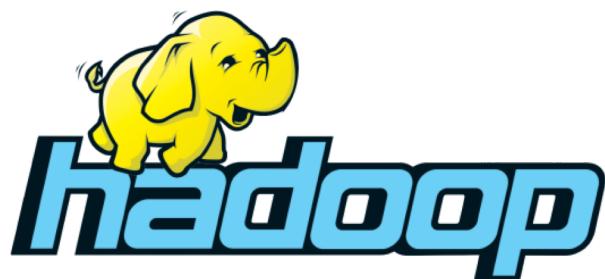


# Hands-on Introduction to Apache Hadoop and Spark

Douglas Eadline

(Part 1)



# Presenter

## Douglas Eadline

*deadline@eadline.org*

*@thedeadline@mast.hpc.social*

*@thedeadline*

*<http://www.limulus-computing.com>*



- HPC/Hadoop Consultant/Writer/Educator
- Teach several graduate courses as part of Masters in Data Science

# Outline Day 1

- **Segment 1:** Introduction and Quick Overview of Hadoop, Spark, Kafka (45 mins)
- **Segment 2:** Installing and Running the Linux Hadoop Minimal Virtual Machine (20 mins)
- **Segment 3:** Using the Hadoop Distributed File System (HDFS) (25 mins)
- Break 10 minutes
- **Segment 4:** Running and Monitoring Hadoop Applications (40 mins)
- **Segment 5:** Running Apache Sqoop (40 mins)

# Outline Day 2

- **Segment 6:** Using Apache Hive (30 mins)
- **Segment 7:** Running Apache Spark (50 mins)
- Break: 10 minutes
- **Segment 8:** Running Apache Kafka (50 mins)
- **Segment 9:** Introduction and Example Analytics Application using Apache Zeppelin (30 mins)
- **Segment 10:** Wrap-up/ Resources/Where to Go Next (10 mins)

# Recommended Approach To Class

- Course covers a lot of material!
- Designed to get you started (“hello.c” approach)
- Sit back and watch the examples
- All examples are provided in a notes file
- I will refer to file throughout the class  
(cut and paste)
- The notes files are available for download along  
with some help on installing software (URL on class  
web page) <https://www.clustermonkey.net/scalable-analytics/doku.php>

# User Programming Environment

There are four options (URLs provided at end of course):

1. Use the Linux Hadoop Minimal Sandbox virtual machine that can run under VirtualBox or VMWare.
2. Use the Hortonworks Sandbox, a full featured Hadoop/Spark virtual machine that runs under Docker, VirtualBox, or VMWare.
3. Install software from Apache.org (instructions provided) Best for Linux machines/laptops
4. Use a resource available to you: local cluster/cloud

# Courses In Scalable Data Pipelines

1. Apache Hadoop, Spark, and Kafka Foundations (3 hours-1 day)
2. Bash Programming Quick-start for Data Science (4 hours-1 day)
- 3. Hands-on Introduction to Apache Hadoop, Spark, and Kafka Programming (6 hours-2 days)**
4. Getting Started with Apache Kafka (4 hours-1 day)
5. Kafka Methods and Administration in Practice (4 hours-1 day)

# Course Webpage

<https://www.clustermonkey.net/scalable-analytics>

The screenshot shows a web browser displaying a course page. At the top left is a yellow cartoon elephant icon. To its right, the text reads "Live On-Line Training: Scalable Data Pipelines with Hadoop, Spark, and Kafka". On the far right of the header are links for "Search", "Recent Changes", "Media Manager", and "Sitemap". Below the header, a navigation bar includes "Trace: • start" on the left and "start" on the right. The main content area features a heading "Welcome to the Effective Data Pipelines Series" followed by the subtext "(previously Scalable Analytics with Apache Hadoop and Spark)". Below this, a bold statement reads "The six essential courses on the path to scalable data science pipelines nirvana—or at least a good start". A section titled "Course Descriptions and Links" contains the instruction "Click on the course name for availability and further information. New courses are being added." To the right of the main content is a "Table of Contents" sidebar with a list of course links:

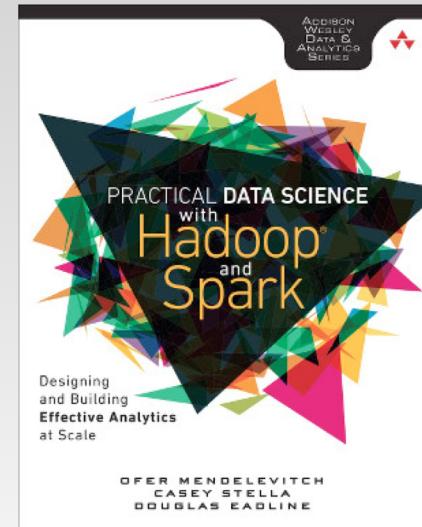
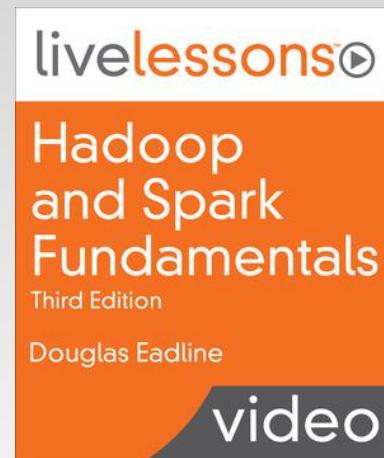
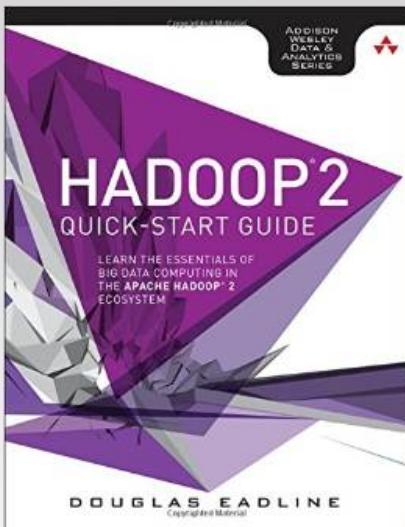
- ◆ Course Descriptions and Links
- ◆ Class Notes for Hands-on Introduction to Apache Hadoop and Spark Programming
- ◆ Zeppelin Notebook for Scalable Data Science with Hadoop and Spark
- ◆ DOS to Linux and Hadoop HDFS Help:
- ◆ Linux Hadoop Minimal (LHM) Virtual Machine Sandbox
- ◆ Cloudera-Hortonworks HDP

# Supporting Materials

Hadoop 2 Quick-Start

Hadoop Fundamentals (Video Course)

Practical Data Science with Hadoop and Spark



# Cluster Used for Class

## Limulus™ Desk-side Hadoop/Spark Cluster

- Up to 64 physical fast cores (+64 hyper-threads) on 8 motherboards
- Up to 512 GBytes of Memory
- Up to 64 TB of SSD Hadoop HDFS storage
- Up to 112 TB of central HDD (RAID 6) storage
- Full internal 25 GbE Ethernet networking
- Residential/office/classroom power
- Cool and quiet operation
- Turn-key ready to run Linux appliance fully installed HPC and Hadoop/Spark

