Cloudera

Sorayut Glomglome

Hadoop using Cloudera on Amazon EC2

March 2016

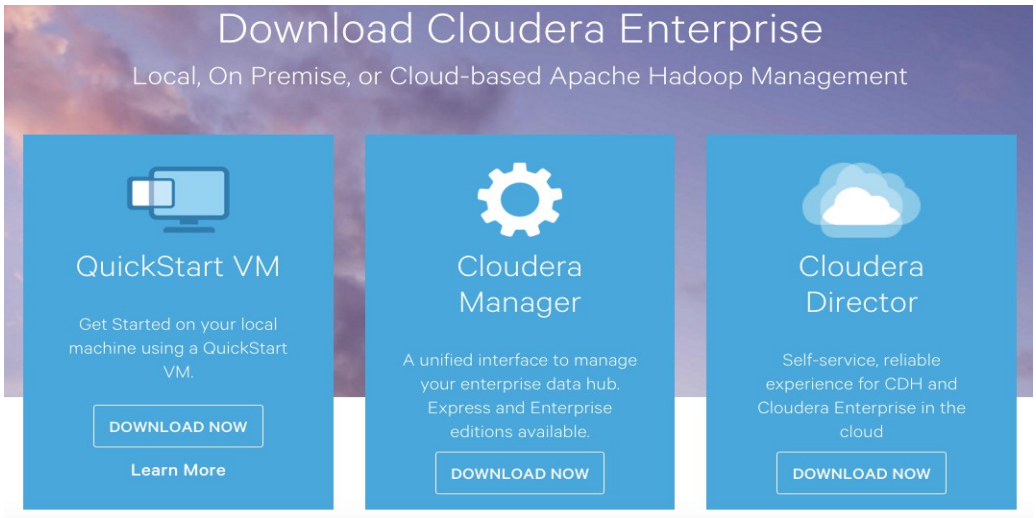
Dr.Thanachart Numnonda
IMC Institute
thanachart@imcinstitute.com

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

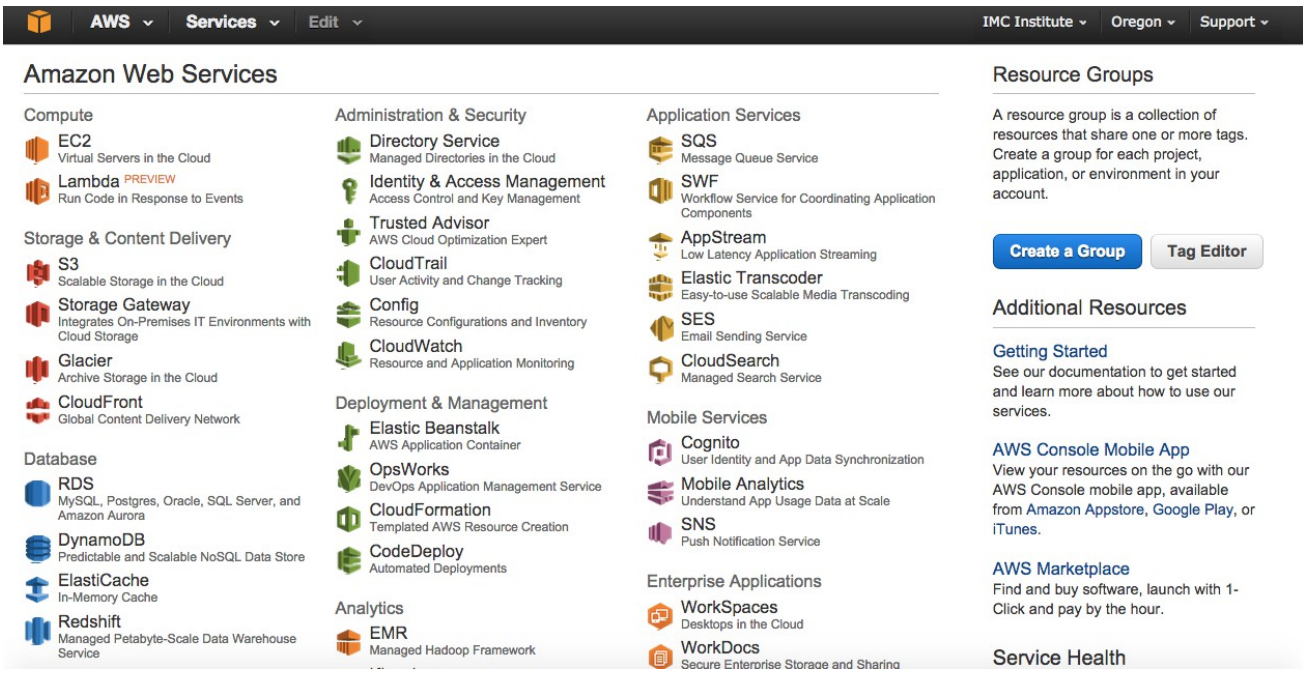
Cloudera VM

This lab will use a EC2 virtual server on AWS to install Cloudera, However, you can also use Cloudera QuickStart VM which can be downloaded from:

http://www.cloudera.com/content/www/en-us/downloads.html



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

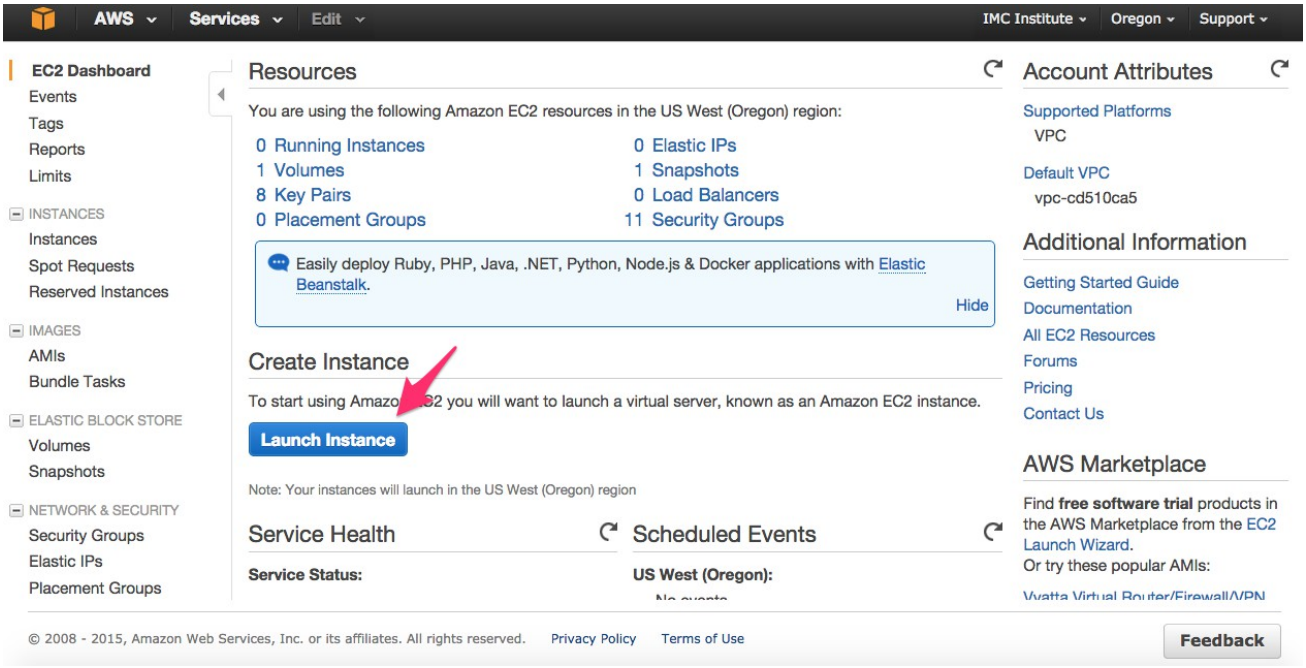


Virtual Server

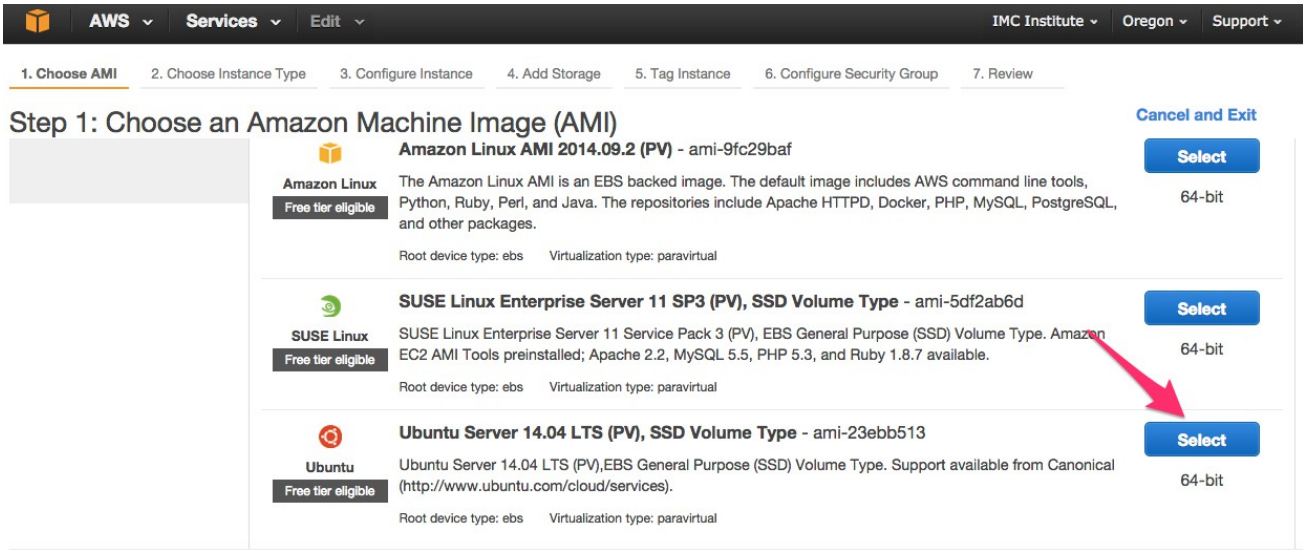
This lab will use a EC2 virtual server to install a Cloudera Cluster using the following features:

- Ubuntu Server 14.04 LTS
- Four m3.xLarge 4vCPU, 15 GB memory,80 GB SSD
- Security group: default
- Keypair: imchadoop

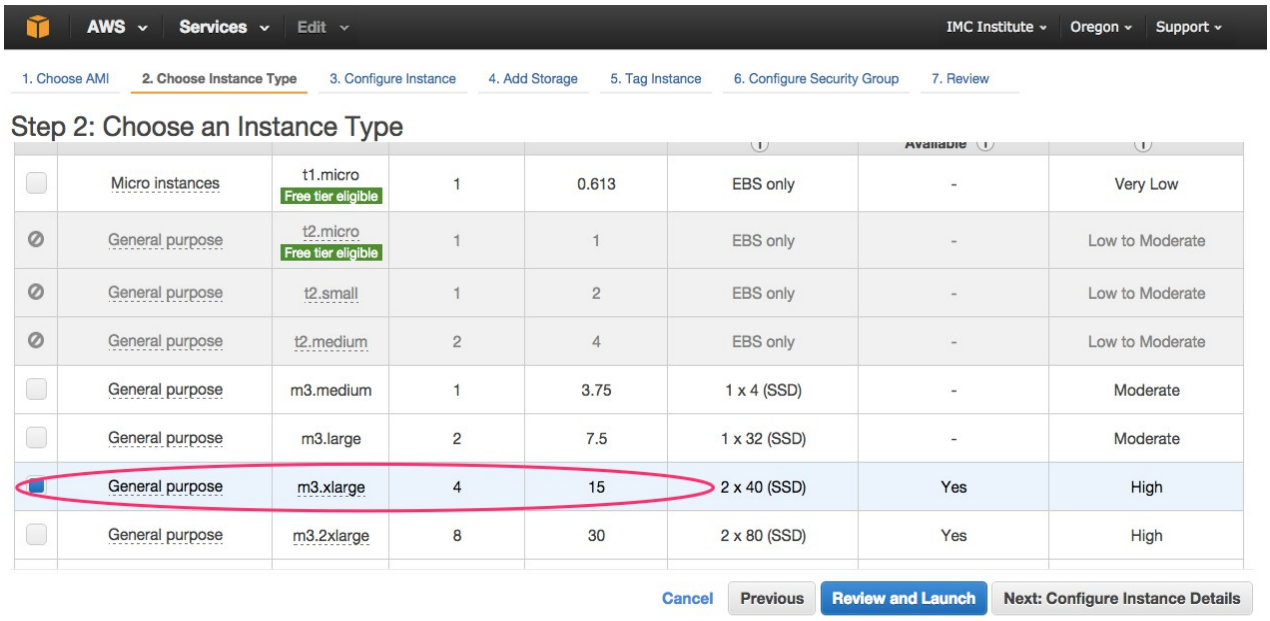
Select a EC2 service and click on Lunch Instance



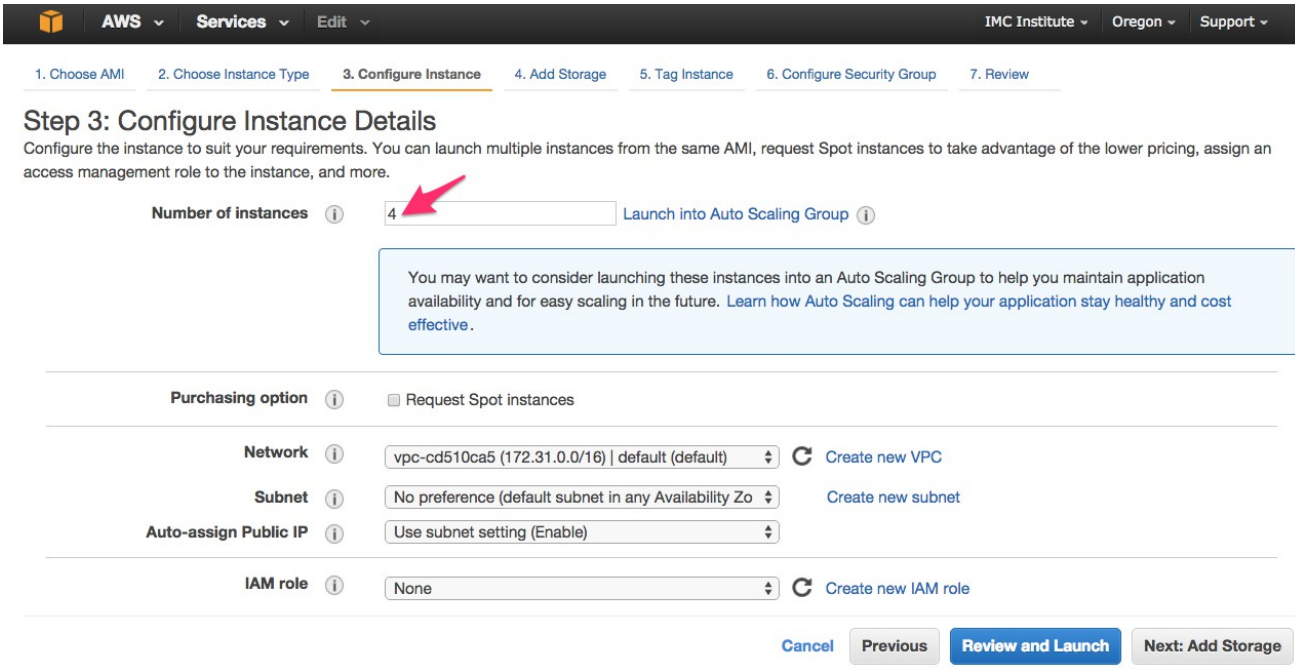
Select an Amazon Machine Image (AMI) and Ubuntu Server 14.04 LTS (PV)



