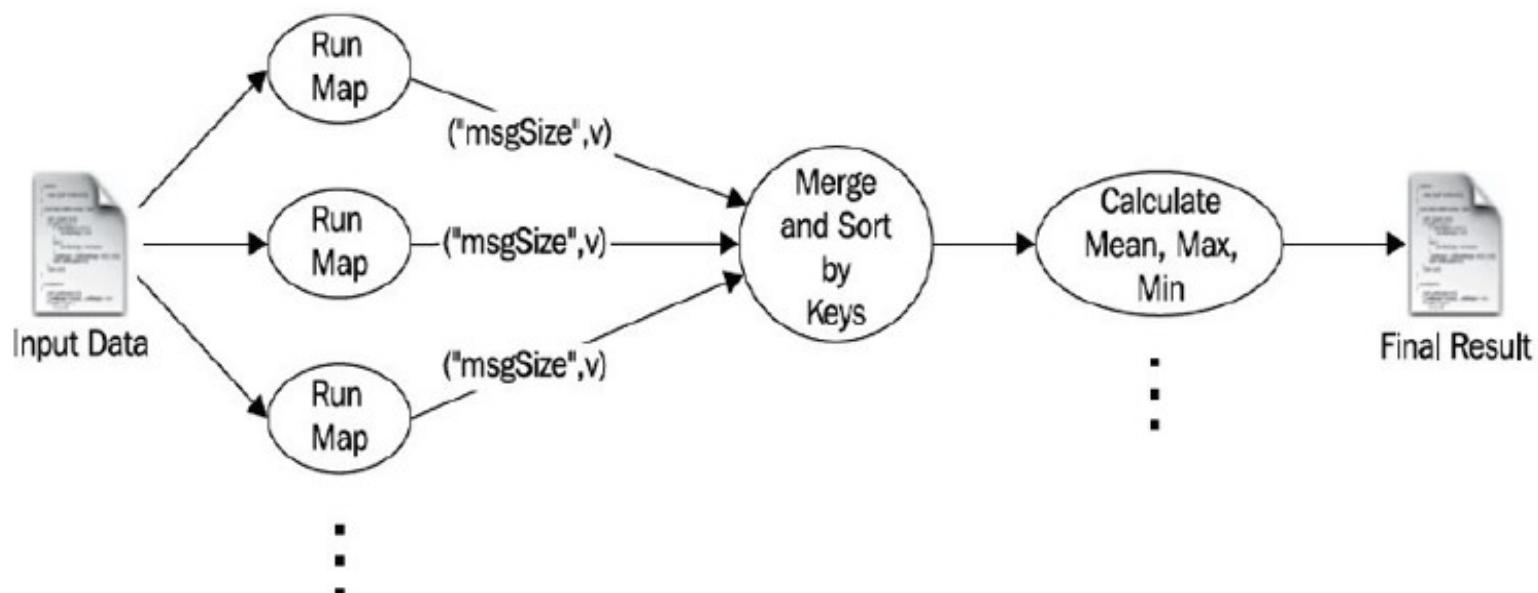
is a real-life dataset collected using the requests received by NASA web servers.

Download the weblog dataset from *ftp://ita.ee.lbl.gov/traces/NASA_access_log_Jul95.gz* and unzip it. We call the extracted folder as DATA_DIR.

```
$ hadoopdfs -mkdir /data  
$ hadoopdfs -put <DATA_DIR>/NASA_access_log_Jul95 /data/input1
```

Simple analytics using MapReduce

Aggregative values (for example, Mean, Max, Min, standard deviation, and so on) provide the basic analytics about a dataset..



Source: Hadoop MapReduce CookBook

WebLogMessageSizeAggregator.java

```
package analysis;

import java.io.IOException;
import java.util.*;
import java.util.regex.Matcher;
import java.util.regex.Pattern;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapred.*;
import org.apache.hadoop.util.*;
import org.apache.hadoop.conf.*;

public class WebLogMessageSizeAggregator {

    public static final Pattern httplogPattern = Pattern
        .compile("([^\s]+) - - \[\.(+)\]\ \\"([^\s]+) ([^\s]*) HTTP/[^\s]+\""
        "[^\s]+ ([0-9]+)");

    public static class AMapper extends MapReduceBase implements Mapper<LongWritable,
    Text, Text, IntWritable> {
```

WebLogMessageSizeAggregator.java

```
public void map(LongWritable key, Text value, OutputCollector<Text, IntWritable> output, Reporter reporter) throws IOException {
    Matcher matcher = httplogPattern.matcher(value.toString());
    if (matcher.matches()) {
        int size = Integer.parseInt(matcher.group(5));
        output.collect(new Text("msgSize"), new IntWritable(size));
    }
}
```

WebLogMessageSizeAggregator.java

```
public static class AReducer extends MapReduceBase implements Reducer<Text, IntWritable, Text, IntWritable> {

    public void reduce(Text key, Iterator<IntWritable> values,
OutputCollector<Text, IntWritable> output,Reporter reporter) throws IOException {

        double tot = 0;
        int count = 0;
        int min = Integer.MAX_VALUE;
        int max = 0;

        while (values.hasNext()) {
            int value = values.next().get();
            tot = tot + value;
            count++;
            if (value < min) {
                min = value;
            }
            if (value > max) {
                max = value;
            }
        }
    }
}
```

WebLogMessageSizeAggregator.java

```
        output.collect(new Text("Mean"), new IntWritable((int) tot / count));
        output.collect(new Text("Max"), new IntWritable(max));
        output.collect(new Text("Min"), new IntWritable(min));
    }
}

public static void main(String[] args) throws Exception {
    JobConf job = new JobConf(WebLogMessageSizeAggregator.class);
    job.setJarByClass(WebLogMessageSizeAggregator.class);
    job.setMapperClass(AMapper.class);
    job.setReducerClass(AReducer.class);
    job.setMapOutputKeyClass(Text.class);
    job.setMapOutputValueClass(IntWritable.class);
    FileInputFormat.setInputPaths(job, new Path(args[0]));
    FileOutputFormat.setOutputPath(job, new Path(args[1]));
    JobClient.runJob(job);
}
```

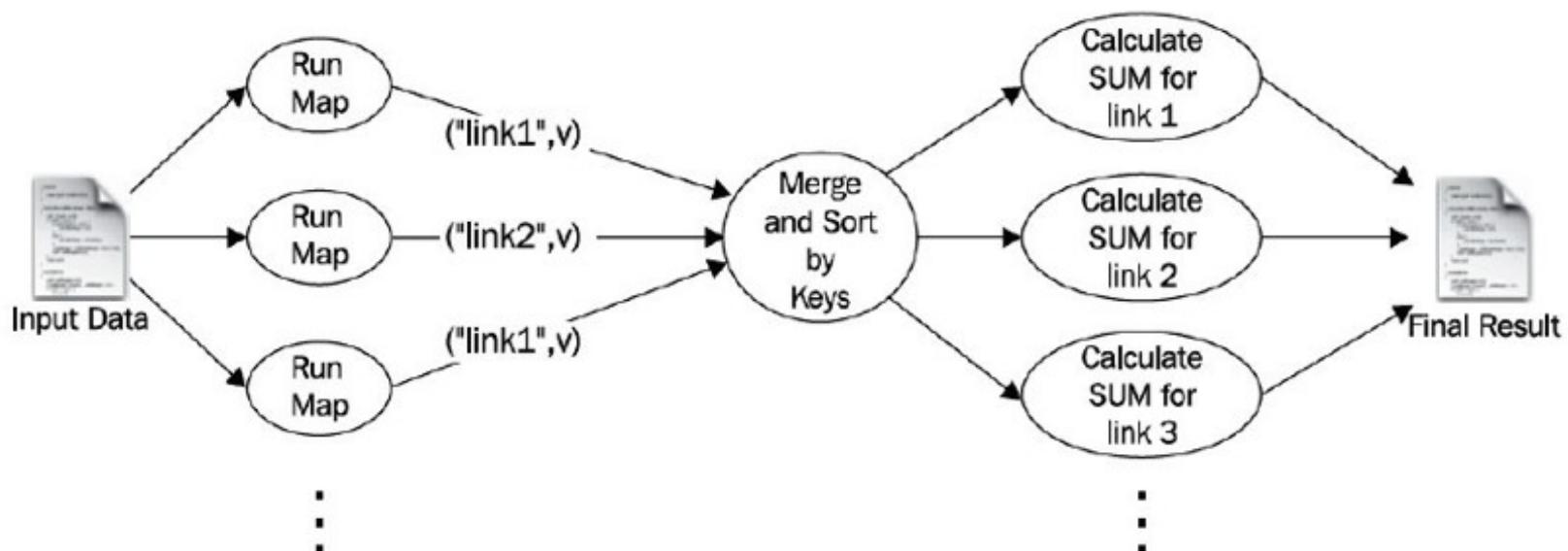
Compile, Build JAR, Submit Job, Review Result

```
$ cd /home/hduser
$ javac -classpath /usr/local/hadoop/hadoop-core-0.20.205.0.jar -d WebLog WebLogMessageSizeAggregator.java
$ jar -cvf ./weblog.jar -C WebLog .
$ hadoop jar ./weblog.jar analysis.WebLogMessageSizeAggregator /data/* /output/result_weblog
Output:
.....
$ hadoop dfs -cat /output/result_weblog/part-00000
```

Mean	1150
Max	6823936
Min	0

Performing Group-By using MapReduce

A MapReduce to group data into simple groups and calculate the analytics for each group.