**<https://www.limulus-computing.com>**



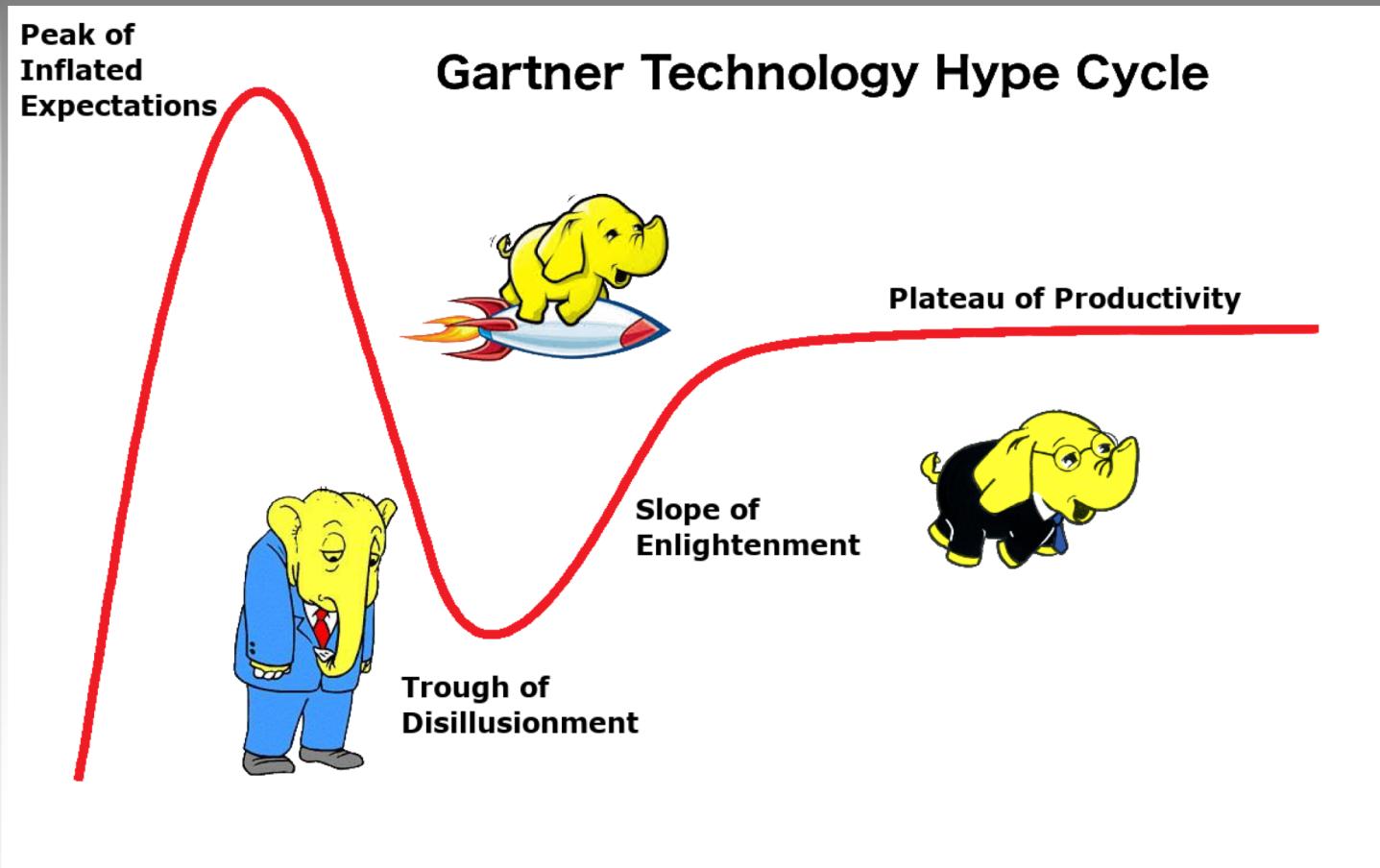
# Segment 1

## Quick Overview of Hadoop and Spark

# Big Data Analytics

- Three V's (Volume, Velocity, Variability)
- Data that are large (lake), growing fast (waterfall), unstructured (recreation) - not all may apply.
- May not fit in a "relational model and method"
- Can Include: video, audio, photos, system/web logs, click trails, IoT, text messages/email/tweets, documents, books, research data, stock transactions, customer data, public records, human genome, and many others.

# Is Hadoop Going Away ? (No)

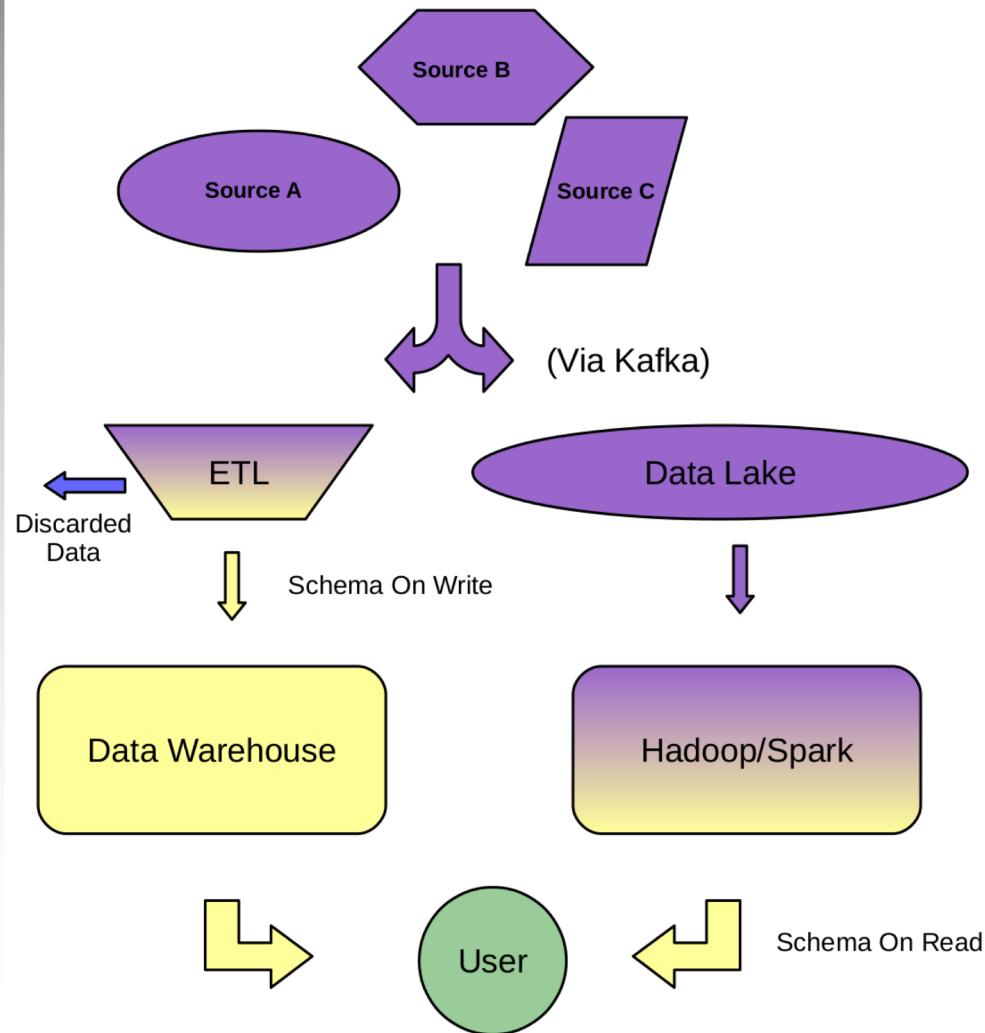


# Hadoop Data Lake

Data Warehouse applies “schema on write” and has an Extract, Transform, and Load (ETL) step.

Hadoop/Spark applies “schema on read” and the ETL step is part of processing. All input data are placed in the lake in raw form.