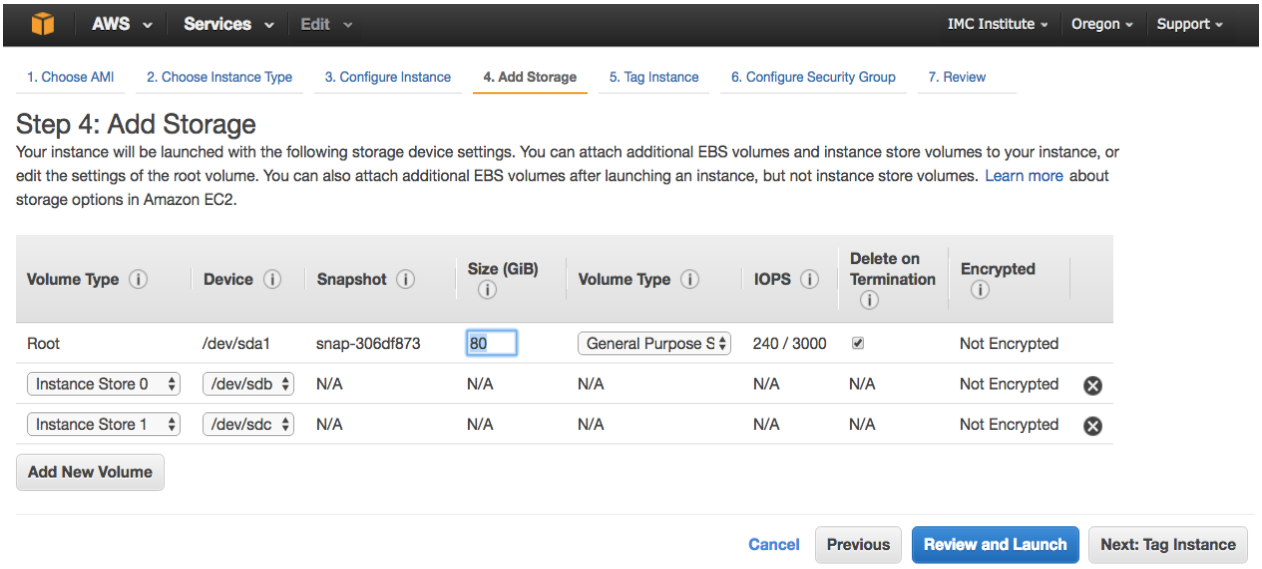
Choose m3.xlarge Type virtual server



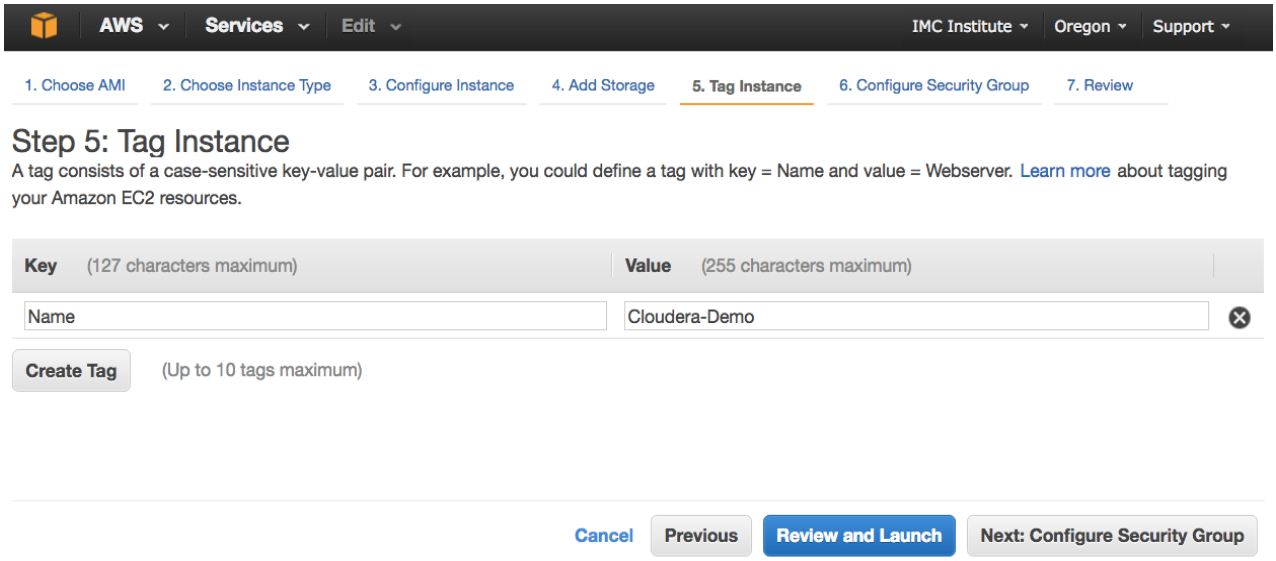
Set the number of Instance to 4



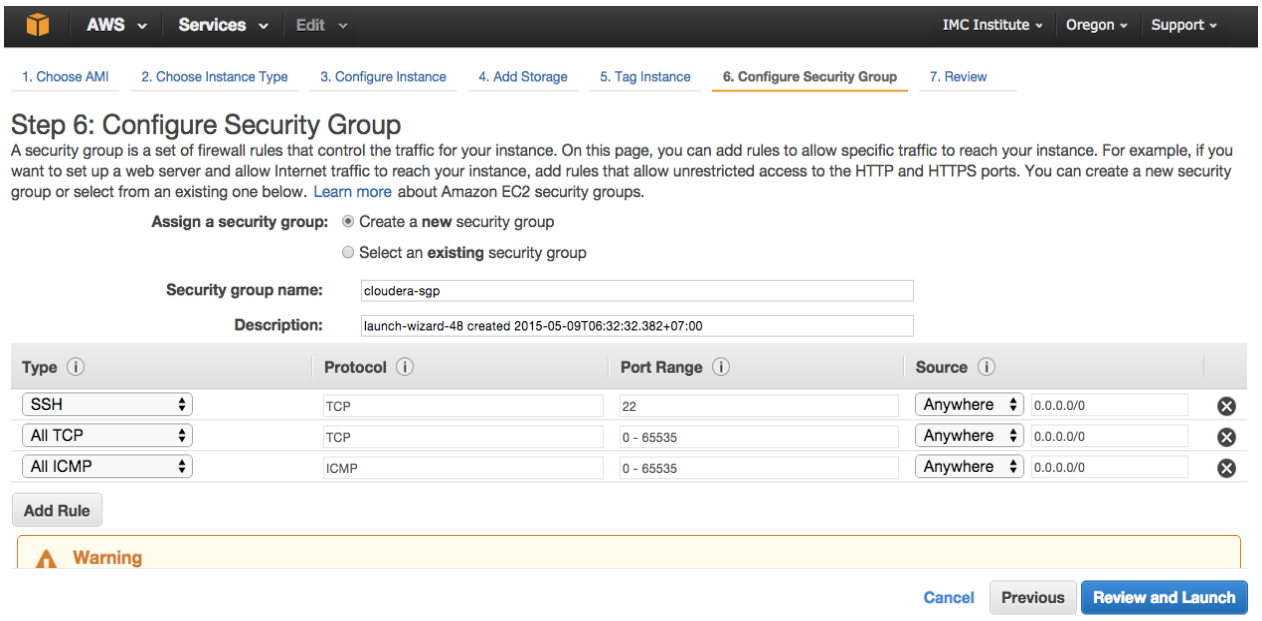
Add Storage: 80 GB



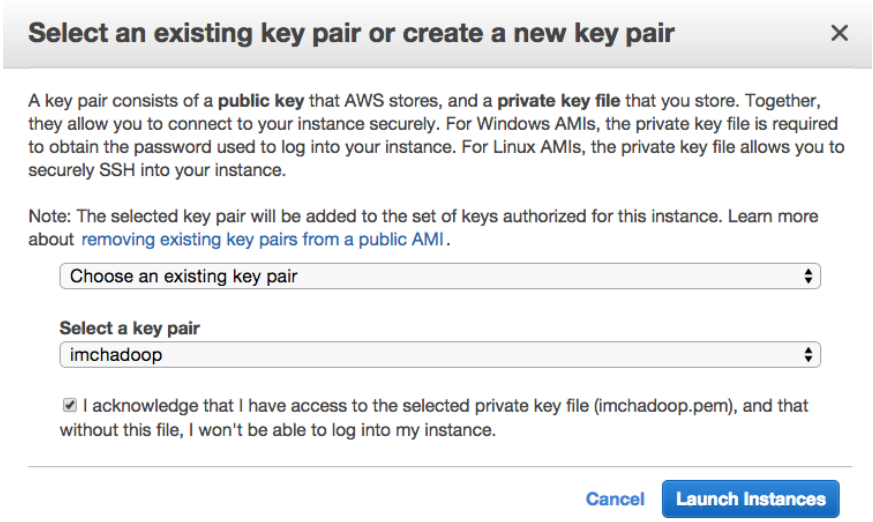
Name the instance



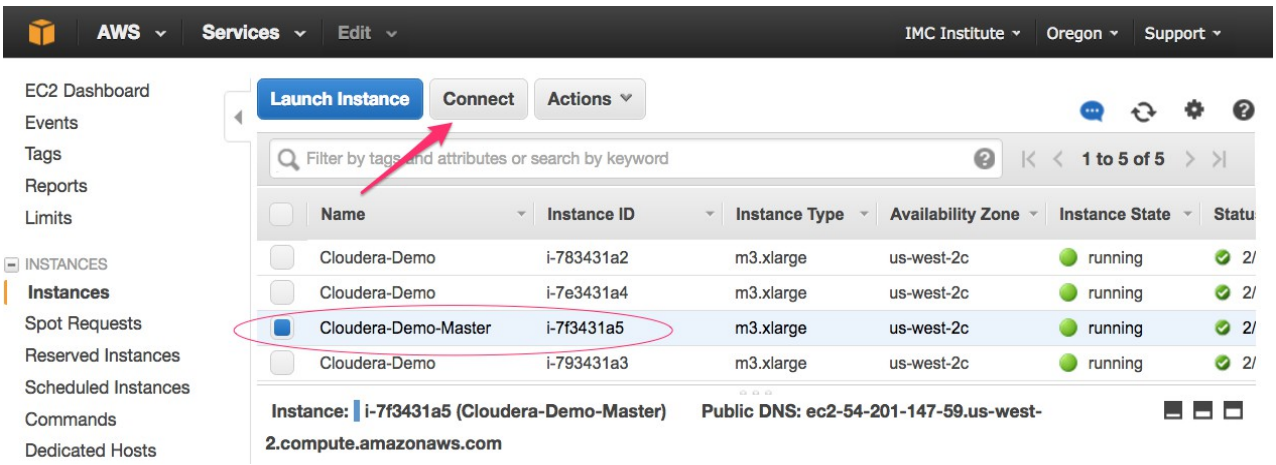
Select Create a new security group > Add Rule as follows



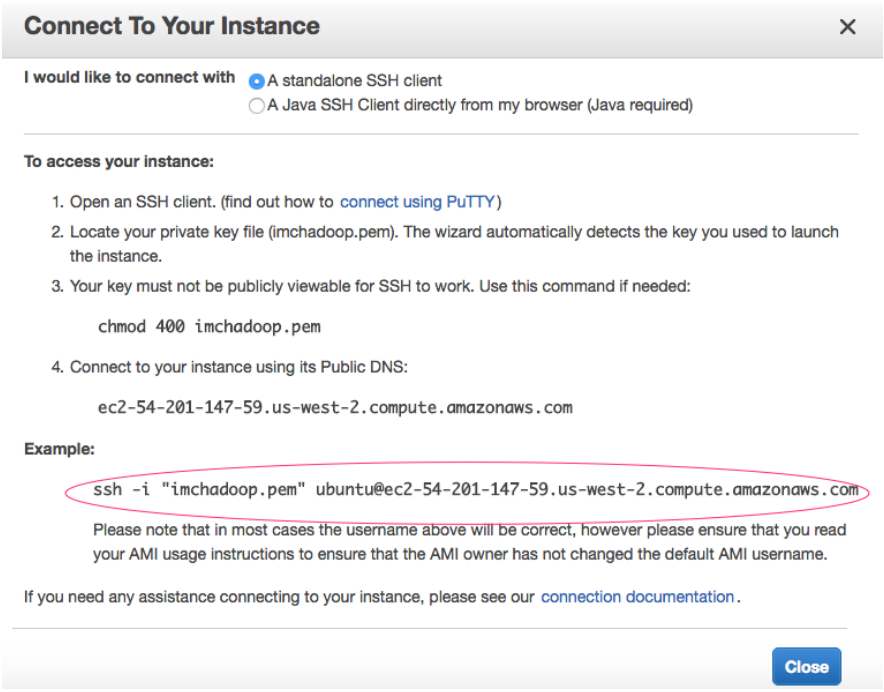
Click Launch and choose imchadoop as a key pair



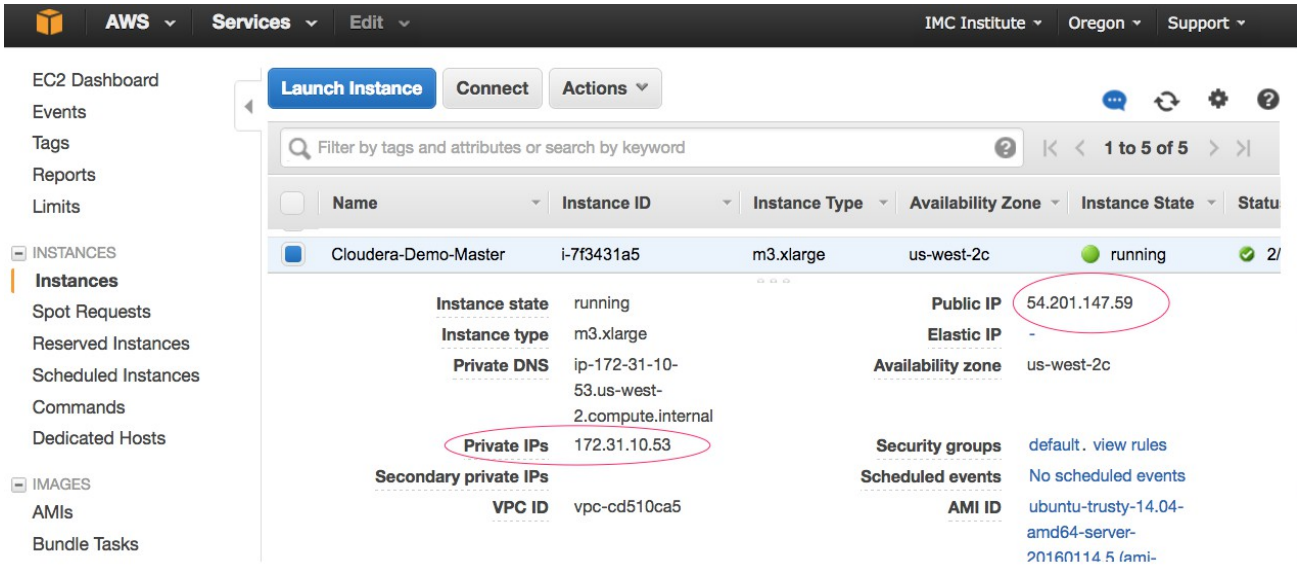
Review an instance and rename one instance as a master / click Connect for an instruction to connect to the instance



