Hadoop Workshop using Cloudera on Amazon EC2

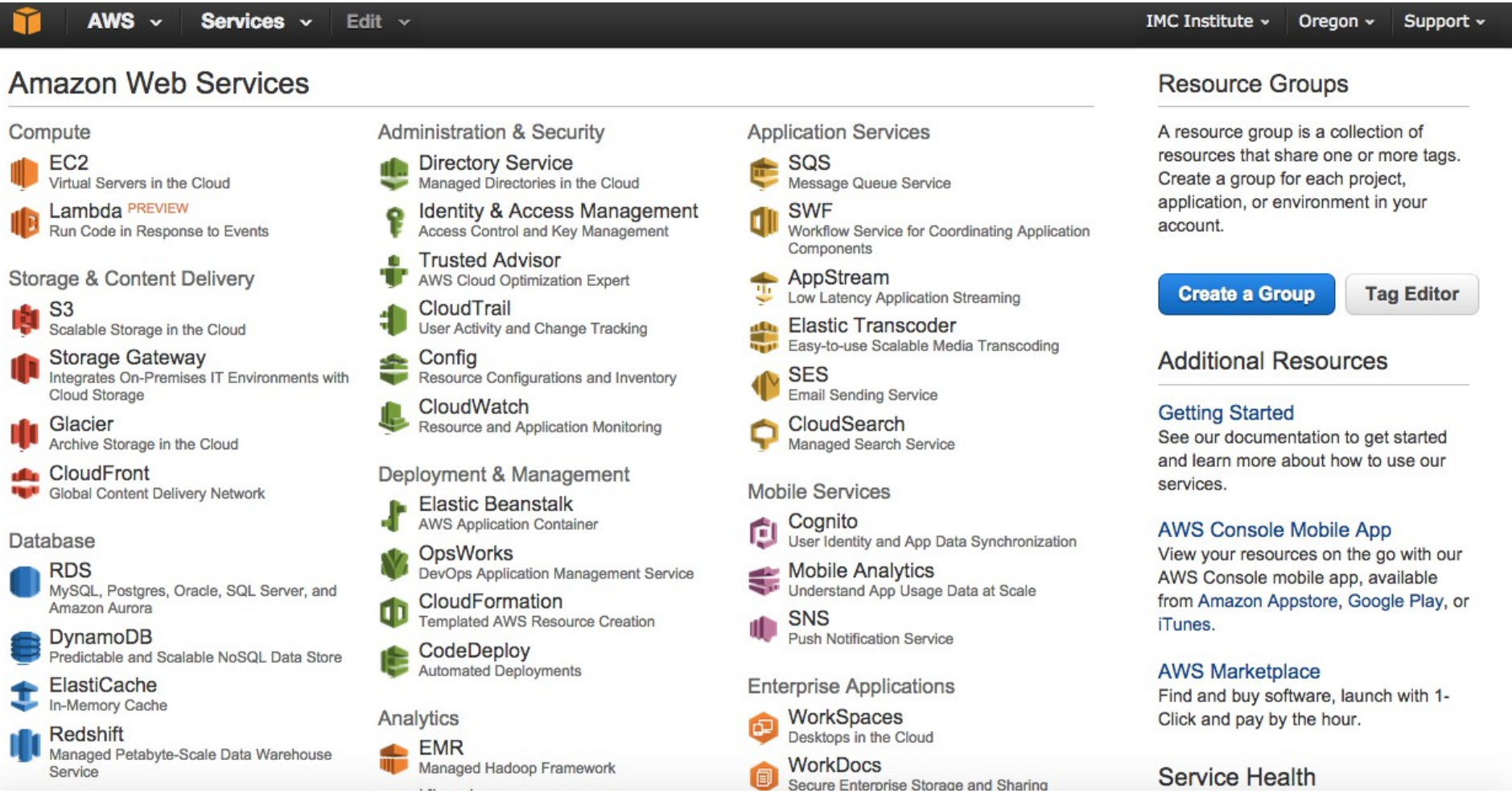
May 2015

Dr.Thanachart Numnonda
IMC Institute
thanachart@imcinstitute.com

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

Hands-On: Launch a virtual server on EC2 Amazon Web Services

Danairat T., , danairat@gmail.com: Thanachart Numnonda, thanachart@imcinstitute.com

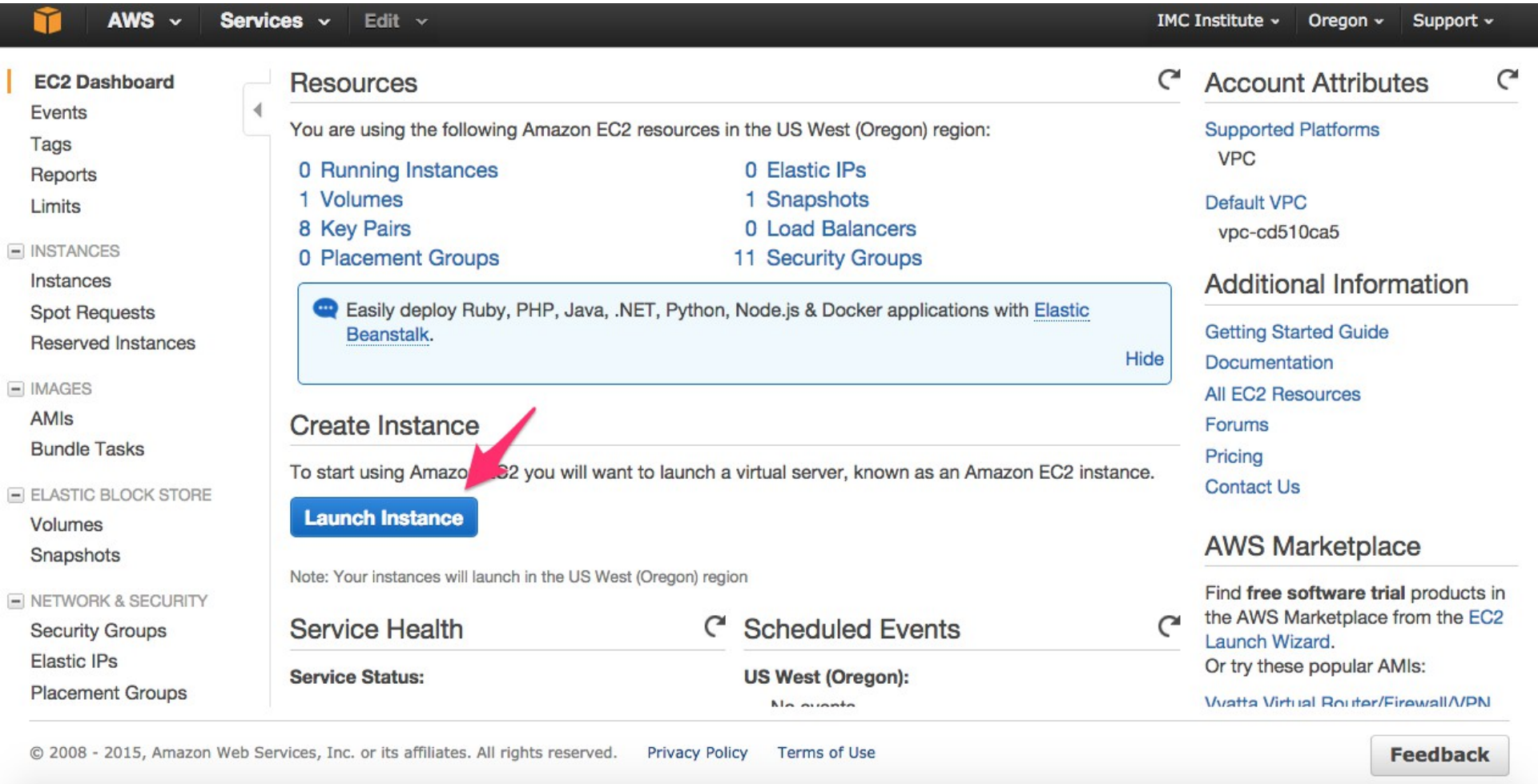

AWS
Services
Edit
IMC Institute
Oregon
Support
Amazon Web Services
Compute
EC2
Virtual Servers in the Cloud
Lambda PREVIEW
Run Code in Response to Events
Storage & Content Delivery
S3
Scalable Storage in the Cloud
Storage Gateway
Integrates On-Premises IT Environments with Cloud Storage
Glacier
Archive Storage in the Cloud
CloudFront
Global Content Delivery Network
Database
RDS
MySQL, Postgres, Oracle, SQL Server, and Amazon Aurora
DynamoDB
Predictable and Scalable NoSQL Data Store
ElastiCache
In-Memory Cache
Redshift
Managed Petabyte-Scale Data Warehouse Service
Administration & Security
Directory Service
Managed Directories in the Cloud
Identity & Access Management
Access Control and Key Management
Trusted Advisor
AWS Cloud Optimization Expert
CloudTrail
User Activity and Change Tracking
Config
Resource Configurations and Inventory
CloudWatch
Resource and Application Monitoring
Deployment & Management
Elastic Beanstalk
AWS Application Container
OpsWorks
DevOps Application Management Service
CloudFormation
Templated AWS Resource Creation
CodeDeploy
Automated Deployments
Analytics
EMR
Managed Hadoop Framework
Application Services
SQS
Message Queue Service
SWF
Workflow Service for Coordinating Application Components
AppStream
Low Latency Application Streaming
Elastic Transcoder
Easy-to-use Scalable Media Transcoding
SES
Email Sending Service
CloudSearch
Managed Search Service
Mobile Services
Cognito
User Identity and App Data Synchronization
Mobile Analytics
Understand App Usage Data at Scale
SNS
Push Notification Service
Enterprise Applications
WorkSpaces
Desktops in the Cloud
WorkDocs
Secure Enterprise Storage and Sharing
Resource Groups
A resource group is a collection of resources that share one or more tags. Create a group for each project, application, or environment in your account.
Create a Group
Tag Editor
Additional Resources
Getting Started
See our documentation to get started and learn more about how to use our services.
AWS Console Mobile App
View your resources on the go with our AWS Console mobile app, available from Amazon Appstore, Google Play, or iTunes.
AWS Marketplace
Find and buy software, launch with 1-Click and pay by the hour.
Service Health

Virtual Server

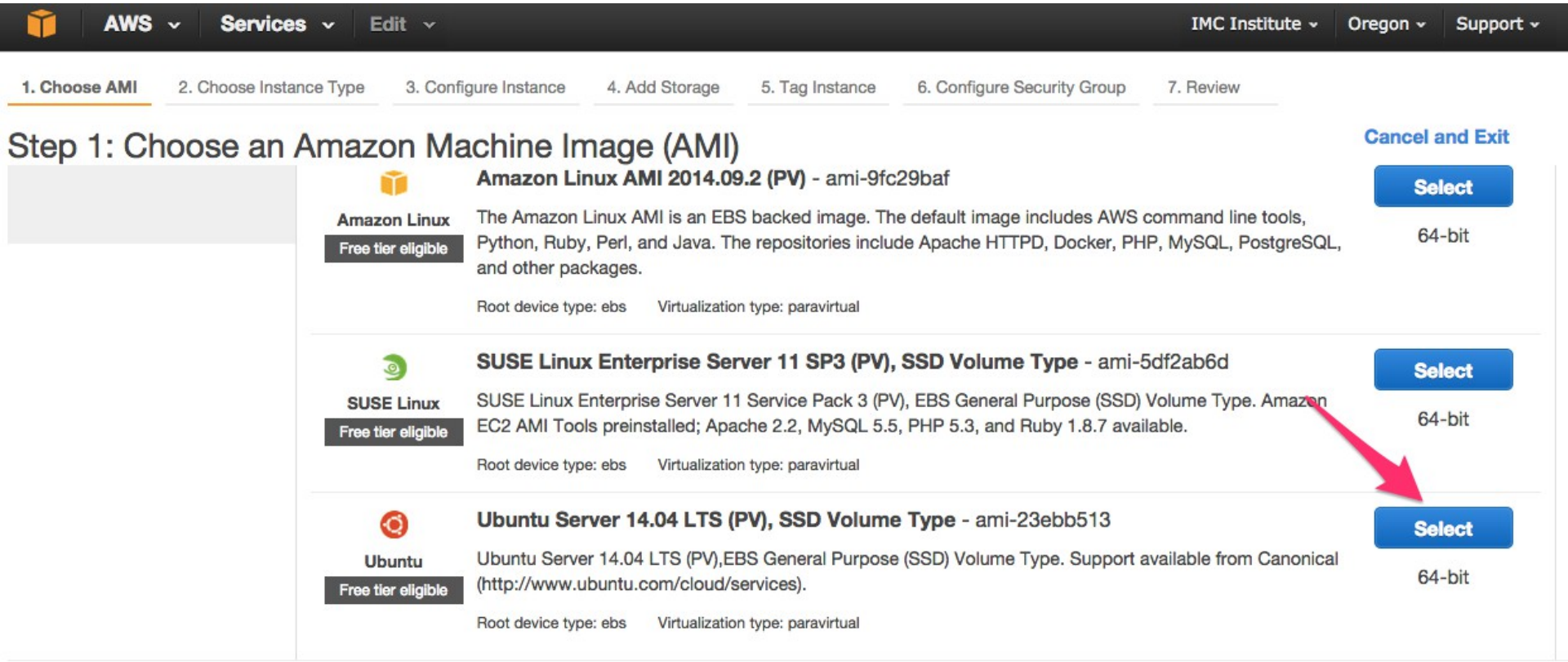
This lab will use a EC2 virtual server to install a Hadoop server using the following features:

1. Ubuntu Server 14.04 LTS
2. m3.xLarge 4vCPU, 15 GB memory, 80 GB SSD
3. Security group: create new
4. Keypair: imchadoop

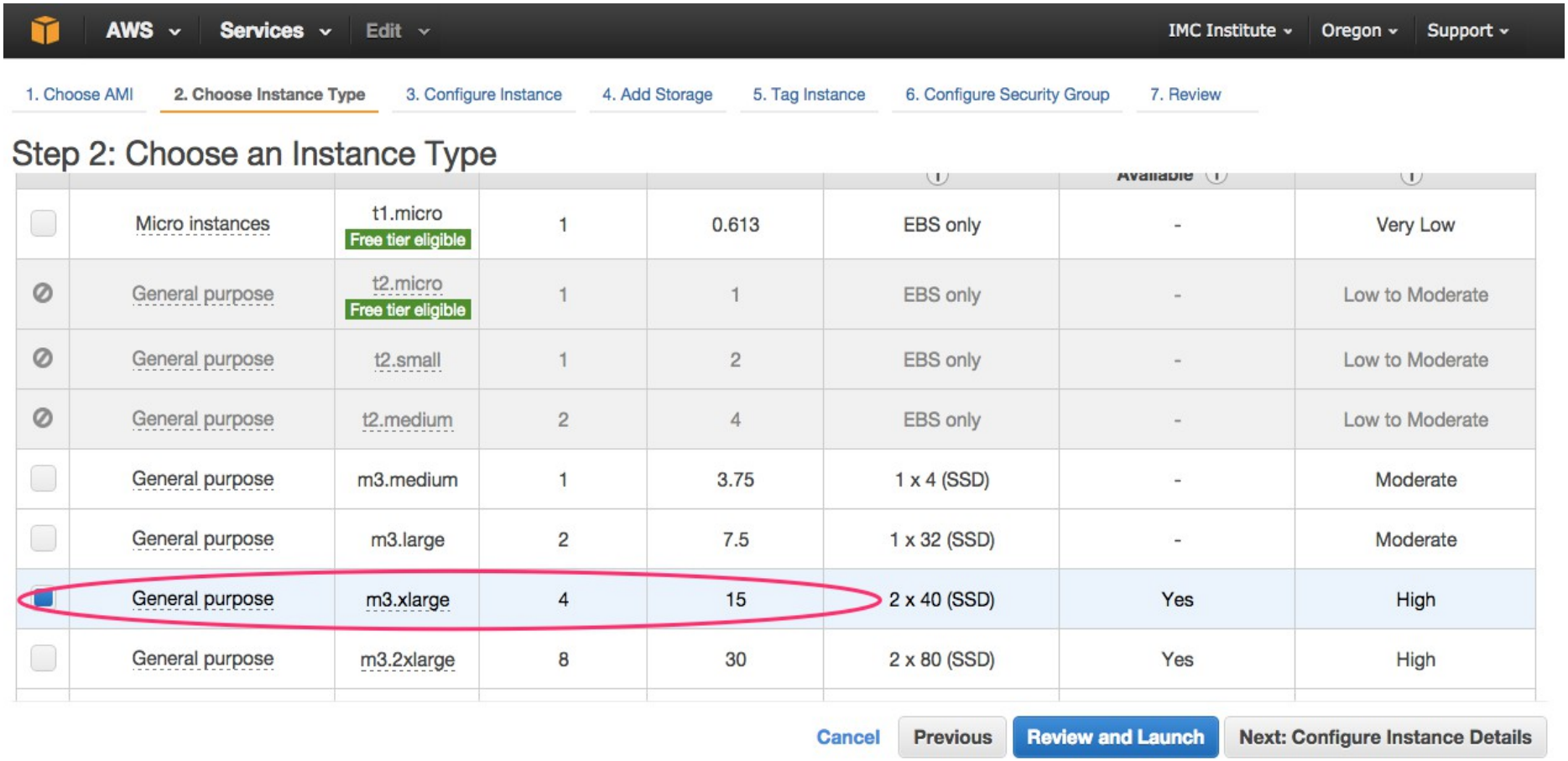
Select a EC2 service and click on Lunch Instance



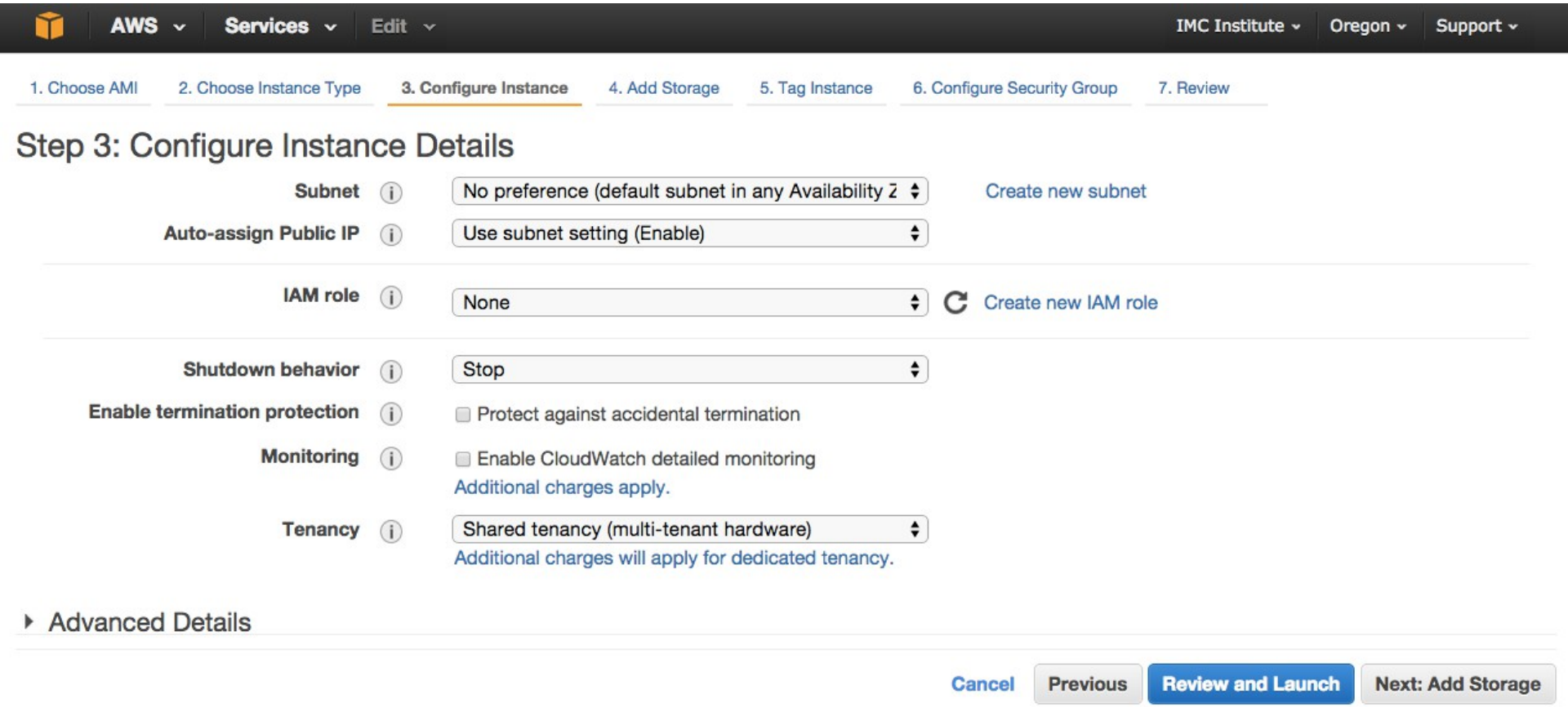
Select an Amazon Machine Image (AMI) and Ubuntu Server 14.04 LTS (PV)



Choose m3.xlarge Type virtual server



Leave configuration details as default



Add Storage: 30 GB

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

Step 4: Add Storage

Your instance will be launched with the following storage device settings. You can attach additional EBS volumes and instance store volumes to your instance, or edit the settings of the root volume. You can also attach additional EBS volumes after launching an instance, but not instance store volumes. Learn more about storage options in Amazon EC2.

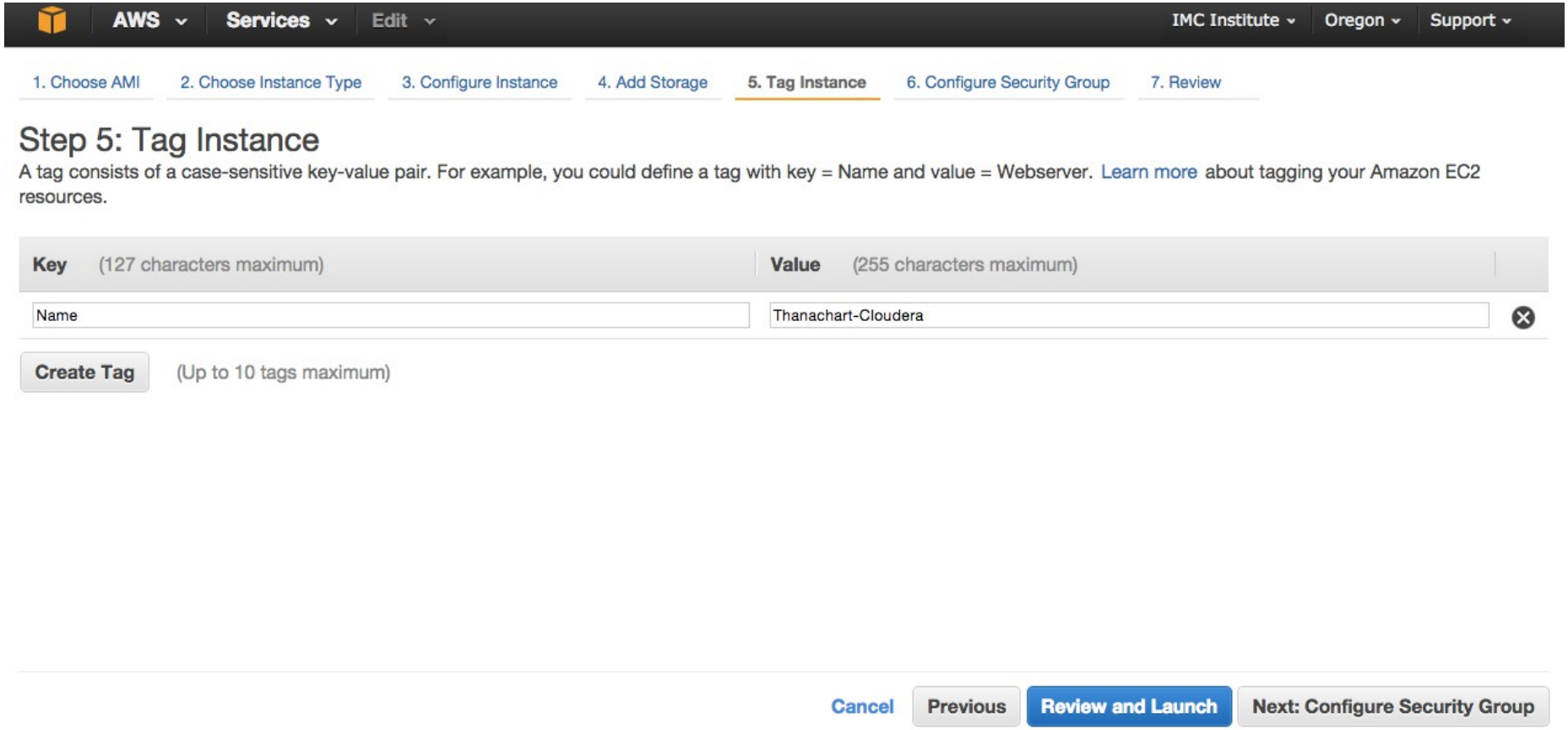
Type	Device	Snapshot	Size (GiB)	Volume Type	IOPS	Delete on Termination	Encrypted	
Root	/dev/sda1	snap-0e023b4e	30	General Purpose (S	90 / 3000	☑	Not Encrypted	
Instance Store 0	/dev/sdb	N/A	N/A	N/A	N/A	N/A	Not Encrypted	⊗

Add New Volume

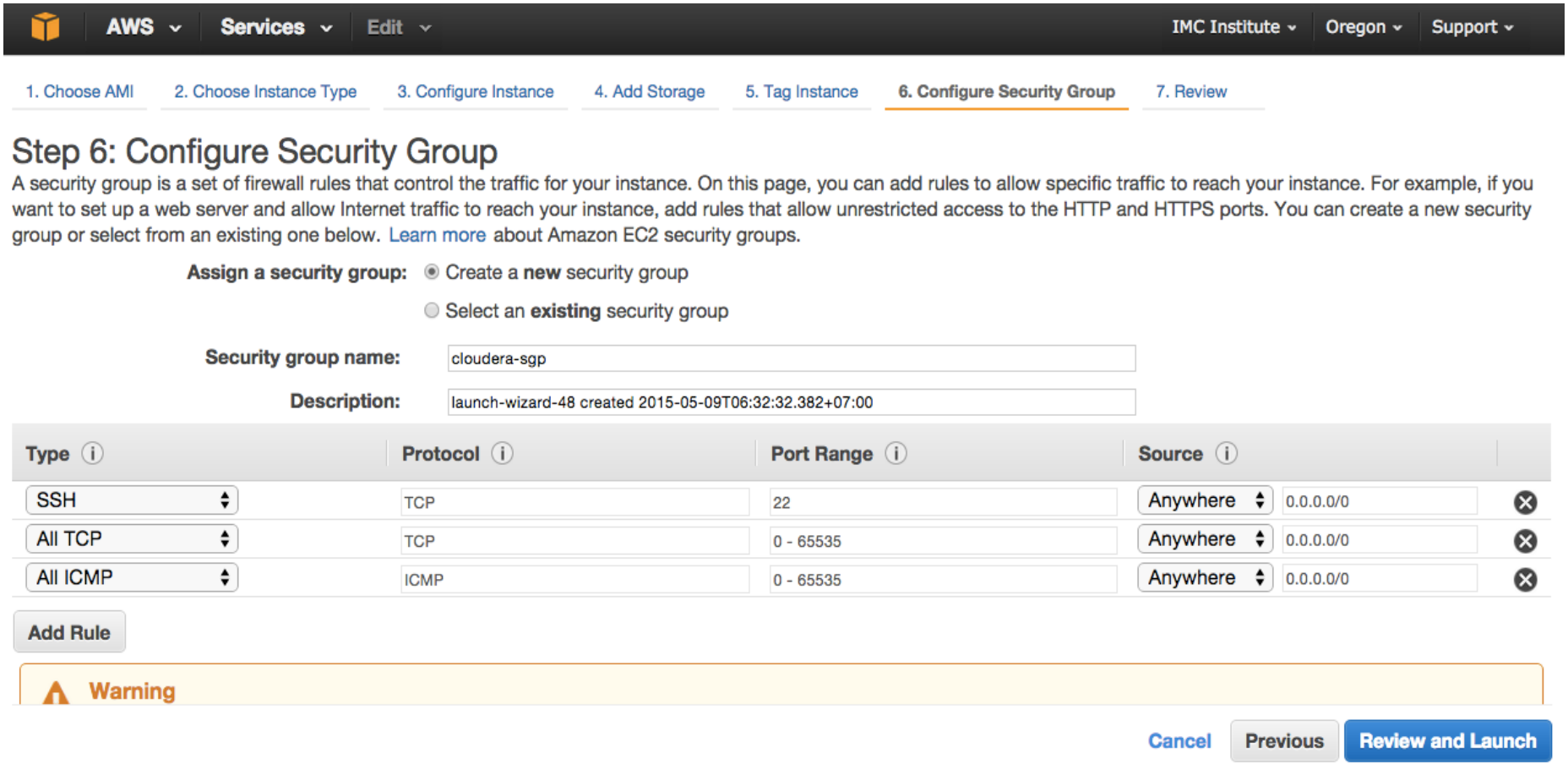
Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage. Learn more about free usage tier eligibility and usage restrictions.