## Data Warehouse vs. Hadoop



# Defining Hadoop (Version 2+)

***A PLATFORM for data lake analysis that supports software tools, libraries, and methodologies***

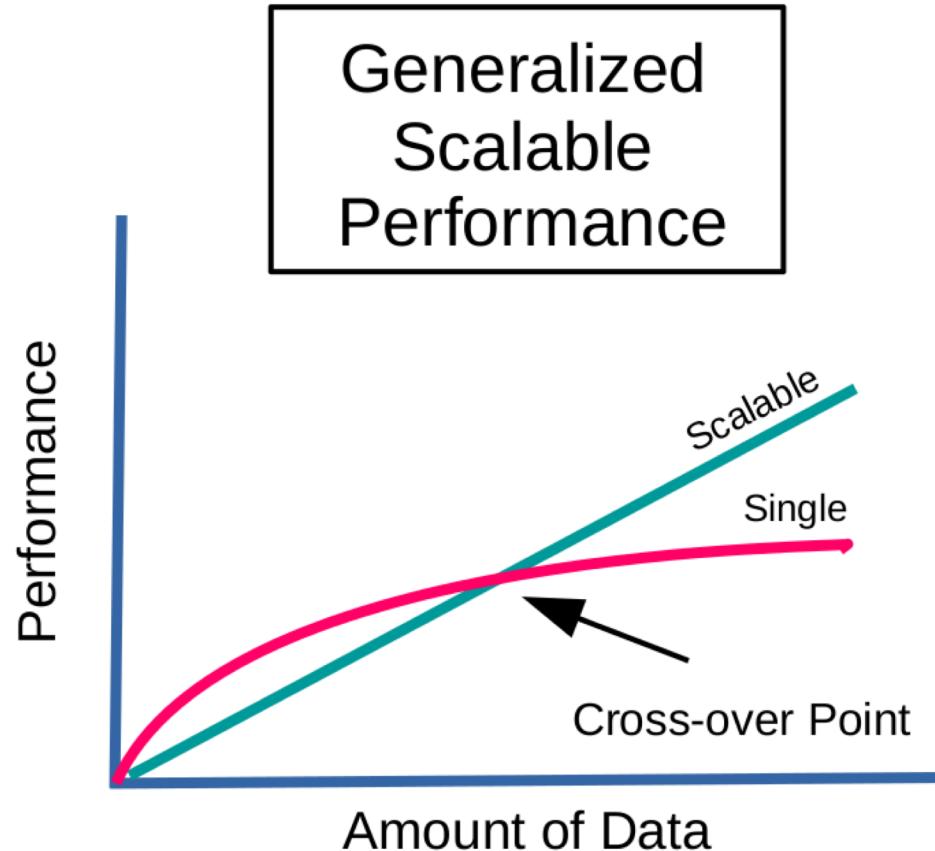
- Core and most tools are open source (Apache License)
- Written in Java, but not exclusively a Java platform
- Primarily GNU/Linux, Windows versions available
- Scalable from single server to thousands of machines
- Runs on commodity hardware and in the cloud
- Application level fault tolerance possible
- Multiple processing models and libraries (Frameworks)
- Current Version is 3.x

# Scalable Behavior

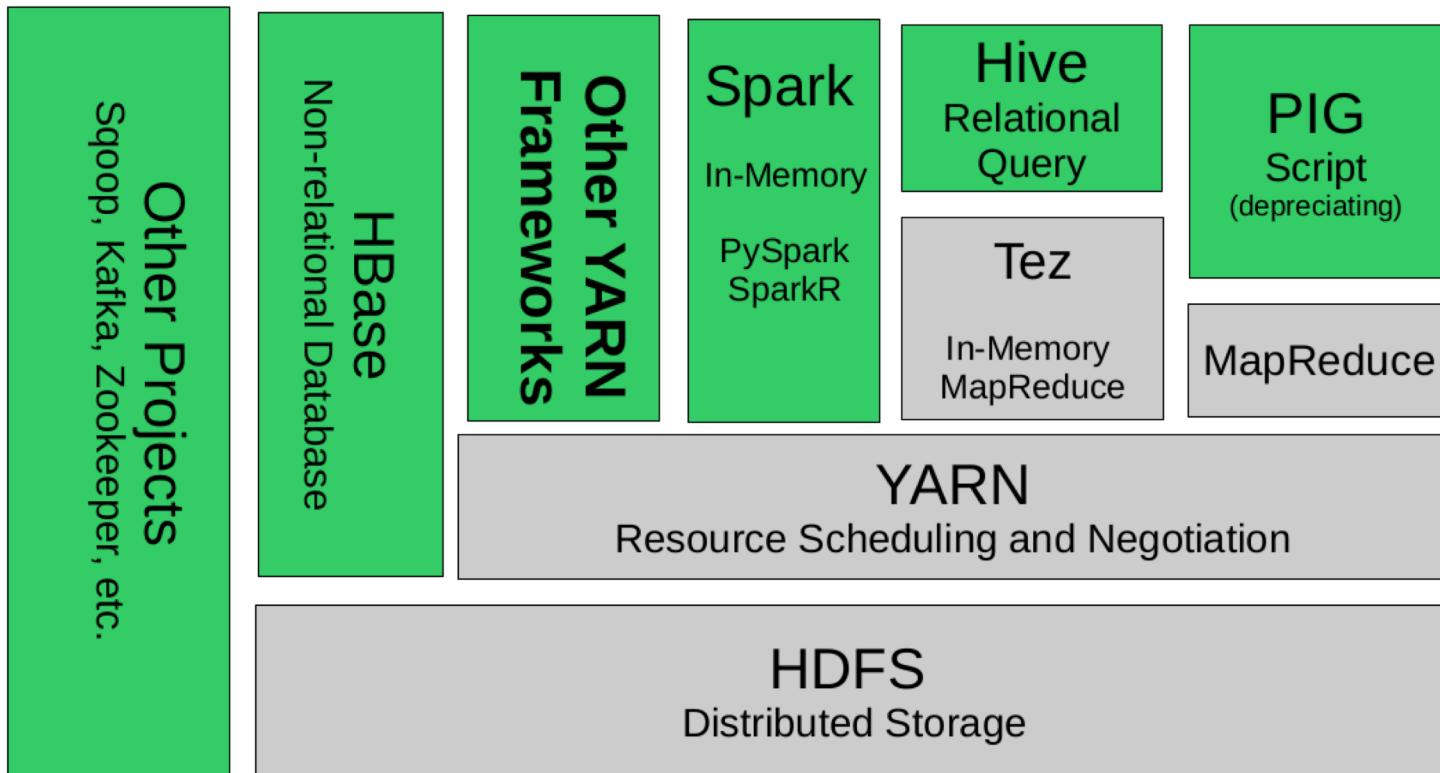
## Scalable Systems

The ability to apply resources (servers, VM Instances) to a problem as a means to increase performance (green line) over single server (red line)