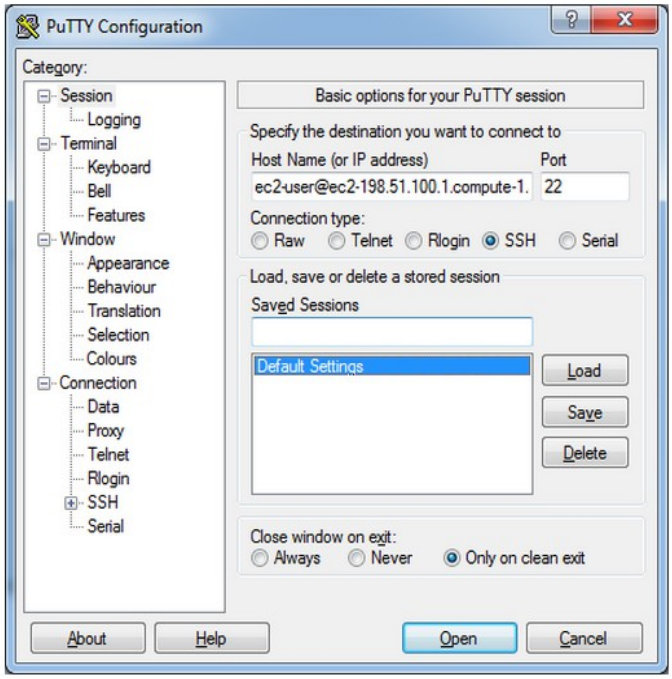
Connect to an instance from Mac/Linux



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



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:~$
```

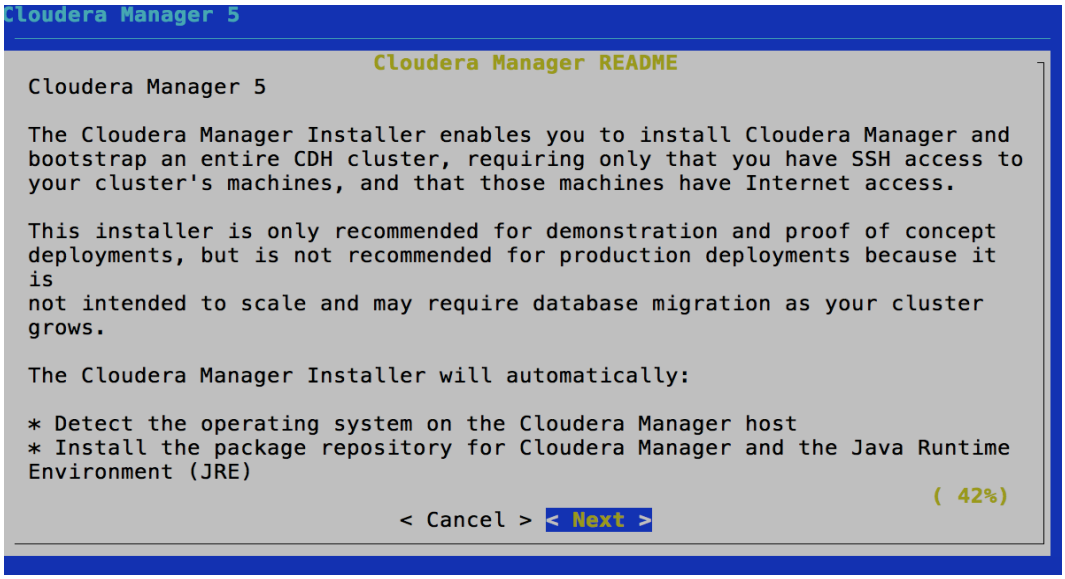
Hands-On: Installing Cloudera on EC2

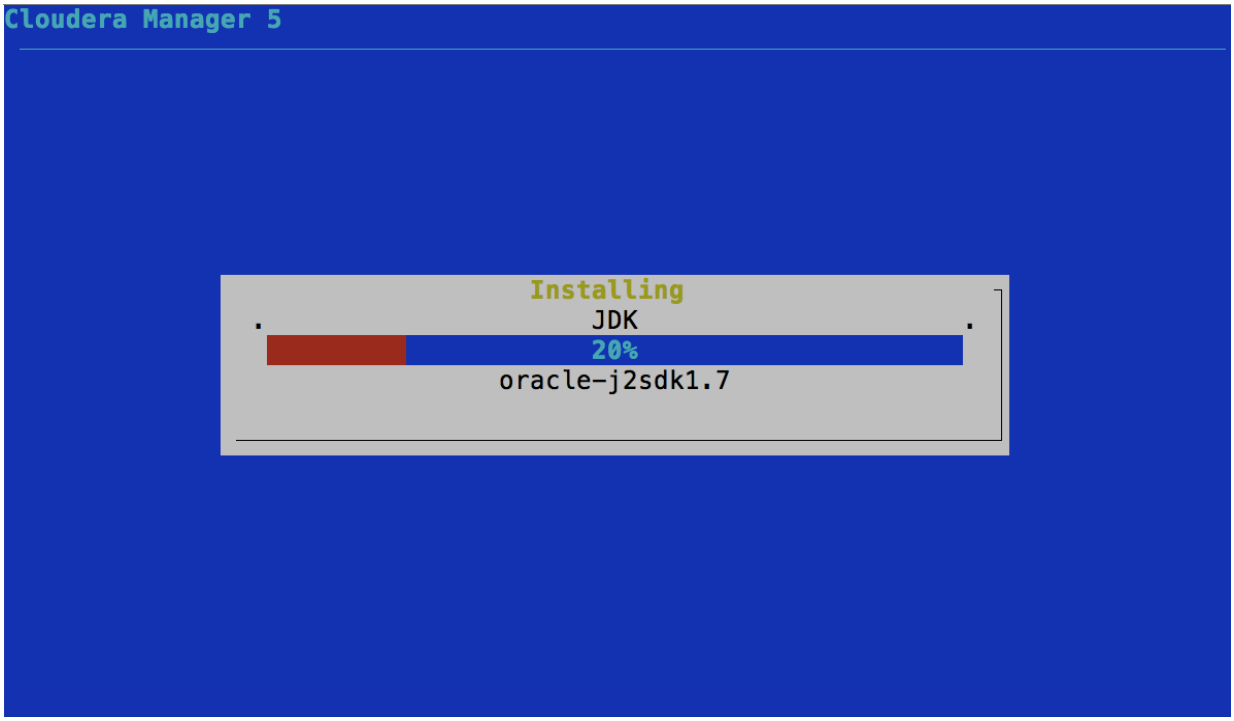
Download Cloudera Manager

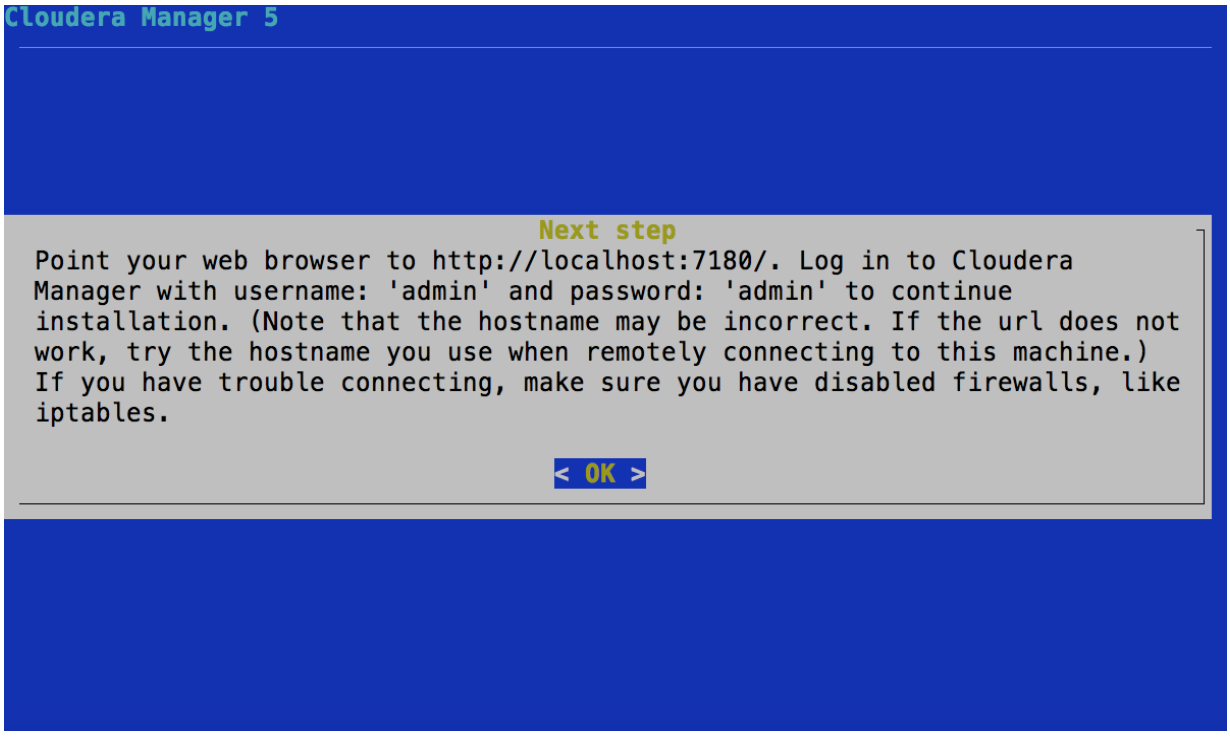
1) Type command >wget http://archive.cloudera.com/cm5/installer/latest/cloudera-manager-installer.bin

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

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



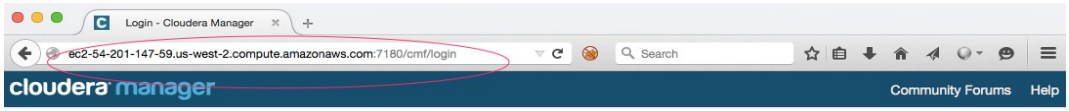

Cloudera Manager 5
Installing
JDK
20%
oracle-j2sdk1.7

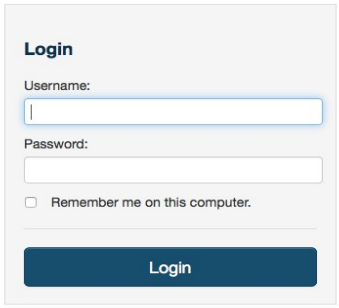

Cloudera Manager 5
Next step
Point your web browser to http://localhost:7180/. Log in to Cloudera Manager with username: 'admin' and password: 'admin' to continue installation. (Note that the hostname may be incorrect. If the url does not work, try the hostname you use when remotely connecting to this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

Login to Cloudera Manager

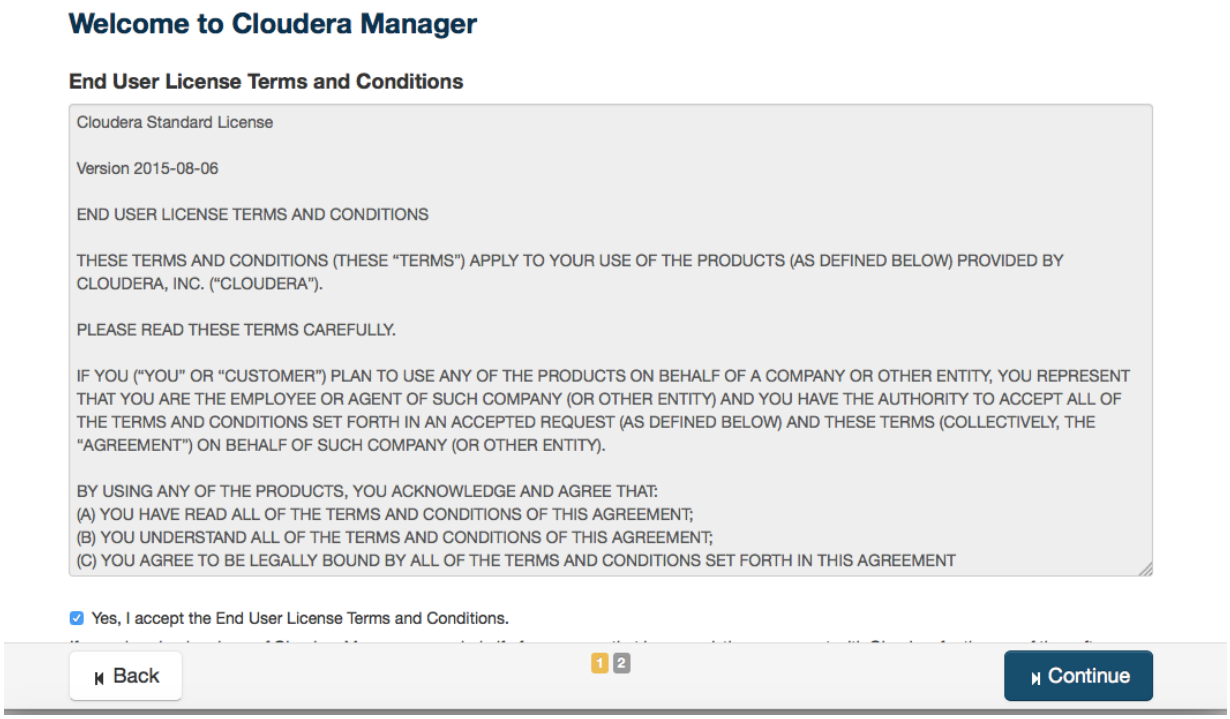
Wait several minutes for the Cloudera Manager Server to complete its startup.

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

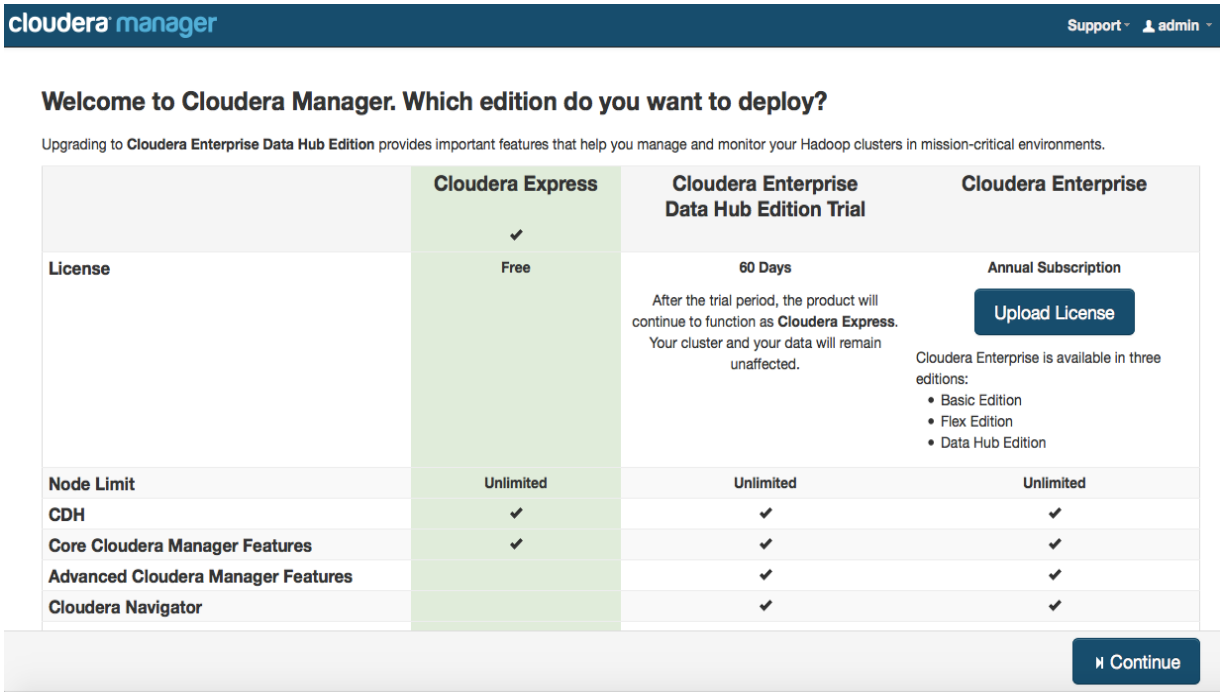