Cancel | Previous | Review and Launch | Next: Tag Instance

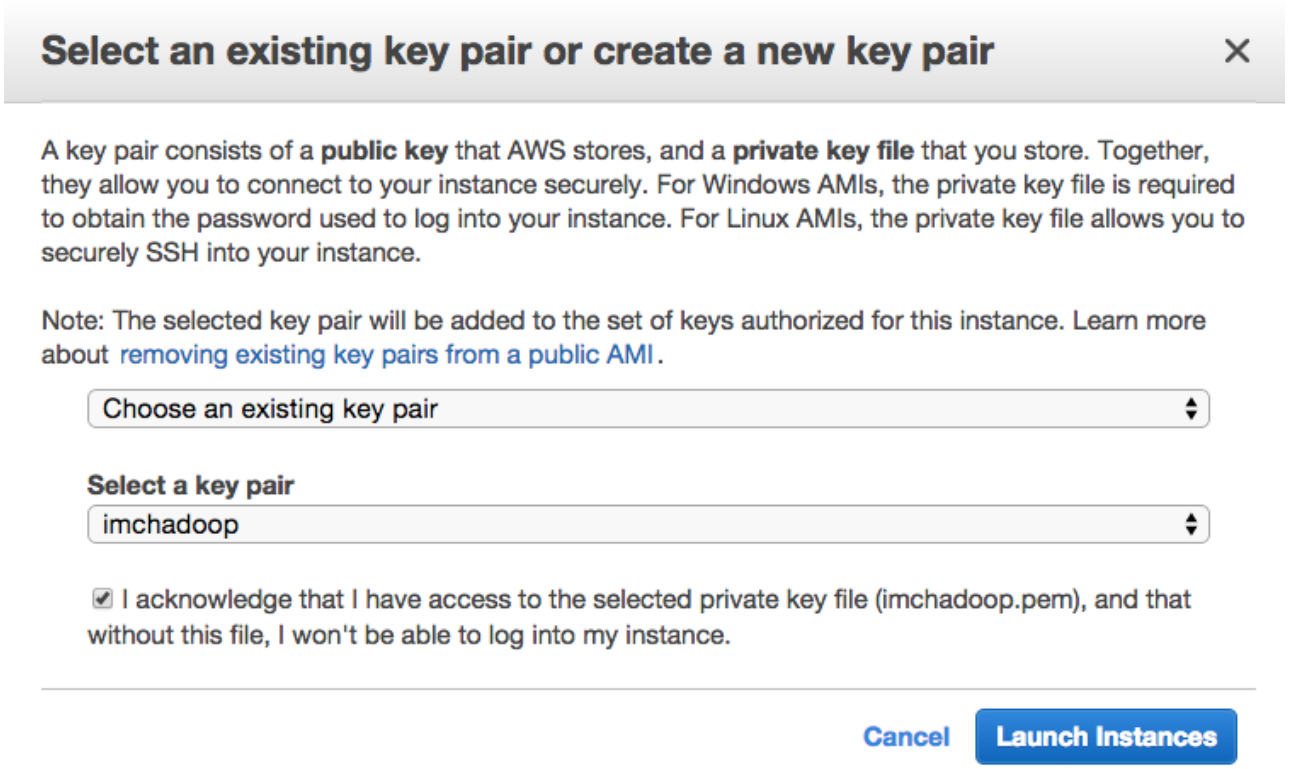
Name the instance



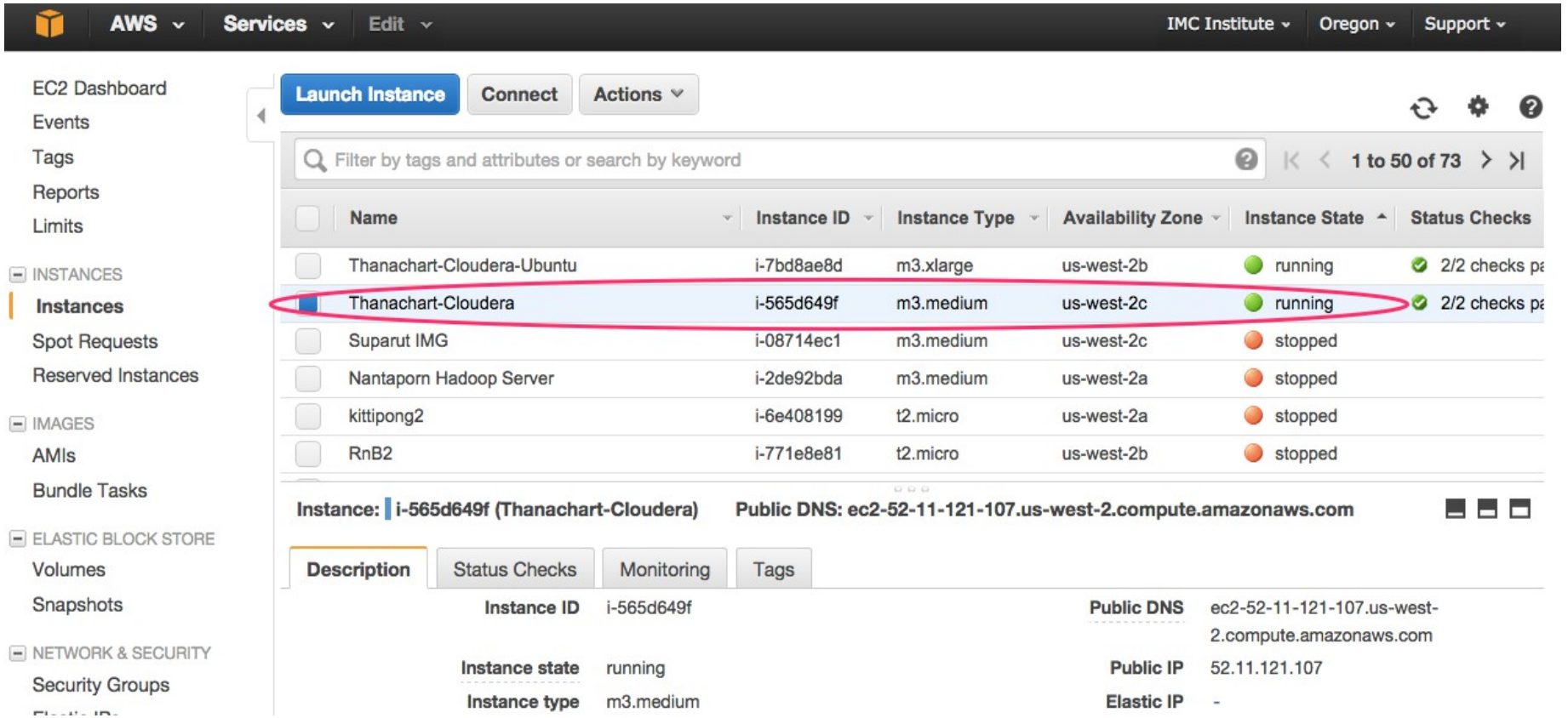
Select Create a new security group > Add Rule as follows



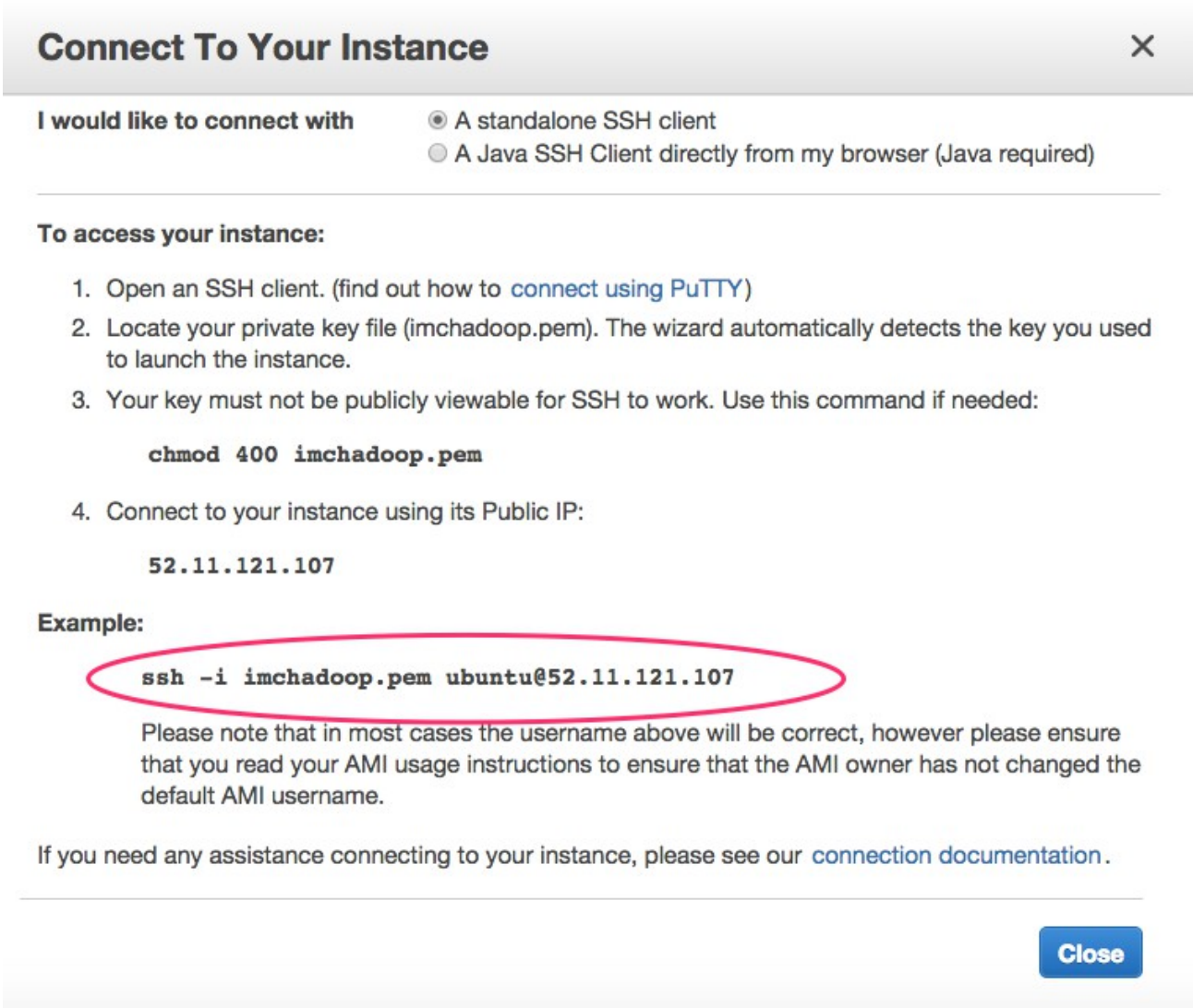
Click Launch and choose imchadoop as a key pair



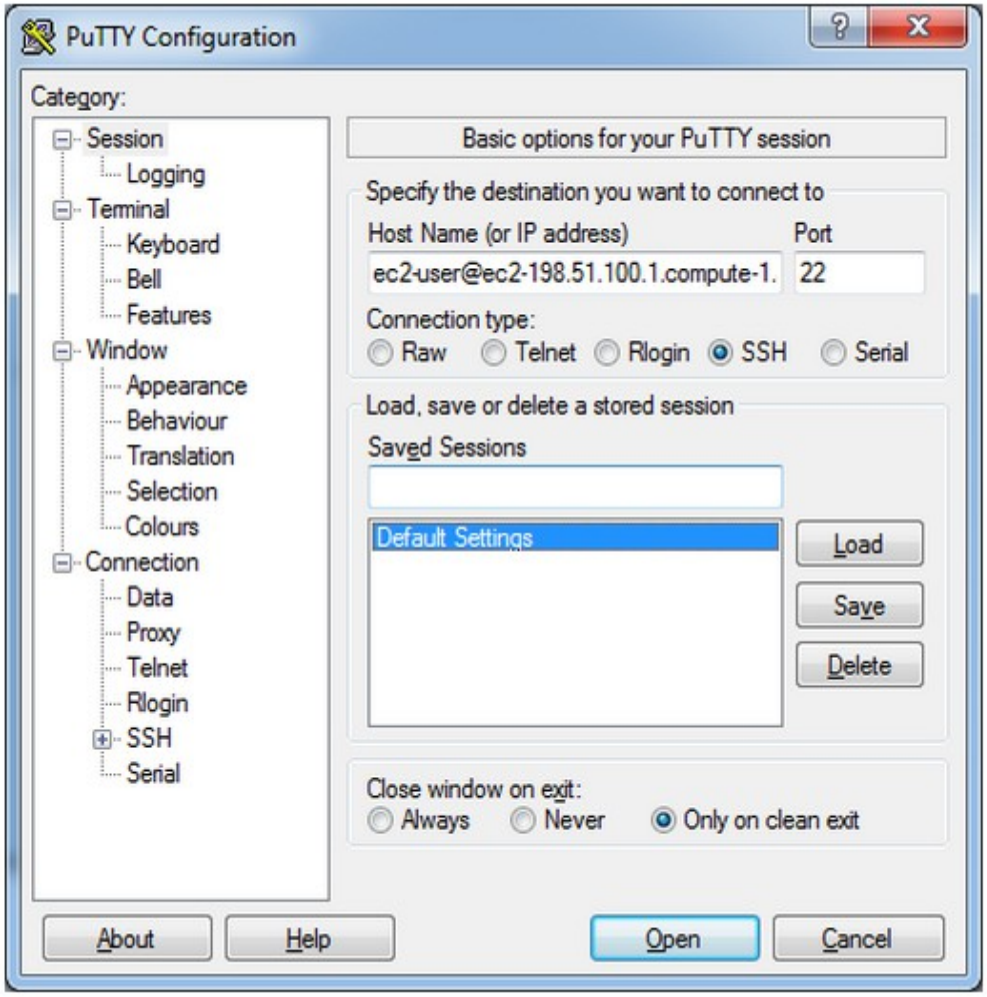
Review an instance / click Connect for an instruction to connect to the instance



