Install Apache Hadoop for Development/Production using Open source Distributions

Install Pure Apache Hadoop

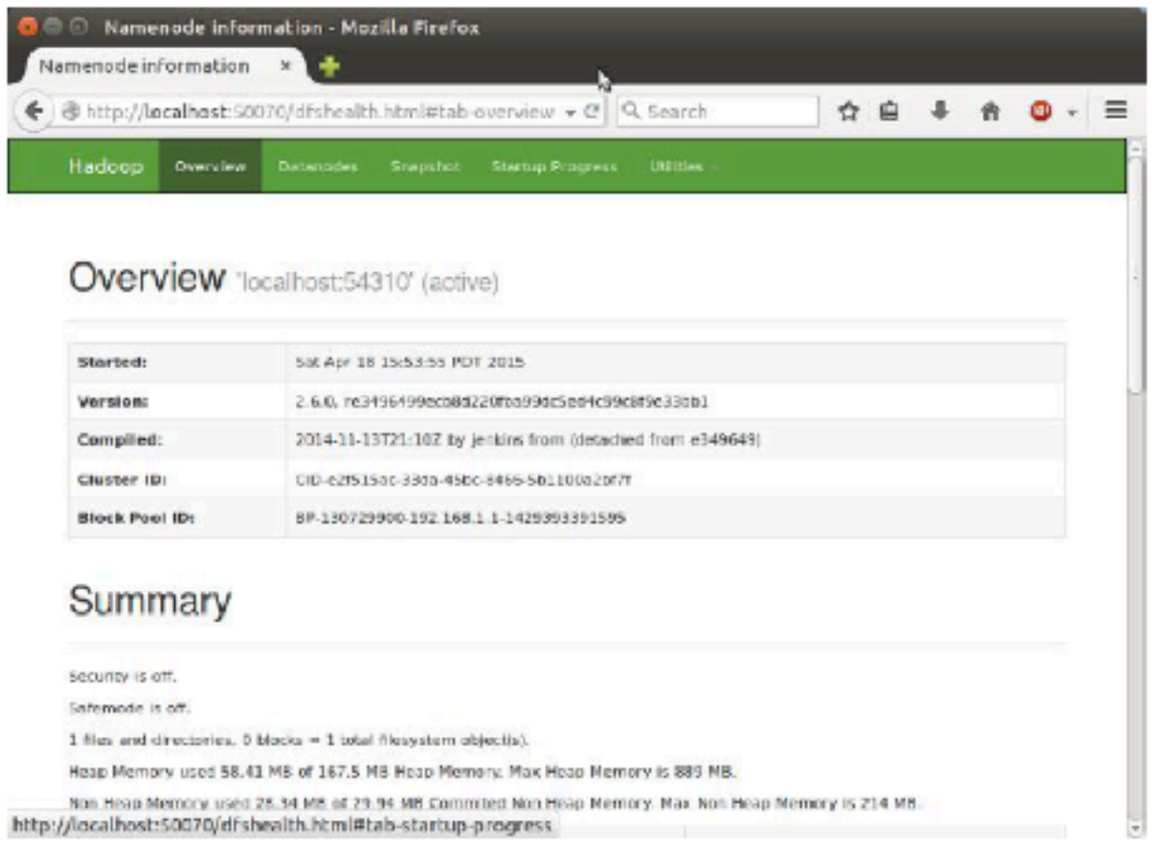
CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

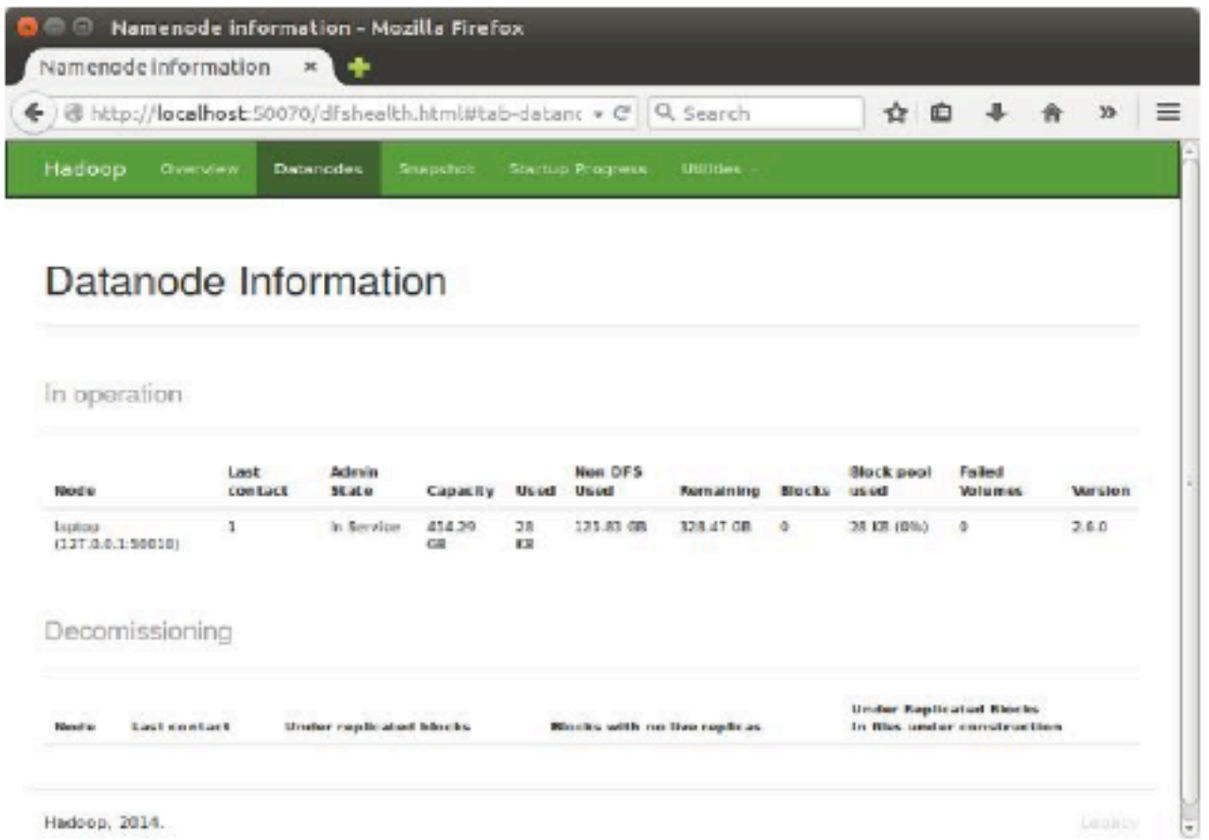
1. Ubuntu or Centos (Linux)
2. Installing JDK
3. Download/Extract Hadoop
4. Installing Hadoop
5. Configure xml files
6. Formatting HDFS
7. Start Hadoop and then see Hadoop Web Console.
8. Install Services (Hive , Sqoop ,HBASE,.....)
9. Stop Hadoop (if needed.)

```
<configuration>
<property>
  <name>yarn.resourcemanager.hostname</name>
    <value>localhost</value>
</property>
<property>
  <name>yarn.resourcemanager.scheduler.address</name>
    <value>localhost:8030</value>
</property>
<property>
  <name>yarn.resourcemanager.resource-tracker.address</name>
    <value>localhost:8031</value>
</property>
<property>
  <name>yarn.resourcemanager.address</name>
    <value>localhost:8032</value>
</property>
```

```
<configuration>
<property>
  <name>fs.defaultFS</name>
    <value>hdfs://localhost:9000</value>
</property>
</configuration>
```

Hadoop User Web Interfaces





Hadoop SecondaryNameNode - Mozilla Firefox

Hadoop SecondaryNa...

http://localhost:50090/status.jsp

Search

SecondaryNameNode

Version:	2.6.0, e3496499ecb8d220fba99dc5ed4c99c8f9e33bb1
Compiled:	2014-11-13T21:10Z by jenkins from (detached from e349649)

```
SecondaryNameNode Status
Name Node Address      : localhost/127.0.0.1:54310
Start Time             : Sat Apr 18 16:43:38 PDT 2015
Last Checkpoint        : 79 seconds ago
Checkpoint Period      : 3600 seconds
Checkpoint Transactions: 1000000
Checkpoint Dirs        : [file:///app/hadoop/tmp/dfs/namesecondary]
Checkpoint Edits Dirs  : [file:///app/hadoop/tmp/dfs/namesecondary]
```

Directory: /logs/ - Mozilla Firefox

Directory: /logs/

http://localhost:50090/logs/

Search

Directory: /logs/

SecurityAuth-hduser.audit	0 bytes Apr 18, 2015 3:40:58 PM
hadoop-hduser-datanode-laptop.log	72879 bytes Apr 18, 2015 4:44:13 PM
hadoop-hduser-datanode-laptop.out	718 bytes Apr 18, 2015 4:43:21 PM
hadoop-hduser-datanode-laptop.out.1	718 bytes Apr 18, 2015 3:53:49 PM
hadoop-hduser-datanode-laptop.out.2	718 bytes Apr 18, 2015 3:41:03 PM
hadoop-hduser-namenode-laptop.log	121216 bytes Apr 18, 2015 4:52:23 PM
hadoop-hduser-namenode-laptop.out	718 bytes Apr 18, 2015 4:43:16 PM
hadoop-hduser-namenode-laptop.out.1	718 bytes Apr 18, 2015 3:53:44 PM
hadoop-hduser-namenode-laptop.out.2	718 bytes Apr 18, 2015 3:40:58 PM
hadoop-hduser-secondarynamenode-laptop.log	51913 bytes Apr 18, 2015 4:52:38 PM
hadoop-hduser-secondarynamenode-laptop.out	718 bytes Apr 18, 2015 4:43:37 PM
hadoop-hduser-secondarynamenode-laptop.out.1	718 bytes Apr 18, 2015 3:54:06 PM
hadoop-hduser-secondarynamenode-laptop.out.2	718 bytes Apr 18, 2015 3:42:52 PM
userlogs/	4096 bytes Apr 18, 2015 4:52:22 PM
yarn-hduser-nodemanager-laptop.log	81625 bytes Apr 18, 2015 4:44:32 PM

Logs

Hadoo

Logged in as: dr.who

:44:02 PM
:54:32 PM
:43:10 PM
:44:32 PM
:44:00 PM
:54:29 PM
:43:08 PM

hadoop

About the Cluster

- Cluster
  - About
  - Nodes
  - Node Labels
  - Applications
    - NEW
    - NEW_SAVING
    - SUBMITTED
    - ACCEPTED
    - RUNNING
    - FINISHED
    - FAILED
    - KILLED
  - Scheduler
- Tools

Cluster Metrics

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
0	0	0	0	0	0 B	8 GB	0 B	0	8	0	1	0	0	0	0

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation
Capacity Scheduler	[MEMORY]	<memory:1024, vCores:1>	<memory:8192, vCores:8>

Cluster overview

Cluster ID:	1465353454396
ResourceManager state:	STARTED
ResourceManager HA state:	active
ResourceManager HA zookeeper connection state:	ResourceManager HA is not enabled.
ResourceManager RMStateStore:	org.apache.hadoop.yarn.server.resourcemanager.recovery.NullRMStateStore
ResourceManager started on:	Wed Jun 08 09:37:34 +0700 2016
ResourceManager version:	2.7.2 from b165c4fe8a74265c792ce23f546c64604acf0e41 by jenkins source checksum c63f7cc71b8f03249e35129f0f7492d on 2016-01-26T00:16Z
Hadoop version:	2.7.2 from b165c4fe8a74265c792ce23f546c64604acf0e41 by jenkins source checksum d0fda26633fa762bff87ec759ebe689c on 2016-01-26T00:08Z

Hadoop Distribution

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

On-Premise

- Pure Apache Hadoop
- Cloudera
- Hortonworks
- MapR
- Pivotal
- IBM InfoSphere BigInsight

On-Cloud (Hadoop as a Service)

- Amazon EMR
- Microsoft Azure HDInsight
- Google Cloud Platform

The Forrester Wave: Big Data Hadoop Distributions Q1 2016

The Forrester Wave: Big Data Hadoop Cloud Q1 2016

Hadoop Development: Sandbox

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

Recommended

Cloudera Quickstart

Hortonworks Sandbox

On PC/Mac

Require to install VMWare Player or VirtualBox

On Linux Virtual Server

Need to install Docker Engine

http://hortonworks.com/downloads/#sandbox

Hortonworks Sandbox

Hortonworks Sandbox on a VM

HDP® 2.5 on Hortonworks Sandbox

Tutorials | Release Notes | Import on Virtual Box | MD5 : b08f1dce17ab3ae2431532be74bdbbbb — DOWNLOAD FOR VIRTUALBOX

Tutorials | Release Notes | Import on VMware | MD5 : fe1e91bc26e6879fdc6dde6e3778c262 — DOWNLOAD FOR VMWARE

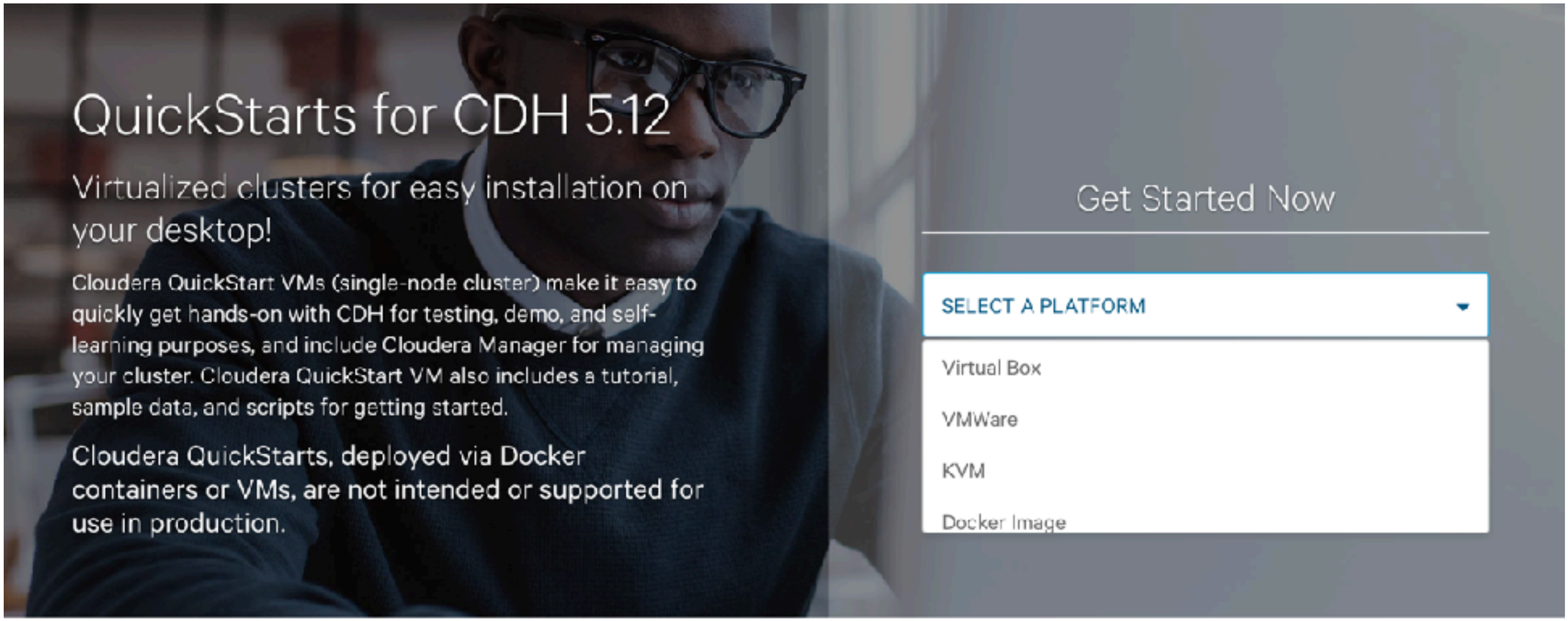
Tutorials | Release Notes | Import on Docker | MD5 : 2a710f236135e620ec8488a1229af07e — DOWNLOAD FOR DOCKER

Hortonworks Sandbox in the Cloud

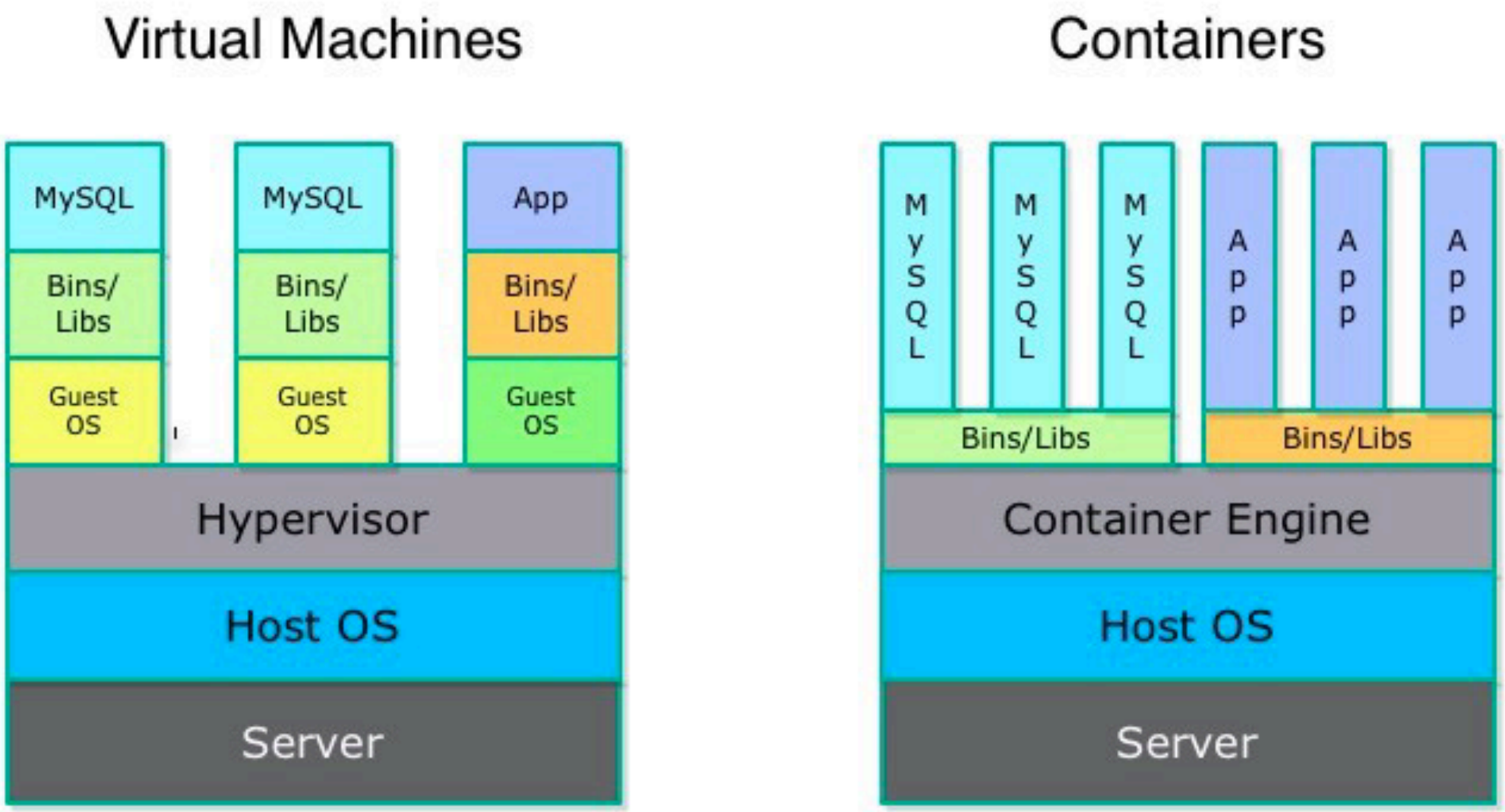
HDP 2.4 on Azure with Hortonworks Sandbox

Tutorial: Sandbox on Azure | Try it one month for free — ONE MONTH TRIAL

http://www.cloudera.com/downloads.html



Install a Docker Engine



Pull Cloudera Quickstart

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

$ sudo docker pull cloudera/quickstart:latest

```
ubuntu@ip-172-31-30-238:~$ sudo docker pull cloudera/quickstart:latest
latest: Pulling from cloudera/quickstart
2cda82941cb7: Already exists
Digest: sha256:f91bee4cdfa2c92ea3652929a22f729d4d13fc838b00f120e630f91c941acb63
Status: Downloaded newer image for cloudera/quickstart:latest
ubuntu@ip-172-31-30-238:~$
```

$ sudo docker images

```
ubuntu@ip-172-31-30-238:~$ sudo docker images
REPOSITORY            TAG                 IMAGE ID            CREATED
  VIRTUAL SIZE
cloudera/quickstart   latest              2cda82941cb7        9 weeks ago
  6.336 GB
```

Run Cloudera quickstart

$ sudo docker run --hostname=quickstart.cloudera --privileged=true -t -i [OPTIONS] [IMAGE] /usr/bin/docker-quickstart

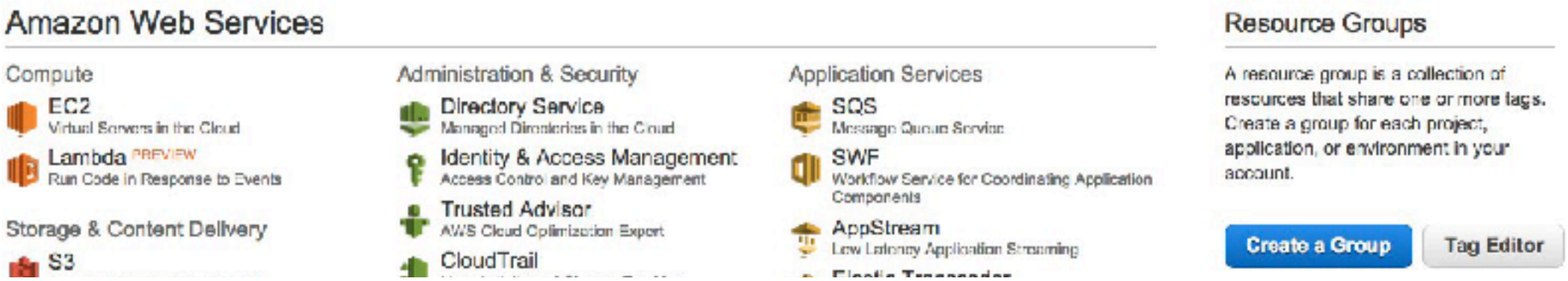
<u>Example</u>: $ sudo docker run --hostname=quickstart.cloudera --privileged=true -t -i -p 8888:8888 cloudera/quickstart /usr/bin/docker-quickstart

```
ubuntu@ip-172-31-30-238:~$  sudo docker run --hostname=quickstart.cloudera --pri
vileged=true -t -i  -p 8888:8888  -p 7180:7180  cloudera/quickstart  /usr/bin/do
cker-quickstart
Starting mysqld:                                           [  OK  ]

if [ "$1" == "start" ] ; then
    if [ "${EC2}" == 'true' ]; then
        FIRST_BOOT_FLAG=/var/lib/cloudera-quickstart/.ec2-key-installed
        if [ ! -f "${FIRST_BOOT_FLAG}" ]; then
            METADATA_API=http://169.254.169.254/latest/meta-data
            KEY_URL=${METADATA_API}/public-keys/0/openssh-key
```

Install Cloudera Cluster on AWS

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

Launch a virtual server on EC2 Amazon Web Services



Select an Amazon Machine Image (AMI) and Ubuntu Server 14.04 LTS (PV)

Install Cloudera Cluster on AWS

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

Choose m3.xlarge Type virtual server

1. Choose AMI | 2. Choose Instance Type | 3. Configure Instance | 4. Add Storage | 5. Tag Instance | 6. Configure Security Group | 7. Review

Step 2: Choose an Instance Type

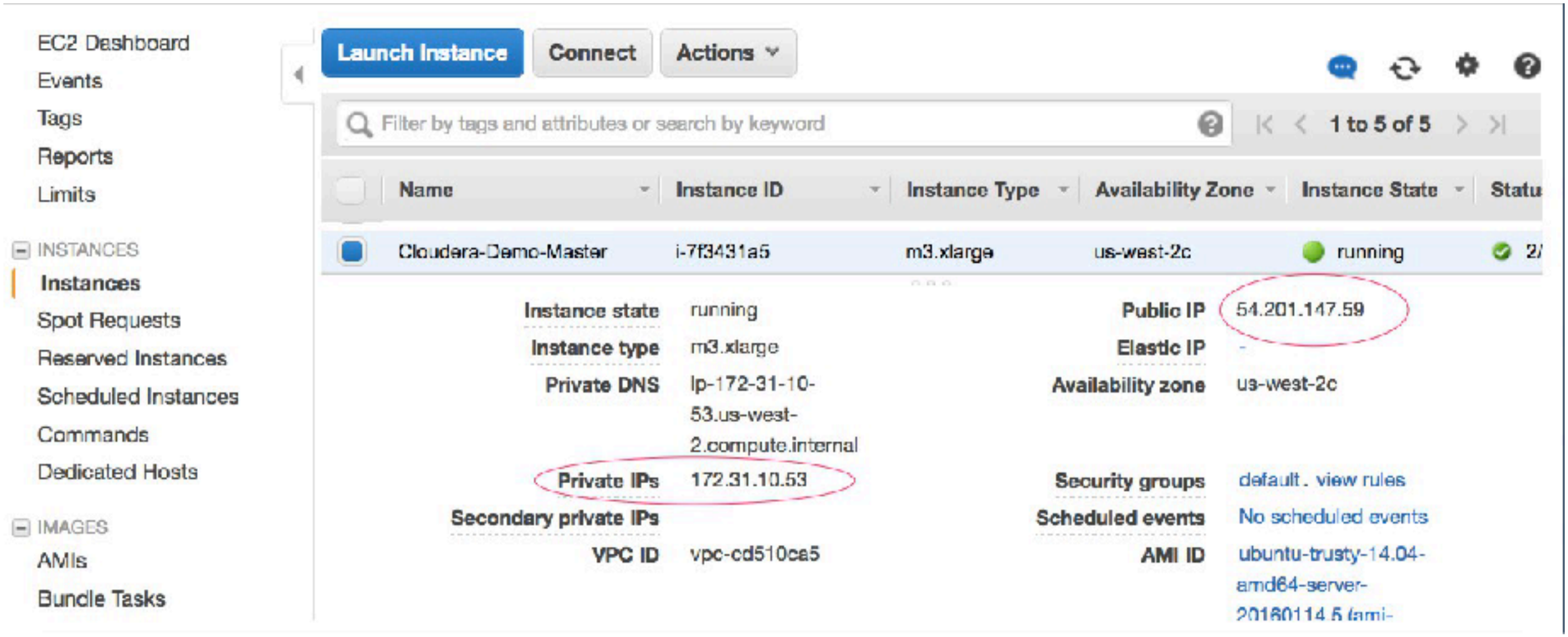
☐	Micro instances	t1.micro Free tier eligible	1	0.613	EBS only	-	Very Low
⊘	General purpose	t2.micro Free tier eligible	1	1	EBS only	-	Low to Moderate
⊘	General purpose	t2.small	1	2	EBS only	-	Low to Moderate
⊘	General purpose	t2.medium	2	4	EBS only	-	Low to Moderate
☐	General purpose	m3.medium	1	3.75	1 x 4 (SSD)	-	Moderate
☐	General purpose	m3.large	2	7.5	1 x 32 (SSD)	-	Moderate
☑	General purpose	m3.xlarge	4	15	2 x 40 (SSD)	Yes	High
☐	General purpose	m3.2xlarge	8	30	2 x 80 (SSD)	Yes	High

Cancel | Previous | Review and Launch | Next: Configure Instance Details

Install Cloudera Cluster on AWS

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

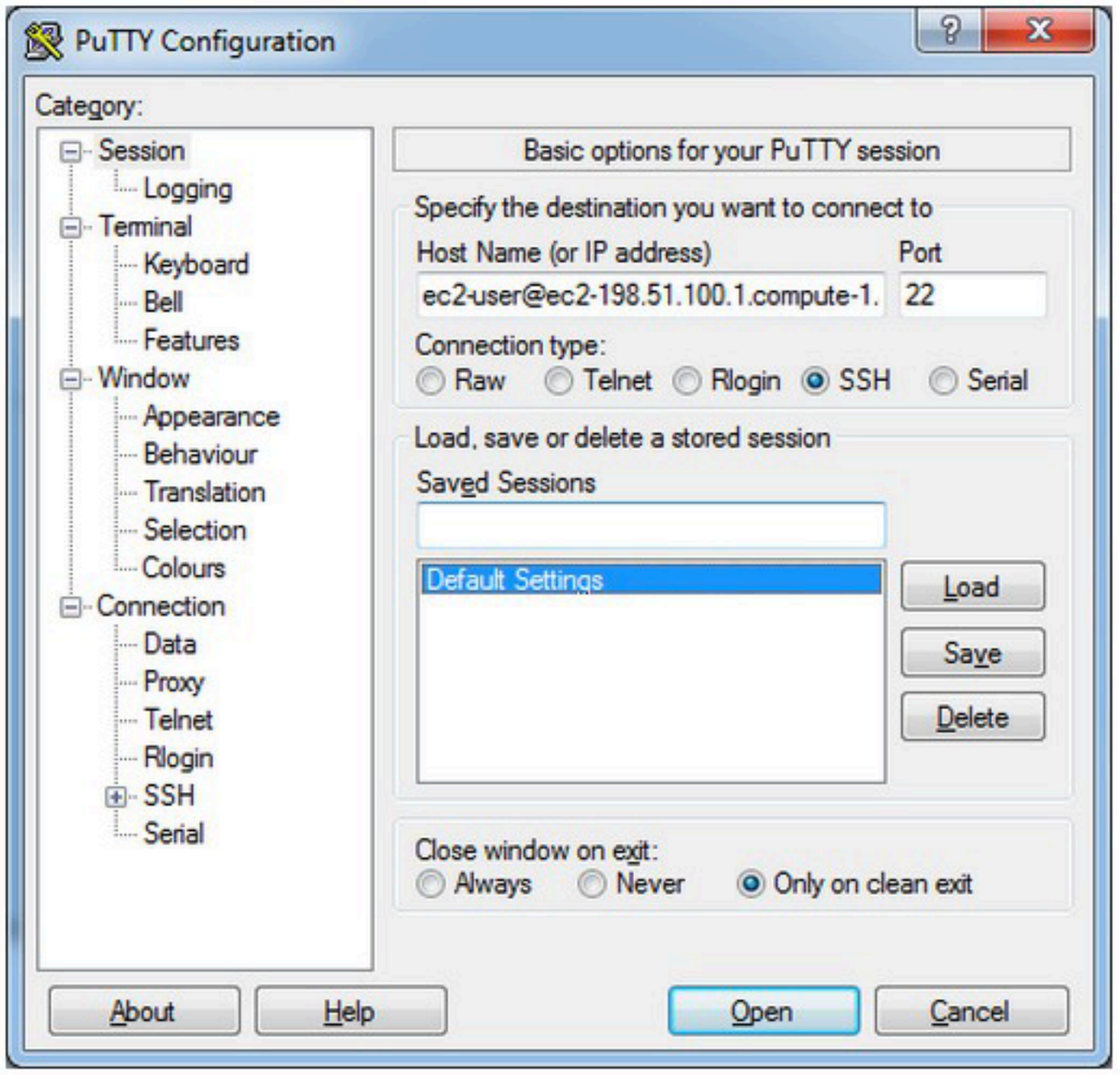
Can also view details of the instance such as Public IP and Private IP



Install Cloudera Cluster on AWS

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

Connect to an instance from Windows using Putty



Install Cloudera Cluster on AWS

CHULALONGKORN
BUSINESS SCHOOL
FLAGSHIP FOR LIFE

Connect to the instance

```
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

_____________________________________________________________________
WARNING! Your environment specifies an invalid locale.
 This can affect your user experience significantly, including the
 ability to manage packages. You may install the locales by running:

   sudo apt-get install language-pack-UTF-8
     or
   sudo locale-gen UTF-8

To see all available language packs, run:
   apt-cache search "^language-pack-[a-z][a-z]$"
To disable this message for all users, run:
   sudo touch /var/lib/cloud/instance/locale-check.skip
_____________________________________________________________________

ubuntu@ip-172-31-1-242:~$
```

Installing Cloudera on EC2

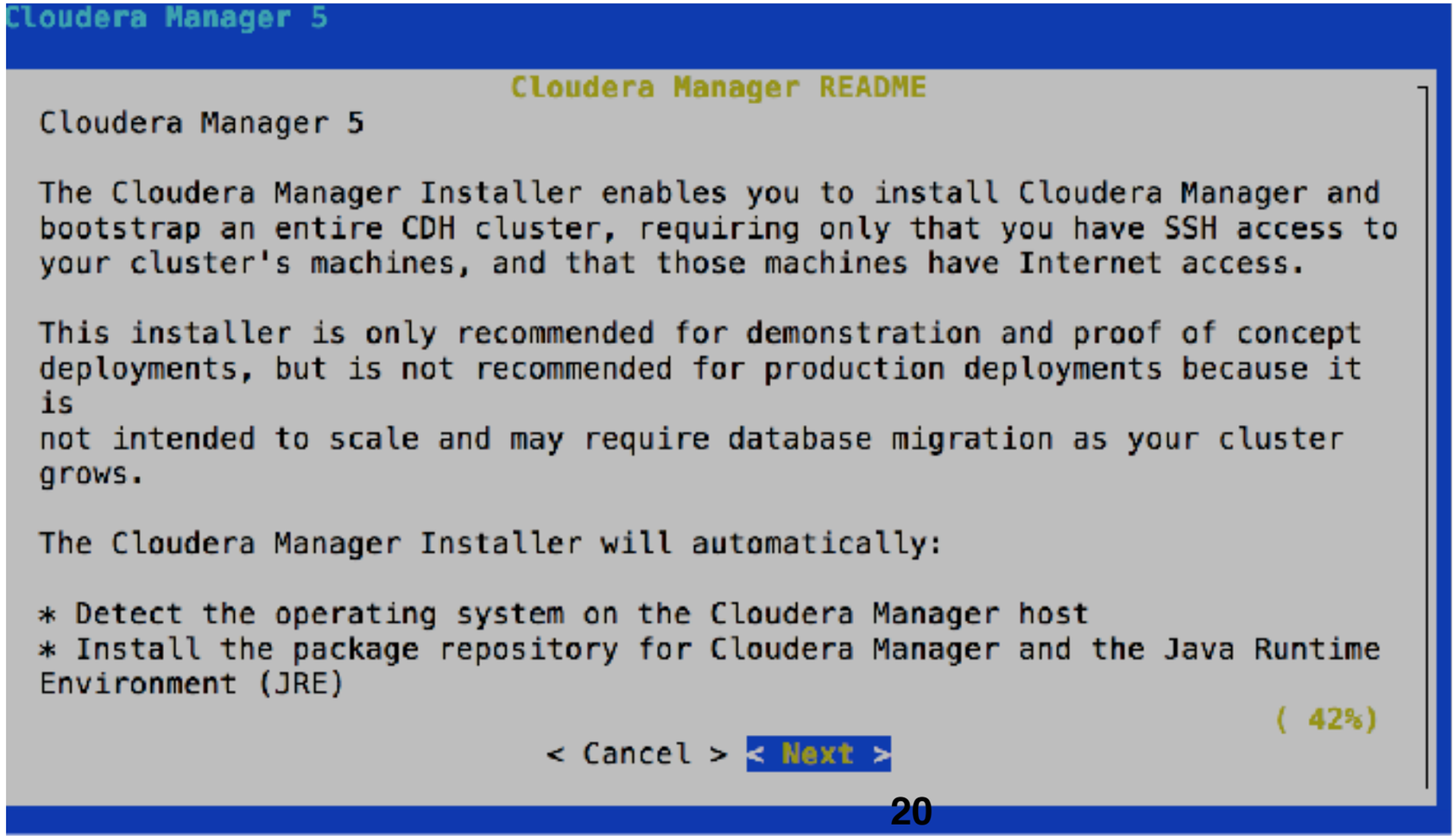
CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

Download Cloudera Manager

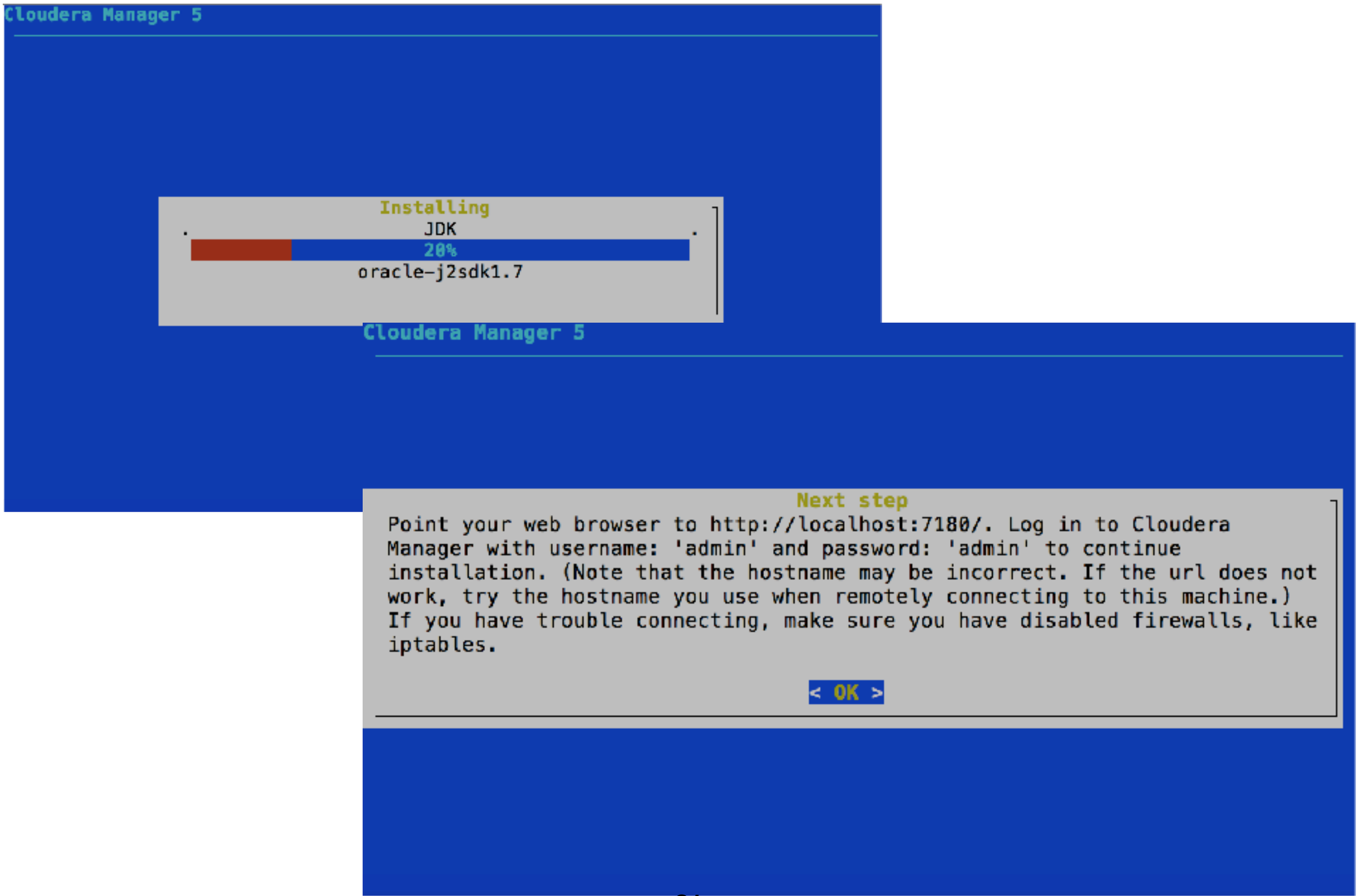
1) Type command > wget http://archive.cloudera.com/cm5/installer/latest/cloudera-manager-installer.bin

2) Type command > chmod u+x cloudera-manager-installer.bin

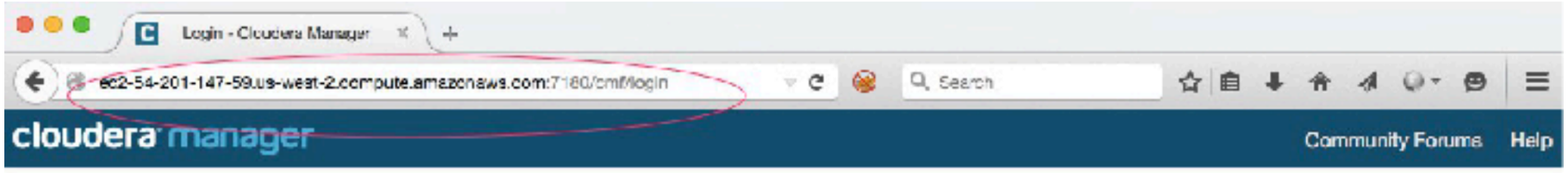
3) Type command > sudo ./cloudera-manager-installer.bin

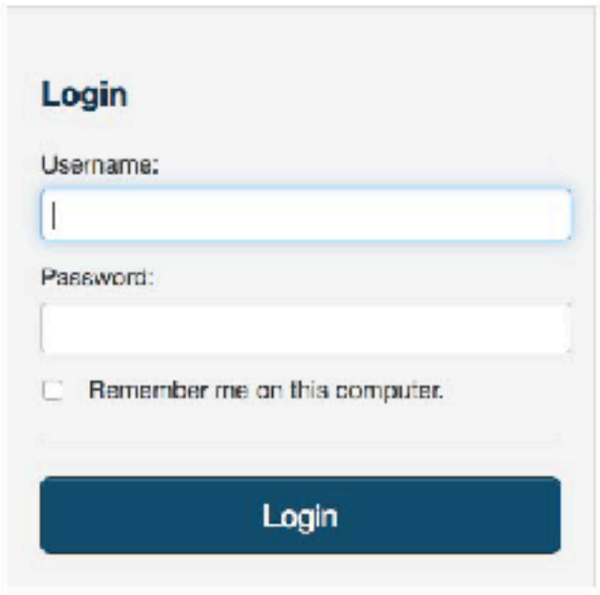


Installing Cloudera on EC2



Wait several minutes for the Cloudera Manager Server to complete its startup. Then running web browser: http:// public-dns: 7180





Welcome to Cloudera Manager

End User License Terms and Conditions

Cloudera Standard License

Version 2015-08-06

END USER LICENSE TERMS AND CONDITIONS

THESE TERMS AND CONDITIONS (THESE "TERMS") APPLY TO YOUR USE OF THE PRODUCTS (AS DEFINED BELOW) PROVIDED BY CLOUDERA, INC. ("CLOUDERA").

PLEASE READ THESE TERMS CAREFULLY.

IF YOU ("YOU" OR "CUSTOMER") PLAN TO USE ANY OF THE PRODUCTS ON BEHALF OF A COMPANY OR OTHER ENTITY, YOU REPRESENT THAT YOU ARE THE EMPLOYEE OR AGENT OF SUCH COMPANY (OR OTHER ENTITY) AND YOU HAVE THE AUTHORITY TO ACCEPT ALL OF THE TERMS AND CONDITIONS SET FORTH IN AN ACCEPTED R
"AGREEMENT") ON BEHALF OF SUCH COMPANY (OR OTHER E

BY USING ANY OF THE PRODUCTS, YOU ACKNOWLEDGE AND
(A) YOU HAVE READ ALL OF THE TERMS AND CONDITIONS OF
(B) YOU UNDERSTAND ALL OF THE TERMS AND CONDITIONS
(C) YOU AGREE TO BE LEGALLY BOUND BY ALL OF THE TERM

☑ Yes, I accept the End User License Terms and Conditions.

Back

cloudera manager — Support — admin

Welcome to Cloudera Manager. Which edition do you want to deploy?

Upgrading to **Cloudera Enterprise Data Hub Edition** provides important features that help you manage and monitor your Hadoop clusters in mission-critical environments.

	Cloudera Express	Cloudera Enterprise Data Hub Edition Trial	Cloudera Enterprise
	✔		
License	Free	60 Days After the trial period, the product will continue to function as Cloudera Express. Your cluster and your data will remain unaffected.	Annual Subscription Upload License Cloudera Enterprise is available in three editions: • Basic Edition • Flex Edition • Data Hub Edition
Node Limit	Unlimited	Unlimited	Unlimited
CDH	✔	✔	✔
Core Cloudera Manager Features	✔	✔	✔
Advanced Cloudera Manager Features		✔	✔
Cloudera Navigator		✔	✔

Continue

cloudera manager

Support admin

Thank you for choosing Cloudera Manager and CDH.