Source: Hadoop MapReduce CookBook

WeblogHitsByLinkProcessor.java

```
public class WeblogHitsByLinkProcessor {  
  
    public static final Pattern httplogPattern = Pattern  
        .compile("([^\s]+) - - \[(.+)\] \"([^\s]+) ([^\s]*) HTTP/[^\s]+\\"  
        [^\s]+ ([0-9]+)";  
  
    public static class AMapper extends MapReduceBase implements Mapper<LongWritable,  
    Text, Text, IntWritable> {  
  
        private final static IntWritable one = new IntWritable(1);  
        private Text word = new Text();  
  
        public void map(LongWritable key, Text value, OutputCollector<Text,  
        IntWritable> output, Reporter reporter) throws IOException {  
            Matcher matcher = httplogPattern.matcher(value.toString());  
            if (matcher.matches()) {  
                String linkUrl = matcher.group(4);  
                word.set(linkUrl);  
                output.collect(word, one);  
            }  
        }  
    }  
}
```

WeblogHitsByLinkProcessor.java

```
public static class AReducer extends MapReduceBase implements Reducer<Text,  
IntWritable, Text, IntWritable> {  
  
    private IntWritable result = new IntWritable();  
  
    public void reduce(Text key, Iterator<IntWritable> values,  
OutputCollector<Text, IntWritable> output,Reporter reporter) throws IOException {  
  
        int sum = 0;  
        while (values.hasNext()) {  
            sum += values.next().get();  
        }  
        result.set(sum);  
        output.collect(key, result);  
    }  
}
```

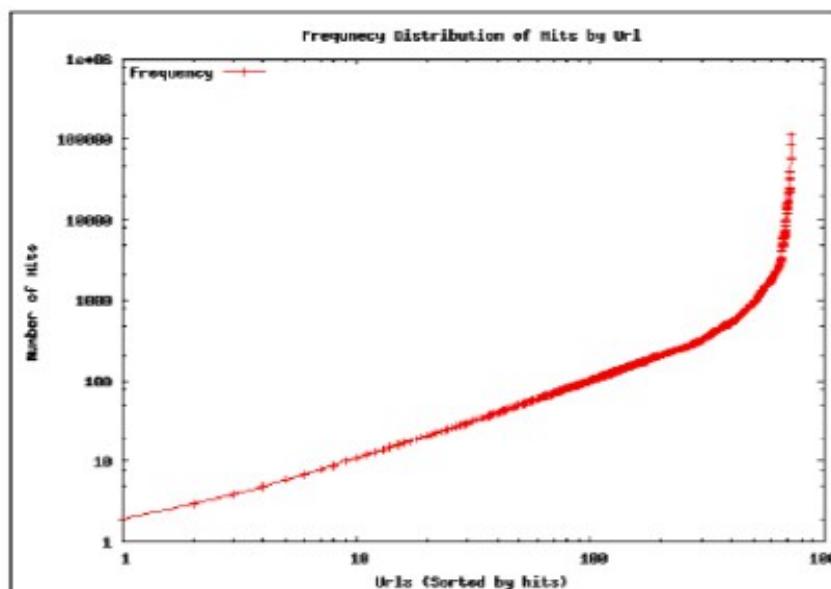
Compile, Build JAR, Submit Job, Review Result

```
$ cd /home/hduser
$ javac -classpath /usr/local/hadoop/hadoop-core-0.20.205.0.jar -d WebLogHit WeblogHitsByLinkProcessor.java
$ jar -cvf ./webloghit.jar -C WebLogHit .
$ hadoop jar ./webloghit.jar analysis.WeblogHitsByLinkProcessor /data/* /output/result_webloghit
Output:
.....
$ hadoop dfs -cat /output/result_webloghit/part-00000
```

```
/      32667
/67Edowns/home.html    2
/67Edowns/launchup.gif 2
/67edowns/home.html   3
./     3
/.ksc.html       6
//     3
//biomed/climate/gif/f16pcfinmed.gif   1
//biomed/gif/   1
//biomed/gif/aerial.gif 1
//elv/bakgro.gif    2
//elv/elvhead2.gif   1
//elv/elvhead3.gif   1
//elv/elvpage.htm   2
//elv/endball.gif   1
//elv/vidpicp.htm   1
//elv/whnew.htm  1
```

Calculating frequency distributions and sorting using MapReduce

Frequency distribution is the number of hits received by each URL sorted in the Ascending order, by the number hits received by a URL. We have already calculated the number of hits in the previous program.



Source: Hadoop MapReduce CookBook

Lecture

Understanding Hive



Introduction

A Petabyte Scale Data Warehouse Using Hadoop



Hive is developed by Facebook, designed to enable easy data summarization, ad-hoc querying and analysis of large volumes of data. It provides a simple query language called Hive QL, which is based on SQL

What Hive is NOT

Hive is **not designed for online transaction processing** and does not offer real-time queries and row level updates. It is best used for batch jobs over large sets of immutable data (like web logs, etc.).

Hive Metastore

- Store Hive metadata
- Configurations
 - Embedded: in-process metastore, in-process database
 - Local: in-process metastore, out-of-process database
 - Remote: out-of-process metastore,out-of-process database

Hive Schema-On-Read

- Faster loads into the database (simply copy or move)
- Slower queries
- Flexibility – multiple schemas for the same data

HiveQL

- Hive Query Language
- SQL dialect
- No support for:
 - UPDATE, DELETE
 - Transactions
 - Indexes
 - HAVING clause in SELECT
 - Updateable or materialized views
 - Stored procedure

Hive Tables

- Managed- CREATE TABLE
 - LOAD- File moved into Hive's data warehouse directory
 - DROP- Both data and metadata are deleted.
- External- CREATE EXTERNAL TABLE
 - LOAD- No file moved
 - DROP- Only metadata deleted
 - Use when sharing data between Hive and Hadoop applications or you want to use multiple schema on the same data