Connect to an instance from Mac/Linux



Connect to an instance from Windows using Putty



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:~$
```

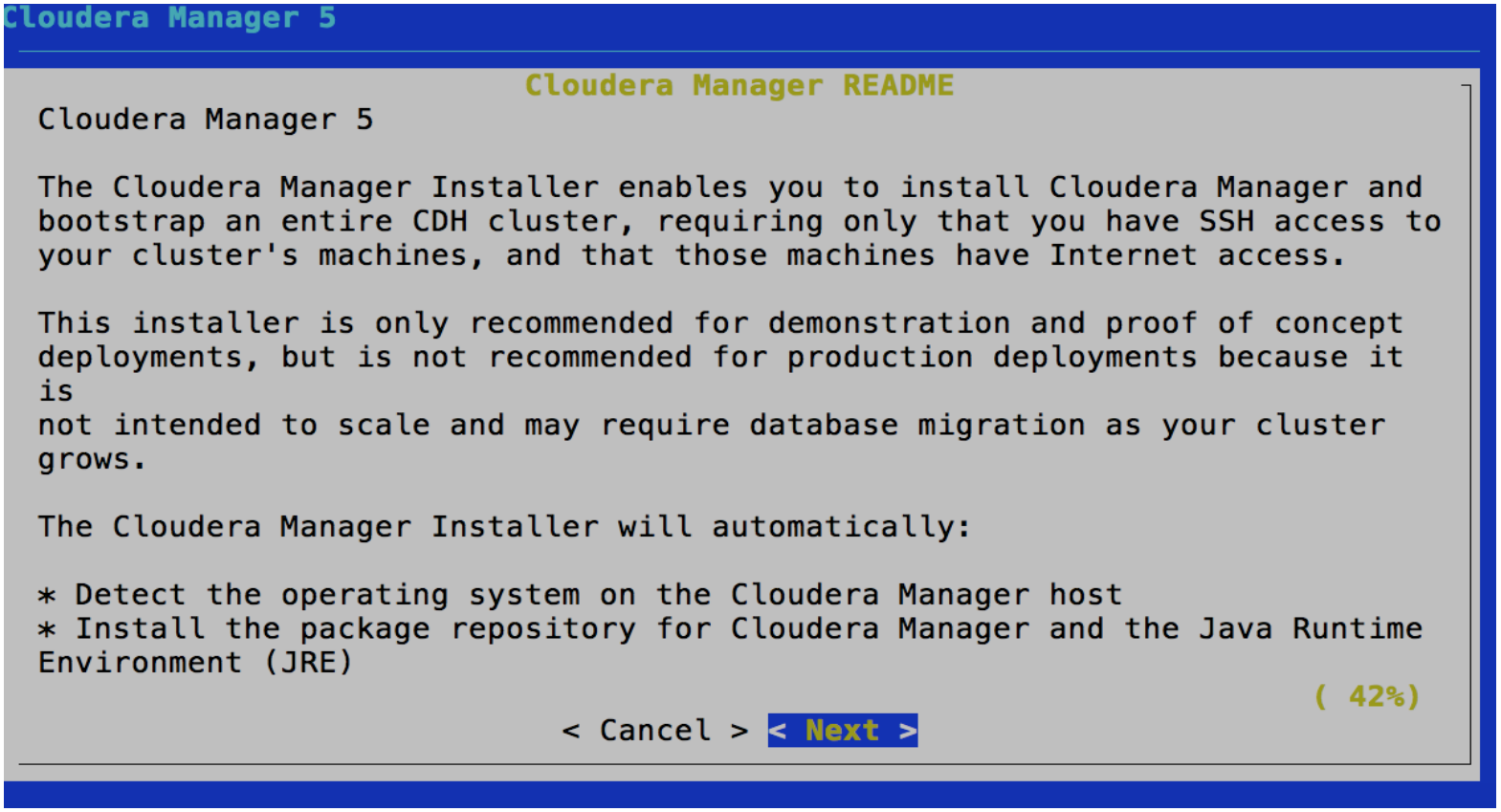
Hands-On: Installing Cloudera on EC2

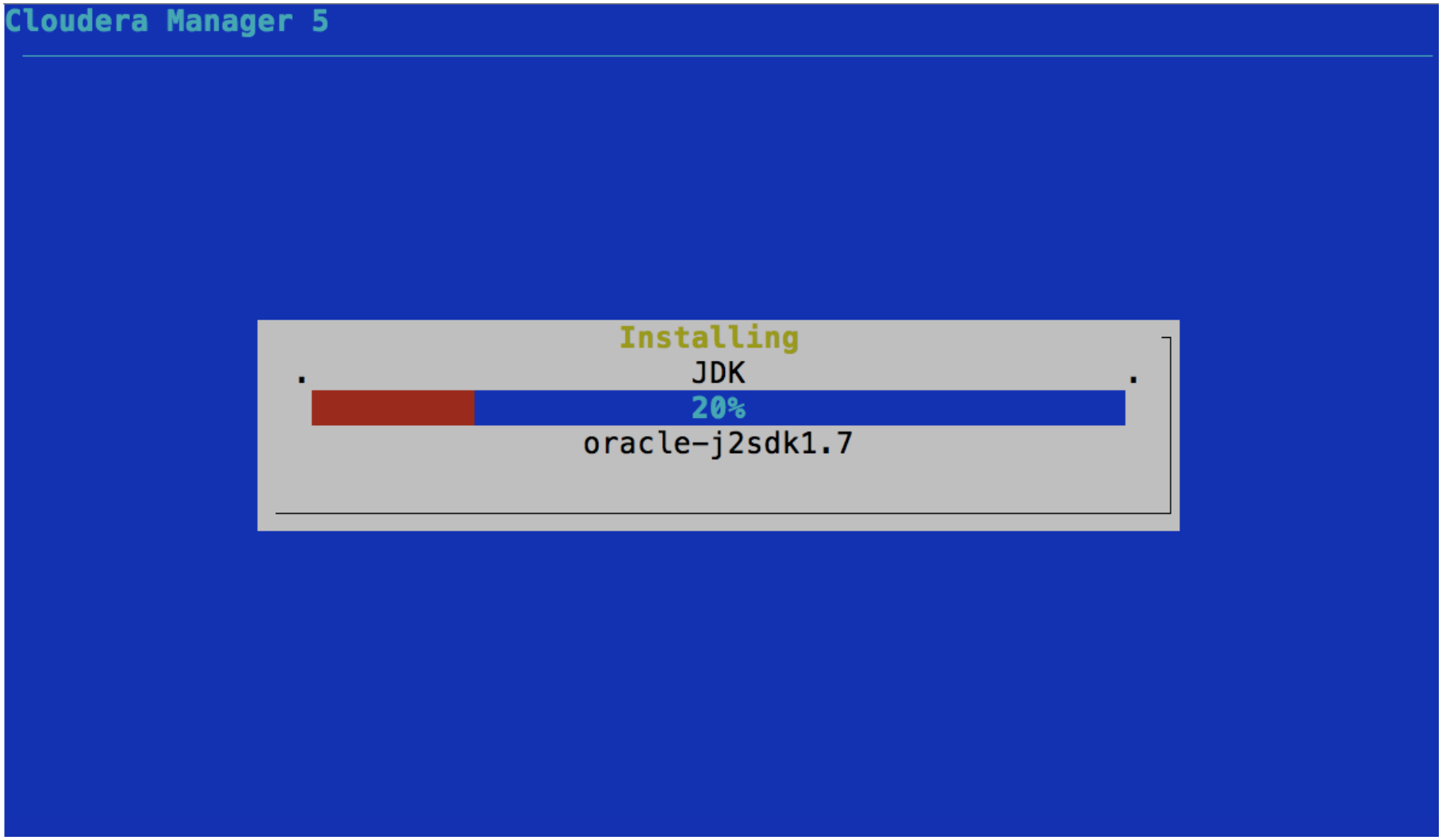
Download Cloudera Manager

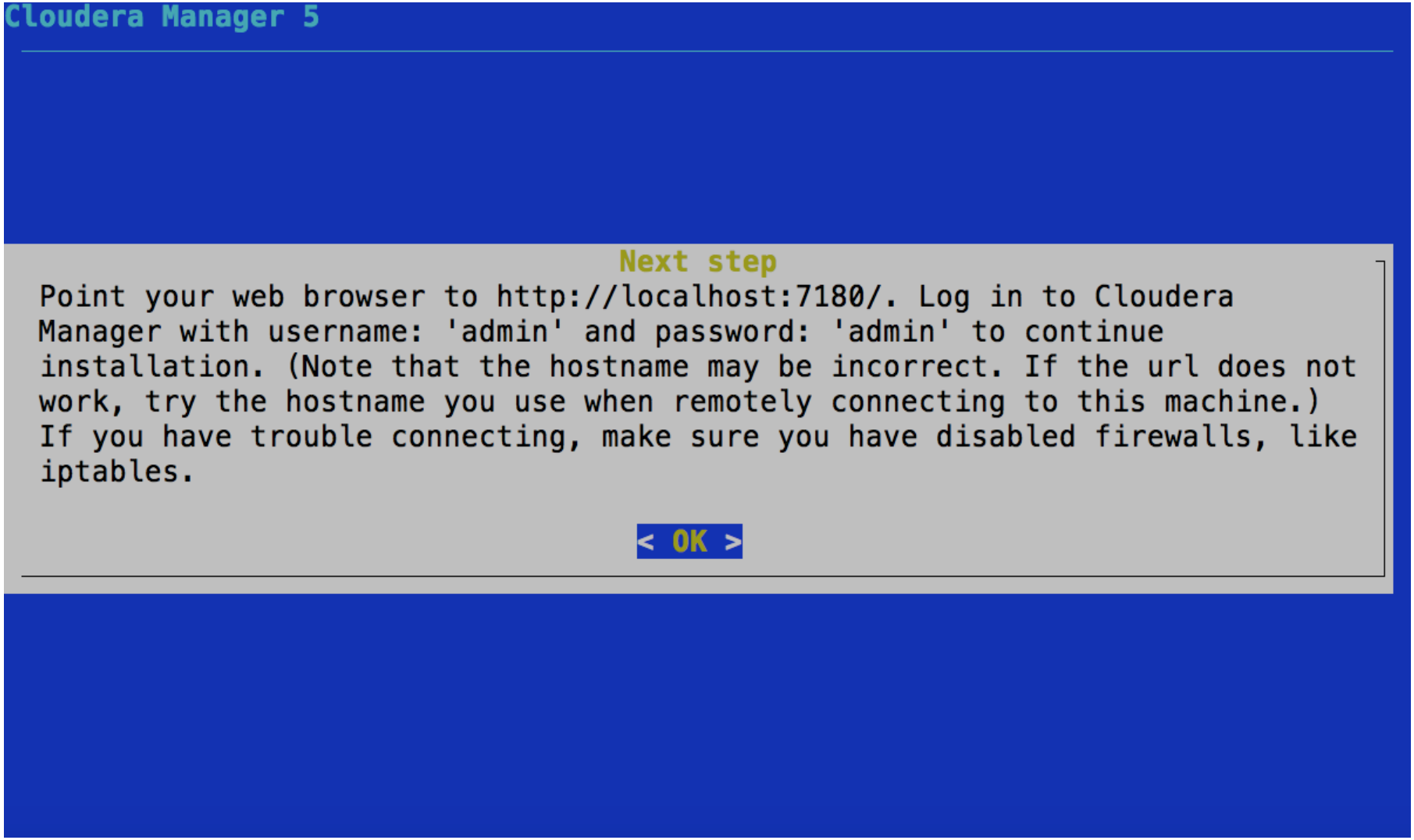
1) Type command >wget http://archive.cloudera.com/cm5/installer/latest/cloudera-manager-installer.bin

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

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



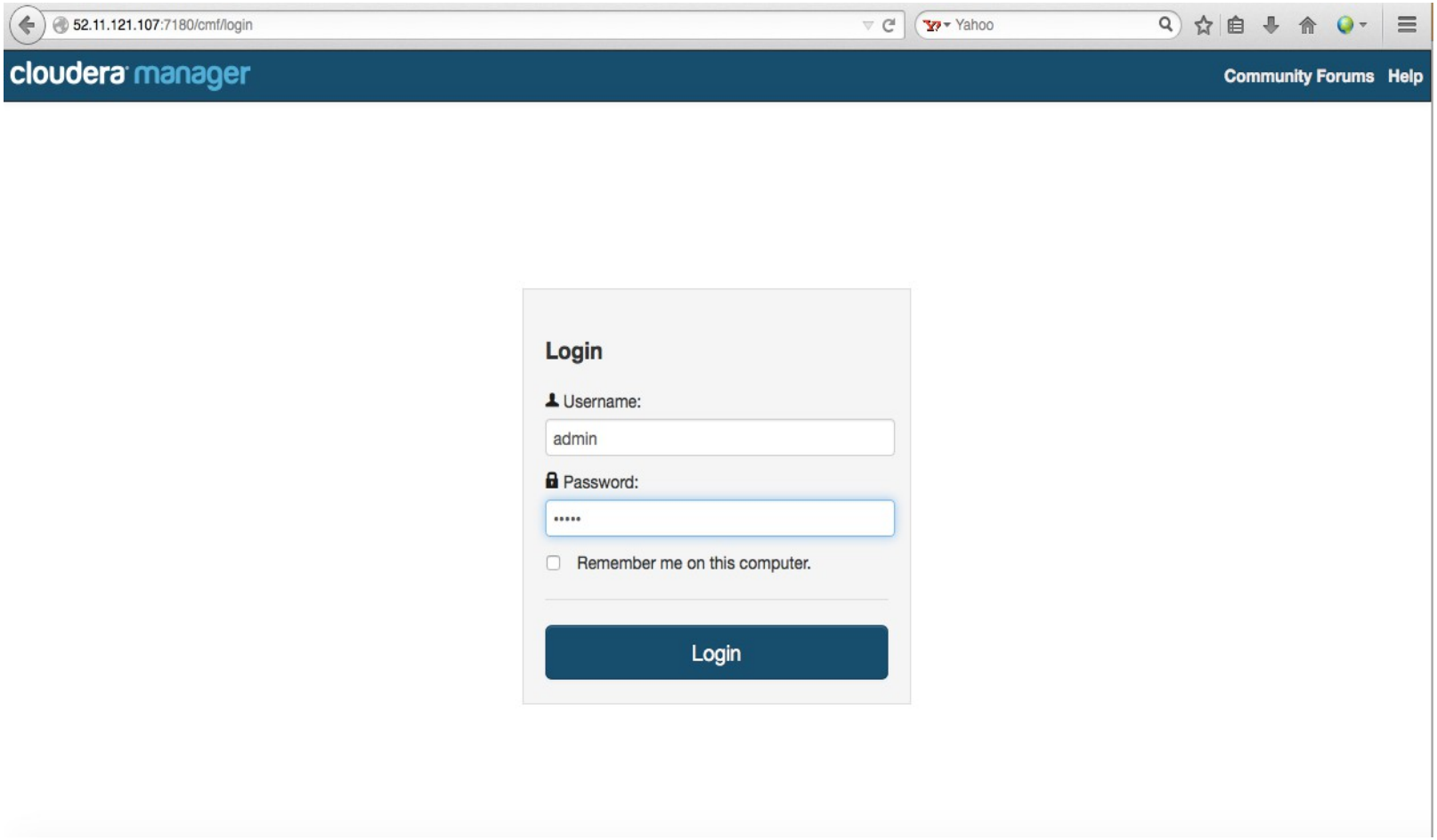

Cloudera Manager 5
Installing
JDK
20%
oracle-j2sdk1.7


Cloudera Manager 5
Next step
Point your web browser to http://localhost:7180/. Log in to Cloudera Manager with username: 'admin' and password: 'admin' to continue installation. (Note that the hostname may be incorrect. If the url does not work, try the hostname you use when remotely connecting to this machine.) If you have trouble connecting, make sure you have disabled firewalls, like iptables.
< OK >

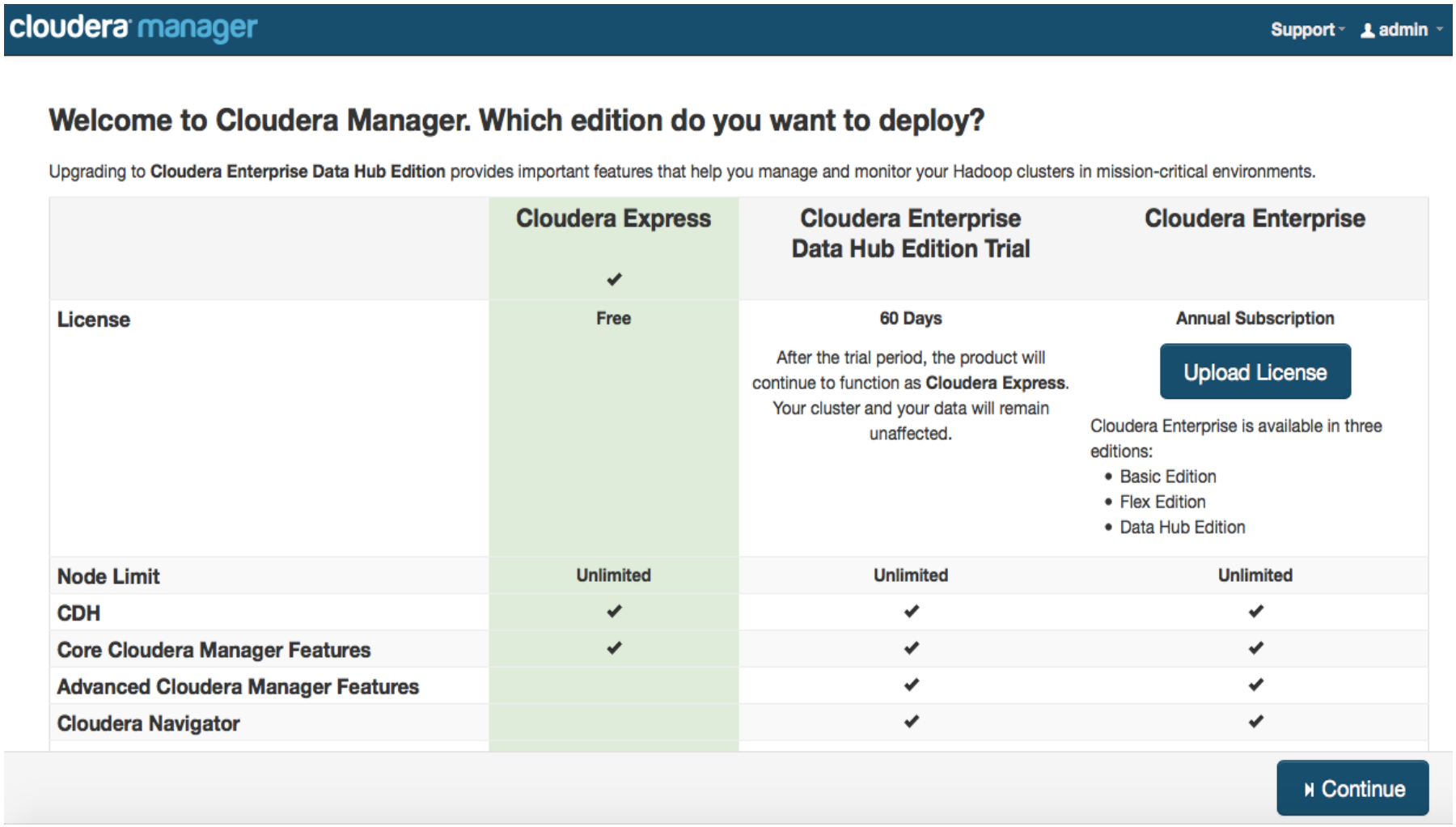
Login to Cloudera Manager

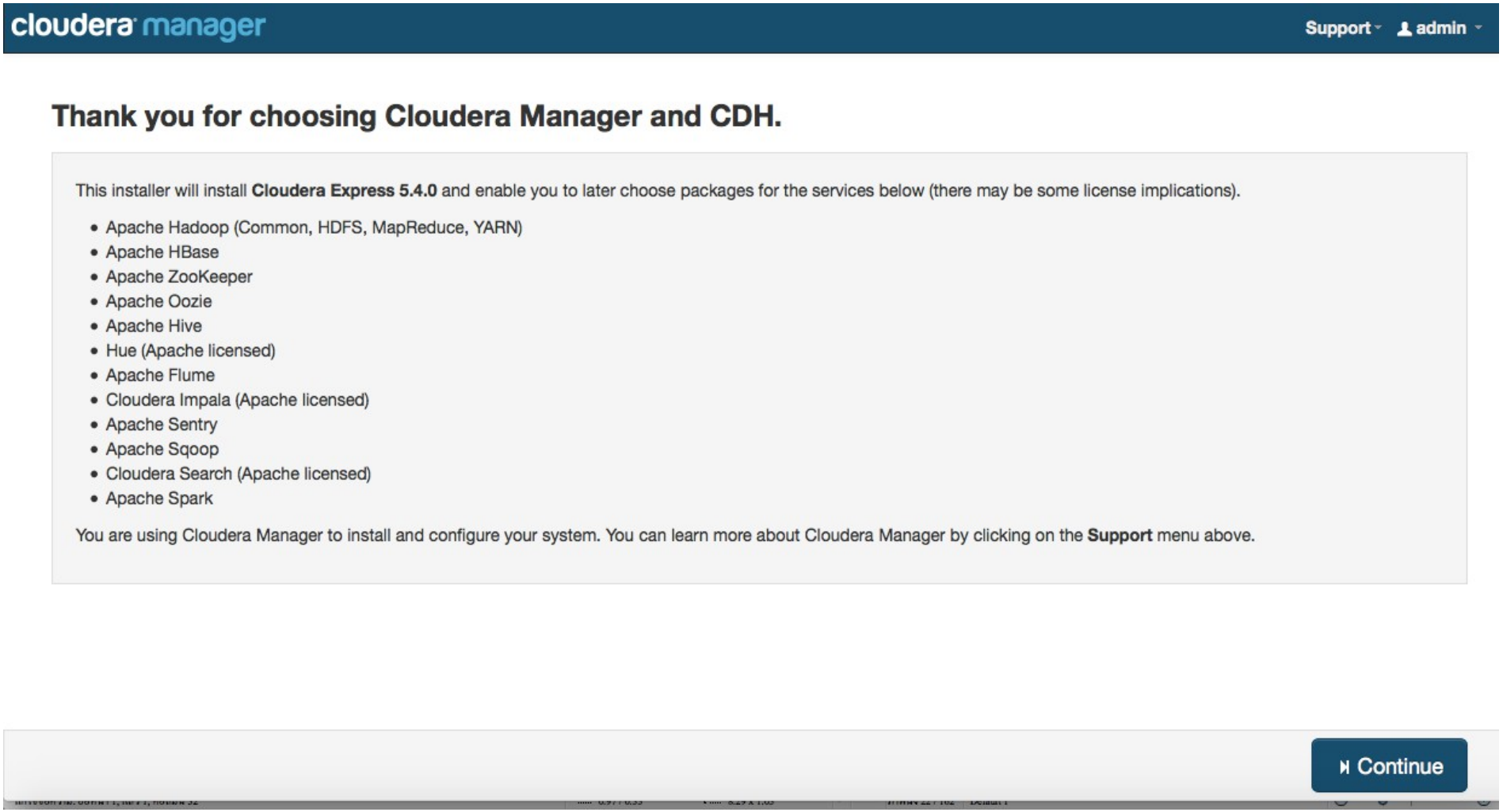
Wait several minutes for the Cloudera Manager Server to complete its startup.

Then running web browser: http:// public-ip: 7180

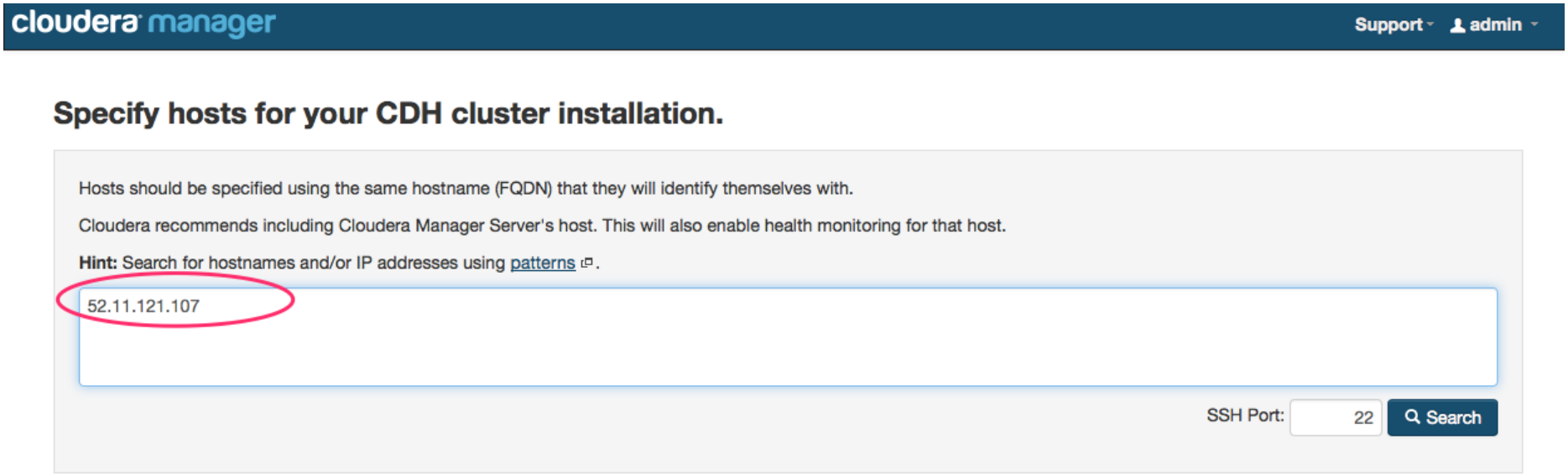


Select Cloudera Express Edition




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 instance <public 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 will also enable health monitoring for that host.

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

1 hosts scanned, 1 running SSH.

New Search

Expanded Query	Hostname (FQDN)	IP Address	Currently Managed	Result
52.11.121.107	52.11.121.107	52.11.121.107	No	✓ Host ready: 0 ms response time.

Back

Continue

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-5.4.0-1.cdh5.4.0.p0.27
- ○ CDH-4.7.1-1.cdh4.7.1.p0.47

Versions of CDH that are too new for this version of Cloudera Manager (5.4.0) will not be shown.

Additional Parcels

- ○ ACCUMULO-1.6.0-1.cdh5.1.4.p0.116
- ○ ACCUMULO-1.4.4-1.cdh4.5.0.p0.65
- ◉ None

- ○ KAFKA-0.8.2.0-1.kafka1.3.0.p0.29
- ◉ None

- ○ KEYTRUSTEE-5.4.0-1.cdh5.4.0.p0.193

Back

1 2 3 4 5 6 7 8

Continue

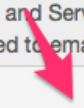
Cluster Installation

JDK Installation Options

Oracle Binary Code License Agreement for the Java SE Platform Products and JavaFX

ORACLE AMERICA, INC. ("ORACLE"), FOR AND ON BEHALF OF ITSELF AND ITS SUBSIDIARIES AND AFFILIATES UNDER COMMON CONTROL, IS WILLING TO LICENSE THE SOFTWARE TO YOU ONLY UPON THE CONDITION THAT YOU ACCEPT ALL OF THE TERMS CONTAINED IN THIS BINARY CODE LICENSE AGREEMENT AND SUPPLEMENTAL LICENSE TERMS (COLLECTIVELY "AGREEMENT"). PLEASE READ THE AGREEMENT CAREFULLY. BY SELECTING THE "ACCEPT LICENSE AGREEMENT" (OR THE EQUIVALENT) BUTTON AND/OR BY USING THE SOFTWARE YOU ACKNOWLEDGE THAT YOU HAVE READ THE TERMS AND AGREE TO THEM. IF YOU ARE AGREEING TO THESE TERMS ON BEHALF OF A COMPANY OR OTHER LEGAL ENTITY, YOU REPRESENT THAT YOU HAVE THE LEGAL AUTHORITY TO BIND THE LEGAL ENTITY TO THESE TERMS. IF YOU DO NOT HAVE SUCH AUTHORITY, OR IF YOU DO NOT WISH TO BE BOUND BY THE TERMS, THEN SELECT THE "DECLINE LICENSE AGREEMENT" (OR THE EQUIVALENT) BUTTON AND YOU MUST NOT USE THE SOFTWARE ON THIS SITE OR ANY OTHER MEDIA ON WHICH THE SOFTWARE IS CONTAINED.

1. DEFINITIONS. "Software" means the software identified above in binary form that you selected for download, install or use (in the version You selected for download, install or use) from Oracle or its authorized licensees, any other machine readable materials (including, but not limited to, libraries, source files, header files, and data files), any updates or error corrections provided by Oracle, and any user manuals, programming guides and other documentation provided to you by Oracle under this Agreement. "General Purpose Desktop Computers and Servers" means computers, including desktop and laptop computers, or servers, used for general computing functions under end user control (such as but not specifically limited to email, general purpose Internet browsing, and office suite productivity tools). The use of Software in systems and solutions that provide dedicated



☑ Install Oracle Java SE Development Kit (JDK)