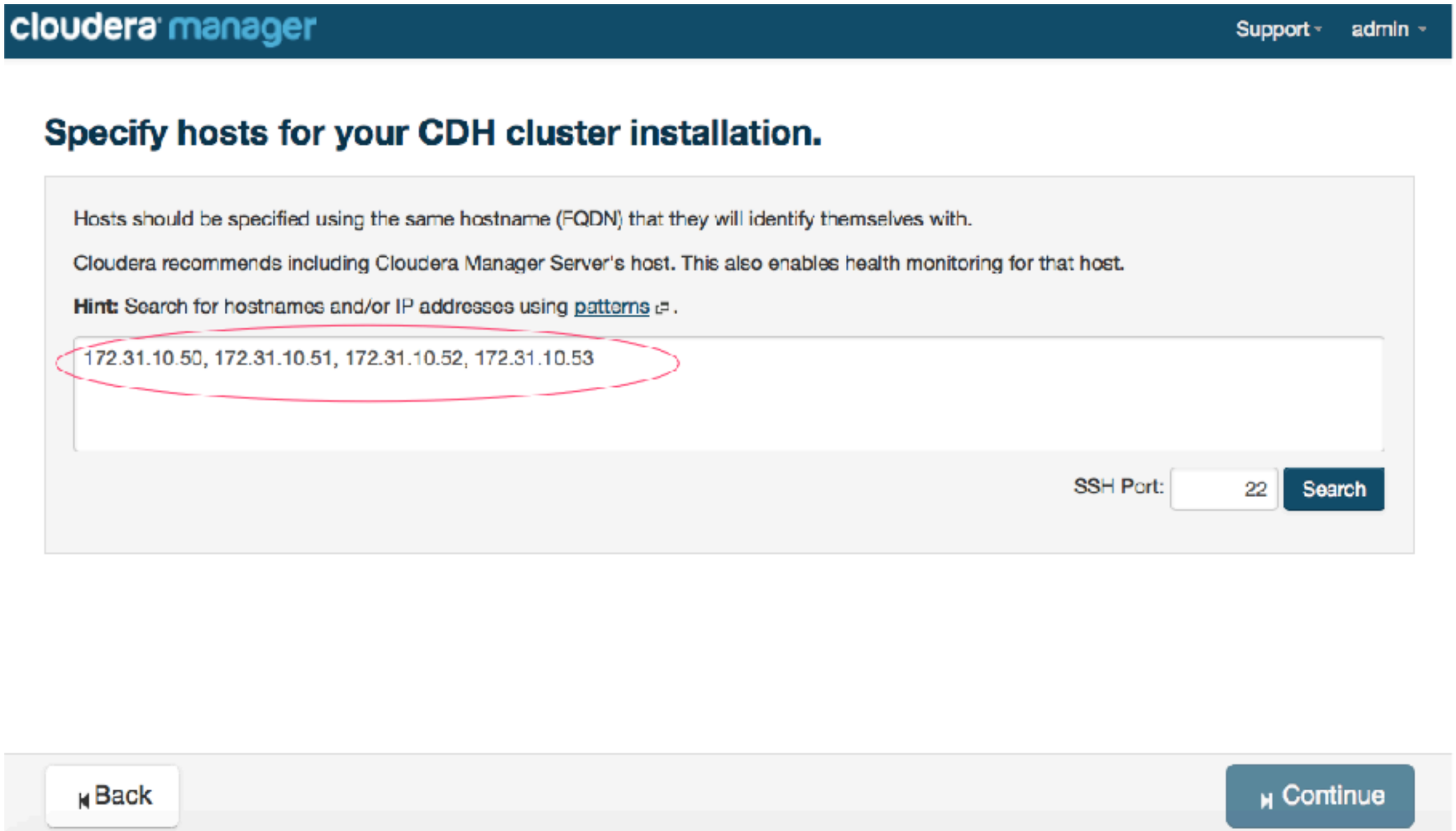
This installer will install **Cloudera Express 5.4.0** and enable you to later choose packages for the services below (there may be some license implications).

- Apache Hadoop (Common, HDFS, MapReduce, YARN)
- Apache HBase
- Apache ZooKeeper
- Apache Oozie
- Apache Hive
- Hue (Apache licensed)
- Apache Flume
- Cloudera Impala (Apache licensed)
- Apache Sentry
- Apache Sqoop
- Cloudera Search (Apache licensed)
- Apache Spark

You are using Cloudera Manager to install and configure your system. You can learn more about Cloudera Manager by clicking on the **Support** menu above.

Continue

Provide your instances <private ip> addresses in the cluster



cloudera manager — Support - admin -

Specify hosts for your CDH cluster installation.

Hosts should be specified using the same hostname (FQDN) that they will identify themselves with.

Cloudera recommends including Cloudera Manager Server's host. This also enables health monitoring for that host.

Hint: Search for hostnames and/or IP addresses using patterns.

4 hosts scanned, 4 running SSH. New Search

Expanded Query	Hostname (FQDN)
172.31.10.50	ip-172-31-10-50.us-west-2.compute.i
172.31.10.51	ip-172-31-10-51.us-west-2.compute.i
172.31.10.52	ip-172-31-10-52.us-west-2.compute.i
172.31.10.53	ip-172-31-10-53.us-west-2.compute.i

Back

cloudera manager — Support - admin -

Cluster Installation

Select Repository

Cloudera recommends the use of parcels for installation over packages, because parcels enable Cloudera Manager to easily manage the software on your cluster, automating the deployment and upgrade of service binaries. Electing not to use parcels will require you to manually upgrade packages on all hosts in your cluster when software updates are available, and will prevent you from using Cloudera Manager's rolling upgrade capabilities.

Choose Method
- Use Packages
- Use Parcels (Recommended) — More Options

Select the version of CDH
- CDH-5.6.0-1.cdh5.6.0.p0.45
- CDH-4.7.1-1.cdh4.7.1.p0.47

Versions of CDH that are too new for this version of Cloudera Manager (5.6.0) will not be shown.

Additional Parcels
- ACCUMULO-1.6.0-1.cdh5.1.4.p0.116

Back — 1 2 3 4 5 6 7 8 — Continue

cloudera manager — Support · admin ·

Cluster Installation

Enable Single User Mode

Only supported for CDH 5.2 and above.

By default, service processes run as distinct users on the system. For example, HDFS DataNodes run as user "hdfs" and HBase RegionServers run as user "hbase." Enabling "single user mode" configures Cloudera Manager to run service processes as a single user by default "cloudera-scm", thereby prioritizing isolation between managed services and the rest of the system over isolation between the managed services.

The major benefit of this option is that the Agent does not run as root. How
directories which in the regular mode are created automatically by the Agent.
must be set up for the configured user.

Switching back and forth between single user mode and regular mode is not

Single User Mode ☐

Back

cloudera manager — Support · admin ·

Cluster Installation

Installation completed successfully.

4 of 4 host(s) completed successfully.

Hostname	IP Address	Progress	Status	
ip-172-31-10-50.us-west-2.compute.internal	172.31.10.50		Installation completed successfully.	Details
ip-172-31-10-51.us-west-2.compute.internal	172.31.10.51		Installation completed successfully.	Details
ip-172-31-10-52.us-west-2.compute.internal	172.31.10.52		Installation completed successfully.	Details
ip-172-31-10-53.us-west-2.compute.internal	172.31.10.53		Installation completed successfully.	Details

Back — 1 2 3 4 5 6 7 8 — Continue

cloudera manager — Support — admin

Cluster Setup

Choose the CDH 5 services that you want to install on your cluster.

Choose a combination of services to install.

- **Core Hadoop**
  HDFS, YARN (MapReduce 2 Included), ZooKeeper, Oozie, Hive, Hue, and Sqoop
- **Core with HBase**
  HDFS, YARN (MapReduce 2 Included), Zo
- **Core with Impala**
  HDFS, YARN (MapReduce 2 Included), Zo
- **Core with Search**
  HDFS, YARN (MapReduce 2 Included), Zo
- **Core with Spark**
  DFS, YARN (MapReduce 2 Included), Zo
- **All Services**
- **Custom Services**
  Choose your own services. Services requi

This wizard will also install the **Cloudera Man**

Back

cloudera manager — Support — admin

Cluster Setup

Customize Role Assignments

You can customize the role assignments for your new cluster here, but if assignments are made incorrectly, such as assigning too many roles to a single host, this can impact the performance of your services. Cloudera does not recommend altering assignments unless you have specific requirements, such as having pre-selected a specific host for a specific role.

You can also view the role assignments by host. View By Host

HBase

Master × 1 New	HBase REST Server	HBase Thrift Server	RegionServer × 3 New
ip-172-31-26-220.us-west-2.compute.i...	Select hosts	Select hosts	Same As DataNode

HDFS

NameNode × 1 New	SecondaryNameNode × 1 New	Balancer × 1 New	HttpFS
ip-172-31-26-220.us-west-2.compute.i...	ip-172-31-26-220.us-west-2.compute.i...	ip-172-31-26-220.us-west-2.compute.i...	Select hosts

NFS Gateway	DataNode × 3 New
Select hosts	ip-172-31-26-[221-223].us-west-2.compute.

Hive

Back — 1 2 3 4 5 6 — Continue

cloudera manager — Support — admin

Cluster Setup

First Run Command

Status: Running Start Time: Jan 20, 4:41:25 PM Abort

Details Completed 5 of 8 step(s). All Failed Only Running Only

Step	Context	Start Time	Duration	Actions
Deploy Client Configuration Successfully deployed all client configurations.	Cluster 1	Jan 20, 4:41:25 PM	15.96s	
Start Cloudera Management Service, ZooKeeper Successfully completed 2 steps.		Jan 20, 4:41:41 PM	25.72s	
Start HDFS Successfully completed 1 steps.				
Start HBase, Solr Successfully completed 2 steps.				
Start YARN (MR2 Included), Key-Value Store Indexer Successfully completed 2 steps.				
Start Spark 0/1 steps completed.				

Back

Creating Hive Metastore Database
Created Hive Metastore Database.
Details

Creating Hive user directory
Successfully created HDFS directory.
Details

Creating Hive warehouse directory
Successfully created HDFS directory.
Details

Starting Hive Service
Service started successfully.
Details

Creating Oozie database
Oozie database created successfully.
Details

Installing Oozie ShareLib in HDFS
Successfully installed Oozie ShareLib.
Details

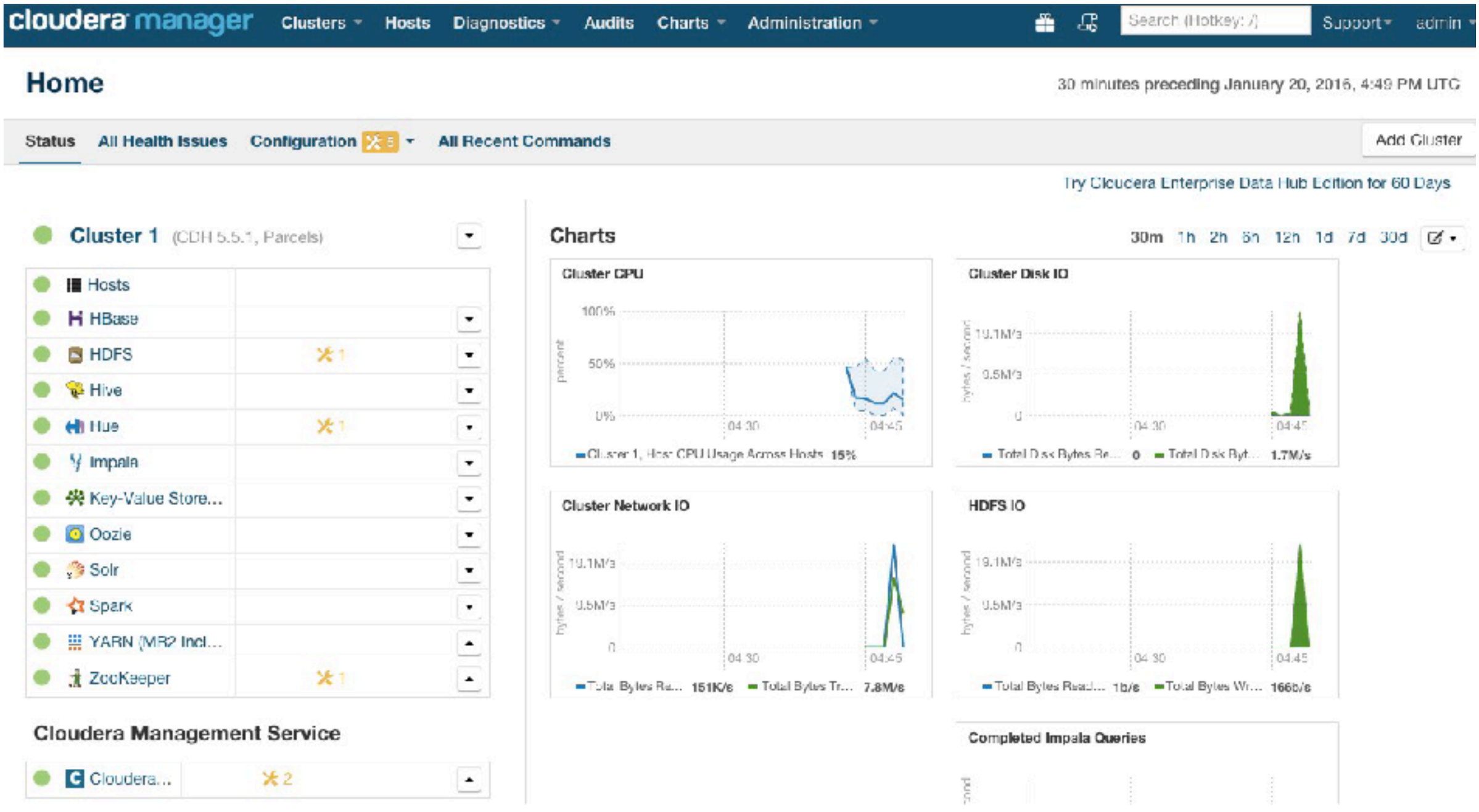
Starting Oozie Service
Service started successfully.
Details

Starting Hue Service
Service started successfully.
Details

Deploying Client Configuration
Successfully deployed all client configurations.
Details

Cloudera Manager

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE



Hadoop as a Service

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

Recommended

Amazon EMR

Azure HDInsight

Amazon EMR

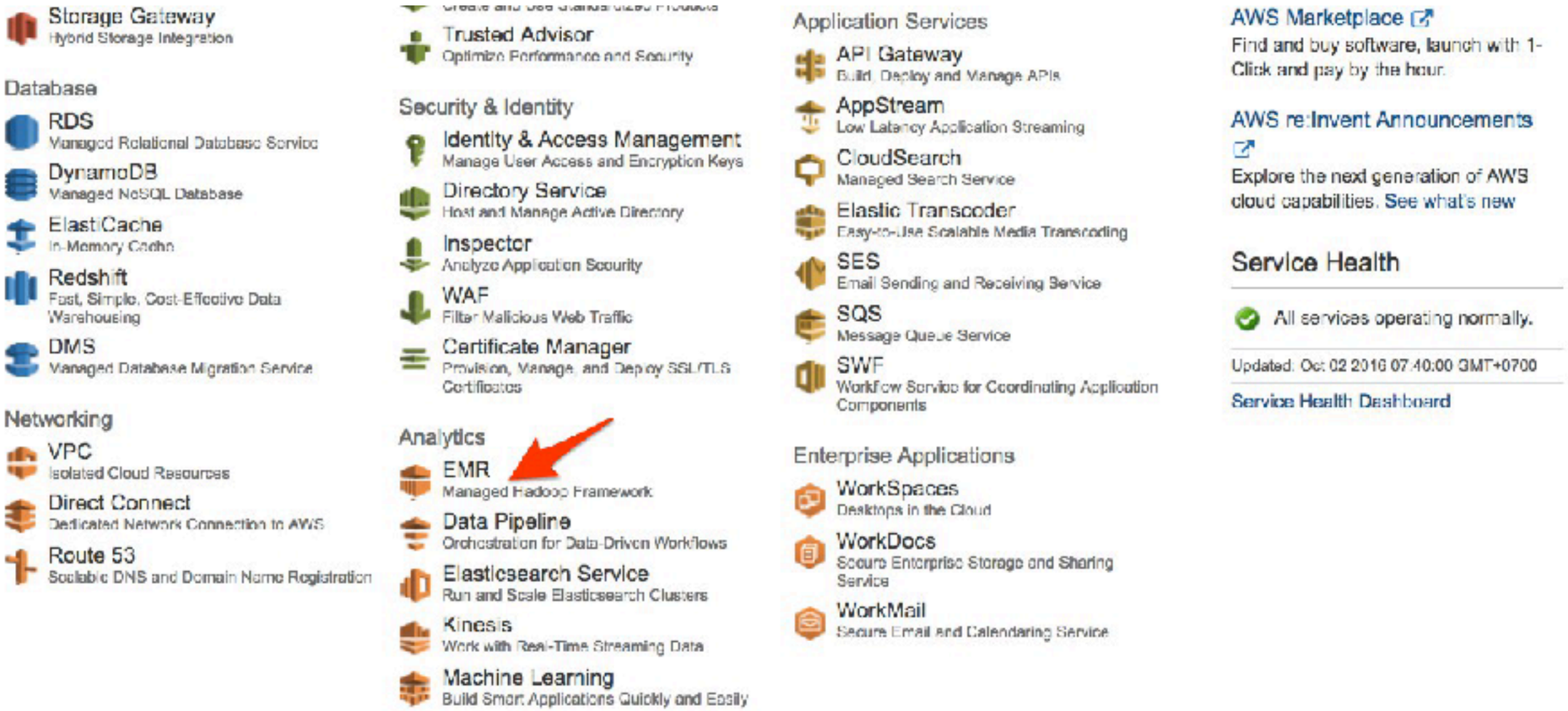
Install a cluster on 4 EC2 AWS

Azure HD Insight

Install a cluster on 4 Vitual Servers Microsoft Azure

Launch a EMR Cluster

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE



AWS | Services | Edit | bobbylovemovie | N. Virginia | Support

Create Cluster - Quick Options

Go to advanced options

General Configuration

Cluster name train_bigdata

Logging

S3 folder s3://aws-logs-438211486020-us-east-1/elasticmapreduce/

Launch mode Cluster | Step execution

Software configuration

Vendor Amazon | MapR

Release emr-5.0.0

Applications

- Core Hadoop: Hadoop 2.7.2 with Ganglia 3.7.2, Hive 2.1.0, Hue 3.10.0, Mahout 0.12.2, Pig 0.16.0, and Tez 0.8.4
- HBase: HBase 1.2.2 with Ganglia 3.7.2, Hadoop 2.7.2, Hive 2.1.0, Hue 3.10.0, Phoenix 4.7.0, and ZooKeeper 3.4.8
- Presto: Presto 0.150 with Hadoop 2.7.2 HDFS and Hive 2.1.0 Metastore
- Spark: Spark 2.0.0 on Hadoop 2.7.2 YARN with Ganglia 3.7.2 and Zeppelin 0.6.1

Hardware configuration

Instance type m3.xlarge

Number of instances 4 (1 master and 3 core nodes)

Connect to the instance

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

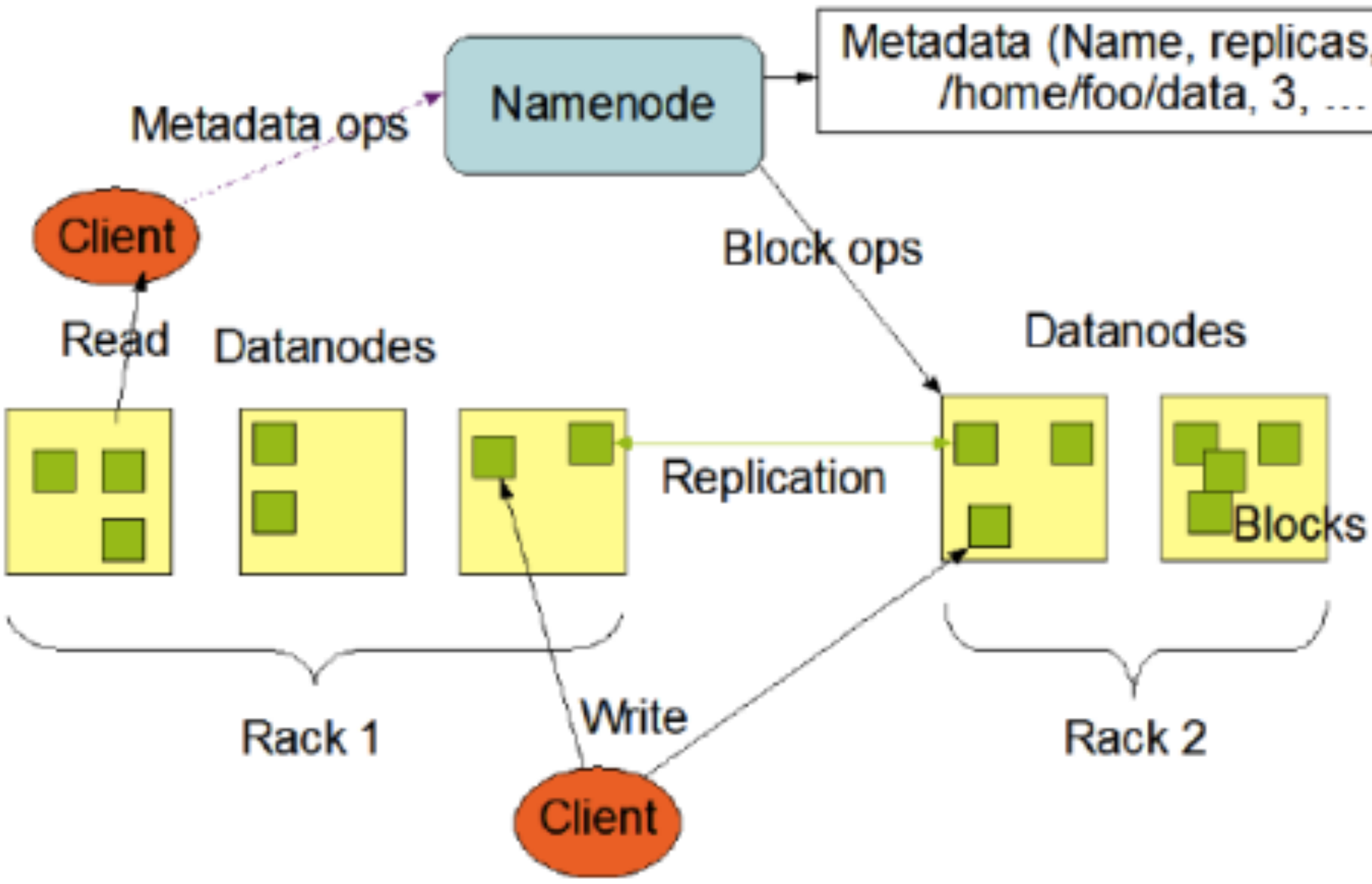
```
https://aws.amazon.com/amazon-linux-ami/2016.03-release-notes/
26 package(s) needed for security, out of 34 available
Run "sudo yum update" to apply all updates.
Amazon Linux version 2016.09 is available.
-bash: warning: setlocale: LC_CTYPE: cannot change locale (UTF-8): No such file
or directory

EEEEEEEEEEEEEEEEEEE MMMMMMMM           MMMMMMMM RRRRRRRRRRRRRRR
E::::::::::::::::::E M:::::::M         M:::::::M R::::::::::::::R
EE:::::EEEEEEEEE:::E M::::::::M       M::::::::M R:::::RRRRRR:::::R
  E::::E       EEEEE M:::::::::M     M:::::::::M RR::::R      R::::R
  E::::E             M::::::M:::M   M:::M::::::M   R:::R      R::::R
  E:::::EEEEEEEEEE   M:::::M M:::M M:::M M:::::M   R:::RRRRRR:::::R
  E::::::::::::::E   M:::::M  M:::M:::M  M:::::M   R::::::::::RR
  E:::::EEEEEEEEEE   M:::::M   M:::::M   M:::::M   R:::RRRRRR::::R
  E::::E             M:::::M    M:::M    M:::::M   R:::R      R::::R
  E::::E       EEEEE M:::::M     MMM     M:::::M   R:::R      R::::R
EE:::::EEEEEEEE:::E M:::::M             M:::::M   R:::R      R::::R
E::::::::::::::::::E M:::::M             M:::::M RR::::R      R::::R
EEEEEEEEEEEEEEEEEEE MMMMMMM             MMMMMMM RRRRRRR      RRRRRR

[hadoop@ip-172-31-39-165 ~]$
```

Hadoop File System (HDFS)

HDFS

- Default storage for the Hadoop cluster
- Data is distributed and replicated over multiple machines
- Designed to handle very large files with streaming data access patterns.
- NameNode/DataNode
- Master/slave architecture (1 master 'n' slaves)
- Designed for large files (64 MB default, but configurable) across all the nodes

HDFS Architecture



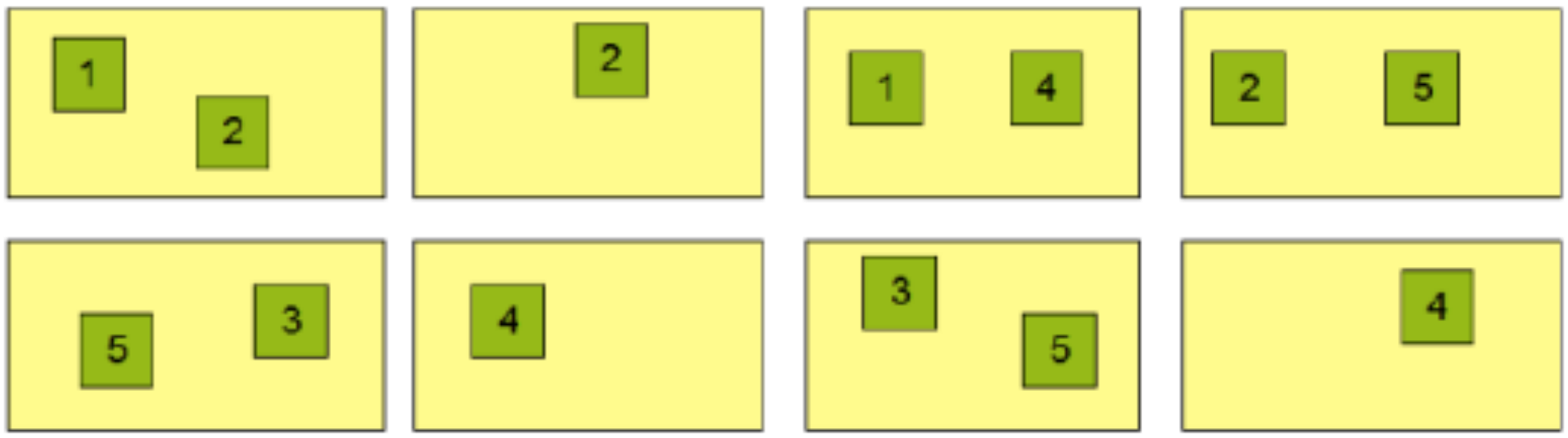
Source Hadoop: Shashwat Shriparv

Data Replication in HDFS

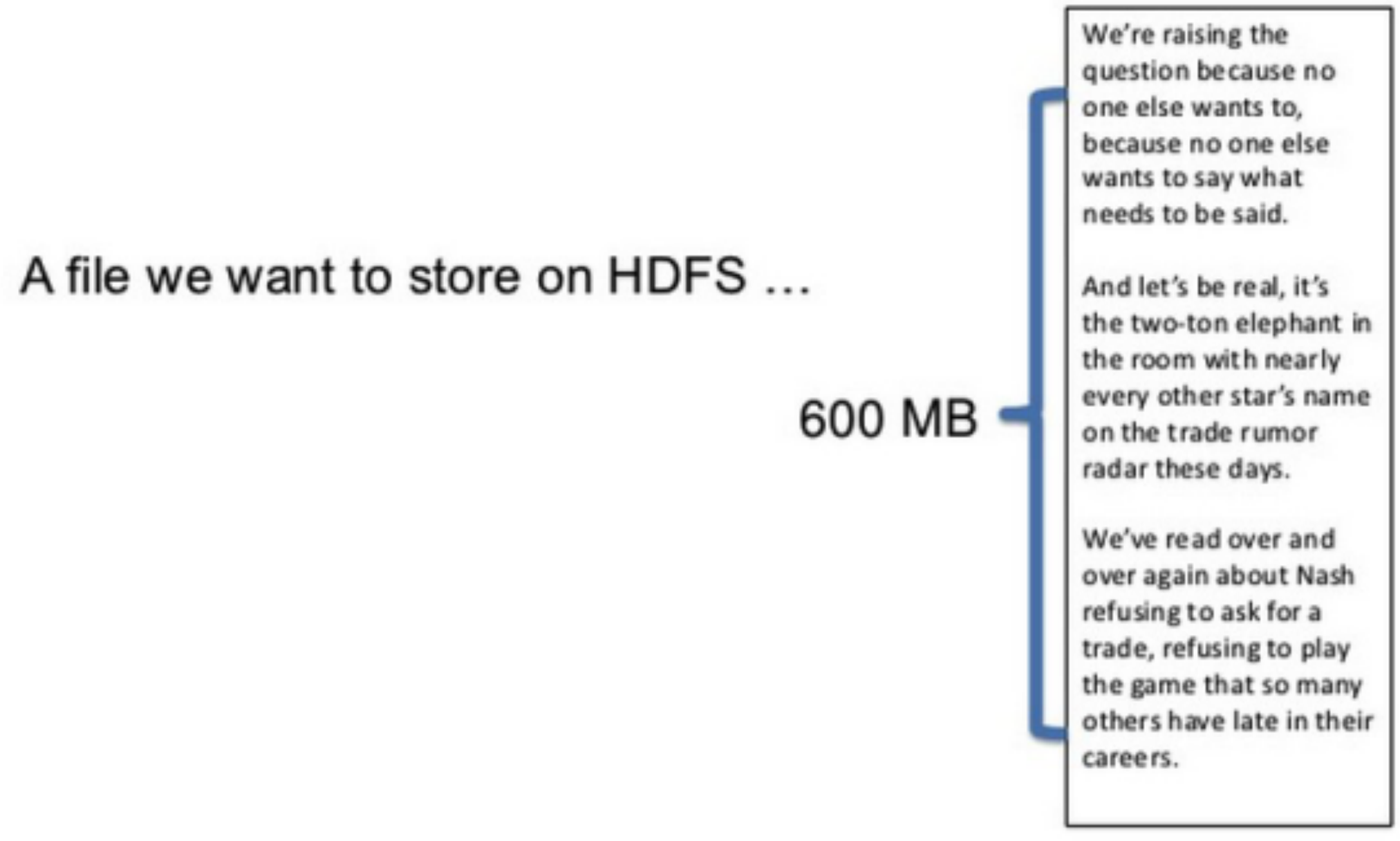
Block Replication

Namenode (Filename, numReplicas, block-ids, …)
/users/sameerp/data/part-0, r:2, {1,3}, …
/users/sameerp/data/part-1, r:3, {2,4,5}, …

Datanodes



How does HDFS work?



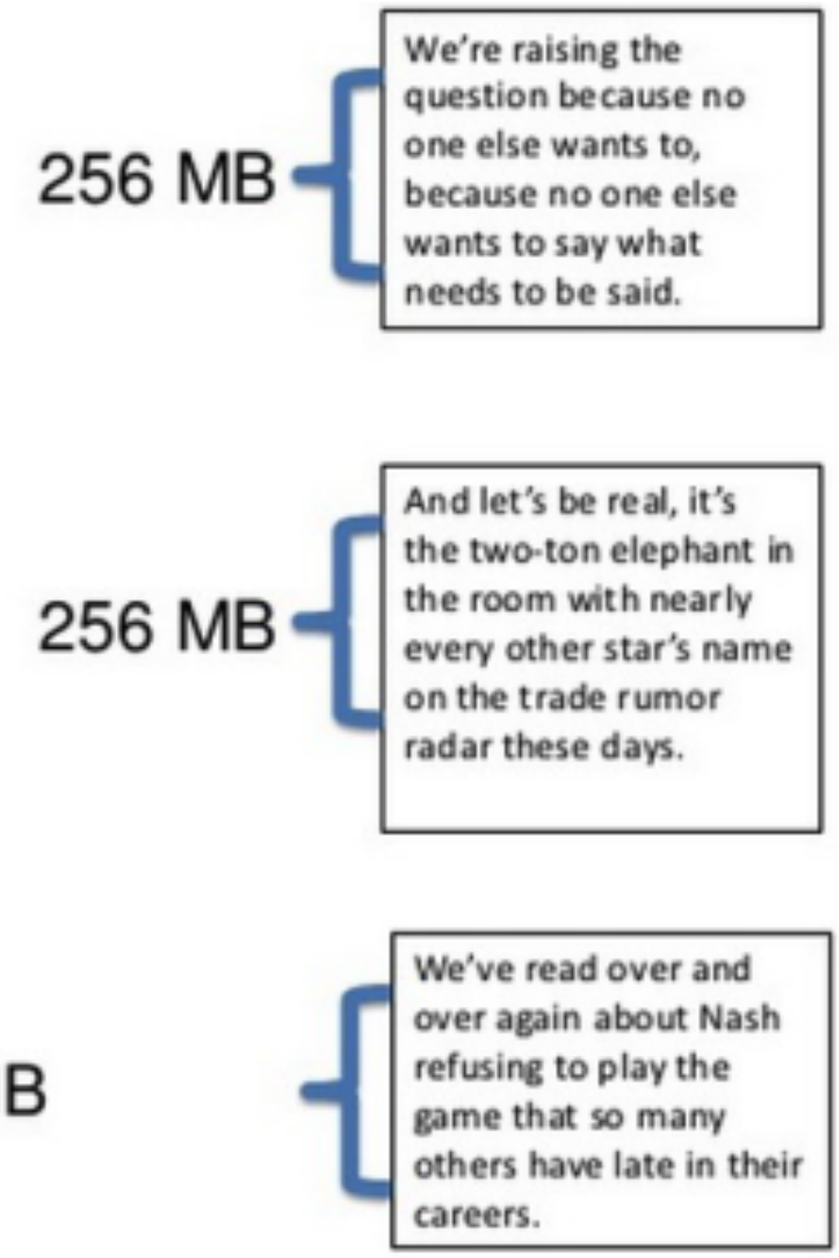
Source Introduction to Apache Hadoop-Pig: PrashantKommireddi

How does HDFS work?

CHULALONGKORN BUSINESS SCHOOL

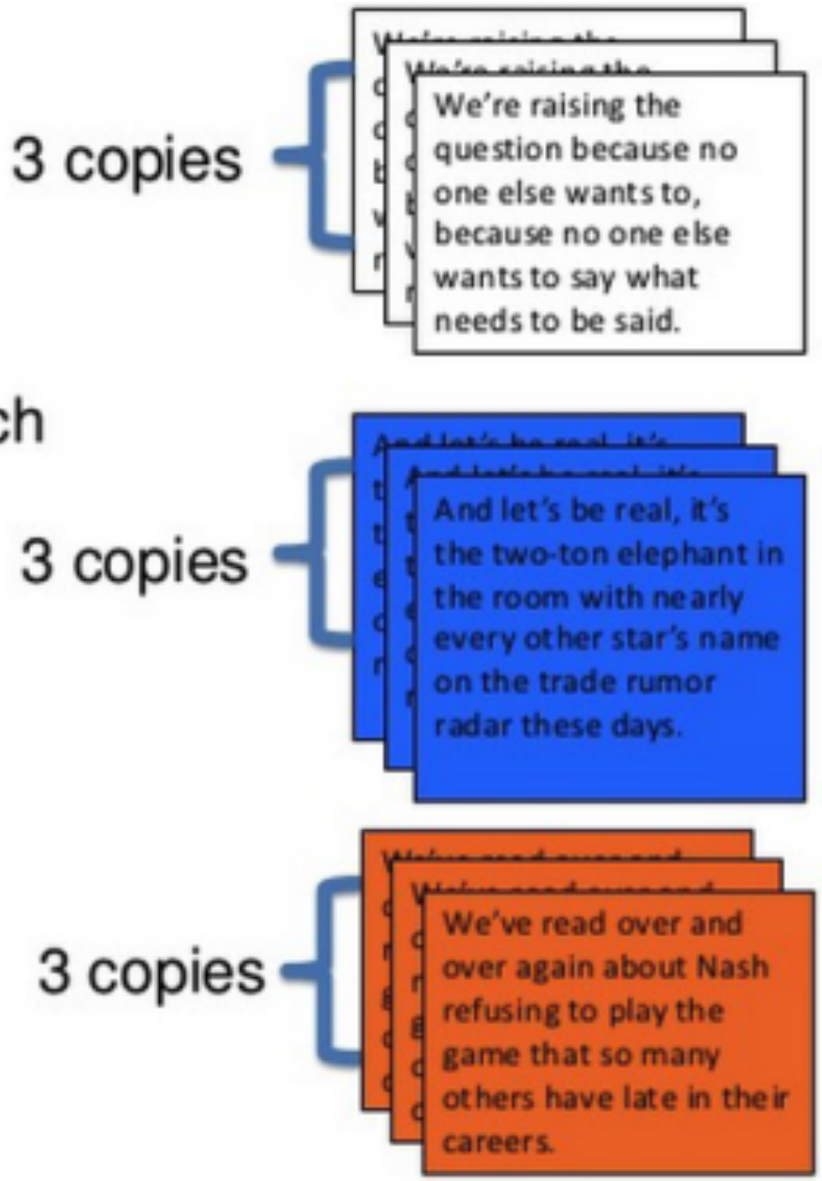
FLAGSHIP FOR LIFE

HDFS Splits file into blocks …



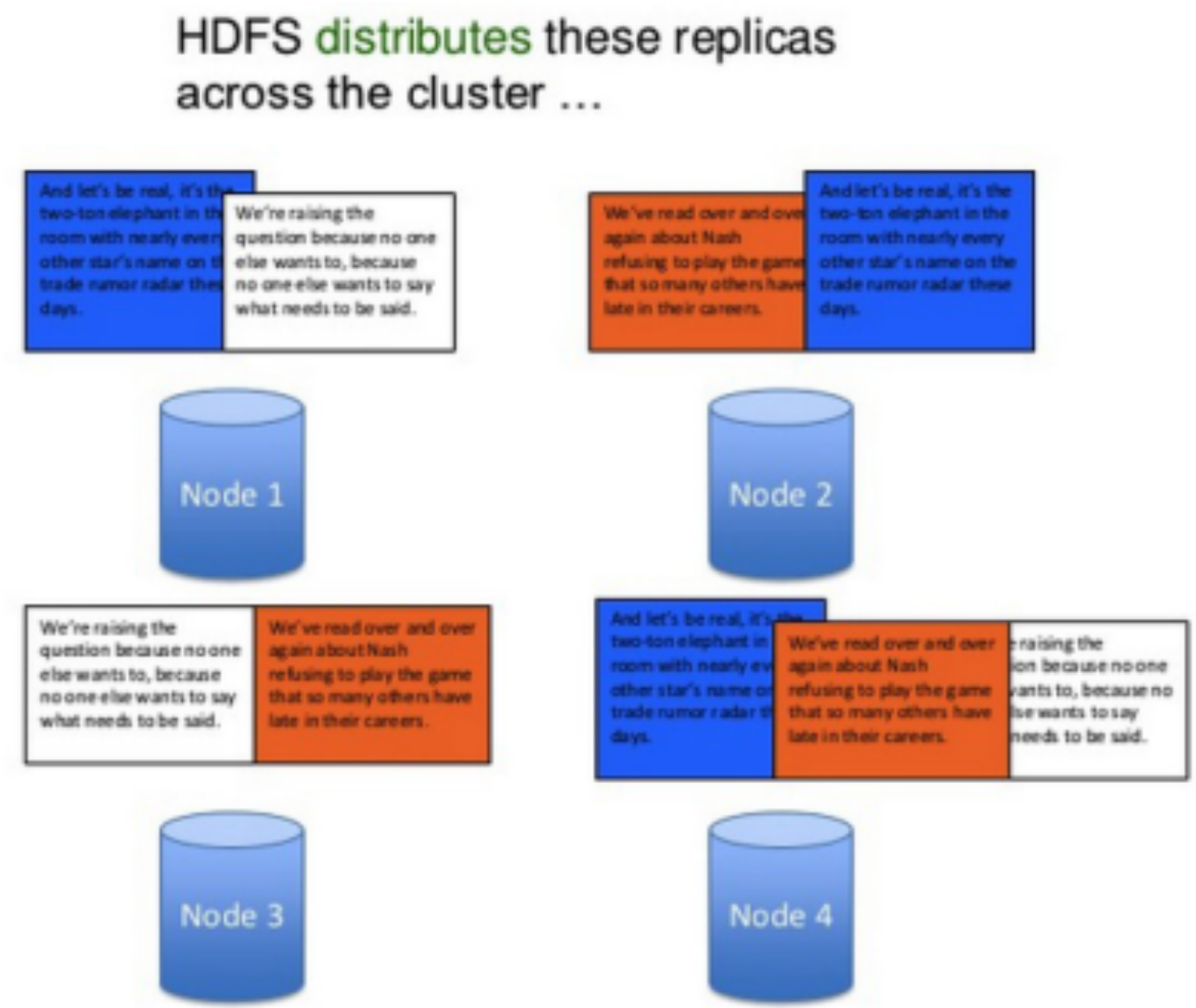
Source Introduction to Apache Hadoop-Pig: PrashantKommireddi

How does HDFS work?



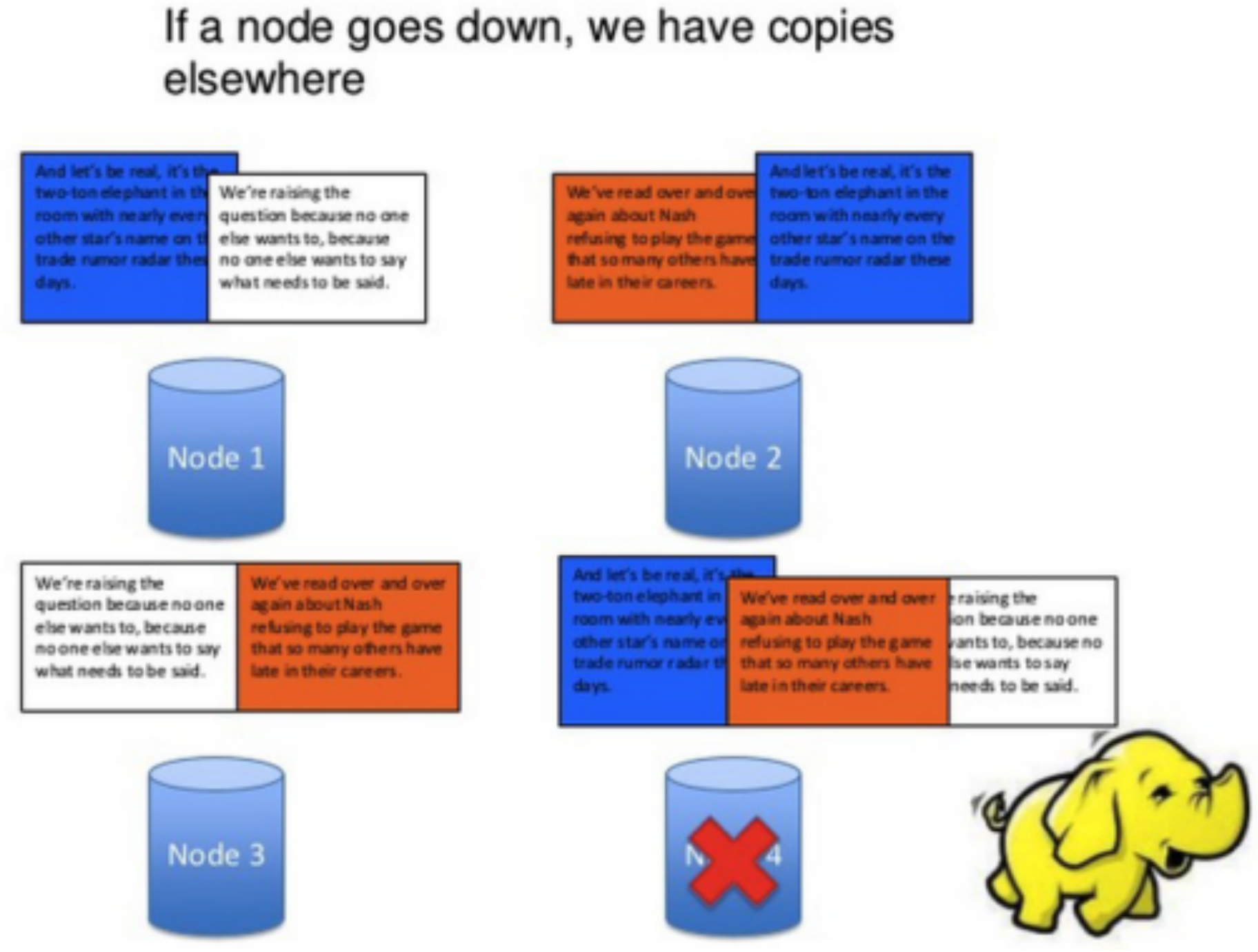
Source Introduction to Apache Hadoop-Pig: PrashantKommireddi

How does HDFS work?



Source Introduction to Apache Hadoop-Pig: PrashantKommireddi

How does HDFS work?