**But,** initially performance may lag due to overhead



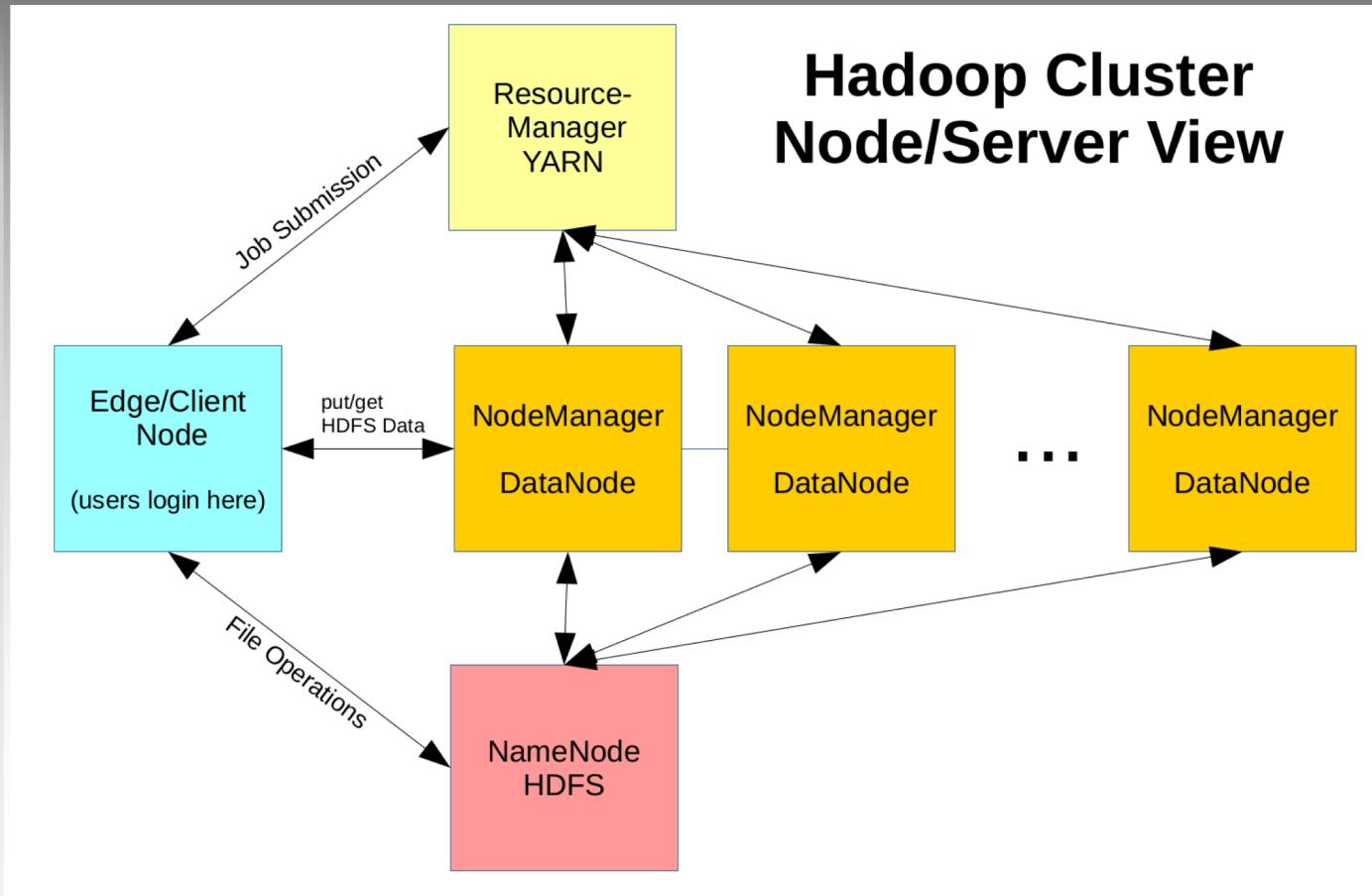
# Hadoop Components



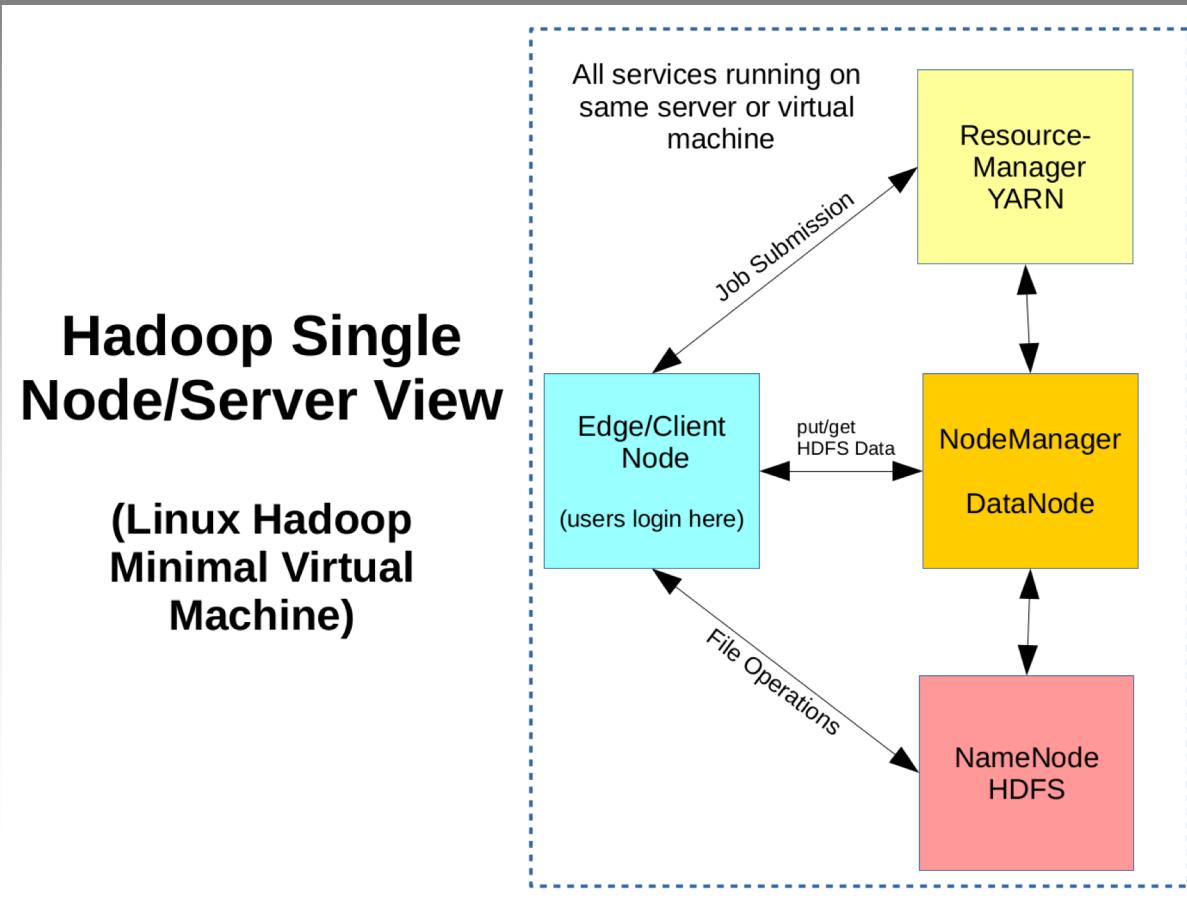
# Hadoop Core Components

- **HDFS – Hadoop Distributed File System.** Designed to be a fault tolerant streaming file system. Default block size is 64 MB vs. 4 KB for Linux ext4. Runs on commodity servers, director “NameNode” and multiple “DataNodes.” Data are replicated on multiple servers.
- **YARN – Yet Another Resource Negotiator.** Master scheduler and resource allocator for the entire Hadoop cluster. User jobs ask YARN for resources (containers) and data locality. Provides dynamic resource allocation(run-time). Runs on commodity servers, master “ResourceManager” and multiple “NodeManagers.”
- **MapReduce/Tez – YARN Application.** Provides classic MapReduce and optimized MapReduce processing.
- **Cluster servers (nodes) are usually both DataNodes and NodeManagers (Hyperconverged) Move processing to data.**

# Hadoop Core Components Node View



# Hadoop Core Components on Single Node



# Apache Components We Will Use

- **Apache Sqoop** is a tool for transferring bulk data between Hadoop and relational databases.
- **Apache Hive** is an SQL compatible language for data summarization, ad-hoc queries, and the analysis of large datasets
- **Apache Spark** is an easy to use language for writing analytics applications that includes MapReduce, SQL queries and other tools.
- **Apache Kafka** Kafka is a high throughput, scalable, service that records all kinds of data in real-time (a message broker)
- **Apache Zeppelin** is web-GUI front end for creating data analytics applications