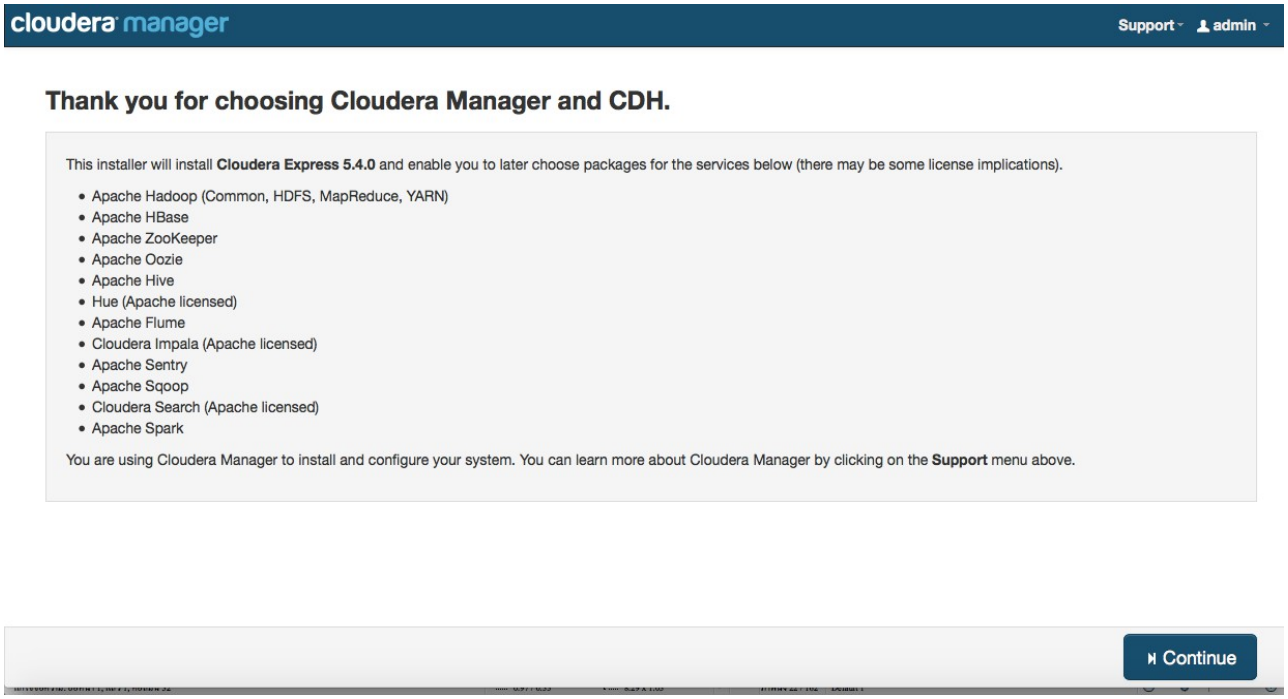


Accept the User License terms

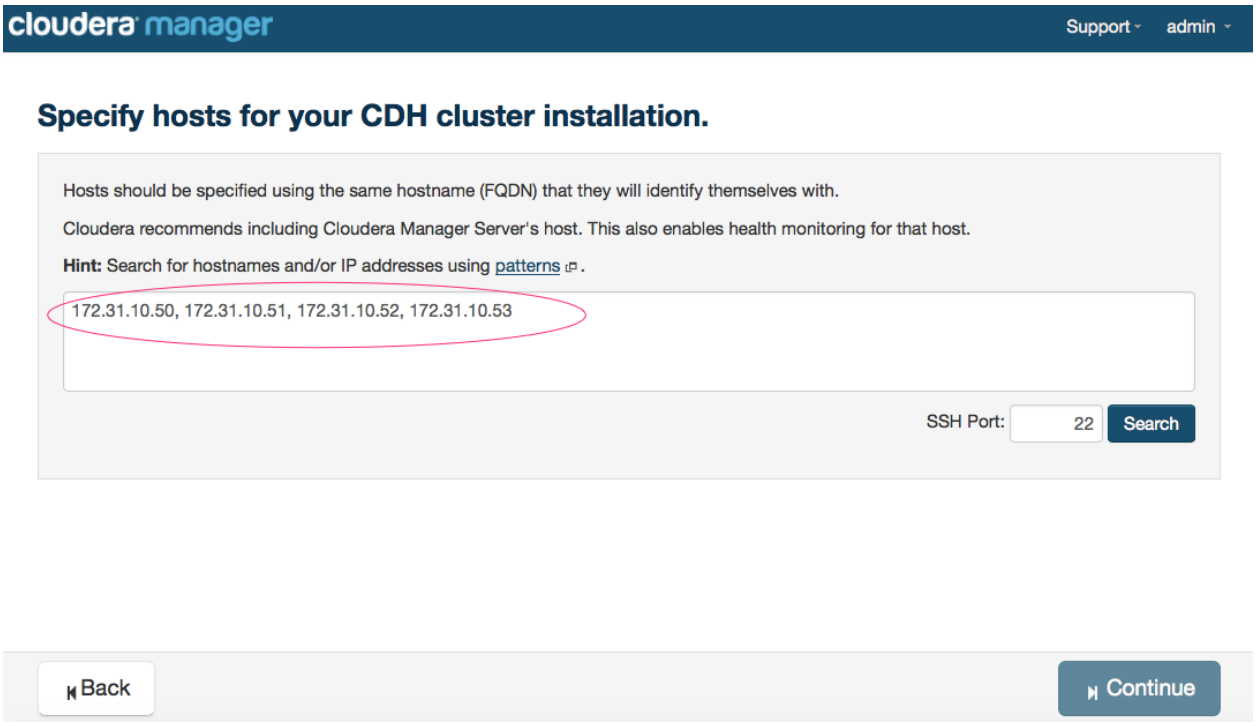


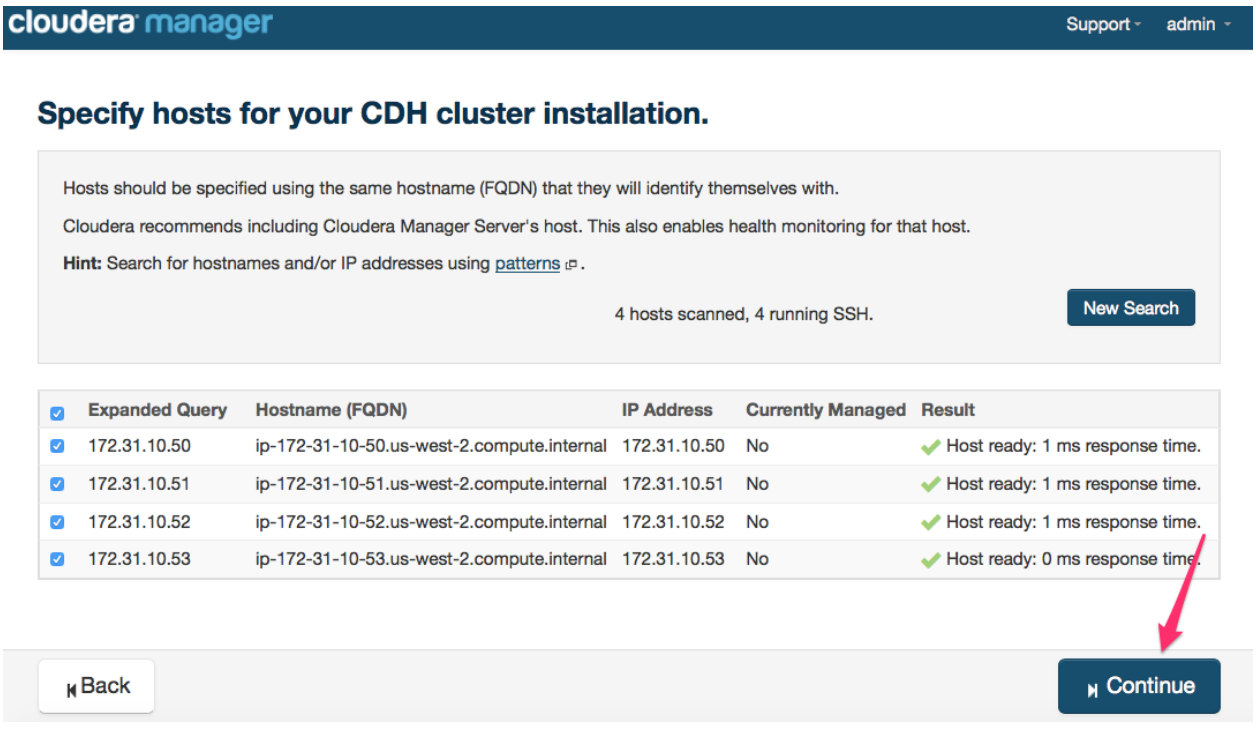
Select Cloudera Express Edition

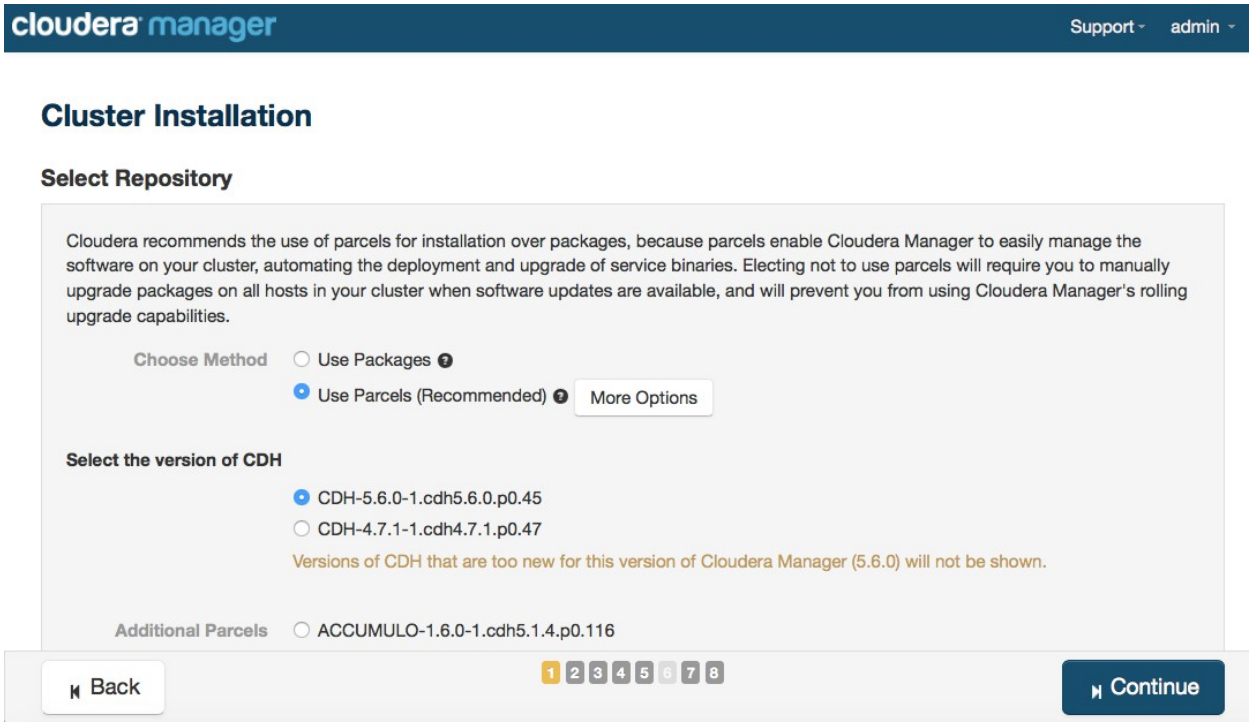




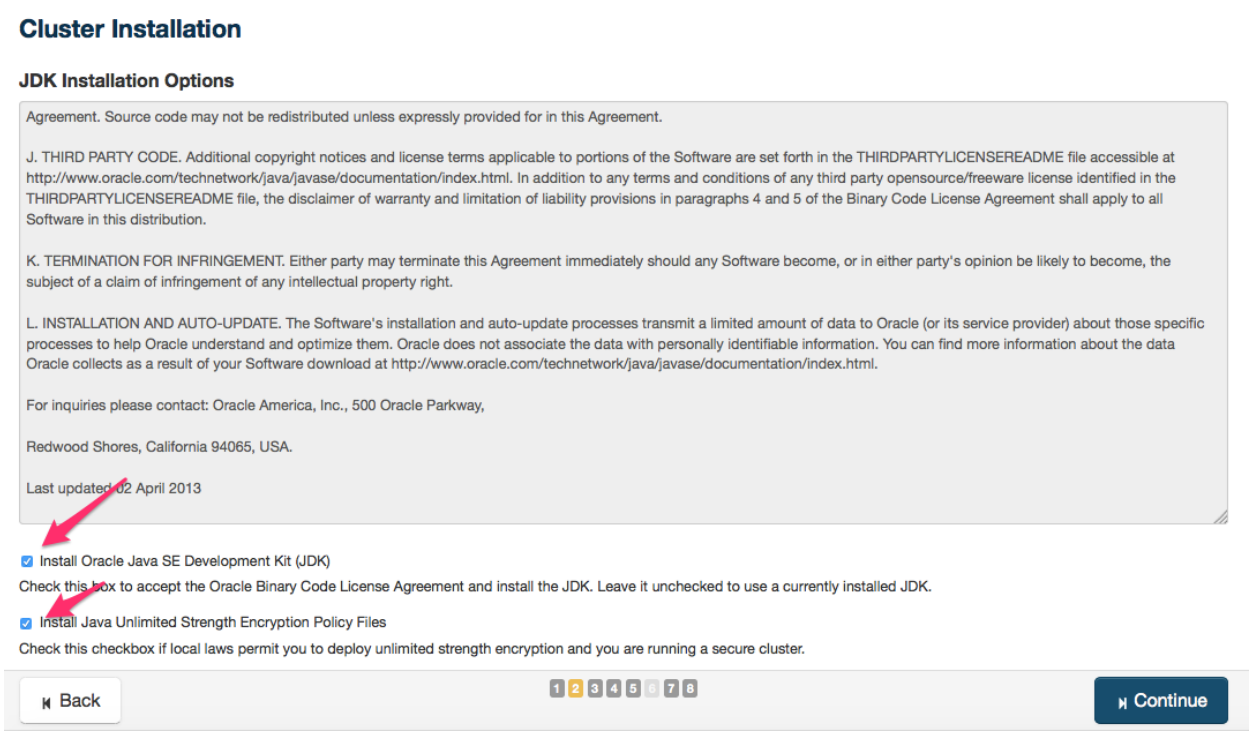
Provide your 4 instances <private ip> addresses in the cluster

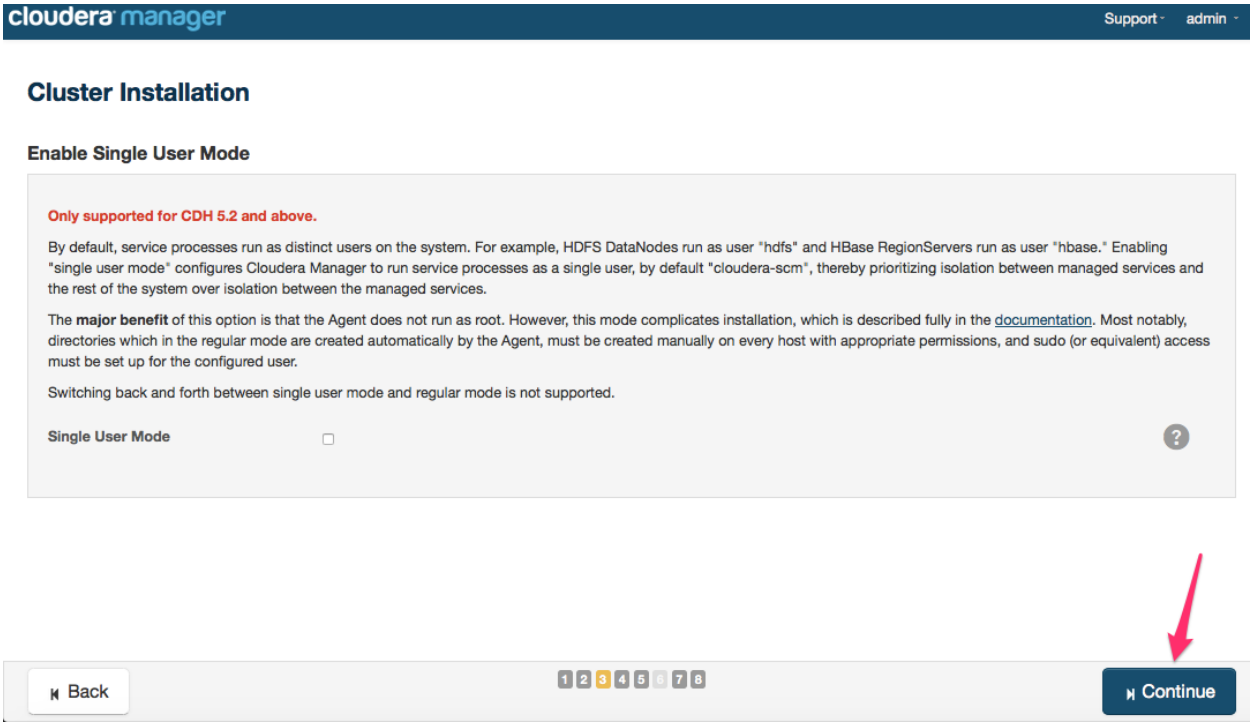




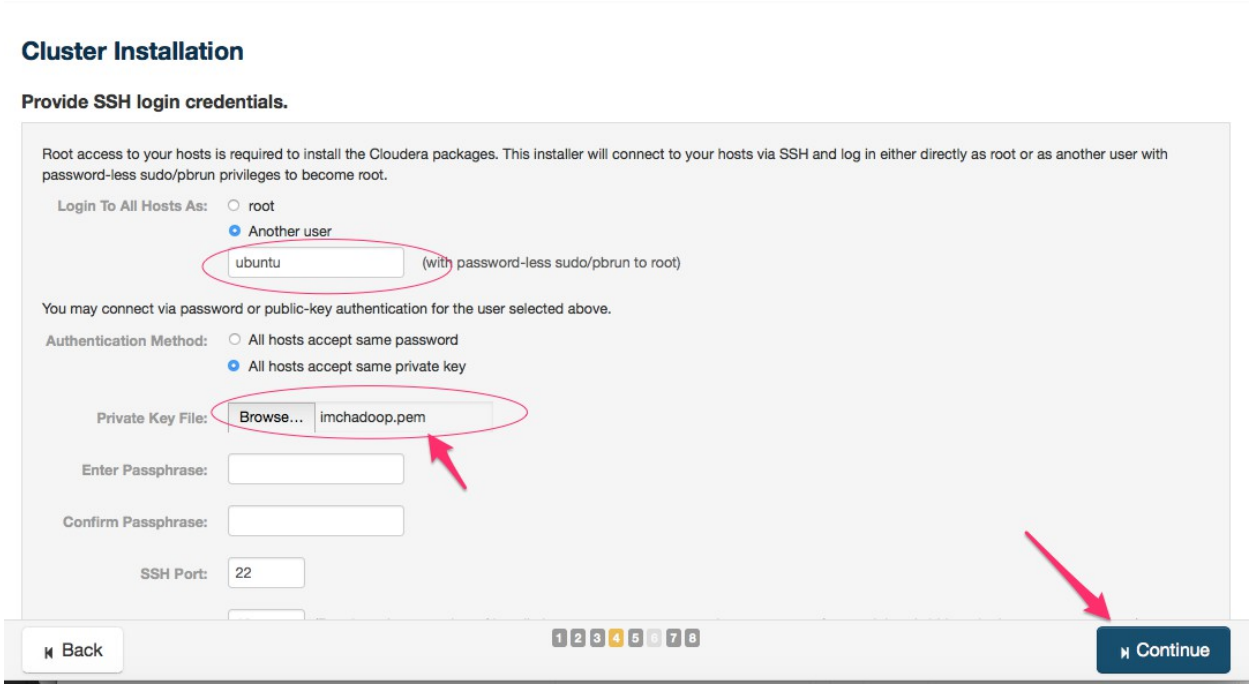


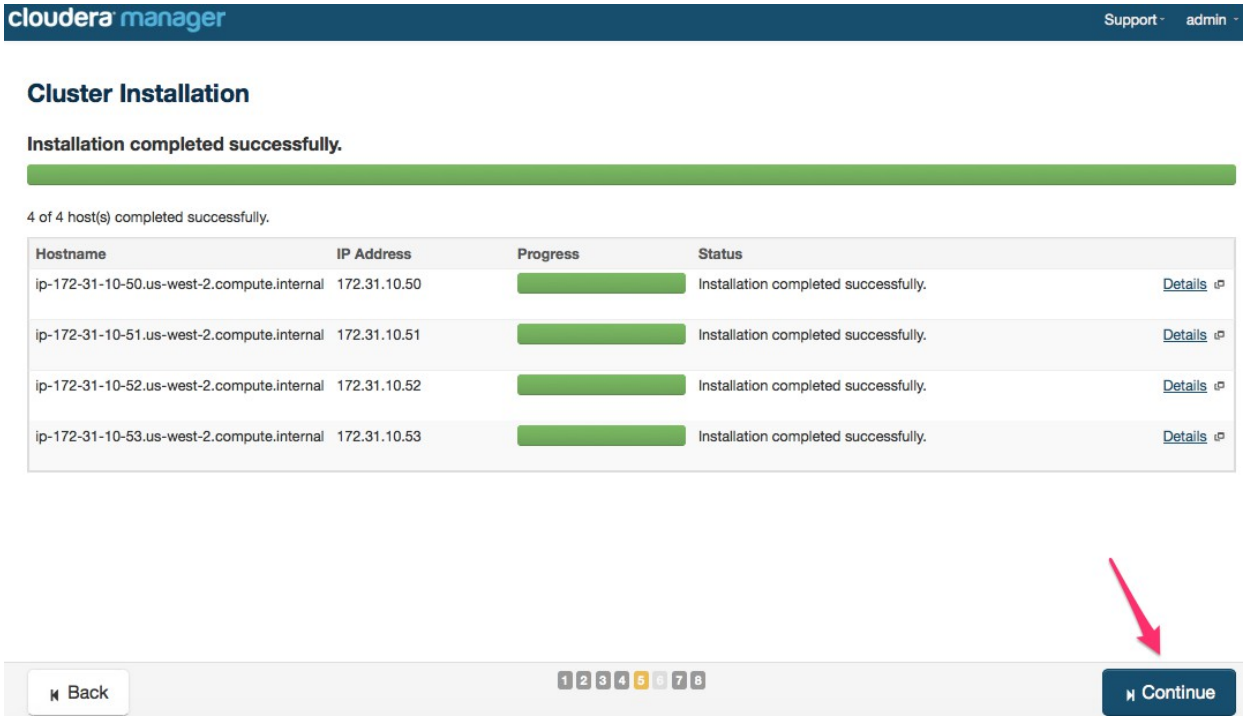
Select install Oracle Java & Java unlimited

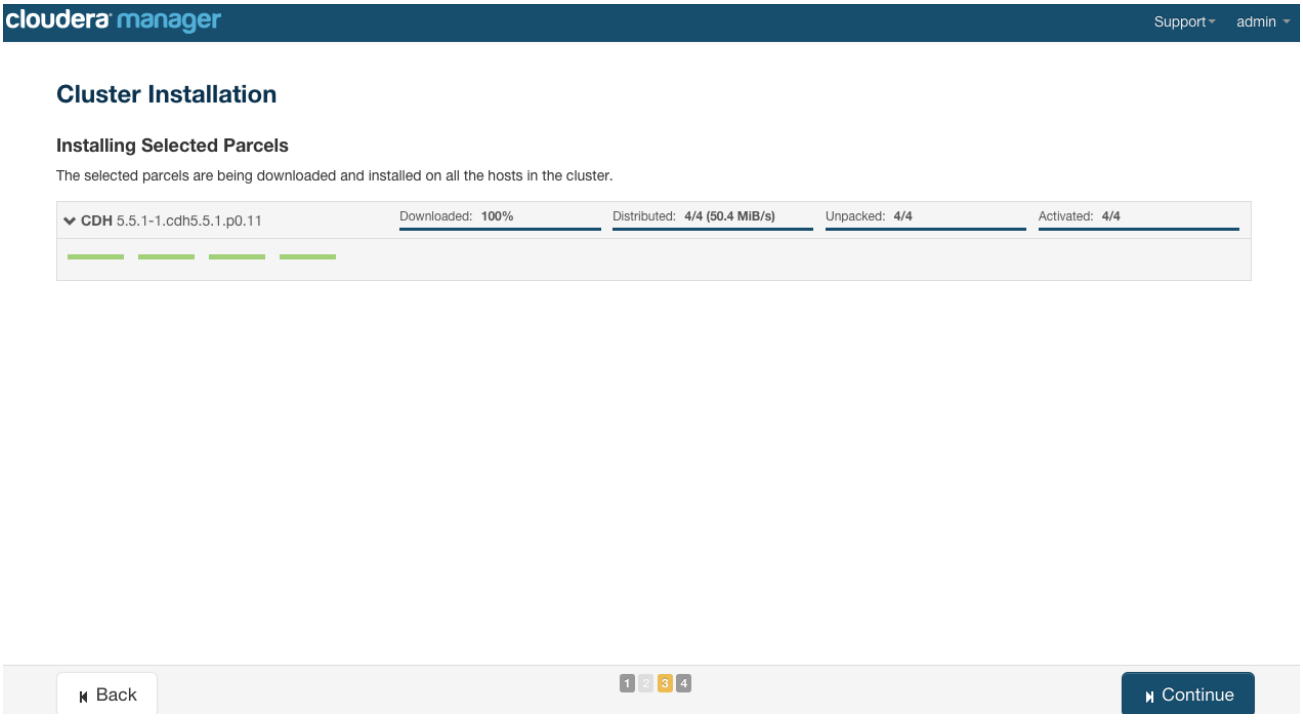


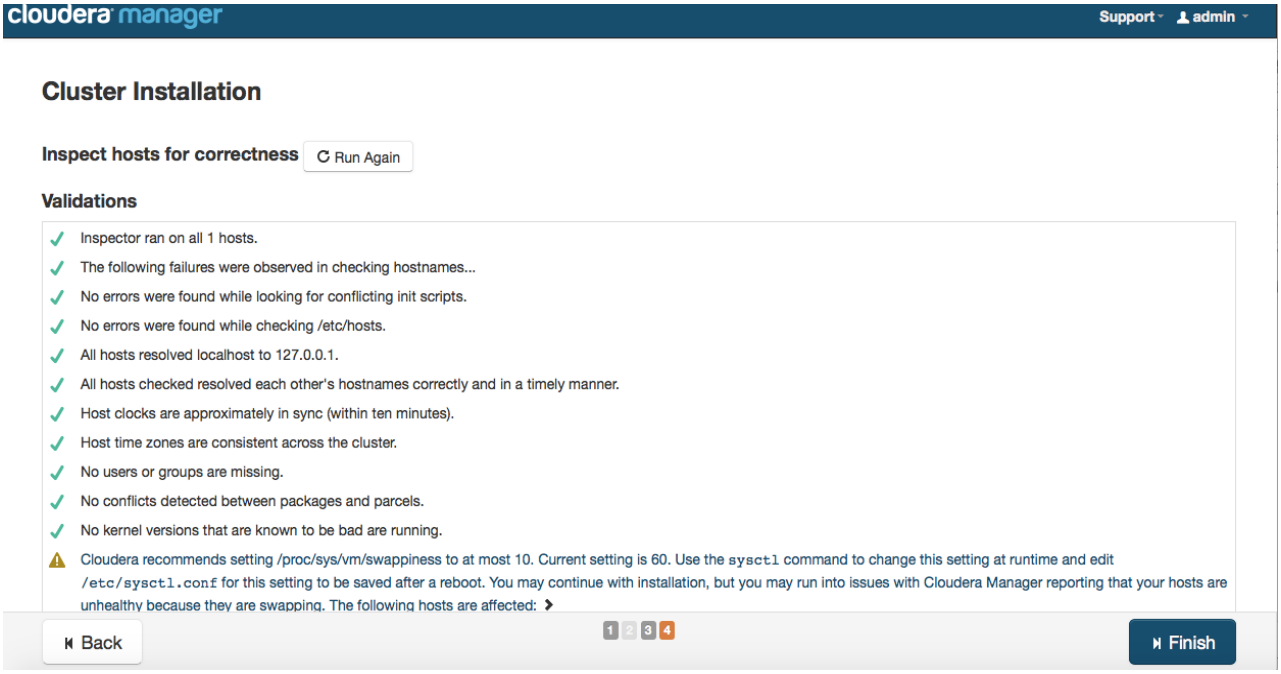


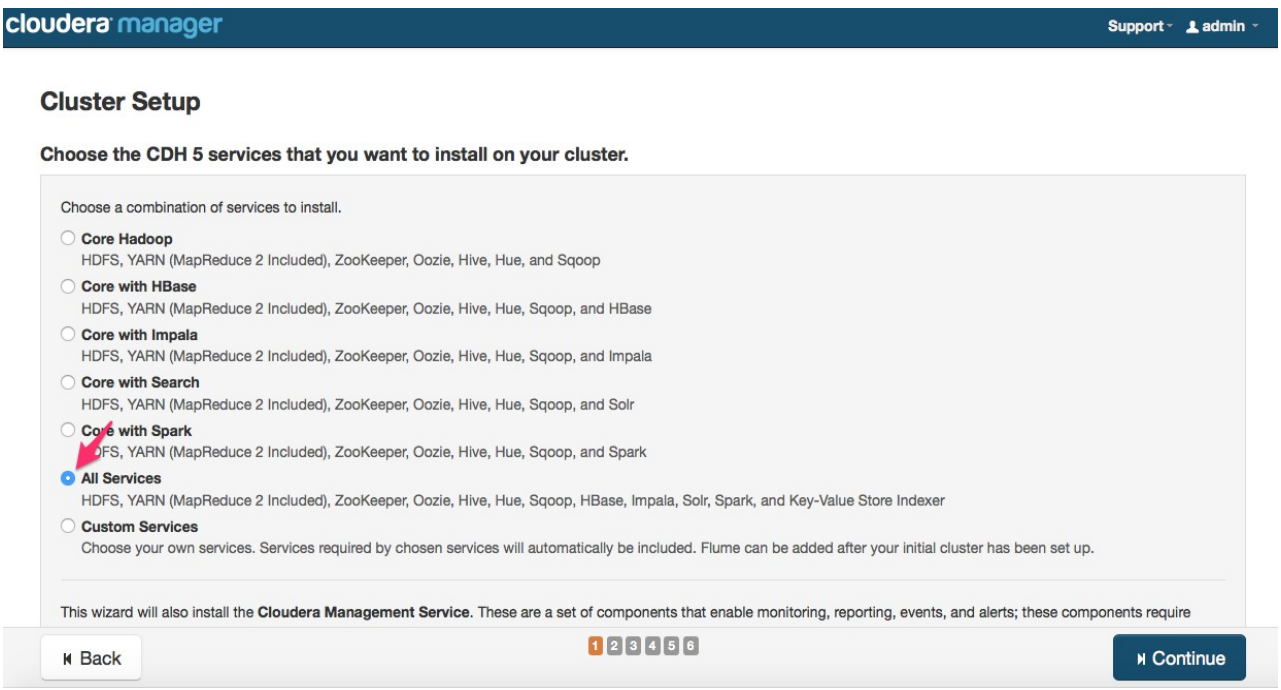
Define user as ubuntu & Browse the private key (imchadoop.pem) file which we have downloaded in the previous part. Keep Passphrase as blank

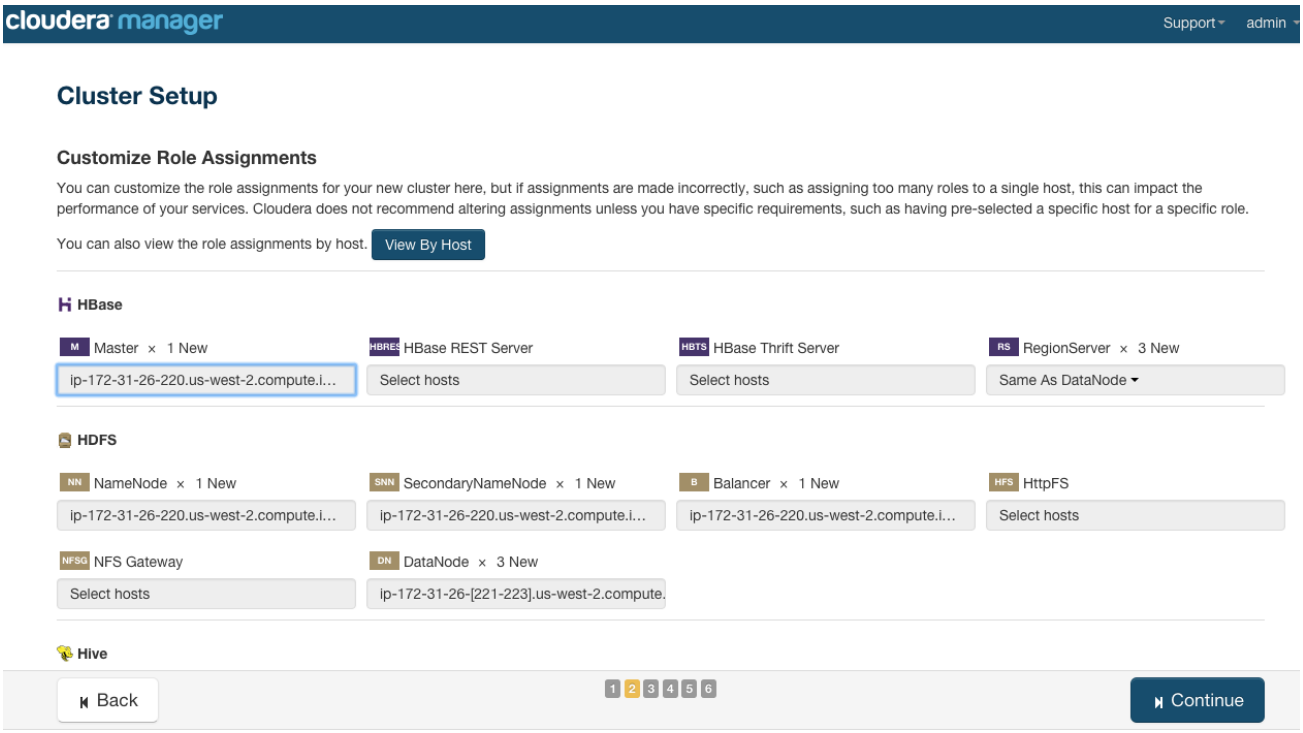


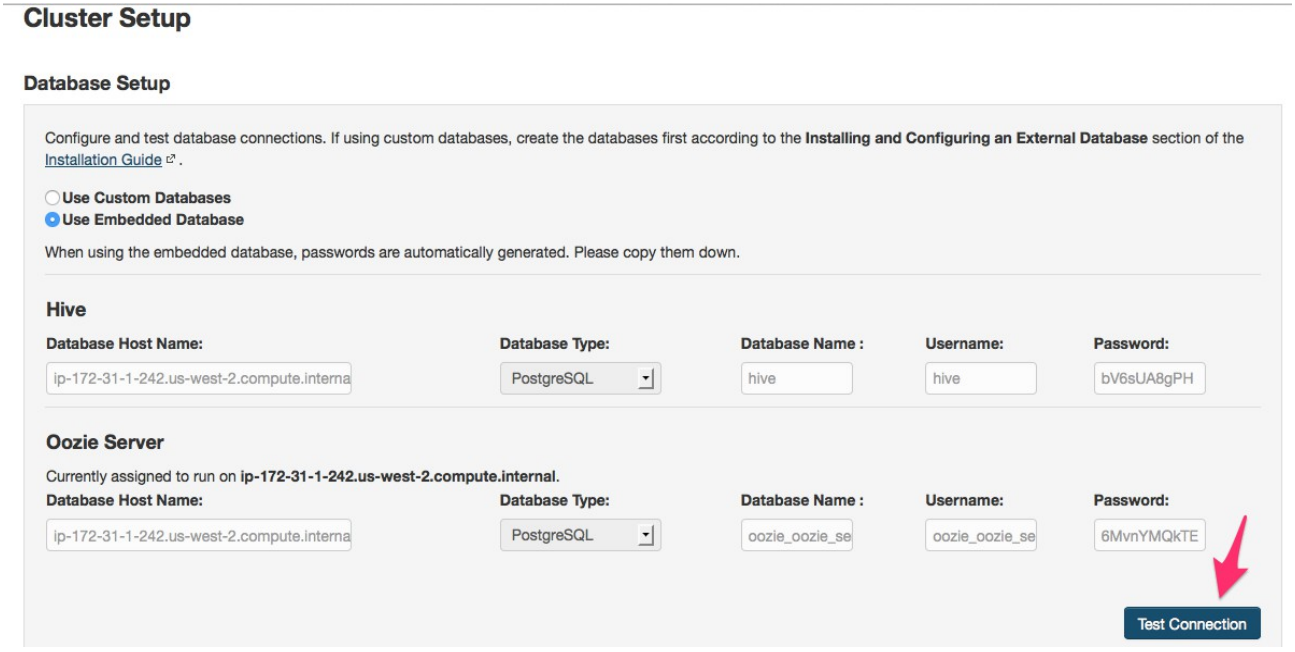

cloudera manager
Support
admin
Cluster Installation
Installation completed successfully.
4 of 4 host(s) completed successfully.
Hostname
IP Address
Progress
Status
ip-172-31-10-50.us-west-2.compute.internal
172.31.10.50
Installation completed successfully.
Details
ip-172-31-10-51.us-west-2.compute.internal
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

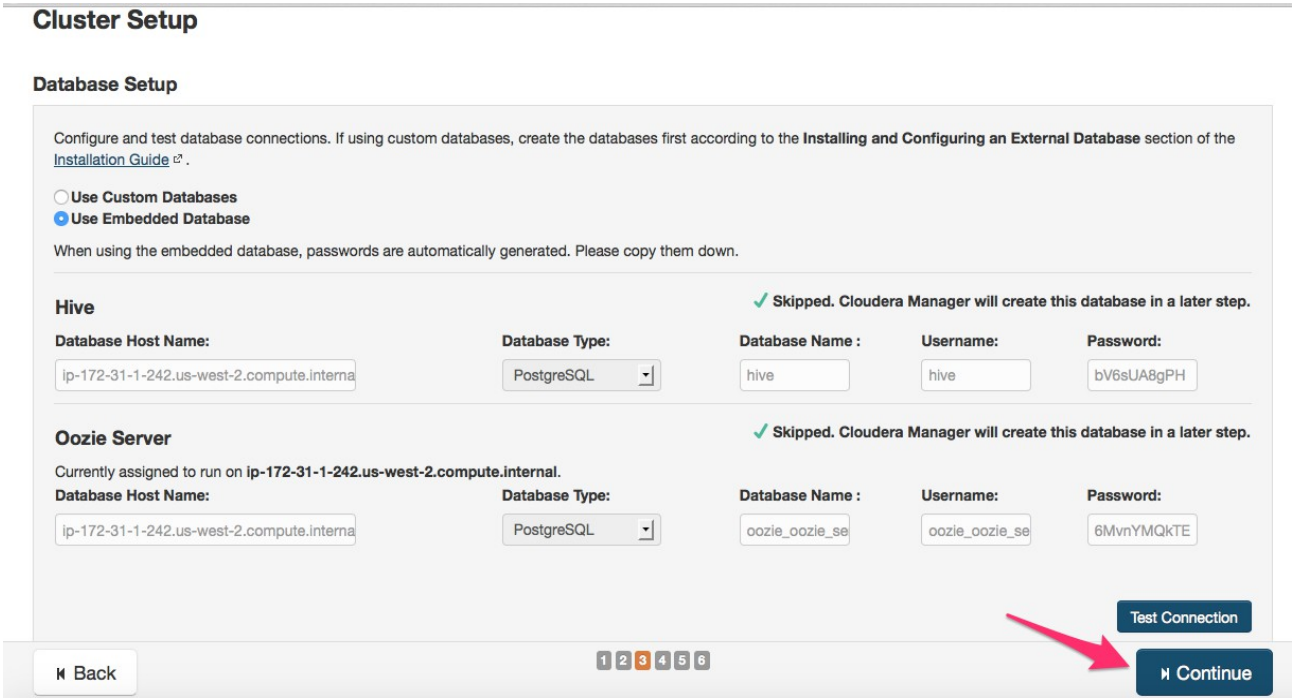

cloudera manager
Support
admin
Cluster Installation