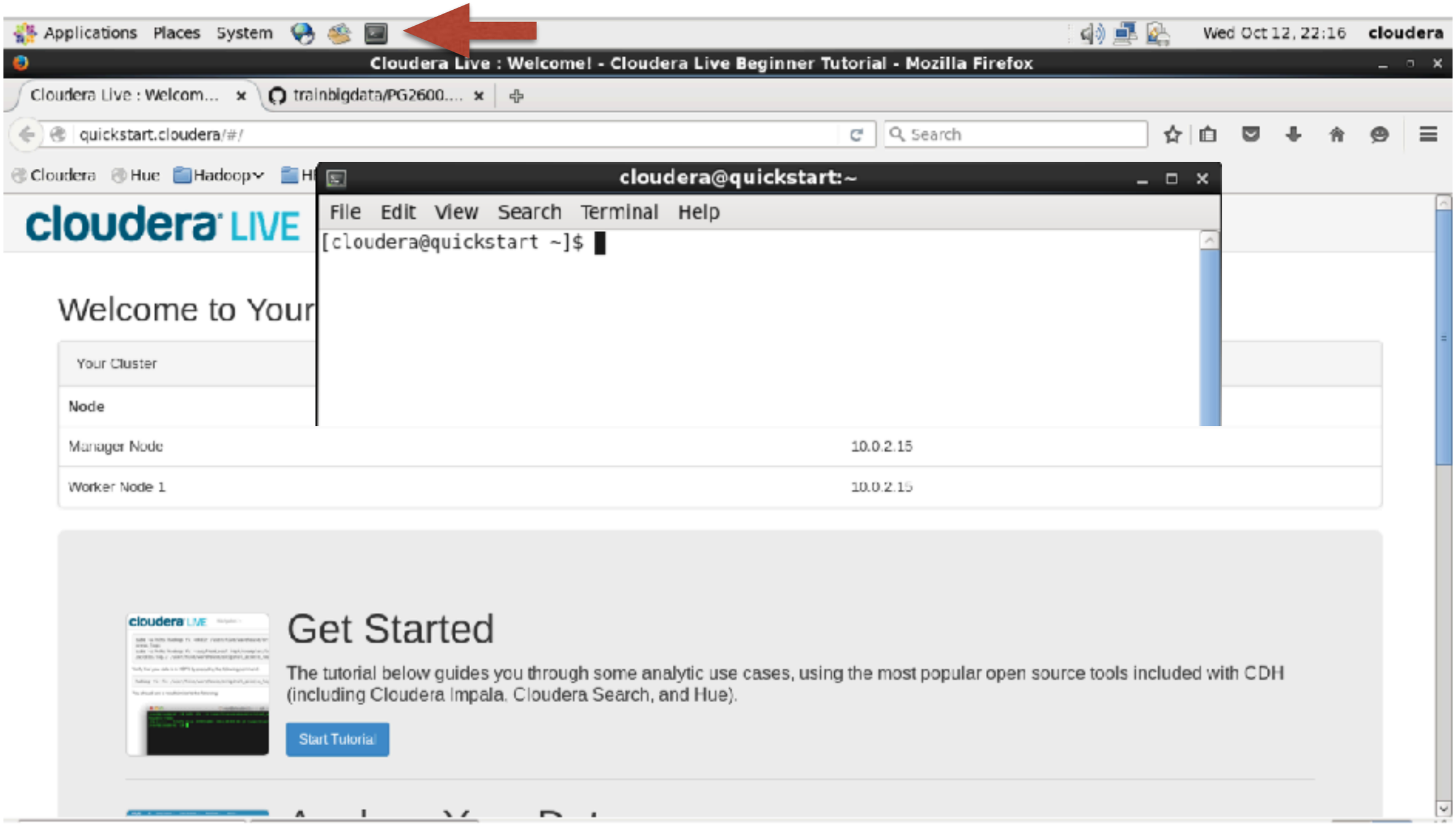


Source Introduction to Apache Hadoop-Pig: PrashantKommireddi

Importing/Exporting Data to HDFS

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

Download an example text file via SSH



Importing/Exporting Data to HDFS

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

$ mkdir guest1

$ cd guest1

$ wget https://github.com/bobbylovemovie/trainbigdata/raw/master/HDFS/PG2600.txt

cloudera@quickstart:~/guest1

File Edit View Search Terminal Help

```
[cloudera@quickstart ~]$ mkdir guest1
[cloudera@quickstart ~]$ cd guest1
[cloudera@quickstart guest1]$ wget https://github.com/bobbylovemovie/trainbigdat
a/raw/master/HDFS/PG2600.txt
--2016-10-12 22:24:11--  https://github.com/bobbylovemovie/trainbigdata/raw/mast
er/HDFS/PG2600.txt
Resolving github.com... 192.30.253.112
Connecting to github.com|192.30.253.112|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://raw.githubusercontent.com/bobbylovemovie/trainbigdata/master/H
DFS/PG2600.txt [following]
--2016-10-12 22:24:13--  https://raw.githubusercontent.com/bobbylovemovie/trainb
igdata/master/HDFS/PG2600.txt
Resolving raw.githubusercontent.com... 151.101.100.133
Connecting to raw.githubusercontent.com|151.101.100.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3226638 (3.1M) [text/plain]
Saving to: “PG2600.txt”

100%[======================================>] 3,226,638   1.25M/s   in 2.5s

2016-10-12 22:24:17 (1.25 MB/s) - “PG2600.txt” saved [3226638/3226638]
```

Upload Data to Hadoop

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

$hadoop fs -mkdir /user/cloudera/input

$hadoop fs -ls /user/cloudera/input

$hadoop fs -rm /user/cloudera/input/*

$hadoop fs -put PG2600.txt /user/cloudera/input/

$hadoop fs -ls /user/cloudera/input

```
[cloudera@quickstart guest1]$ hadoop fs -mkdir /user/cloudera/input
[cloudera@quickstart guest1]$ hadoop fs -rm /user/cloudera/input/*
rm: `/user/cloudera/input/*': No such file or directory
[cloudera@quickstart guest1]$ hadoop fs -put PG2600.txt /user/cloudera/input/
[cloudera@quickstart guest1]$ hadoop fs -ls /user/cloudera/input/*
-rw-r--r--   1 cloudera cloudera    3226638 2016-10-12 23:06 /user/cloudera/inpu
t/PG2600.txt
```

Hadoop syntax for HDFS

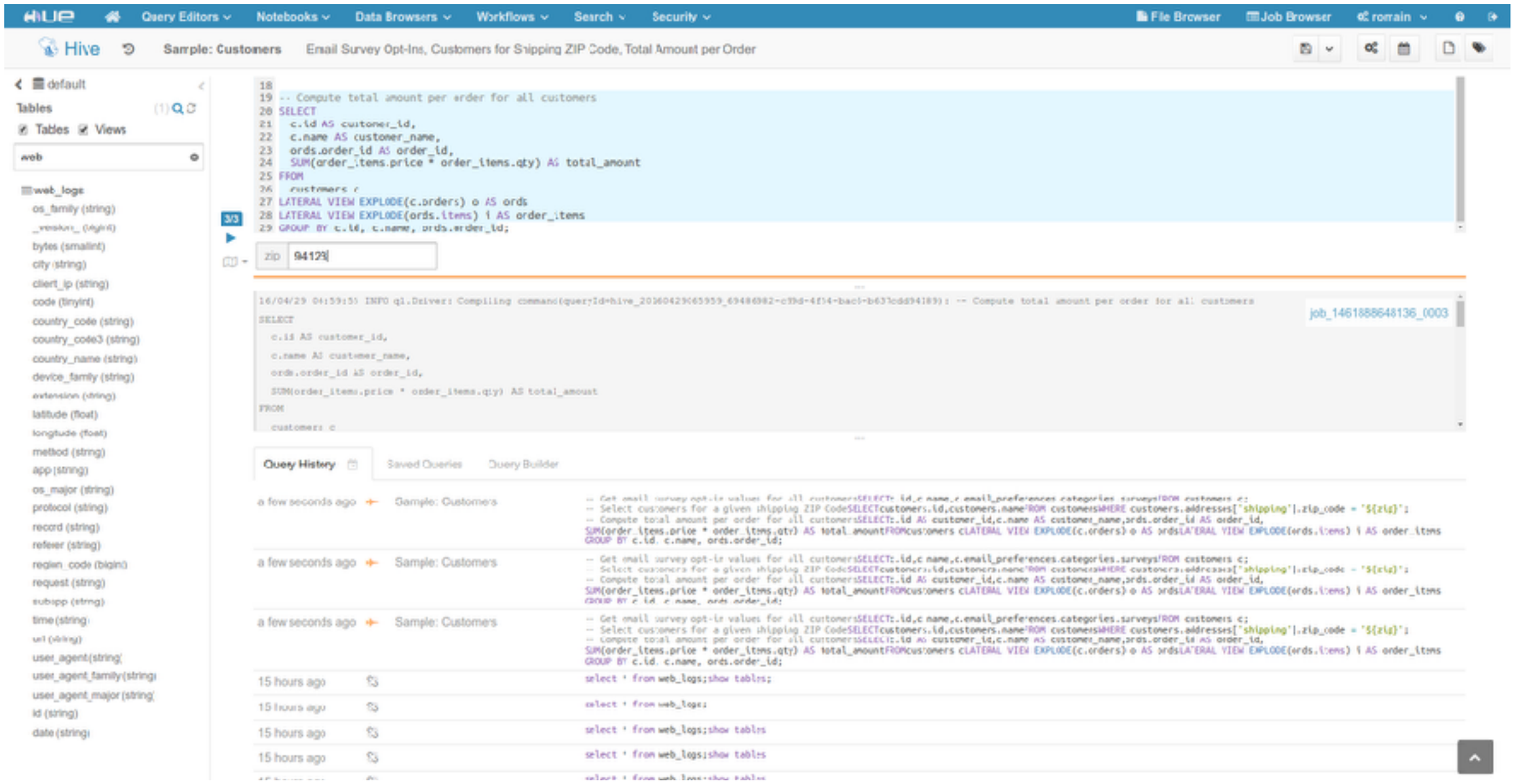
CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

Command	Syntax
Listing of files in a directory	`hadoop fs -ls /user`
Create a new directory	`hadoop fs -mkdir /user/guest/newdirectory`
Copy a file from a local machine to Hadoop	`hadoop fs -put C:\Users\Administrator\Downloads\localfile.csv /user/rajn/newdirectory/hadoopfile.txt`
Copy a file from Hadoop to a local machine	`hadoop fs –get /user/rajn/newdirectory/hadoopfile.txt C:\Users\Administrator\Desktop\`
Tail last few lines of a large file in Hadoop	`hadoop fs -tail /user/rajn/newdirectory/hadoopfile.txt`
View the complete contents of a file in Hadoop	`hadoop fs -cat /user/rajn/newdirectory/hadoopfile.txt`
Remove a complete directory from Hadoop	`hadoop fs –rm -r /user/rajn/newdirectory`
Check the Hadoop filesystem space utilization	`hadoop fs –du /`

Importing/Exporting Data to HDFS

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

Hue - Hadoop User Experience - The Apache Hadoop UI



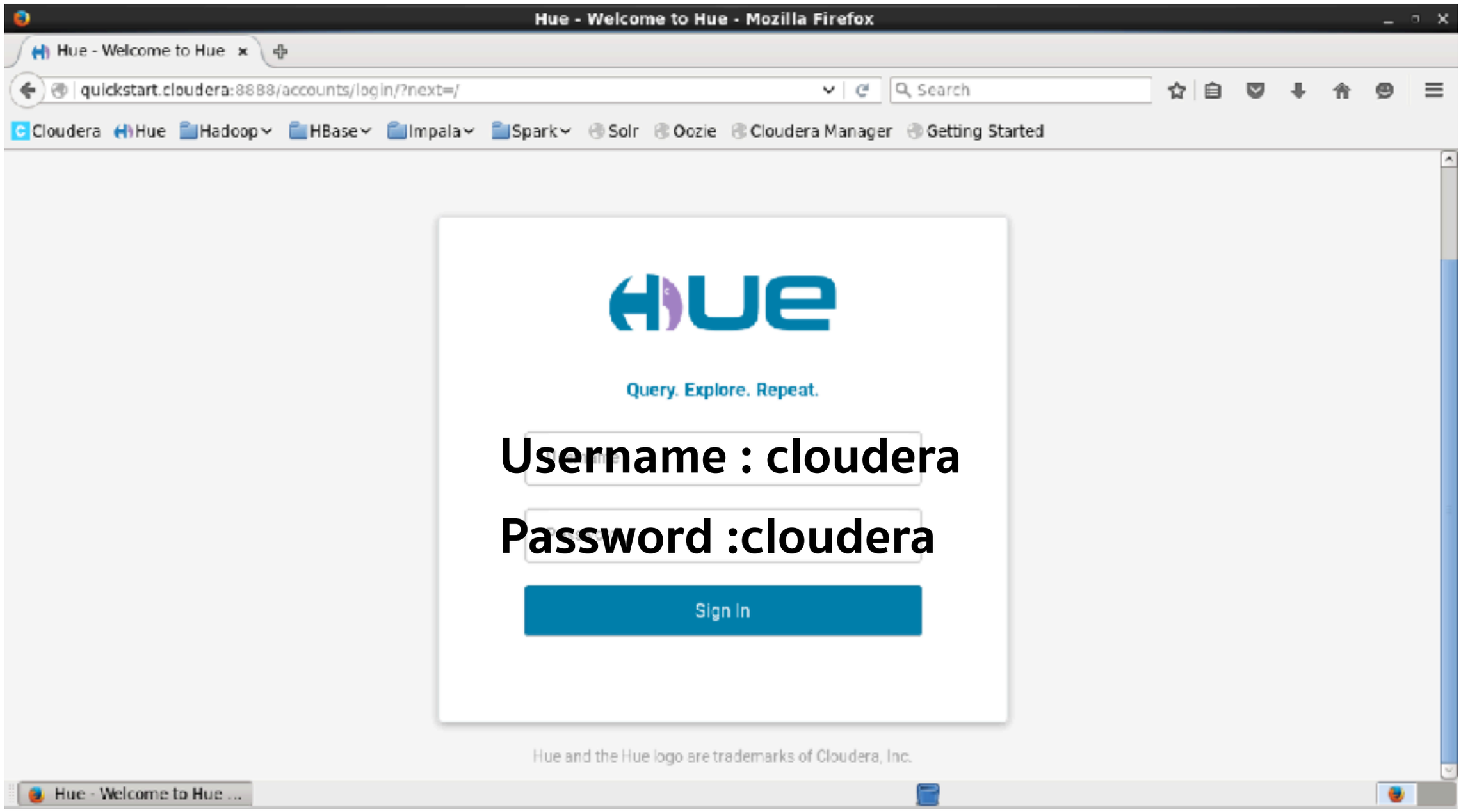
Available in

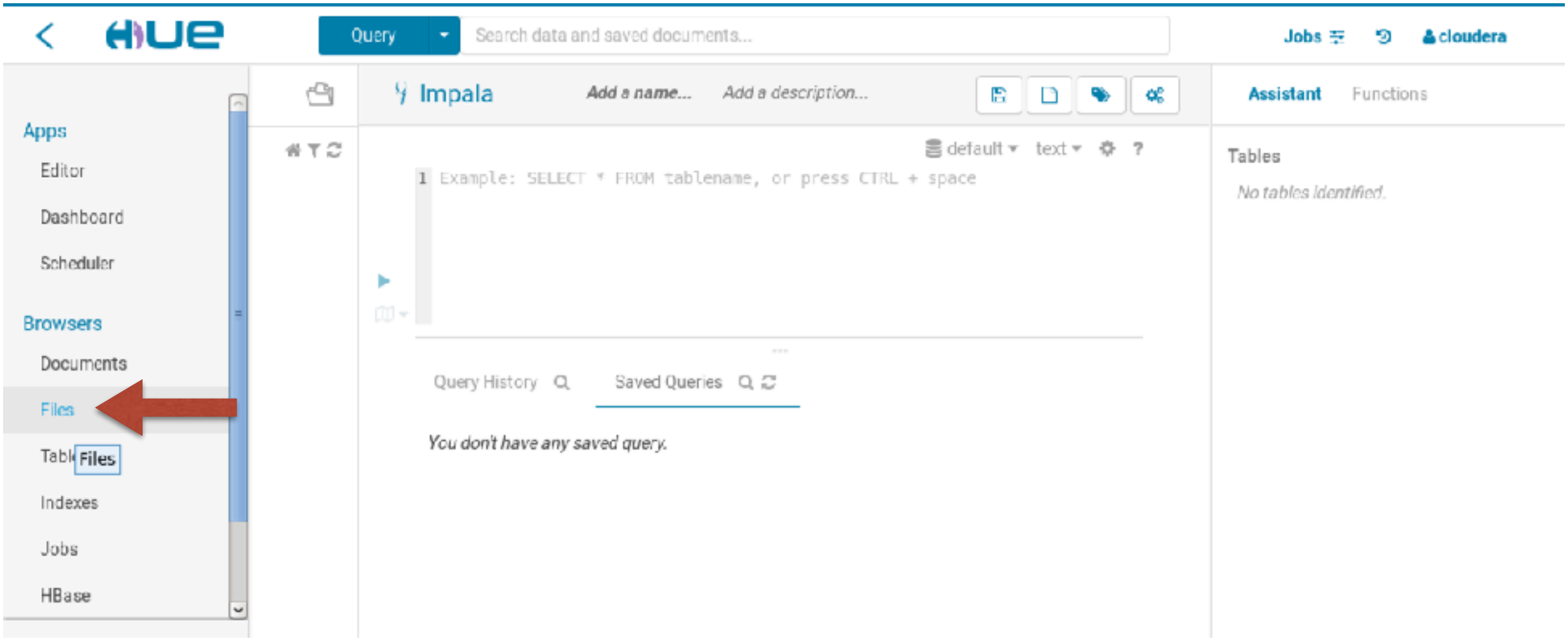
Thaveewat Khanan

Review file in Hadoop HDFS using File Browse

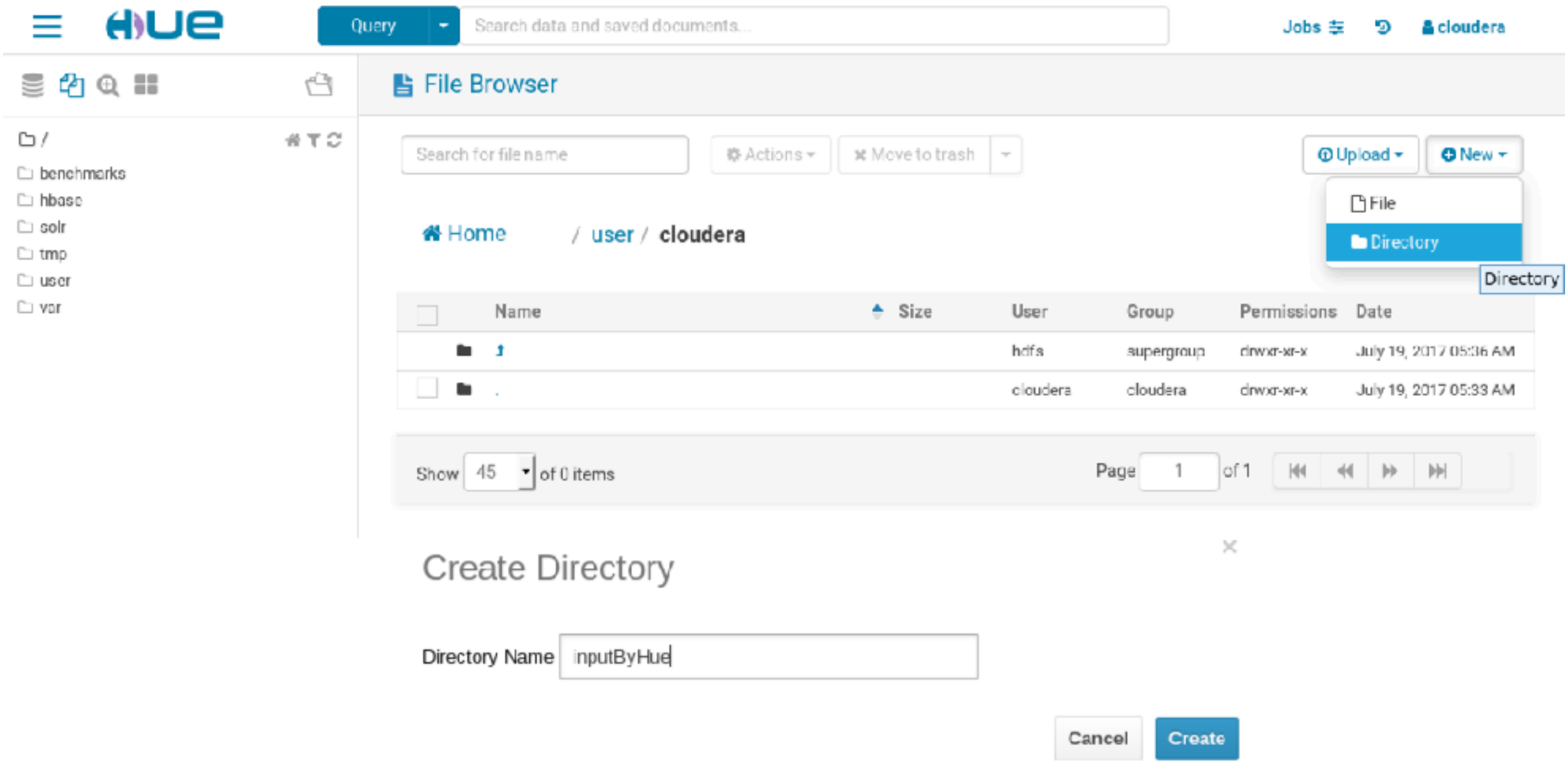
CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

Open Web Browser : http://quickstart.cloudera:8888



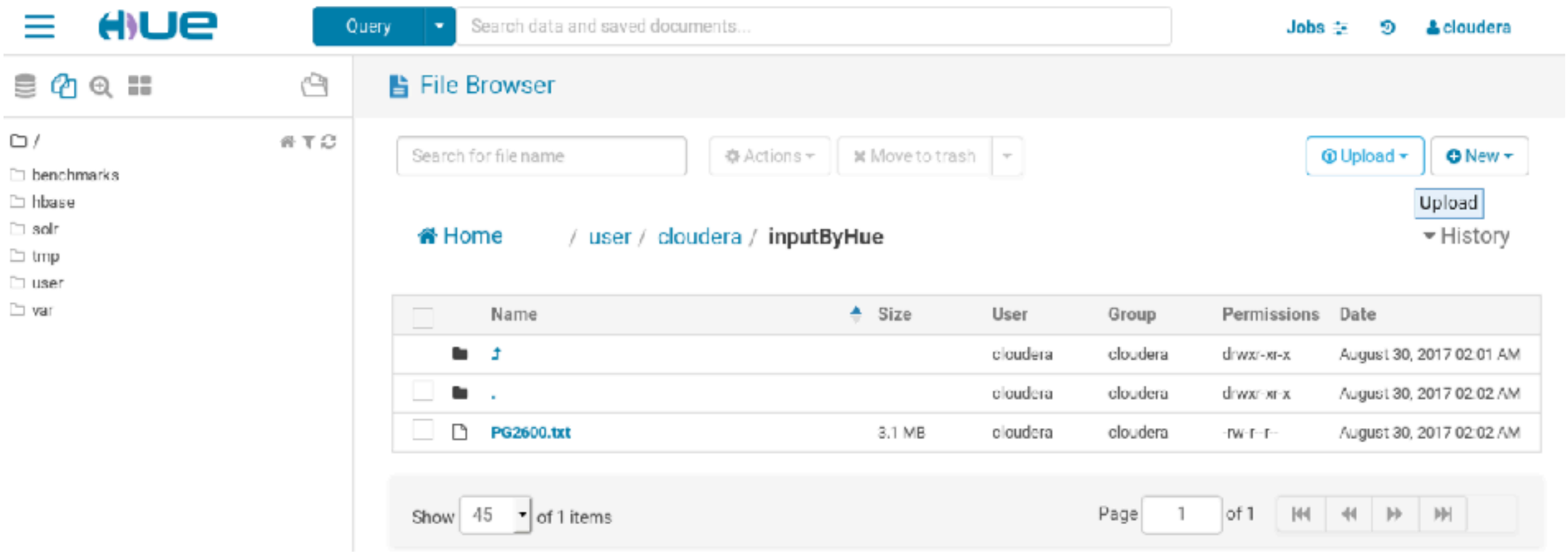

HUE
Query
Search data and saved documents...
Jobs
cloudera
Impala
Add a name...
Add a description...
Assistant
Functions
Apps
Editor
Dashboard
Scheduler
Browsers
Documents
Files
Tables
Indexes
Jobs
HBase
default
text
1 Example: SELECT * FROM tablename, or press CTRL + space
Tables
No tables identified.
Query History
Saved Queries
You don't have any saved query.

Create a new directory name as: inputByHue , output



Upload a local file to HDFS

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE


HUE
Query
Search data and saved documents...
Jobs
cloudera
File Browser
/
benchmarks
hbase
solr
tmp
user
var
Search for file name
Actions
Move to trash
Upload
New
Upload
History
Home / user / cloudera / inputByHue
Name
Size
User
Group
Permissions
Date
cloudera cloudera drwxr-xr-x August 30, 2017 02:01 AM
. cloudera cloudera drwxr-xr-x August 30, 2017 02:02 AM
PG2600.txt 3.1 MB cloudera cloudera -rw-r--r-- August 30, 2017 02:02 AM
Show 45 of 1 items
Page 1 of 1

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

Big Data Processing

MapReduce

Before MapReduce...

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

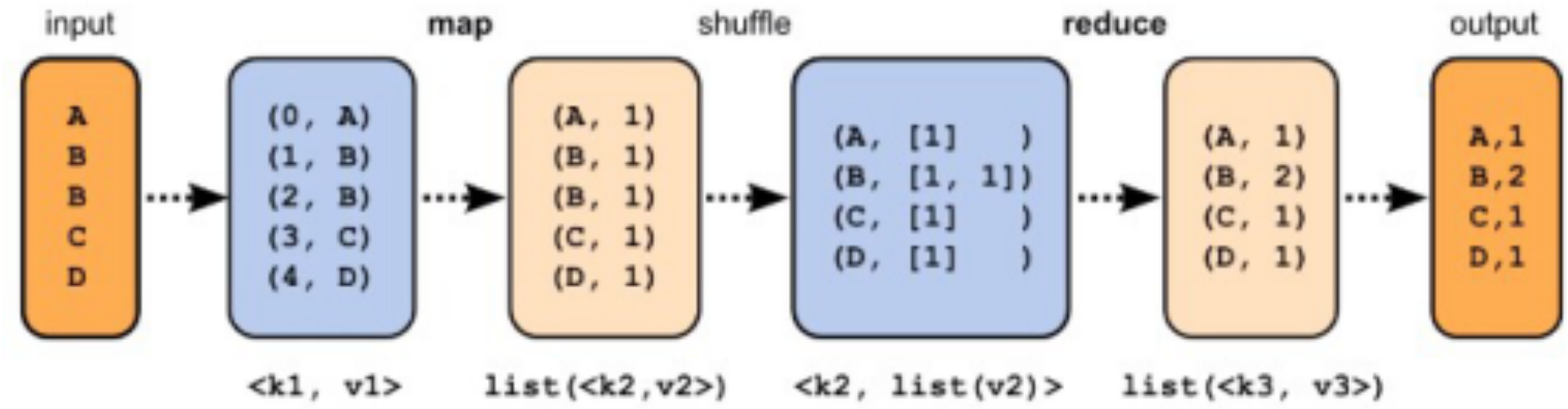
Large scale data processing was difficult!

- 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!

How Map and Reduce Work Together

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



Example MapReduce: WordCount

CHULALONGKORN BUSINESS SCHOOL

FLAGSHIP FOR LIFE

$cd /guest1

$wget https://github.com/bobbylovemovie/trainbigdata/raw/master/HDFS/wordcount.jar

$hadoop jar wordcount.jar org.myorg.WordCount /user/cloudera/input/* /user/cloudera/output/wordcount

```
[cloudera@quickstart guest1]$ hadoop jar wordcount.jar org.myorg.WordCount /user/cloudera/input/* /user/cloudera/output/wordcount
16/10/12 23:46:17 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
16/10/12 23:46:17 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
16/10/12 23:46:18 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your applica
tion with ToolRunner to remedy this.
16/10/12 23:46:18 INFO mapred.FileInputFormat: Total input paths to process : 1
16/10/12 23:46:18 WARN hdfs.DFSClient: Caught exception
java.lang.InterruptedException
        at java.lang.Object.wait(Native Method)
        at java.lang.Thread.join(Thread.java:1281)
        at java.lang.Thread.join(Thread.java:1355)
        at org.apache.hadoop.hdfs.DFSOutputStream$DataStreamer.closeResponder(DFSOutputStream.java:862)
        at org.apache.hadoop.hdfs.DFSOutputStream$DataStreamer.endBlock(DFSOutputStream.java:600)
        at org.apache.hadoop.hdfs.DFSOutputStream$DataStreamer.run(DFSOutputStream.java:789)
16/10/12 23:46:18 WARN hdfs.DFSClient: Caught exception
java.lang.InterruptedException
        at java.lang.Object.wait(Native Method)
        at java.lang.Thread.join(Thread.java:1281)
        at java.lang.Thread.join(Thread.java:1355)
        at org.apache.hadoop.hdfs.DFSOutputStream$DataStreamer.closeResponder(DFSOutputStream.java:862)
        at org.apache.hadoop.hdfs.DFSOutputStream$DataStreamer.endBlock(DFSOutputStream.java:600)
        at org.apache.hadoop.hdfs.DFSOutputStream$DataStreamer.run(DFSOutputStream.java:789)
16/10/12 23:46:18 INFO mapreduce.JobSubmitter: number of splits:2
16/10/12 23:46:18 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1476334425555_0001
16/10/12 23:46:19 INFO impl.YarnClientImpl: Submitted application application_1476334425555_0001
16/10/12 23:46:19 INFO mapreduce.Job: The url to track the job: http://quickstart.cloudera:8088/proxy/application_1476334425555_0001/
16/10/12 23:46:19 INFO mapreduce.Job: Running job: job_1476334425555_0001
16/10/12 23:46:32 INFO mapreduce.Job: Job job_1476334425555_0001 running in uber mode : false
16/10/12 23:46:32 INFO mapreduce.Job:  map 0% reduce 0%
16/10/12 23:46:54 INFO mapreduce.Job:  map 100% reduce 0%
16/10/12 23:47:06 INFO mapreduce.Job:  map 100% reduce 100%
16/10/12 23:47:07 INFO mapreduce.Job: Job job_1476334425555_0001 completed successfully
16/10/12 23:47:07 INFO mapreduce.Job: Counters: 49
        File System Counters
```

Reviewing MapReduce Job in Hue

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

Job Browser | Jobs | Workflows | Schedules | Bundles | SLAs

user:cloudera | Succeeded | Running | Failed in the last 7 days | Kill

Id	Name	User	Type	Status	Progress	Group	Started	Duration
application_1504064742599_0001	wordcount	cloudera	MAPREDUCE	SUCCEEDED	100	root.cloudera	August 30, 2017 2:05 AM	1m, 0s

Job Browser | Jobs | Workflows | Schedules | Bundles | SLAs

job_1504064742599_0001

ID
job_1504064742...

TYPE
MAPREDUCE

STATUS
SUCCEEDED

USER
cloudera

PROGRESS
100%

MAP
100% 2 / 2

REDUCE
100% 1 / 1

Logs | Tasks | Metadata | Counters | Kill

Filter by name | Succeeded | Running | Failed | Map | Reduce

Type	Id	Elapsed Time	Progress	State	Start Time	Successf
MAP	task_1504064742599_0001_m_000000	28252	1	SUCCEEDED	1504083966966	attempt_1
MAP	task_1504064742599_0001_m_000001	27784	1	SUCCEEDED	1504083967448	attempt_1
REDUCE	task_1504064742599_0001_r_000000	11368	1	SUCCEEDED	1504083997798	attempt_1

Reviewing MapReduce Output Result

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

File Browser

Search for file name | Actions | Move to trash | Upload | New

Home / user / cloudera / **output** | History

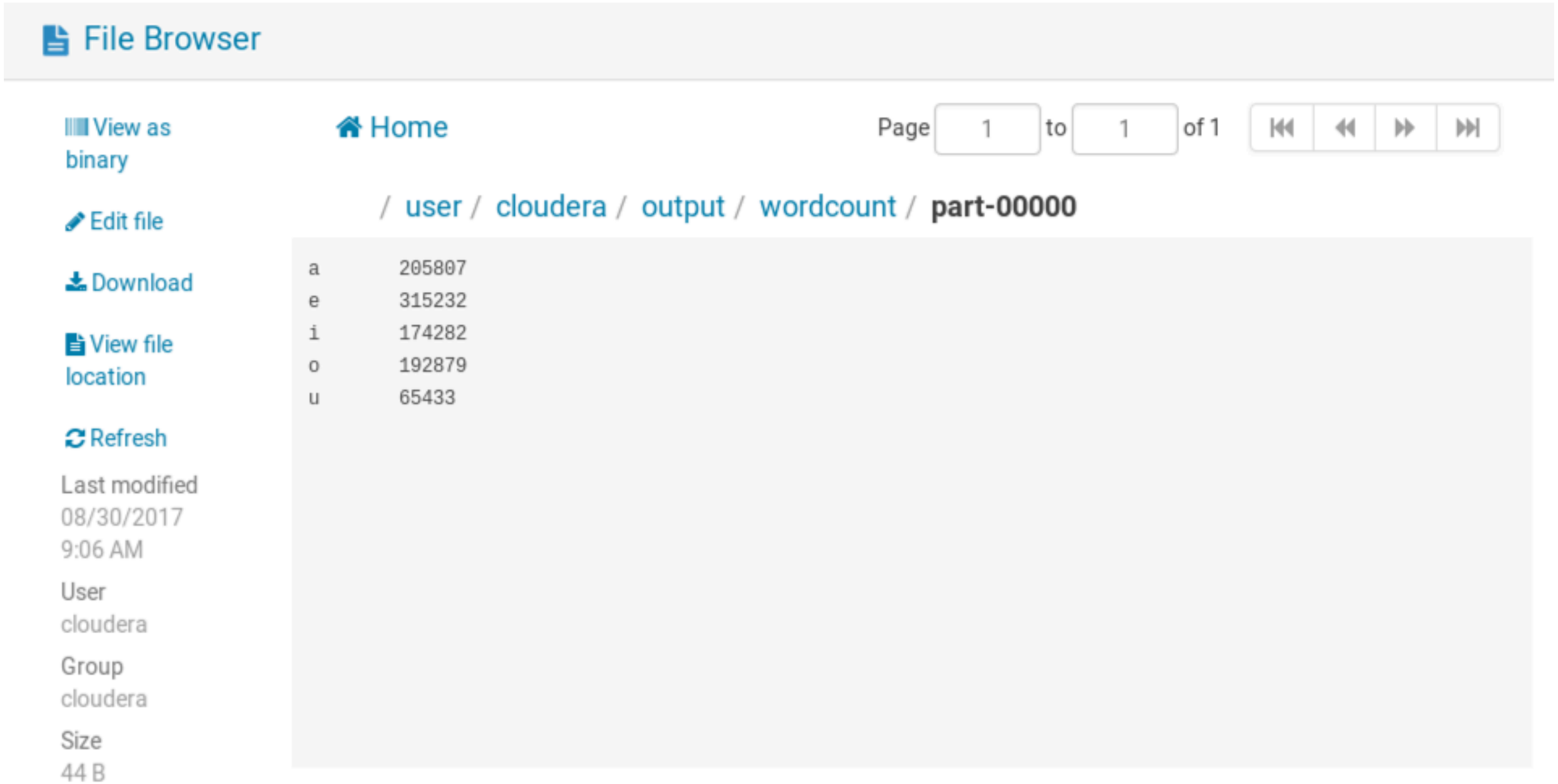
	Name	Size	User	Group	Permissions	Date
	..		cloudera	cloudera	drwxr-xr-x	August 30, 2017 02:06 AM
	.		cloudera	cloudera	drwxr-xr-x	August 30, 2017 02:06 AM
	wordcount		cloudera	cloudera	drwxr-xr-x	August 30, 2017 02:06 AM

Show 45 of 1 items

Page 1 of 1

Reviewing MapReduce Output Result

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE



Apache HBase

Understanding HBase

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

Hive.apache.org

When to use HBase ?

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

- 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

Hive.apache.org

Which one to use ?

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

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 Components

Region

Row of table are stores

Region Server

Hosts the tables

Master

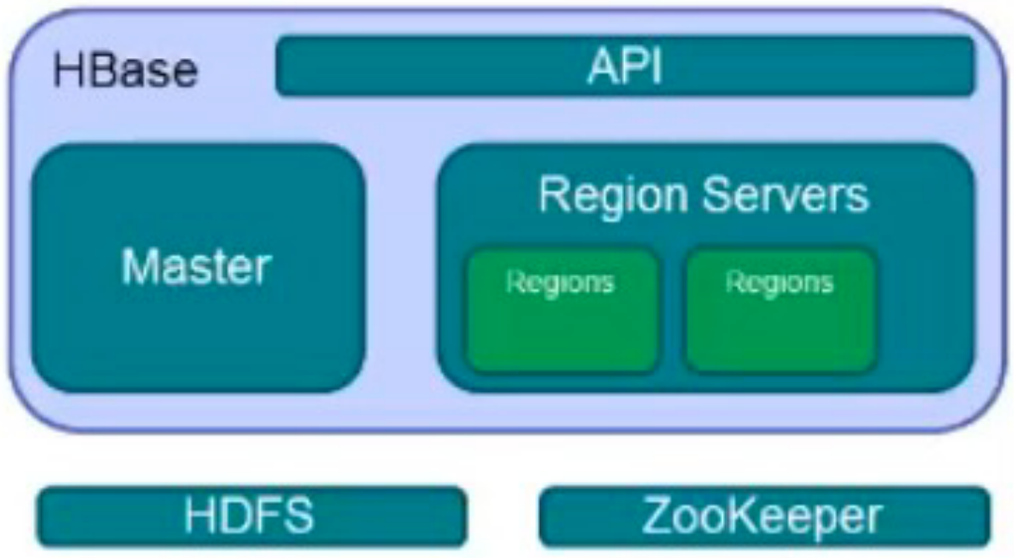
Coordinating the Region Servers

ZooKeeper

HDFS

API

The Java Client API



Hive.apache.org

HBase Shell Commands

List all tables in hbase.

```
hbase> list
```

Create table; pass table name

```
hbase> create 'ns1:t1', {NAME => 'f1', VERSIONS => 5}
hbase> create 't1', {NAME => 'f1'}, {NAME => 'f2'}, {NAME => 'f3'}
```

Put a cell ‘value’ at specified table/row/column

```
hbase> put ‘t1’, ‘r1’, ‘c1’, ‘value’, ts1
```

Get row or cell contents

```
hbase> get ‘t1’, ‘r1’
```

https://learnhbase.wordpress.com/2013/03/02/hbase-shell-commands/

Running HBase

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

$ hbase shell

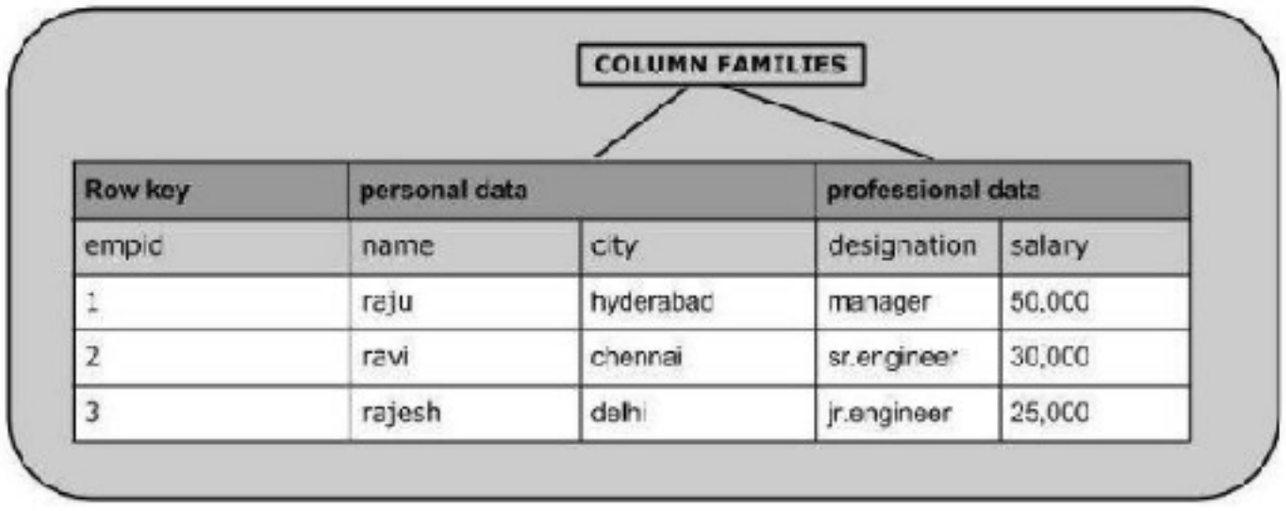
hbase(main):001:0> create 'employee', 'personal data', 'professional data'

hbase(main):002:0> list

```
[cloudera@quickstart guest1]$ hbase shell
2016-10-13 00:39:51,901 INFO  [main] Configuration.deprecation: hadoop.native.lib is deprecated. Instead, use io.native.lib.available
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 1.2.0-cdh5.8.0, rUnknown, Thu Jun 16 12:46:57 PDT 2016

hbase(main):001:0> create 'employee', 'personal data','professional data'
0 row(s) in 1.6860 seconds

=> Hbase::Table - employee
hbase(main):002:0> list
TABLE
employee
1 row(s) in 0.0180 seconds

=> ["employee"]
hbase(main):003:0>
```

Row key	personal data	professional data

Create Data



hbase(main):004:0> put 'employee','1','personal data:name','raju'

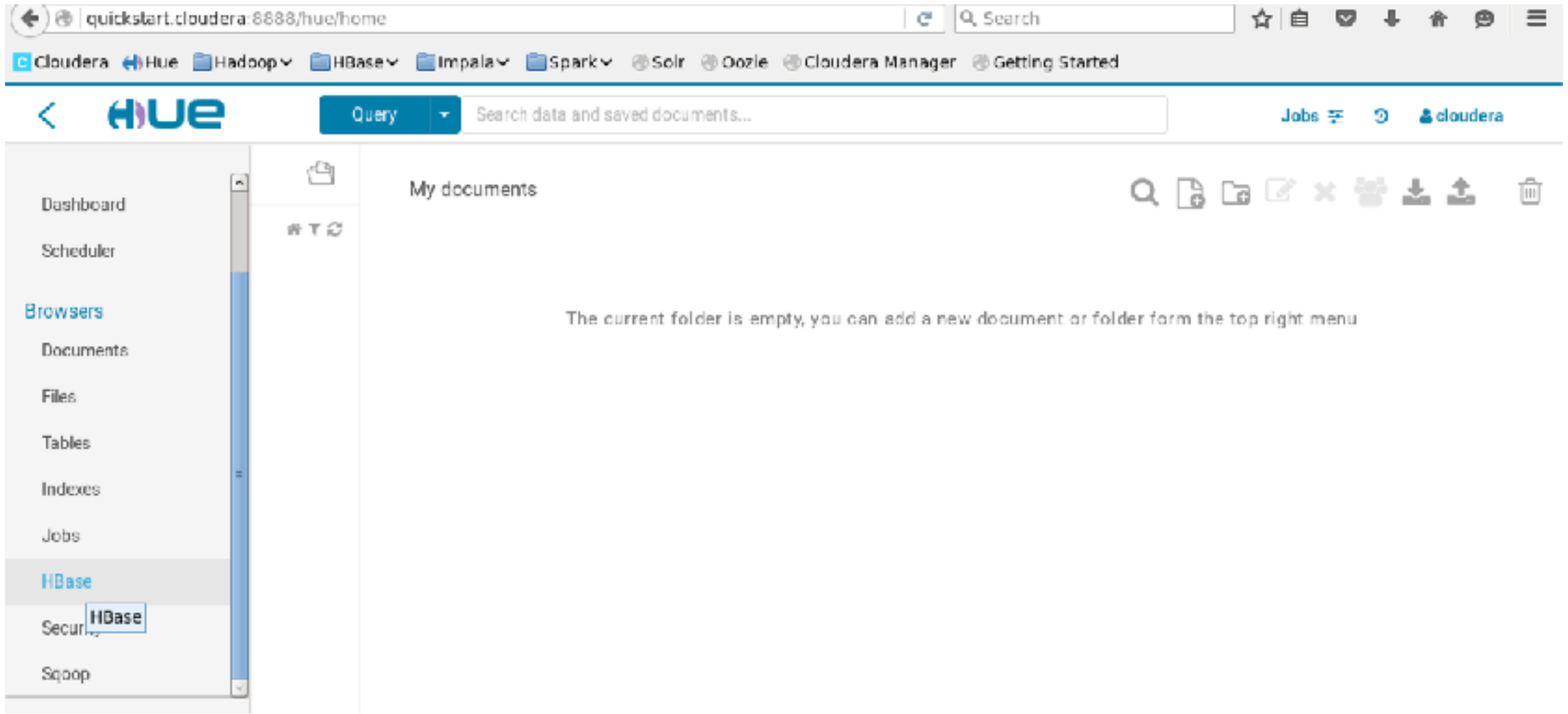
hbase(main):005:0> put 'employee','1','personal data:city','hyderabad'

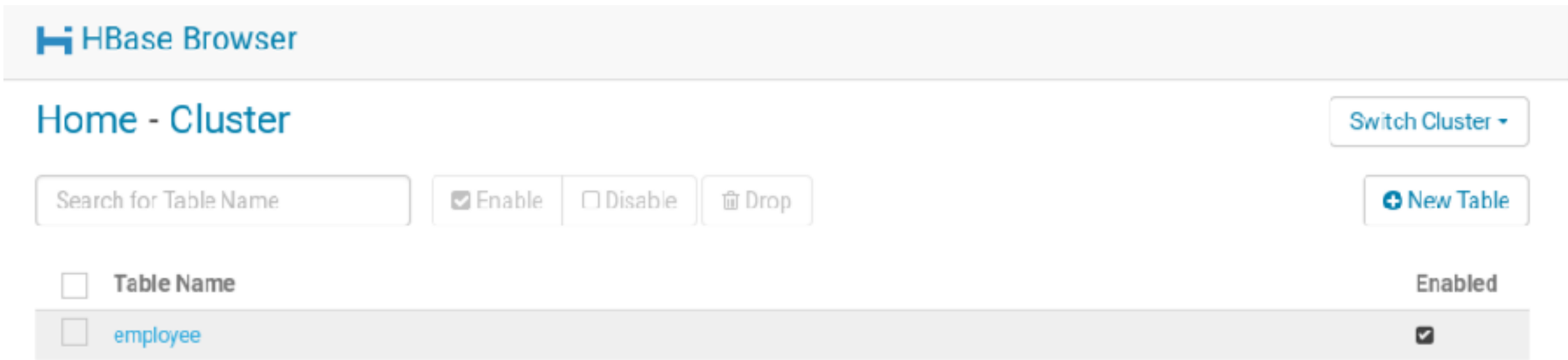
hbase(main):006:0> put 'employee','1','professional data:designation','manager'

hbase(main):007:0> put 'employee','1','professional data:salary','5000'

```
hbase(main):004:0> put 'employee','1','personal data:name','raju'
0 row(s) in 0.0190 seconds

hbase(main):005:0> put 'employee','1','personal data:city','hyderabad'
0 row(s) in 0.0050 seconds

hbase(main):006:0> put 'employee','1','professional data:designation','manager'
0 row(s) in 0.0110 seconds

hbase(main):007:0> put 'employee','1','professional data:salary','5000'
0 row(s) in 0.0050 seconds
```