# Questions ?

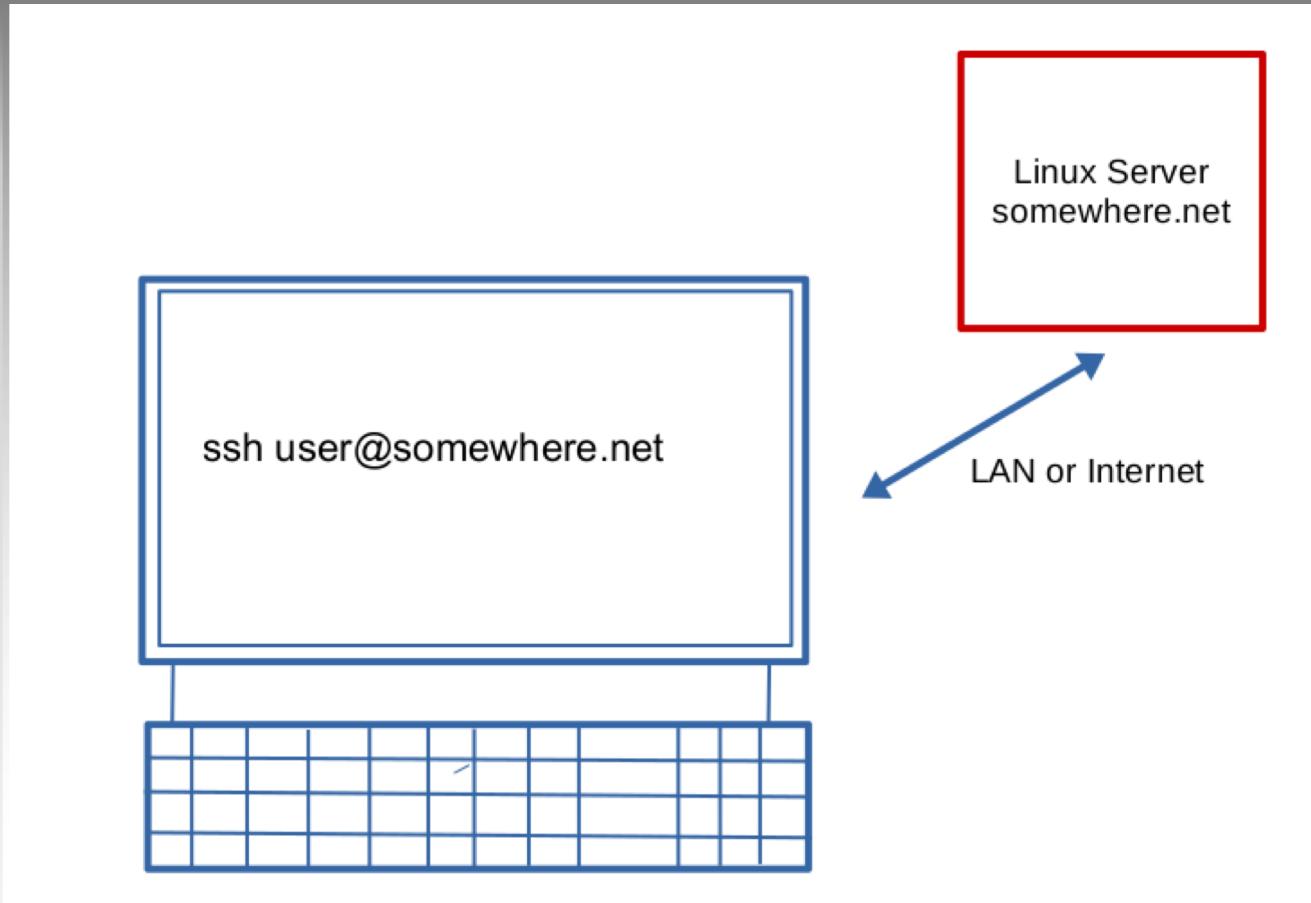
# Segment 2

## The Linux Hadoop Minimal VM

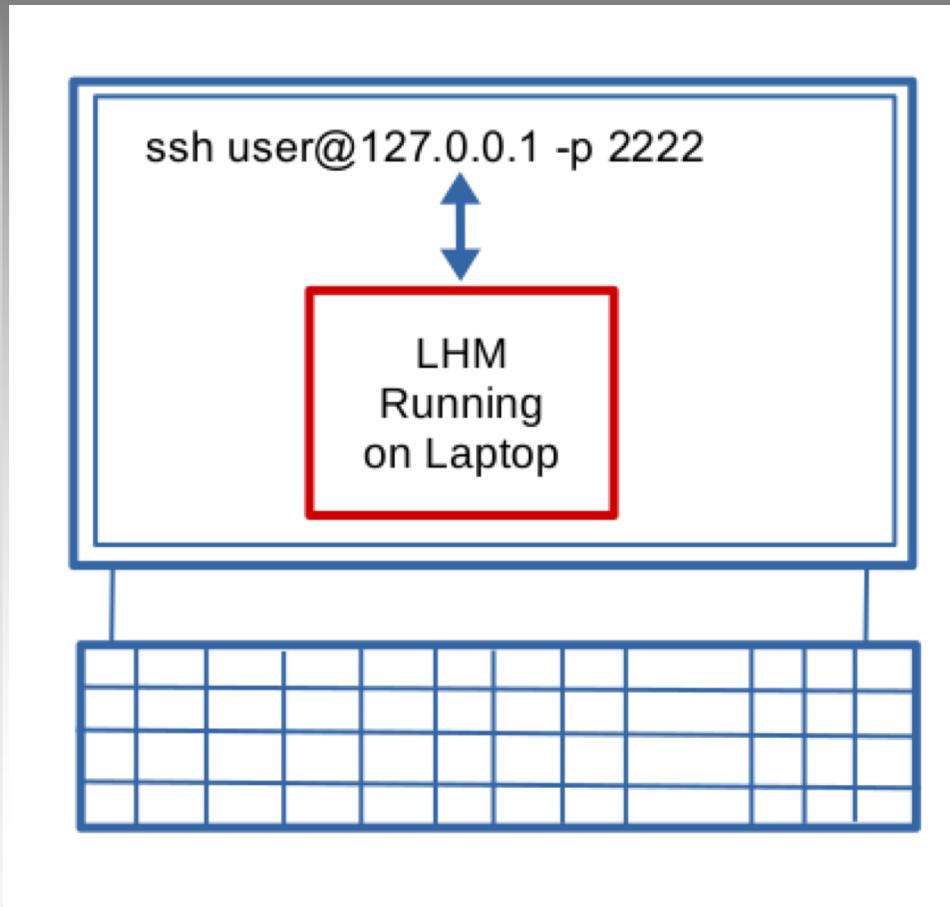
# LHM: Your Personal Linux Server

- Use a virtual machine (VM) to create a real Linux environment.
- Run on Laptop or Desktop with Virtual Box
- The Linux Hadoop Minimal VM is designed to be a small, single server Hadoop/Spark system (used in other courses)
- An x86\_64 based Notebook or Laptop with at least 4 cores/threads, 4 GB of Memory, 70 GB of disk space.
- Linux CentOS 7 (Red Hat rebuild) image
- Walks and talks like a separate computer