Check this box to accept the Oracle Binary Code License Agreement and install the JDK. Leave it unchecked to use a currently installed JDK.

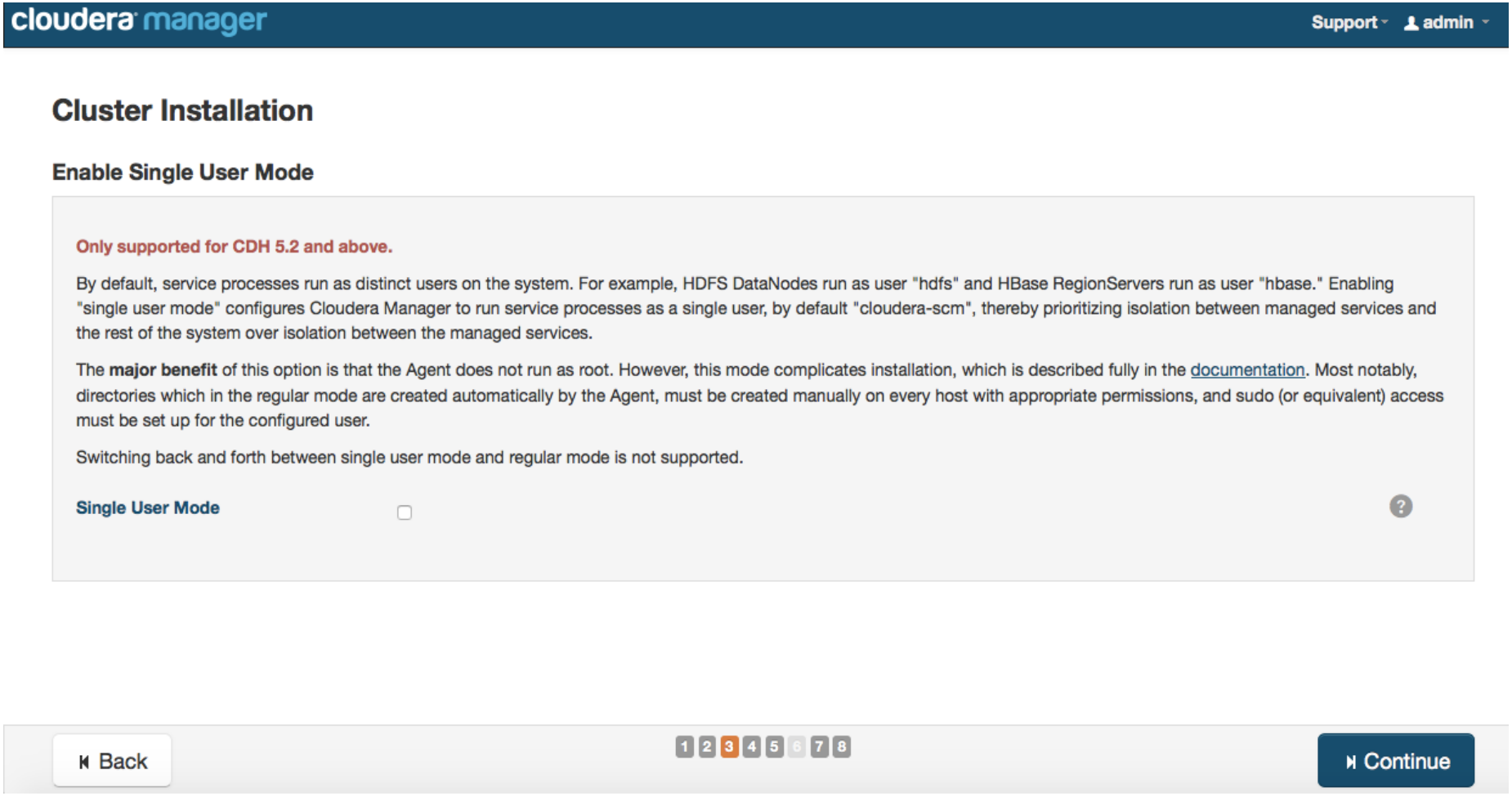
☑ Install Java Unlimited Strength Encryption Policy Files

Check this checkbox if local laws permit you to deploy unlimited strength encryption and you are running a secure cluster.

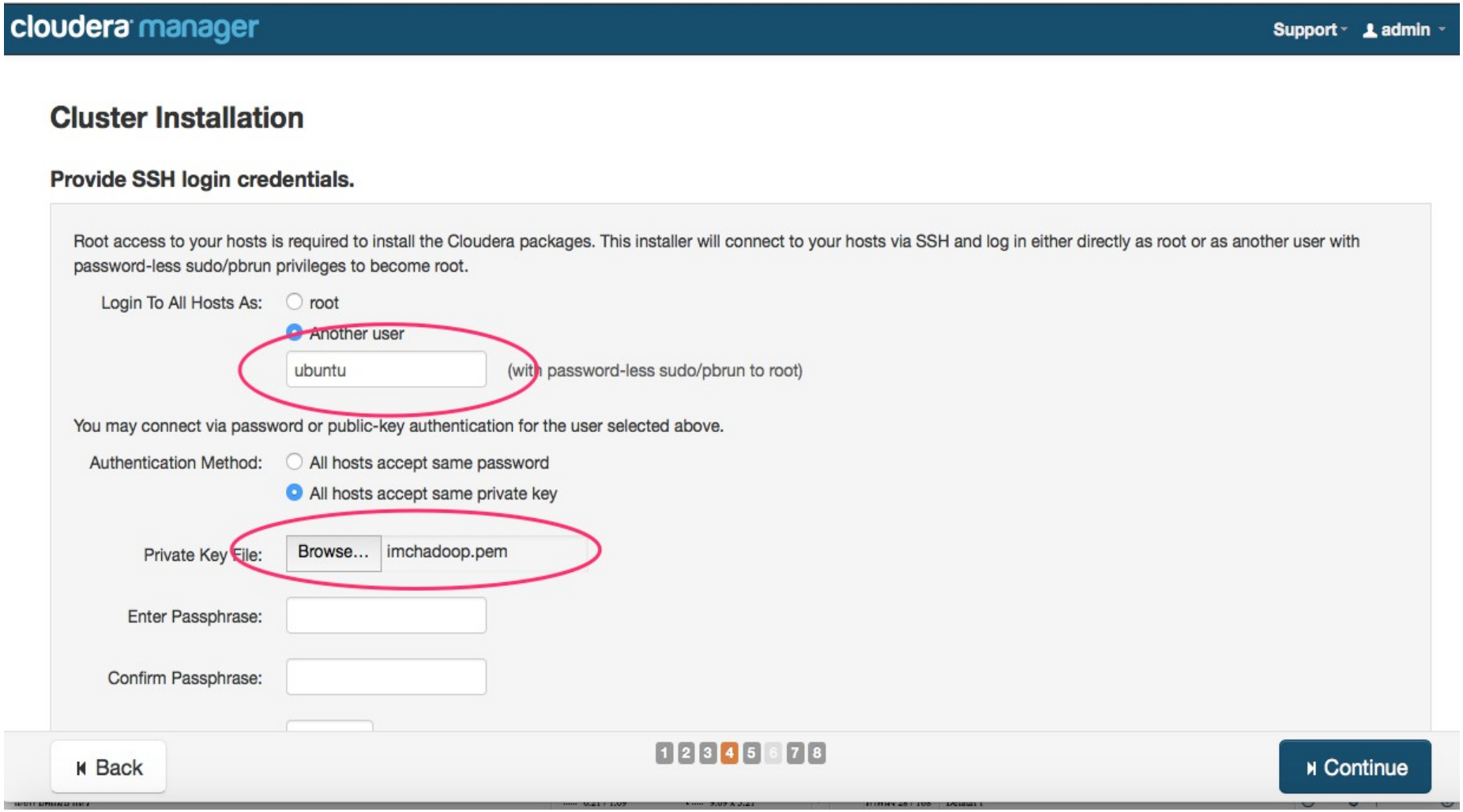
Back

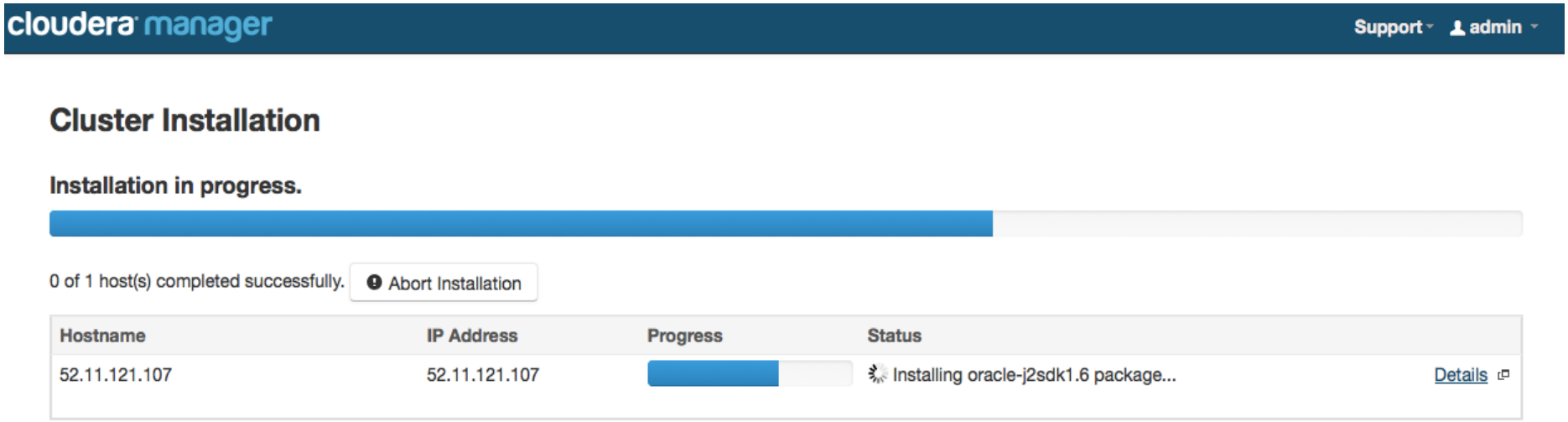
1 2 3 4 5 6 7 8

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. However, this mode complicates installation, which is described fully in the documentation. Most notably, directories which in the regular mode are created automatically by the Agent, must be created manually on every host with appropriate permissions, and sudo (or equivalent) access must be set up for the configured user.
Switching back and forth between single user mode and regular mode is not supported.
Single User Mode
Back
1 2 3 4 5 6 7 8
Continue

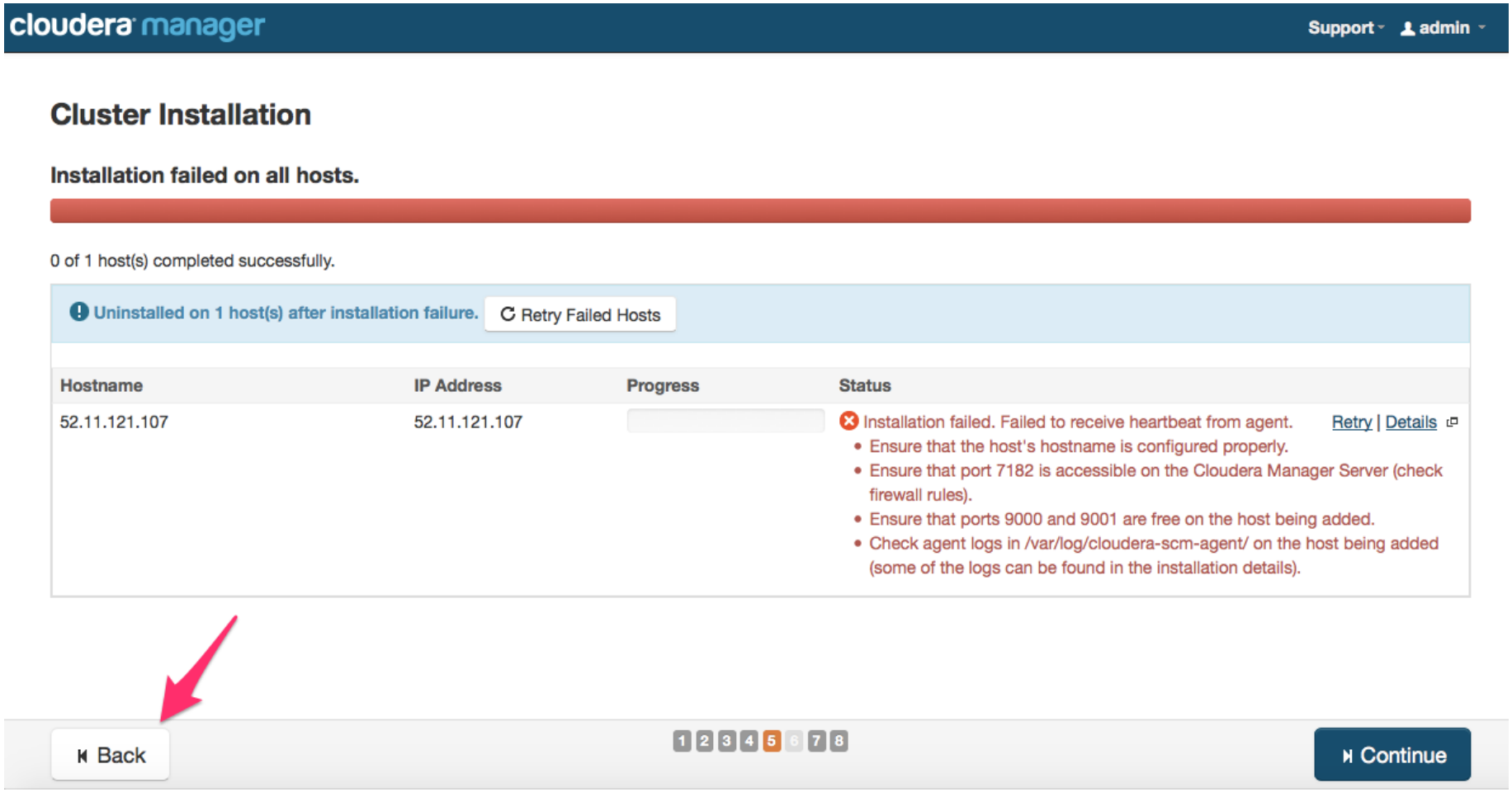
Browse the private key (imchadoop.pem) file which we have downloaded in the previous part. Keep Passphrase as blank

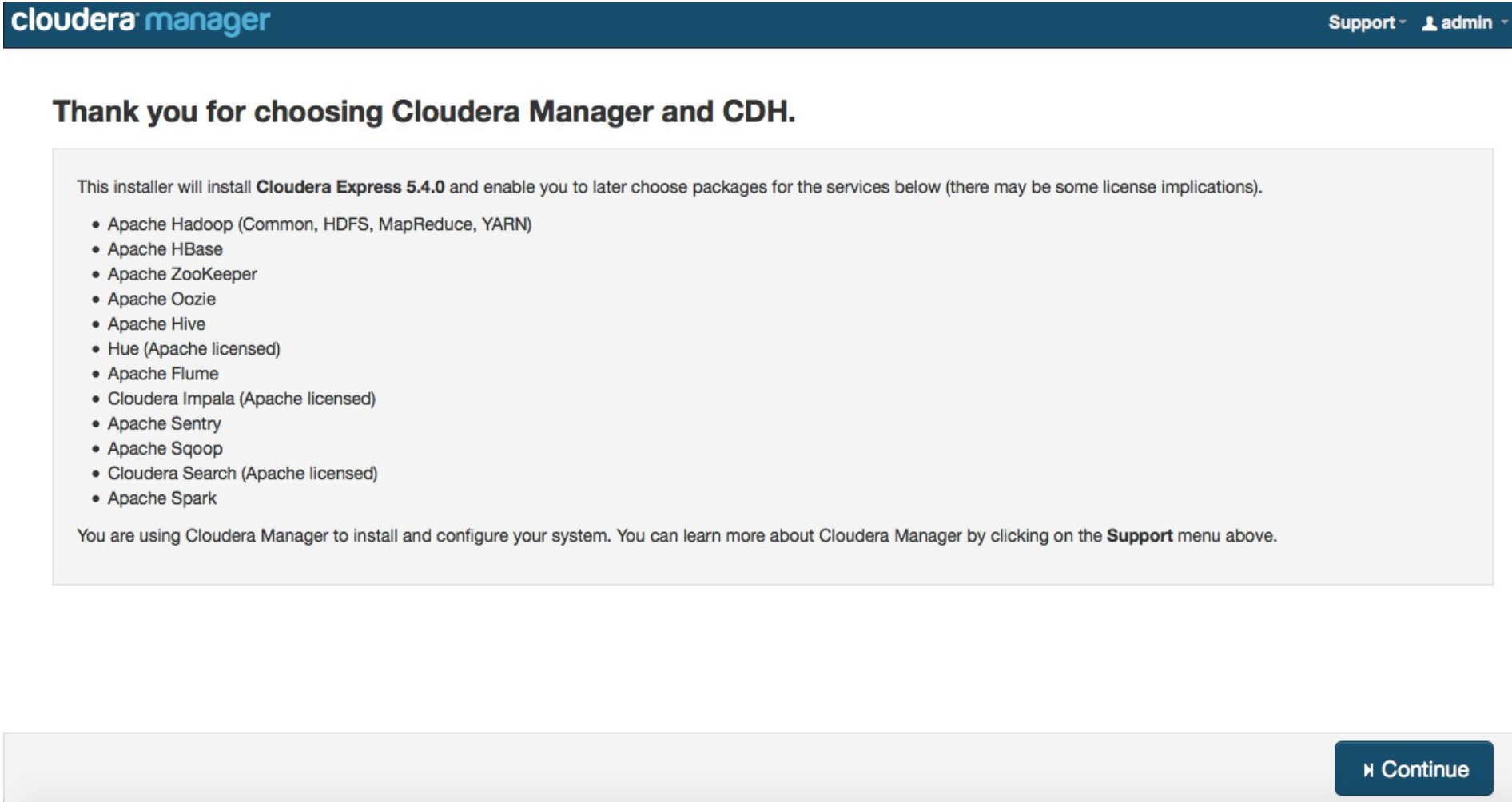



cloudera manager
Support
admin
Cluster Installation
Installation in progress.
0 of 1 host(s) completed successfully.
Abort Installation
Hostname
IP Address
Progress
Status
52.11.121.107
52.11.121.107
Installing oracle-j2sdk1.6 package...
Details

If you see the above error, DO NOT worry at all, it's known issue. You can find the known issue list at Cloudera Issue List.

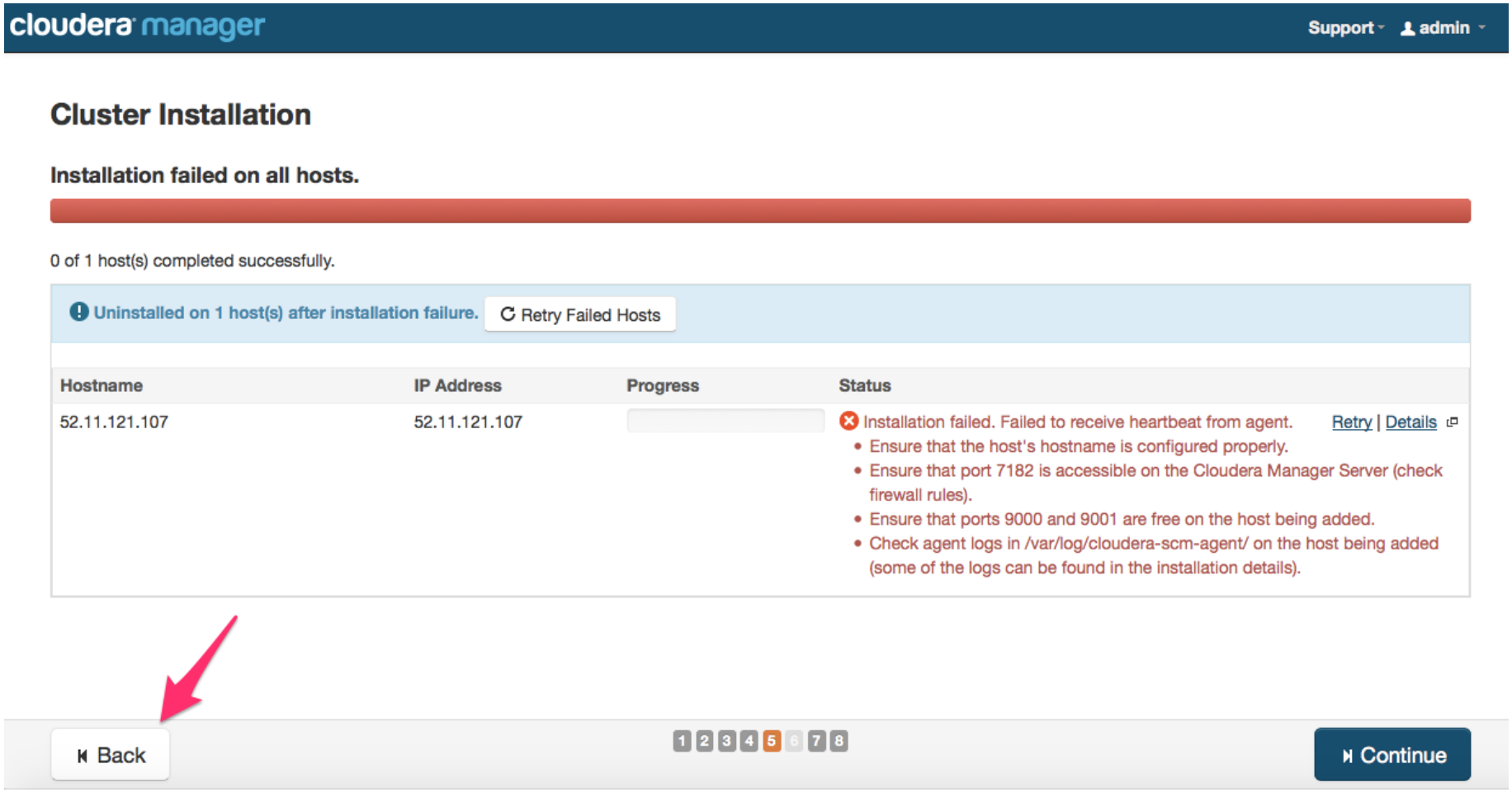
Click "Back" button until home screen then click "Continue" button

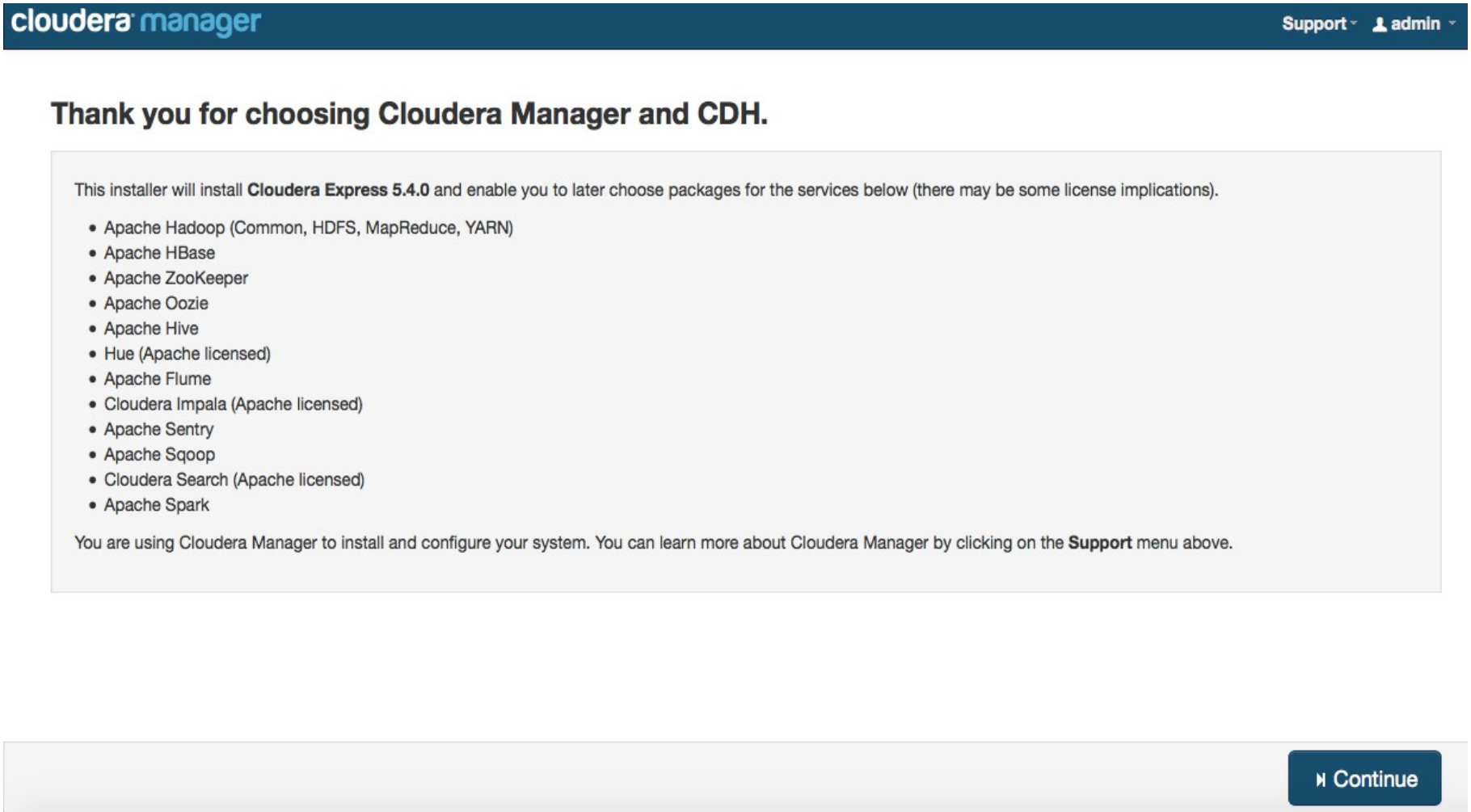



cloudera manager
Support
admin
Thank you for choosing Cloudera Manager and CDH.
This installer will install Cloudera Express 5.4.0 and enable you to later choose packages for the services below (there may be some license implications).
Apache Hadoop (Common, HDFS, MapReduce, YARN)
Apache HBase
Apache 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

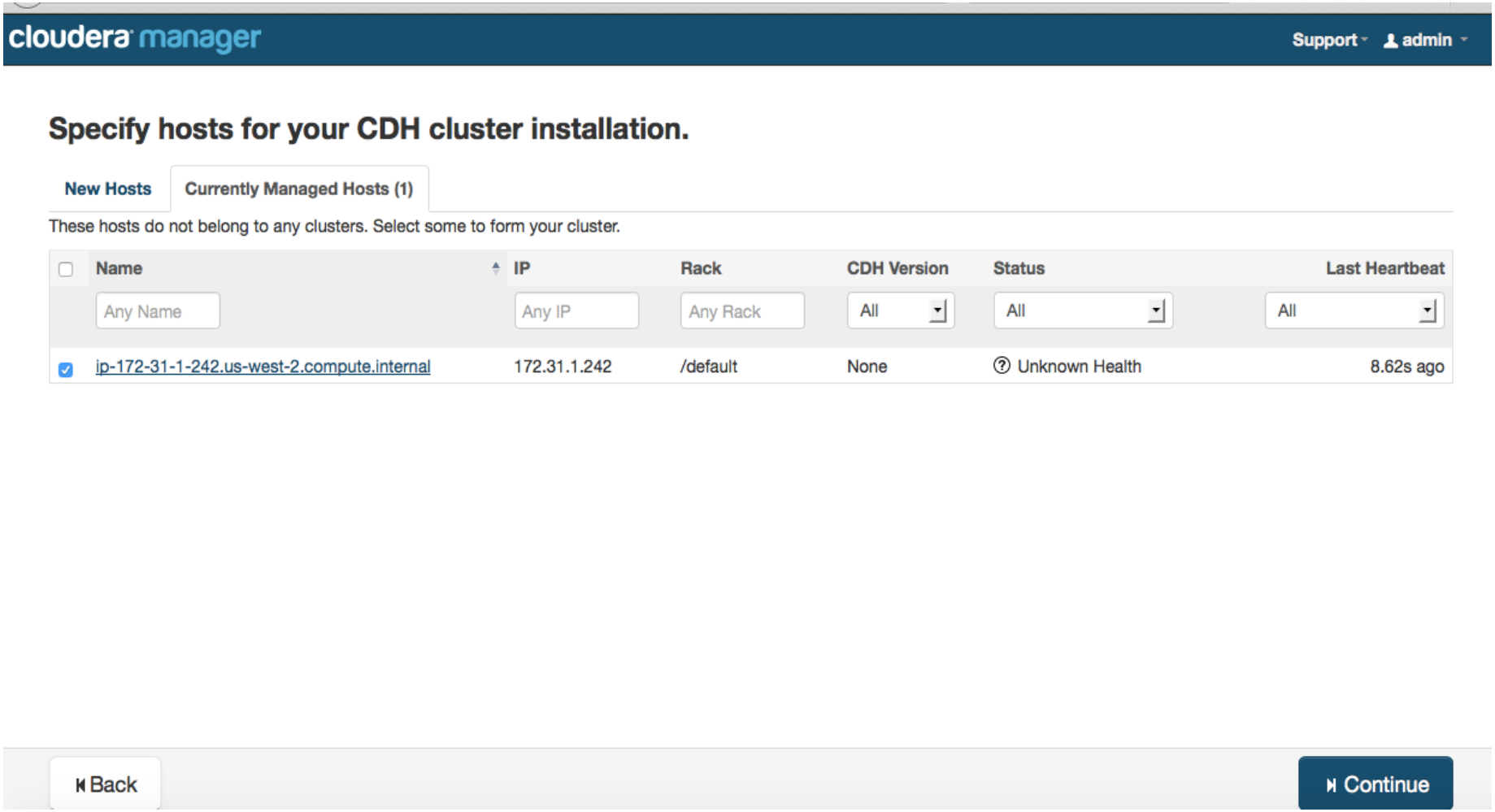
If you see the above error, DO NOT worry at all, it's known issue. You can find the known issue list at Cloudera Issue List.

Click "Back" button until home screen then click "Continue" button




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

Now you will find a tab "Currently Managed Hosts" with their private dns and private ip address. Select all and click "Continue"



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-5.4.0-1.cdh5.4.0.p0.27
- ○ CDH-4.7.1-1.cdh4.7.1.p0.47

Versions of CDH that are too new for this version of Cloudera Manager (5.4.0) will not be shown.

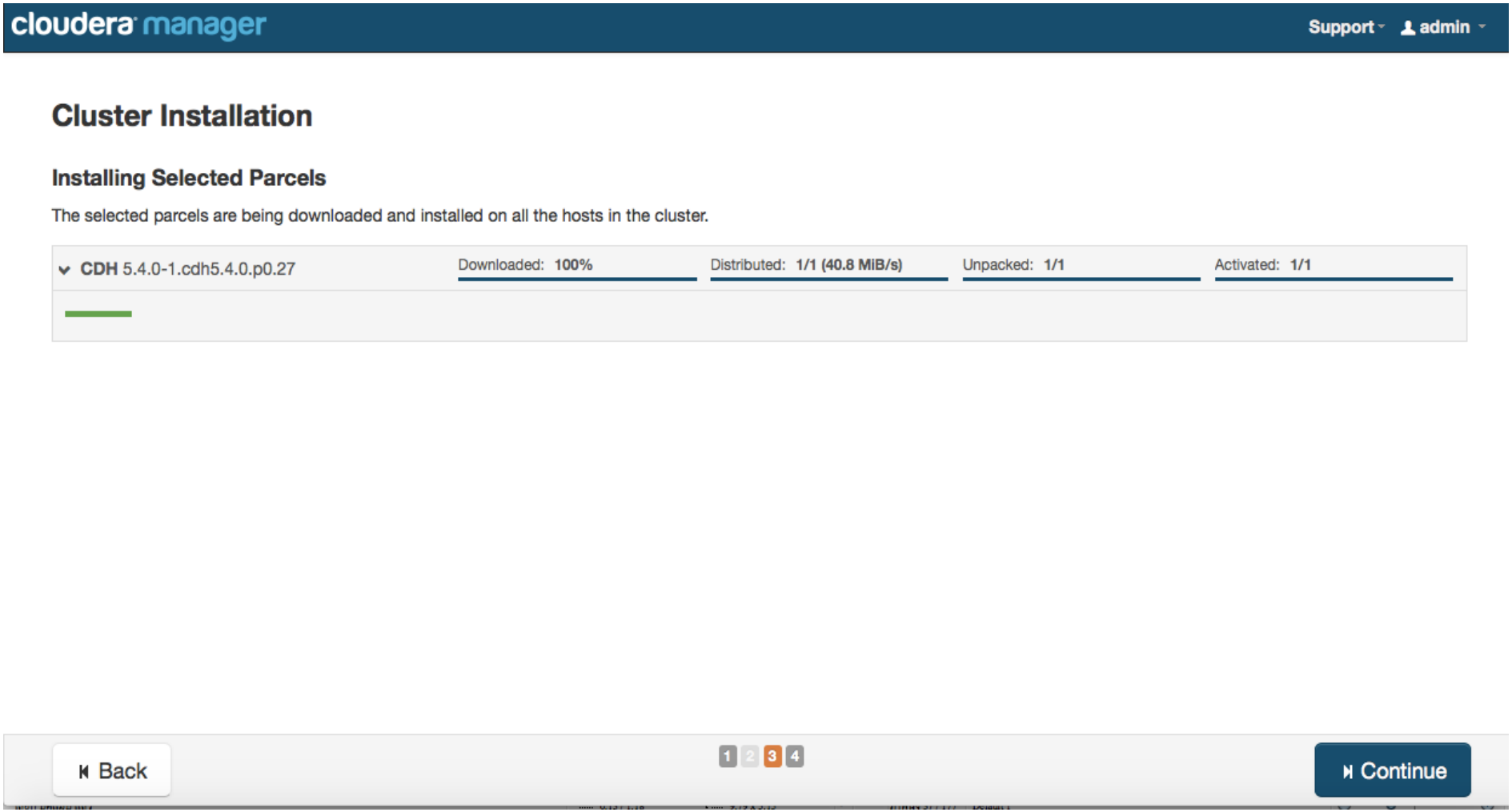
Additional Parcels

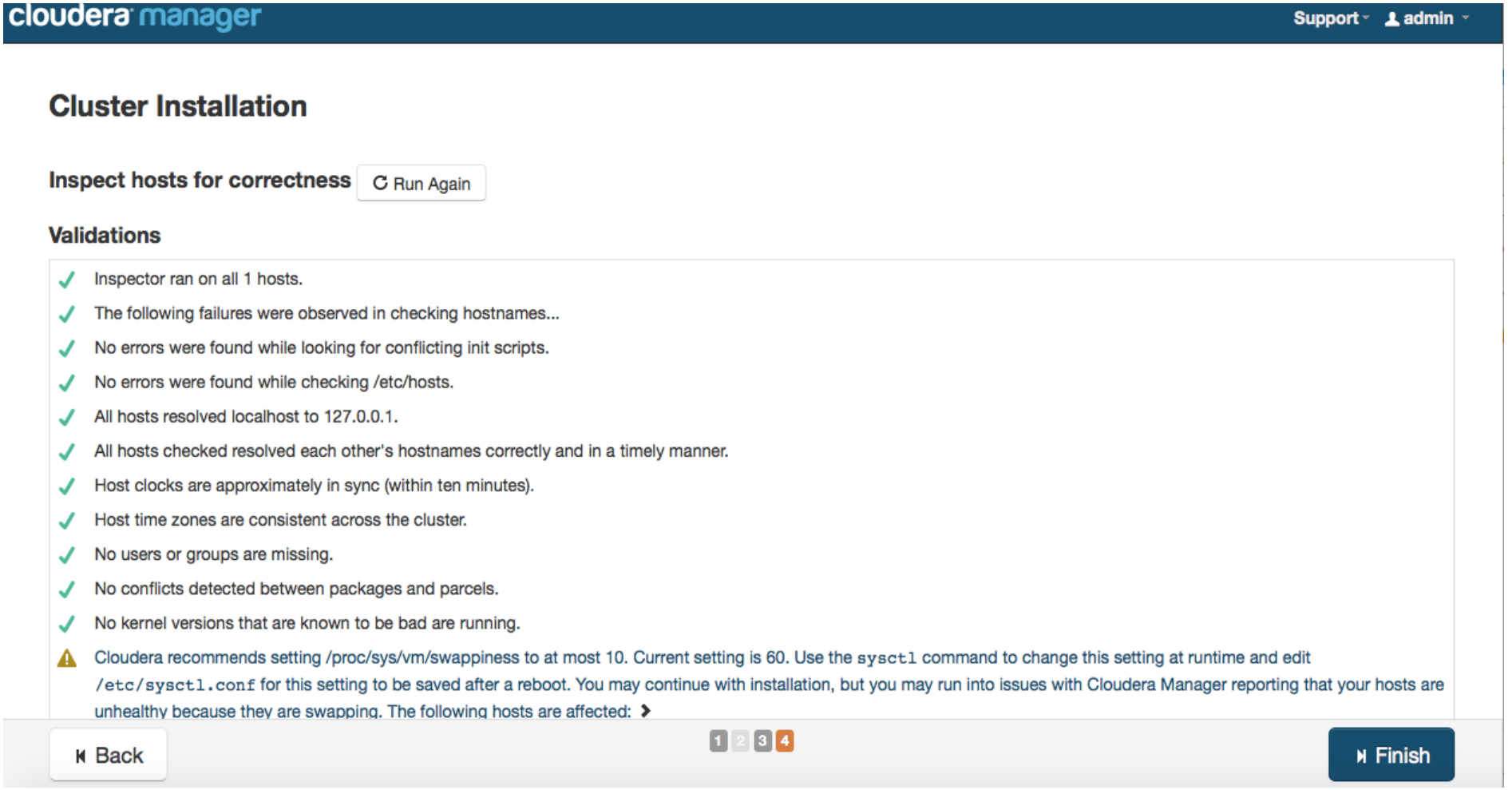
- ○ ACCUMULO-1.6.0-1.cdh5.1.4.p0.116
- ○ ACCUMULO-1.4.4-1.cdh4.5.0.p0.65
- ◉ None

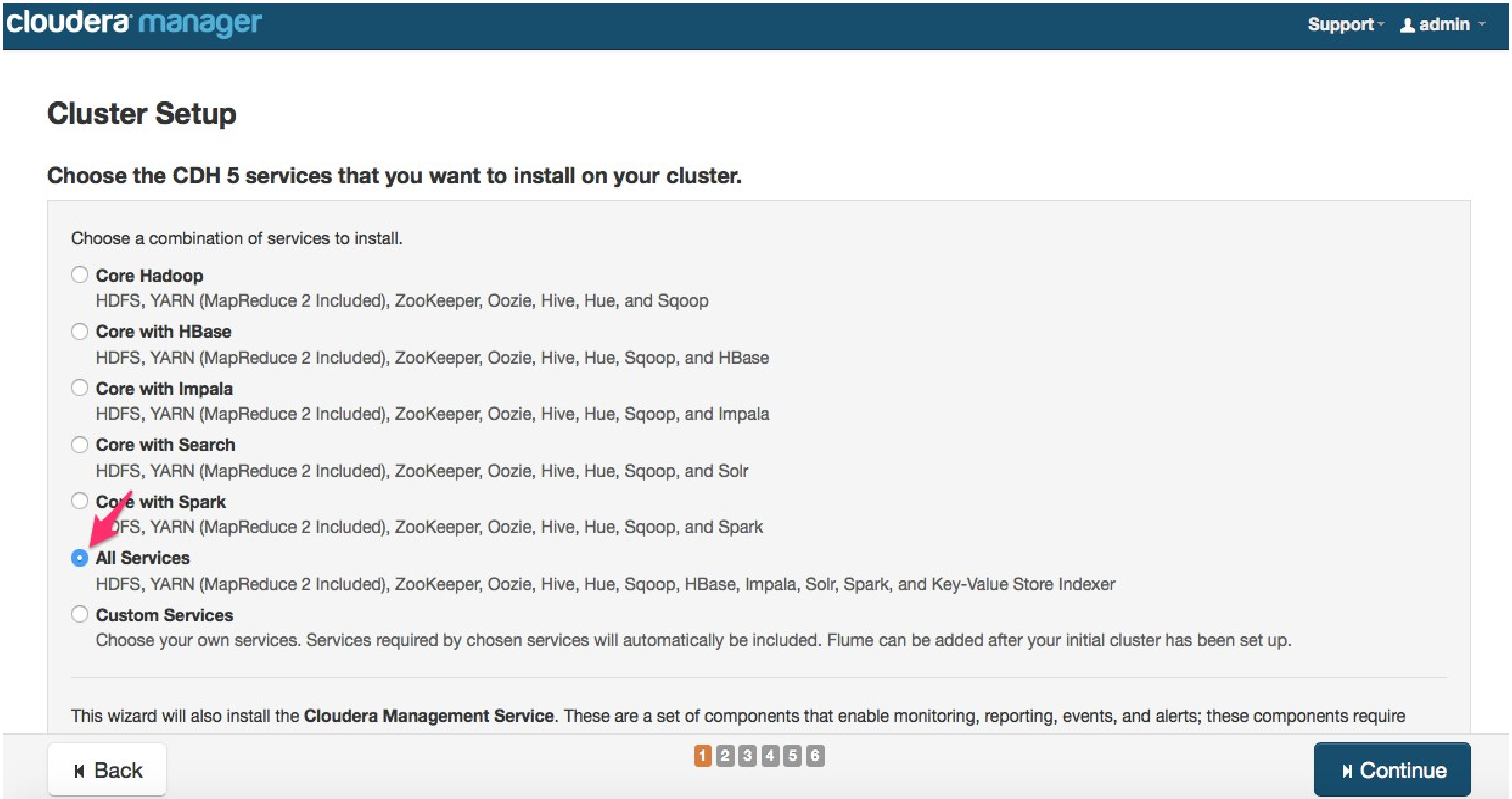
- ○ KAFKA-0.8.2.0-1.kafka1.3.0.p0.29
- ◉ None

- ○ KEYTRUSTEE-5.4.0-1.cdh5.4.0.p0.193

Back 1 2 3 4 Continue


cloudera manager
Support
admin
Cluster Installation
Installing Selected Parcels
The selected parcels are being downloaded and installed on all the hosts in the cluster.
CDH 5.4.0-1.cdh5.4.0.p0.27
Downloaded: 100%
Distributed: 1/1 (40.8 MiB/s)
Unpacked: 1/1
Activated: 1/1
Back
1 2 3 4
Continue


cloudera manager
Support
admin
Cluster Installation
Inspect hosts for correctness
Run Again
Validations
Inspector ran on all 1 hosts.
The following failures were observed in checking hostnames...
No errors were found while looking for conflicting init scripts.
No errors were found while checking /etc/hosts.
All hosts resolved localhost to 127.0.0.1.
All hosts checked resolved each other's hostnames correctly and in a timely manner.
Host clocks are approximately in sync (within ten minutes).
Host time zones are consistent across the cluster.
No users or groups are missing.
No conflicts detected between packages and parcels.
No kernel versions that are known to be bad are running.
Cloudera recommends setting /proc/sys/vm/swappiness to at most 10. Current setting is 60. Use the sysctl command to change this setting at runtime and edit /etc/sysctl.conf for this setting to be saved after a reboot. You may continue with installation, but you may run into issues with Cloudera Manager reporting that your hosts are unhealthy because they are swapping. The following hosts are affected:
Back
1 2 3 4
Finish


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), ZooKeeper, Oozie, Hive, Hue, Sqoop, and HBase
Core with Impala
HDFS, YARN (MapReduce 2 Included), ZooKeeper, Oozie, Hive, Hue, Sqoop, and Impala
Core with Search
HDFS, YARN (MapReduce 2 Included), ZooKeeper, Oozie, Hive, Hue, Sqoop, and Solr
Core with Spark
HDFS, YARN (MapReduce 2 Included), ZooKeeper, Oozie, Hive, Hue, Sqoop, and Spark
All Services
HDFS, YARN (MapReduce 2 Included), ZooKeeper, Oozie, Hive, Hue, Sqoop, HBase, Impala, Solr, Spark, and Key-Value Store Indexer
Custom Services
Choose your own services. Services required by chosen services will automatically be included. Flume can be added after your initial cluster has been set up.
This wizard will also install the Cloudera Management Service. These are a set of components that enable monitoring, reporting, events, and alerts; these components require
Back
1 2 3 4 5 6
Continue

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

M Master × 1 New	HBRES HBase REST Server	HBTS HBase Thrift Server	RS RegionServer × 1 New
Same As DataNode	Select hosts	Select hosts	Same As DataNode

HDFS

NN NameNode × 1 New	SNN SecondaryNameNode × 1 New	B Balancer × 1 New	HFS HttpFS
Same As DataNode	Same As DataNode	Same As DataNode	Select hosts

NFSG NFS Gateway	DN DataNode × 1 New
Select hosts	ip-172-31-1-242.us-west-2.compute.interna

Hive

Back 1 2 3 4 5 6 Continue

Cluster Setup

Database Setup

Configure and test database connections. If using custom databases, create the databases first according to the **Installing and Configuring an External Database** section of the Installation Guide .

- ○ **Use Custom Databases**
- ◉ **Use Embedded Database**

When using the embedded database, passwords are automatically generated. Please copy them down.

Hive

Database Host Name:	Database Type:	Database Name :	Username:	Password:
ip-172-31-1-242.us-west-2.compute.interna	PostgreSQL	hive	hive	bV6sUA8gPH

Oozie Server

Currently assigned to run on **ip-172-31-1-242.us-west-2.compute.internal**.

Database Host Name:	Database Type:	Database Name :	Username:	Password:
ip-172-31-1-242.us-west-2.compute.interna	PostgreSQL	oozie_oozie_se	oozie_oozie_se	6MvnYMQkTE

Test Connection

Cluster Setup

Database Setup

Configure and test database connections. If using custom databases, create the databases first according to the **Installing and Configuring an External Database** section of the Installation Guide.

- ○ **Use Custom Databases**
- ◉ **Use Embedded Database**

When using the embedded database, passwords are automatically generated. Please copy them down.

Hive

✓ **Skipped. Cloudera Manager will create this database in a later step.**

Database Host Name:	Database Type:	Database Name :	Username:	Password:
ip-172-31-1-242.us-west-2.compute.interna	PostgreSQL	hive	hive	bV6sUA8gPH

Oozie Server

✓ **Skipped. Cloudera Manager will create this database in a later step.**

Currently assigned to run on **ip-172-31-1-242.us-west-2.compute.internal**.

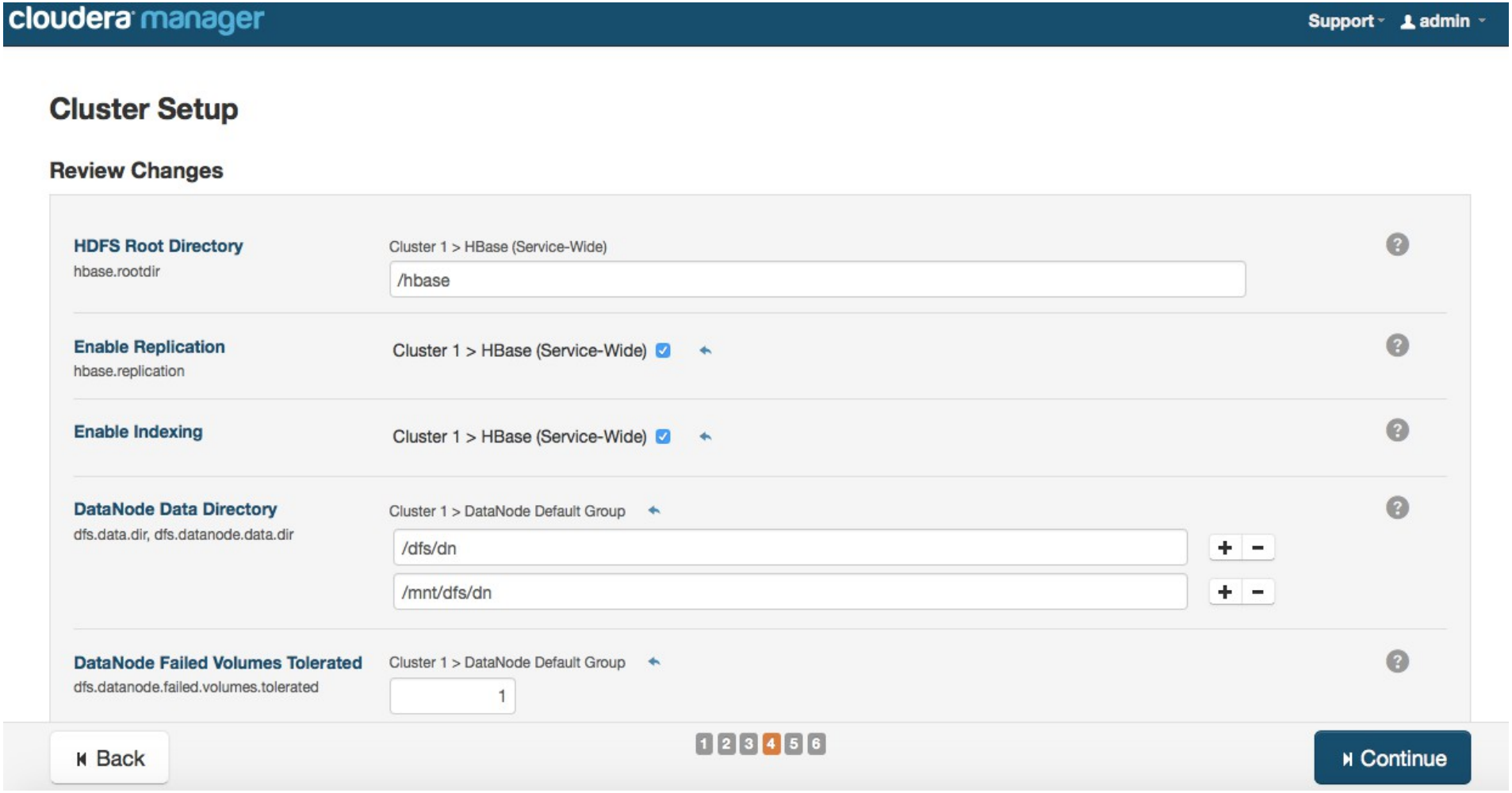
Database Host Name:	Database Type:	Database Name :	Username:	Password:
ip-172-31-1-242.us-west-2.compute.interna	PostgreSQL	oozie_oozie_se	oozie_oozie_se	6MvnYMQkTE

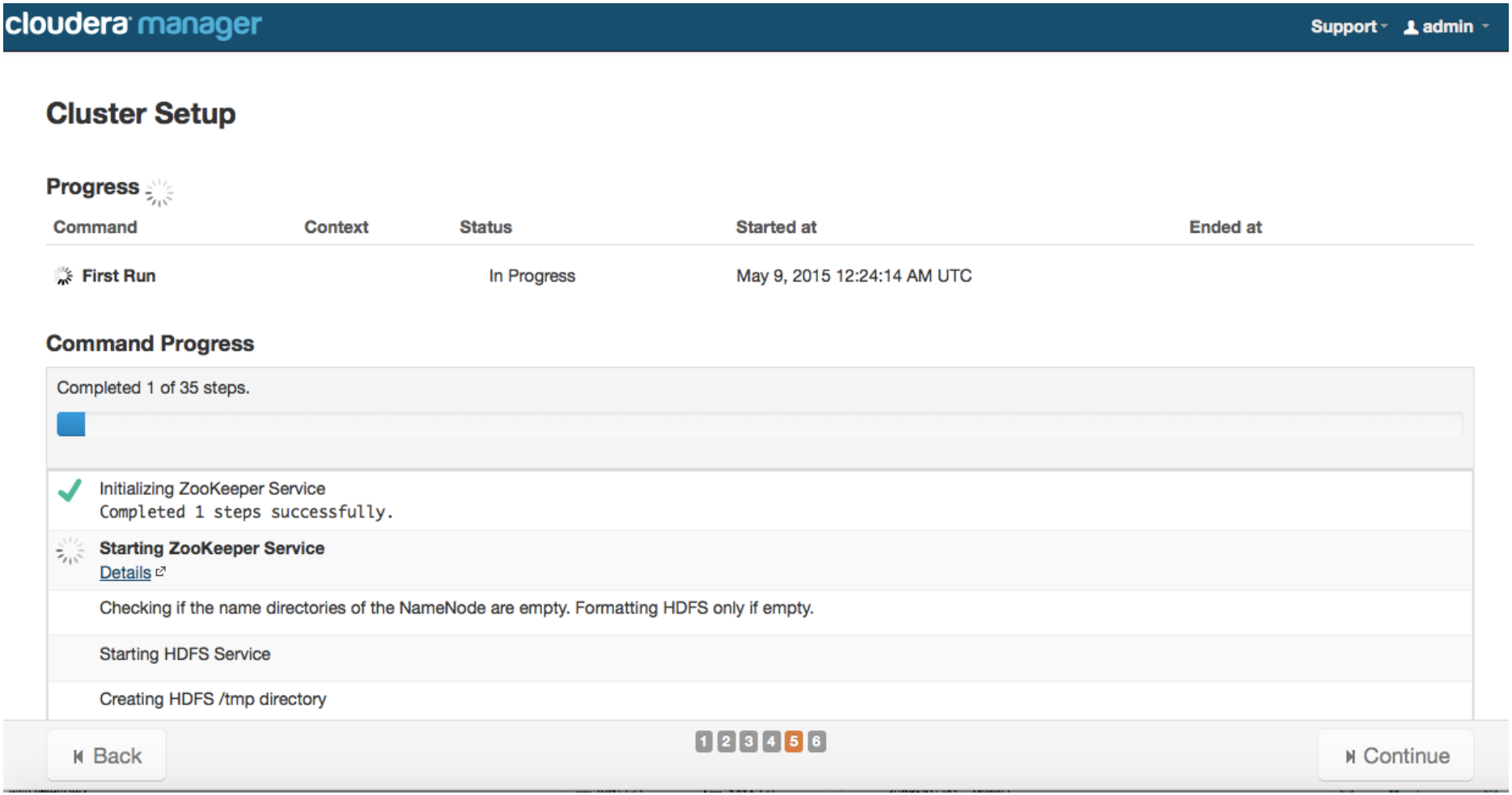
Test Connection

◀ Back

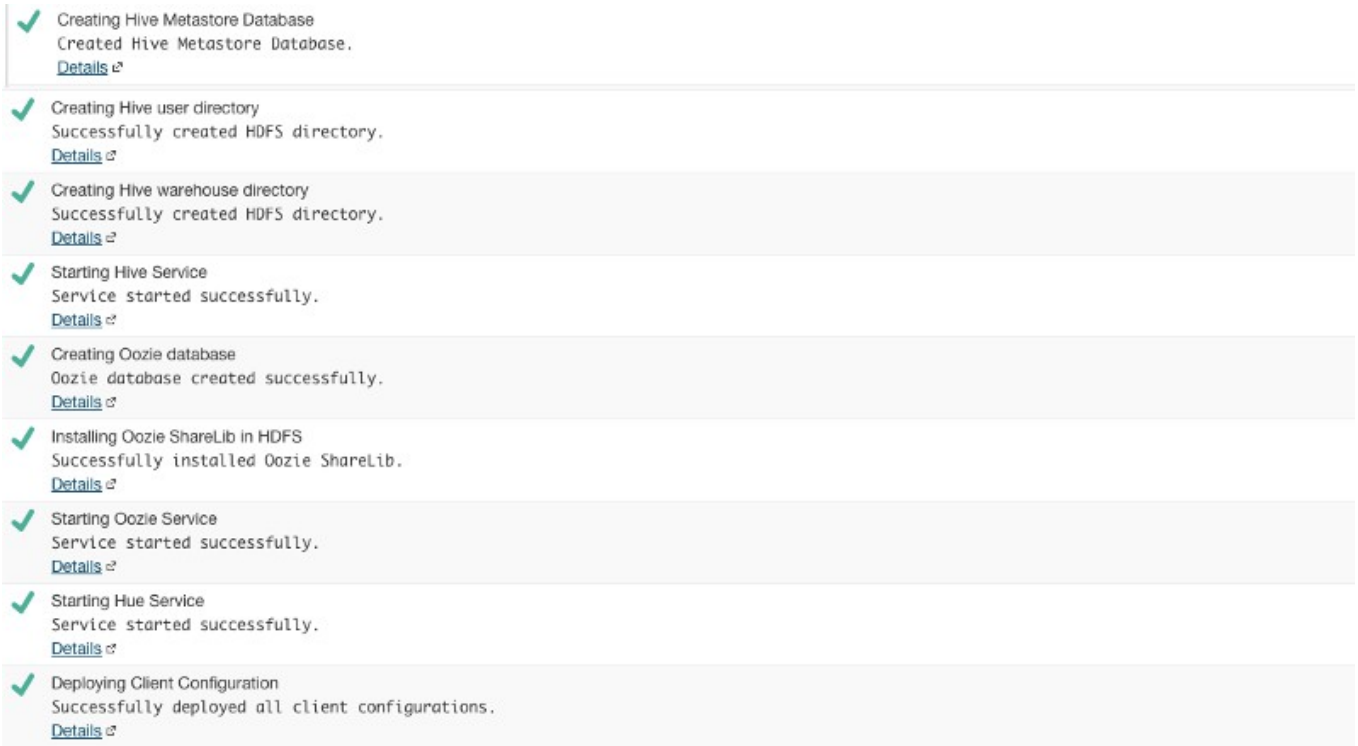
1 2 3 4 5 6

Continue ▶

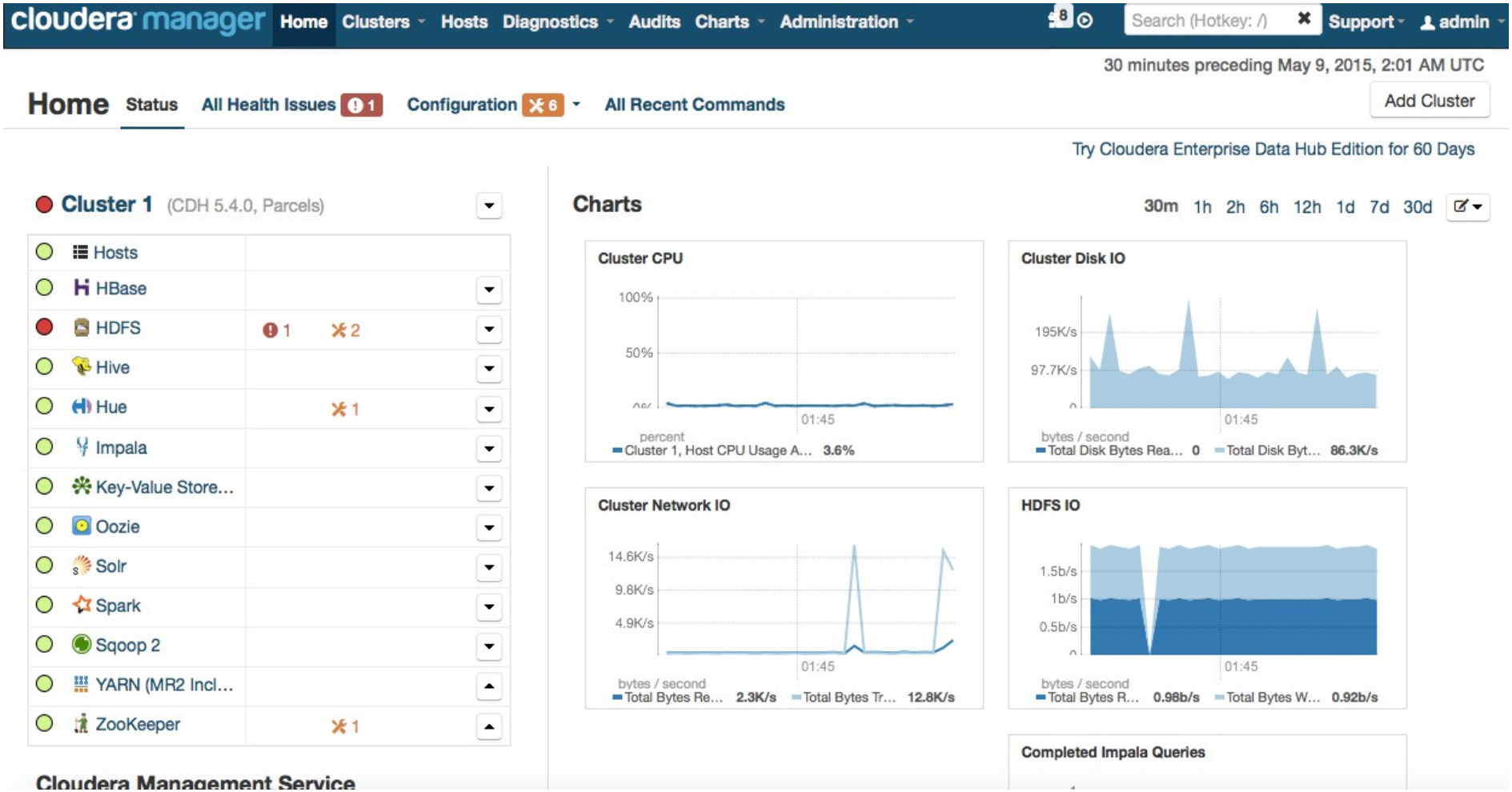

cloudera manager
Support
admin
Cluster Setup
Review Changes
HDFS Root Directory
hbase.rootdir
Cluster 1 > HBase (Service-Wide)
/hbase
Enable Replication
hbase.replication
Cluster 1 > HBase (Service-Wide)
Enable Indexing
Cluster 1 > HBase (Service-Wide)
DataNode Data Directory
dfs.data.dir, dfs.datanode.data.dir
Cluster 1 > DataNode Default Group
/dfs/dn
/mnt/dfs/dn
DataNode Failed Volumes Tolerated
dfs.datanode.failed.volumes.tolerated
Cluster 1 > DataNode Default Group
1
Back
1 2 3 4 5 6
Continue


cloudera manager
Support
admin
Cluster Setup
Progress
Command
Context
Status
Started at
Ended at
First Run
In Progress
May 9, 2015 12:24:14 AM UTC
Command Progress
Completed 1 of 35 steps.
Initializing ZooKeeper Service
Completed 1 steps successfully.
Starting ZooKeeper Service
Details
Checking if the name directories of the NameNode are empty. Formatting HDFS only if empty.
Starting HDFS Service