Running HBase Browser





Viewing Employee Table

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

HBase Browser

Home - Cluster / employee

Switch Cluster

row_key, row_prefix* +scan_len [col1, family:col2, fam3:, col_prefix

personal data: professional data:

Filter Columns/Families

All

Sort By **ASC**

1

personal data: city	personal data: name	professional data: designation	professional data: salary
hyderabad	raju	manager	5000

hbase(main):025:0> put 'employee','2','personal data:name','bobby'

1

Filter Column Names/Family

Sort By **ASC**

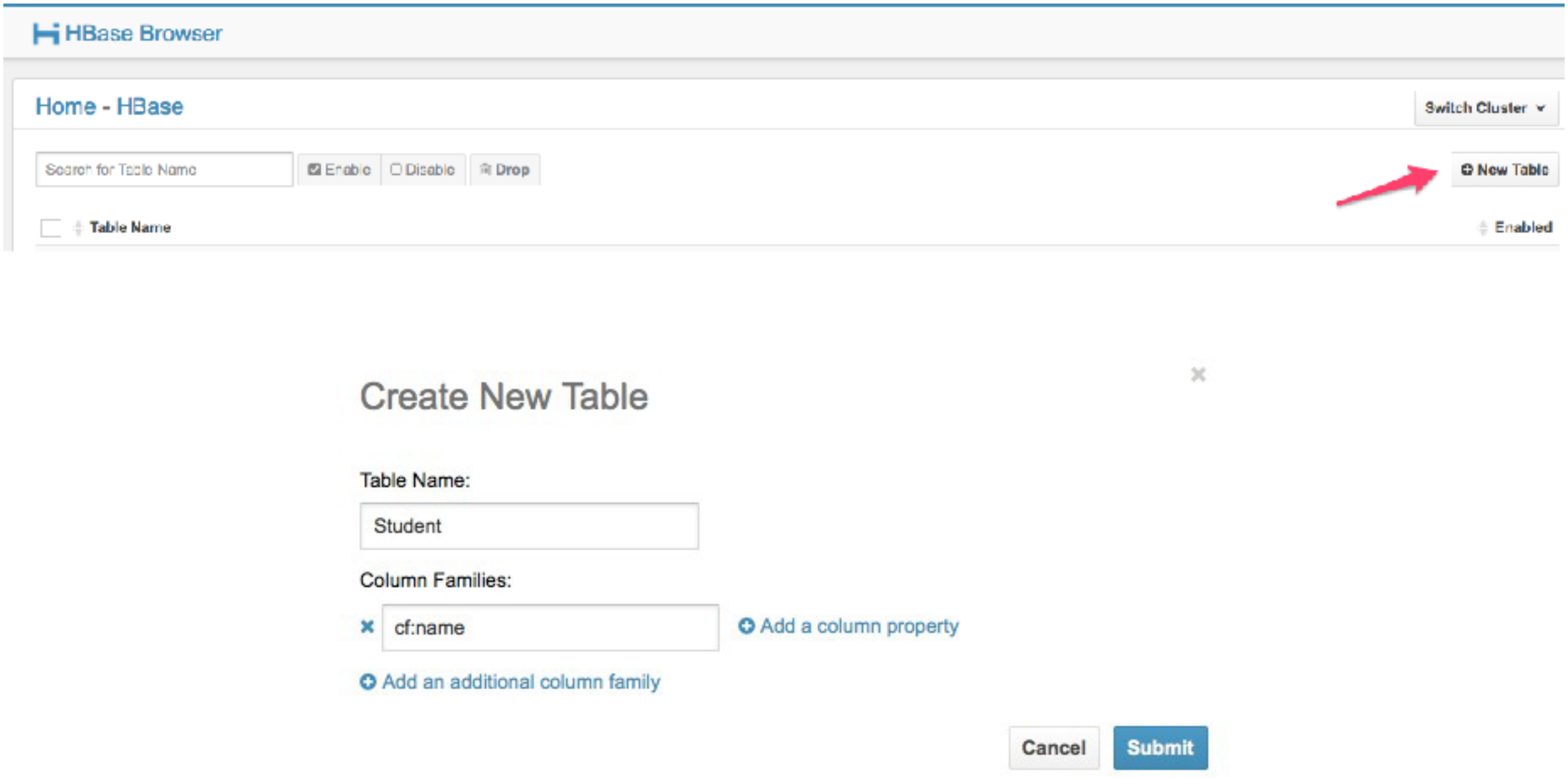
Drop Columns

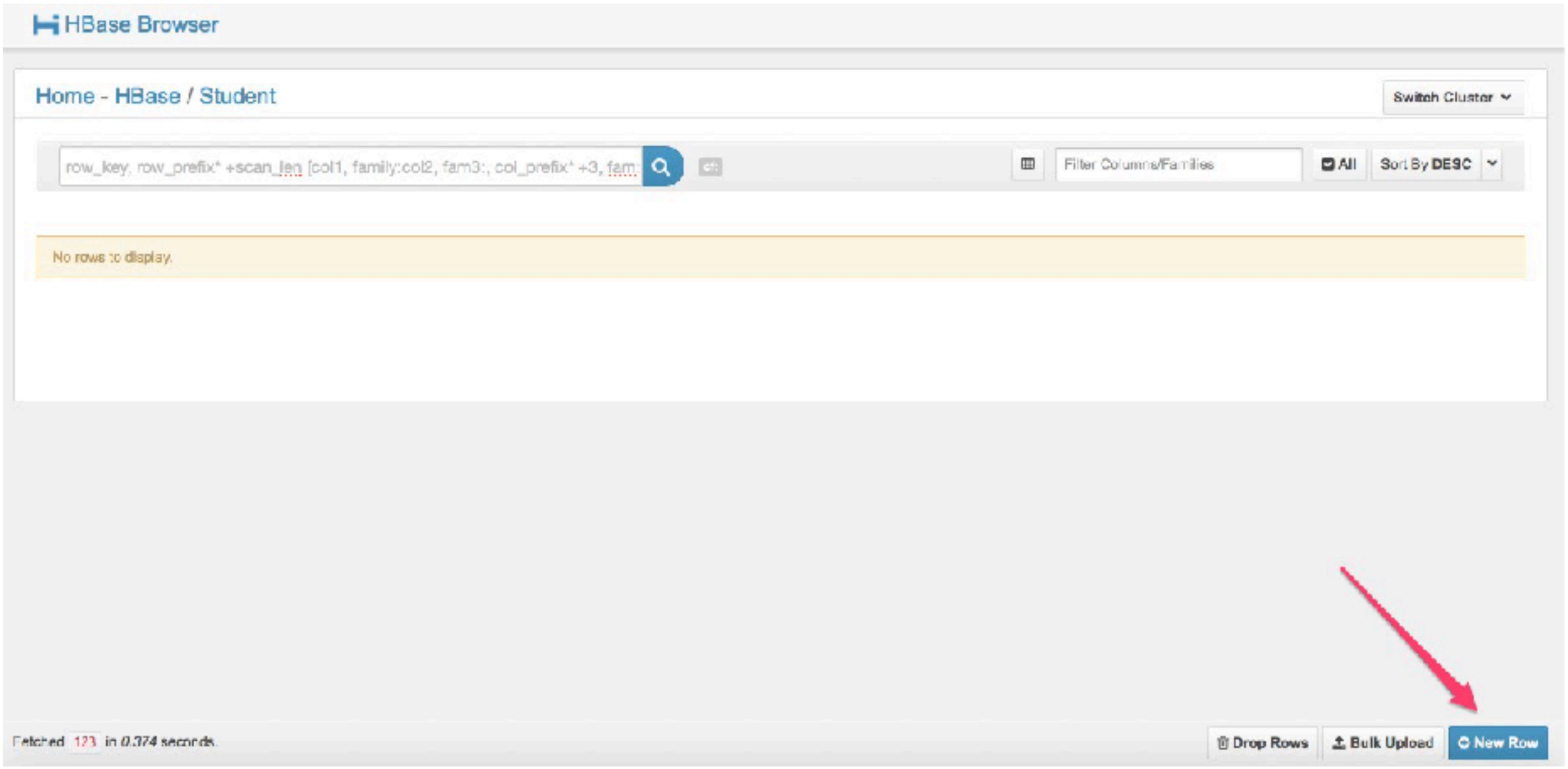
personal data: city	personal data: name	professional data: designation	professional data: salary
hyderabad	raju	manager	5000

2

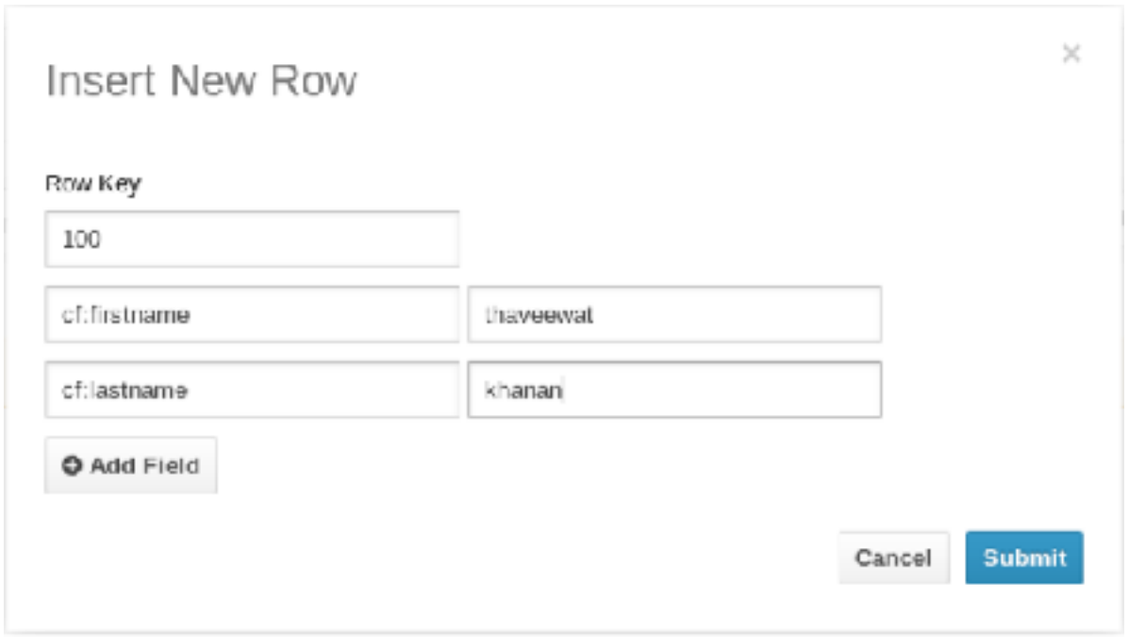
personal data: name
bobby

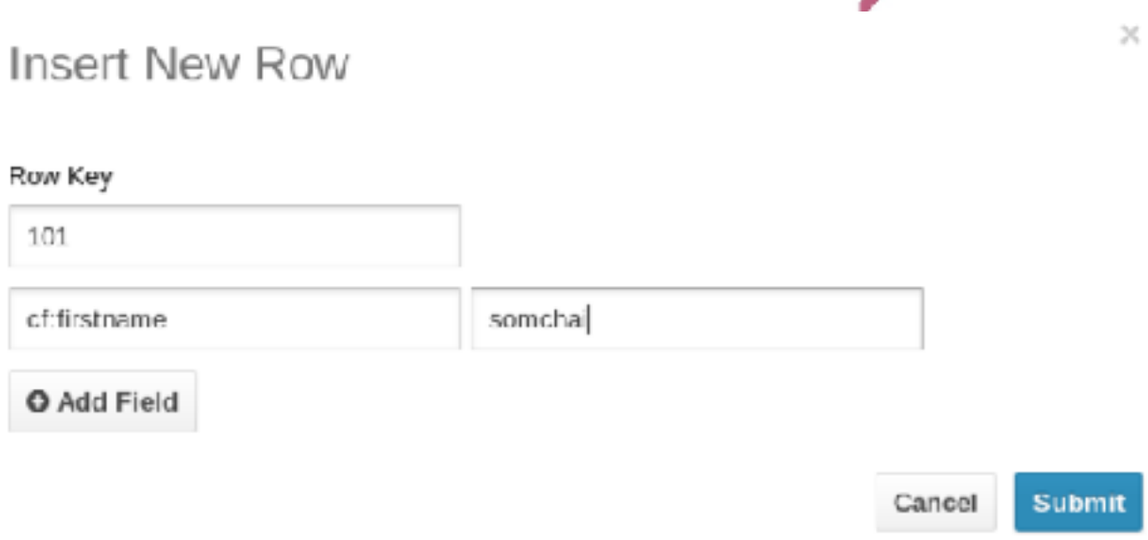
Create a table in HBase

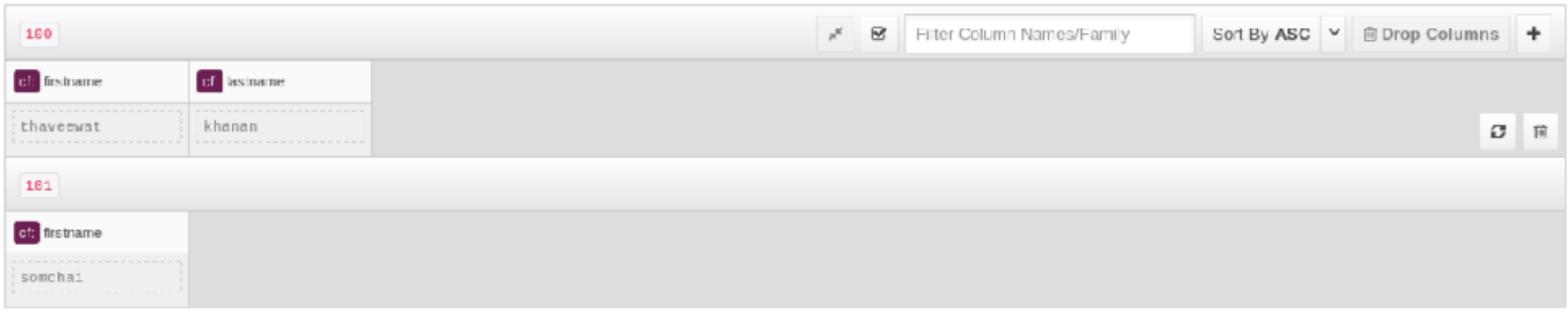



HBase Browser
Home - HBase / Student
Switch Cluster
row_key, row_prefix* +scan_len [col1, family:col2, fam3:, col_prefix* +3, fam:
Filter Columns/Families
All
Sort By DESC
No rows to display.
Fetched 123 in 0.374 seconds.
Drop Rows
Bulk Upload
New Row

Add field into a new row







APACHE HIVE

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

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

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.).

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
```

Hive Built in Functions

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

Return Type	Function Name (Signature)	Description
BIGINT	round(double a)	returns the rounded BIGINT value of the double
BIGINT	floor(double a)	returns the maximum BIGINT value that is equal or less than the double
BIGINT	ceil(double a)	returns the minimum BIGINT value that is equal or greater than the double
double	rand(), rand(int seed)	returns a random number (that changes from row to row). Specifiying the seed will make sure the generated random number sequence is deterministic.
string	concat(string A, string B,...)	returns the string resulting from concatenating B after A. For example, concat('foo', 'bar') results in 'foobar'. This function accepts arbitrary number of arguments and return the concatenation of all of them.
string	substr(string A, int start)	returns the substring of A starting from start position till the end of string A. For example, substr('foobar', 4) results in 'bar'
string	substr(string A, int start, int length)	returns the substring of A starting from start position with the given length e.g. substr('foobar', 4, 2) results in 'ba'
string	upper(string A)	returns the string resulting from converting all characters of A to upper case e.g. upper('fOoBaR') results in 'FOOBAR'
string	ucase(string A)	Same as upper
string	lower(string A)	returns the string resulting from converting all characters of B to lower case e.g. lower('fOoBaR') results in 'foobar'
string	lcase(string A)	Same as lower
string	trim(string A)	returns the string resulting from trimming spaces from both ends of A e.g. trim(' foobar ') results in 'foobar'
string	ltrim(string A)	returns the string resulting from trimming spaces from the beginning(left hand side) of A. For example, ltrim(' foobar ') results in 'foobar '
string	rtrim(string A)	returns the string resulting from trimming spaces from the end(right hand side) of A. For example, rtrim(' foobar ') results in ' foobar'
string	regexp_replace(string A, string B, string C)	returns the string resulting from replacing all substrings in B that match the Java regular expression syntax(See Java regular expressions syntax) with C. For example, regexp_replace('foobar', 'oo|ar',) returns 'fb'
string	from_unixtime(int unixtime)	convert the number of seconds from unix epoch (1970-01-01 00:00:00 UTC) to a string representing the timestamp of that moment in the current system time zone in the format of "1970-01-01 00:00:00"
string	to_date(string timestamp)	Return the date part of a timestamp string: to_date("1970-01-01 00:00:00") = "1970-01-01"
int	year(string date)	Return the year part of a date or a timestamp string: year("1970-01-01 00:00:00") = 1970, year("1970-01-01") = 1970
int	month(string date)	Return the month part of a date or a timestamp string: month("1970-11-01 00:00:00") = 11, month("1970-11-01") = 11
int	day(string date)	Return the day part of a date or a timestamp string: day("1970-11-01 00:00:00") = 1, day("1970-11-01") = 1
string	get_json_object(string json_string, string path)	Extract json object from a json string based on json path specified, and return json string of the extracted json object. It will return null if the input json string is invalid

Hive Commands

Command Line

Function	Hive
Run query	`hive -e 'select a.col from tab1 a'`
Run query silent mode	`hive -S -e 'select a.col from tab1 a'`
Set hive config variables	`hive -e 'select a.col from tab1 a' -hiveconf hive.root.logger=DEBUG,console`
Use initialization script	`hive -i initialize.sql`
Run non-interactive script	`hive -f script.sql`

Hive Shell

Function	Hive
Run script inside shell	`source file_name`
Run ls (dfs) commands	`dfs -ls /user`
Run ls (bash command) from shell	`!ls`
Set configuration variables	`set mapred.reduce.tasks=32`
TAB auto completion	`set hive.<TAB>`
Show all variables starting with hive	`set`
Revert all variables	`reset`
Add jar to distributed cache	`add jar jar_path`
Show all jars in distributed cache	`list jars`
Delete jar from distributed cache	`delete jar jar_name`

Hive Tables

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

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

Hive External Table


```
– CREATE EXTERNAL TABLE external_Table (dummy STRING)
– LOCATION '/user/notroot/external_table';
```

Dropping External Table using Hive
Hive will delete metadata from metastore
Hive will NOT delete the HDFS file
You need to manually delete the HDFS file

HiveQL and MySQL Comparison

CHULALONGKORN BUSINESS SCHOOL

FLAGSHIP FOR LIFE

Metadata

Function	MySQL	HiveQL
Selecting a database	`USE database;`	`USE database;`
Listing databases	`SHOW DATABASES;`	`SHOW DATABASES;`
Listing tables in a database	`SHOW TABLES;`	`SHOW TABLES;`
Describing the format of a table	`DESCRIBE table;`	`DESCRIBE (FORMATTED|EXTENDED) table;`
Creating a database	`CREATE DATABASE db_name;`	`CREATE DATABASE db_name;`
Dropping a database	`DROP DATABASE db_name;`	`DROP DATABASE db_name (CASCADE);`

HiveQL and MySQL Query Comparison

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

Query

Function	MySQL	HiveQL
Retrieving information	`SELECT from_columns FROM table WHERE conditions;`	`SELECT from_columns FROM table WHERE conditions;`
All values	`SELECT * FROM table;`	`SELECT * FROM table;`
Some values	`SELECT * FROM table WHERE rec_name = "value";`	`SELECT * FROM table WHERE rec_name = "value";`
Multiple criteria	`SELECT * FROM table WHERE rec1="value1" AND rec2="value2";`	`SELECT * FROM TABLE WHERE rec1 = "value1" AND rec2 = "value2";`
Selecting specific columns	`SELECT column_name FROM table;`	`SELECT column_name FROM table;`
Retrieving unique output records	`SELECT DISTINCT column_name FROM table;`	`SELECT DISTINCT column_name FROM table;`
Sorting	`SELECT col1, col2 FROM table ORDER BY col2;`	`SELECT col1, col2 FROM table ORDER BY col2;`
Sorting backward	`SELECT col1, col2 FROM table ORDER BY col2 DESC;`	`SELECT col1, col2 FROM table ORDER BY col2 DESC;`
Counting rows	`SELECT COUNT(*) FROM table;`	`SELECT COUNT(*) FROM table;`
Grouping with counting	`SELECT owner, COUNT(*) FROM table GROUP BY owner;`	`SELECT owner, COUNT(*) FROM table GROUP BY owner;`
Maximum value	`SELECT MAX(col_name) AS label FROM table;`	`SELECT MAX(col_name) AS label FROM table;`
Selecting from multiple tables (Join same table using alias w/"AS")	`SELECT pet.name, comment FROM pet, event WHERE pet.name = event.name;`	`SELECT pet.name, comment FROM pet JOIN event ON (pet.name = event.name);`

Loading Data using Hive

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

Start Hive

$ hive

Quit from Hive

hive> quit;

```
[cloudera@quickstart guest1]$ hive
2016-10-13 02:08:13,500 WARN  [main] mapreduce.TableMapReduceUtil: The hbase-prefix-tree module jar containing PrefixTreeCodec is not present.  Continuing wi
thout it.

Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j.properties
WARNING: Hive CLI is deprecated and migration to Beeline is recommended.
hive>
```

Create Hive Table

hive> CREATE TABLE TEST_TBL(ID INT,COUNTRY STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE ;

hive> SHOW TABLES;

hive> describe test_tbl;

```
hive> CREATE TABLE TEST_TBL(ID INT,COUNTRY STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE ;
OK
Time taken: 1.2 seconds
hive> SHOW TABLES;
OK
test_tbl
Time taken: 0.234 seconds, Fetched: 1 row(s)
hive> describe test_tbl
    > ;
OK
id                      int
country                 string
Time taken: 0.209 seconds, Fetched: 2 row(s)
```

Reviewing Hive Table in HDFS

Alter and Drop Hive Table

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

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

Preparing Large Dataset

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

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

grouplens 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.

Help our research lab: Please take a short survey about the MovieLens datasets

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.

- README.txt

Datasets

MovieLens

HetRec 2011

WikiLens

Book-Crossing

Jester

EachMovie

MovieLens Dataset

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

1)Open Terminal

2)Type command > mkdir movielens_dataset

3)Type command >cd movielens_dataset

4)Type command > wget http://files.grouplens.org/datasets/movielens/ml-100k.zip

5)Type command > unzip ml-100k.zip

6)Type command > more ml-100k/u.user

```
[cloudera@quickstart ~]$ mkdir movielens_dataset
[cloudera@quickstart ~]$ cd movielens_dataset/
[cloudera@quickstart movielens_dataset]$ wget http://files.grouplens.org/dat
s/movielens/ml-100k.zip
--2016-10-13 03:07:20--  http://files.grouplens.org/datasets/movielens/ml-10
ip
Resolving files.grouplens.org... 128.101.34.146
Connecting to files.grouplens.org|128.101.34.146|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 4924029 (4.7M) [application/zip]
Saving to: "ml-100k.zip"

100%[======================================>] 4,924,029    226K/s   in 36s

2016-10-13 03:08:02 (135 KB/s) - "ml-100k.zip" saved [4924029/4924029]

[cloudera@quickstart movielens_dataset]$ unzip ml-100k.zip
Archive:  ml-100k.zip
   creating: ml-100k/
  inflating: ml-100k/allbut.pl
  inflating: ml-100k/mku.sh
  inflating: ml-100k/README
  inflating: ml-100k/u.data
```

```
[cloudera@quickstart movielens_dataset]$ more ml-100k/u.user
1|24|M|technician|85711
2|53|F|other|94043
3|23|M|writer|32067
4|24|M|technician|43537
5|33|F|other|15213
6|42|M|executive|98101
7|57|M|administrator|91344
8|36|M|administrator|05201
9|29|M|student|01002
10|53|M|lawyer|90703
11|39|F|other|30329
12|28|F|other|06405
13|47|M|educator|29206
14|45|M|scientist|55106
15|49|F|educator|97301
16|21|M|entertainment|10309
17|30|M|programmer|06355
```

Moving dataset to HDFS

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

1)Type command > cd ml-100k

2)Type command > hadoop fs -mkdir /user/cloudera/movielens

3)Type command > hadoop fs -put u.user /user/cloudera/movielens

4)Type command > hadoop fs -ls /user/cloudera/movielens

```
[cloudera@quickstart movielens_dataset]$ cd ml-100k
[cloudera@quickstart ml-100k]$ hadoop fs -mkdir /user/cloudera/movielens
[cloudera@quickstart ml-100k]$ hadoop fs -put u.user /user/cloudera/movielens
[cloudera@quickstart ml-100k]$ hadoop fs -ls /user/cloudera/movielens
Found 1 items
-rw-r--r--   1 cloudera cloudera      22628 2016-10-13 03:16 /user/cloudera/movi
elens/u.user
[cloudera@quickstart ml-100k]$
```

CREATE & SELECT TABLE

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

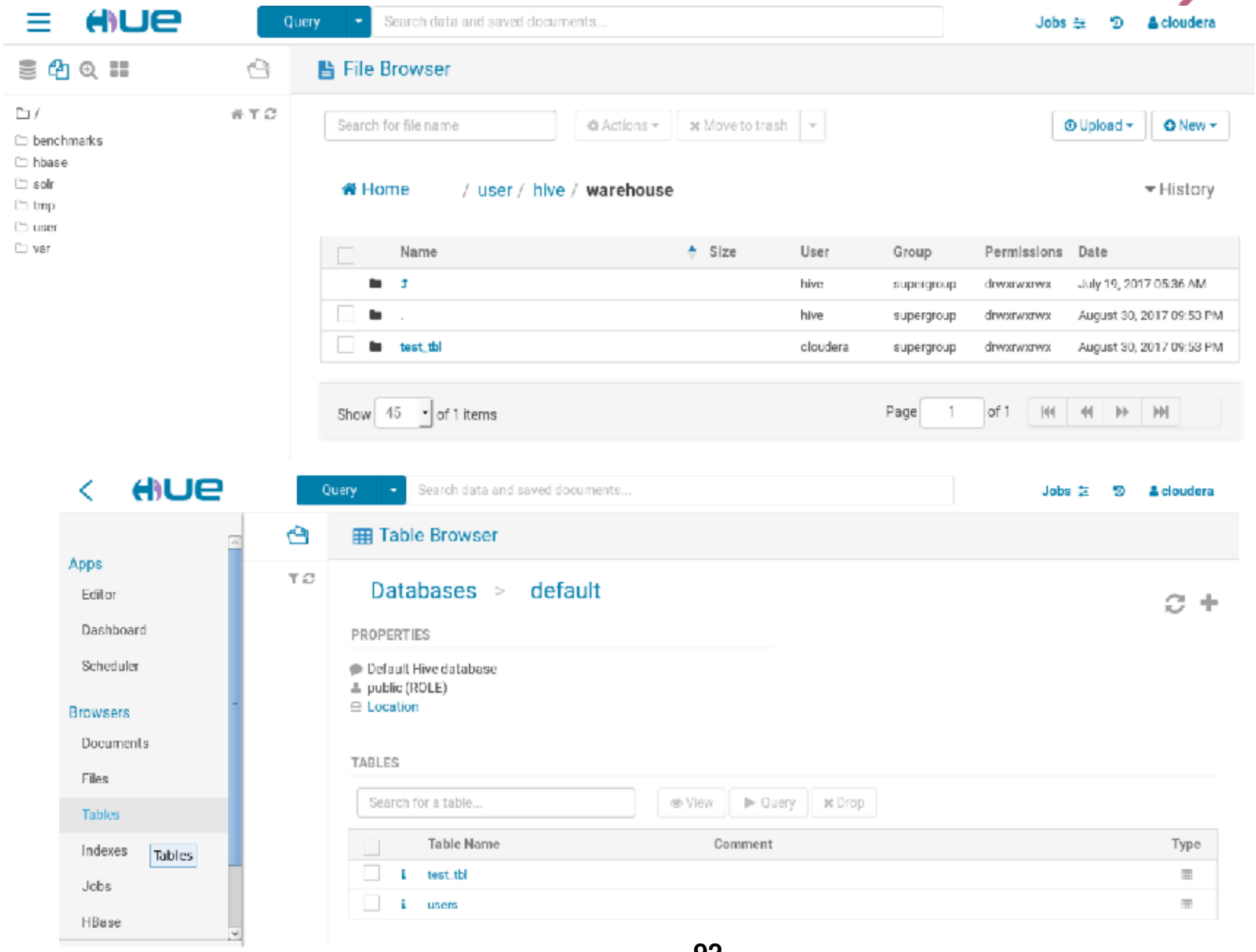
Type command > hive

```
hive> create external table users (userid int,age int,
gender string,occupation string ,zipcode string)
row format delimited fields terminated by '|'
stored as textfile location '/user/cloudera/movielens' ;
```

hive > select * from users;

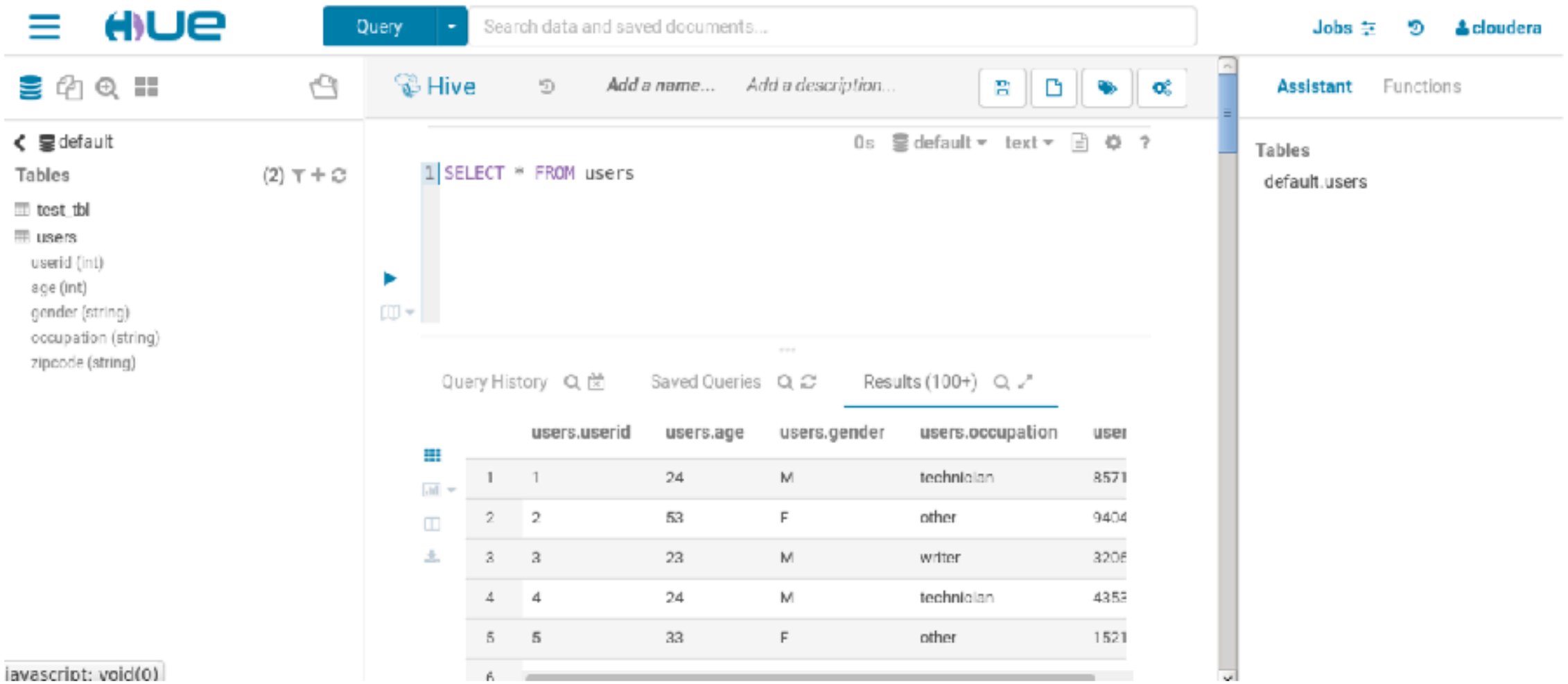
```
hive> CREATE EXTERNAL TABLE users (userid INT,  age INT,
    >   gender STRING, occupation STRING, zipcode STRING) ROW FORMAT
    > DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE
    > LOCATION '/user/cloudera/movielens';
OK
Time taken: 0.646 seconds
hive> SELECT * FROM users;
OK
1       24      M       technician      85711
2       53      F       other   94043
3       23      M       writer  32067
4       24      M       technician      43537
5       33      F       other   15213
6       42      M       executive       98101
7       57      M       administrator   91344
8       36      M       administrator   05201
```

CREATE EXTERNAL TABLE Not File Moved



Starting Hive Editor on HUE

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE



IMPALA

open source massively parallel processing (MPP) SQL query engine

Cloudera Impala is a query engine that runs on Apache Hadoop. Impala brings scalable parallel database technology to Hadoop, enabling users to issue low-latency SQL queries to data stored in HDFS and Apache HBase without requiring data movement or transformation.

What is Impala ?

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

General--- purpose SQL engine

Real--time queries in Apache Hadoop

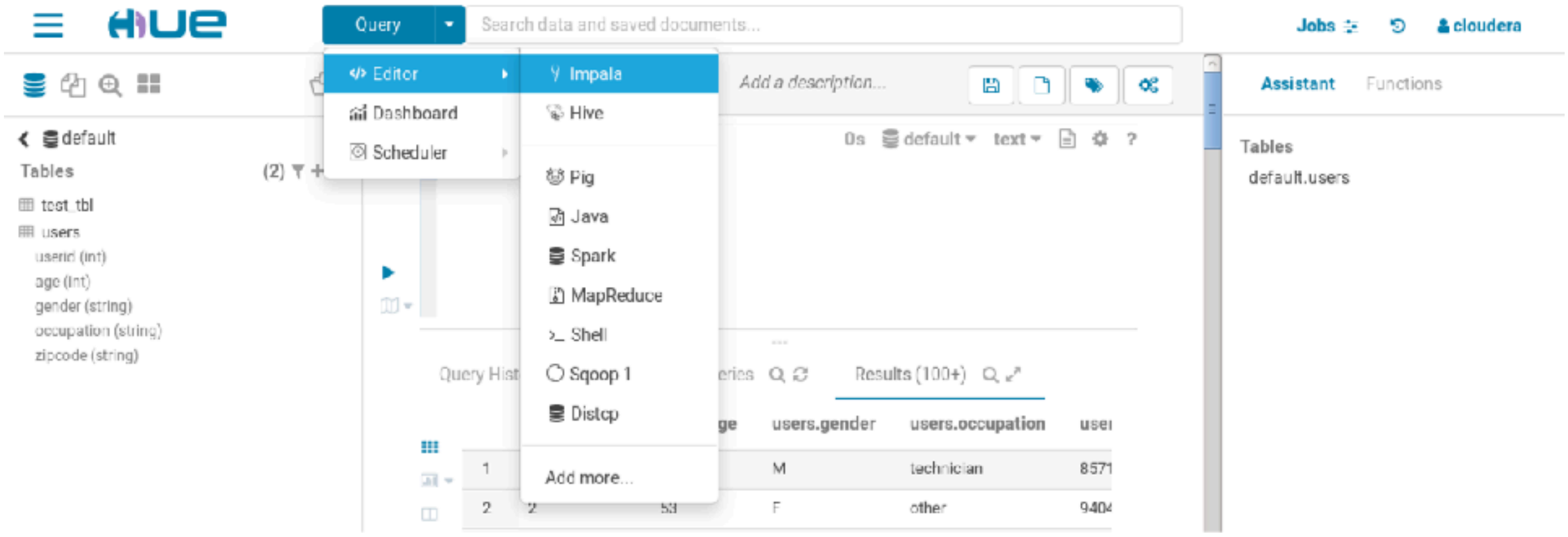
Opensource under Apache License

Runs directly within Hadoop

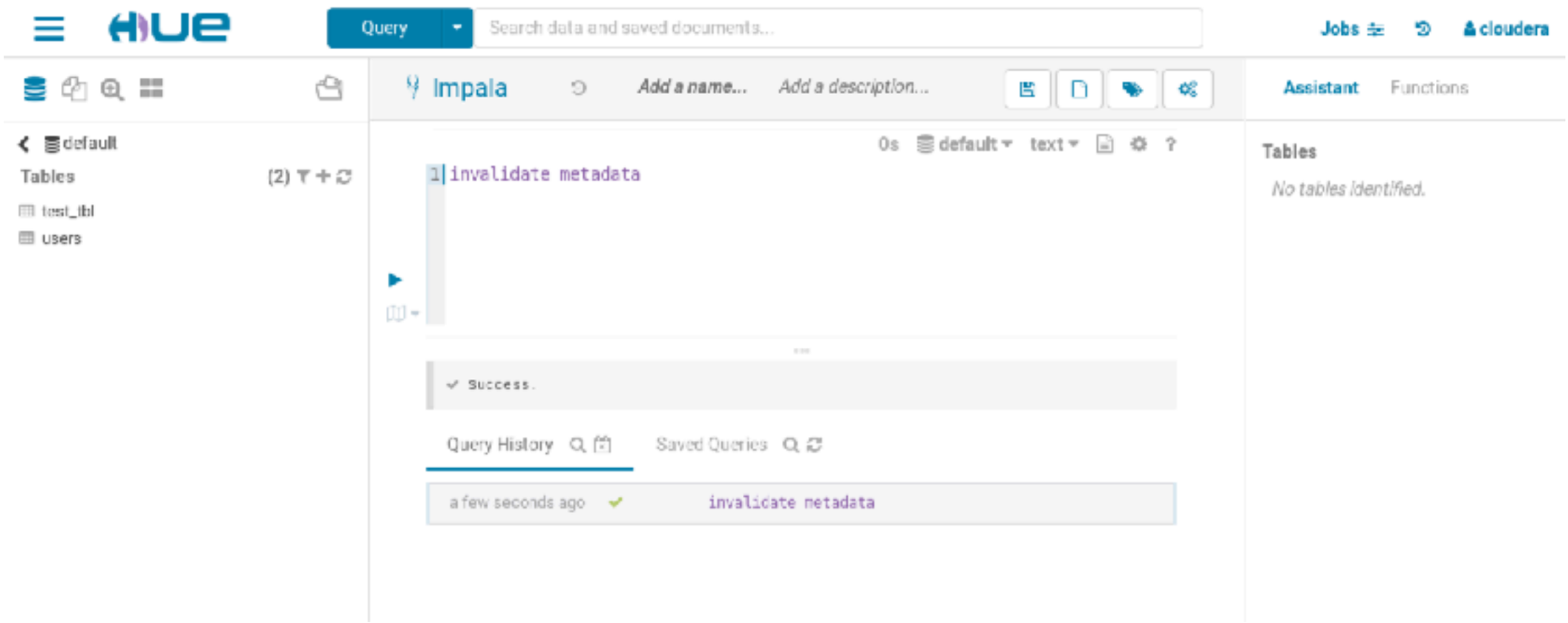
High performance

- C++ instead of Java
- Runtime code generator
- Roughly 4-100 x Hive

Start Impala Query Editor



Update the list of tables/metadata by execute the command invalidate metadata



Run Impala On Terminal

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

Open Terminal

Type command > impala-shell

[quickstart.cloudera:21000] > select * from users;