# ssh Out to LAN/Internet

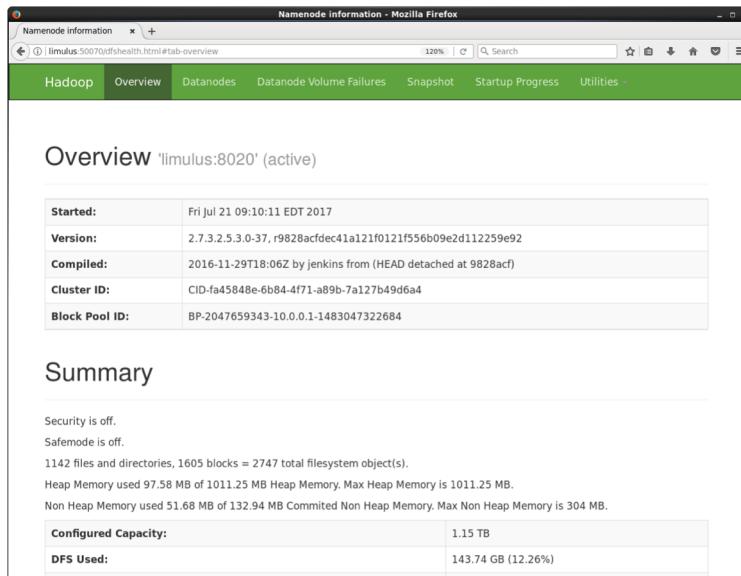


# Connect to the LHM



# Segment 3

## Using the Hadoop Distributed File System



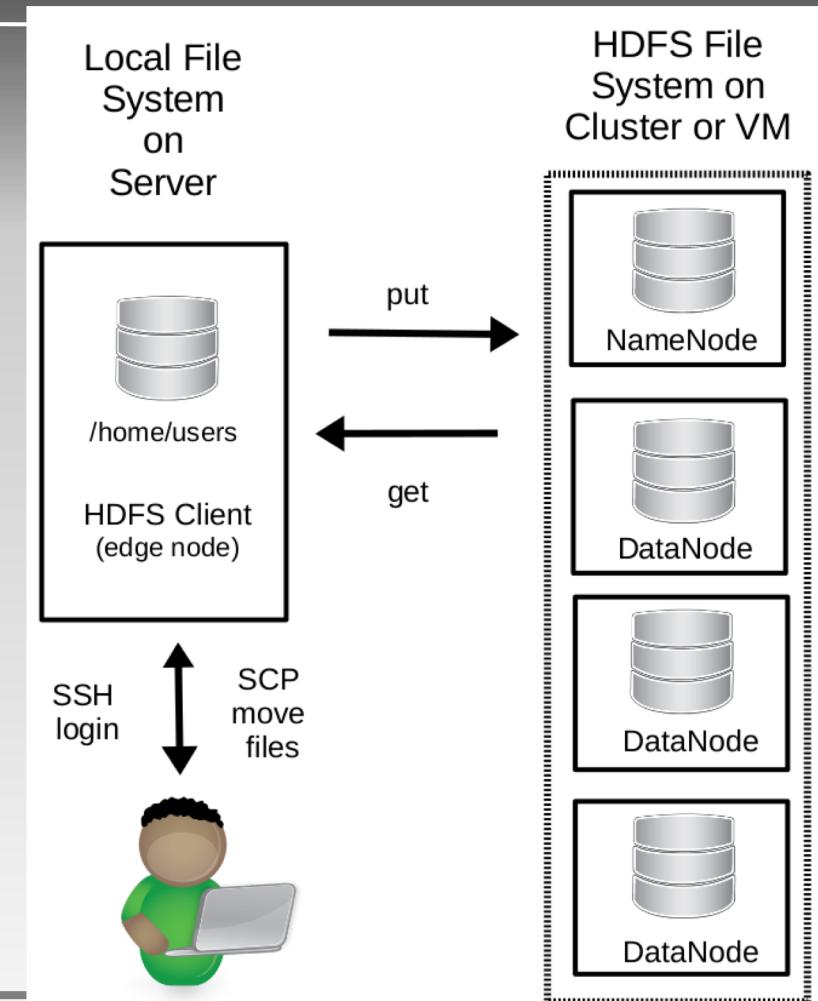
The screenshot shows a Mozilla Firefox browser window displaying the 'Namenode information' page. The URL in the address bar is 'limulus:50070/dfshealth.html#tab-overview'. The page has a green header bar with tabs: 'Hadoop' (selected), 'Overview', 'Datanodes', 'Datanode Volume Failures', 'Snapshot', 'Startup Progress', and 'Utilities'. The main content area is titled 'Overview' and shows the cluster identifier 'limulus:8020' (active). Below this, there is a table with the following data:

Started:	Fri Jul 21 09:10:11 EDT 2017
Version:	2.7.3.2.5.3.0-37, r9828acfdec41a121f0121f556b09e2d112259e92
Compiled:	2016-11-29T18:06Z by jenkins from (HEAD detached at 9828acf)
Cluster ID:	CID-fa45848e-6b84-4f71-a89b-7a127b49d6a4
Block Pool ID:	BP-2047659343-10.0.0.1-1483047322684

Below the table, under the 'Summary' section, it says 'Security is off.' and 'Safemode is off.' It also displays statistics: '1142 files and directories, 1605 blocks = 2747 total filesystem object(s.)', 'Heap Memory used 97.58 MB of 1011.25 MB Heap Memory, Max Heap Memory is 1011.25 MB.', 'Non Heap Memory used 51.68 MB of 132.94 MB Committed Non Heap Memory, Max Non Heap Memory is 304 MB.', 'Configured Capacity: 1.15 TB', and 'DFS Used: 143.74 GB (12.26%)'.

# How the User “Sees” HDFS

- HDFS is a separate file system from the host server
- Data must be moved to server (from user source or another server/web)
- Data must be moved to HDFS using put and get commands
- All Hadoop processing happens in HDFS



# HDFS Notes

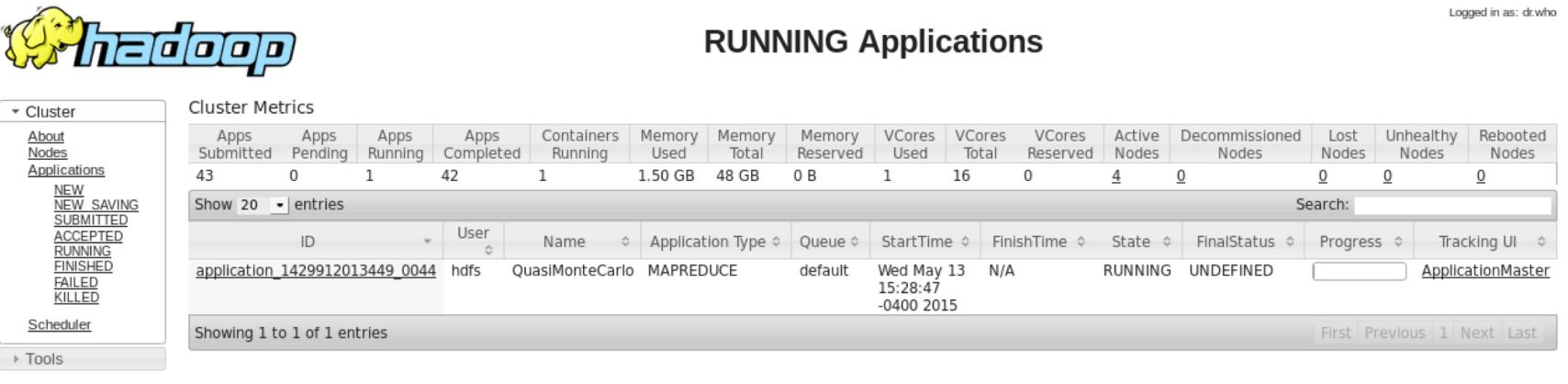
1. All user HDFS commands start with “`hdfs dfs`” because it operates a separate file system.
2. User **`hdfs`** is “root” for HDFS. Only user **`hdfs`** can perform certain administrative tasks.
3. Data files are sliced and distributed over HDFS DataNodes (default set by admin or user can override) and **data splits** are defined programmatically.

# Questions ?

# 10 Minute BREAK

# Segment 4

## Running and Monitoring Hadoop Applications



The screenshot shows the Hadoop Job Tracker interface. At the top left is the Hadoop logo. To its right, the text "Logged in as: dr.who". Below the logo is the title "RUNNING Applications". On the left, there's a sidebar with "Cluster Metrics" and a table showing cluster statistics:

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	Vcores Used	Vcores Total	Vcores Reserved	Active Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Rebooted Nodes
43	0	1	42	1	1.50 GB	48 GB	0 B	1	16	0	4	0	0	0	0

Below this is a search bar with "Search:" and a dropdown showing "Show 20 entries". The main table lists the running application:

ID	User	Name	Application Type	Queue	StartTime	FinishTime	State	FinalStatus	Progress	Tracking UI
application_1429912013449_0044	hdfs	QuasiMonteCarlo	MAPREDUCE	default	Wed May 13 15:28:47 -0400 2015	N/A	RUNNING	UNDEFINED		ApplicationMaster

At the bottom of the table are links for "Showing 1 to 1 of 1 entries", "First", "Previous", "1", "Next", and "Last".

# What Is Map-Reduce?

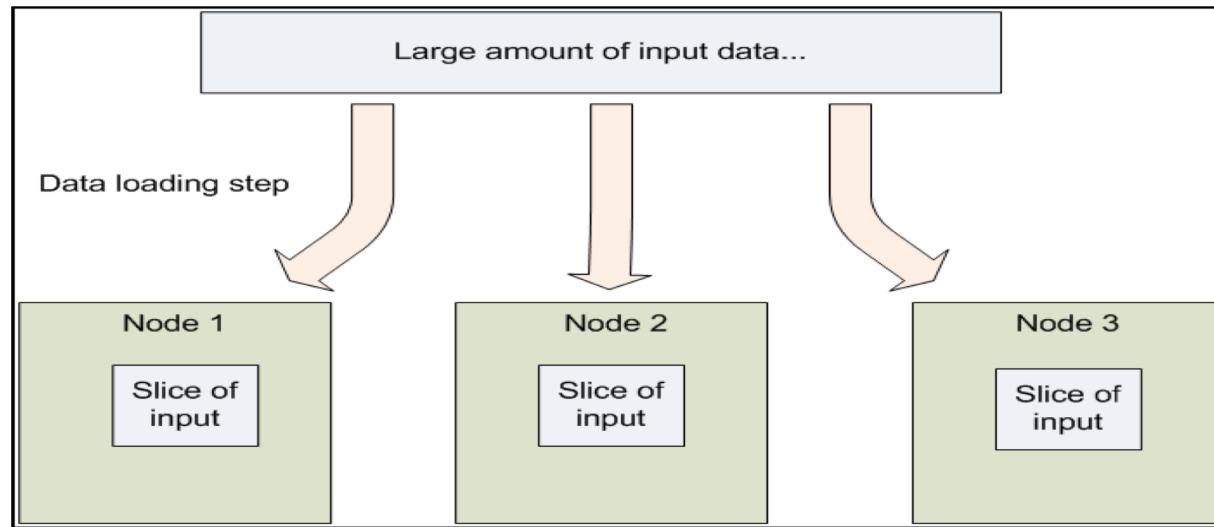
## Map Reduce Is a Simple Algorithm

(and can run in parallel due to no side-effects)

- Distinct steps and one-way communication
- Map then Reduce
- Uses (key, value) pair
- Hadoop Map-Reduce used in Pig and Hive
- Spark uses its own Map-Reduce library
- Works great in many cases (even for some HPC applications), but it is not the only algorithm for parallel computing.