Running Hive

Hive Shell

- **Interactive**

hive

- **Script**

hive -f myscript

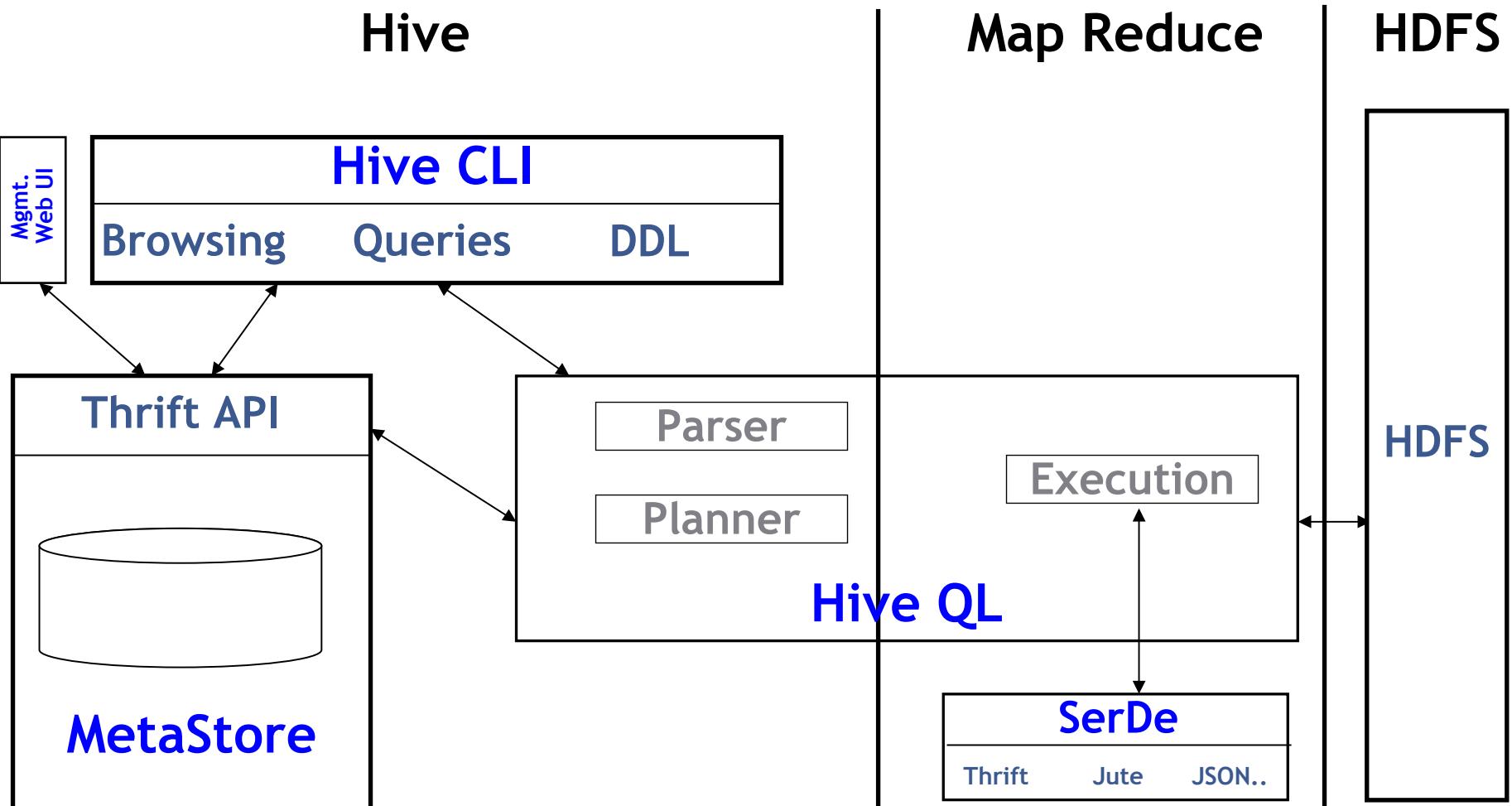
- **Inline**

*hive -e 'SELECT * FROM mytable'*

System Architecture and Components

- **Metastore:** To store the meta data.
- **Query compiler and execution engine:** To convert SQL queries to a sequence of map/reduce jobs that are then executed on Hadoop.
- **SerDe and ObjectInspectors:** Programmable interfaces and implementations of common data formats and types.
A SerDe is a combination of a Serializer and a Deserializer (hence, Ser-De). The Deserializer interface takes a string or binary representation of a record, and translates it into a Java object that Hive can manipulate. The Serializer, however, will take a Java object that Hive has been working with, and turn it into something that Hive can write to HDFS or another supported system.
- **UDF and UDAF:** Programmable interfaces and implementations for user defined functions (scalar and aggregate functions).
- **Clients:** Command line client similar to Mysql command line.

Architecture Overview



Sample HiveQL

The Query compiler uses the information stored in the metastore to convert SQL queries into a sequence of map/reduce jobs, e.g. the following query

```
SELECT * FROM t where t.c = 'xyz'
```

```
SELECT t1.c2 FROM t1 JOIN t2 ON (t1.c1 = t2.c1)
```

```
SELECT t1.c1, count(1) from t1 group by t1.c1
```

Hands-On: Creating Table and Retrieving Data using Hive

Running Hive from terminal

Starting Hive

```
[cloudera@localhost ~]$ hive
Logging initialized using configuration in jar:file:/usr/lib/hive/lib/hive-common-0.10.0-cdh4.4.0.jar!/hive-log4j.properties
Hive history file=/tmp/cloudera/hive_job_log_23b573ac-35ea-432e-93f3-b699630503ec_1889776701.txt
hive> █
```

Quit from Hive

```
hive> quit;
```

Starting Hive Editor from Hue

The screenshot shows the Hue Query Editor interface. At the top, there is a navigation bar with icons for Home, Clusters, Jobs, Saved Queries, History, and Settings. A user profile for 'cloudera' is also present. Below the navigation bar, the main area is titled 'Query Editor'. On the left side, there is a sidebar with the following sections and buttons:

- DATABASE**: A dropdown menu set to 'default'.
- SETTINGS**: An 'Add' button.
- FILE RESOURCES**: An 'Add' button.
- USER-DEFINED FUNCTIONS**: An 'Add' button.
- PARAMETERIZATION**: A checked checkbox labeled 'Enable Parameterization'.
- EMAIL NOTIFICATION**: An unchecked checkbox labeled 'Email me on completion.'

In the center, there is a large text input field with the placeholder text: 'Example: SELECT * FROM tablename, or press CTRL + space'. At the bottom of this field, there is a note: 'or create a'. Below the text input, there are several buttons: 'Execute' (highlighted in blue), 'Save as...', 'Explain', and 'New query'.

Scroll Down
the web page

Creating Hive Table

```
hive (default)> CREATE TABLE test_tbl(id INT, country STRING) ROW
FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE;
OK

Time taken: 4.069 seconds

hive (default)> show tables;
OK

test_tbl

Time taken: 0.138 seconds

hive (default)> describe test_tbl;
OK

id    int
country  string

Time taken: 0.147 seconds

hive (default)>
```

See also: <https://cwiki.apache.org/Hive/languagemanual-ddl.html>

Using Hue Query Editor

Query Editor

```
1 CREATE TABLE test_tbl (id INT, country STRING) ROW FORMAT DELIMITED
2 FIELDS TERMINATED BY ',' STORED AS TEXTFILE;
```

ExecuteSave as...Explain

or create a

New query

See also: <https://cwiki.apache.org/Hive/languagemanual-ddl.html>

Using Hue Query Editor

My Queries

Search for query

View result Edit Copy View trash New query

Usage history Move to trash

Recent Saved Queries 0 Recent Run Queries 3

Time	Name	Query	State
06/16/14 20:21:47	My saved query	SHOW TABLES;	available
06/16/14 20:20:10	My saved query	CREATE TABLE test_tbl (id INT, country STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORE...	available
06/16/14 20:13:56	My saved query	Example: SELECT * FROM tablename, or press CTRL + space	failed

See also: <https://cwiki.apache.org/Hive/languagemanual-ddl.html>

Using Hue Query Editor

Query Editor My Queries Saved Queries History Settings

Query Results: Unsaved Query

DOWNLOADS

Download as CSV
Download as XLS
Save

MR JOBS

No Hadoop jobs were launched in running this query.

Results Query Log Columns

	tab_name
0	test_tbl

Did you know? ×

- If the result contains a large number of columns, click a row to select a column to jump to

See also: <https://cwiki.apache.org/Hive/languagemanual-ddl.html>

Reviewing Hive Table in HDFS

The screenshot shows the Cloudera File Browser interface. At the top, there is a toolbar with various icons: Home, File, Database, Cluster, Job, User, Help, and a search bar. To the right of the search bar is a dropdown menu for 'cloudera'.