```
[[quickstart.cloudera:21000] > select * from users;
Query: select * from users
+--------+-----+--------+--------------+---------+
| userid | age | gender | occupation   | zipcode |
+--------+-----+--------+--------------+---------+
| 1      | 24  | M      | technician   | 85711   |
| 2      | 53  | F      | other        | 94043   |
| 3      | 23  | M      | writer       | 32067   |
| 4      | 24  | M      | technician   | 43537   |
| 5      | 33  | F      | other        | 15213   |
| 6      | 42  | M      | executive    | 98101   |
| 7      | 57  | M      | administrator| 91344   |
| 8      | 36  | M      | administrator| 05201   |
| 9      | 29  | M      | student      | 01002   |
| 10     | 53  | M      | lawyer       | 90703   |
| 11     | 39  | F      | other        | 30329   |
| 12     | 28  | F      | other        | 06405   |
| 13     | 47  | M      | educator     | 29206   |
| 14     | 45  | M      | scientist    | 55106   |
| 15     | 49  | F      | educator     | 97301   |
| 16     | 21  | M      | entertainment| 10309   |
```

Apache Flume

Introduction

Apache Flume is:

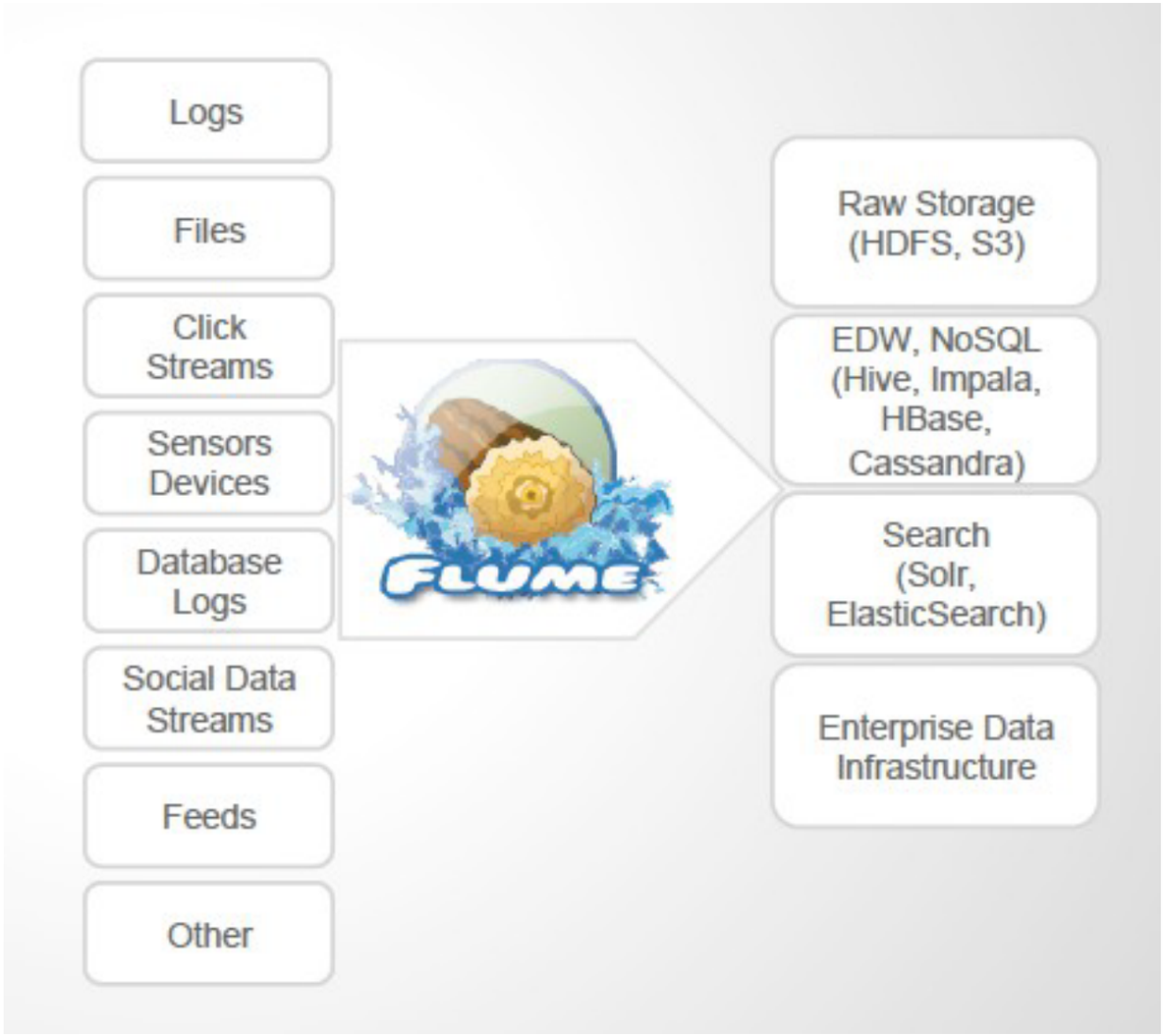
- **A distributed data transport and aggregation system for event- or log-structured data**
- **Principally designed for continuous data ingestion into Hadoop... But more flexible than that**

What is Flume?

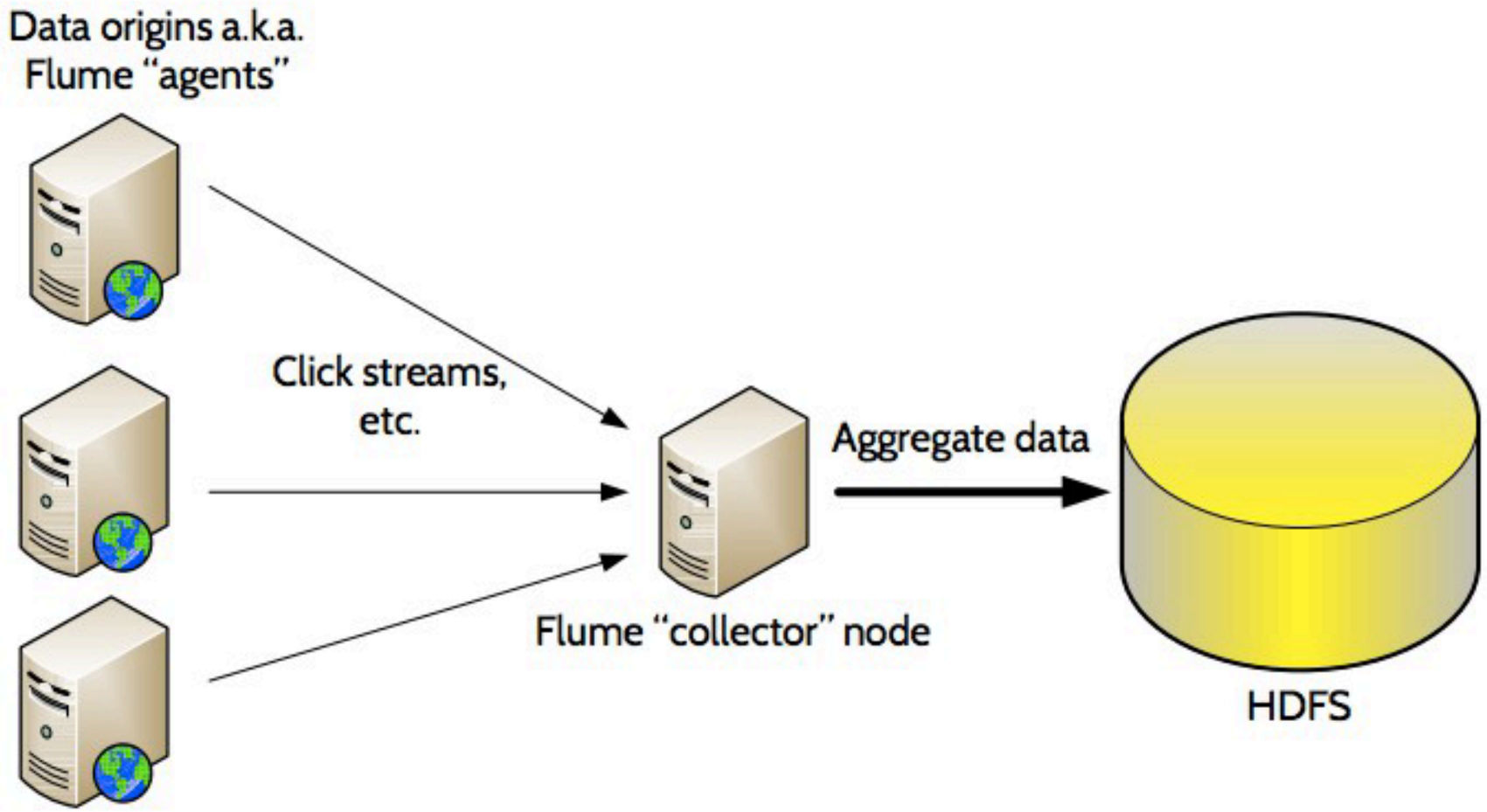
CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

Apache Flume is a continuous data ingestion system that is...

- **open-source,**
- **reliable,**
- **scalable,**
- **manageable,**
- **Customizable,**
- **and designed for Big Data ecosystem**



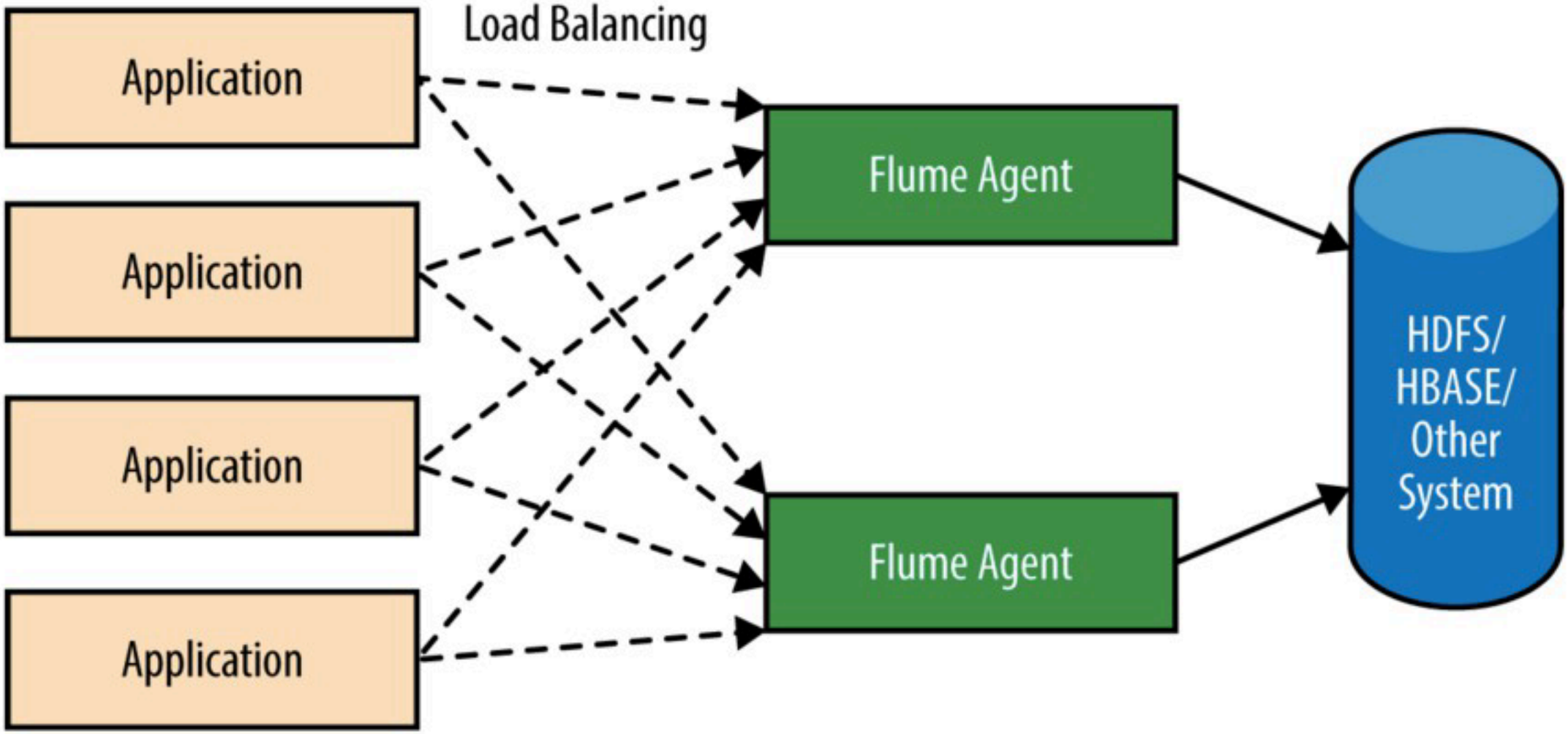
Architecture Overview



Flume Agent

- A source writes events to one or more channels.
- A channel is the holding area as events are passed from a source to a sink.
- A sink receives events from one channel only.
- An agent can have many channels.

Flow



A Simple Flow

Source: Using Flume, Hari Shreedharan, 2014

Flume Agent Configuration : Example

CHULALONGKORN
BUSINESS SCHOOL
FLAGSHIP FOR LIFE

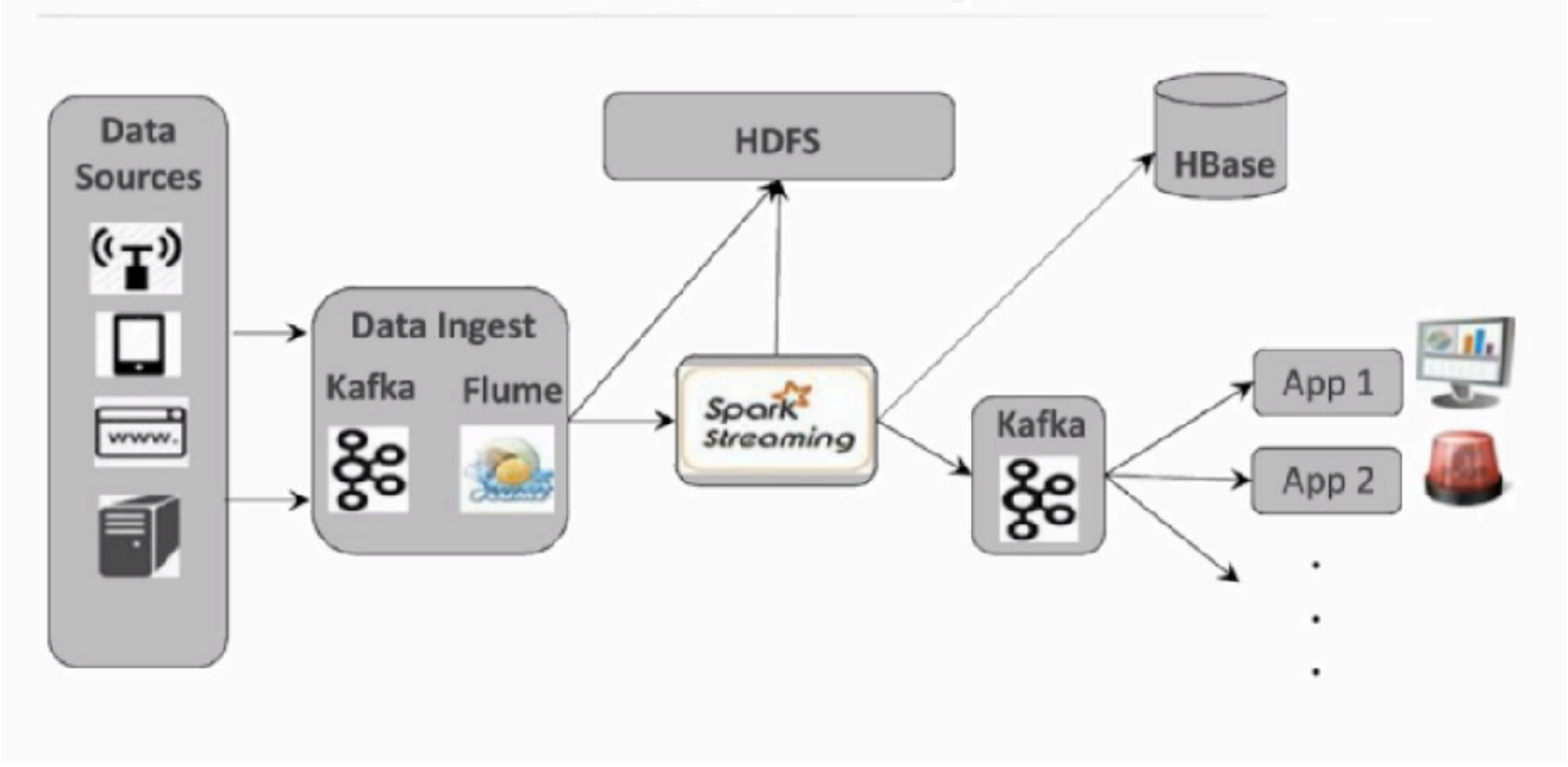
```
agent.sources = httpSrc
agent.channels = memory1 memory2
agent.sinks = hdfsSink hbaseSink

agent.sources.httpSrc.type = http
agent.sources.httpSrc.channels = memory1 memory2

# Bind to all interfaces
agent.sources.httpSrc.bind = 0.0.0.0
agent.sources.httpSrc.port = 4353

# Removing this line will disable SSL
agent.sources.httpSrc.ssl = true
agent.sources.httpSrc.keystore = /tmp/keystore
agent.sources.httpSrc.keystore-password = UsingFlume

agent.sources.httpSrc.handler = usingflume.ch03.HTTPSourceXMLHandler
agent.sources.httpSrc.handler.insertTimestamp = true

agent.sources.httpSrc.interceptors = hostInterceptor
agent.sources.httpSrc.interceptors.hostInterceptor.type = host
```

Flume Agent Configuration : Example

CHULALONGKORN
BUSINESS SCHOOL
FLAGSHIP FOR LIFE

```
# Initializes a memory channel with default configuration
agent.channels.memory1.type = memory

# Initializes a memory channel with default configuration
agent.channels.memory2.type = memory

# HDFS Sink
agent.sinks.hdfsSink.type = hdfs
agent.sinks.hdfsSink.channel = memory1
agent.sinks.hdfsSink.hdfs.path = /Data/UsingFlume/%{topic}/%Y/%m/%d/%H/%M
agent.sinks.hdfsSink.hdfs.filePrefix = UsingFlumeData

agent.sinks.hbaseSink.type = asynchbase
agent.sinks.hbaseSink.channel = memory2
agent.sinks.hbaseSink.serializer = usingflume.ch05.AsyncHBaseDirectSerializer
agent.sinks.hbaseSink.table = usingFlumeTable
```

Stream Processing Architecture



Flume Loading Data to HDFS

$ cd /etc/flume-ng/conf/

$ sudo rm flume.conf

$sudo wget https://github.com/bobbylovemovie/trainbigdata/raw/master/flume/flume.conf

$cat flume.conf

```
agent.sources = netsource
agent.sinks = hdfssink
agent.channels = memorychannel
agent.sources.netsource.type = netcat
agent.sources.netsource.bind = localhost
agent.sources.netsource.port = 3030
agent.sources.netsource.interceptors = ts
agent.sources.netsource.interceptors.ts.type = org.apache.flume.interceptor.TimestampInterceptor$Builder
agent.sinks.hdfssink.type = hdfs
agent.sinks.hdfssink.hdfs.path = hdfs://localhost:8020/user/cloudera/flume/events
agent.sinks.hdfssink.hdfs.filePrefix = log
agent.sinks.hdfssink.hdfs.rollInterval = 0
agent.sinks.hdfssink.hdfs.rollCount = 5
agent.sinks.hdfssink.hdfs.fileType = DataStream
agent.channels.memorychannel.type = memory
agent.channels.memorychannel.capacity = 100
agent.channels.memorychannel.transactionCapacity = 100
agent.sources.netsource.channels = memorychannel
agent.sinks.hdfssink.channel = memorychannel
```

Flume Loading Data to HDFS

start flume-service

$sudo service flume-ng-agent restart

```
[cloudera@quickstart ~]$ sudo service flume-ng-agent restart
Flume agent is not running                                 [  OK  ]
Starting Flume NG agent daemon (flume-ng-agent):           [  OK  ]
```

start flume Agent

$sudo flume-ng agent --conf /etc/flume-ng/conf/ --conf-file /etc/flume-ng/conf/flume.conf --name agent -Dflume.root.logger=INFO,console

```
2016-10-31 03:20:39,476 (lifecycleSupervisor-1-3) [INFO - org.apache.flume.sourc
e.NetcatSource.start(NetcatSource.java:169)] Created serverSocket:sun.nio.ch.Ser
verSocketChannelImpl[/127.0.0.1:3030]
```

Flume Loading Data to HDFS

Datasource Connect By Telnet

Open New Terminal

$sudo yum install telnet

$telnet localhost 3030

```
[cloudera@quickstart ~]$ telnet localhost 3030
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
Hello Bigdata from Bobby Thaveewat
OK
a,b,c
OK
d,e,f
OK
g,h,i
OK
j,k,l
OK
```

View Result

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

File Browser

View as binary

Edit file

Download

View file location

Refresh

Last modified
09/01/2017
4:01 AM

User
root

Group
cloudera

Size
28 B

Home

Page 1 to 1 of 1

/ user / cloudera / flume / events / **log.1504238424110**

```
a,b,c
d,e,f
g,h,i
j,k,l
```

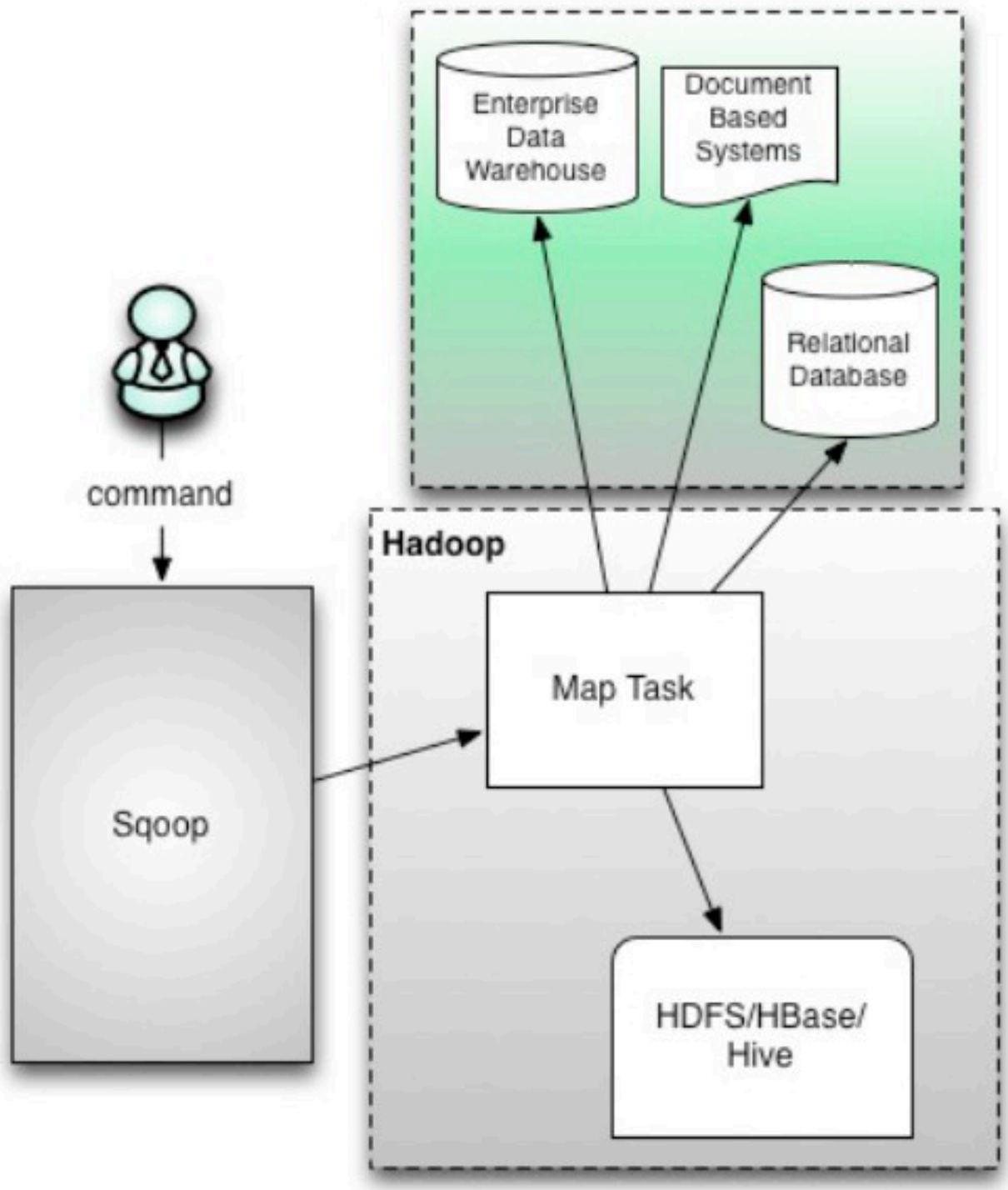
APACHE SQOOP

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**

Architecture Overview



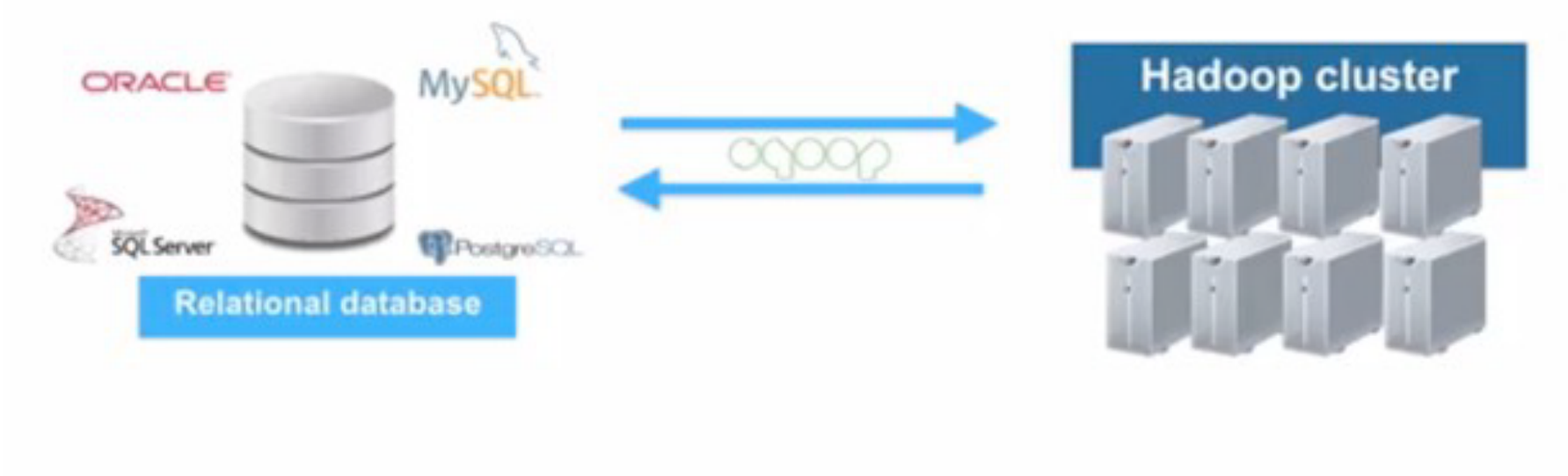
Sqoop Benefit

- Leverages RDBMS metadata to get the column data types
- It is simple to script and uses SQL
- It can be used to handle change data capture by importing daily transactional data to Hadoop
- It uses MapReduce for export and import that enables parallel and efficient data movement

Sqoop Mode

Sqoop import: Data moves from RDBMS to Hadoop

Sqoop export: Data moves from Hadoop to RDBMS



Use Case : Data Consolidation

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

- Integrate data from various organizational “data stores” to Hadoop for various data processing requirements

Source: Mastering Apache Sqoop, David Yahalom, 2016

Import Commands

Parameters	Description
`--connect <jdbc-uri>`	Specifies the server or database to connect to. It also specifies the port. For example: `--connect jdbc:mysql://host:port/databaseName`
`--connection-manager <class-name>`	Specifies the connection manager class name.
`--driver <class-name>`	Specifies the fully qualified name of the JDBC driver class.
`--hadoop-home <dir>`	This parameter is used to override the `$HADOOP_HOME` environment variable.
`-P`	If a user doesn't want to specify the database password along with the command, we can use the `-P` option to read the password from the console.
`--password <password>`	Sets the authentication password required to connect to the input source.
`--username <username>`	Sets the authentication username.
`--connection-param-file <properties-file>`	Specifies the connection parameter's file.
`--help`	This option will provide the usage instructions.
`--verbose`	Prints more information during a query execution.

Export Commands

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

Parameters	Description
`--direct`	Use the direct mode to perform the export quickly. Note that it is only supported for MySQL.
`--export-dir<dir>`	The location of input files in HDFS.
`--table <table-name>`	Name of the output table (the RDBMS table).
`-m,--num-mappers <n>`	Refers to the number of map tasks.
`--update-mode <mode>`	Specifies how updates are performed when new rows are found with non-matching keys in the database. Legal values for the mode include `updateonly` (default) and `allowinsert`.
`--update-key <col-name>`	The value of this column is used to identify the records that a user wants to update during the update mode. Use a comma-separated list of columns if there is more than one column.
`--staging-table <staging-table-name>`	Specifies the name of the staging table. The staging table is used to stage the data before inserting it into the destination table.
`--clear-staging-table`	This argument is used to clean the data from the staging table.

Loading Data from RDBMS to Hadoop

CHULALONGKORN BUSINESS SCHOOL

FLAGSHIP FOR LIFE

Configuring MySQL On Cloudera.Quickstart

```
$ sudo /usr/bin/mysql_secure_installation
Enter current password for root (enter for none): cloudera
OK, successfully used password, moving on...
Set root password? [Y/n] N
Remove anonymous users? [Y/n] Y
Disallow root login remotely? [Y/n] N
Remove test database and access to it [Y/n] Y
Reload privilege tables now? [Y/n] Y
All done!
```

Running MySQL

$ mysql -uroot -p"cloudera"

```
[cloudera@quickstart ~]$ mysql -uroot -p"cloudera"
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 389
Server version: 5.1.73 Source distribution

Copyright (c) 2000, 2013, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

mysql> show databases;

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| cm                 |
| firehose           |
| hue                |
| metastore          |
| mysql              |
| nav                |
| navms              |
| oozie              |
| retail_db          |
| rman               |
| sentry             |
+--------------------+
12 rows in set (0.01 sec)
```

Prepare a test database table

CHULALONGKORN
BUSINESS SCHOOL
FLAGSHIP FOR LIFE

```
mysql> CREATE DATABASE test_mysql_db;
mysql> USE test_mysql_db;
mysql> CREATE TABLE country_tbl(id INT NOT NULL, country
VARCHAR(50), PRIMARY KEY (id));
mysql> INSERT INTO country_tbl VALUES(1, 'USA');
mysql> INSERT INTO country_tbl VALUES(2, 'CANADA');
mysql> INSERT INTO country_tbl VALUES(3, 'Mexico');
mysql> INSERT INTO country_tbl VALUES(4, 'Brazil');
mysql> INSERT INTO country_tbl VALUES(61, 'Japan');
mysql> INSERT INTO country_tbl VALUES(65, 'Singapore');
mysql> INSERT INTO country_tbl VALUES(66, 'Thailand');
```

View data in the table

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

mysql> SELECT * FROM country_tbl;

```
+----+-----------+
| id | country   |
+----+-----------+
|  1 | USA       |
|  2 | CANADA    |
|  3 | Mexico    |
|  4 | Brazil    |
| 61 | Japan     |
| 65 | Singapore |
| 66 | Thailand  |
+----+-----------+
7 rows in set (0.00 sec)
```

mysql> exit;

Importing data from MySQL to HDFS

CHULALONGKORN BUSINESS SCHOOL

FLAGSHIP FOR LIFE

```
$ sqoop import --connect jdbc:mysql://localhost/test_mysql_db --
username root --password cloudera --table country_tbl --target-dir /
user/cloudera/test_table -m 1
```

Importing data from MySQL to Hive Table

$ sqoop import --connect jdbc:mysql://localhost/test_mysql_db --username root --password cloudera --table country_tbl --hive-import --hive-table country -m 1

Reviewing data from Hive Table

CHULALONGKORN BUSINESS SCHOOL

FLAGSHIP FOR LIFE

[cloudera@quickstart ~]$ hive

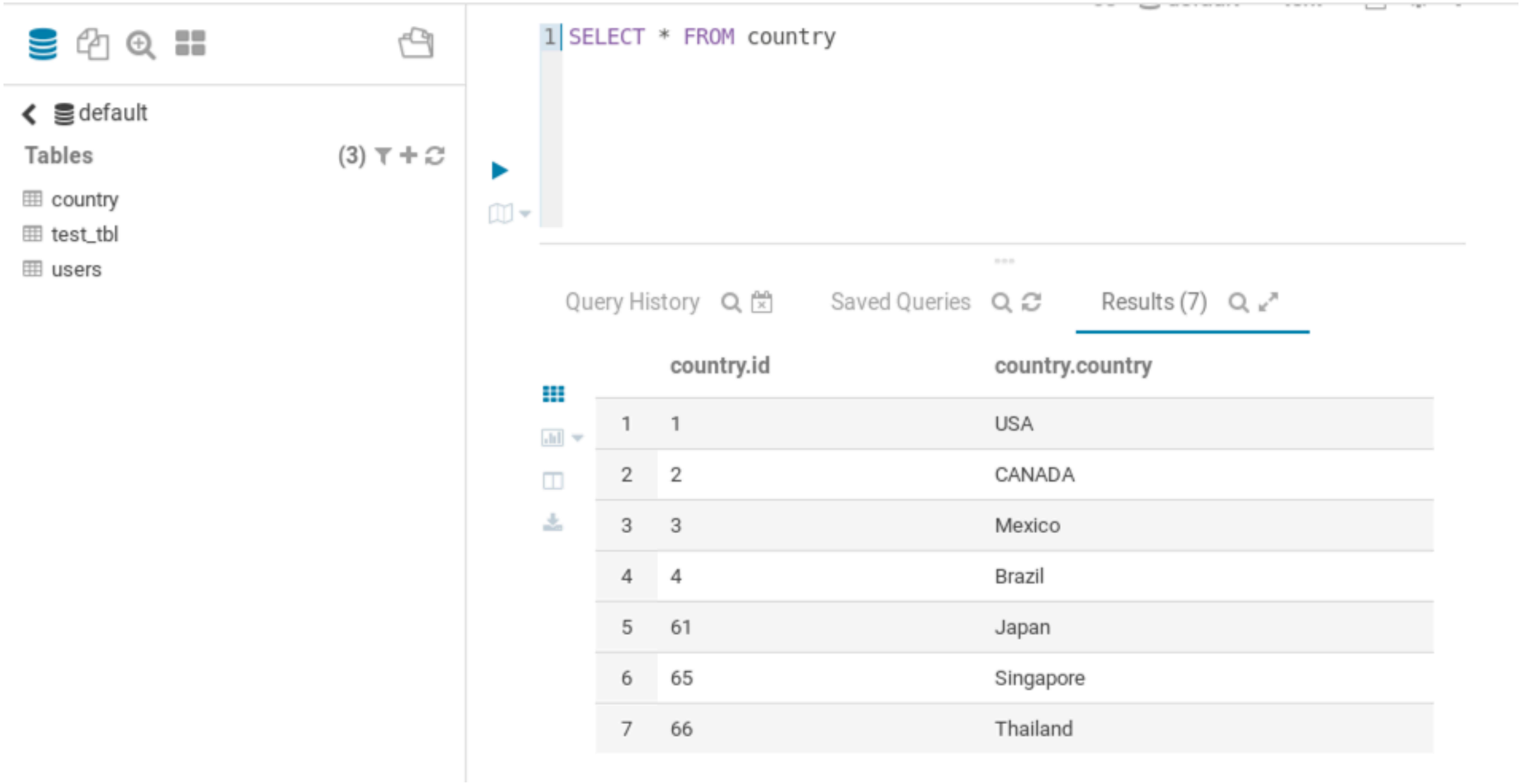
```
Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j.prope
rties
WARNING: Hive CLI is deprecated and migration to Beeline is recommended.
```

hive> show tables;

hive> select * from country;

```
1       USA
2       CANADA
3       Mexico
4       Brazil
61      Japan
65      Singapore
66      Thailand
Time taken: 0.587 seconds, Fetched: 7 row(s)
```

Running from Hue: Beewax



	country.id	country.country
1	1	USA
2	2	CANADA
3	3	Mexico
4	4	Brazil
5	61	Japan
6	65	Singapore
7	66	Thailand

Importing data from MySQL to HBase

CHULALONGKORN BUSINESS SCHOOL

FLAGSHIP FOR LIFE

$ sqoop import --connect jdbc:mysql://localhost/test_mysql_db --username root --password cloudera --table country_tbl --hbase-table country --column-family hbase_country_cf --hbase-row-key id --hbase-create-table -m 1

Start HBase

$ hbase shell

hbase(main):001:0> list

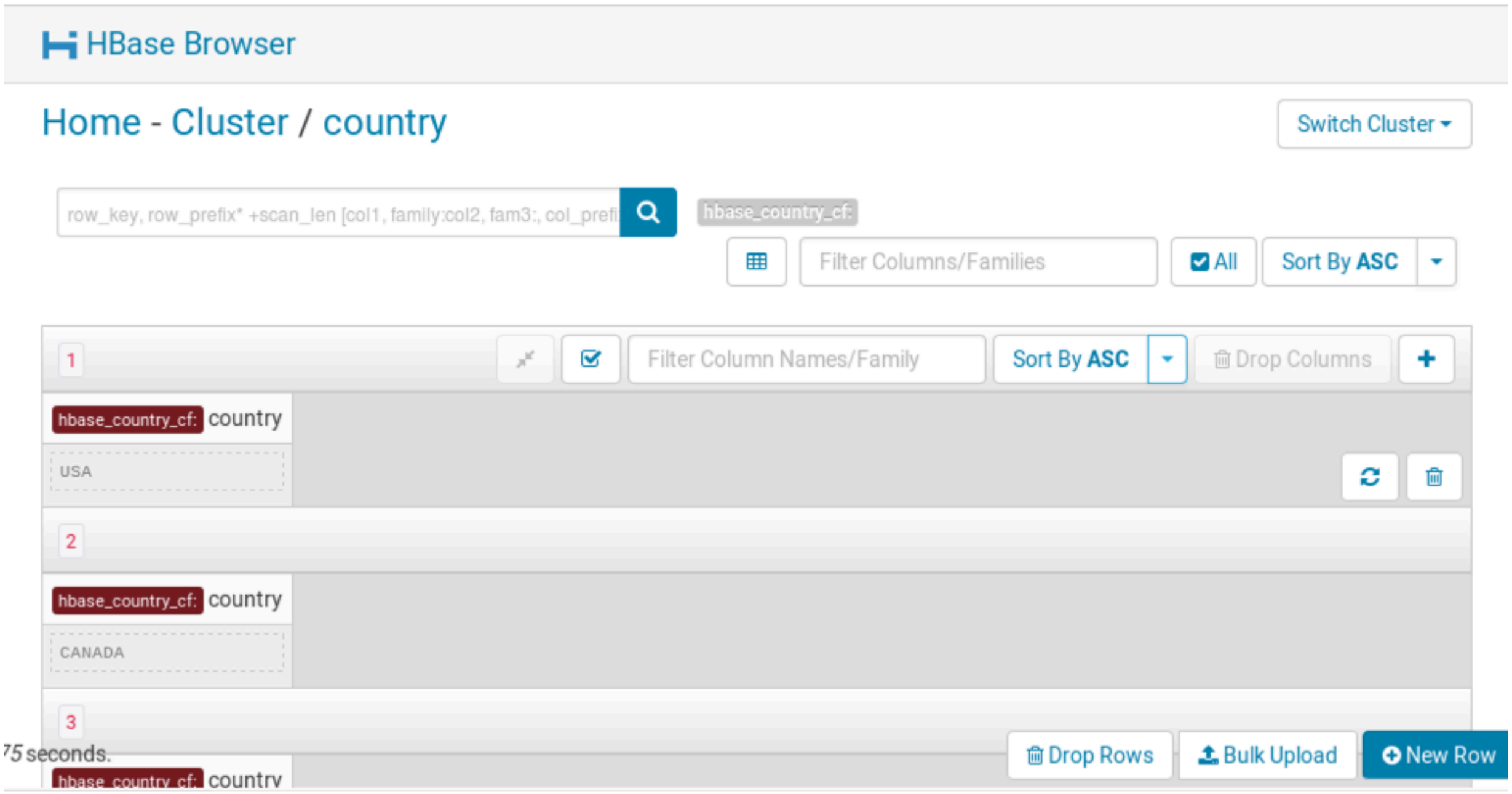
```
hbase(main):001:0> list
TABLE
country
employee
student
3 row(s) in 0.3720 seconds

=> ["country", "employee", "student"]
```

Viewing Hbase data

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