Installing Selected Parcels
The selected parcels are being downloaded and installed on all the hosts in the cluster.
CDH 5.5.1-1.cdh5.5.1.p0.11
Downloaded: 100%
Distributed: 4/4 (50.4 MiB/s)
Unpacked: 4/4
Activated: 4/4
Back
1 2 3 4
Continue

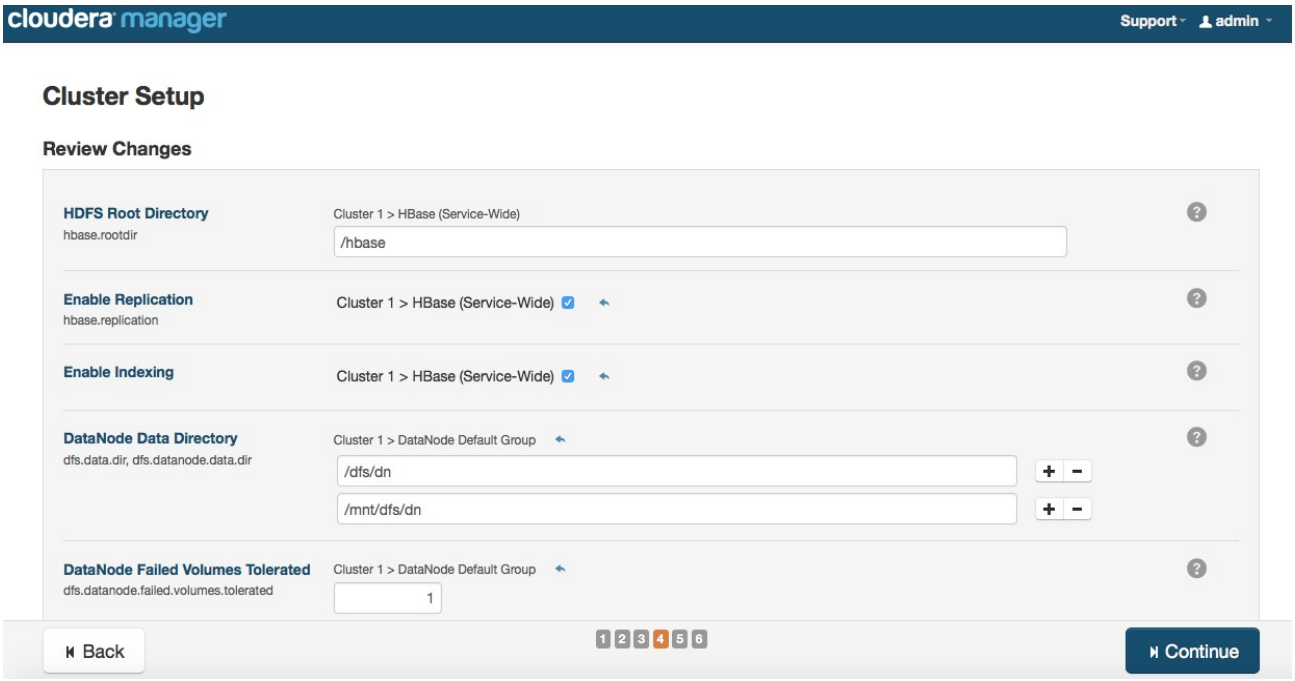

cloudera manager
Support
admin
Cluster Installation
Inspect hosts for correctness
Run Again
Validations
Inspector ran on all 1 hosts.
The following failures were observed in checking hostnames...
No errors were found while looking for conflicting init scripts.
No errors were found while checking /etc/hosts.
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

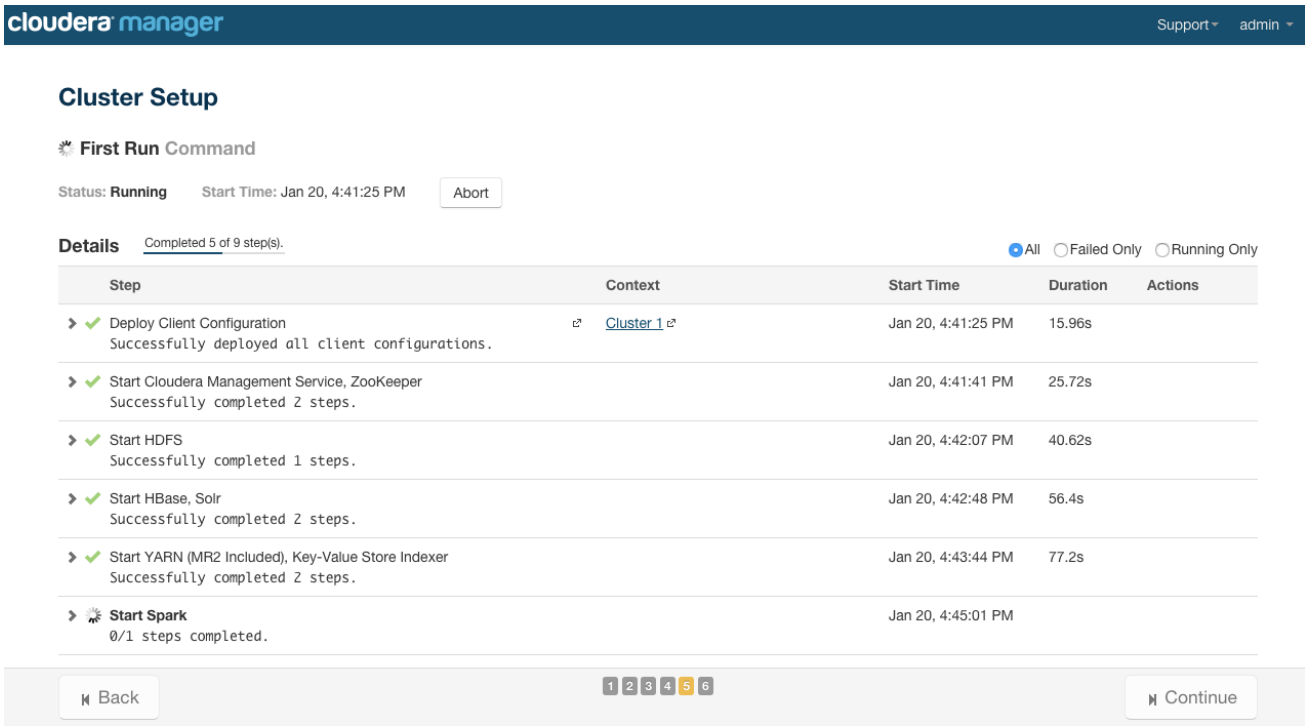

cloudera manager
Support
admin
Cluster Setup
Choose the CDH 5 services that you want to install on your cluster.
Choose a combination of services to install.
Core Hadoop
HDFS, YARN (MapReduce 2 Included), ZooKeeper, Oozie, Hive, Hue, and Sqoop
Core with HBase
HDFS, YARN (MapReduce 2 Included), ZooKeeper, 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


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
ip-172-31-26-220.us-west-2.compute.i...
HBase REST Server
Select hosts
HBase Thrift Server
Select hosts
RegionServer × 3 New
Same As DataNode
HDFS
NameNode × 1 New
ip-172-31-26-220.us-west-2.compute.i...
SecondaryNameNode × 1 New