Creating HDFS /tmp directory
Back
1 2 3 4 5 6
Continue

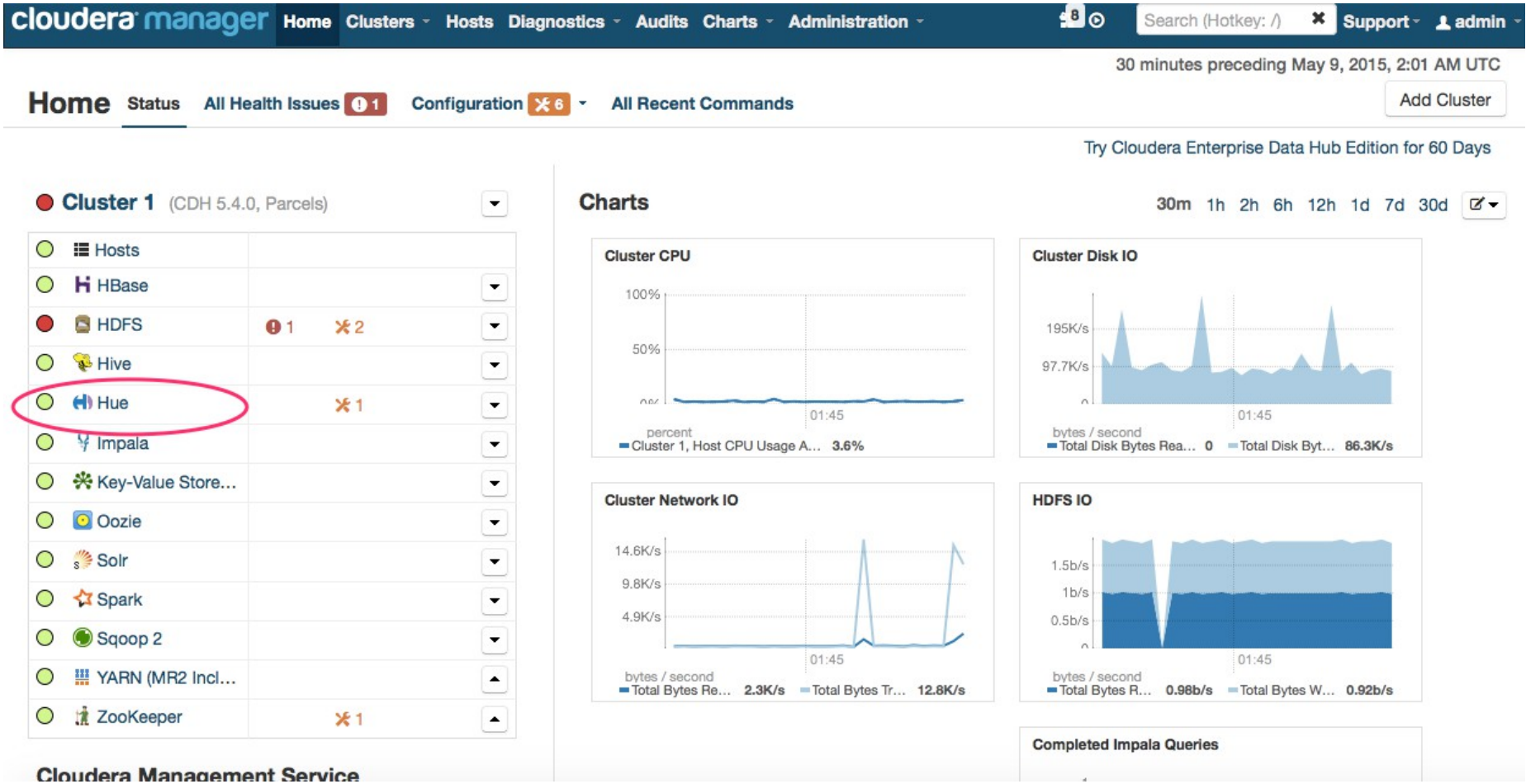
Finish


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

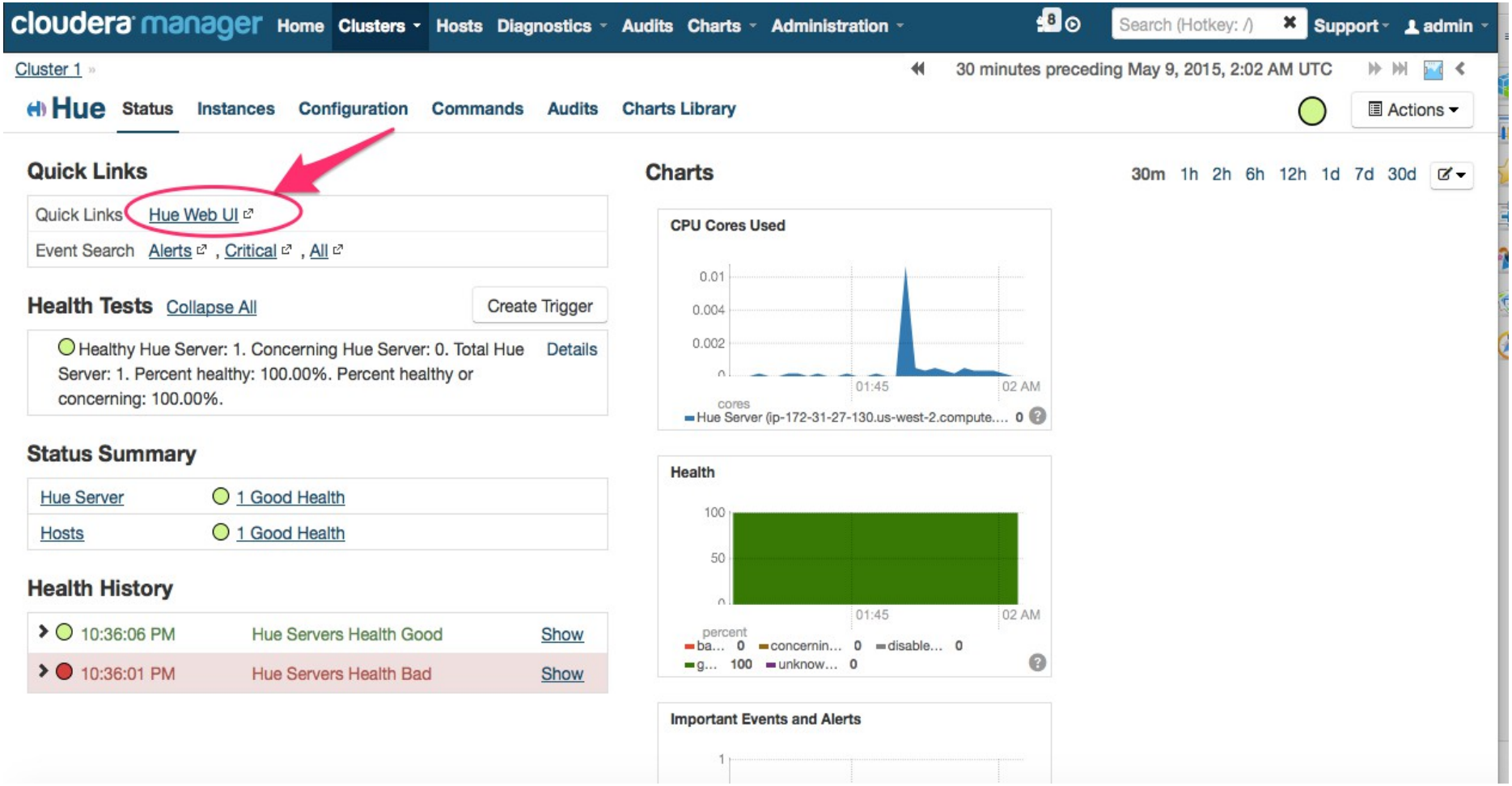

Cluster Setup
Congratulations!
The services are installed, configured, and running on your cluster.


cloudera manager
Home
Clusters
Hosts
Diagnostics
Audits
Charts
Administration
8
Search (Hotkey: /)
Support
admin
30 minutes preceding May 9, 2015, 2:01 AM UTC
Home
Status
All Health Issues 1
Configuration 6
All Recent Commands
Add Cluster
Try Cloudera Enterprise Data Hub Edition for 60 Days
Cluster 1 (CDH 5.4.0, Parcels)
Hosts
HBase
HDFS 1 2
Hive
Hue 1
Impala
Key-Value Store...
Oozie
Solr
Spark
Sqoop 2
YARN (MR2 Incl...
ZooKeeper 1
Cloudera Management Service
Charts
30m 1h 2h 6h 12h 1d 7d 30d
Cluster CPU
100%
50%
01:45
percent
Cluster 1, Host CPU Usage A... 3.6%
Cluster Disk IO
195K/s
97.7K/s
01:45
bytes / second
Total Disk Bytes Rea... 0
Total Disk Byt... 86.3K/s
Cluster Network IO
14.6K/s
9.8K/s
4.9K/s
01:45
bytes / second
Total Bytes Re... 2.3K/s
Total Bytes Tr... 12.8K/s
HDFS IO
1.5b/s
1b/s
0.5b/s
01:45
bytes / second
Total Bytes R... 0.98b/s
Total Bytes W... 0.92b/s
Completed Impala Queries

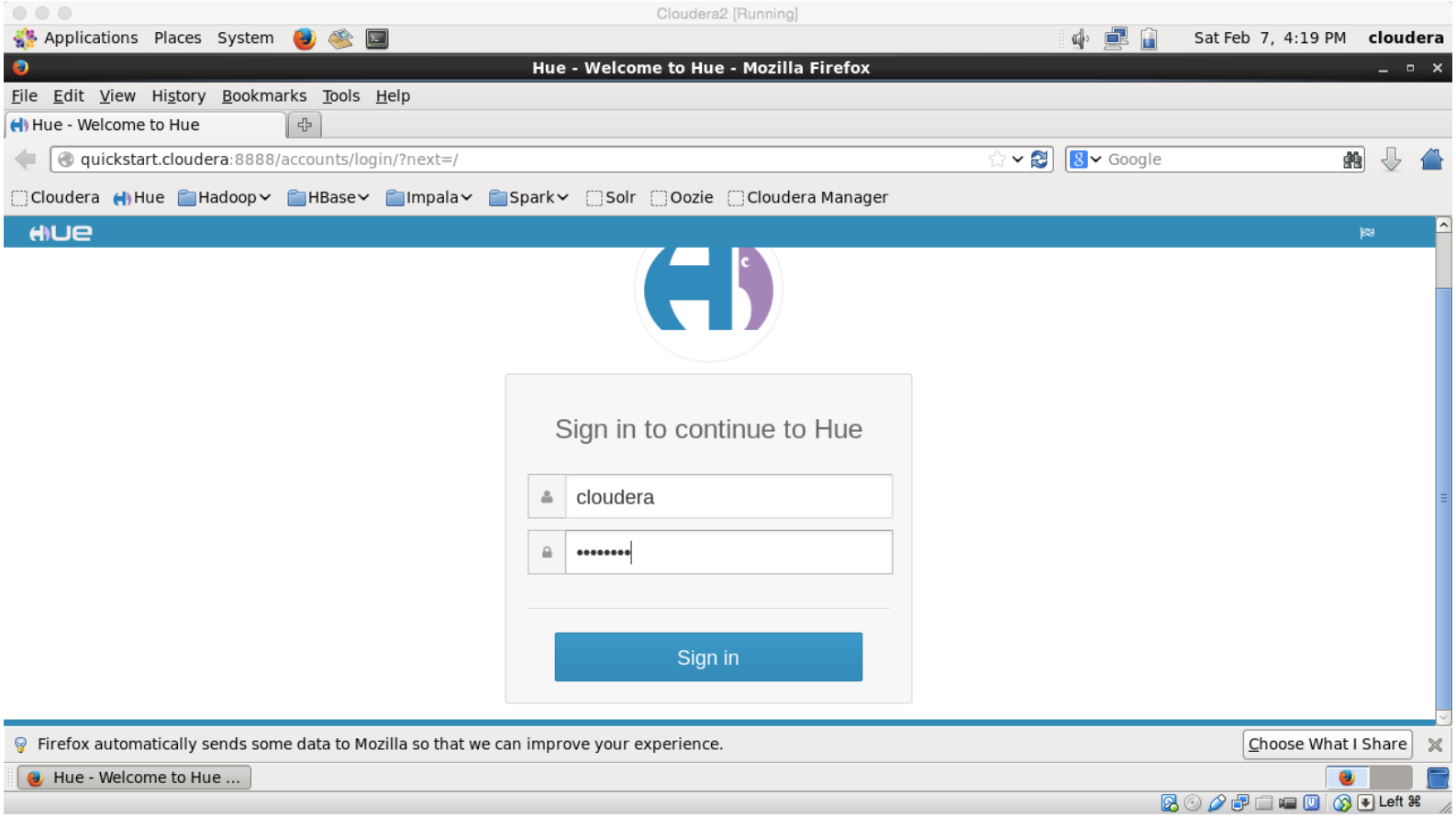
Running Hue


cloudera manager
Home
Clusters
Hosts
Diagnostics
Audits
Charts
Administration
Search (Hotkey: /)
Support
admin
30 minutes preceding May 9, 2015, 2:01 AM UTC
Home
Status
All Health Issues 1
Configuration 6
All Recent Commands
Add Cluster
Try Cloudera Enterprise Data Hub Edition for 60 Days
Cluster 1 (CDH 5.4.0, Parcels)
Hosts
HBase
HDFS 1 2
Hive
Hue 1
Impala
Key-Value Store...
Oozie
Solr
Spark
Sqoop 2
YARN (MR2 Incl...
ZooKeeper 1
Cloudera Management Service
Charts
30m 1h 2h 6h 12h 1d 7d 30d
Cluster CPU
100%
50%
01:45
percent
Cluster 1, Host CPU Usage A... 3.6%
Cluster Disk IO
195K/s
97.7K/s
01:45
bytes / second
Total Disk Bytes Rea... 0
Total Disk Byt... 86.3K/s
Cluster Network IO
14.6K/s
9.8K/s
4.9K/s
01:45
bytes / second
Total Bytes Re... 2.3K/s
Total Bytes Tr... 12.8K/s
HDFS IO
1.5b/s
1b/s
0.5b/s
01:45
bytes / second
Total Bytes R... 0.98b/s
Total Bytes W... 0.92b/s
Completed Impala Queries

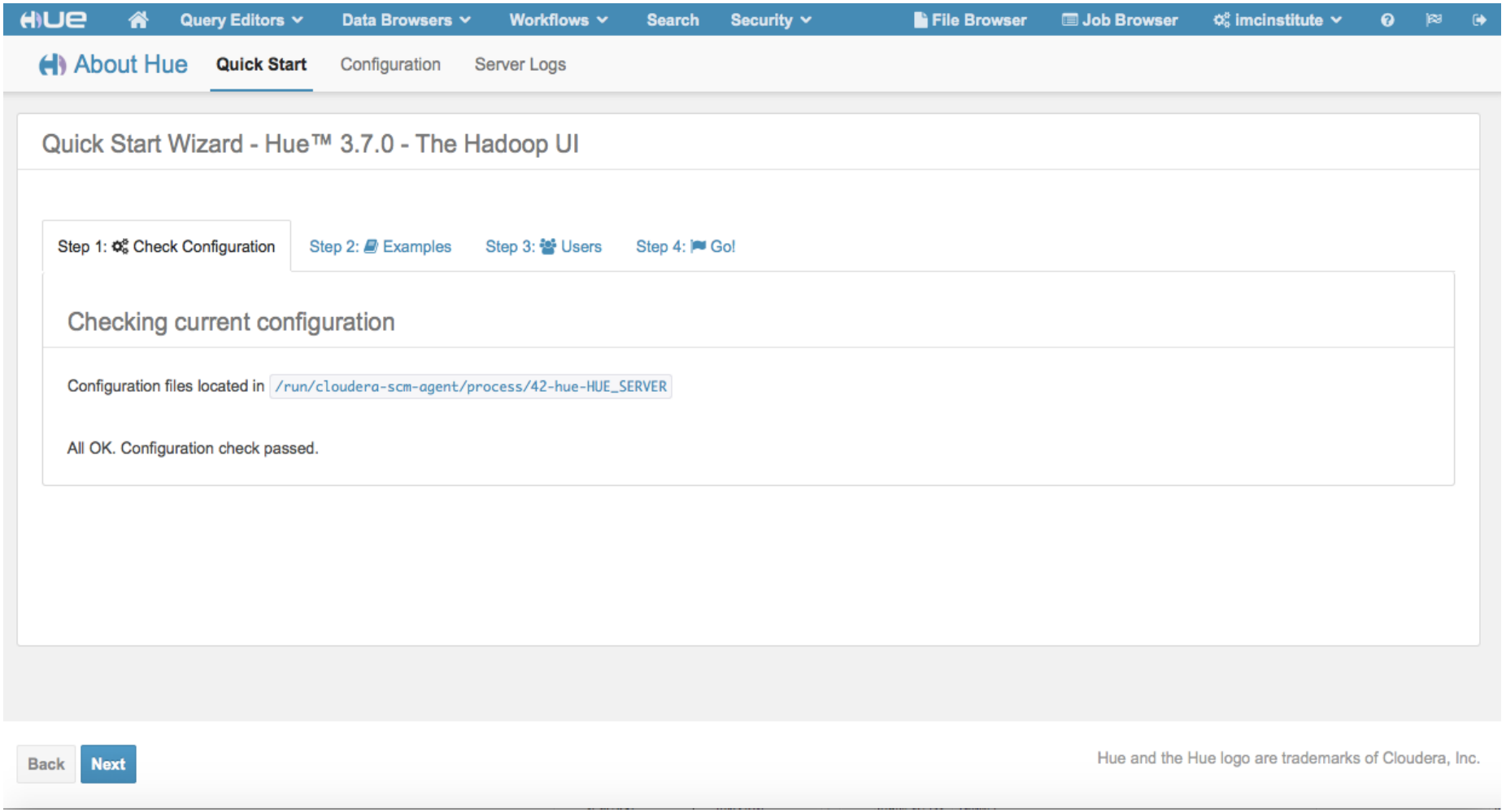
Running Hue

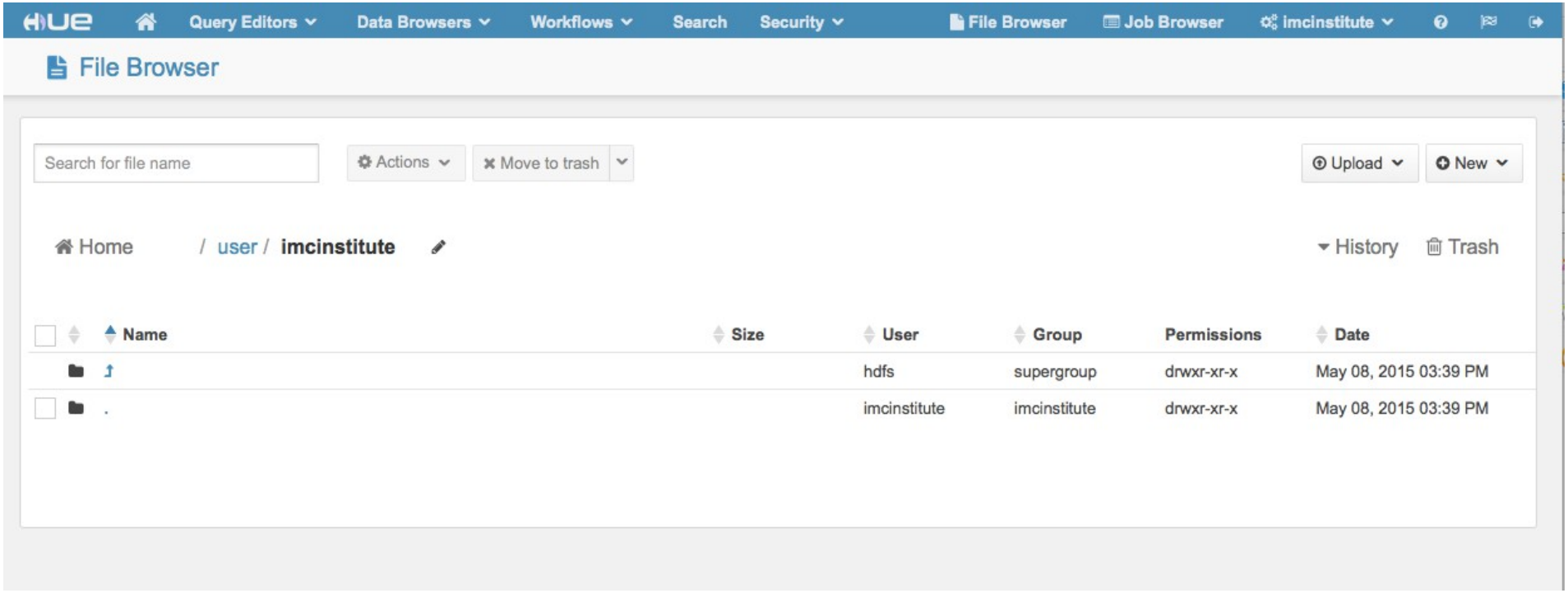

cloudera manager
Home
Clusters
Hosts
Diagnostics
Audits
Charts
Administration
Search (Hotkey: /)
Support
admin
Cluster 1
30 minutes preceding May 9, 2015, 2:02 AM UTC
Hue
Status
Instances
Configuration
Commands
Audits
Charts Library
Actions
Quick Links
Quick Links
Hue Web UI
Event Search
Alerts , Critical , All
Health Tests
Collapse All
Create Trigger
Healthy Hue Server: 1. Concerning Hue Server: 0. Total Hue Server: 1. Percent healthy: 100.00%. Percent healthy or concerning: 100.00%.
Details
Status Summary
Hue Server
1 Good Health
Hosts
1 Good Health
Health History
10:36:06 PM
Hue Servers Health Good
Show
10:36:01 PM
Hue Servers Health Bad
Show
Charts
30m 1h 2h 6h 12h 1d 7d 30d
CPU Cores Used
0.01
0.004
0.002
0
01:45
02 AM
cores
Hue Server (ip-172-31-27-130.us-west-2.compute.... 0
Health
100
50
0
01:45
02 AM
percent
ba... 0
concernin... 0
disable... 0
g... 100
unknow... 0
Important Events and Alerts
1

Sign in to Hue

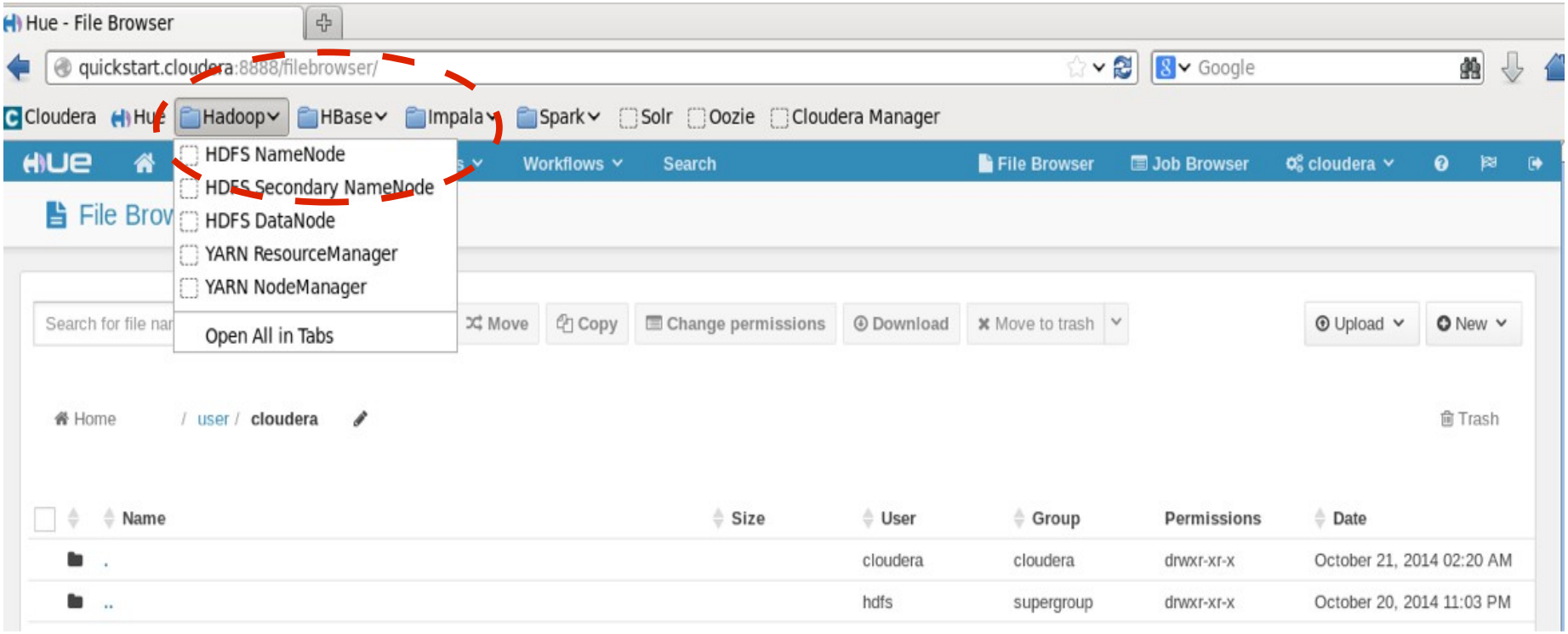


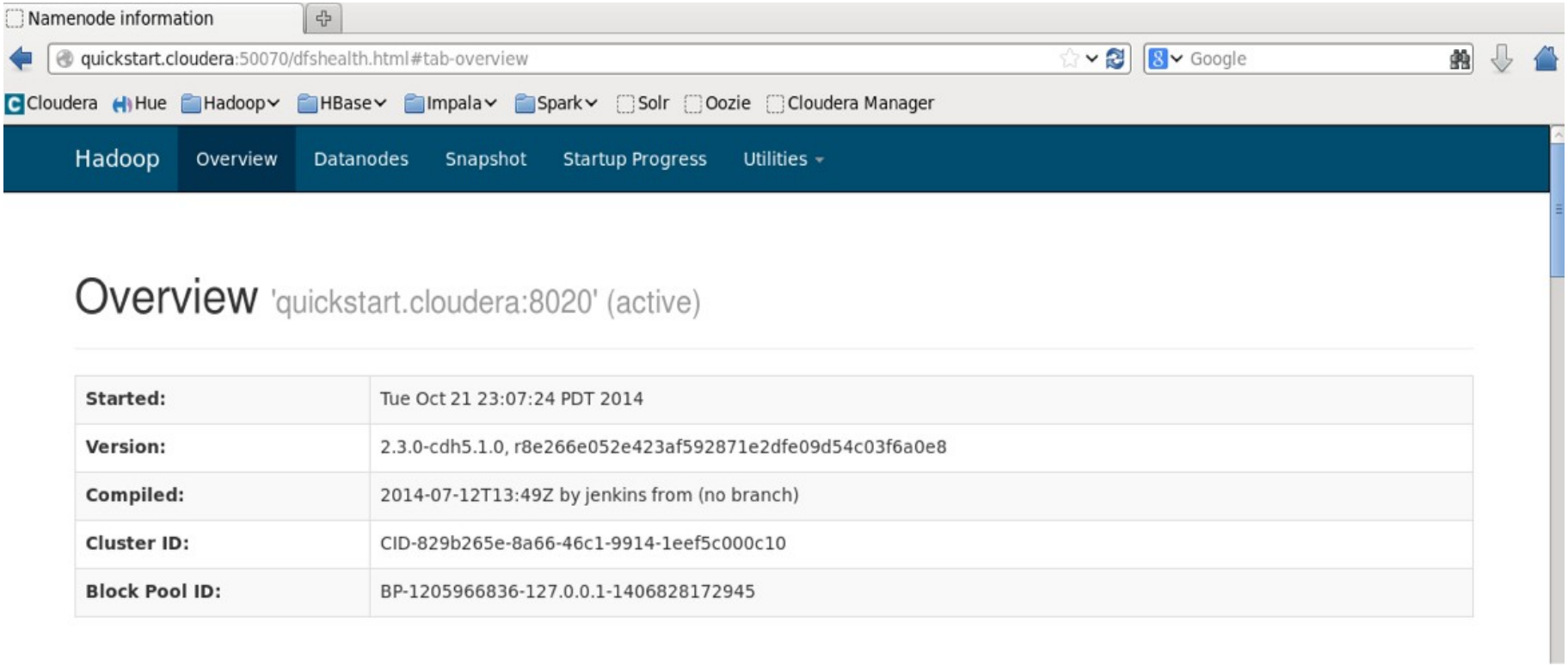
Starting Hue on Cloudera




HUE
Query Editors
Data Browsers
Workflows
Search
Security
File Browser
Job Browser
imcinstitute
File Browser
Search for file name
Actions
Move to trash
Upload
New
Home / user / imcinstitute
History
Trash
Name
Size
User
Group
Permissions
Date
hdfs
supergroup
drwxr-xr-x
May 08, 2015 03:39 PM
.
imcinstitute
imcinstitute
drwxr-xr-x
May 08, 2015 03:39 PM

Viewing HDFS



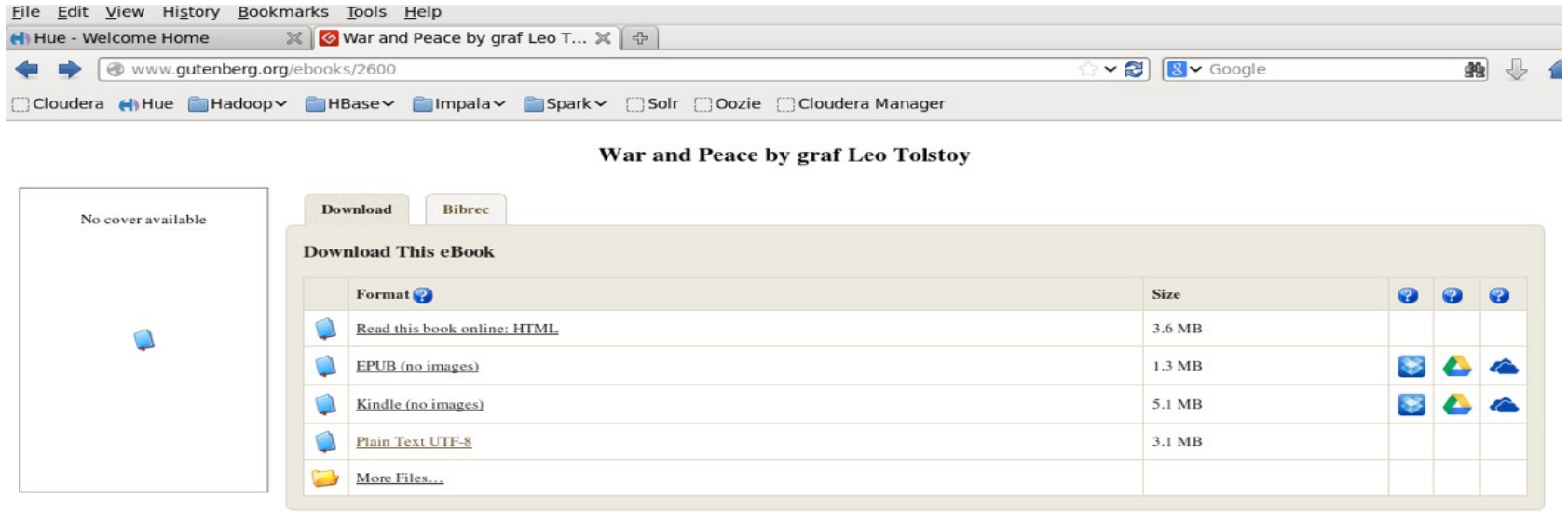

Namenode information
quickstart.cloudera:50070/dfshealth.html#tab-overview
Google
Cloudera
Hue
Hadoop
HBase
Impala
Spark
Solr
Oozie
Cloudera Manager
Hadoop
Overview
Datanodes
Snapshot
Startup Progress
Utilities
Overview 'quickstart.cloudera:8020' (active)
Started: Tue Oct 21 23:07:24 PDT 2014
Version: 2.3.0-cdh5.1.0, r8e266e052e423af592871e2dfe09d54c03f6a0e8
Compiled: 2014-07-12T13:49Z by jenkins from (no branch)
Cluster ID: CID-829b265e-8a66-46c1-9914-1eef5c000c10
Block Pool ID: BP-1205966836-127.0.0.1-1406828172945

Hands-On: Importing/Exporting Data to HDFS

Importing Data to Hadoop

Download War and Peace Full Text