File Browser

Search for file name Rename Move Copy Change permissions Download

Move to trash ▼

Home / user / hive / warehouse View trash

Type	Name	Size	User	Group	Permissions	Date
..	.		hive	hive	drwxrwxrwx	June 16, 2014 08:20 PM
..	..		hive	hive	drwxrwxrwx	October 07, 2013 08:20 AM
Table	test_tbl		cloudera	hive	drwxrwxrwx	June 16, 2014 08:20 PM

Alter and Drop Hive Table

```
hive (default)> alter table test_tbl add columns (remarks STRING);  
  
hive (default)> describe test_tbl;  
OK  
id    int  
country  string  
remarks  string  
Time taken: 0.077 seconds  
hive (default)> drop table test_tbl;  
OK  
Time taken: 0.9 seconds
```

See also: <https://cwiki.apache.org/Hive/adminmanual-metastoreadmin.html>

Loading Data to Hive Table

Creating Hive table

```
$ hive  
  
hive (default)> CREATE TABLE test_tbl(id INT, country STRING) ROW  
FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE;
```

Loading data to Hive table

```
hive (default)> LOAD DATA LOCAL INPATH '/tmp/country.csv' INTO TABLE test_tbl;  
  
Copying data from file:/tmp/test_tbl_data.csv  
  
Copying file: file:/tmp/test_tbl_data.csv  
  
Loading data to table default.test_tbl  
  
OK  
  
Time taken: 0.241 seconds  
  
hive (default)>
```

Querying Data from Hive Table

```
hive (default)> select * from test_tbl;  
OK  
1 USA  
62 Indonesia  
63 Philippines  
65 Singapore  
66 Thailand  
Time taken: 0.287 seconds  
hive (default)>
```

Insert Overwriting the Hive Table

```
hive (default)> LOAD DATA LOCAL INPATH
'/home/cloudera/Downloads/test_tbl_data_updated.csv' overwrite INTO
TABLE test_tbl;

Copying data from file:/tmp/test_tbl_data_updated.csv
Copying file: file:/tmp/test_tbl_data_updated.csv
Loading data to table default.test_tbl
Deleted hdfs://localhost:54310/user/hive/warehouse/test_tbl
OK
Time taken: 0.204 seconds
hive (default)>
```

MovieLens

<http://grouplens.org/datasets/movielens/>



[about](#) [datasets](#) [publications](#) [blog](#)

MovieLens

GroupLens Research has collected and made available rating data sets from the MovieLens web site (<http://movielens.org>). The data sets were collected over various periods of time, depending on the size of the set. Before using these data sets, please review their README files for the usage licenses and other details.

MovieLens 100k

100,000 ratings from 1000 users on 1700 movies.

- [README.txt](#)
- [ml-100k.zip](#)
- [Index of unzipped files](#)

MovieLens 1M

1 million ratings from 6000 users on 4000 movies.

Datasets

[MovieLens](#)

[HetRec 2011](#)

[WikiLens](#)

[Book-Crossing](#)

[Jester](#)

[EachMovie](#)

Create the Hive Table for movielens

```
hive (default)> CREATE TABLE u_data (
    userid INT,
    movieid INT,
    rating INT,
    unixtime STRING)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY '\t'
STORED AS TEXTFILE;

hive (default)> LOAD DATA LOCAL INPATH
'/home/cloudera/Downloads/u.data' overwrite INTO TABLE u_data;
```

Create the Hive Table for Apache Log

```
hive (default)> CREATE TABLE apachelog (
    host STRING,
    identity STRING,
    user STRING,
    time STRING,
    request STRING,
    status STRING,
    size STRING,
    referer STRING,
    agent STRING)
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.RegexSerDe'
WITH SERDEPROPERTIES (
    "input.regex" = "([^\"]*) ([^\"]*) ([^\"]*) (-|\\\[^\\"\\]*\\])"
    "([^\\"\\]*|\"[^\\\"]*\\\") (-|[0-9]*) (-|[0-9]*) (?: ([^\\"\\]*|\".*\\")"
    "([^\\"\\]*|\".*\\\"))?"
)
STORED AS TEXTFILE;
```



Lecture

Understanding Pig

Introduction

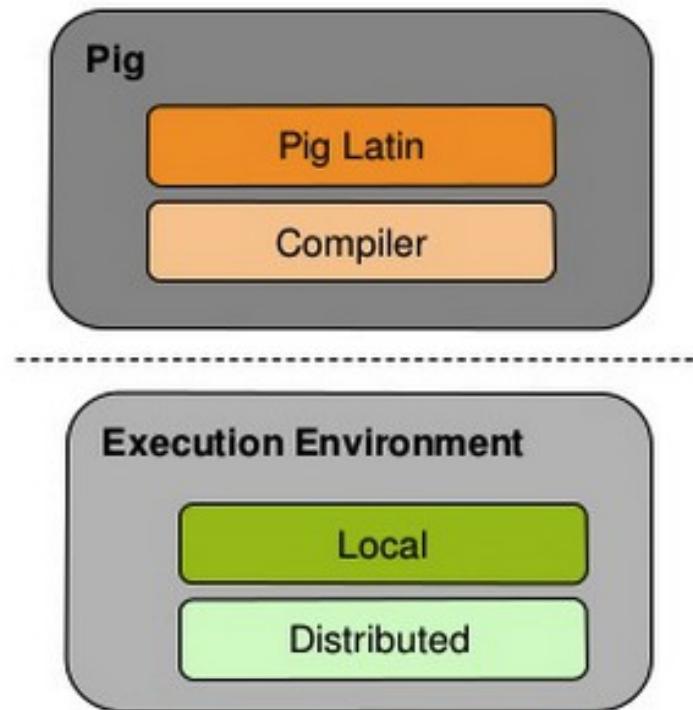
A high-level platform for creating MapReduce programs Using Hadoop



Pig is a platform for analyzing large data sets that consists of a high-level language for expressing data analysis programs, coupled with infrastructure for evaluating these programs. The salient property of Pig programs is that their structure is amenable to substantial parallelization, which in turns enables them to handle very large data sets.

Pig Components

- **Two Components**
 - Language (Pig Latin)
 - Compiler
- **Two Execution Environments**
 - Local
 - pig -x local*
 - **Distributed**
 - pig -x mapreduce*