```
hbase(main):003:0> scan 'country'
ROW                     COLUMN+CELL
 1                      column=hbase_country_cf:country, timestamp=1468081466623, val
                        ue=USA
 2                      column=hbase_country_cf:country, timestamp=1468081466623, val
                        ue=CANADA
 3                      column=hbase_country_cf:country, timestamp=1468081466623, val
                        ue=Mexico
 4                      column=hbase_country_cf:country, timestamp=1468081466623, val
                        ue=Brazil
 61                     column=hbase_country_cf:country, timestamp=1468081466623, val
                        ue=Japan
 65                     column=hbase_country_cf:country, timestamp=1468081466623, val
                        ue=Singapore
 66                     column=hbase_country_cf:country, timestamp=1468081466623, val
                        ue=Thailand
7 row(s) in 0.1670 seconds
```

Viewing data from Hbase browser




Spark

Introduction

A fast and general engine for large scale data processing

An open source big data processing framework built around speed, ease of use, and sophisticated analytics. Spark enables applications in Hadoop clusters to run up to 100 times faster in memory and 10 times faster even when running on disk.

What is Spark ?

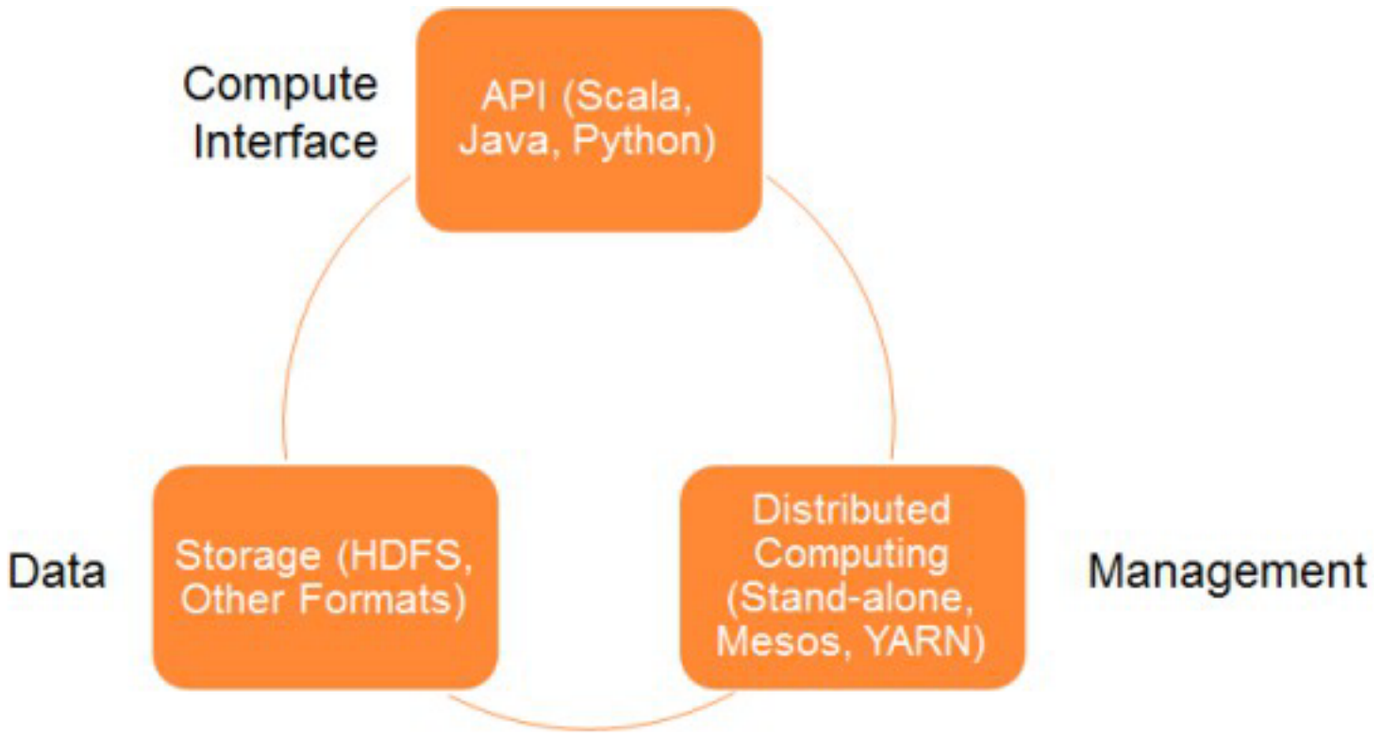
CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

Framework for distributed processing.

In-memory, fault tolerant data structures

Flexible APIs in Scala, Java, Python, SQL, R

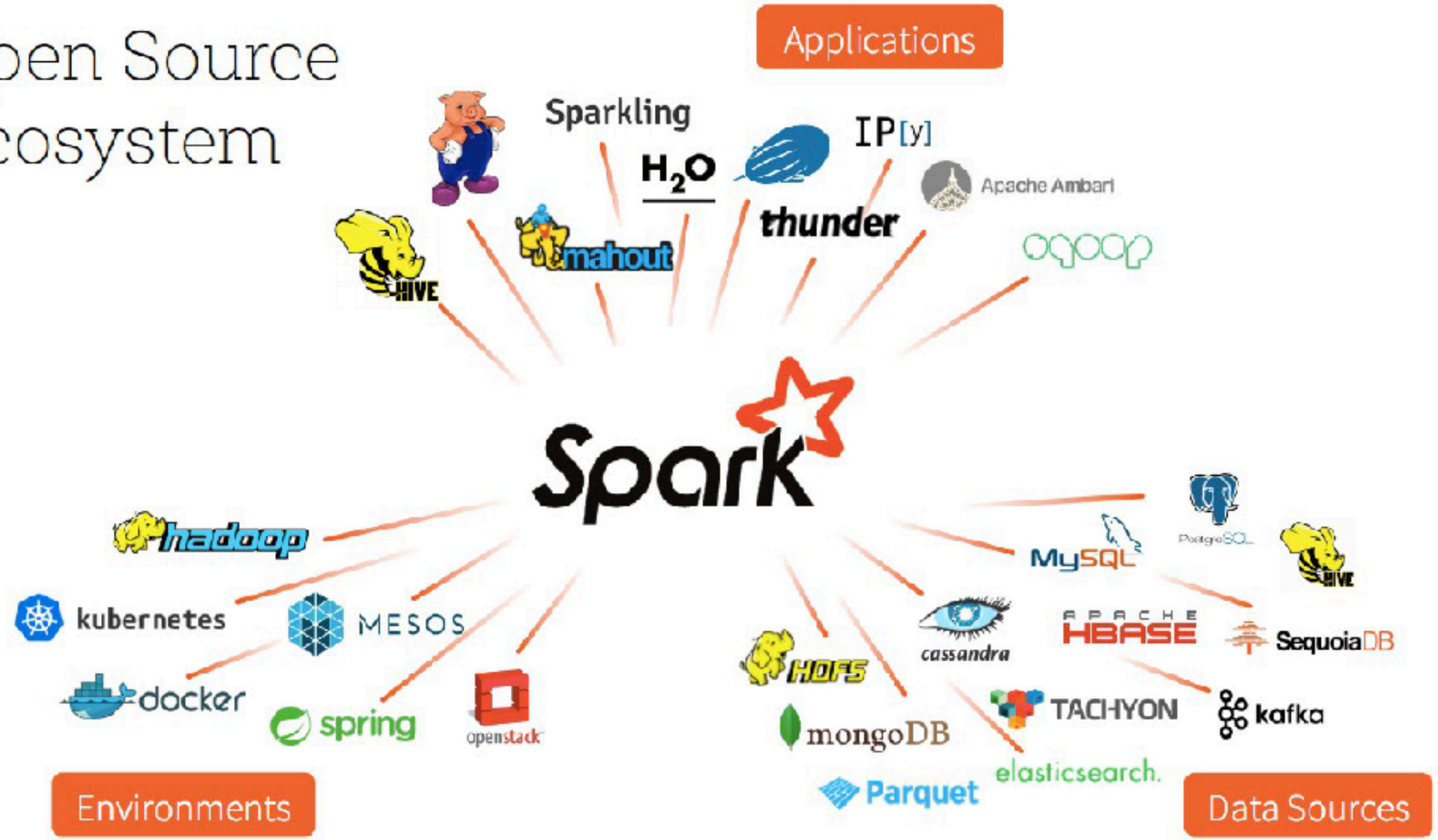
Open source



Why Spark ?

- Handle Petabytes of data
- Significant faster than MapReduce
- Simple and intutive APIs
- General framework
  - Runs anywhere
  - Handles (most) any I/O
  - Interoperable libraries for specific use-cases

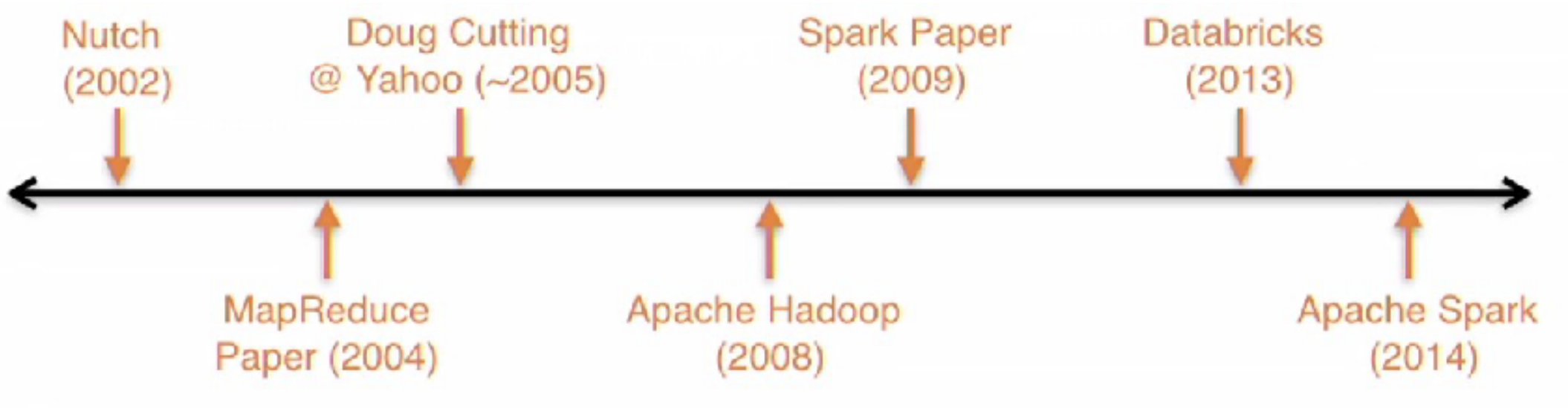
Open Source Ecosystem

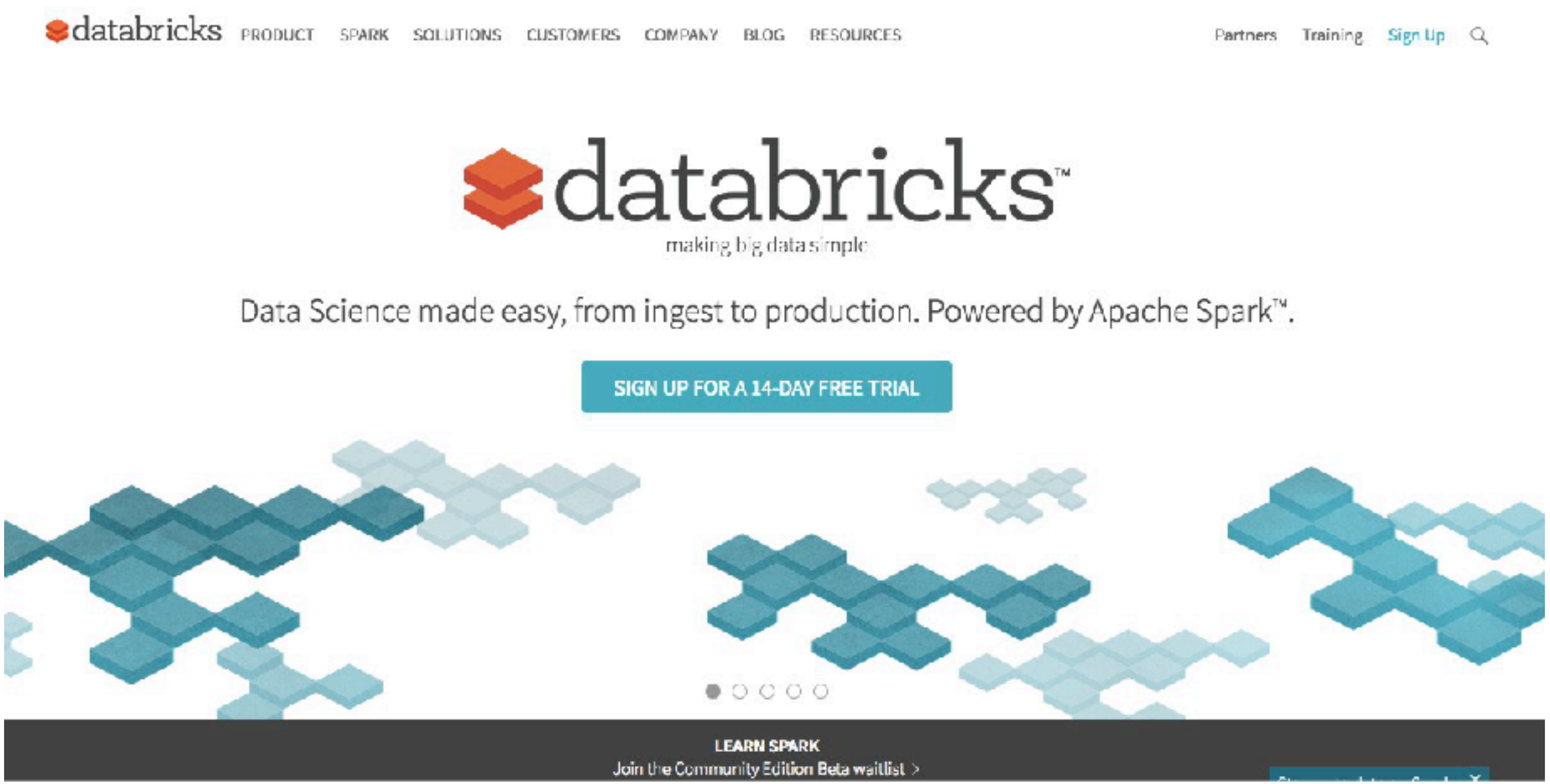


Source: Jump start into Apache Spark and Databricks

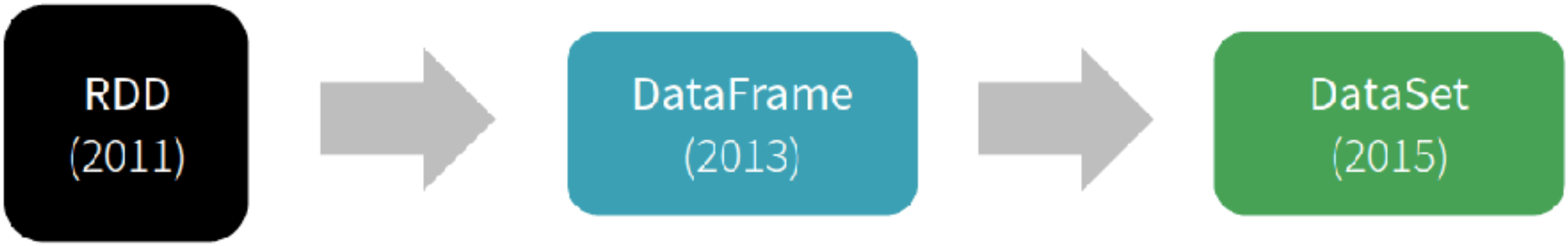
Spark: History

- Founded by AMPlab, UC Berkeley
- Created by Matei Zaharia (PhD Thesis)
- Maintained by Apache Software Foundation
- Commercial support by Databricks




databricks
PRODUCT
SPARK
SOLUTIONS
CUSTOMERS
COMPANY
BLOG
RESOURCES
Partners
Training
Sign Up
databricks™
making big data simple
Data Science made easy, from ingest to production. Powered by Apache Spark™.
SIGN UP FOR A 14-DAY FREE TRIAL
LEARN SPARK
Join the Community Edition Beta waitlist >

History of Spark APIs



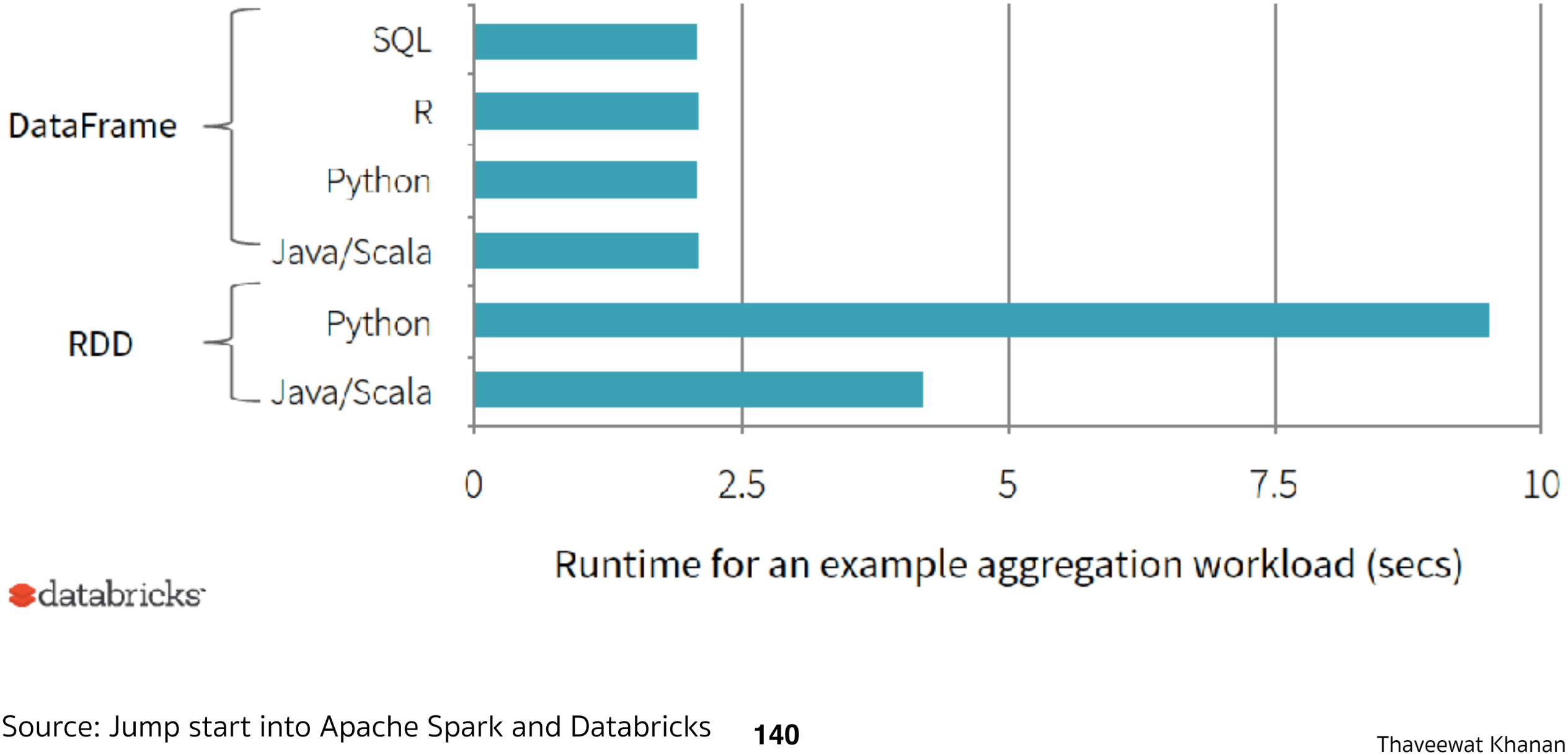
- Distribute collection of JVM objects
- Functional Operators (map, filter, etc.)

- Distribute collection of Row objects
- Expression-based operations and UDFs
- Logical plans and optimizer
- Fast/efficient internal representations

- Internally rows, externally JVM objects
- "Best of both worlds" **type safe + fast**

Source: Jump start into Apache Spark and Databricks

Benefit of Logical Plan: Performance Parity Across Languages



databricks

Source: Jump start into Apache Spark and Databricks

What is a RDD ?

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

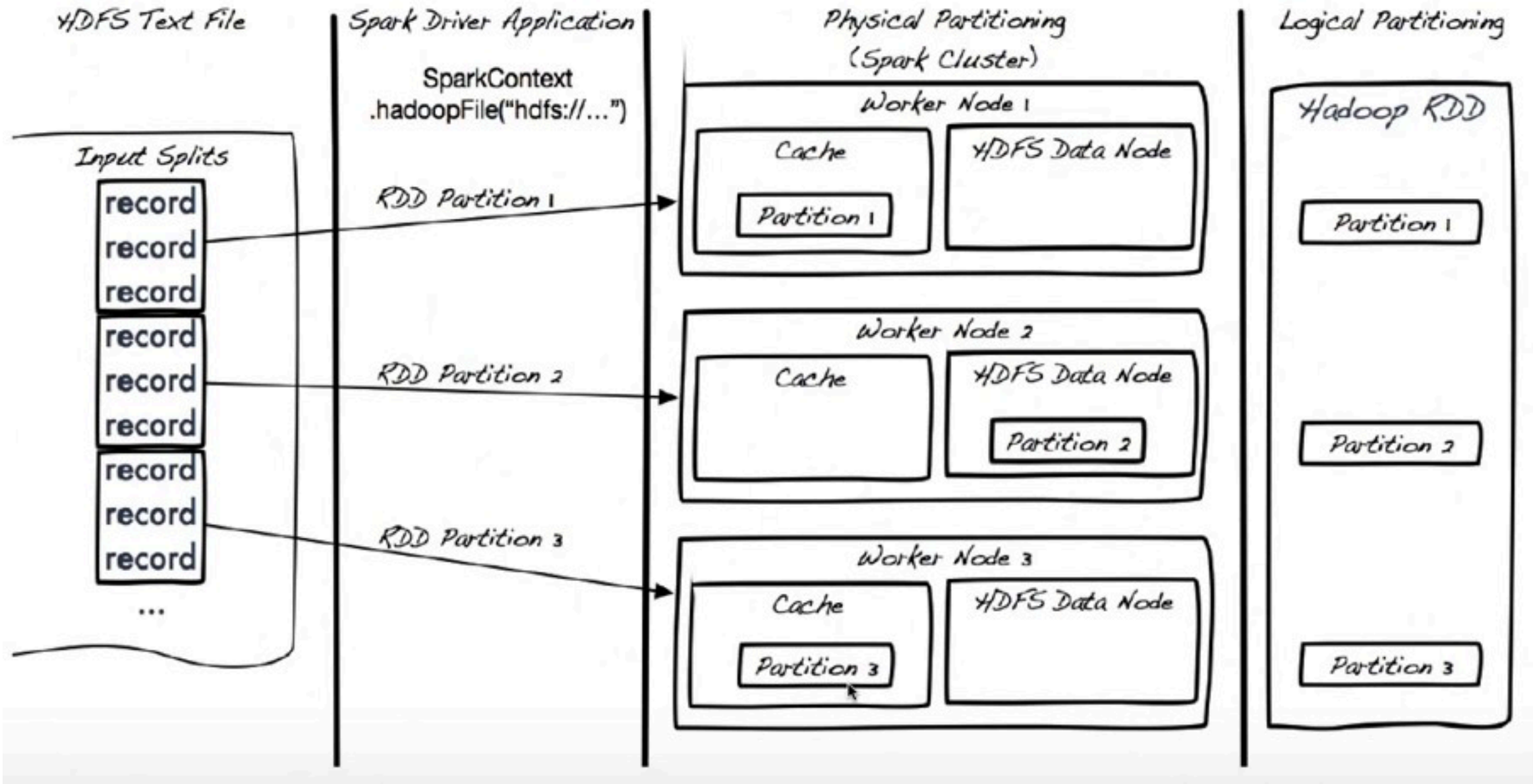
Resilient: if the data in memory (or on a node) is lost, it can be recreated.

Distributed: data is chucked into partitions and stored in memory acress the custer.

Dataset: initial data can come from a table or be created programmatically

RDD Creation

hdfsData = sc.textFile("hdfs://data.txt")

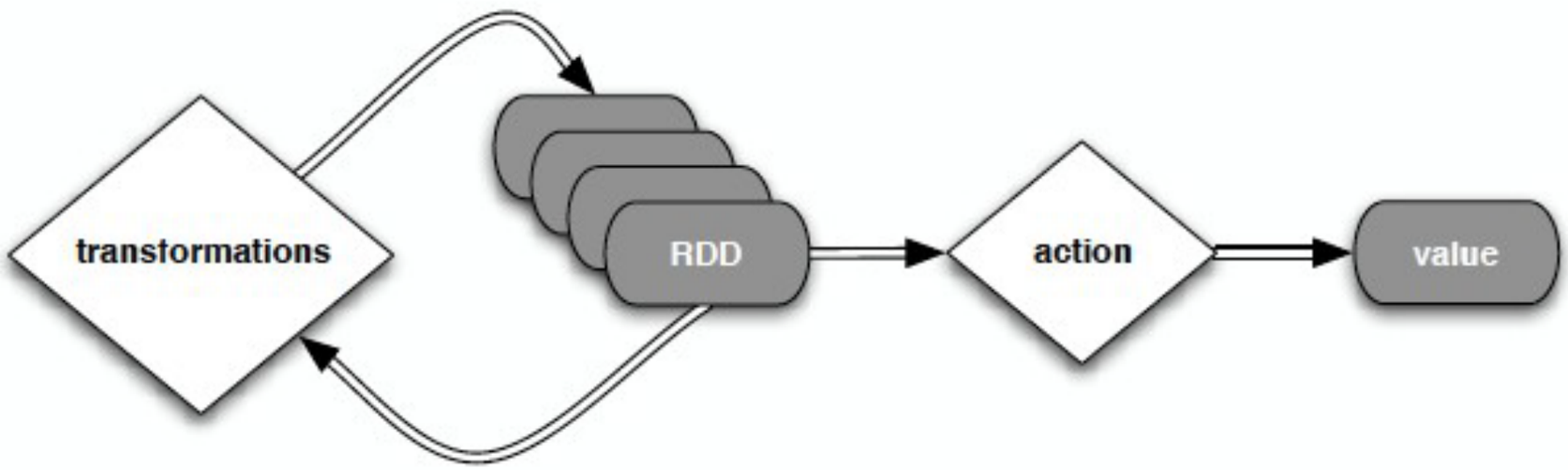


Source: Pspark: A brain-friendly introduction

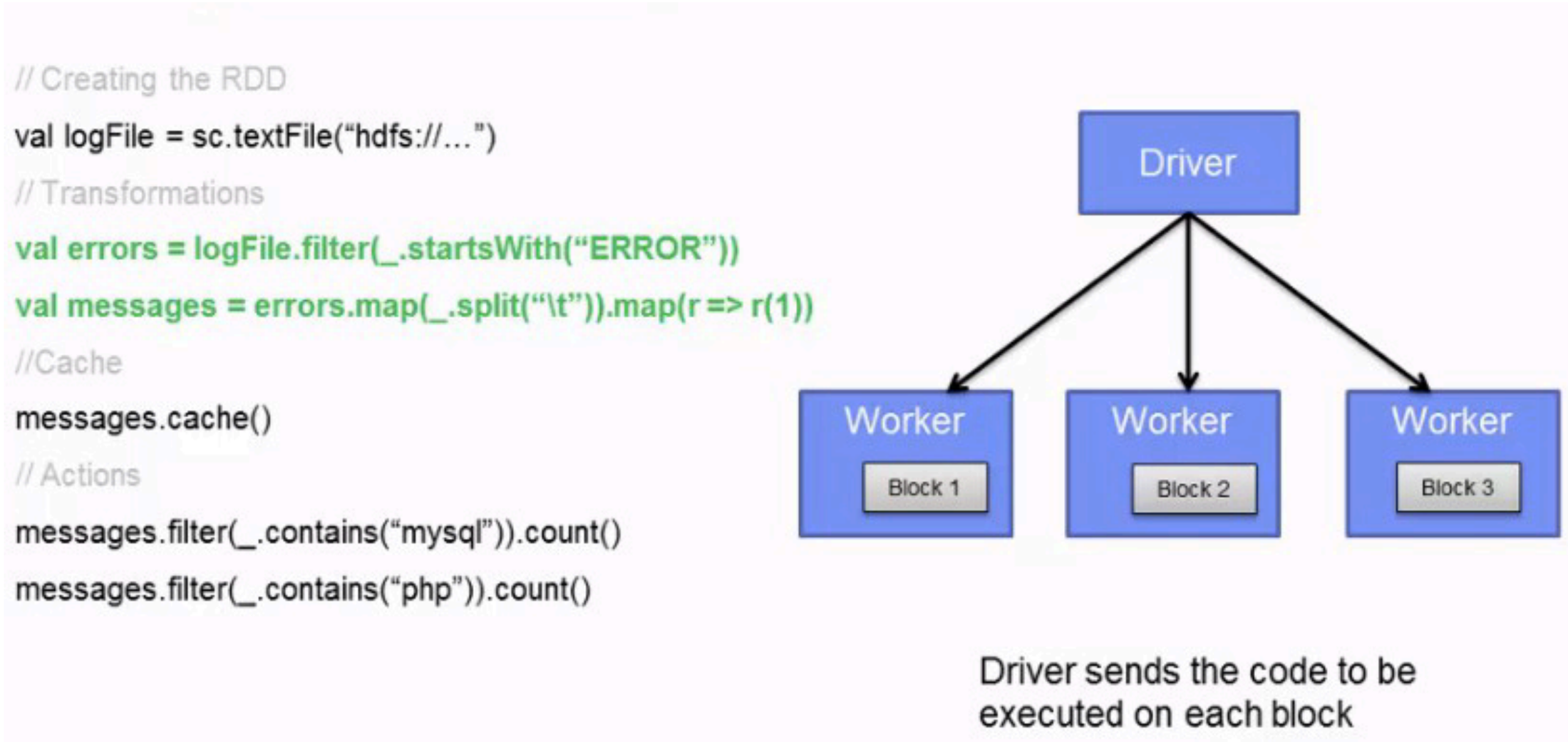
RDD: Operations

Transformations: transformations are lazy (not computed immediately)

Actions: the transformed RDD gets recomputed when an action is run on it (default)

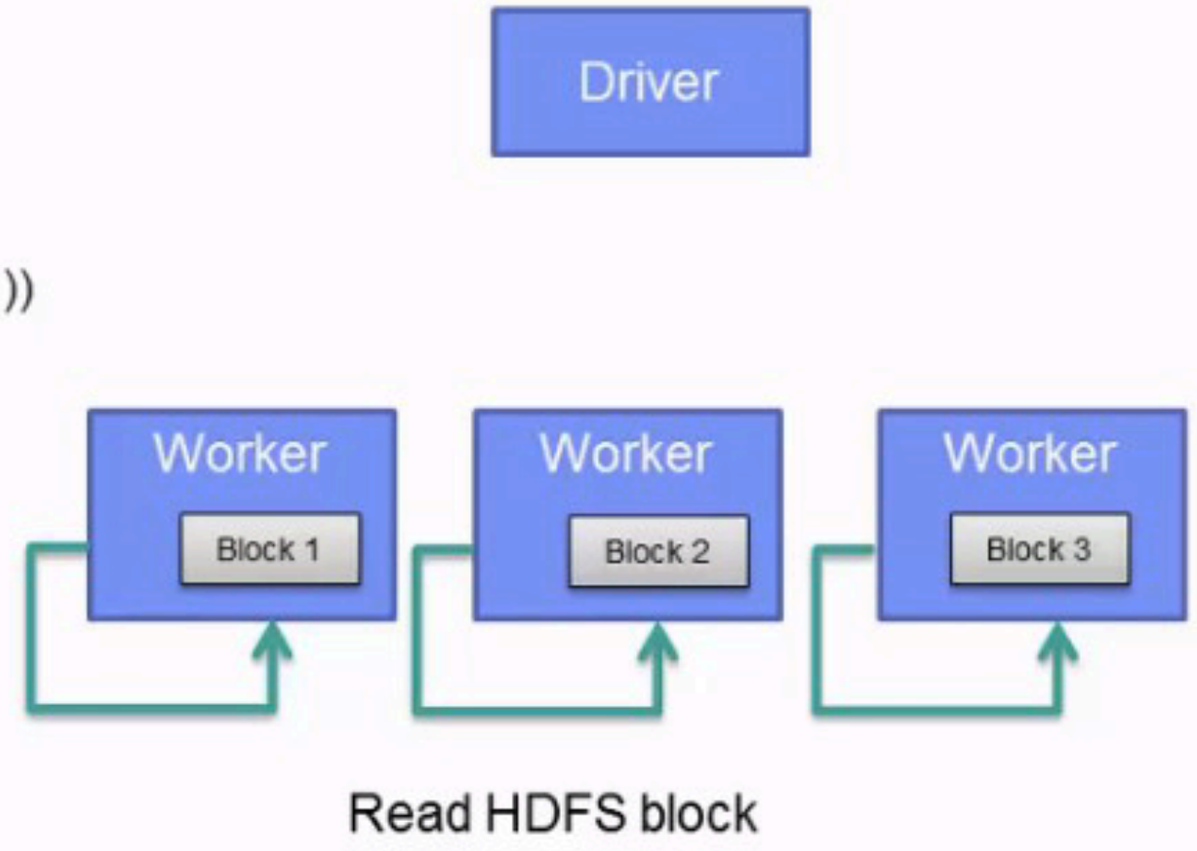


What happens when an action is executed



What happens when an action is executed


```
// Creating the RDD
val logFile = sc.textFile("hdfs://...")
// Transformations
val errors = logFile.filter(_.startsWith("ERROR"))
val messages = errors.map(_.split("\t")).map(r => r(1))
//Caching
messages.cache()
// Actions
messages.filter(_.contains("mysql")).count()
messages.filter(_.contains("php")).count()
```

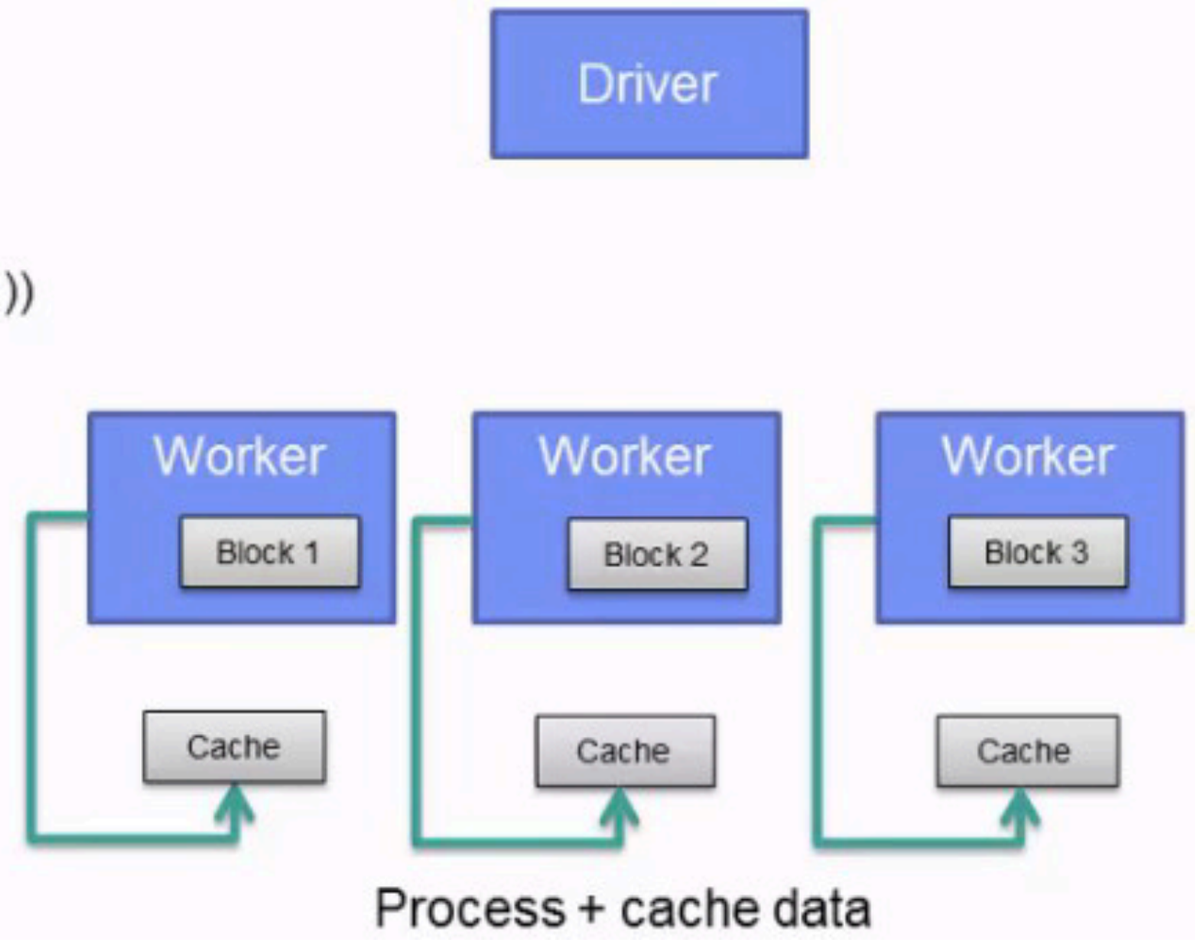


What happens when an action is executed

CHULALONGKORN BUSINESS SCHOOL

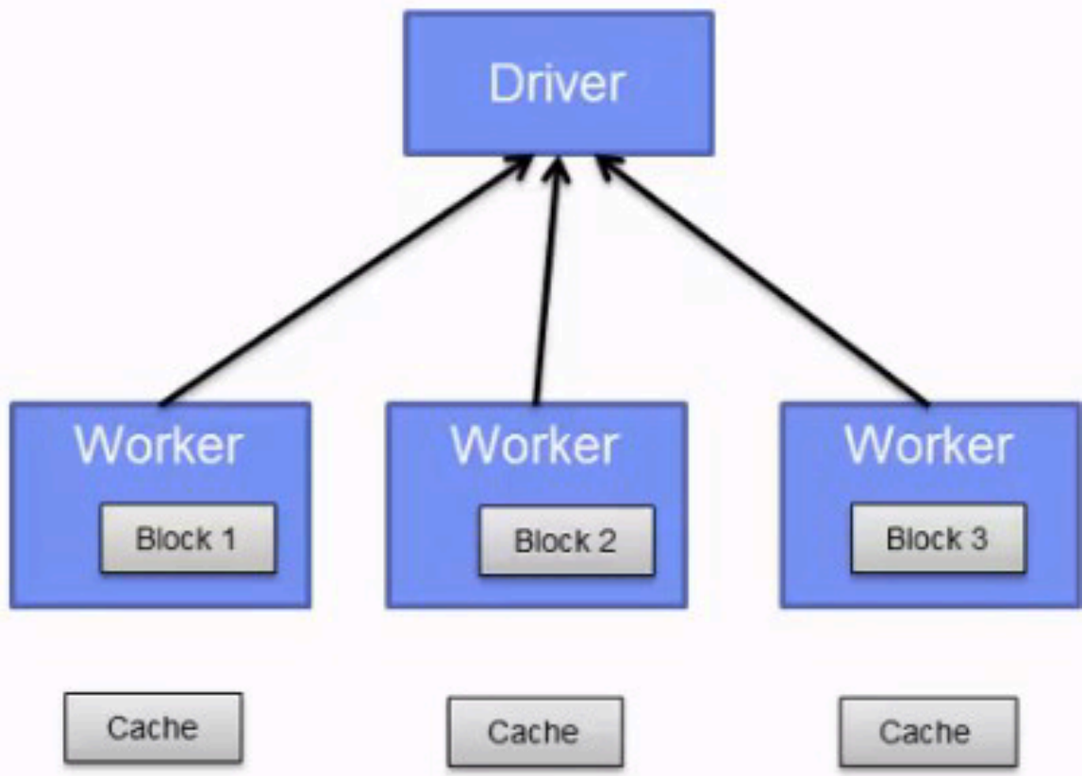
FLAGSHIP FOR LIFE

```
// Creating the RDD
val logFile = sc.textFile("hdfs://...")
// Transformations
val errors = logFile.filter(_.startsWith("ERROR"))
val messages = errors.map(_.split("\t")).map(r => r(1))
//Caching
messages.cache()
// Actions
messages.filter(_.contains("mysql")).count()
messages.filter(_.contains("php")).count()
```

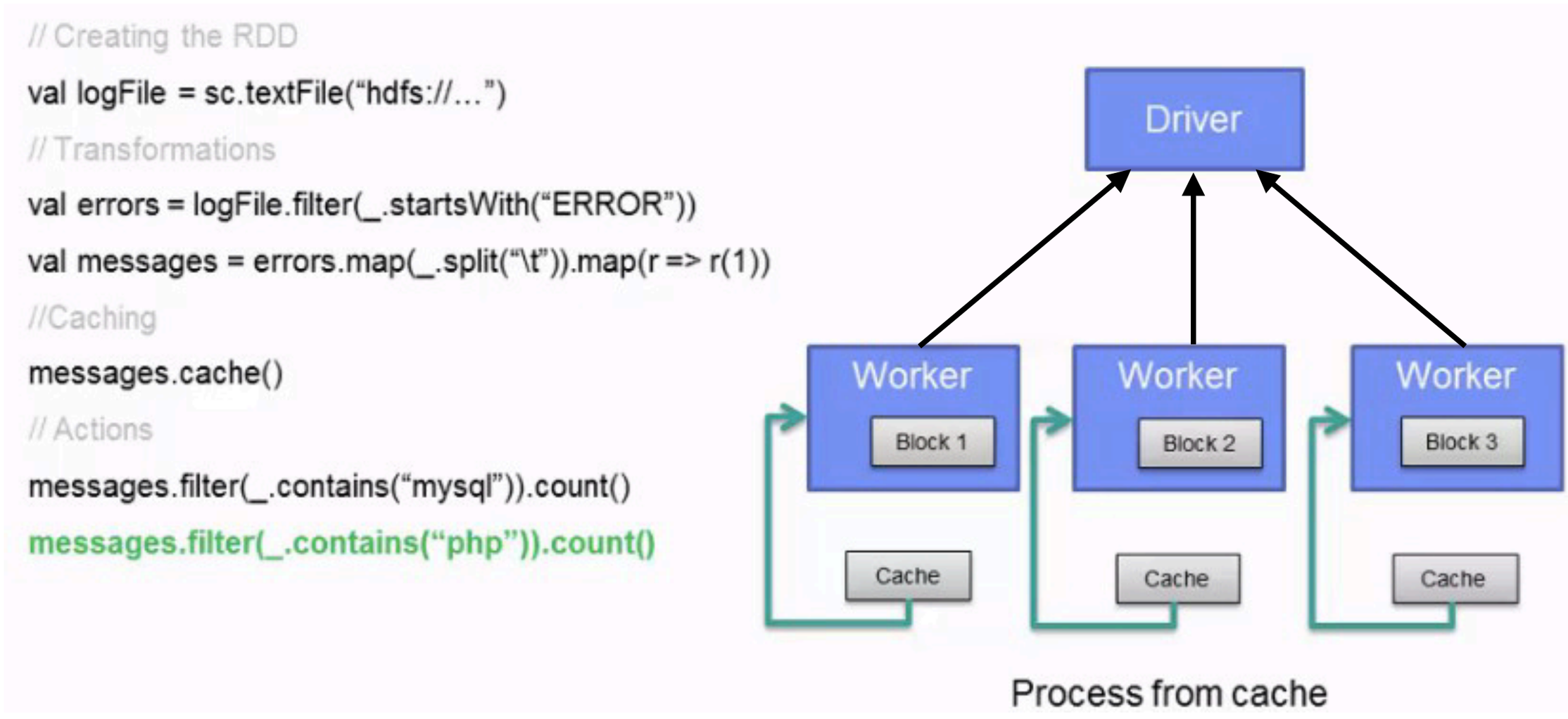


What happens when an action is executed


```
// Creating the RDD
val logFile = sc.textFile("hdfs://...")
// Transformations
val errors = logFile.filter(_.startsWith("ERROR"))
val messages = errors.map(_.split("\t")).map(r => r(1))
//Caching
messages.cache()
// Actions
messages.filter(_.contains("mysql")).count()
messages.filter(_.contains("php")).count()
```



What happens when an action is executed



What happens when an action is executed

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

```
16/11/01 01:03:54 INFO storage.BlockManagerMaster: Registered BlockManager
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /__ / .__/\_,_/_/ /_/\_\   version 1.6.0
      /_/

Using Python version 2.6.6 (r266:84292, Jul 23 2015 15:22:56)
SparkContext available as sc, HiveContext available as sqlContext.
>>> rdd1 = sc.parallelize(range(1, 11))
>>> rdd1 = rdd1.filter(lambda n : n % 2 == 0)
>>> rdd1 = rdd1.map(lambda n : n ** 2)
>>> rdd1 = rdd1.map(lambda n : n / 2)
>>> ans = rdd1.reduce(lambda a, b : a + b)
16/11/01 01:04:52 INFO scheduler.DAGScheduler: ResultStage 0 (reduce at <stdin>:
1) finished in 0.360 s
16/11/01 01:04:52 INFO scheduler.TaskSetManager: Finished task 0.0 in stage 0.0
(TID 0) in 329 ms on localhost (1/1)
16/11/01 01:04:52 INFO scheduler.TaskSchedulerImpl: Removed TaskSet 0.0, whose t
asks have all completed, from pool
16/11/01 01:04:52 INFO scheduler.DAGScheduler: Job 0 finished: reduce at <stdin>
:1, took 0.990547 s
>>> print(ans)
110
```

[1,2,3,4,5,6,7,8,9,10]

[2,4,6,8,10]

[4, 16, 36, 64, 100]

[2, 8, 18, 32, 50]

Spark:Transformation

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

transformation	*description*
map(*func*)	return a new distributed dataset formed by passing each element of the source through a function *func*
filter(*func*)	return a new dataset formed by selecting those elements of the source on which *func* returns true
flatMap(*func*)	similar to map, but each input item can be mapped to 0 or more output items (so *func* should return a Seq rather than a single item)
sample(*withReplacement, fraction, seed*)	sample a fraction *fraction* of the data, with or without replacement, using a given random number generator seed
union(*otherDataset*)	return a new dataset that contains the union of the elements in the source dataset and the argument
distinct([*numTasks*]))	return a new dataset that contains the distinct elements of the source dataset

Spark:Transformation

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

transformation	*description*
groupByKey([*numTasks*])	when called on a dataset of (K, V) pairs, returns a dataset of (K, Seq[V]) pairs
reduceByKey(*func*, [*numTasks*])	when called on a dataset of (K, V) pairs, returns a dataset of (K, V) pairs where the values for each key are aggregated using the given reduce function
sortByKey([*ascending*], [*numTasks*])	when called on a dataset of (K, V) pairs where K implements Ordered, returns a dataset of (K, V) pairs sorted by keys in ascending or descending order, as specified in the boolean ascending argument
join(*otherDataset*, [*numTasks*])	when called on datasets of type (K, V) and (K, W), returns a dataset of (K, (V, W)) pairs with all pairs of elements for each key
cogroup(*otherDataset*, [*numTasks*])	when called on datasets of type (K, V) and (K, W), returns a dataset of (K, Seq[V], Seq[W]) tuples – also called groupWith
cartesian(*otherDataset*)	when called on datasets of types T and U, returns a dataset of (T, U) pairs (all pairs of elements)

Single RDD Transformation

filter females to analyze female buying patterns

```
male1, male2, female1 -> female1
```

map squared values

```
2, 5, 6 -> 4, 25, 36
```

flatMap to break up a sentence into words

```
my name is ray -> my, name, is, ray
```

find the distinct values in a dataset

```
apple, apple, banana -> apple, banana
```

sample two values at random

```
apple, banana, guava -> banana, apple
```

Source: Jspark: A brain-friendly introduction

Multiple RDD Transformation

union

```
apple, orange, banana, guava,
banana, pear
```

intersection

```
banana
```

Dataset A
apple
orange
banana

subtract anything shown in Dataset B from Dataset A

```
apple, orange
```

cartesian (every possible pair combo)

```
(apple, guava), (apple, banana), ...
```

Dataset B
guava
banana
pear

Source: Jspark: A brain-friendly introduction

Pair RDD Transformation

- reduceByKey
- groupByKey
- combineByKey
- mapValues
- flatMapValues
- keys
- values
- subtractByKey
- join
- rightOuterJoin
- leftOuterJoin
- cogroup
- sortByKey

Source: Jspark: A brain-friendly introduction

Spark:Actions

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

action	*description*
reduce(*func*)	aggregate the elements of the dataset using a function *func* (which takes two arguments and returns one), and should also be commutative and associative so that it can be computed correctly in parallel
collect()	return all the elements of the dataset as an array at the driver program – usually useful after a filter or other operation that returns a sufficiently small subset of the data
count()	return the number of elements in the dataset
first()	return the first element of the dataset – similar to *take(1)*
take(*n*)	return an array with the first *n* elements of the dataset – currently not executed in parallel, instead the driver program computes all the elements
takeSample(*withReplacement, fraction, seed*)	return an array with a random sample of *num* elements of the dataset, with or without replacement, using the given random number generator seed

Spark:Actions

action	*description*
`saveAsTextFile(`*`path`***`)`**	write the elements of the dataset as a text file (or set of text files) in a given directory in the local filesystem, HDFS or any other Hadoop-supported file system. Spark will call `toString` on each element to convert it to a line of text in the file
`saveAsSequenceFile(`*`path`***`)`**	write the elements of the dataset as a Hadoop `SequenceFile` in a given path in the local filesystem, HDFS or any other Hadoop-supported file system. Only available on RDDs of key-value pairs that either implement Hadoop's `Writable` interface or are implicitly convertible to `Writable` (Spark includes conversions for basic types like `Int`, `Double`, `String`, etc).
`countByKey()`	only available on RDDs of type `(K, V)`. Returns a `Map` of `(K, Int)` pairs with the count of each key
`foreach(`*`func`***`)`**	run a function *func* on each element of the dataset – usually done for side effects such as updating an accumulator variable or interacting with external storage systems

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

Transformations

```
>>> nums = sc.parallelize([1,2,3])
>>> squared = nums.map(lambda x : x*x)
>>> even = squared.filter(lambda x: x%2 == 0)
>>> evens = nums.flatMap(lambda x: range(x))
```

Actions

```
>>> nums = sc.parallelize([1,2,3])
>>> nums.collect()
>>> nums.take(2)
>>> nums.count()
>>> nums.reduce(lambda:x, y:x+y)
>>> nums.saveAsTextFile("hdfs:///user/cloudera/output/test”)
```

Spark Programming

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

Functional tools in Python

map

filter

reduce

lambda

IterTools

Chain, flatmap

map

```
>>> a= [1,2,3]
>>> def add1(x) : return x+1
>>> map(add1, a)
```

Result: [2,3,4]

```
val input = sc.parallelize(List(1,2,3,4))
val result = input.map(x => x*x)
println(result.collect().mkString(","))
```

Result: [1,4,9,16]

Filter

```
>>> a= [1,2,3,4]
>>> def isOdd(x) : return x%2==1
>>> filter(isOdd, a)
```

Result: [1,3]

Reduce