ip-172-31-26-220.us-west-2.compute.i...
Balancer × 1 New
ip-172-31-26-220.us-west-2.compute.i...
HttpFS
Select hosts
NFS Gateway
Select hosts
DataNode × 3 New
ip-172-31-26-[221-223].us-west-2.compute.
Hive
Back
1 2 3 4 5 6
Continue

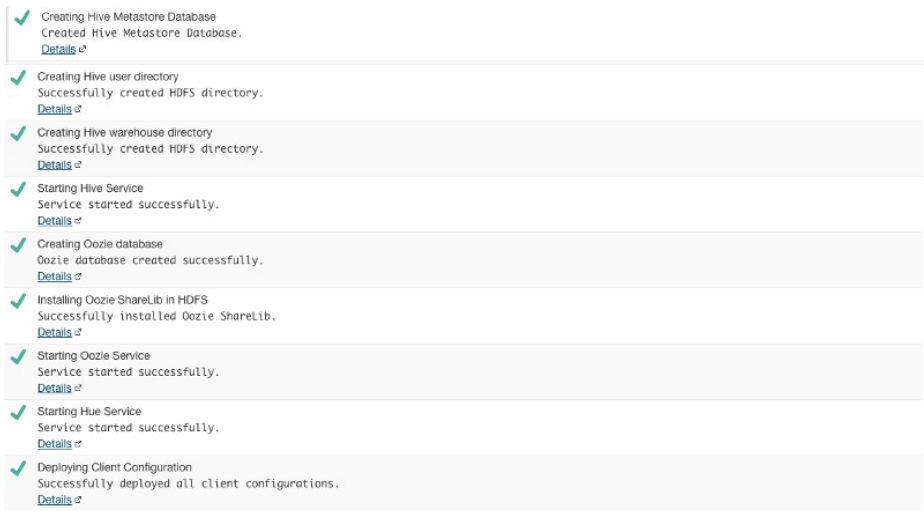

Cluster Setup
Database Setup
Configure and test database connections. If using custom databases, create the databases first according to the Installing and Configuring an External Database section of the Installation Guide .
Use Custom Databases
Use Embedded Database
When using the embedded database, passwords are automatically generated. Please copy them down.
Hive
Database Host Name:
ip-172-31-1-242.us-west-2.compute.interna
Database Type:
PostgreSQL
Database Name :
hive
Username:
hive
Password:
bV6sUA8gPH
Oozie Server
Currently assigned to run on ip-172-31-1-242.us-west-2.compute.internal.
Database Host Name:
ip-172-31-1-242.us-west-2.compute.interna
Database Type:
PostgreSQL
Database Name :
oozie_oozie_se
Username:
oozie_oozie_se
Password:
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

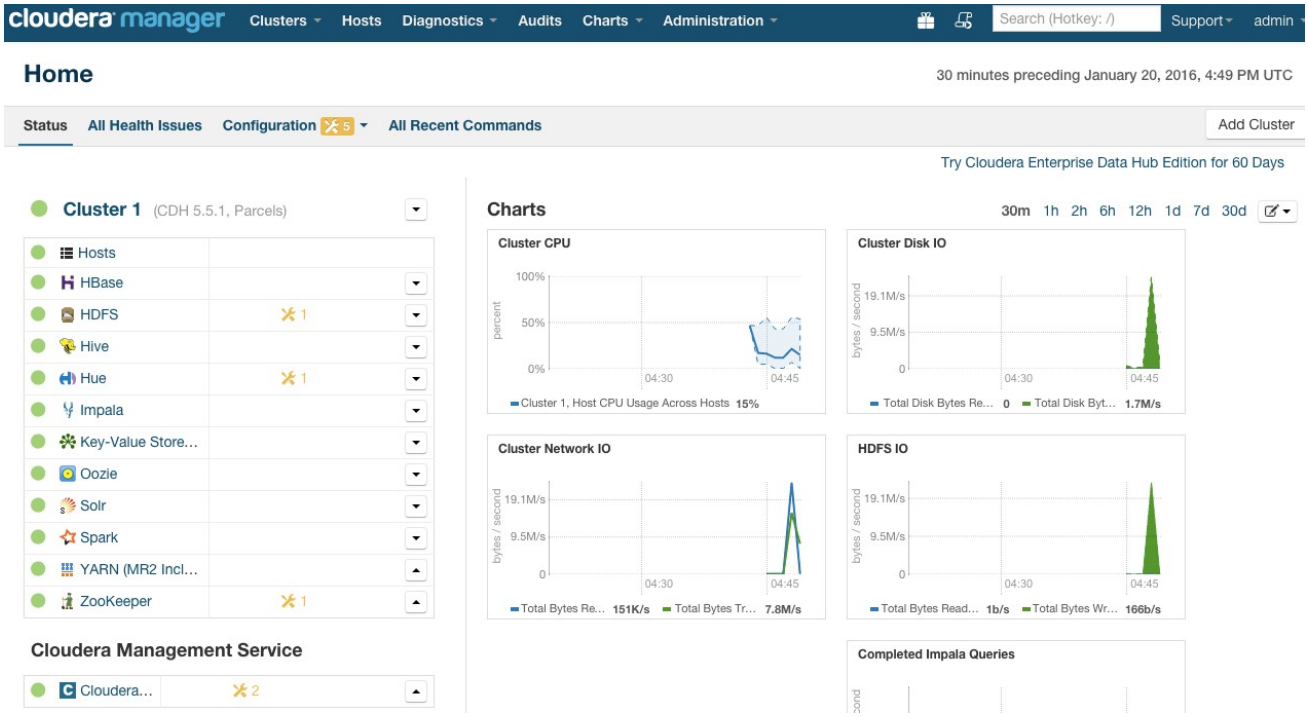

cloudera manager
Support
admin
Cluster Setup
Review Changes
HDFS Root Directory
hbase.rootdir
Cluster 1 > HBase (Service-Wide)
/hbase
Enable Replication
hbase.replication
Cluster 1 > HBase (Service-Wide)