```
www.gutenberg.org/ebooks/2600
```




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

Review file in Hadoop HDFS

List HDFS File

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

Read HDFS File

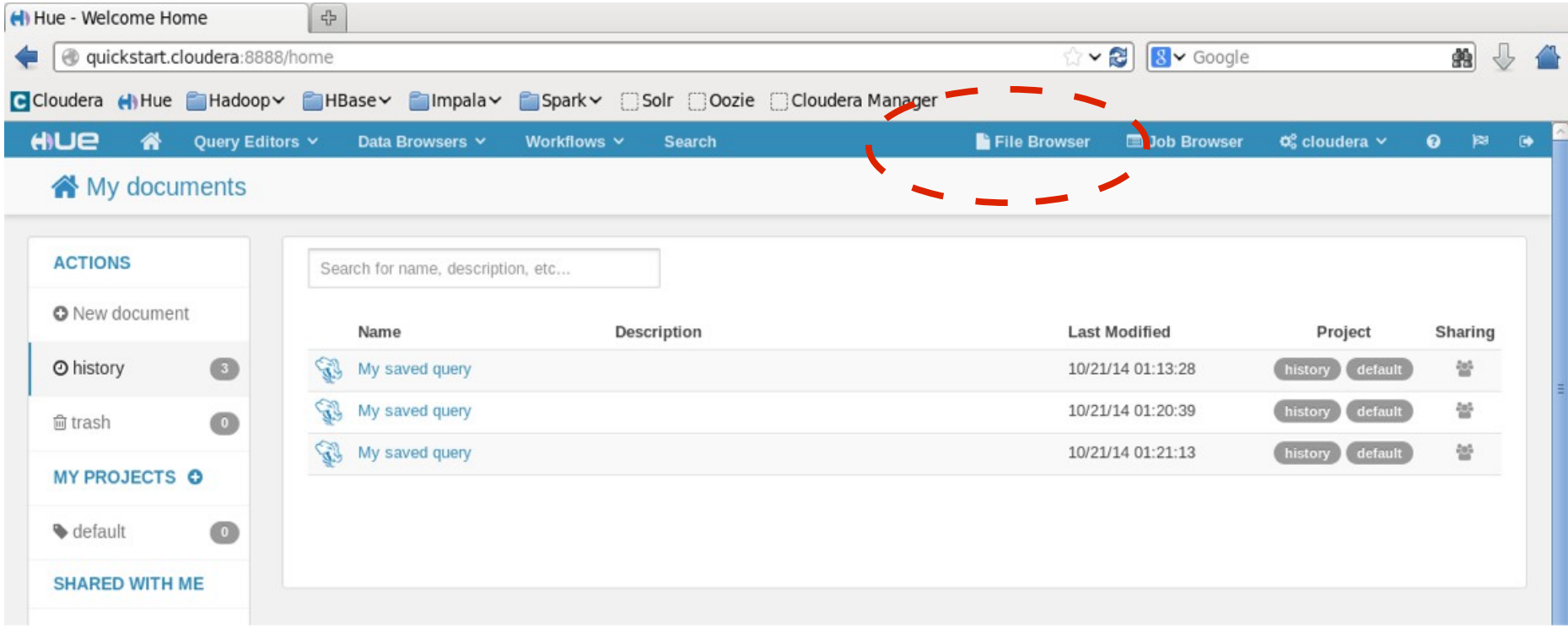
```
[hdadmin@localhost bin]$ hadoop fs -cat input/pg2600.txt
```

Retrieve HDFS File to Local File System

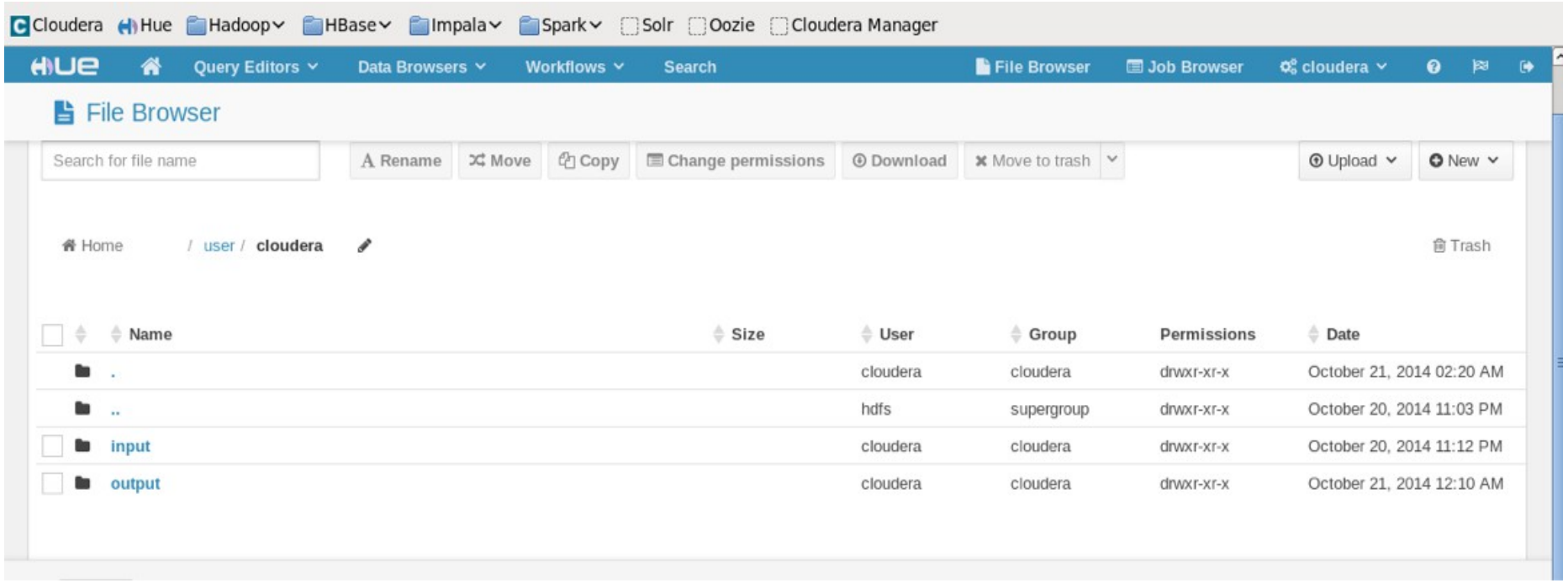
```
[hdadmin@localhost bin]$ hadoop fs -copyToLocal input/pg2600.txt tmp/file.txt
```

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

Review file in Hadoop HDFS using File Browse



Review file in Hadoop HDFS using Hue



Hadoop Port Numbers

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

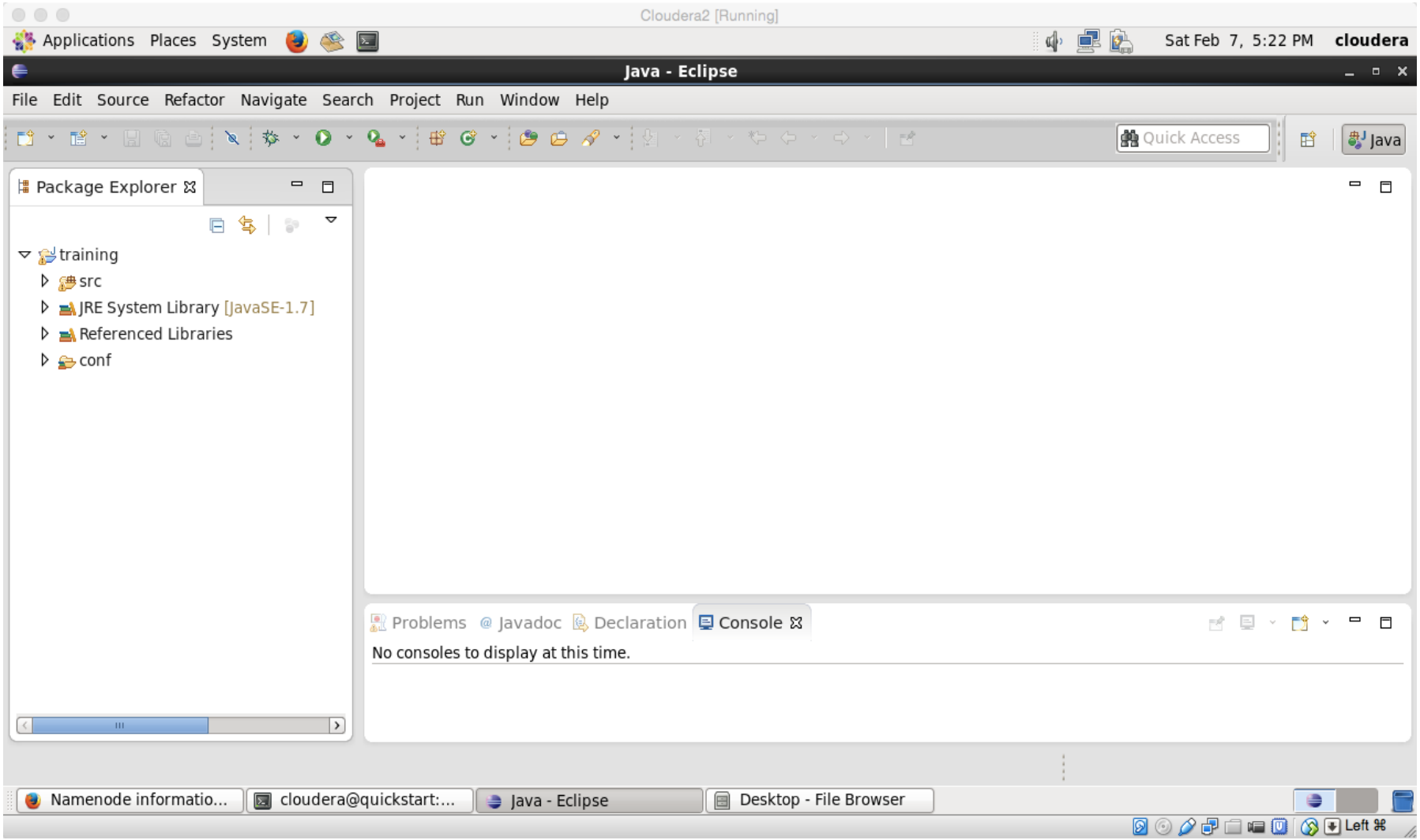
Removing data from HDFS using Shell Command

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

Hands-On: Writing Map/Reduce Program on Eclipse

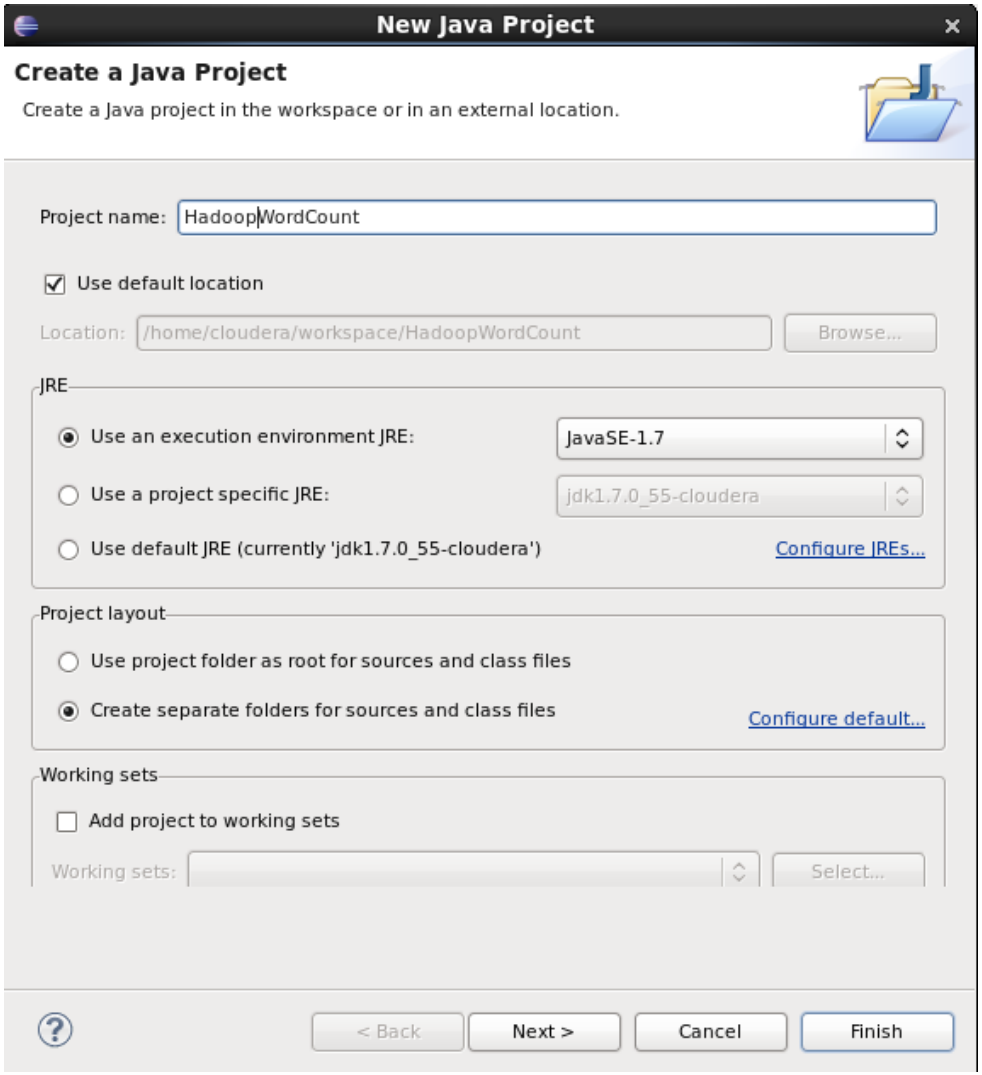
Danairat T., , danairat@gmail.com: Thanachart Numnonda, thanachart@imcinstitute.com

Starting Eclipse in Cloudera VM



Create a Java Project

Let's name it HadoopWordCount

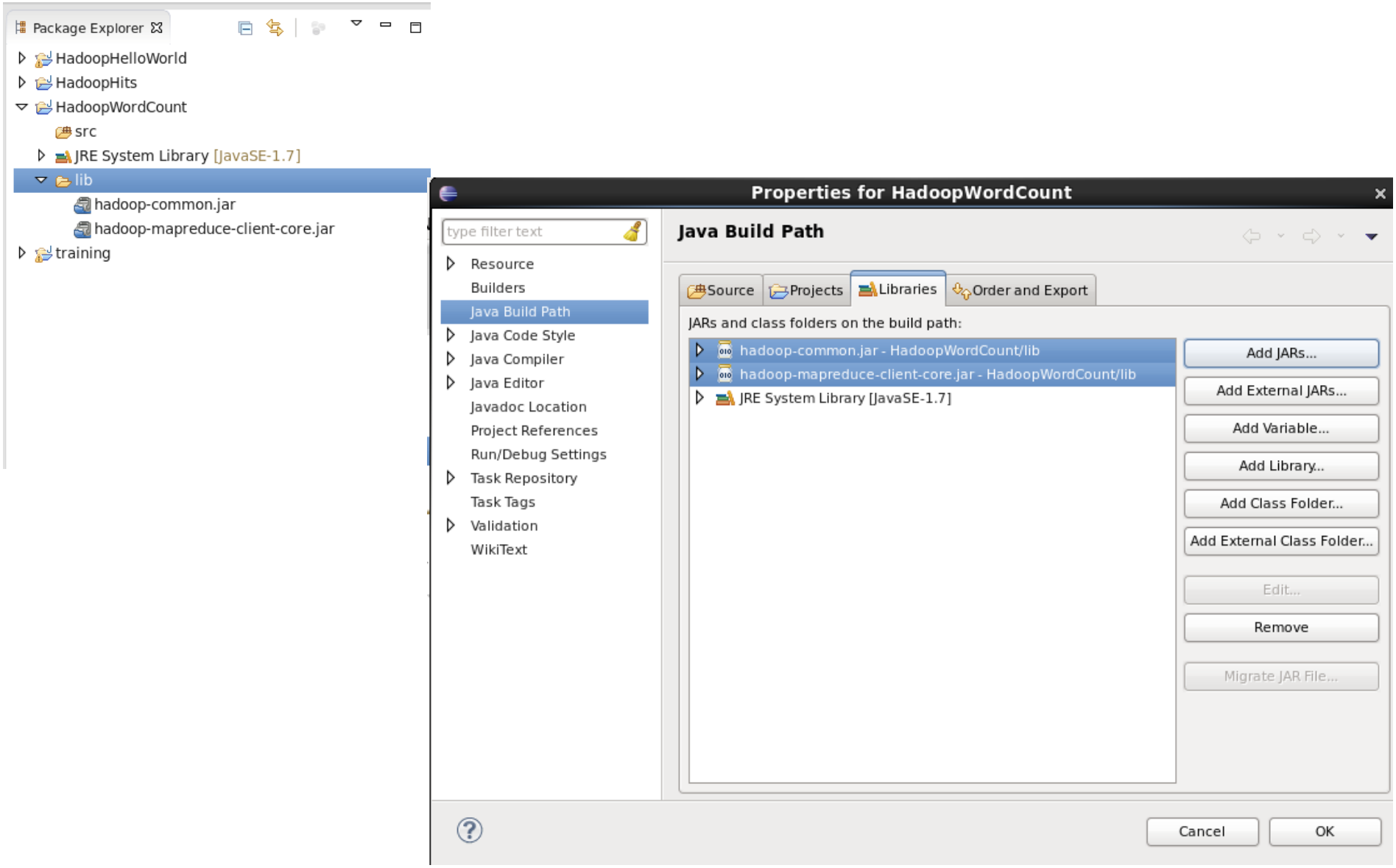


Add dependencies to the project

- Add the following two JARs to your build path
- hadoop-common.jar and hadoop-mapreduce-client-core.jar. Both can be founded at /usr/lib/hadoop/client
- By perform the following steps

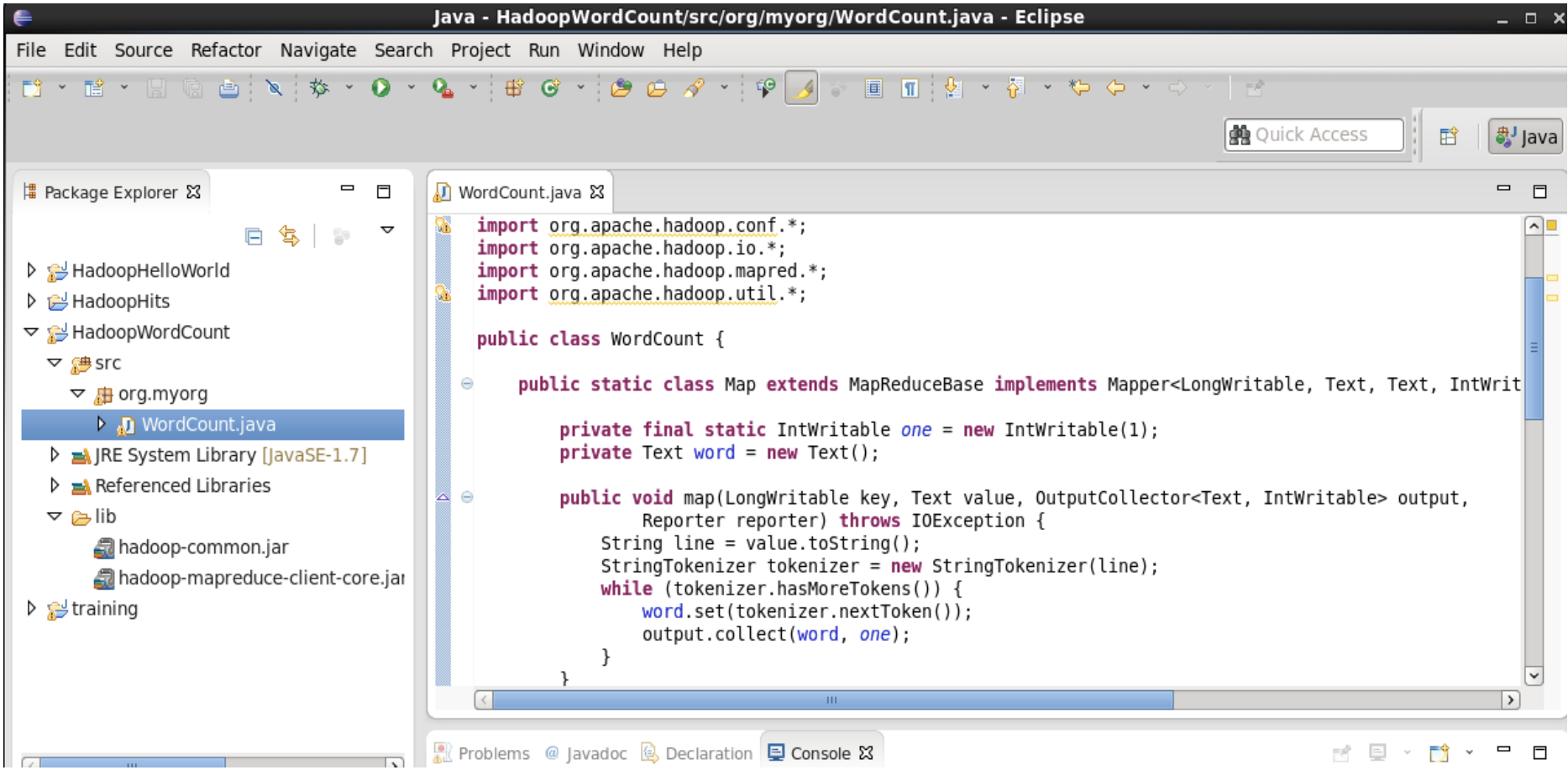
  - Add a folder named lib to the project
  - Copy the mentioned JARs in this folder
  - Right-click on the project name >> select **Build Path** >> then **Configure Build Path**
  - Click on Add Jars, select these two JARs from the lib folder

Add dependencies to the project



Writing a source code

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

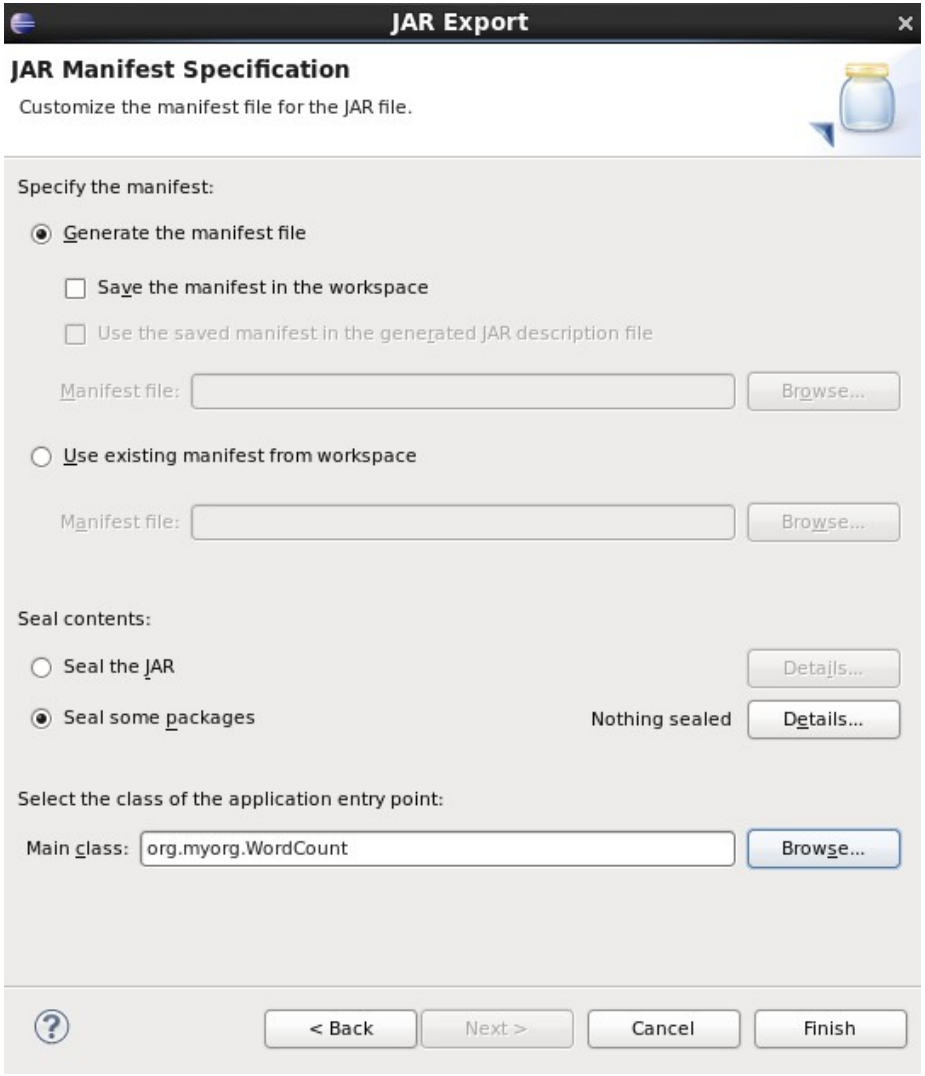

Java - HadoopWordCount/src/org/myorg/WordCount.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
Java
Package Explorer
HadoopHelloWorld
HadoopHits
HadoopWordCount
src
org.myorg
WordCount.java
JRE System Library [JavaSE-1.7]
Referenced Libraries
lib
hadoop-common.jar
hadoop-mapreduce-client-core.jar
training
WordCount.java
import org.apache.hadoop.conf.*;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapred.*;
import org.apache.hadoop.util.*;
public class WordCount {
public static class Map extends MapReduceBase implements Mapper<LongWritable, Text, Text, IntWrit
private final static IntWritable one = new IntWritable(1);
private Text word = new Text();
public void map(LongWritable key, Text value, OutputCollector<Text, IntWritable> output,
Reporter reporter) throws IOException {
String line = value.toString();
StringTokenizer tokenizer = new StringTokenizer(line);
while (tokenizer.hasMoreTokens()) {
word.set(tokenizer.nextToken());
output.collect(word, one);
}
}
Problems Javadoc Declaration Console

Building a Jar file

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

Note: you may need to re-size the dialog font size by select

Windows >> Preferences >> Appearance >> Colors and Fonts


JAR Export
JAR Manifest Specification
Customize the manifest file for the JAR file.
Specify the manifest:
Generate the manifest file
Save the manifest in the workspace
Use the saved manifest in the generated JAR description file
Manifest file:
Browse...
Use existing manifest from workspace
Manifest file:
Browse...
Seal contents:
Seal the JAR
Details...
Seal some packages
Nothing sealed
Details...
Select the class of the application entry point:
Main class:
org.myorg.WordCount
Browse...
< Back
Next >
Cancel
Finish

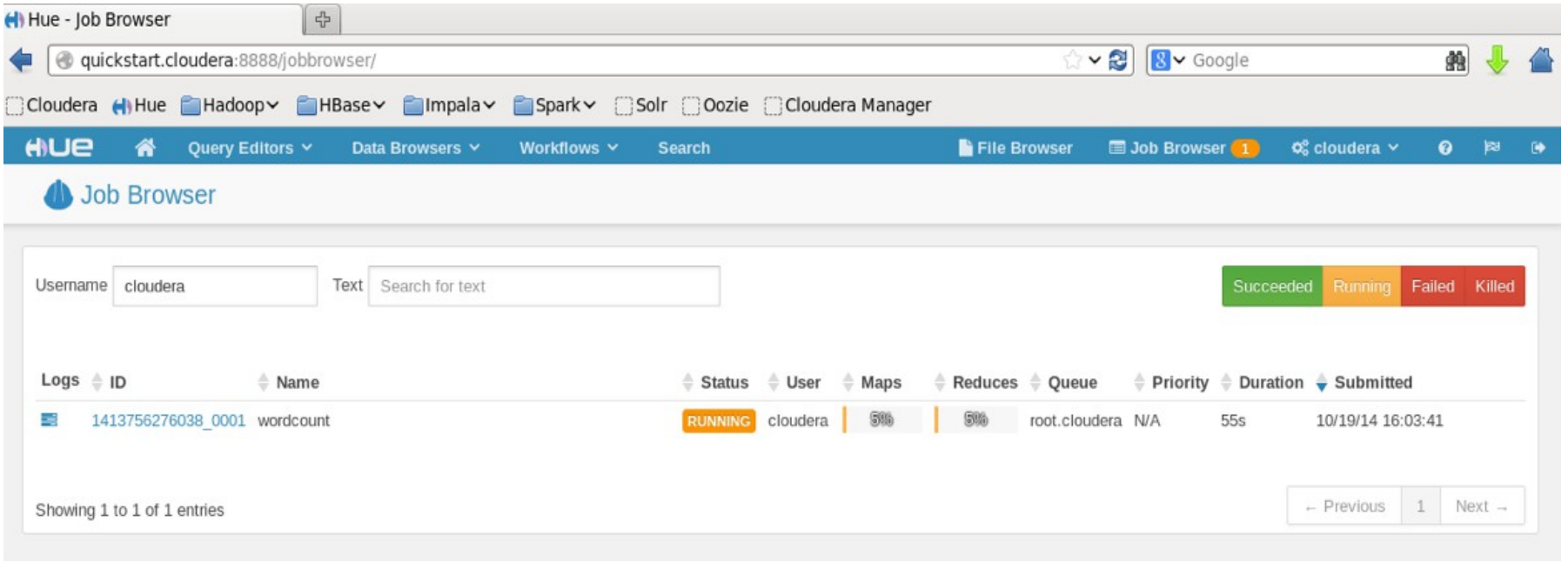
Hands-On: Running Map Reduce and Deploying to Hadoop Runtime Environment

Danairat T., , danairat@gmail.com: Thanachart Numnonda, thanachart@imcinstitute.com

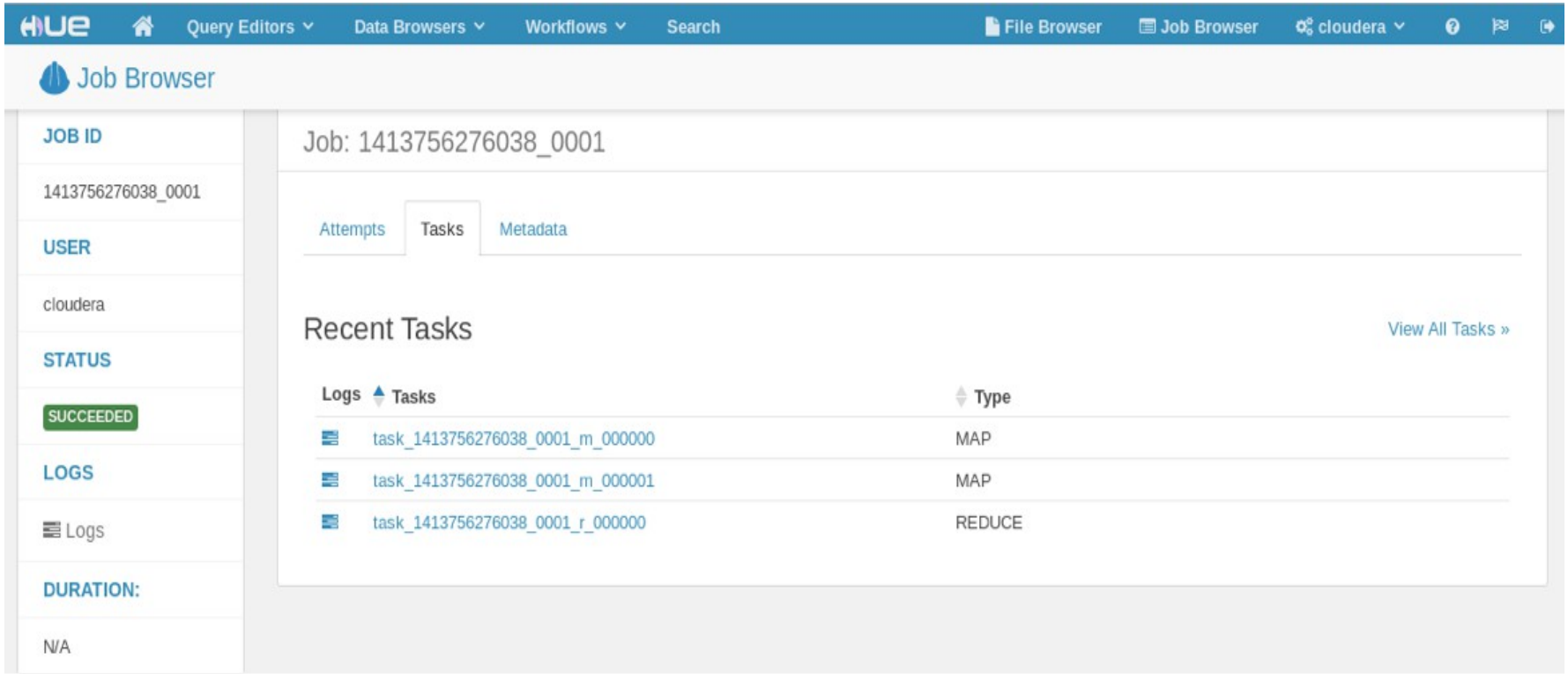
Running Map Reduce Program

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

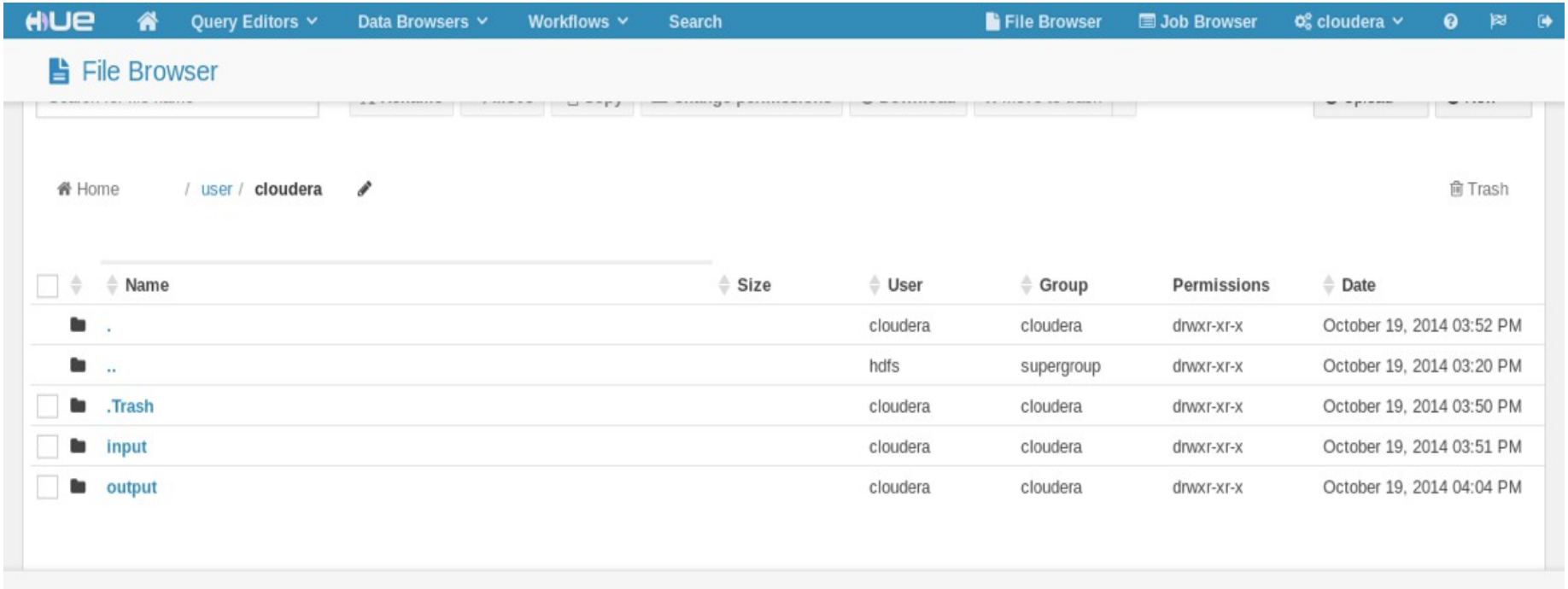
Reviewing MapReduce Job in Hue



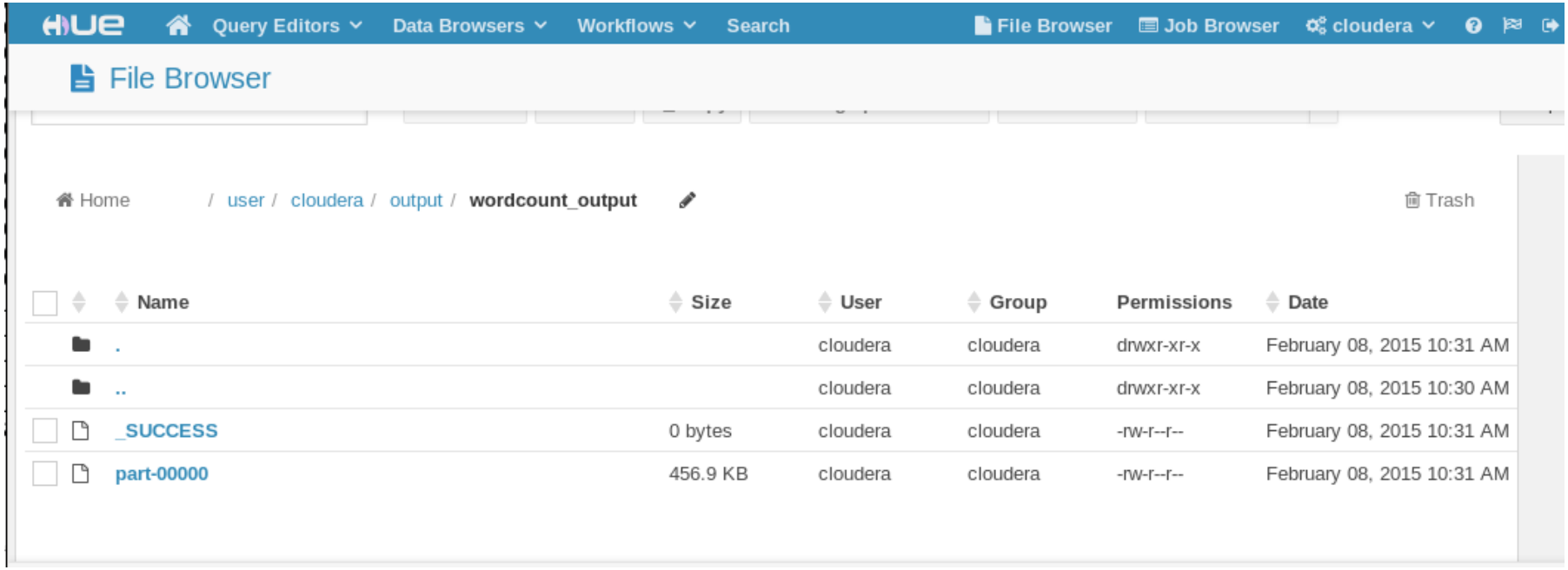
Reviewing MapReduce Job in Hue



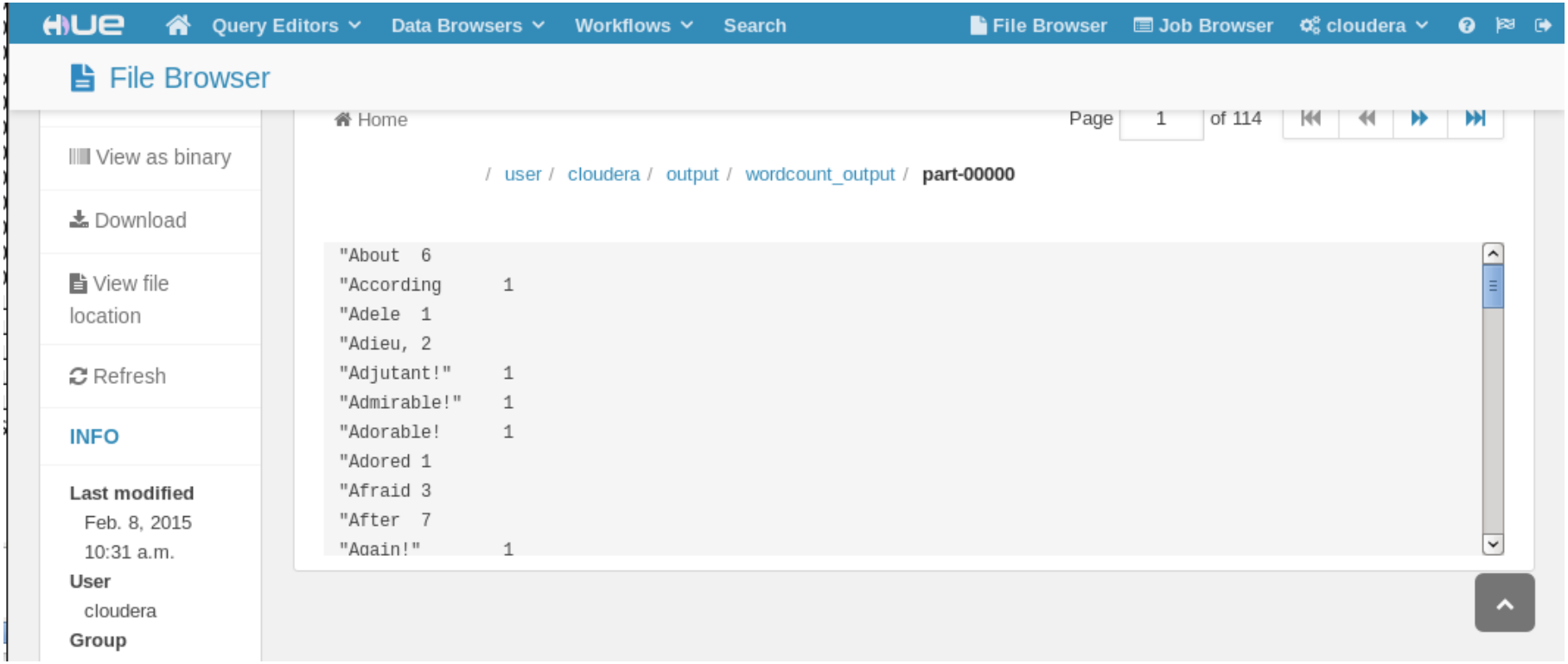
Reviewing MapReduce Output Result



Reviewing MapReduce Output Result



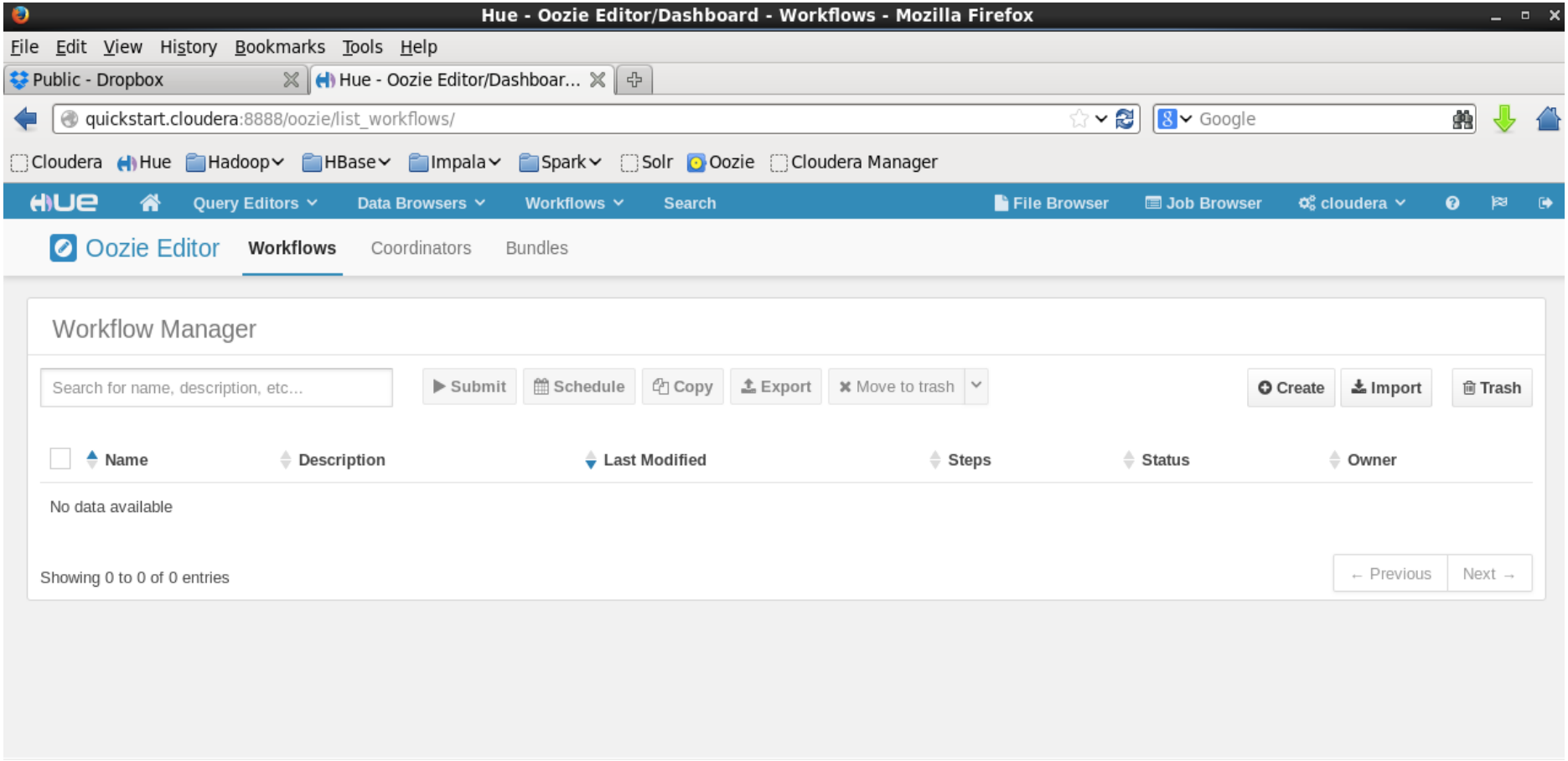
Reviewing MapReduce Output Result



Hands-On: Running Map Reduce using Oozie workflow

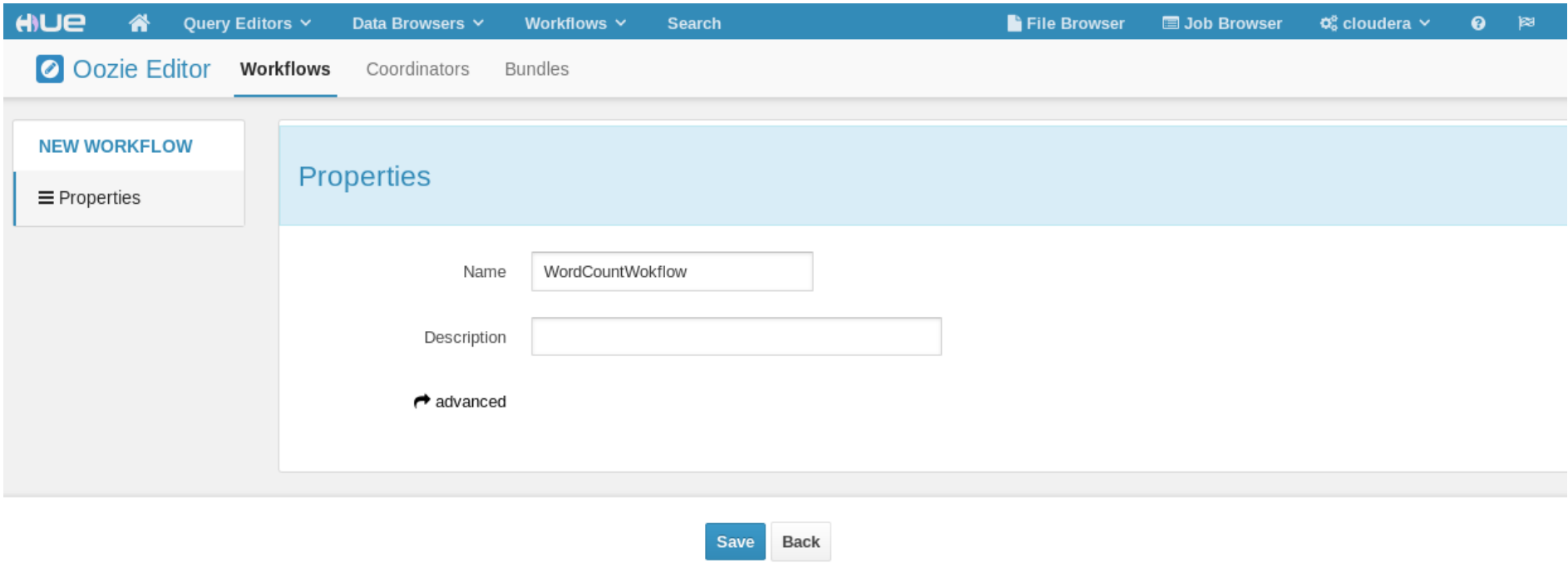
Danairat T., , danairat@gmail.com: Thanachart Numnonda, thanachart@imcinstitute.com

Using Hue: select WorkFlow >> Editor



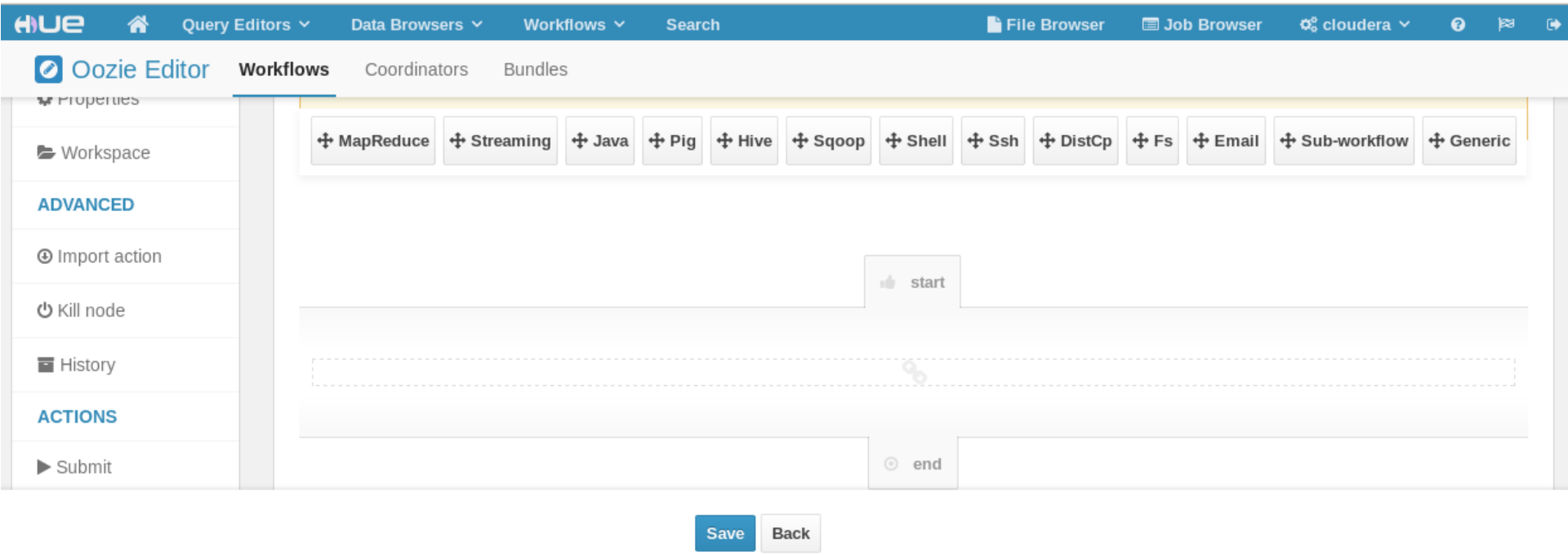
Create a new workflow

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



Select a Java job for the workflow

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

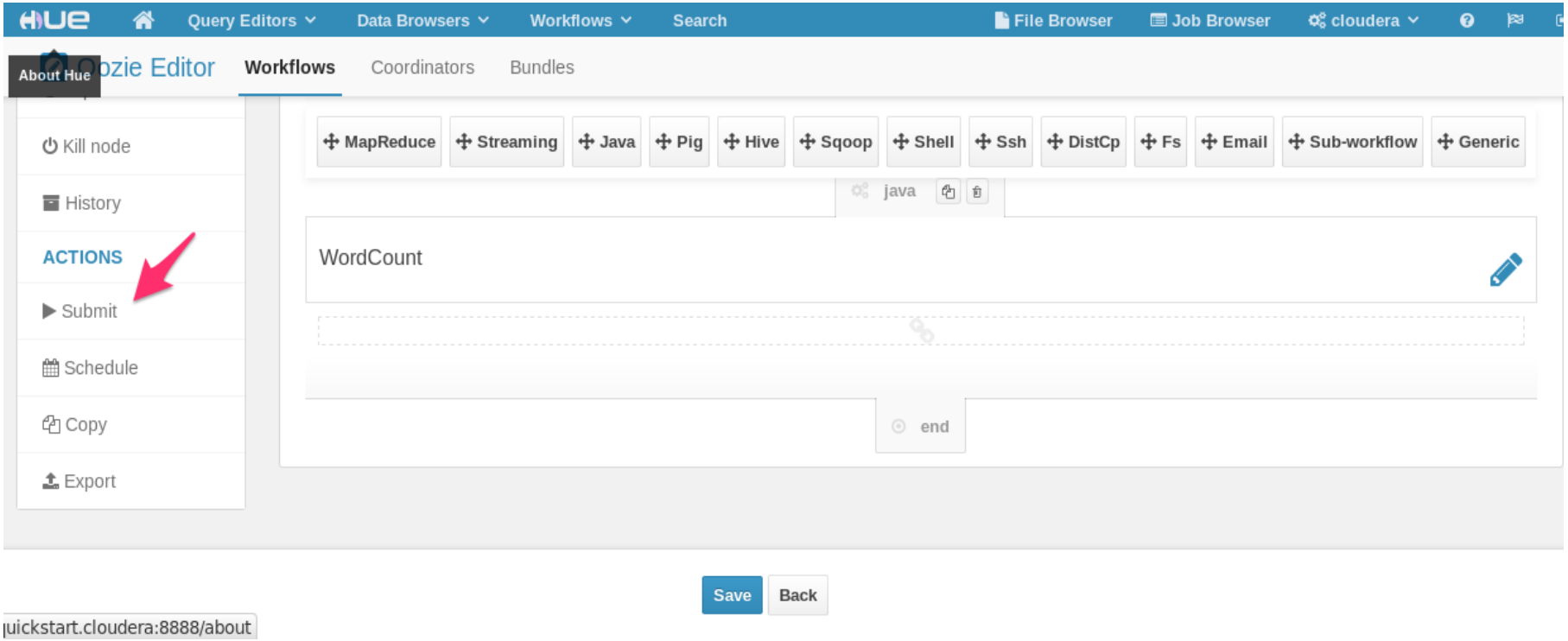


Edit the Java Job

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

Submit the workflow

- Click Done, follow by Save
- Then click submit



Hands-On: Working with a csv data

Danairat T., , danairat@gmail.com: Thanachart Numnonda, thanachart@imcinstitute.com

A sample CSV data

- The input data is access logs with the following form

  Date, Requesting-IP-Address

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

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

HitsByCountryMapper.java