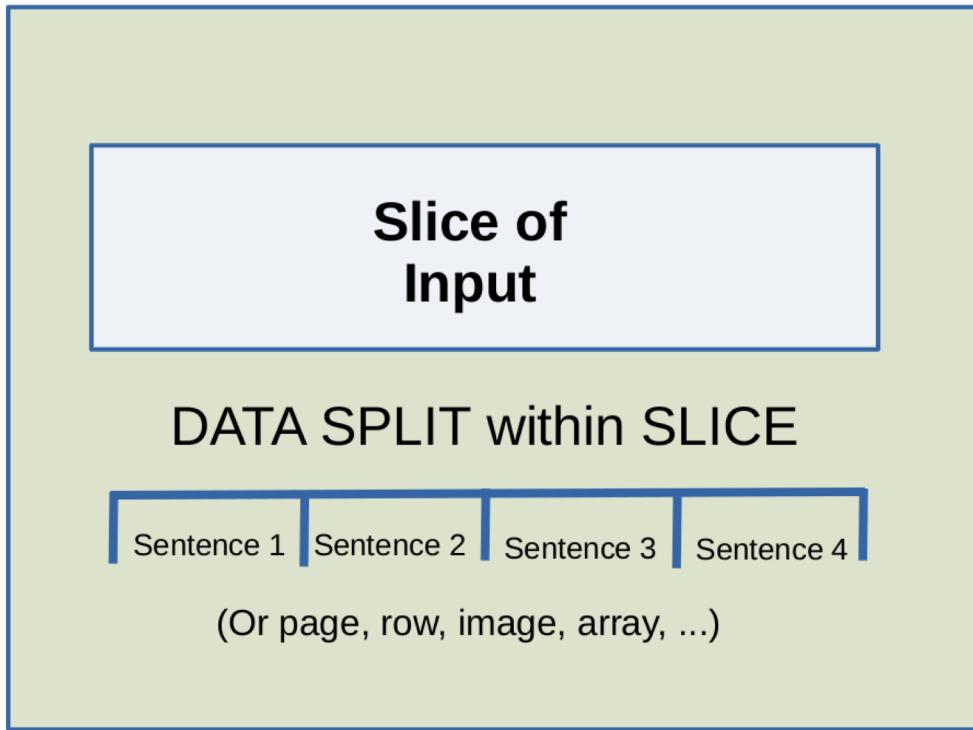
# Hadoop MapReduce Big Picture

1. When data are moved into HDFS files are sliced and placed on different servers (nodes).
2. Queries are applied (mapped) to each distributed data set defined by “data splits”
3. Then, if needed, results are reduced to single single answer



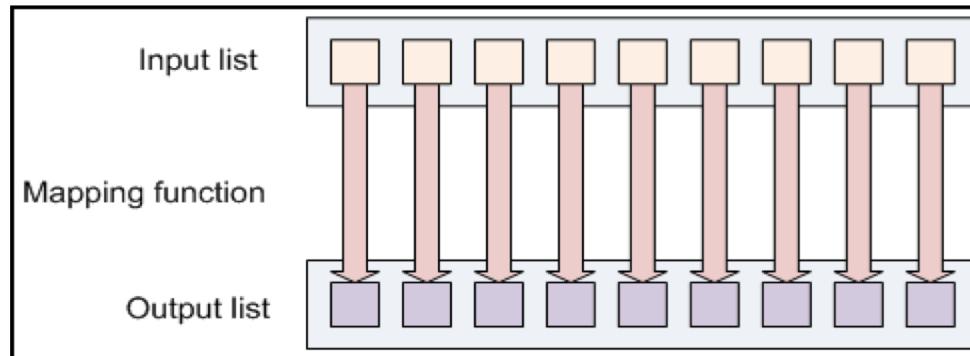
# Slices and Splits

Each Slice will have programmatically defined data splits



# Scalable Mapping Step

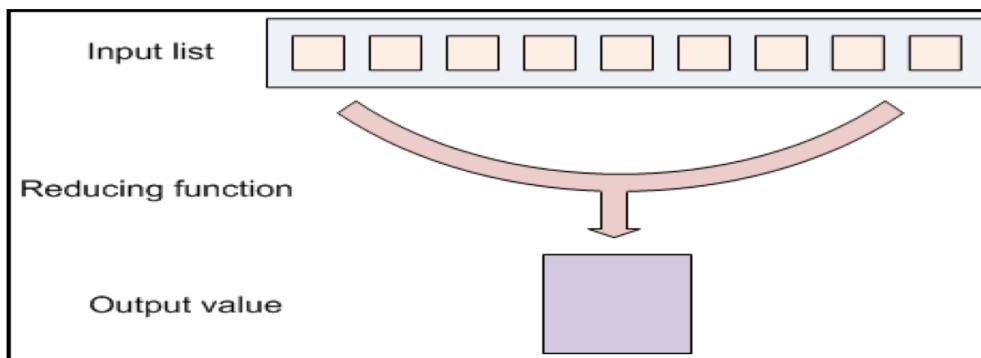
1. Slice data into parts.
2. Apply the same **Mapping** function to Input List (user defined splits)



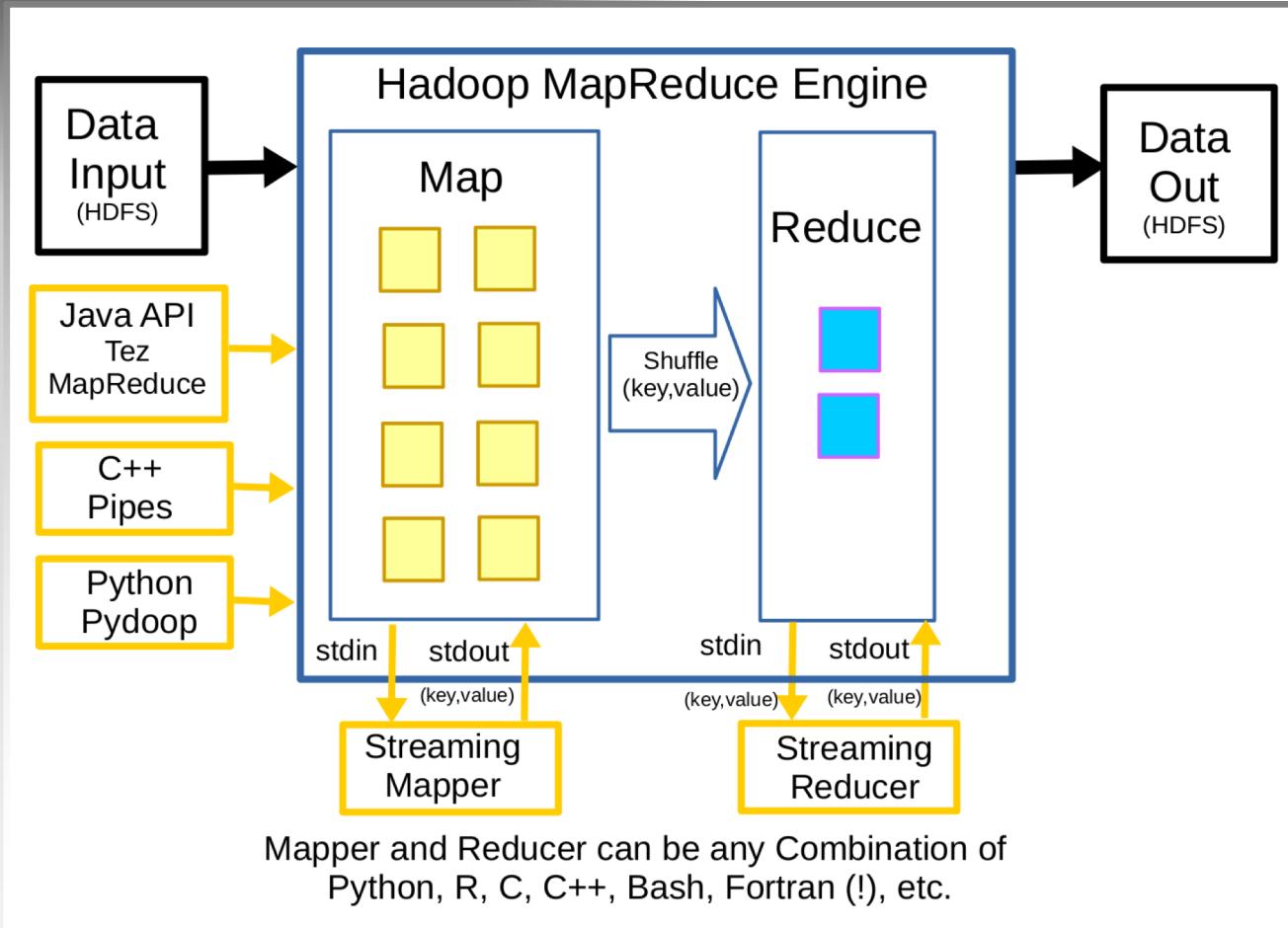
3. Each step provides a separate set of results (Output list).