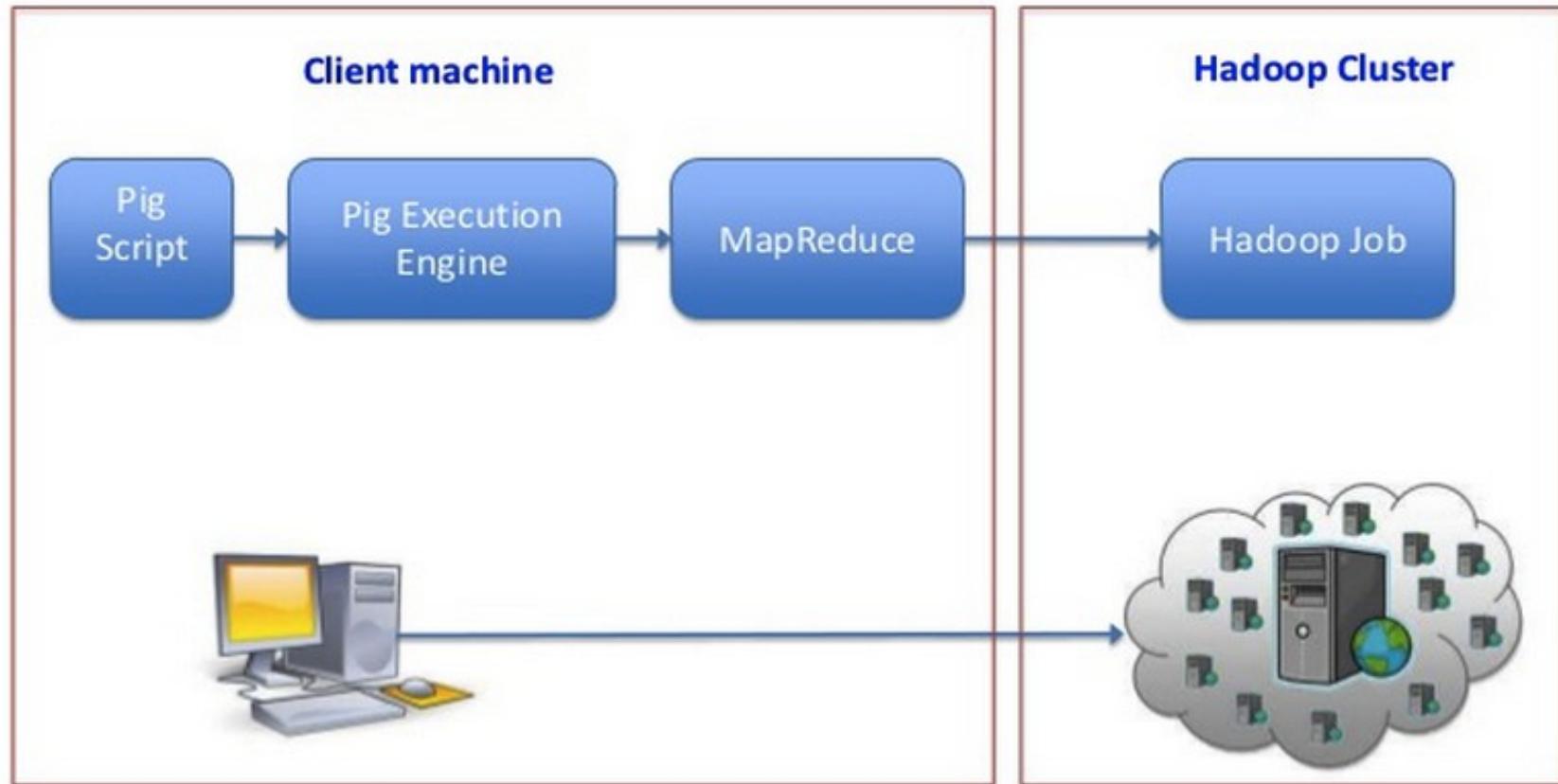
Running Pig

- **Script**
pig myscript
- **Command line (Grunt)**
pig
- **Embedded**
Writing a java program

Pig Latin

```
Users = load 'users' as (name, age);
Fltrd = filter Users by
    age >= 18 and age <= 25;
Pages = load 'pages' as (user, url);
Jnd = joinFltrdby name, Pages by user;
Grpd = group Jnd by url;
Smmd = foreach Grpd generate group,
COUNT(Jnd) as clicks;
Srtd = order Smmd by clicks desc;
Top5 = limit Srtd 5;
store Top5 into 'top5sites';
```

Pig Execution Stages



Source Introduction to Apache Hadoop-Pig: PrashantKommireddi

Hive.apache.org

Why Pig?

- **Makes writing Hadoop jobs easier**
 - 5% of the code, 5% of the time
 - You don't need to be a programmer to write Pig scripts
- **Provide major functionality required for DatawareHouse and Analytics**
 - *Load, Filter, Join, Group By, Order, Transform*
 - **User can write custom UDFs (User Defined Function)**

Running MapReduce Job Using Oozie : Select Java

Edit Node: WordCountDemo

You can parameterize values using case sensitive `#{parameter}`.

Jar name	/user/WordCount.jar
Main class	org.myorg.WordCount
Arguments	input/* output/wordcount_output_oozie
Java options	
Capture output	<input type="checkbox"/>



<i>Characteristic</i>	<i>Pig</i>	<i>Hive</i>
Developed by	Yahoo!	Facebook
Language name	Pig Latin	HiveQL
Type of language	Data flow	Declarative (SQL dialect)
Data structures it operates on	Complex, nested	
Schema optional?	Yes	No, but data can have many schemas
Relational complete?	Yes	Yes
Turing complete?	Yes when extended with Java UDFs	Yes when extended with Java UDFs

Hands-On: Running a Pig script

Starting Pig Command Line

```
[hdadmin@localhost ~]$ pig -x local
2013-08-01 10:29:00,027 [main] INFO org.apache.pig.Main - Apache Pig
version 0.11.1 (r1459641) compiled Mar 22 2013, 02:13:53
2013-08-01 10:29:00,027 [main] INFO org.apache.pig.Main - Logging error
messages to: /home/hdadmin/pig_1375327740024.log
2013-08-01 10:29:00,066 [main] INFO org.apache.pig.impl.util.Utils -
Default bootup file /home/hdadmin/.pigbootup not found
2013-08-01 10:29:00,212 [main] INFO
org.apache.pig.backend.hadoop.executionengine.HExecutionEngine - Connecting
to hadoop file system at: file:///
grunt>
```

Starting Pig from Hue

The screenshot shows the Hue web interface for managing Hadoop data. At the top, there's a navigation bar with icons for Home, Cloudera Manager, and various Hadoop services like HDFS, MapReduce, and YARN. Below the bar, a sub-navigation menu shows 'Editor' is selected, along with 'Scripts' and 'Dashboard'. On the left, a sidebar titled 'EDITOR' contains links for 'Pig' (which is currently selected), 'Properties', 'Save', 'RUN', 'Submit', 'Logs', 'FILE', 'Script', and a help icon. The main content area is titled 'Unsaved script' and contains a single line of Pig Latin code: '1 ie. A = LOAD '/user/cloudera/data';'. There's also a large blue play button icon.

Writing a Pig Script

```
#Preparing Data
```

```
Download hdi-data.csv
```

```
#Edit Your Script
```

```
[hdadmin@localhost ~]$ cd Downloads/
```

```
[hdadmin@localhost ~]$ vi countryFilter.pig
```

countryFilter.pig

```
A = load 'hdi-data.csv' using PigStorage(',') AS (id:int, country:chararray, hdi:float,  
lifeex:int, mysch:i  
nt, eysch:int, gni:int);  
B = FILTER A BY gni > 2000;  
C = ORDER B BY gni;  
dump C;
```

Running a Pig Script

```
[hdadmin@localhost ~]$ cd Downloads  
[hdadmin@localhost ~]$ pig -x local  
grunt > run countryFilter.pig
```

....

```
(150,Cameroon,0.482,51,5,10,2031)  
(126,Kyrgyzstan,0.615,67,9,12,2036)  
(156,Nigeria,0.459,51,5,8,2069)  
(154,Yemen,0.462,65,2,8,2213)  
(138,Lao People's Democratic Republic,0.524,67,4,9,2242)  
(153,Papua New Guinea,0.466,62,4,5,2271)  
(165,Djibouti,0.43,57,3,5,2335)  
(129,Nicaragua,0.589,74,5,10,2430)  
(145,Pakistan,0.504,65,4,6,2550)
```



Lecture: Understanding Sqoop

Introduction

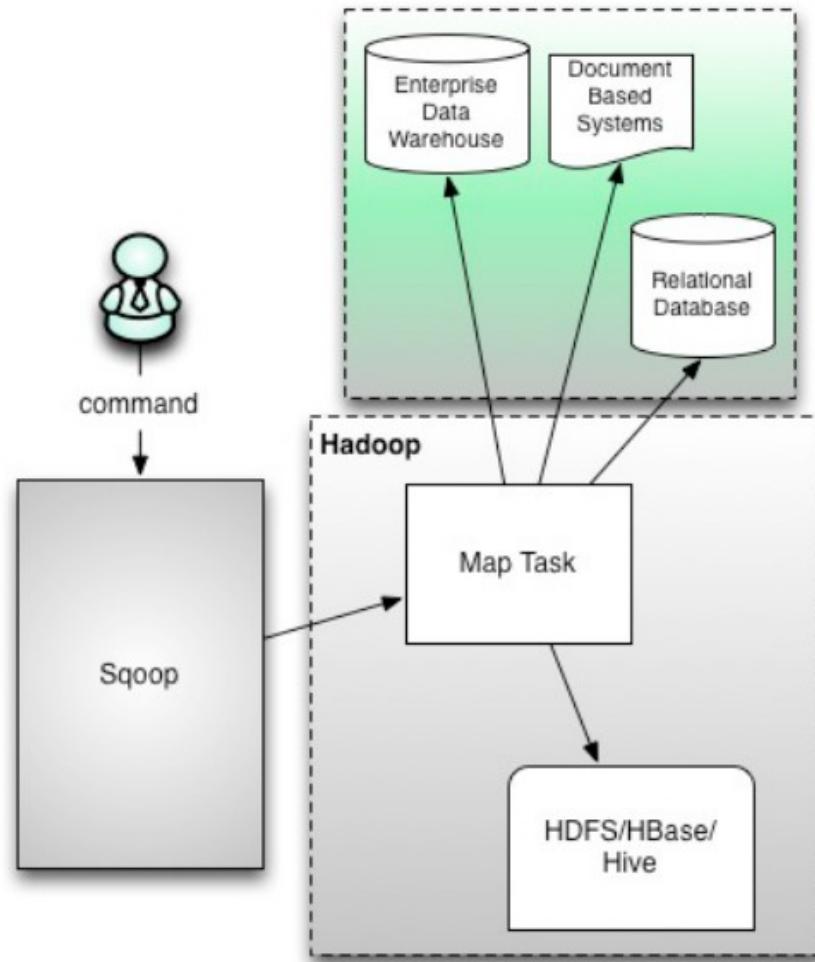


Sqoop (“SQL-to-Hadoop”) is a straightforward command-line tool with the following capabilities:

- **Imports individual tables or entire databases to files in HDFS**
- **Generates Java classes to allow you to interact with your imported data**
- **Provides the ability to import from SQL databases straight into your Hive data warehouse**

See also: <http://sqoop.apache.org/docs/1.4.2/SqoopUserGuide.html>

Architecture Overview



Hive.apache.org

Hands-On: Loading Data from DBMS to Hadoop HDFS

Loading Data into MySQL DB

```
[root@localhost ~]# service mysqld start
Starting mysqld: [ OK ]
[root@localhost ~]# mysql -u root -p
Password: cloudera

mysql> create database countrydb;
Query OK, 1 row affected (0.00 sec)

mysql> use countrydb;
Database changed

mysql> create table country_tbl(id INT, country VARCHAR(100));
Query OK, 0 rows affected (0.02 sec)

mysql> LOAD DATA LOCAL INFILE '/home/cloudera/Downloads/country.csv' INTO
TABLE country_tbl FIELDS terminated by ',' LINES TERMINATED BY '\n';
Query OK, 237 rows affected, 4 warnings (0.00 sec)

Records: 237 Deleted: 0 Skipped: 0 Warnings: 0
```

Loading Data into MySQL DB

Testing data query from MySQL DB

```
mysql> select * from country_tbl;
+----+-----+
| id | country |
+----+-----+
| 93 | Afghanistan |
| 355 | Albania |
| 213 | Algeria |
| 1684 | AmericanSamoa |
| 376 | Andorra |
...
+----+-----+
237 rows in set (0.00 sec)

mysql>
```

Importing data from MySQL to Hive Table

```
[hdadmin@localhost ~]$ sqoop import --connect
jdbc:mysql://localhost/countrydb --username root -P --table
country_tbl --hive-import --hive-table country_tbl -m 1

Warning: /usr/lib/hbase does not exist! HBase imports will fail.

Please set $HBASE_HOME to the root of your HBase installation.

Warning: $HADOOP_HOME is deprecated.
```

Enter password: <enter here>

```
13/03/21 18:07:43 INFO tool.BaseSqoopTool: Using Hive-specific delimiters for output. You can override
13/03/21 18:07:43 INFO tool.BaseSqoopTool: delimiters with --fields-terminated-by, etc
13/03/21 18:07:43 INFO manager.MySQLManager: Preparing to use a MySQL streaming resultset
13/03/21 18:07:43 INFO tool.CodeGenTool: Beginning code generation
13/03/21 18:07:44 INFO manager.SqlManager: Executing SQL statement: SELECT 1.* FROM `country_tbl` AS t LIMIT 1
13/03/21 18:07:44 INFO manager.SqlManager: Executing SQL statement: SELECT 1.* FROM `country_tbl` AS t LIMIT 1
13/03/21 18:07:44 INFO core.CompilationManager: HADOOP_HOME is /usr/local/hadoop/lineage...
Note: /tmp/sqoop-hdadmin/compile/04d002bf293e103fbedf3338215/country_tbl.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
13/03/21 18:07:44 INFO core.CompilationManager: Writing jar file: /tmp/sqoop-hdadmin/compile/04d002bf293e103fbedf3338215/country_tbl.jar
13/03/21 18:07:44 WARN manager.MySQLManager: It looks like you are importing from mysql...
13/03/21 18:07:44 WARN manager.MySQLManager: This transfer can be faster! Use the --direct
13/03/21 18:07:44 WARN manager.MySQLManager: option to exercise a MySQL-specific fast path.
13/03/21 18:07:44 INFO manager.MySQLManager: Setting exec CANTONNE behavior to convertNull (sqoop)
13/03/21 18:07:44 INFO mapreduce.ImportJobBase: Beginning import of country_tbl
13/03/21 18:07:45 INFO mapred.JobClient: Running job: job_021303221744_0001
13/03/21 18:07:44 INFO mapred.JobClient: map 0% reduce 0%
13/03/21 18:08:02 INFO mapred.JobClient: map 100% reduce 0%
13/03/21 18:08:07 INFO mapred.JobClient: Job complete: job_021303221744_0001
13/03/21 18:08:07 INFO mapred.JobClient: Counters: 10
13/03/21 18:08:07 INFO mapred.JobClient: Job Counter
13/03/21 18:08:07 INFO mapred.JobClient:   HDFS_BYTES_READ=2114
13/03/21 18:08:07 INFO mapred.JobClient:   Total time spent by all reduces waiting after receiving splits (ms)=0
13/03/21 18:08:07 INFO mapred.JobClient:
```

Reviewing data from Hive Table

```
[hdadmin@localhost ~]$ hive
Logging initialized using configuration in file:/usr/local/hive-0.9.0-
bin/conf/hive-log4j.properties

Hive history
file=/tmp/hdadmin/hive_job_log_hdadmin_201303211810_964909984.txt

hive (default)> show tables;
OK

country_tbl
test_tbl

Time taken: 2.566 seconds

hive (default)> select * from country_tbl;
OK

93  Afghanistan
355  Albania
.....
Time taken: 0.587 seconds

hive (default)> quit;
[hdadmin@localhost ~]$
```

Reviewing HDFS Database Table files

Start Web Browser to <http://localhost:50070/> then navigate to /user/hive/warehouse

The screenshot shows a Mozilla Firefox window titled "HDFS:/user/hive/warehouse - Mozilla Firefox". The address bar displays "HDFS:/user/hive/warehouse". The main content area shows the "Contents of directory /user/hive/warehouse". A "Goto : /user/hive/warehouse" input field with a "go" button is present. Below it is a link "Go to parent directory". A table lists two entries:

Name	Type	Size	Replication	Block Size	Modification Time	Permission	Owner	Group
country_tbl	dir				2013-03-21 18:08	rwxr-xr-x	hdadmin	supergroup
test_tbl	dir				2013-03-17 18:25	rwxr-xr-x	hdadmin	supergroup

At the bottom, there is a link "Go back to DFS home". The taskbar at the bottom of the browser window shows "hdadmin@localhost:~" and "tmp - File Browser".



Lecture

Understanding HBase

Introduction

An open source, non-relational, distributed database



HBase is an open source, non-relational, distributed database modeled after Google's BigTable and is written in Java. It is developed as part of Apache Software Foundation's Apache Hadoop project and runs on top of HDFS (, providing BigTable-like capabilities for Hadoop. That is, it provides a fault-tolerant way of storing large quantities of sparse data.

HBase Features

- Hadoop database modelled after Google's Bigtable
- Column oriented data store, known as Hadoop Database
- Support random realtime CRUD operations (unlike HDFS)
- No SQL Database
- Opensource, written in Java
- Run on a cluster of commodity hardware

When to use Hbase?

- When you need high volume data to be stored
- Un-structured data
- Sparse data
- Column-oriented data
- Versioned data (same data template, captured at various time, time-elapse data)
- When you need high scalability

Which one to use?