```
package learning.bigdata.mapreduce;

import java.io.IOException;

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

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

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

	@Override
	protected void setup(Context context) throws IOException, InterruptedException {

		super.setup(context);
	}

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


		try {
			String valueString = value.toString();

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

HitsByCountryMapper.java

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

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

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


    private static String getCountryNameFromIpAddress(String ipAddress) {

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

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

        return null;
    }
}
```

HitsByCountryReducer.java

```
package learning.bigdata.mapreduce;

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

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

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

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

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

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

HitsByCountry.java

```
package learning.bigdata.main;

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

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


public class HitsByCountry extends Configured implements Tool {

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

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

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

HitsByCountry.java

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

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

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

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

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

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

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

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

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

Lecture: Developing Complex Hadoop MapReduce Applications

Danairat T., , danairat@gmail.com: Thanachart Numnonda, thanachart@imcinstitute.com

Choosing appropriate Hadoop data types

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

Examples

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

```
public class SampleMapper extends Mapper<LongWritable, Text, Text,
IntWritable> {

public void map(LongWritable key, Text value,
    Context context) … {
……  }
}
```

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

```
public class Reduce extends Reducer<Text, IntWritable, Text,
IntWritable> {

public void reduce(Text key,
    Iterable<IntWritable> values, Context context) {
  ……  }
}
```

Hadoop built-in data types

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

Hadoop built-in data types

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

Implementing a custom Hadoop Writable data type

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

Examples

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

```
public class LogWritable implements Writable{

  private Text userIP, timestamp, request;
  privateIntWritableresponseSize, status;

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

Examples

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

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

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

Examples

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

```
public class LogProcessorMap extends Mapper<LongWritable,
Text, Text, LogWritable> {
….
}

public class LogProcessorReduce extends Reducer<Text,
LogWritable, Text, IntWritable> {

public void reduce(Text key,
Iterable<LogWritable> values, Context context) {
     ……  }
}
```

Choosing a suitable Hadoop InputFormat for your input data format

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

InputFormat that Hadoop provide

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

Implementing new input data formats

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

Formatting the results of MapReduce computations – using Hadoop OutputFormats

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

  TextOutputFormat<K,V> as the default *OutputFormat*

Hands-On: Analytics Using MapReduce

Three Analytic MapReduce Examples

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

Preparing Example Data

NASA weblog dataset available from

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

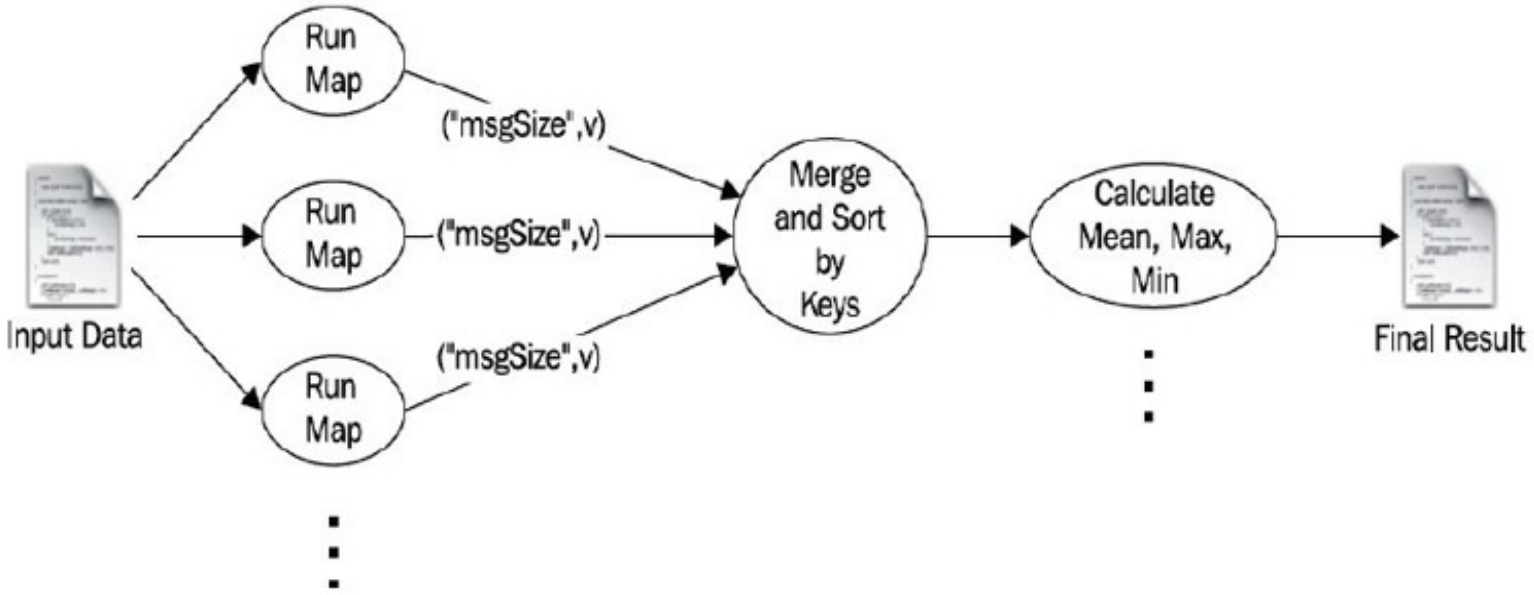
is a real-life dataset collected using the requests received by NASA web servers.

Download the weblog dataset from ftp://ita.ee.lbl.gov/traces/NASA_ access_log_Jul95.gz and unzip it. We call the extracted folder as DATA_DIR.

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

Simple analytics using MapReduce

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



Source: Hadoop MapReduce CookBook

WebLogMessageSizeAggregator.java

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

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

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

WebLogMessageSizeAggregator.java

```
        public void map(LongWritable key, Text value, OutputCollector<Text,
IntWritable> output,Reporter reporter) throws IOException {

            Matcher matcher = httplogPattern.matcher(value.toString());

            if (matcher.matches()) {

                int size = Integer.parseInt(matcher.group(5));

                output.collect(new Text("msgSize"), new IntWritable(size));

            }

        }

    }
```

WebLogMessageSizeAggregator.java

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

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

            double tot = 0;

            int count = 0;

            int min = Integer.MAX_VALUE;

            int max = 0;


            while (values.hasNext()) {

                int value = values.next().get();

                tot = tot + value;

                count++;

                if (value < min) {

                    min = value;

                }

                if (value > max) {

                    max = value;

                }

            }
```

WebLogMessageSizeAggregator.java

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

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

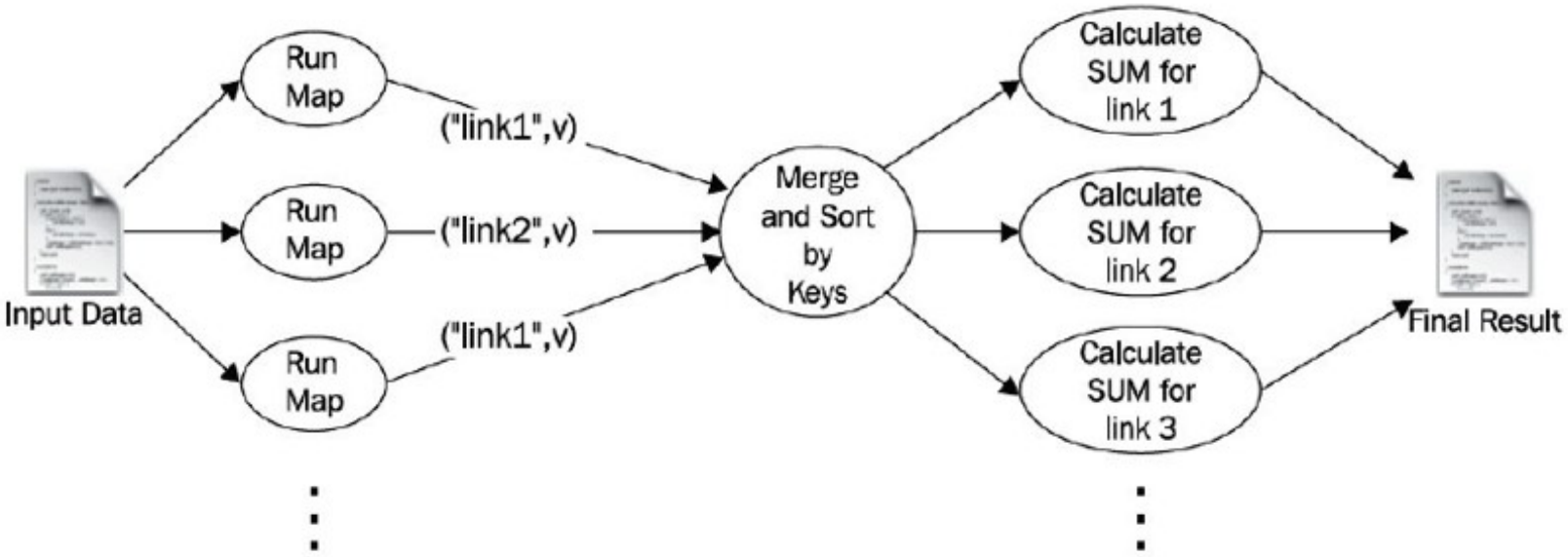
Compile, Build JAR, Submit Job, Review Result

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

```
Mean    1150
Max     6823936
Min     0
```

Performing Group-By using MapReduce

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



Source: Hadoop MapReduce CookBook

WeblogHitsByLinkProcessor.java

```
public class WeblogHitsByLinkProcessor {

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


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


        private final static IntWritable one = new IntWritable(1);

        private Text word = new Text();


        public void map(LongWritable key, Text value, OutputCollector<Text,
IntWritable> output,Reporter reporter) throws IOException {

            Matcher matcher = httplogPattern.matcher(value.toString());

            if (matcher.matches()) {

                String linkUrl = matcher.group(4);

                word.set(linkUrl);

                output.collect(word, one);

            }

        }

    }
```

WeblogHitsByLinkProcessor.java

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


        private IntWritable result = new IntWritable();


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


            int sum = 0;
            while (values.hasNext()) {
                sum += values.next().get();
            }
            result.set(sum);
            output.collect(key, result);
        }
    }
```

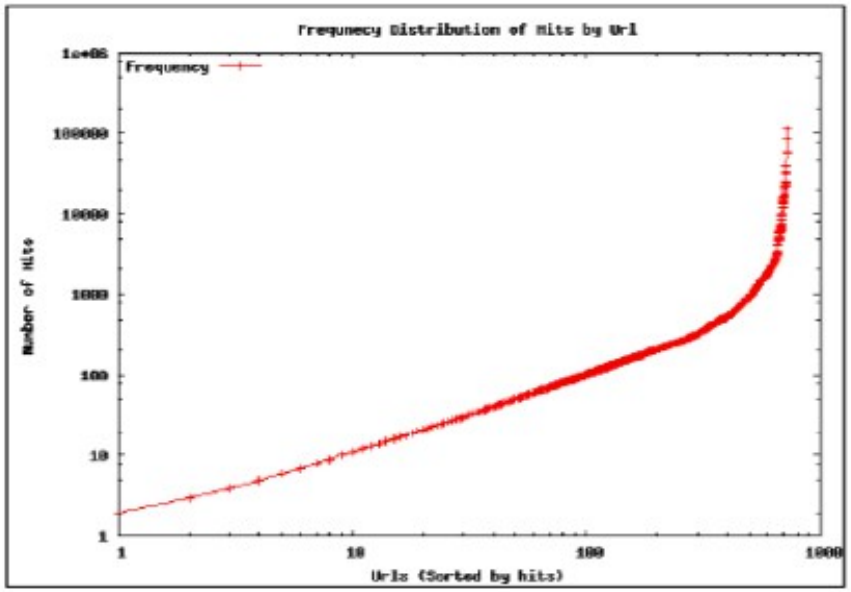
Compile, Build JAR, Submit Job, Review Result

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

```
/       32667
/%7Edowns/home.html     2
/%7Edowns/launchup.gif  2
/%7edowns/home.html     3
/./     3
/.ksc.html      6
//      3
//biomed/climate/gif/f16pcfinmed.gif    1
//biomed/gif/   1
//biomed/gif/aerial.gif 1
//elv/bakgro.gif        2
//elv/elvhead2.gif      1
//elv/elvhead3.gif      1
//elv/elvpage.htm       2
//elv/endball.gif       1
//elv/vidpicp.htm       1
//elv/whnew.htm 1
```

Calculating frequency distributions and sorting using MapReduce

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



Source: Hadoop MapReduce CookBook

Lecture
Understanding HBase

Danairat T., , danairat@gmail.com: Thanachart Numnonda, thanachart@imcinstitute.com

Introduction

An open source, non-relational, distributed database

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

HBase Features

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

Hive.apache.org

When to use Hbase?

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

Hive.apache.org

Which one to use?

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

Hive.apache.org

HBase vs. RDBMS

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

- Given this RDBMS:

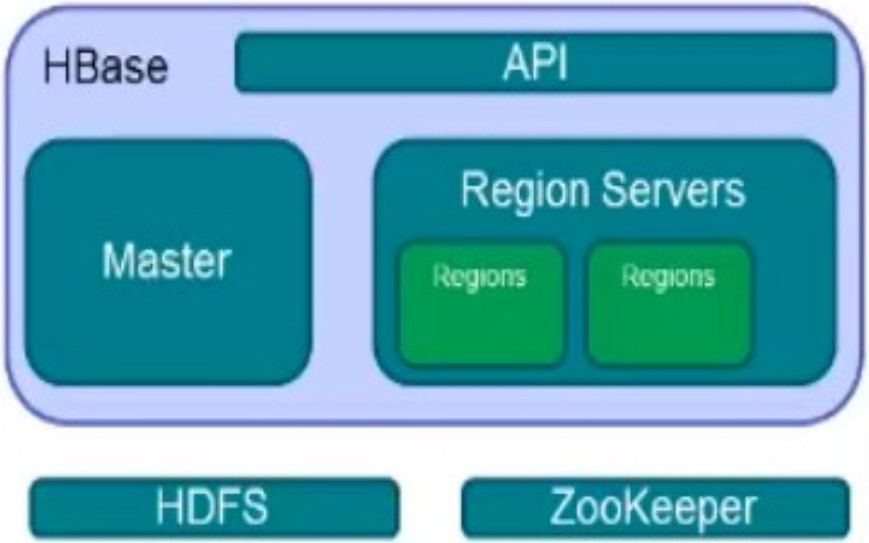
ID (Primary key)	Last name	First name	Password	Timestamp
1234	Smith	John	Hello, world!	20130710
5678	Cooper	Joyce	wysiwyg	20120825
5678	Cooper	Joyce	wisiwig	20130916

- Logical view in HBase:

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

HBase Components

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



Hive.apache.org

HBase Shell Commands

- See the list of the tables

```
list
```

- Create a table:

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

- Insert data into a table:

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

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

- Retrieve data from a table:

  Retrive "rowA" from the table "testTable"

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

- Iterate through a table:

```
– scan 'testTable'
```

- Delete a table:

```
disable 'testTable'
drop 'testTable'
```

Hive.apache.org

Hands-On: Running HBase

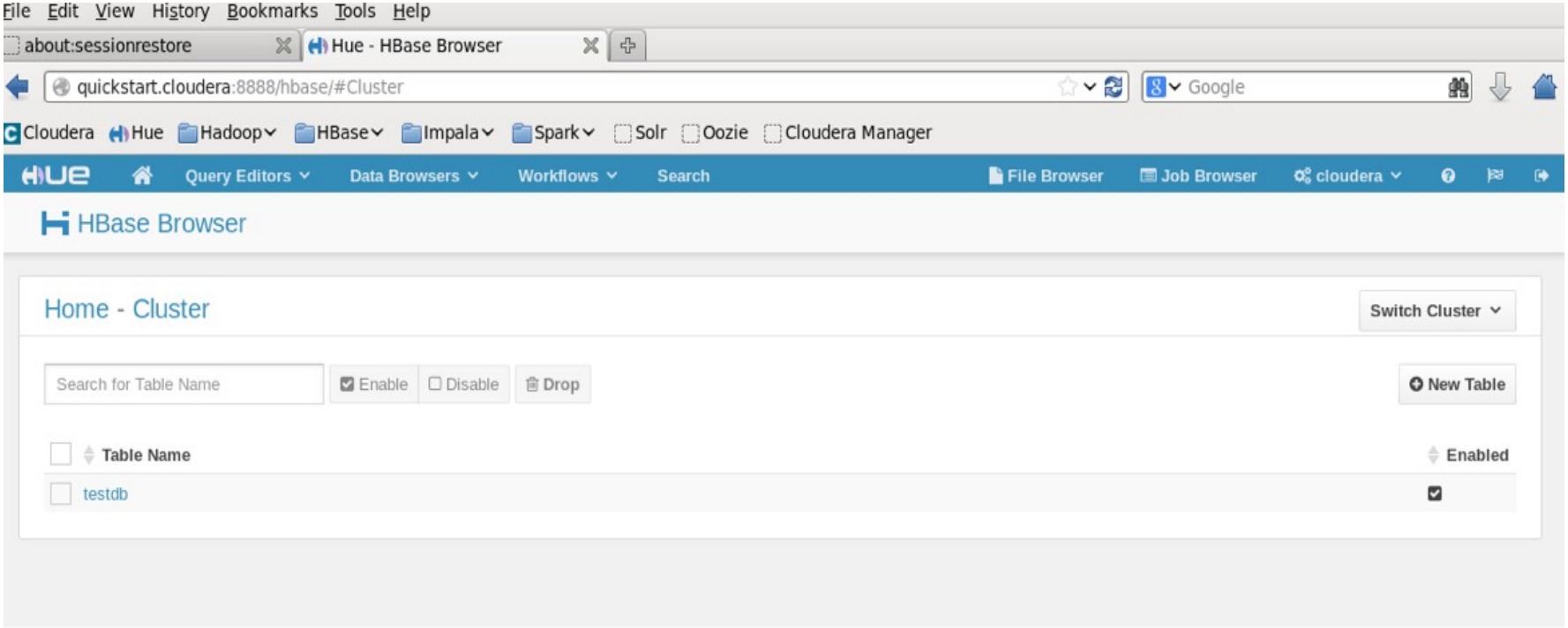
Starting HBase shell

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

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

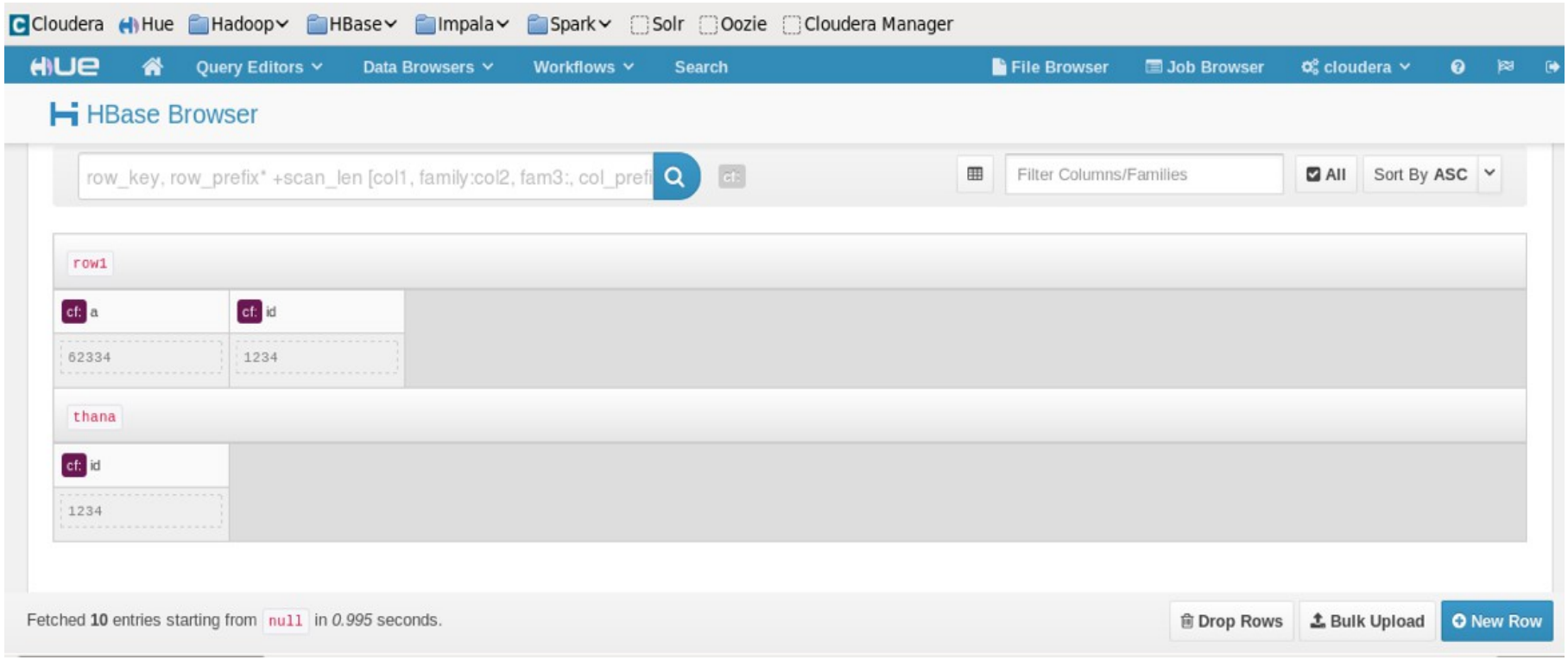
Create a table and insert data in HBase

```
hbase(main):009:0> create 'test', 'cf'
0 row(s) in 1.0830 seconds
hbase(main):010:0> put 'test', 'row1', 'cf:a', 'val1'
0 row(s) in 0.0750 seconds
hbase(main):011:0> scan 'test'
ROW                           COLUMN+CELL
 row1                         column=cf:a, timestamp=1375363287644,
value=val1
1 row(s) in 0.0640 seconds

hbase(main):002:0> get 'test', 'row1'
COLUMN                        CELL
 cf:a                         timestamp=1375363287644, value=val1
1 row(s) in 0.0370 seconds
```

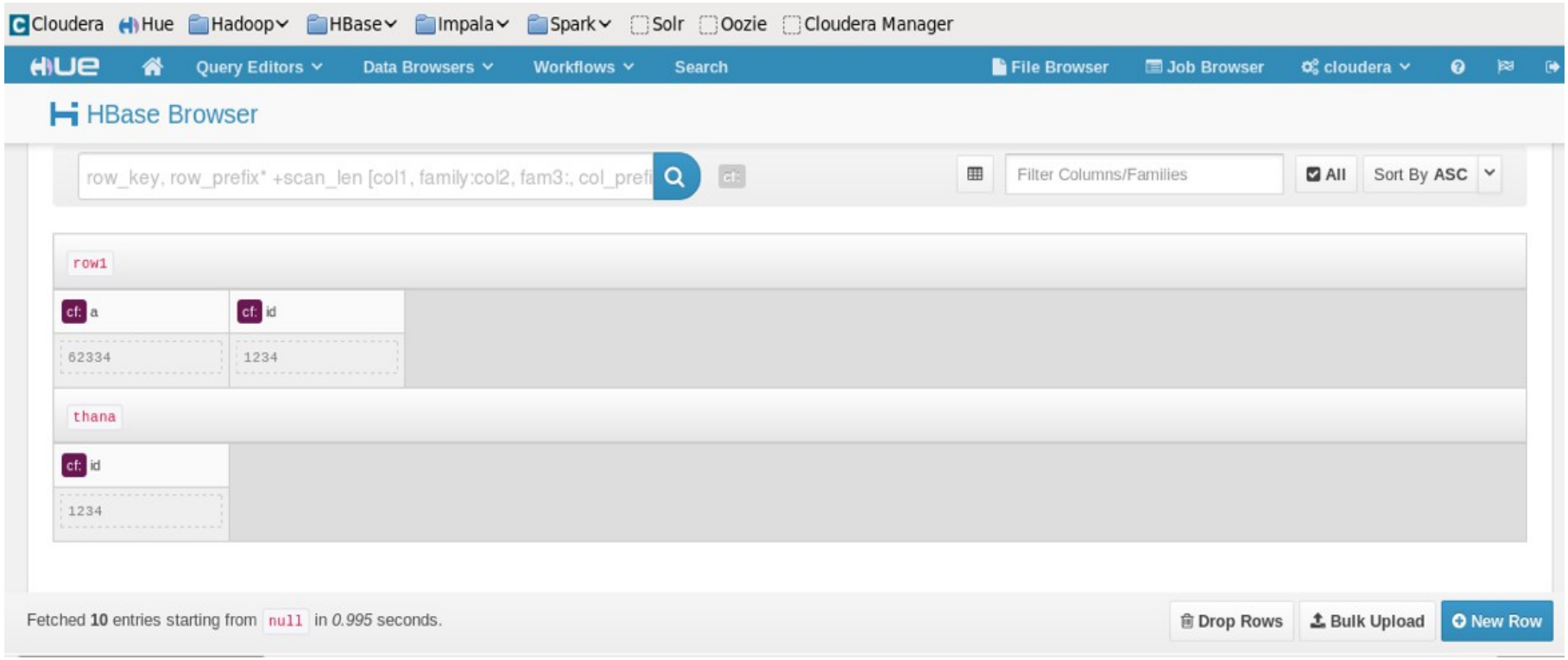
Using Data Browsers in Hue for HBase



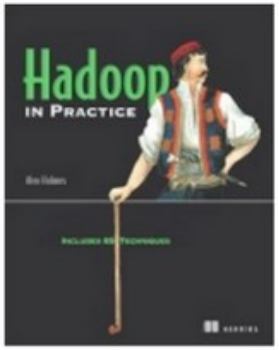
Using Data Browsers in Hue for HBase

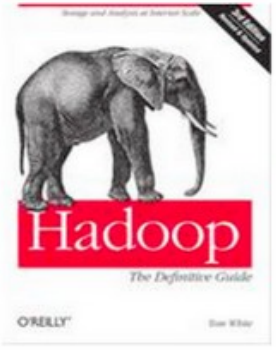


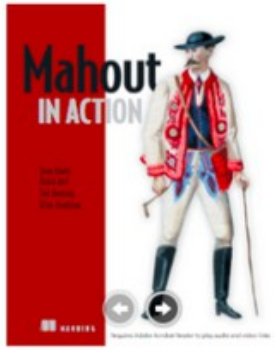
Using Data Browsers in Hue for HBase



Recommendation to Further Study







Thank you

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