Enable Indexing
Cluster 1 > HBase (Service-Wide)
DataNode Data Directory
dfs.data.dir, dfs.datanode.data.dir
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
First Run Command
Status: Running
Start Time: Jan 20, 4:41:25 PM
Abort
Details
Completed 5 of 9 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.
Jan 20, 4:42:07 PM
40.62s
Start HBase, Solr
Successfully completed 2 steps.
Jan 20, 4:42:48 PM
56.4s
Start YARN (MR2 Included), Key-Value Store Indexer
Successfully completed 2 steps.
Jan 20, 4:43:44 PM
77.2s
Start Spark
0/1 steps completed.
Jan 20, 4:45:01 PM
Back
1 2 3 4 5 6
Continue

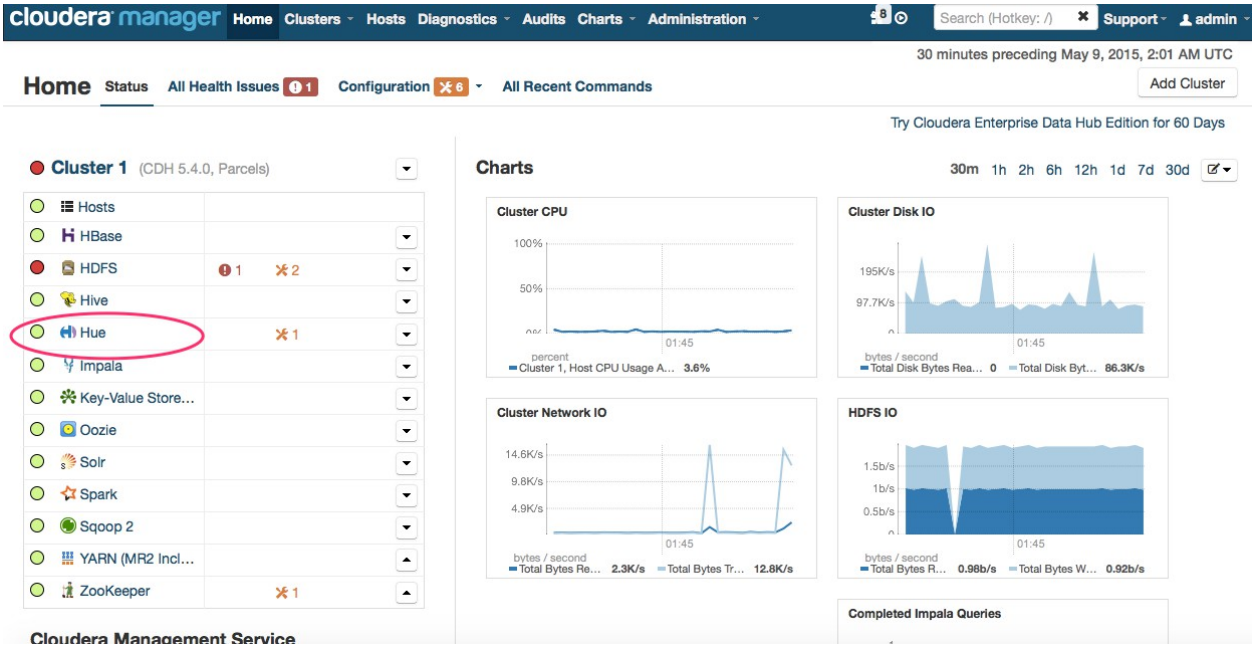
Finish


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

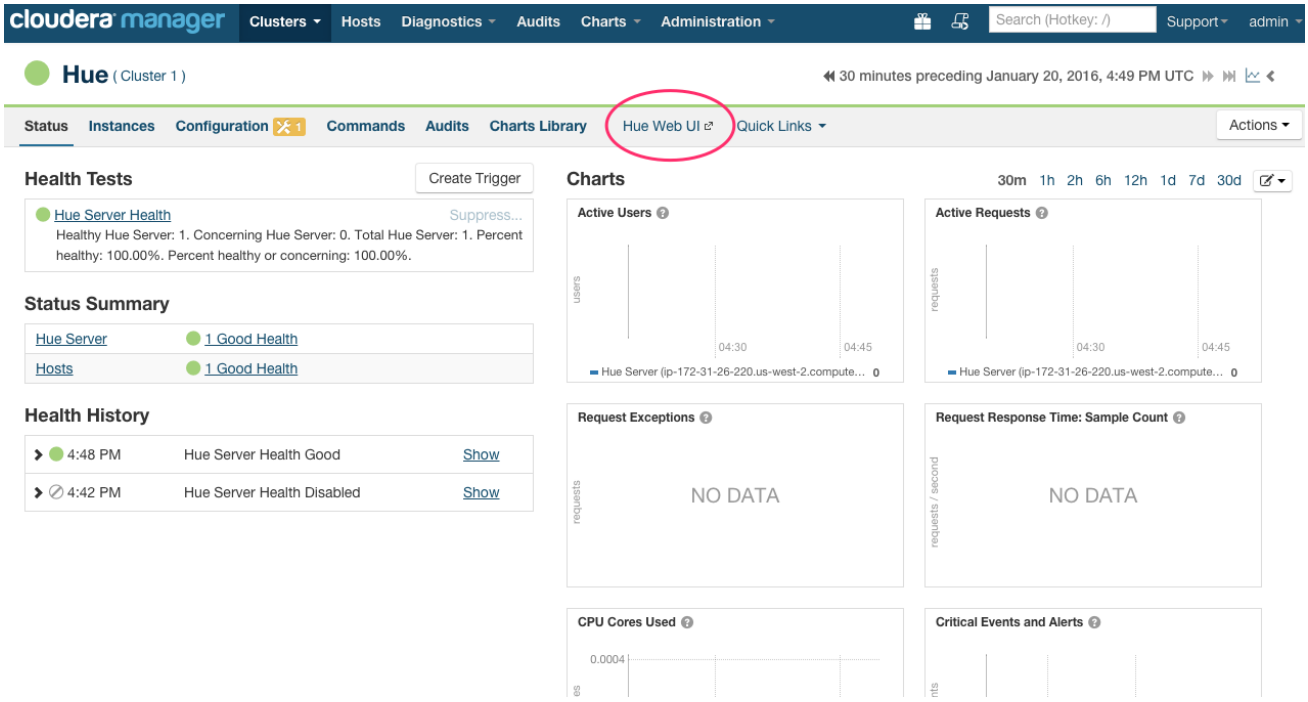

Cluster Setup
Congratulations!
The services are installed, configured, and running on your cluster.


cloudera manager
Clusters
Hosts
Diagnostics
Audits
Charts
Administration
Search (Hotkey: /)
Support
admin
Home
30 minutes preceding January 20, 2016, 4:49 PM UTC
Status
All Health Issues
Configuration
All Recent Commands
Add Cluster
Try Cloudera Enterprise Data Hub Edition for 60 Days
Cluster 1 (CDH 5.5.1, Parcels)
Hosts
HBase
HDFS
Hive
Hue
Impala
Key-Value Store...
Oozie
Solr
Spark
YARN (MR2 Incl...
ZooKeeper
Cloudera Management Service
Cloudera...
Charts
30m 1h 2h 6h 12h 1d 7d 30d