- HDFS
 - Only append dataset (no random write)
 - Read the whole dataset (no random read)
- HBase
 - Need random write and/or read
 - Has thousands of operation per second on TB+ of data
- RDBMS
 - Data fits on one big node
 - Need full transaction support
 - Need real-time query capabilities

HBase vs. RDBMS

	HBase	RDBMS
Hardware architecture	Similar to Hadoop. Clustered commodity hardware. Very affordable.	Typically large scalable multiprocessor systems. Very expensive.
Fault Tolerance	Built into the architecture. Lots of nodes means each is relatively insignificant. No need to worry about individual node downtime.	Requires configuration of the HW and the RDBMS with the appropriate high availability options.
Typical Database Size	Terabytes to Petabytes - hundred of millions to billions of rows.	Gigabytes to Terabytes – hundred of thousands to millions of rows.
Data Layout	A sparse, distributed, persistent, multidimensional sorted map.	Rows or column oriented.
Data Types	Bytes only.	Rich data type support.
Transactions	ACID support on a single row only	Full ACID compliance across rows and tables
Query Language	API primitive commands only, unless combined with Hive or other technology	SQL
Indexes	Row-Key only unless combined with other technologies such as Hive or IBM's BigSQL	Yes
Throughput	Millions of queries per second	Thousands of queries per second

- Given this RDBMS:

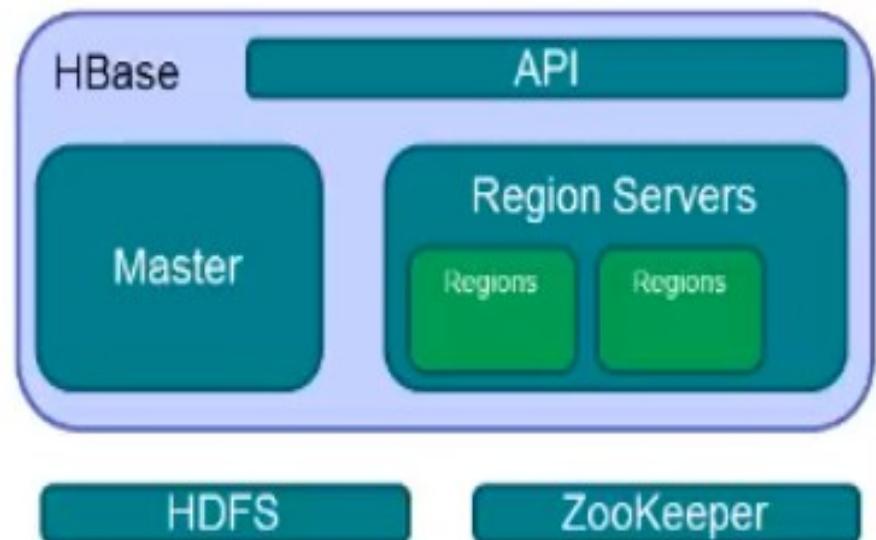
ID (Primary key)	Last name	First name	Password	Timestamp
1234	Smith	John	Hello, world!	20130710
5678	Cooper	Joyce	wysiwyg	20120825
5678	Cooper	Joyce	wisiwig	20130916

- Logical view in HBase:

Row-Key	Value (CF, Qualifier, Version)
1234	info {'lastName': 'Smith', 'firstName': 'John'} pwd {'password': 'Hello, world!'}
5678	info {'lastName': 'Cooper', 'firstName': 'Joyce'} pwd {'password': 'wysiwyg'@ts 20130916, 'password': 'wisiwig'@ts 20120825}

HBase Components

- Region
 - Row of table are stores
- Region Server
 - Hosts the tables
- Master
 - Coordinating the Region Servers
- ZooKeeper
- HDFS
- API
 - The Java Client API



HBase Shell Commands

- See the list of the tables

```
list
```

- Create a table:

```
create 'testTable', 'cf'
```

- Insert data into a table:

Insert at rowA, column "cf:columnName" with a value of "val1"

```
put 'testTable', 'rowA', 'cf:columnName', 'val1'
```

- Retrieve data from a table:

Retrive "rowA" from the table "testTable"

```
get 'testTable', 'rowA'
```

- Iterate through a table:

```
- scan 'testTable'
```

- Delete a table:

```
enable 'testTable'  
drop 'testTable'
```

Hands-On: Running HBase

Starting HBase shell

```
[hdadmin@localhost ~]$ start-hbase.sh
starting master, logging to /usr/local/hbase-0.94.10/logs/hbase-hdadmin-
master-localhost.localdomain.out

[hdadmin@localhost ~]$ jps
3064 TaskTracker
2836 SecondaryNameNode
2588 NameNode
3513 Jps
3327 HMaster
2938 JobTracker
2707 DataNode

[hdadmin@localhost ~]$ hbase shell
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 0.94.10, r1504995, Fri Jul 19 20:24:16 UTC 2013

hbase(main):001:0>
```

Create a table and insert data in HBase

```
hbase(main):009:0> create 'test', 'cf'
0 row(s) in 1.0830 seconds

hbase(main):010:0> put 'test', 'row1', 'cf:a', 'val1'
0 row(s) in 0.0750 seconds

hbase(main):011:0> scan 'test'

ROW                                COLUMN+CELL
row1                               column=cf:a, timestamp=1375363287644,
value=val1

1 row(s) in 0.0640 seconds

hbase(main):002:0> get 'test', 'row1'

COLUMN                                CELL
cf:a                                 timestamp=1375363287644, value=val1

1 row(s) in 0.0370 seconds
```

Using Data Browsers in Hue for HBase

The screenshot shows the Hue - HBase Browser interface. At the top, there's a navigation bar with tabs for File, Edit, View, History, Bookmarks, Tools, and Help. Below the navigation bar is a toolbar with icons for session restore, Hue - HBase Browser, and a plus sign for adding new tabs. The main browser area has a URL bar showing 'quickstart.cloudera:8888/hbase/#Cluster'. The page title is 'Hue - HBase Browser'. The main content area is titled 'Home - Cluster' and contains a search bar for 'Table Name' and buttons for 'Enable', 'Disable', and 'Drop'. A 'New Table' button is also present. Below the search bar, there's a table listing tables: 'Table Name' (with a checkbox) and 'testdb' (with a checked checkbox). To the right of the table, there's a column labeled 'Enabled' with a checked checkbox next to 'testdb'.

Using Data Browsers in Hue for HBase

The screenshot shows the Hue HBase Browser interface. At the top, there is a navigation bar with links to Cloudera, Hue, Hadoop, HBase, Impala, Spark, Solr, Oozie, and Cloudera Manager. Below the navigation bar is a toolbar with icons for Home, Query Editors, Data Browsers, Workflows, Search, File Browser, Job Browser, and cloudera. The main area is titled "HBase Browser". A search bar contains the query: "row_key, row_prefix* +scan_len [col1, family:col2, fam3:, col_prefix]". To the right of the search bar are buttons for "Filter Columns/Families", "All" (checked), and "Sort By ASC". The data view displays two rows of data:

row1	cf: a	cf: id
	62334	1234

thana	cf: id
	1234

At the bottom left, a message says "Fetched 10 entries starting from null in 0.995 seconds." On the bottom right are buttons for "Drop Rows", "Bulk Upload", and "New Row".

Using Data Browsers in Hue for HBase

The screenshot shows the Hue HBase Browser interface. At the top, there is a navigation bar with links to Cloudera, Hue, Hadoop, HBase, Impala, Spark, Solr, Oozie, and Cloudera Manager. Below the navigation bar is a toolbar with icons for Home, Query Editors, Data Browsers, Workflows, Search, File Browser, Job Browser, and cloudera. The main area is titled "HBase Browser". A search bar contains the query: "row_key, row_prefix* +scan_len [col1, family:col2, fam3:, col_prefix]". To the right of the search bar are buttons for "Filter Columns/Families", "All" (checked), and "Sort By ASC". The data view displays two rows of data:

row1	cf: a	cf: id
	62334	1234

thana	cf: id
	1234

At the bottom left, a message says "Fetched 10 entries starting from null in 0.995 seconds." On the bottom right are buttons for "Drop Rows", "Bulk Upload", and "New Row".

Project: Flight

Flight Details Data

<http://stat-computing.org/dataexpo/2009/the-data.html>



ASA Sections on:

[Statistical Computing](#)
[Statistical Graphics](#)

[[Computing, Graphics](#)]

[[Awards, Data expo, Video library](#)]

[[Events, News, Newsletter](#)]

[Data expo '09](#)

Get the data

The data comes originally from [RITA](#) where it is [described in detail](#). You can download the data there, or from the bzipped csv files listed below. These files have derivable variables removed, are packaged in yearly chunks and have been more heavily compressed than the originals.