```
>>> a= [1,2,3,4]
>>> def add(x,y) : return x+y
>>> reduce(add, a)
```

Result: 10

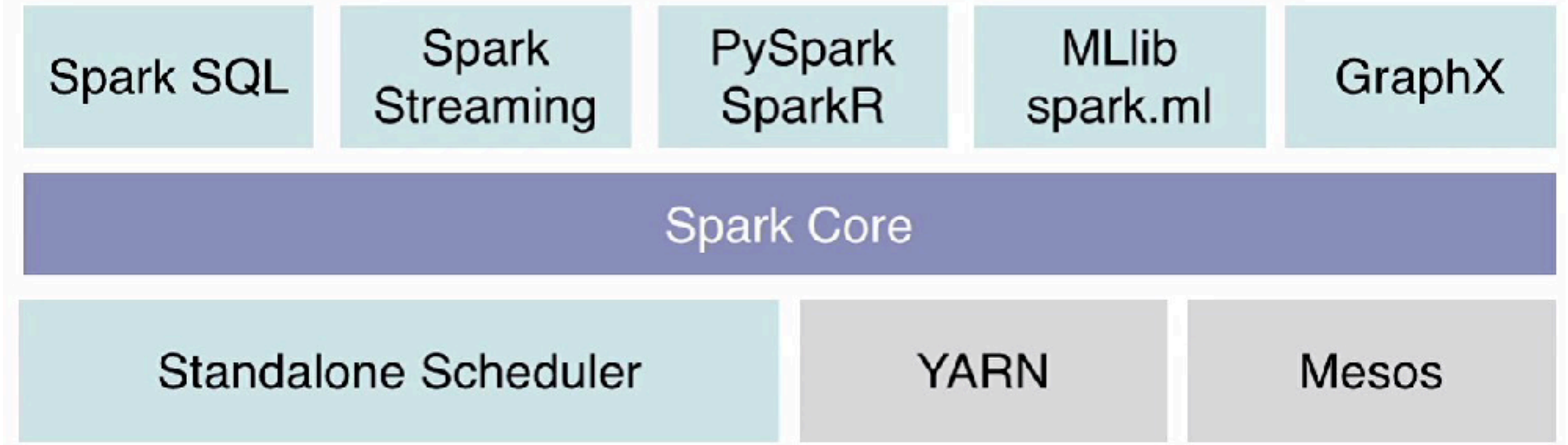
lambda

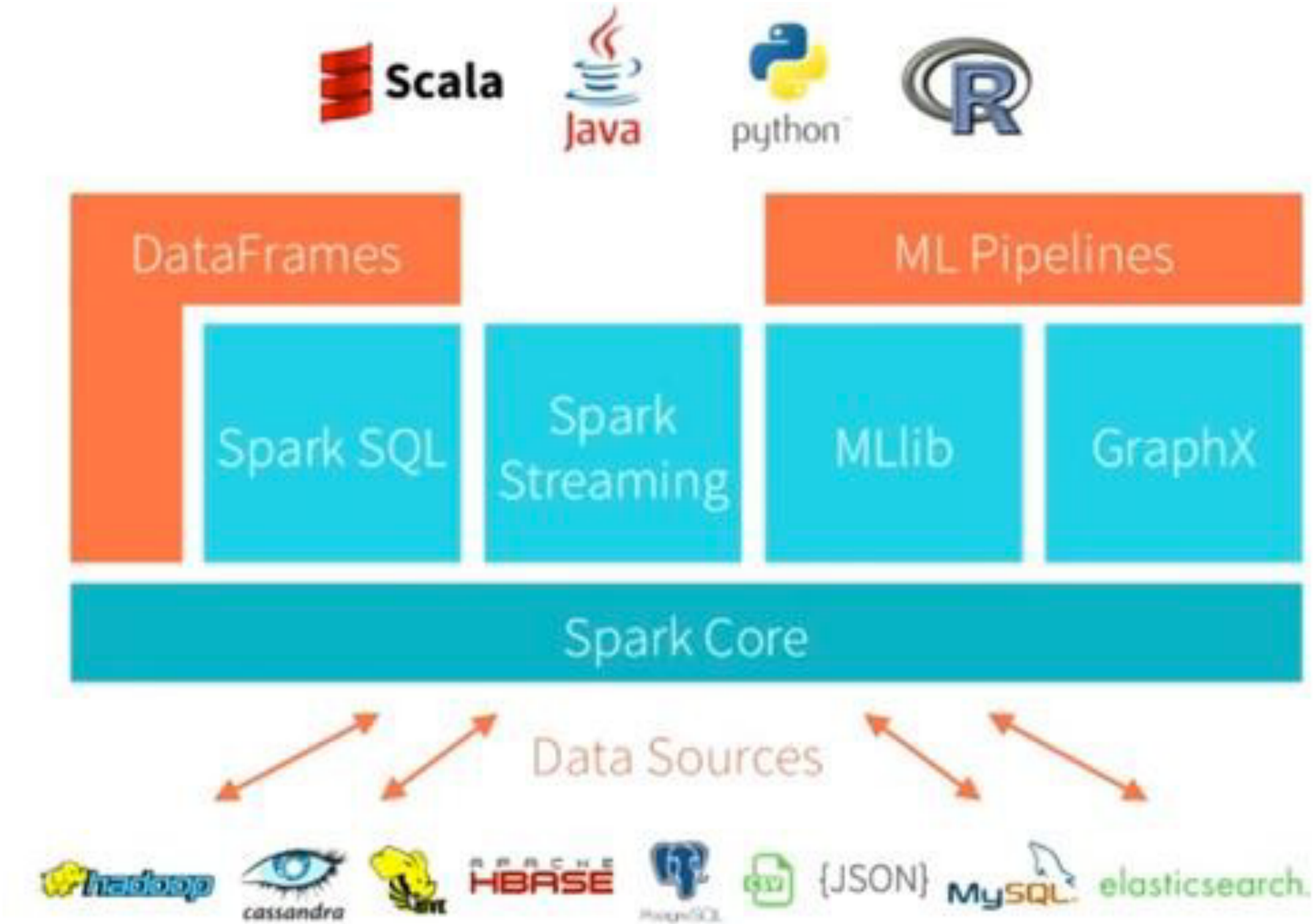
```
>>> (lambda x: x + 1)(3)
Result: 4
>>> map((lambda x: x + 1), [1,2,3])
Result: [2,3,4]
```

Exercises

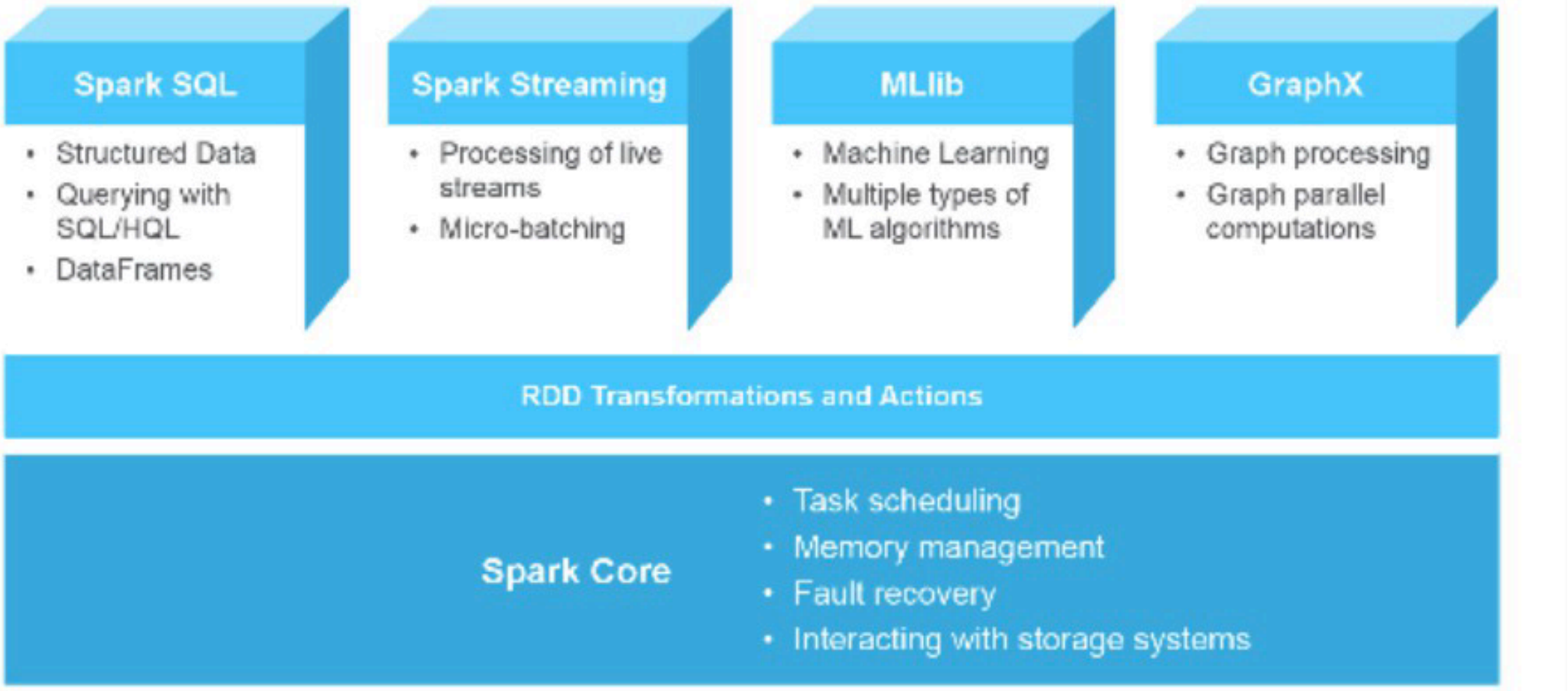
- (lambda x: 2*x)(3) => ?
- map(lambda x: 2*x, [1,2,3]) =>
- map(lambda t: t[0], [(1,2), (3,4), (5,6)]) =>
- reduce(lambda x,y: x+y, [1,2,3]) =>
- reduce(lambda x,y: x+y, map(lambda t: t[0], [(1,2), (3,4), (5,6)]))=>

Spark Platform

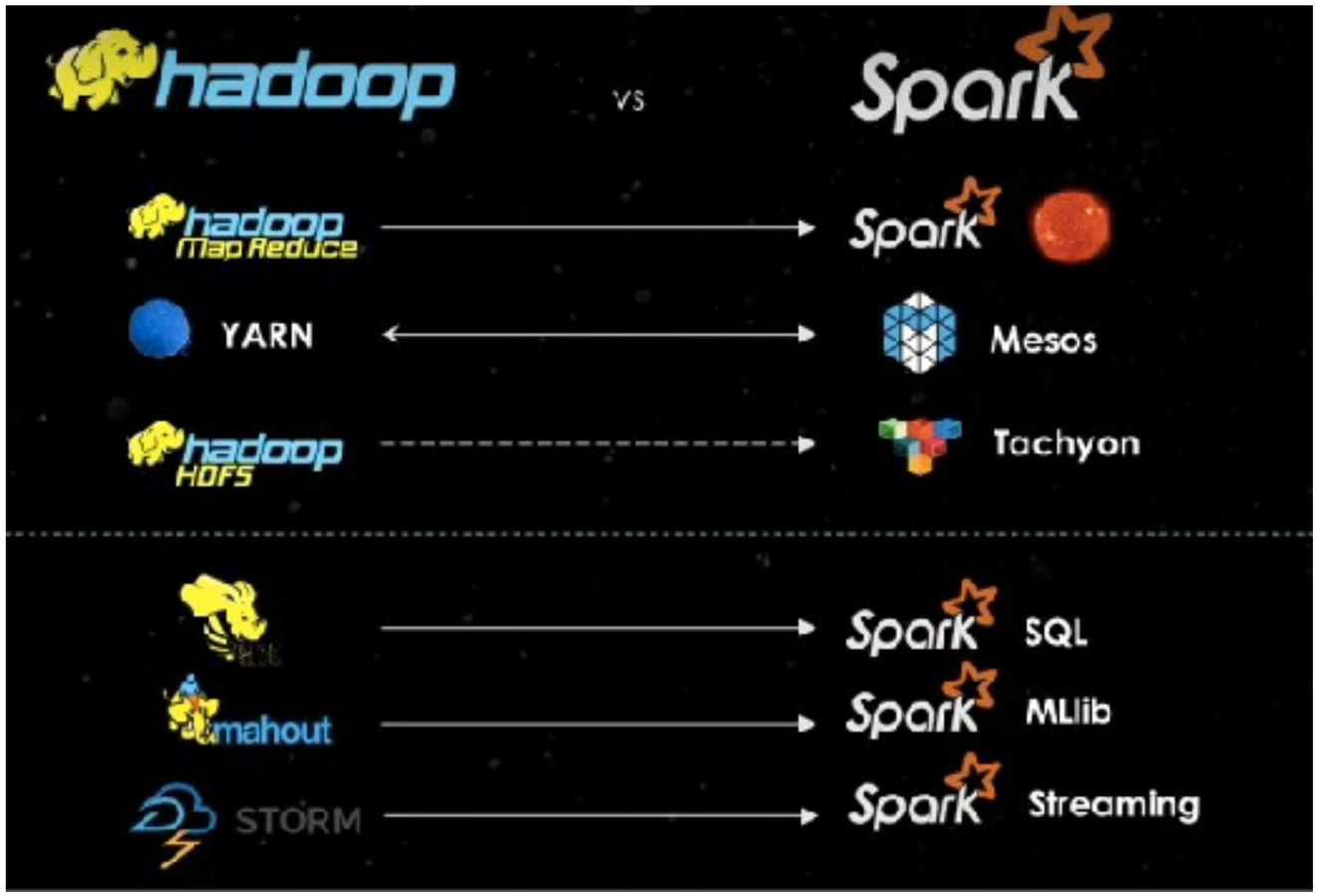


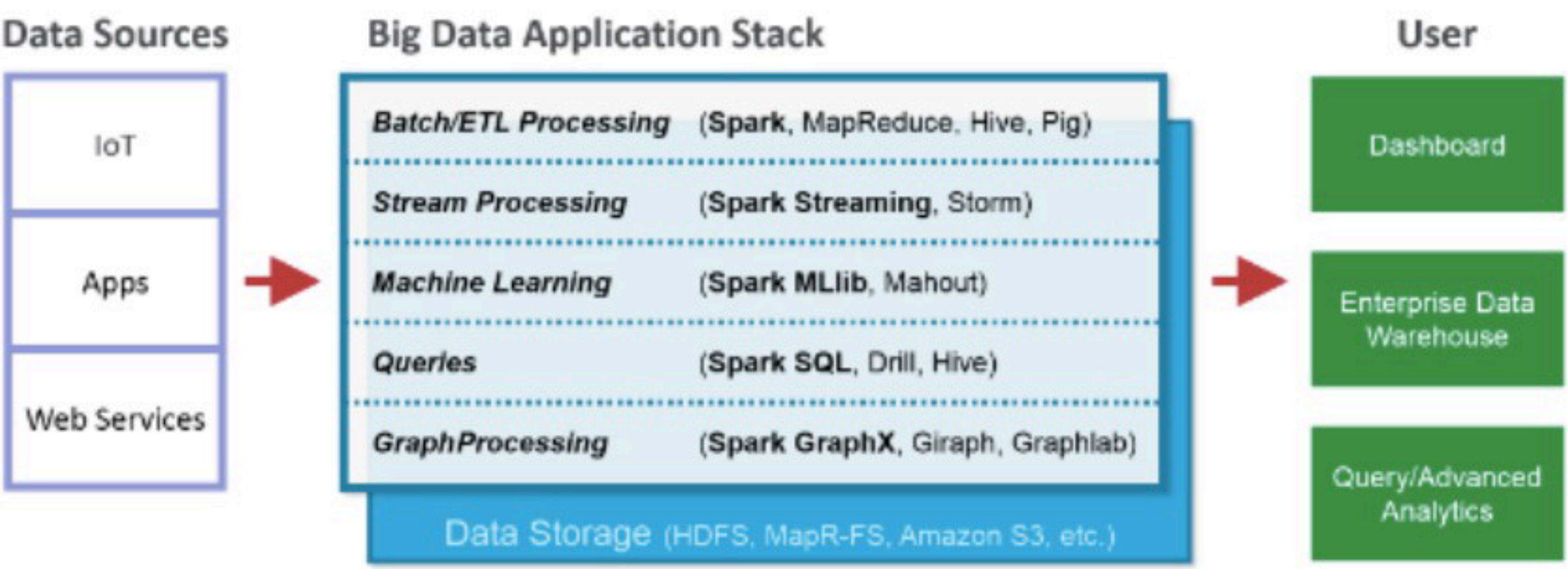

Scala
Java
python
R
DataFrames
ML Pipelines
Spark SQL
Spark Streaming
MLlib
GraphX
Spark Core
Data Sources
hadoop
cassandra
HBASE
{JSON}
MySQL
elasticsearch.

Spark Platform



Source: MapR Academy





Source: MapR Academy

Do we still need Hadoop?

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

Yes, why Hadoop?

- HDFS
- YARN
- MapReduce is mature and still be appropriate for certain workloads
- Other services: Sqoop, Flume, etc.

But you can still use other resource management, storages

- Spark Standalone
- Amazon S3
- Mesos

Start Spark-shell

$spark-shell

```
[root@quickstart 201402_babs_open_data]# spark-shell
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/lib/zookeeper/lib/slf4j-log4j12-1.7.5.jar!/or
g/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/jars/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/
StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel).
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 1.6.0
      /_/

Using Scala version 2.10.5 (Java HotSpot(TM) 64-Bit Server VM, Java 1.7.0_67)
Type in expressions to have them evaluated.
Type :help for more information.
```

Testing SparkContext

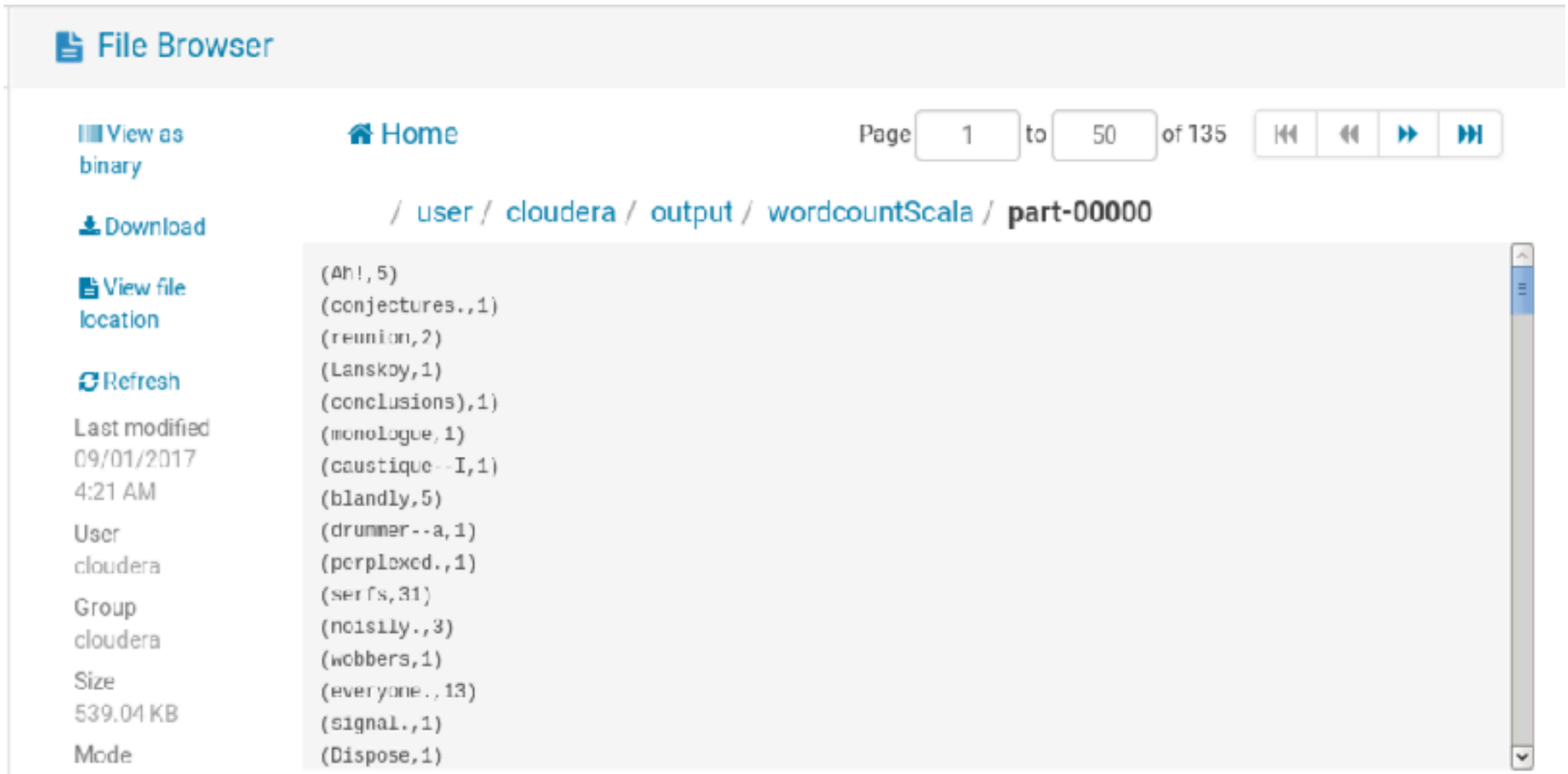
scala> sc

```
scala> sc
res0: org.apache.spark.SparkContext = org.apache.spark.SparkContext@18c07e25
```

Spark Program in Scala: WordCount

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

```
scala> val file = sc.textFile("hdfs:///user/cloudera/input/PG2600.txt")
scala> val wc = file.flatMap(l => l.split(" ")).map(word =>(word,
1)).reduceByKey(_ + _)
scala> wc.saveAsTextFile("hdfs:///user/cloudera/output/wordcountScala")
```

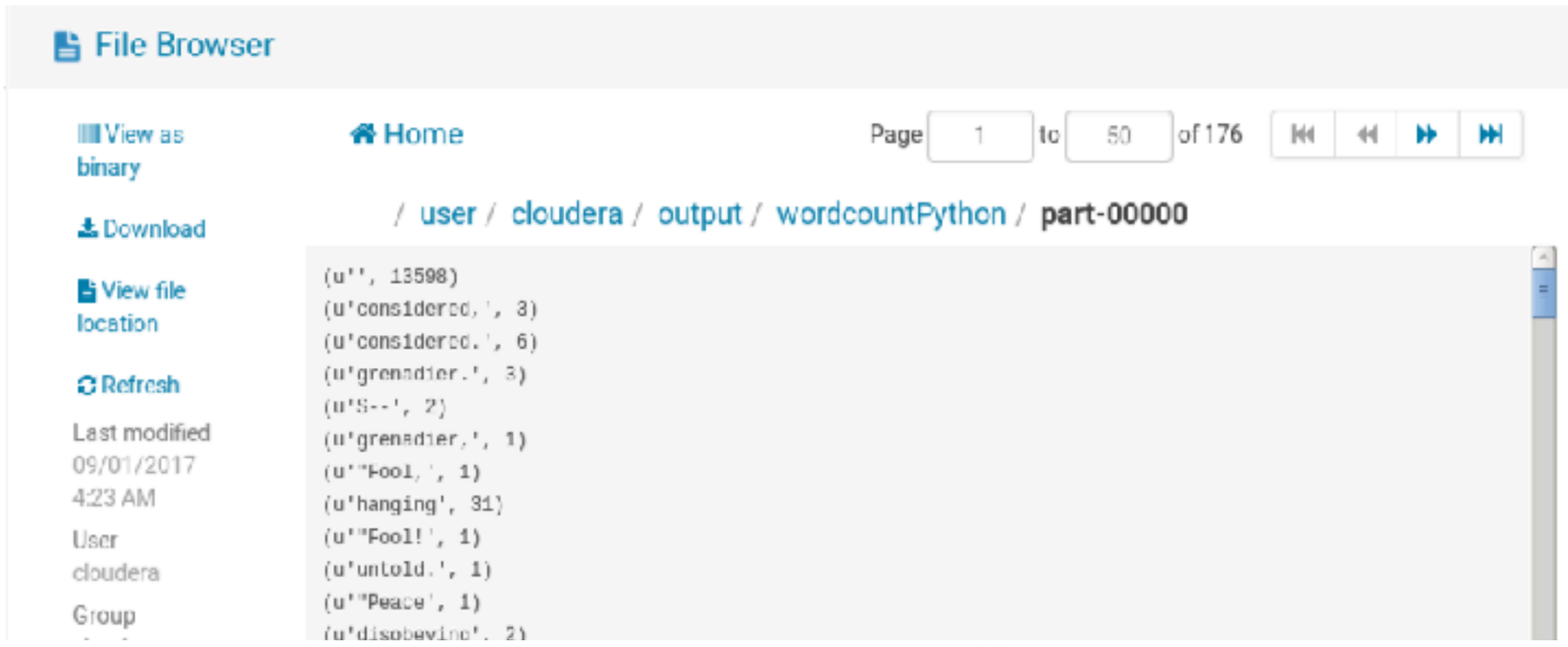


Spark Program in Python: WordCount

CHULALONGKORN BUSINESS SCHOOL

FLAGSHIP FOR LIFE

```
$ pyspark
>>> from operator import add
>>> file =sc.textFile("hdfs:///user/cloudera/input/PG2600.txt")
>>> wc = file.flatMap(lambda x: x.split(' ')).map(lambda x:(x,
1)).reduceByKey(add)
>>> wc.saveAsTextFile("hdfs:///user/cloudera/output/
wordcountPython")
```



Loading data from MySQL

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

Download MySQL driver & Start Spark-shell

Open New Terminal

$ mkdir spark

$ cd spark

$wget https://github.com/bobbylovemovie/trainbigdata/raw/master/Spark/mysql-connector-java-5.1.23.jar

Running Spark-shell

$ spark-shell --jars mysql-connector-java-5.1.23.jar

```
16/06/28 15:23:35 WARN shortcircuit.DomainSocketFactory: The short-circuit local rea
ds feature cannot be used because libhadoop cannot be loaded.
SQL context available as sqlContext.

scala>
```

```
$ scala> :paste
val url="jdbc:mysql://localhost:3306/test_mysql_db"
val username = "root"
val password = "cloudera"
import org.apache.spark.rdd.JdbcRDD
import java.sql.{Connection, DriverManager, ResultSet}
Class.forName("com.mysql.jdbc.Driver").newInstance
val myRDD = new JdbcRDD( sc, () =>
DriverManager.getConnection(url,username,password) ,
"SELECT * FROM country_tbl LIMIT ?, ?", 0, 5, 2, r =>r.getString("id") + ", " +
r.getString("country"))
myRDD.count
myRDD.foreach(println)
```

Output


```
// Exiting paste mode, now interpreting.

1, USA
2, CANADA
4, Brazil
61, Japan
65, Singapore
66, Thailand
url: String = jdbc:mysql://localhost:3306/test_mysql_db
username: String = root
password: String = cloudera
import org.apache.spark.rdd.JdbcRDD
import java.sql.{Connection, DriverManager, ResultSet}
myRDD: org.apache.spark.rdd.JdbcRDD[String] = JdbcRDD[0] at JdbcRDD at <console>
:39
```

scala> myRDD.saveAsTextFile("hdfs:///user/cloudera/output/mysqlFromSpark")

Spark SQL

Spark SQL

DataFrame

A distributed collection of rows organied into named columns.

An abstraction for selecting, filtering, aggregating, and plotting structured data.

Previously => SchemaRDD

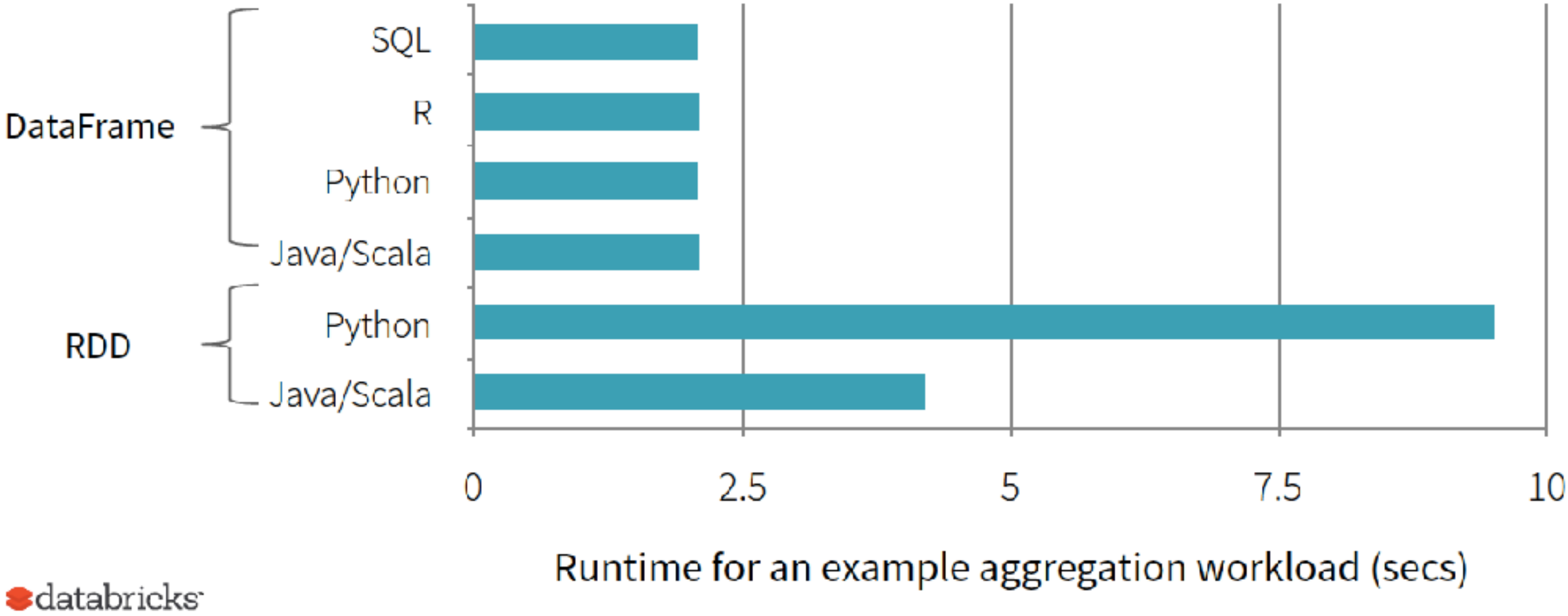
Creating and running Spark program faster

– Write less code

– Read less data

– Let the optimizer do the hard work

Benefit of Logical Plan: Performance Parity Across Languages



databricks

Source: Jump start into Apache Spark and Databricks

SparkSQL can leverage the Hive metastore

CHULALONGKORN BUSINESS SCHOOL

FLAGSHIP FOR LIFE

Hive Metastore can also be leveraged by a wide array of applications

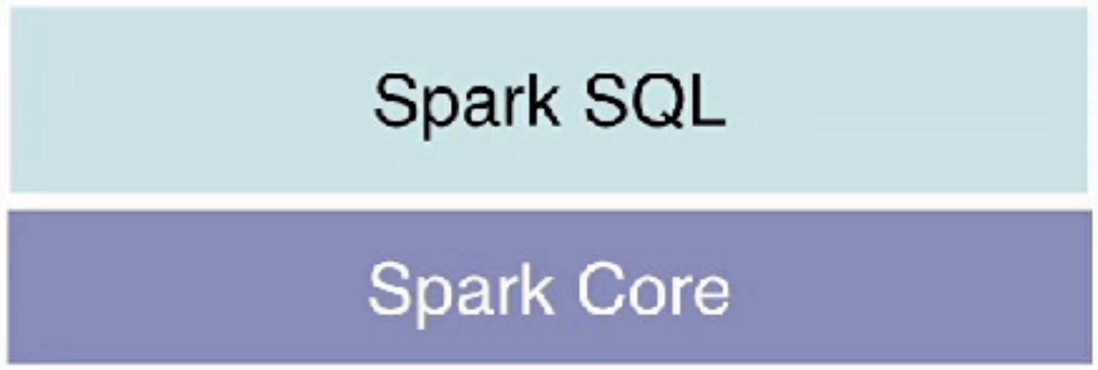
– Spark

– Hive

– Impala

Available from HiveContext

```
context = ps.HiveContext(sc)

# query with SQL
results = context.sql(
  "SELECT * FROM people")

# apply Python transformation
names = results.map(lambda p: p.name)
```

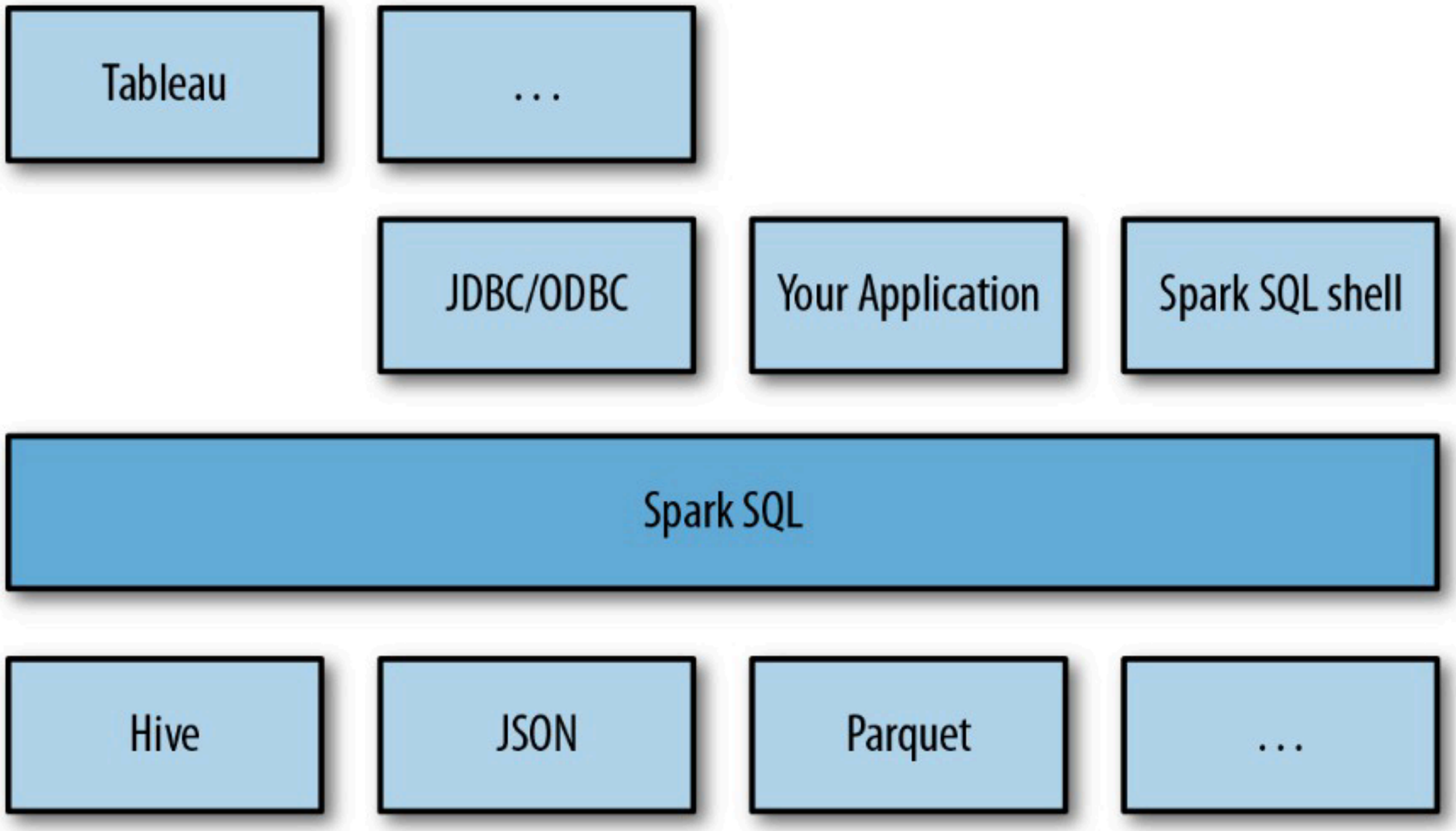


Unified interface for structured data

Image credit: http://barrymieny.deviantart.com/

amplab UC BERKELEY databricks

Spark SQL usage



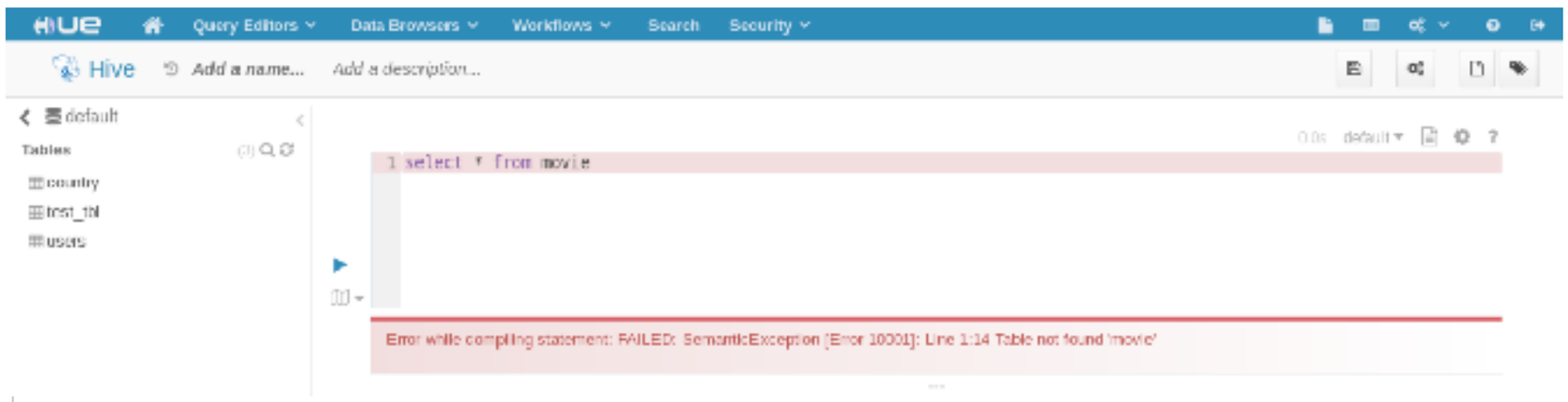
Source: Learning Spark, O'Reilly Media, Inc.

Link Hive Metastore with Spark-Shell

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

```
$ spark-shell --jars mysql-connector-java-5.1.23.jar
scala > val sqlContext = new org.apache.spark.sql.hive.HiveContext(sc)
scala> sqlContext.sql("CREATE TABLE IF NOT EXISTS movie(userid
STRING, movieid STRING, rating INT, timestamp STRING) ROW FORMAT
DELIMITED FIELDS TERMINATED BY '\t' LINES TERMINATED BY '\n'")
scala> sqlContext.sql("LOAD DATA LOCAL INPATH '/home/cloudera/
movielens_dataset/ml-100k/u.data' INTO TABLE movie")
scala> val result = sqlContext.sql("SELECT * FROM movie")
scala> result.show()
```

```
scala> result.show()
+------+-------+------+---------+
|userid|movieid|rating|timestamp|
+------+-------+------+---------+
|   196|    242|     3|881250949|
|   186|    302|     3|891717742|
|    22|    377|     1|878887116|
|   244|     51|     2|880606923|
|   166|    346|     1|886397596|
|   298|    474|     4|884182806|
```



Link Hive Metastore with Spark-Shell

CHULALONGKORN BUSINESS SCHOOL

FLAGSHIP FOR LIFE

Copy the configuration file

$sudo cp /usr/lib/hive/conf/hive-site.xml /usr/lib/spark/conf/

$ spark-shell

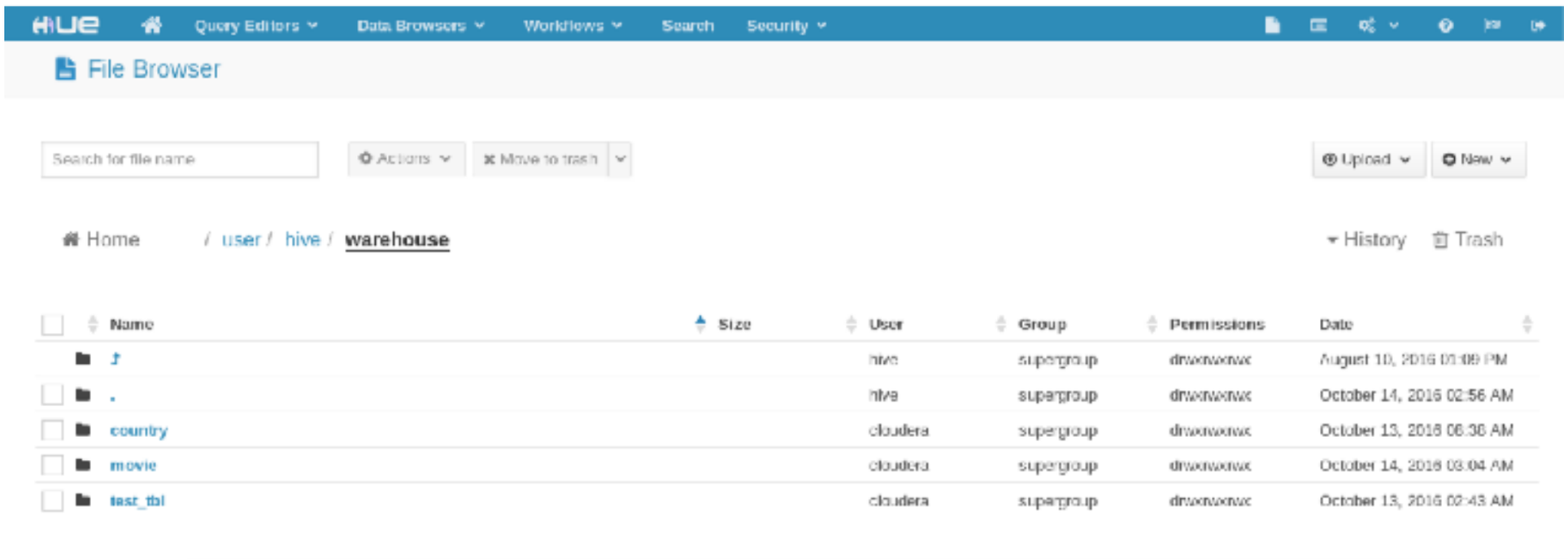
scala > val sqlContext = new org.apache.spark.sql.hive.HiveContext(sc)

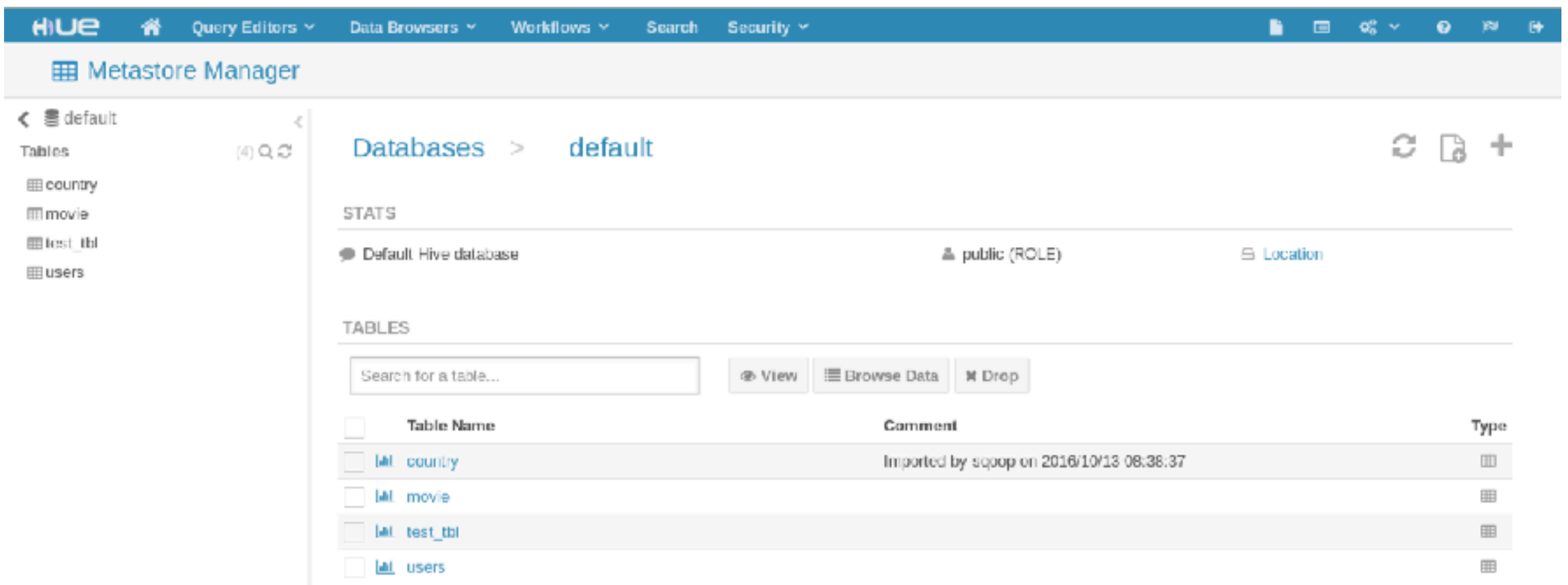
scala> sqlContext.sql("CREATE TABLE IF NOT EXISTS movie(userid STRING, movieid STRING, rating INT, timestamp STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t' LINES TERMINATED BY '\n'")

scala> sqlContext.sql("LOAD DATA LOCAL INPATH '/home/cloudera/movielens_dataset/ml-100k/u.data' INTO TABLE movie")

scala> val result = sqlContext.sql("SELECT * FROM movie")

scala> result.show()


HUE
Query Editors
Data Browsers
Workflows
Search
Security
File Browser
Search for file name
Actions
Move to trash
Upload
New
Home / user / hive / warehouse
History
Trash
Name
Size
User
Group
Permissions
Date
.Trash
hive
supergroup
drwxrwxrwx
August 10, 2016 01:09 PM
.
hive
supergroup
drwxrwxrwx
October 14, 2016 02:56 AM
country
cloudera
supergroup
drwxrwxrwx
October 13, 2016 06:36 AM
movie
cloudera
supergroup
drwxrwxrwx
October 14, 2016 03:04 AM
test_tbl
cloudera
supergroup
drwxrwxrwx
October 13, 2016 02:43 AM


HUE
Query Editors
Data Browsers
Workflows
Search
Security
Metastore Manager
default
Tables (4)
country
movie
test_tbl
users
Databases > default
STATS
Default Hive database
public (ROLE)
Location
TABLES
Search for a table...
View
Browse Data
Drop
Table Name
Comment
Type
country
Imported by sqoop on 2016/10/13 08:38:37
movie
test_tbl
users

Spark SQL Meals Data

Upload a data to HDFS

$ wget https://github.com/bobbylovemovie/trainbigdata/raw/master/Spark/events.txt

$ wget https://github.com/bobbylovemovie/trainbigdata/raw/master/Spark/meals.txt

$ hadoop fs -put events.txt /user/cloudera/input

$ hadoop fs -put meals.txt /user/cloudera/input

Spark SQL : Preparing data


```
$ pyspark
>>> meals_rdd =sc.textFile("hdfs:///user/cloudera/input/meals.txt")
>>> events_rdd =sc.textFile("hdfs:///user/cloudera/input/events.txt")
>>> header_meals = meals_rdd.first()
>>> header_events = events_rdd.first()
>>> meals_no_header = meals_rdd.filter(lambda row:row != header_meals)
>>> events_no_header =events_rdd.filter(lambda row:row != header_events)
>>> meals_json = meals_no_header.map(lambda
row:row.split(';')).map(lambda row_list:dict(zip(header_meals.split(';'),
row_list)))
>>> events_json = events_no_header.map(lambda
row:row.split(';')).map(lambda row_list:dict(zip(header_events.split(';'),
row_list)))
```

```
>>> import json
>>> def type_conversion(d, columns) :
...     for c in columns:
...         d[c] = int(d[c])
...     return d
...
>>> meal_typed = meals_json.map(lambda
j:json.dumps(type_conversion(j, ['meal_id','price'])))
>>> event_typed = events_json.map(lambda
j:json.dumps(type_conversion(j, ['meal_id','userid'])))
```

Spark SQL : Create DataFrame

```
>>> meals_dataframe = sqlContext.jsonRDD(meal_typed)
>>> events_dataframe = sqlContext.jsonRDD(event_typed)
>>> meals_dataframe.head()
```

```
Row(dt=u'2013-01-01', meal_id=1, price=10, type=u'french')
```

```
>>> meals_dataframe.printSchema()
```

```
root
 |-- dt: string (nullable = true)
 |-- meal_id: long (nullable = true)
 |-- price: long (nullable = true)
 |-- type: string (nullable = true)
```

Running SQL Query

```
>>> meals_dataframe.registerTempTable('meals')
>>> events_dataframe.registerTempTable('events')
>>> sqlContext.sql("SELECT * FROM meals LIMIT 5").collect()
```

```
[Row(dt=u'2013-01-01', meal_id=1, price=10, type=u'french'), Row(dt=u'2013-01-01
', meal_id=2, price=13, type=u'chinese'), Row(dt=u'2013-01-02', meal_id=3, price
=9, type=u'mexican'), Row(dt=u'2013-01-03', meal_id=4, price=9, type=u'italian')
, Row(dt=u'2013-01-03', meal_id=5, price=12, type=u'chinese')]
```