# Reduction Step

- Results from each **map** (as applied to each split) are combined and **reduced** to a single Output value.



# Programming Hadoop MapReduce



# Questions ?

# Segment 7

## Running Apache Sqoop

<http://sqoop.apache.org>

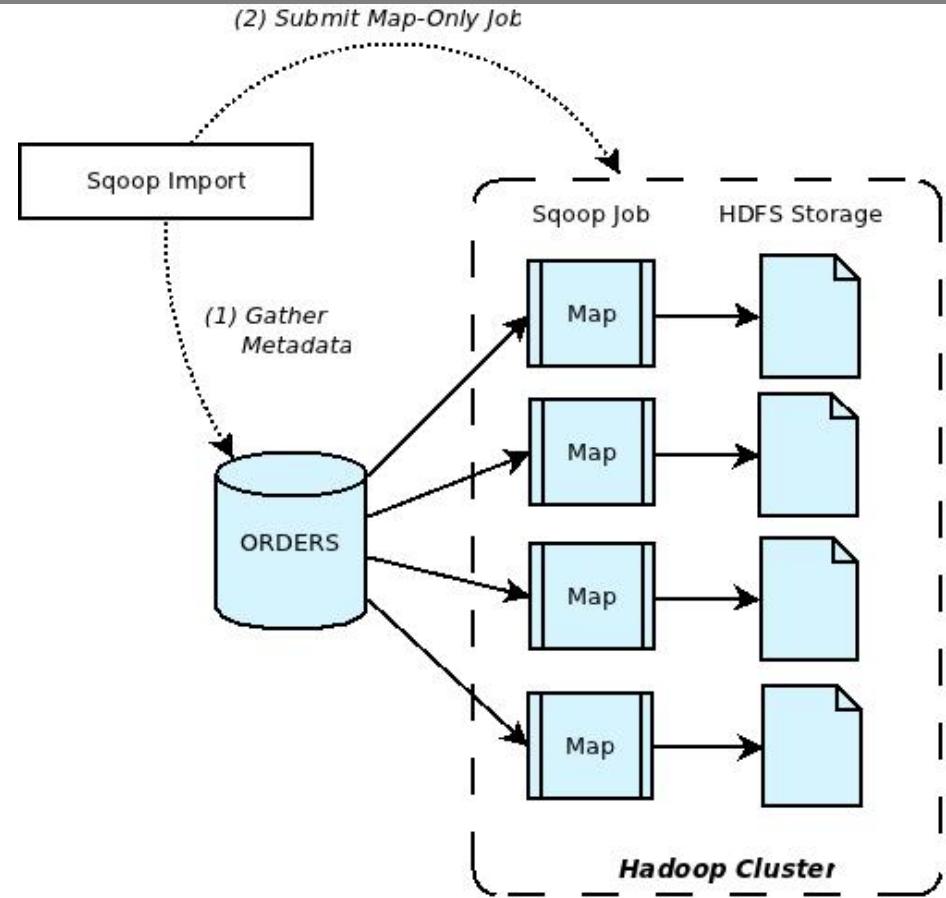


# Apache Sqoop Background

Sqoop is a tool designed to transfer data between Hadoop and relational databases or mainframes. You can use Sqoop to import data from a relational database management system (RDBMS) such as MySQL or Oracle or a mainframe into the Hadoop Distributed File System (HDFS), transform the data in Hadoop MapReduce, and then export the data back into an RDBMS.

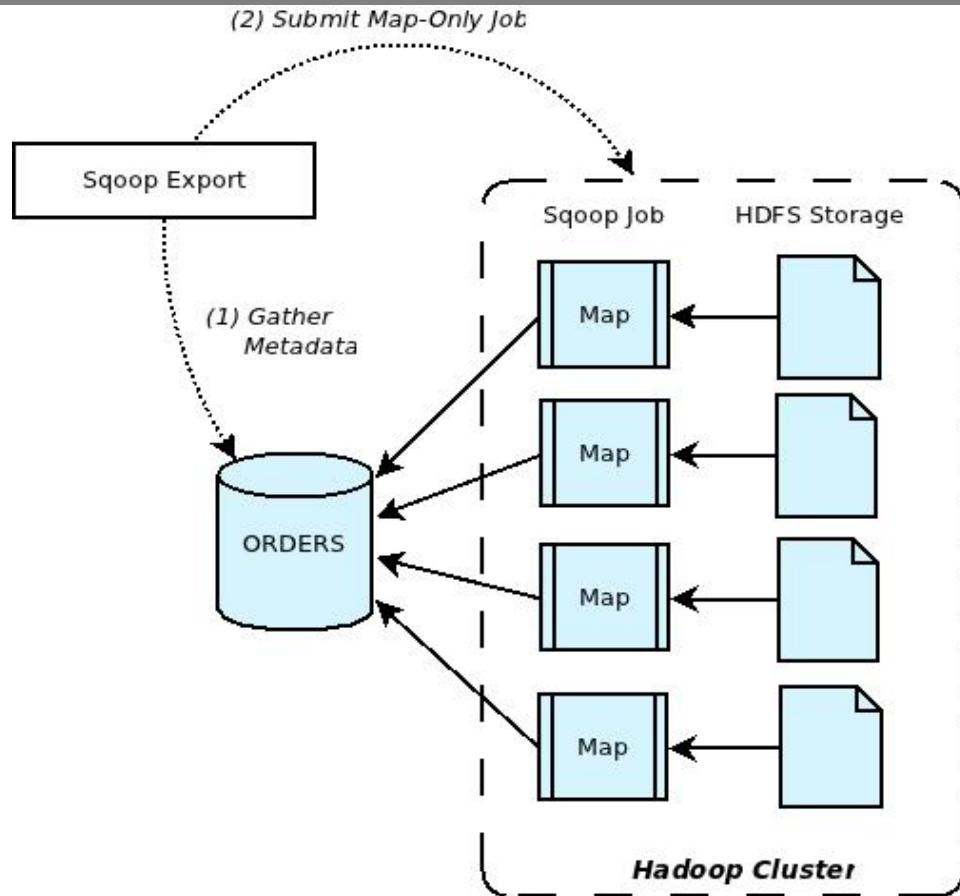
# Sqoop Data Import

- Sqoop works with local RDBMS
- Sqoop gathers metadata about DB
- Can work sequentially or in parallel
- In parallel each map process contacts the RDBMS system and “pulls” its slice of data



# Sqoop Data Export

- Sqoop works with local RDBMS
- Sqoop gathers metadata about DB
- Can work sequentially or in parallel
- In parallel each map process contacts the RDBMS system and “pushes” its slice of data



# Questions ?

## Continue Tomorrow