Download individual years:

[1987](#), [1988](#), [1989](#), [1990](#), [1991](#), [1992](#), [1993](#), [1994](#), [1995](#), [1996](#), [1997](#), [1998](#), [1999](#), [2000](#), [2001](#),
[2002](#), [2003](#), [2004](#), [2005](#), [2006](#), [2007](#), [2008](#)

Data expo 09

- [Posters & results](#)
- [Competition description](#)
- [Download the data](#)
- [Supplemental data sources](#)
- [Using a database](#)
- [Intro to command line tools](#)

Data Description

Name	Description
1 Year	1987-2008
2 Month	1-12
3 DayofMonth	1-31
4 DayOfWeek	1 (Monday) - 7 (Sunday)
5 DepTime	actual departure time (local, hhmm)
6 CRSDepTime	scheduled departure time (local, hhmm)
7 ArrTime	actual arrival time (local, hhmm)
8 CRSArrTime	scheduled arrival time (local, hhmm)
9 UniqueCarrier	<u>unique carrier code</u>
10 FlightNum	flight number
11 TailNum	plane tail number
12 ActualElapsedTime	in minutes
13 CRSElapsedTime	in minutes
14 AirTime	in minutes
15 ArrDelay	arrival delay, in minutes
16 DepDelay	departure delay, in minutes
17 Origin	origin <u>IATA airport code</u>
18 Dest	destination <u>IATA airport code</u>
19 Distance	in miles
20 TaxiIn	taxi in time, in minutes
21 TaxiOut	taxi out time in minutes
22 Cancelled	was the flight cancelled?
23 CancellationCode	reason for cancellation (A = carrier, B = weather, C = NAS, D = security)
24 Diverted	1 = yes, 0 = no
25 CarrierDelay	in minutes
26 WeatherDelay	in minutes
27 NASDelay	in minutes
28 SecurityDelay	in minutes
29 LateAircraftDelay	in minutes

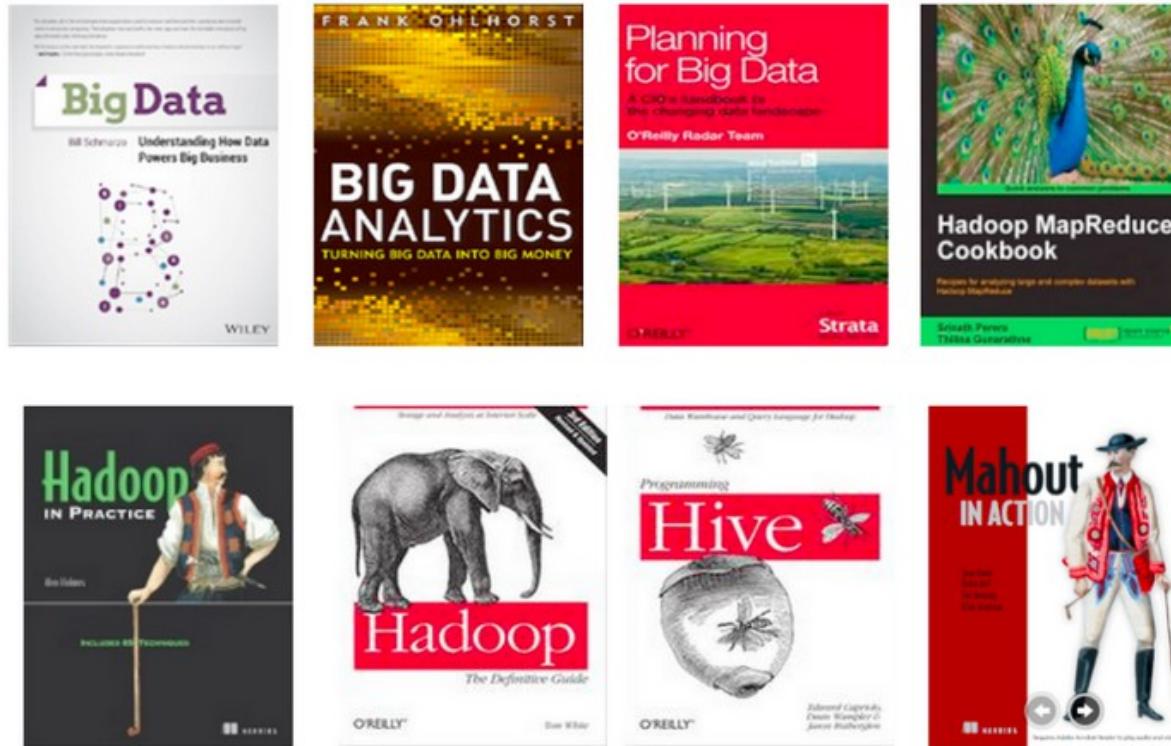
Snapshot of Dataset

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	
1	Year	Month	DayofMo	DayOfWe	DepTime	CRSDepTl	ArrTime	CRSArrTl	UniqueCa	FlightNum	TailNum	ActualElap	CRSElapse	AirTime	ArrDelay	DepDelay	Origin	Dest	Distance	TaxiIn	TaxiOut
2	2008	1	5	6	2243	1415	45	1625	WN	1684	N347SW	62	70	41	500	508	SAN	PHX	304	2	
3	2008	1	5	6	1940	1220	2111	1350	WN	1684	N347SW	91	90	64	441	440	SFO	SAN	447	5	
4	2008	1	7	1	111	1845	308	2045	WN	405	N644SW	117	120	103	383	386	MDW	JAN	666	4	
5	2008	1	7	1	2213	1700	2317	1655	WN	1827	N759GS	124	55	75	382	313	IND	MDW	162	10	
6	2008	1	7	1	2143	1720	26	1820	WN	1430	N644SW	163	60	83	366	263	STL	MDW	251	24	
7	2008	1	7	1	117	2020	302	2135	WN	490	N651SW	105	75	87	327	297	STL	TUL	351	5	
8	2008	1	7	1	2358	1855	105	2000	WN	490	N651SW	67	65	50	305	303	MDW	STL	251	4	
9	2008	1	3	4	2245	1730	2354	1850	WN	186	N792SW	69	80	59	304	315	JAN	HOU	359	3	
10	2008	1	7	1	2219	1730	35	1935	WN	2474	N710SW	76	65	67	300	289	MDW	CMH	284	2	
11	2008	1	5	6	2129	1620	2246	1750	WN	1924	N408WN	77	90	56	296	309	SFO	LAS	414	4	
12	2008	1	3	4	1615	1130	1623	1135	WN	10	N617SW	68	65	56	288	285	MAF	ABQ	332	4	
13	2008	1	3	4	1736	1305	2031	1555	WN	1837	N761RR	255	290	268	276	271	MDW	SFO	1855	4	
14	2008	1	5	6	2236	1805	2400	1930	WN	646	N283WN	84	85	71	270	271	LAX	SFO	337	6	
15	2008	1	3	4	2021	1700	2303	1835	WN	2005	N302SW	162	95	73	268	201	LAS	SFO	414	4	
16	2008	1	3	4	2059	1620	2216	1750	WN	1924	N761RR	77	90	60	266	279	SFO	LAS	414	6	
17	2008	1	7	1	2348	2105	307	2250	WN	3137	N358SW	259	165	244	257	163	MCO	MDW	989	1	
18	2008	1	3	4	2255	1820	509	55	WN	1924	N761RR	194	215	176	254	275	LAS	IND	1591	9	
19	2008	1	9	3	1458	1040	1725	1315	WN	2556	N501SW	87	95	76	250	258	BNA	BWI	588	4	
20	2008	1	7	1	2300	1835	113	2105	WN	2804	N420WN	253	270	240	248	265	MDW	PDX	1751	5	
21	2008	1	5	6	47	2040	151	2145	WN	505	N435WN	64	65	51	246	247	BWI	PVD	328	5	
22	2008	1	5	6	1558	1225	14	2010	WN	505	N442WN	316	285	250	244	213	SAN	BWI	2295	5	
23	2008	1	5	6	1931	1540	2104	1705	WN	1179	N718SW	93	85	77	239	231	SAN	OAK	446	7	
24	2008	1	4	5	1822	1425	2003	1605	WN	753	N726SW	101	100	88	238	237	PDX	OAK	543	6	

Problem

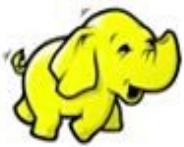
- Write a map reduce to count the frequency of arrival delay in each airport by classify in to three groups:
 - *Very late* : $delay > 30$
 - Late : $5 < delay \leq 30$
 - Ontime: $delay \leq 5$
- Export the result to excel and plot the histogram

Recommendation to Further Study



Big Data Certification Course

120 Hrs: Start 12 March 2015



<http://www.imcinstitute.com/bigdatacert>

Tel : 088-192-7975

Thank you

www.imcinstitute.com
www.facebook.com/imcinstitute