```
>>> meals_dataframe.take(5)
```

```
[Row(dt=u'2013-01-01', meal_id=1, price=10, type=u'french'), Row(dt=u'2013-01-01
', meal_id=2, price=13, type=u'chinese'), Row(dt=u'2013-01-02', meal_id=3, price
=9, type=u'mexican'), Row(dt=u'2013-01-03', meal_id=4, price=9, type=u'italian')
, Row(dt=u'2013-01-03', meal_id=5, price=12, type=u'chinese')]
```

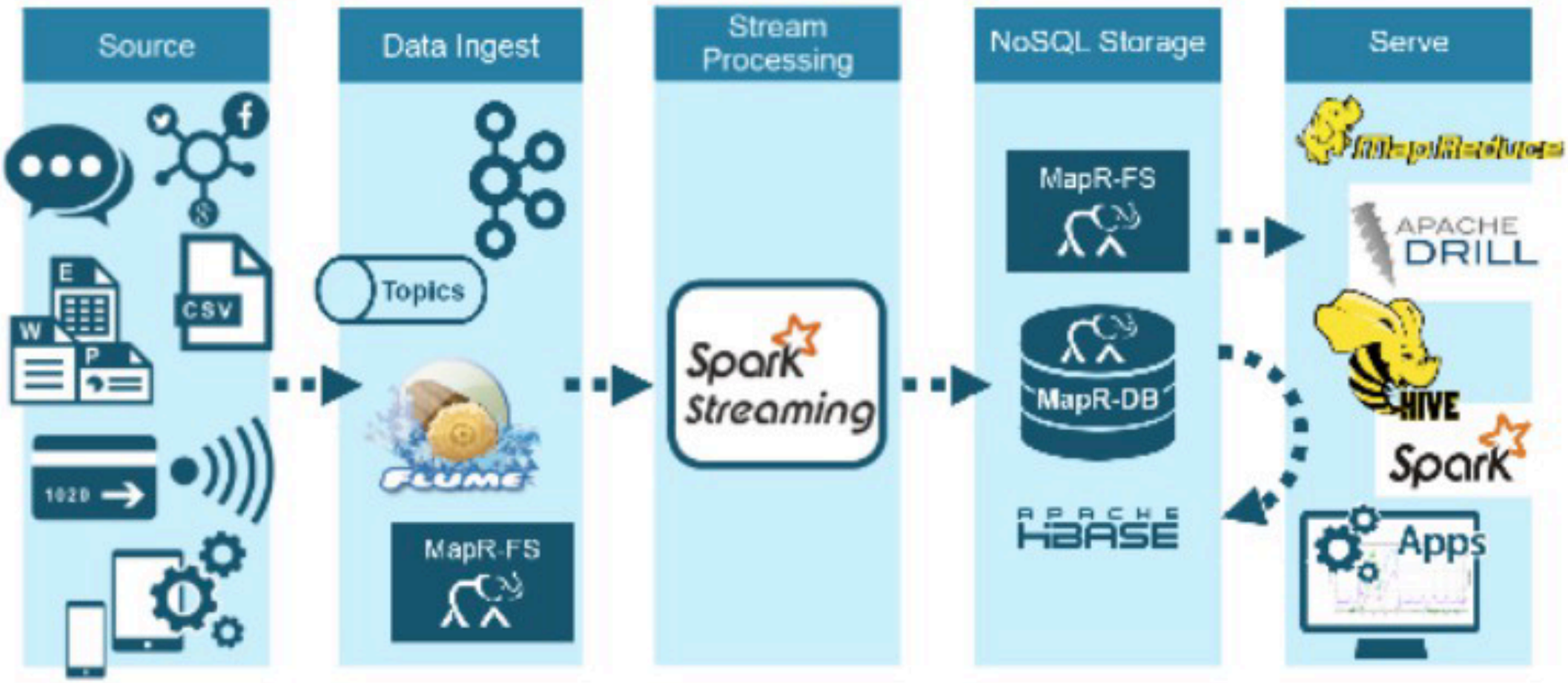
Spark SQL : More complex query

```
>>> sqlContext.sql("""
    SELECT type, COUNT(type) AS cnt FROM
    meals
    INNER JOIN
    events on meals.meal_id = events.meal_id
    WHERE
    event = 'bought'
    GROUP BY
    type
    ORDER BY cnt DESC
    """).collect()
```

```
[Row(type=u'italian', cnt=22575), Row(type=u'french', cnt=16179), Row(type=u'mex
ican', cnt=8792), Row(type=u'japanese', cnt=6921), Row(type=u'chinese', cnt=6267
), Row(type=u'vietnamese', cnt=3535)]
```

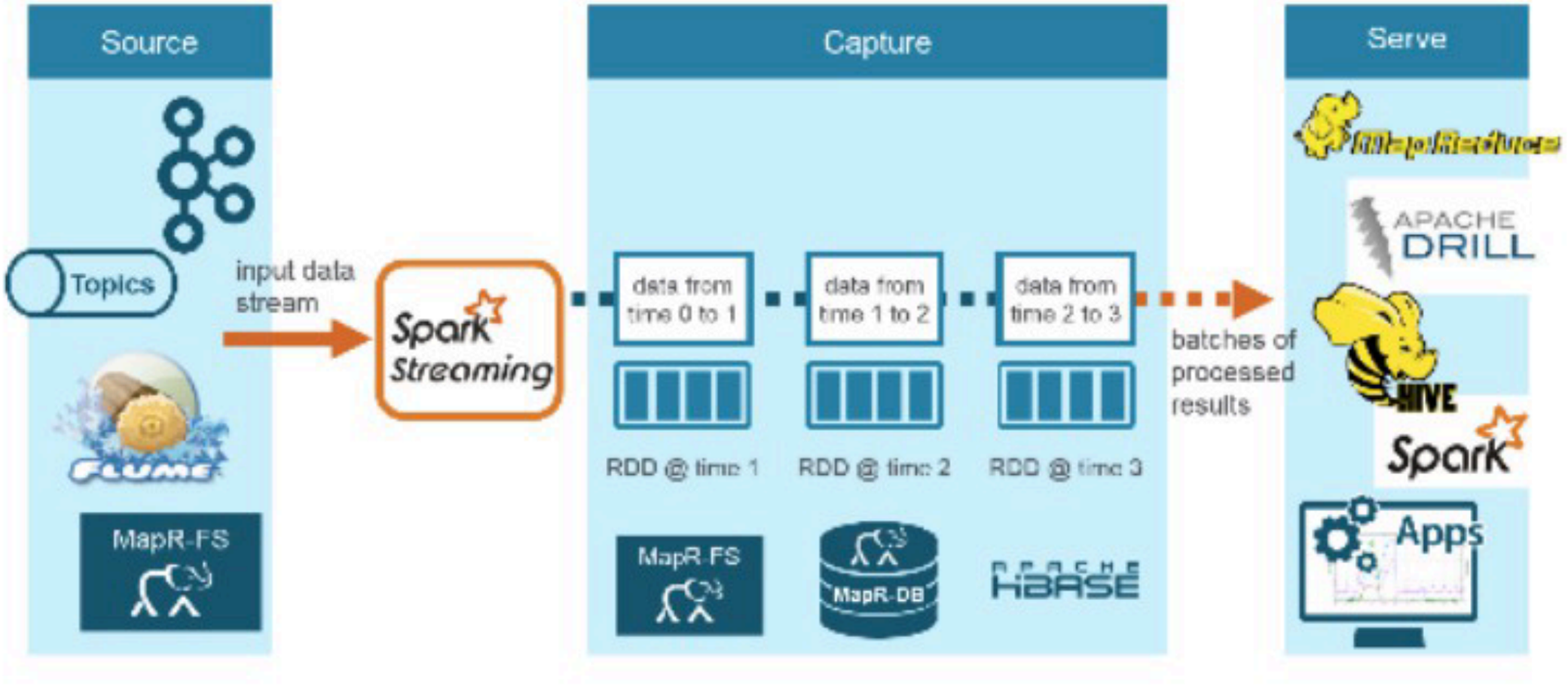
Spark Streaming

Stream Process Architecture



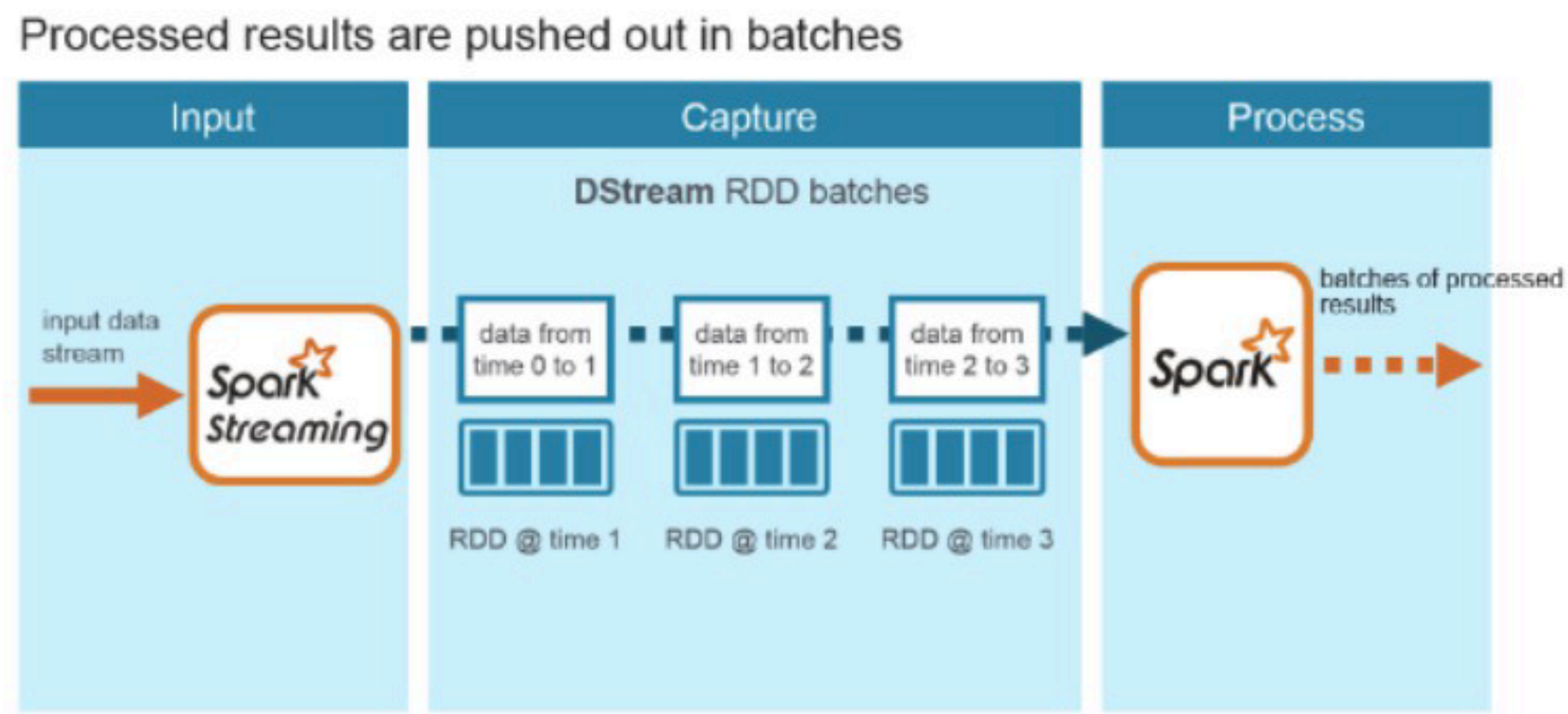
Source: MapR Academy

Spark Streaming Architecture



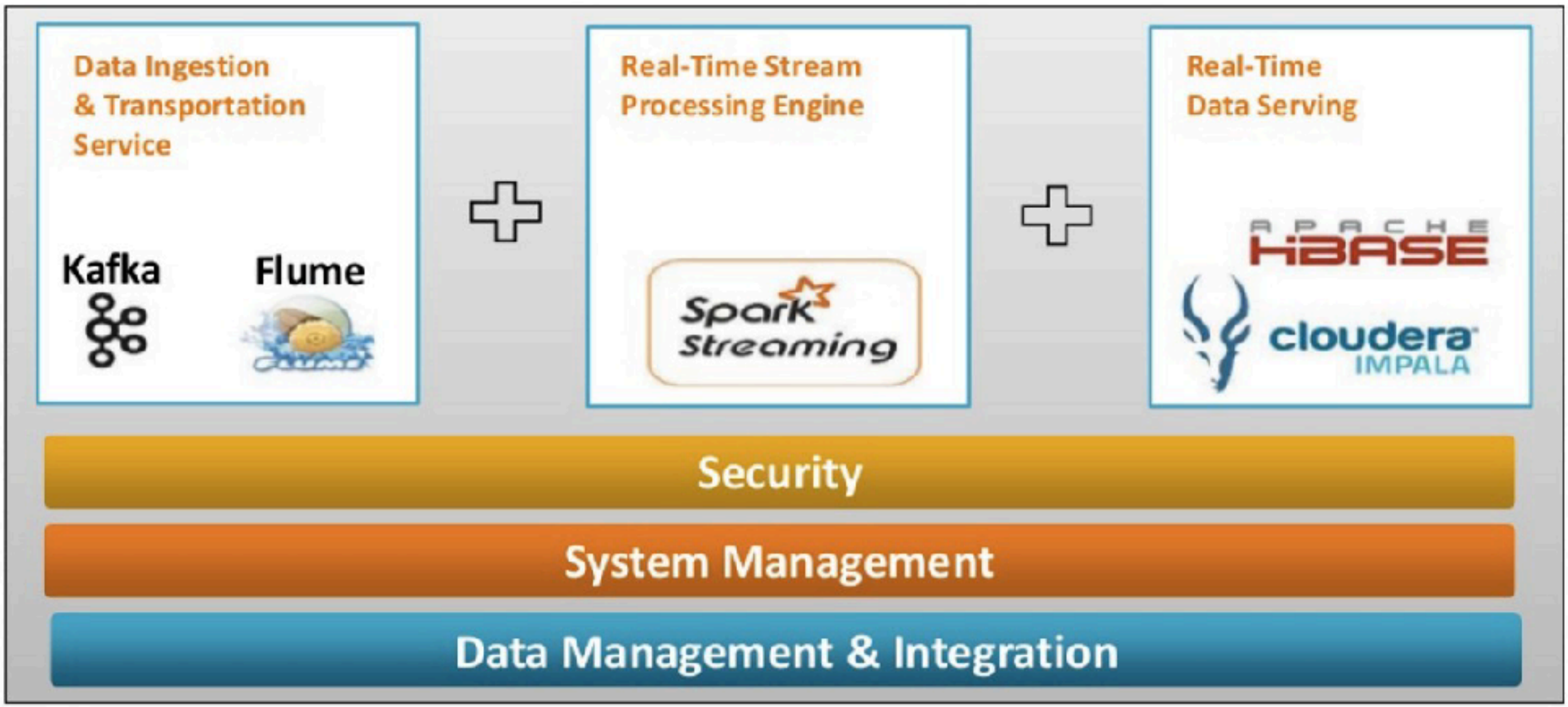
Source: MapR Academy

Processing Spark DStreams



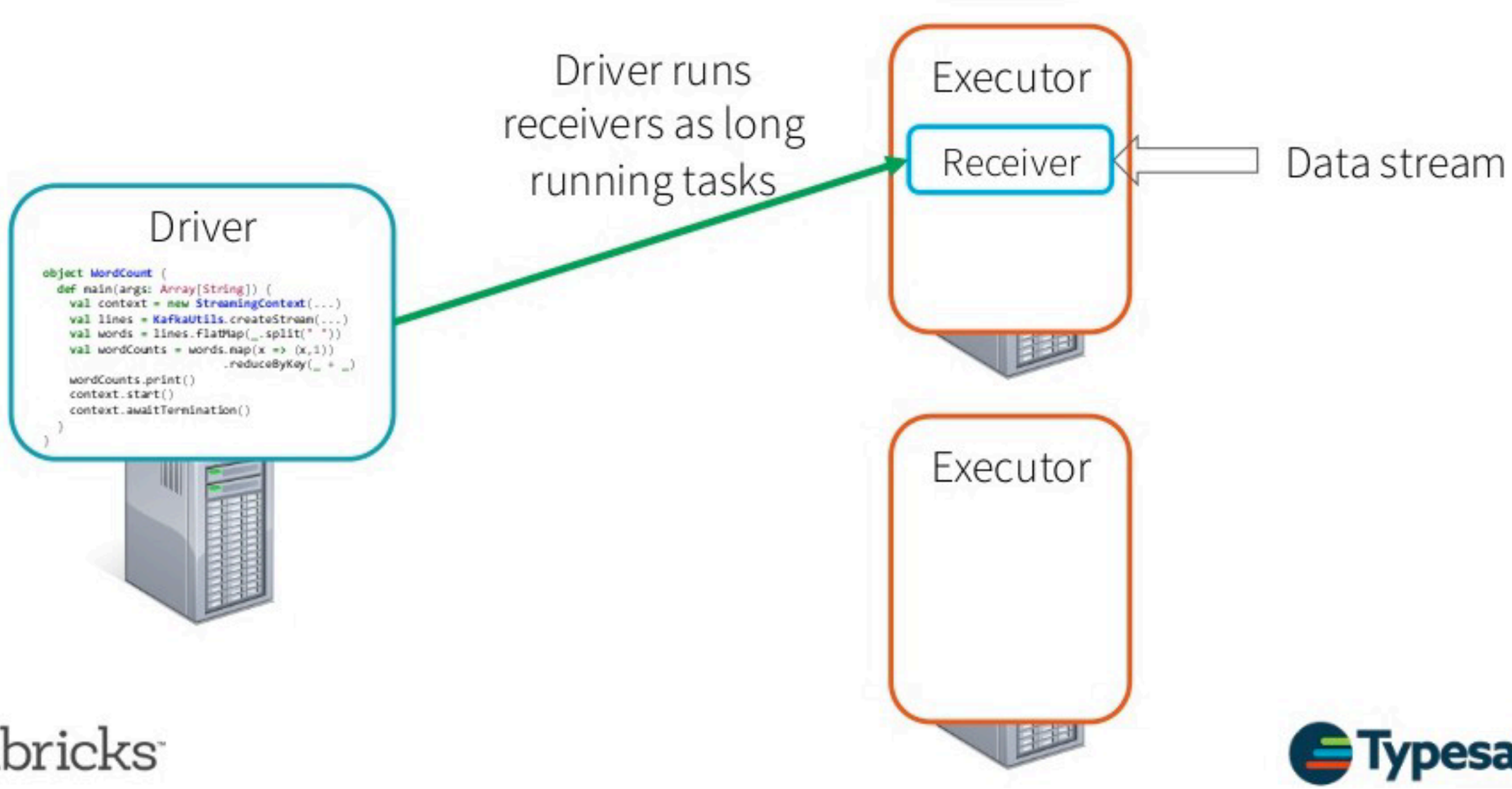
Source: MapR Academy

Streaming Architecture



Source: Cloudera

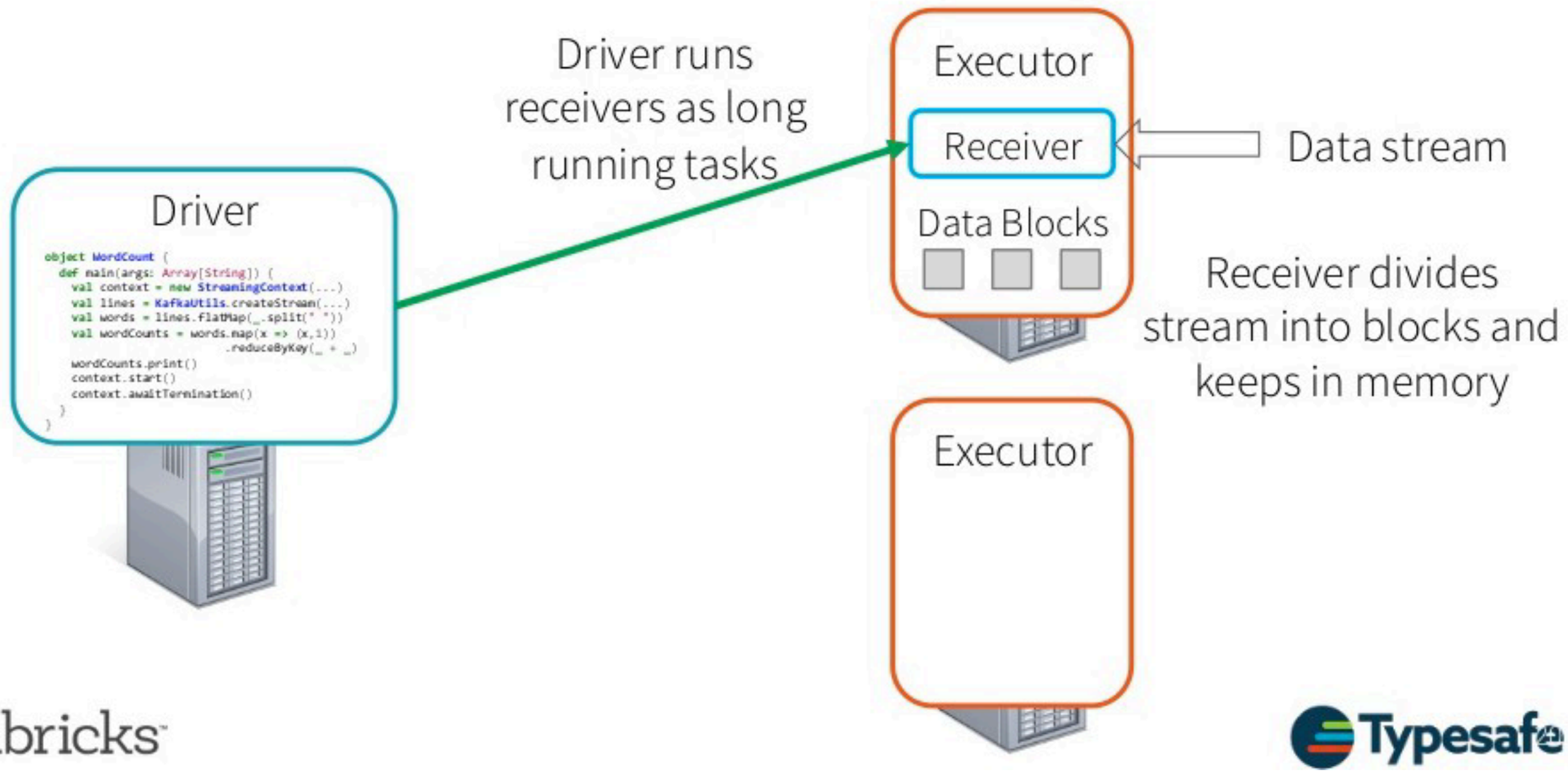
Spark Streaming Application: Receive data



databricks

Typesafe

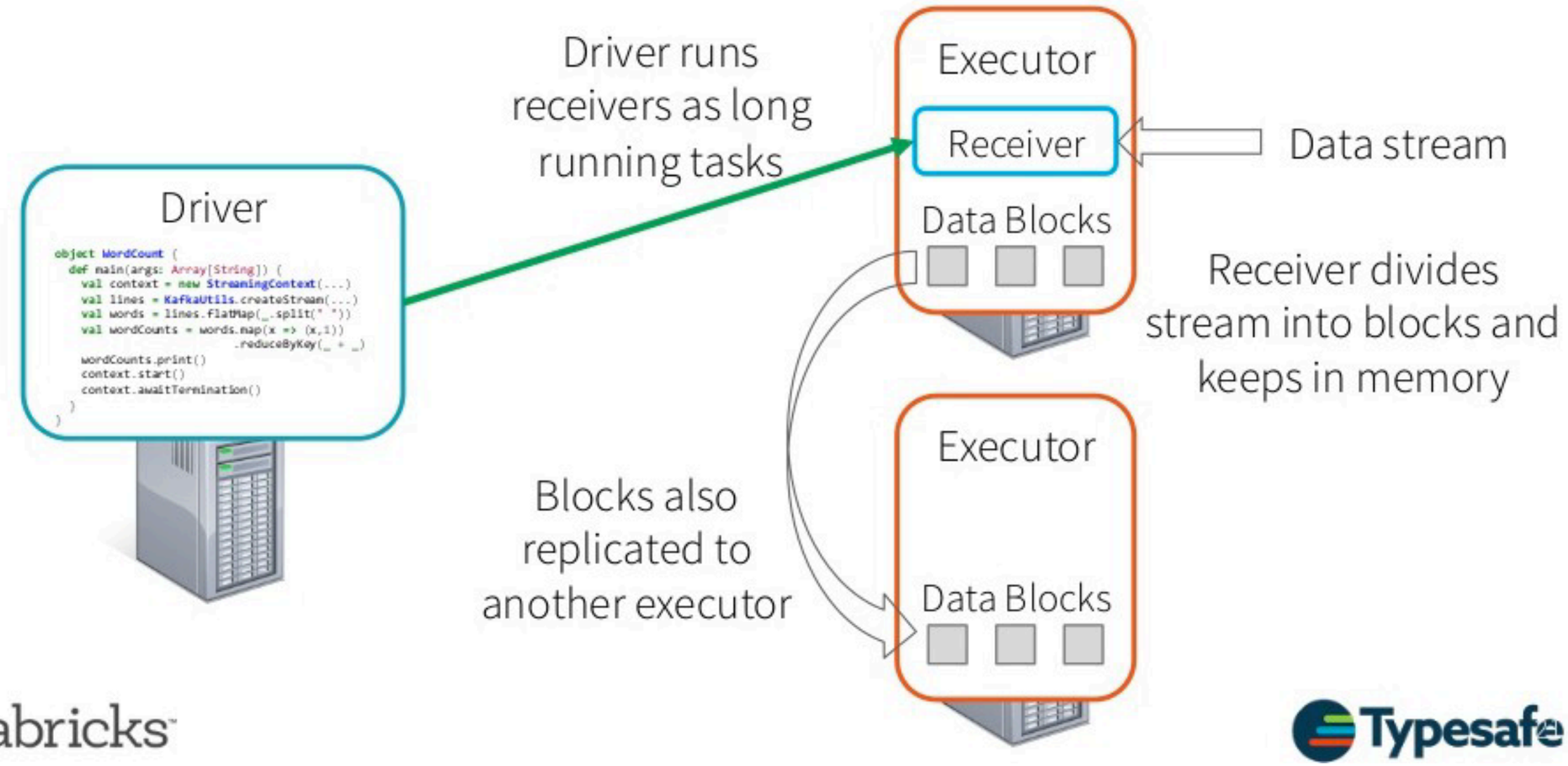
Spark Streaming Application: Receive data



databricks

Typesafe

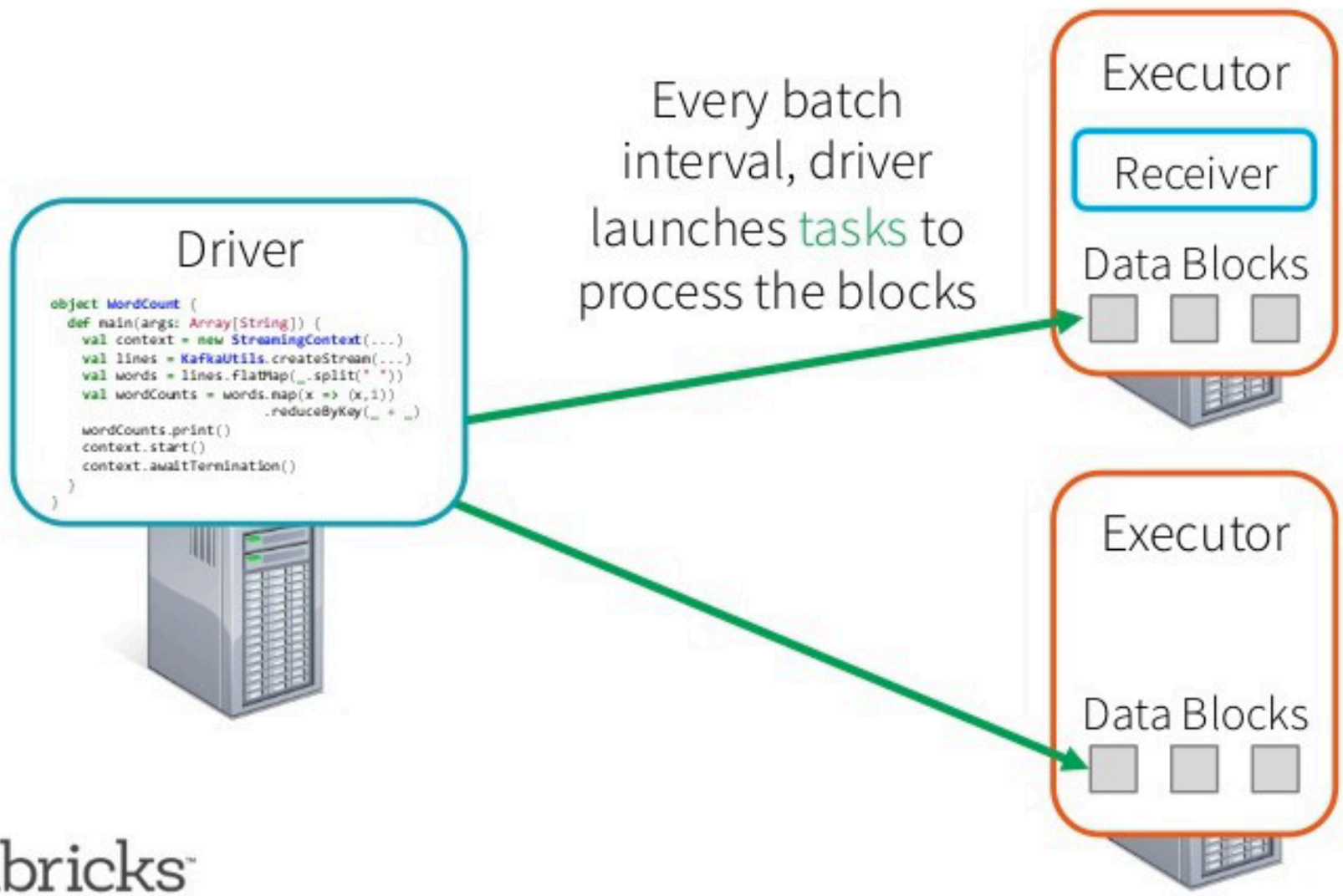
Spark Streaming Application: Receive data



databricks

Typesafe

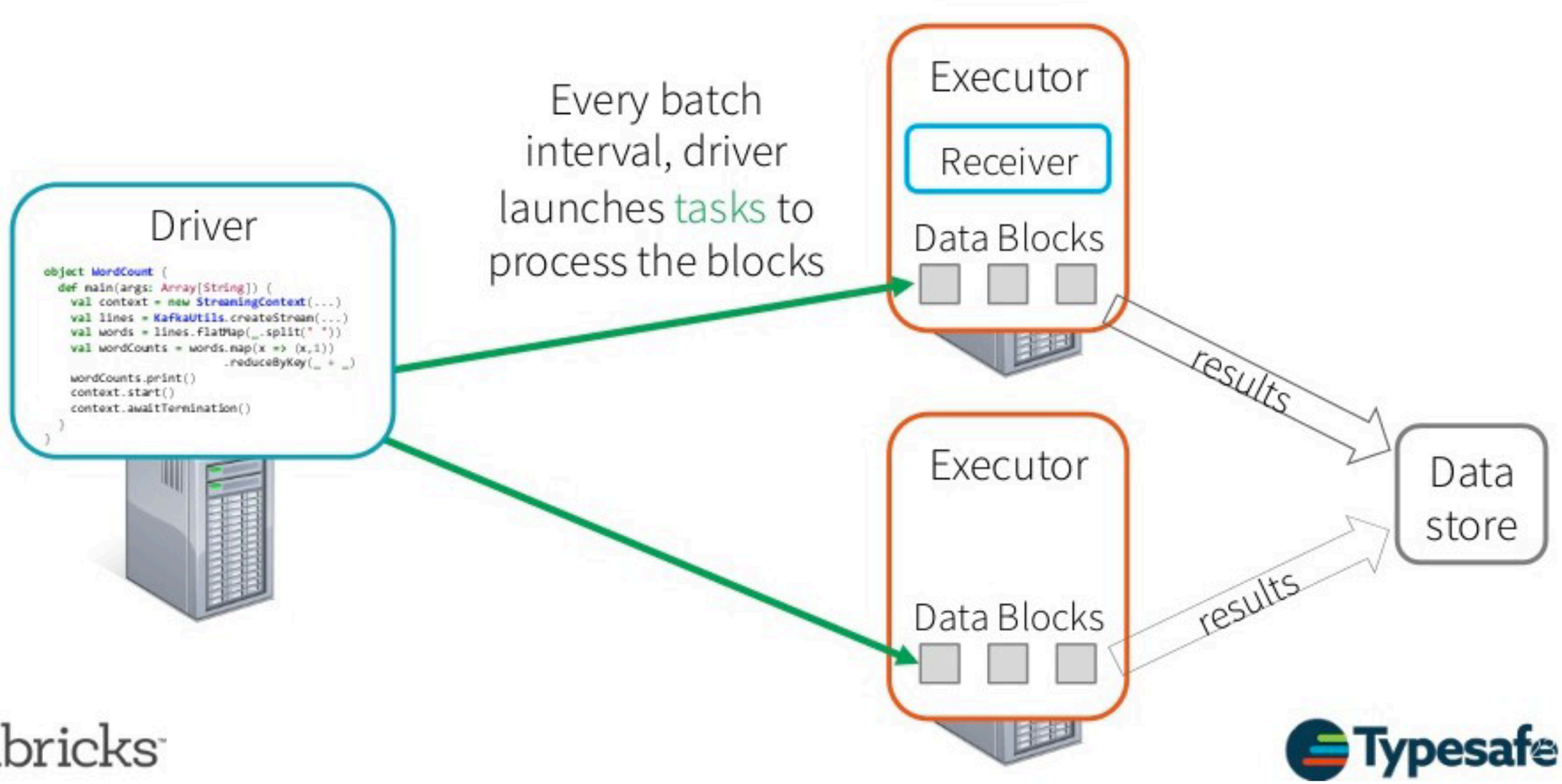
Spark Streaming Application: Process data



databricks

Typesafe

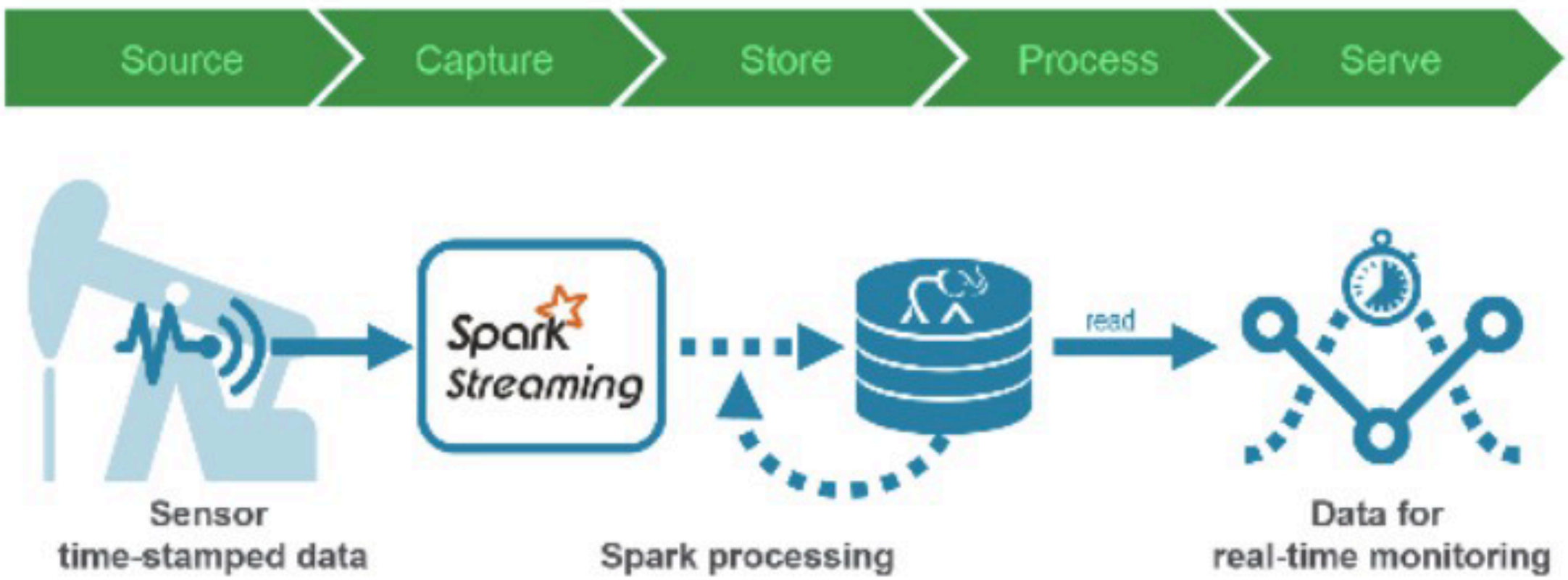
Spark Streaming Application: Process data



databricks

Typesafe

Use Case: Time Series Data



Source: MapR Academy

DStream Functional operations

flatMap(flatMapFunc)

filter(filterFunc)

map(mapFunc)

mapPartitions(mapPartFunc, preservePartitioning)

forEachRDD(forEachFunc

DStream Output operations

print()

saveAsHadoopFiles(...)

saveAsTextFiles(...)

saveAsObjectFiles(...)

saveAsNewAPIHadoopFiles(...)

forEachRDD(..)

Spark Streaming

Get Example Code

$ cd spark

$ wget https://github.com/bobbylovemovie/trainbigdata/raw/master/Spark/StreamingWordCount.py

```
1  from pyspark import SparkContext
2  from pyspark.streaming import StreamingContext
3
4  # Create a local StreamingContext with two working thread and batch interval of 1 second
5  sc = SparkContext("local[2]", "NetworkWordCount")
6  ssc = StreamingContext(sc, 10)
7
8  lines = ssc.socketTextStream("localhost", 9999)
9  words = lines.flatMap(lambda line: line.split(" "))
10
11 # Count each word in each batch
12 pairs = words.map(lambda word: (word, 1))
13 wordCounts = pairs.reduceByKey(lambda x, y: x + y)
14
15 # Print the first ten elements of each RDD generated in this DStream to the console
16 wordCounts.pprint()
17 #wordCounts.saveAsTextFiles("hdfs:///user/cloudera/output/sparkstream/sparkstream")
18 ssc.start()             # Start the computation
19 ssc.awaitTermination()  # Wait for the computation to terminate
```

Spark Streaming

Run Python Spark

$ spark-submit StreamingWordCount.py

Running the netcat server on another window

$ nc -lk 9999

```
[cloudera@quickstart ~]$ nc -lk 9999
Hello Bigdata Training
```

Result SparkStreaming

```
-------------------------------------------
Time: 2016-10-15 08:22:30
-------------------------------------------
(u'', 1)
(u'Training', 1)
(u'Hello', 1)
(u'Bigdata', 1)
```

Streaming Twitter data

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

Create a new Twitter App Login to your Twitter @ twitter.com

Create a new Twitter App https://apps.twitter.com

Create a new Twitter App (cont.)

CHULALONGKORN BUSINESS SCHOOL

FLAGSHIP FOR LIFE

Enter all the details in the application:

Application Details

Name *

Bobby_SparkStreaming_Demo_App

Your application name. This is used to attribute the source of a tweet and in user-facing authorization screens. 32 characters max.

Description *

Streaming Twitter data Demo App

Your application description, which will be shown in user-facing authorization screens. Between 10 and 200 characters max.

Website *

http://www.it.acc.chula.ac.th

Your application's publicly accessible home page, where users can go to download, make use of, or find out more information about your application. This fully-qualified URL is used in the source attribution for tweets created by your application and will be shown in user-facing authorization screens.
(If you don't have a URL yet, just put a placeholder here but remember to change it later.)

Callback URL

Where should we return after successfully authenticating? OAuth 1.0a applications should explicitly specify their oauth_callback URL on the request token step, regardless of the value given here. To restrict your application from using callbacks, leave this field blank.

Developer Agreement

Yes, I have read and agree to the Twitter Developer Agreement.

Create a new Twitter App (cont.)

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

Your application will be created:

Bobby_SparkStreaming_Demo_App Test OAuth

Details | Settings | Keys and Access Tokens | Permissions

Streaming Twitter data Demo App

http://www.it.acc.chula.ac.th

Organization

Information about the organization or company associated with your application. This information is optional.

Organization	None
Organization website	None

Application Settings

Your application's Consumer Key and Secret are used to authenticate requests to the Twitter Platform.

Access level	Read and write (modify app permissions)
Consumer Key (API Key)	eSevkYKyO94uGtFxxHoGwXDpX (manage keys and access tokens)
Callback URL	None
Callback URL Locked	No
Sign in with Twitter	Yes
App-only authentication	https://api.twitter.com/oauth2/token
Request token URL	https://api.twitter.com/oauth/request_token

Create a new Twitter App (cont.)

CHULALONGKORN BUSINESS SCHOOL

FLAGSHIP FOR LIFE

Click on Keys and Access Tokens:

Bobby_SparkStreaming_Demo_App

Details | Settings | Keys and Access Tokens | Permissions

Application Settings

Keep the "Consumer Secret" a secret. This key should never be human-readable in your application.

Consumer Key (API Key)	eSevkYKyO94uGtFxxHoGwXDpX
Consumer Secret (API Secret)	OvjqlF8VmNScaeMGZjz9e1fIfq0TxehmwkMJ454wHCCUG0AvED
Access Level	Read and write (modify app permissions)
Owner	bobbylovemovie
Owner ID	344100790

Create a new Twitter App (cont.)

CHULALONGKORN BUSINESS SCHOOL
FLAGSHIP FOR LIFE

Click on Keys and Access Tokens:

Application Actions

Regenerate Consumer Key and Secret | Change App Permissions

Your Access Token

You haven't authorized this application for your own account yet.

By creating your access token here, you will have everything you need to make API calls right away. The access token generated will be assigned your application's current permission level.

Token Actions

Create my access token

Your Access Token

This access token can be used to make API requests on your own account's behalf. Do not share your access token secret with anyone.

Access Token	344100790-rOcjGTw9wdmD5FVwacOaR7ZriGPRILBZiha43Tyg
Access Token Secret	2t41euCCQiwmdkihxkBWAS2rOTfTJpTAHUy27fOAZILWE
Access Level	Read and write
Owner	bobbylovemovie
Owner ID	344100790

Download the third-party libraries

```
$ wget https://github.com/bobbylovemovie/trainbigdata/raw/master/Spark/
twitter4j-core-4.0.2.jar
$ wget https://github.com/bobbylovemovie/trainbigdata/raw/master/Spark/
twitter4j-stream-4.0.2.jar
$ wget https://github.com/bobbylovemovie/trainbigdata/raw/master/Spark/
spark-streaming-twitter_2.10-1.2.0.jar
```

Run Spark-shell

```
$ spark-shell --jars spark-streaming-twitter_2.10-1.2.0.jar,twitter4j-
stream-4.0.2.jar,twitter4j-core-4.0.2.jar
```

Running Spark commands

```
$ scala> :paste
// Entering paste mode (ctrl-D to finish)
import org.apache.spark.streaming.twitter._
import twitter4j.auth._
import twitter4j.conf._
import org.apache.spark.streaming.{Seconds, StreamingContext}
import org.apache.spark._
import org.apache.spark.streaming._
import org.apache.spark.streaming.StreamingContext._
val ssc = new StreamingContext(sc, Seconds(10))
val cb = new ConfigurationBuilder
```

Running Spark commands(cont.)

```
cb.setDebugEnabled(true).setOAuthConsumerKey("eSevkYKyO94uGtFxxHoGwXDpX").setOAuthConsumerSecret("OvjqIF8VmNScaeMGZjz9e1fIfq0TxehmwkMJ454wHCCUG0AvED").setOAuthAccessToken("344100790-rOcjGTw9wdmD5FVwacOaR7ZriGPRlLBZiha43Tyg").setOAuthAccessTokenSecret("2t41euCCQiwmdkihxkBWAS2rOTfTJpTAHUy27fOAZlLWE")
val auth = new OAuthAuthorization(cb.build)
val tweets = TwitterUtils.createStream(ssc,Some(auth))
val status = tweets.map(status => status.getText)
status.print
ssc.checkpoint("hdfs:///user/cloudera/data/tweets")
ssc.start
ssc.awaitTermination
```

HUE | Query Editors | Data Browsers | Workflows | Search | Security

File Browser

Search for file name | Actions | Move to trash | Upload | New

Home / user / cloudera / data / **tweets** | History | Trash

Name	Size	User	Group	Permissions	Date
..		cloudera	cloudera	drwxr-xr-x	October 15, 2016 09:07 AM
.		cloudera	cloudera	drwxr-xr-x	October 15, 2016 09:09 AM
5f370383-6957-4d7f-8e04-20526f5fcec1		cloudera	cloudera	drwxr-xr-x	October 15, 2016 09:07 AM
checkpoint-1476547700000	5.0 KB	cloudera	cloudera	-rw-r--r--	October 15, 2016 09:08 AM
checkpoint-1476547710000	5.0 KB	cloudera	cloudera	-rw-r--r--	October 15, 2016 09:08 AM
checkpoint-1476547720000	5.0 KB	cloudera	cloudera	-rw-r--r--	October 15, 2016 09:08 AM
checkpoint-1476547730000	5.0 KB	cloudera	cloudera	-rw-r--r--	October 15, 2016 09:08 AM
checkpoint-1476547740000	5.0 KB	cloudera	cloudera	-rw-r--r--	October 15, 2016 09:09 AM
checkpoint-1476547750000	5.0 KB	cloudera	cloudera	-rw-r--r--	October 15, 2016 09:09 AM
checkpoint-1476547760000	5.0 KB	cloudera	cloudera	-rw-r--r--	October 15, 2016 09:09 AM