Cluster CPU
Cluster 1, Host CPU Usage Across Hosts 15%
Cluster Disk IO
Total Disk Bytes Re... 0
Total Disk Byt... 1.7M/s
Cluster Network IO
Total Bytes Re... 151K/s
Total Bytes Tr... 7.8M/s
HDFS IO
Total Bytes Read... 1b/s
Total Bytes Wr... 166b/s
Completed Impala Queries

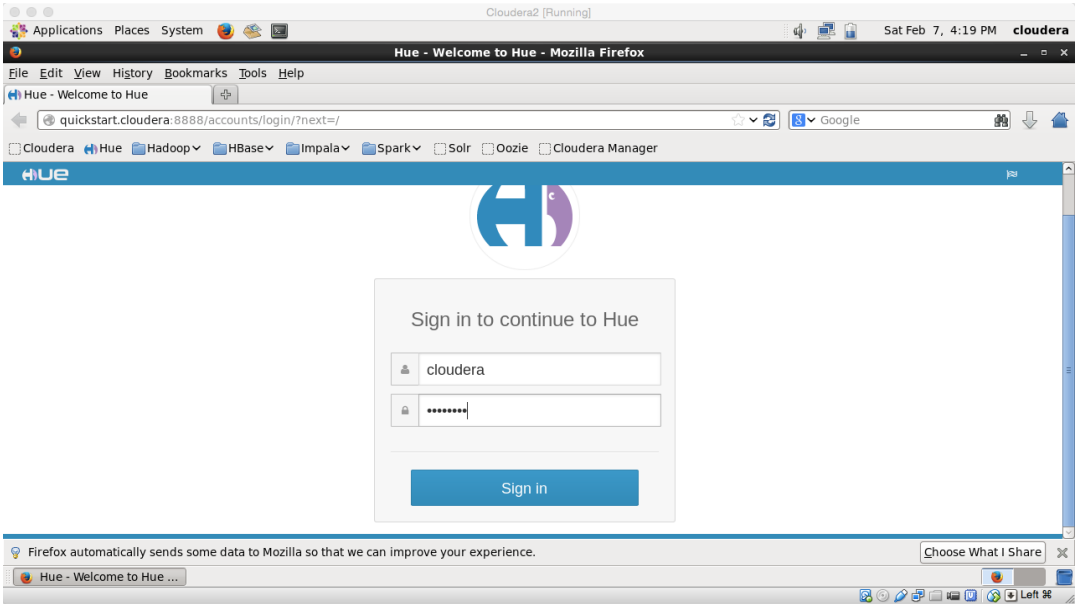
Running Hue


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
Configuration
All Recent Commands
Add Cluster
Try Cloudera Enterprise Data Hub Edition for 60 Days
Cluster 1 (CDH 5.4.0, Parcels)
Hosts
HBase
HDFS
Hive
Hue
Impala
Key-Value Store...
Oozie
Solr
Spark
Sqoop 2
YARN (MR2 Incl...
ZooKeeper
Cloudera Management Service
Charts
30m 1h 2h 6h 12h 1d 7d 30d
Cluster CPU
Cluster 1, Host CPU Usage A... 3.6%
Cluster Disk IO
Total Disk Bytes Rea... 0
Total Disk Byt... 86.3K/s
Cluster Network IO
Total Bytes Re... 2.3K/s
Total Bytes Tr... 12.6K/s
HDFS IO
Total Bytes R... 0.98b/s
Total Bytes W... 0.92b/s
Completed Impala Queries

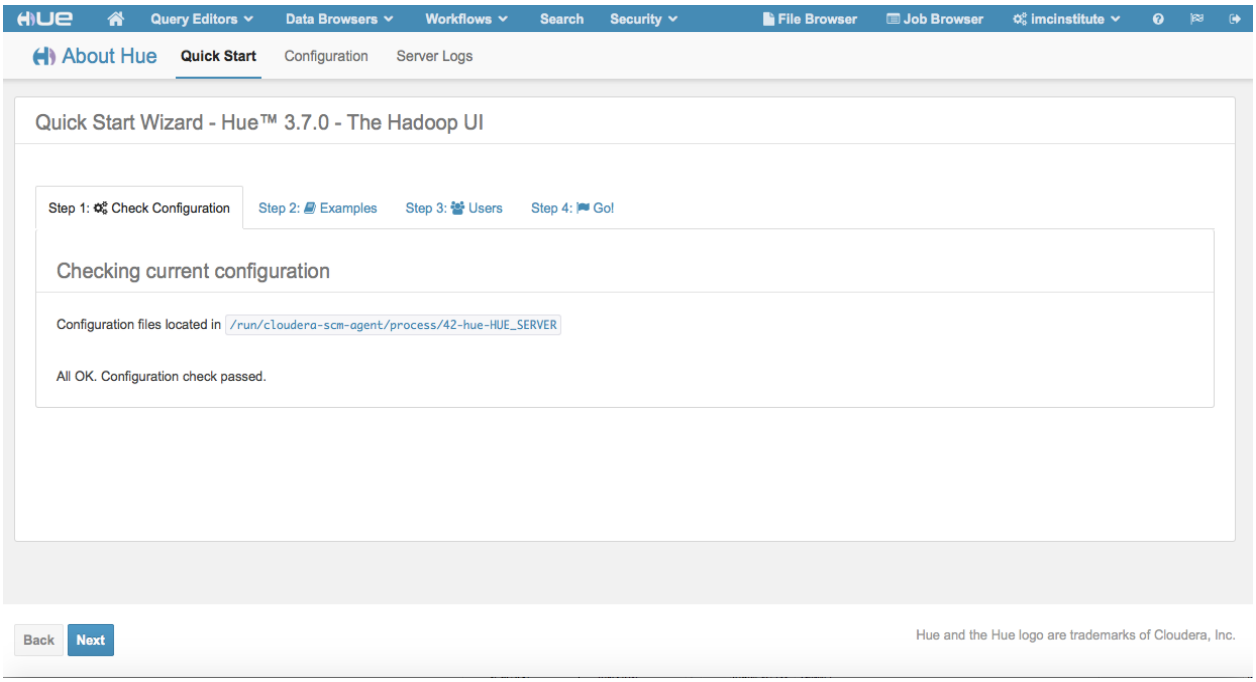
Running Hue

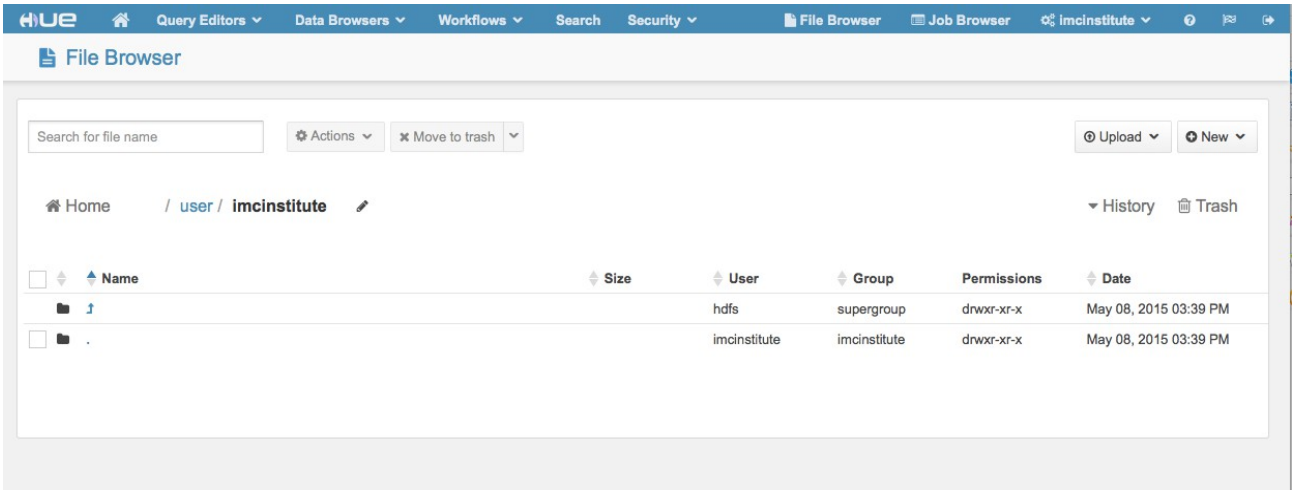


Sign in to Hue

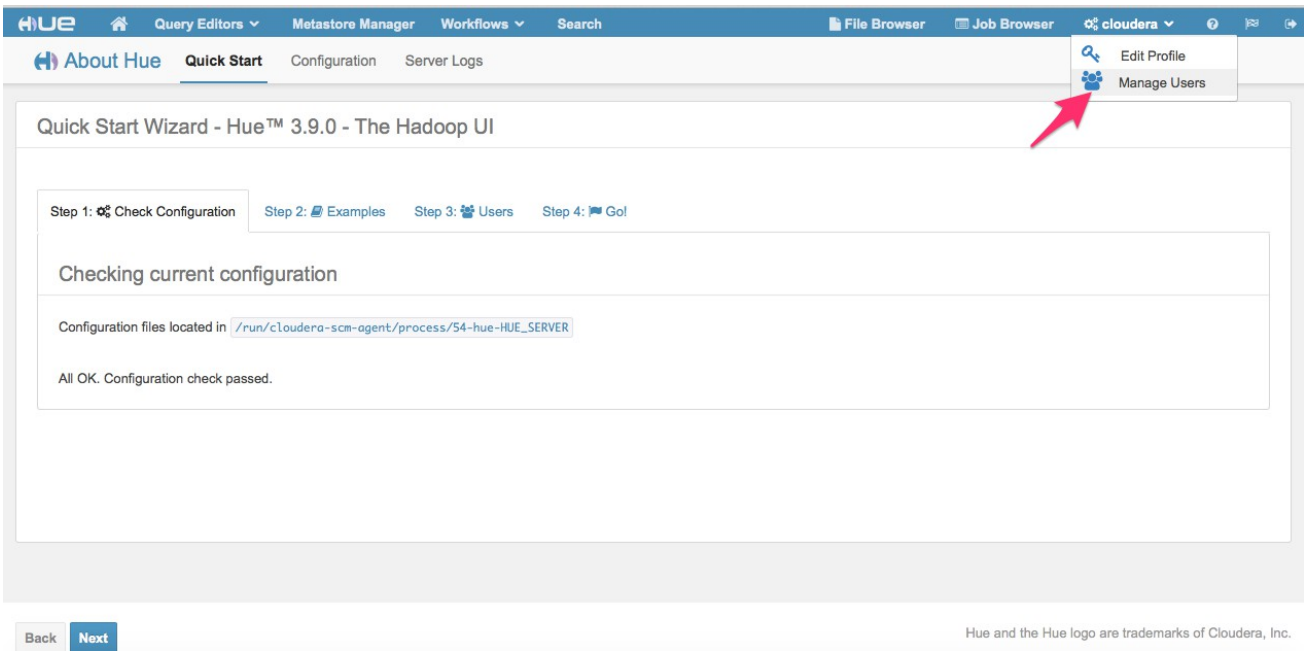


Starting Hue on Cloudera

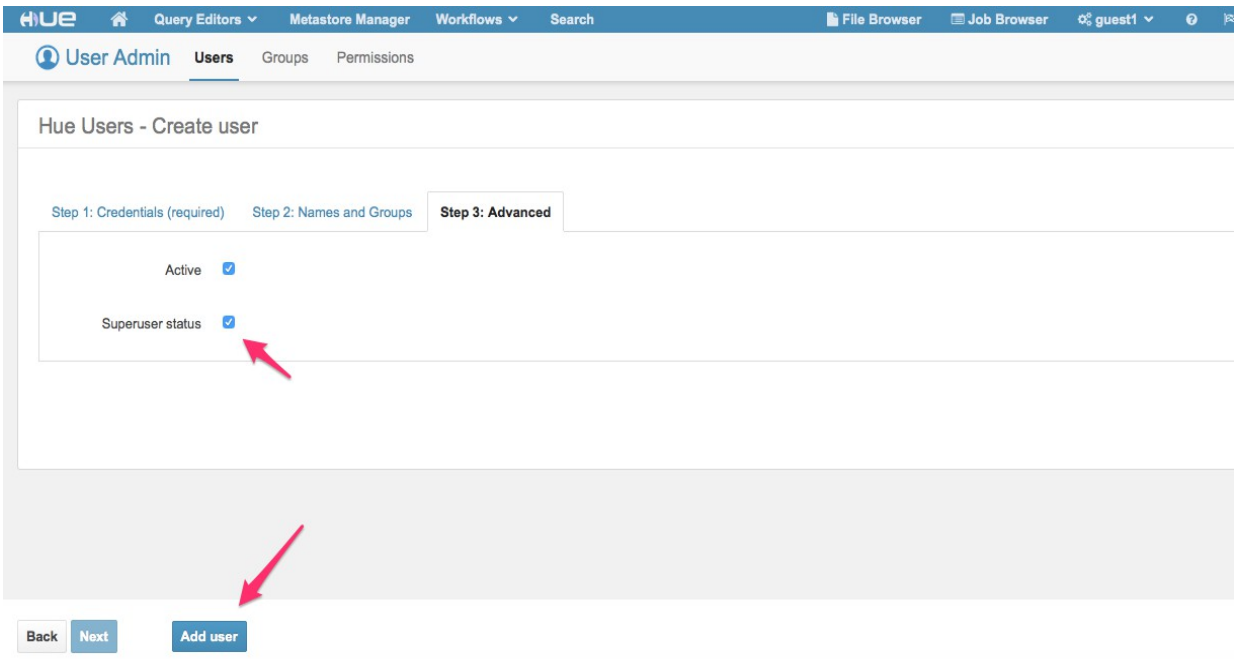




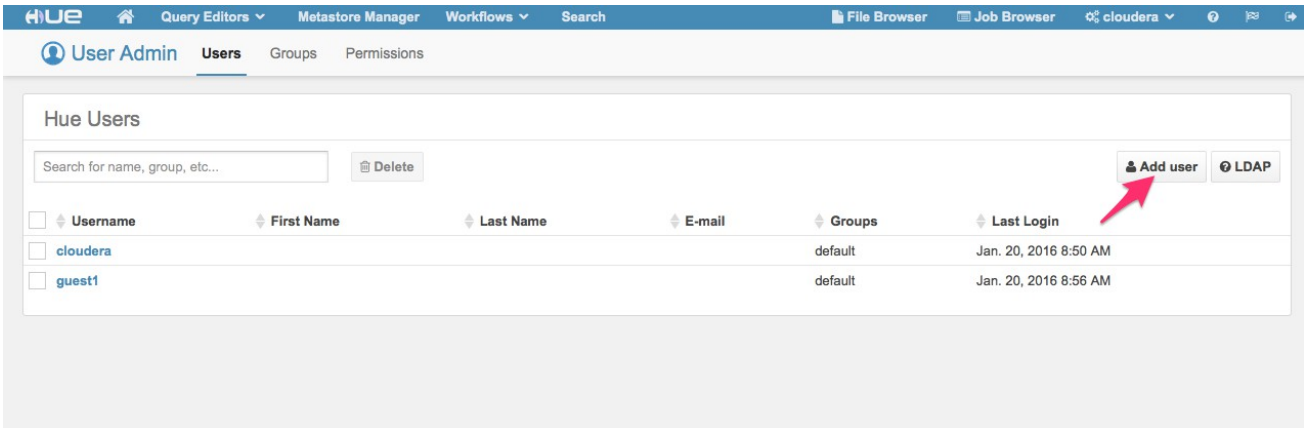
Select Manage User



Add a new user as a superuser

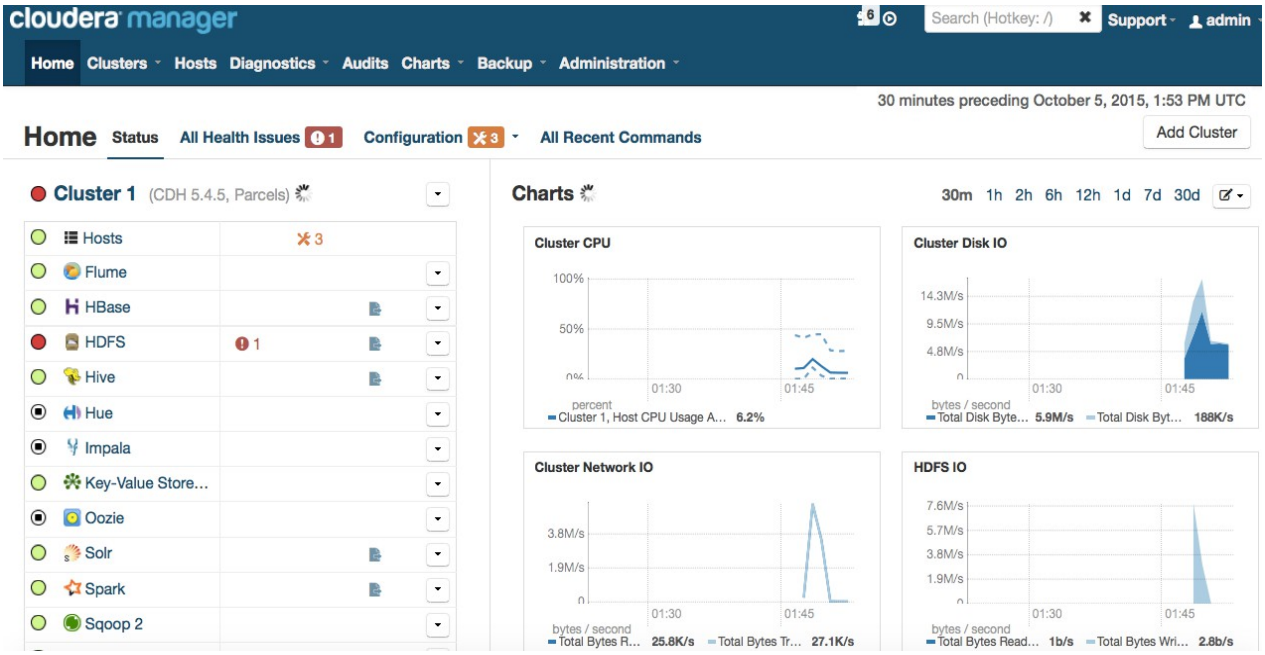


Add a new user

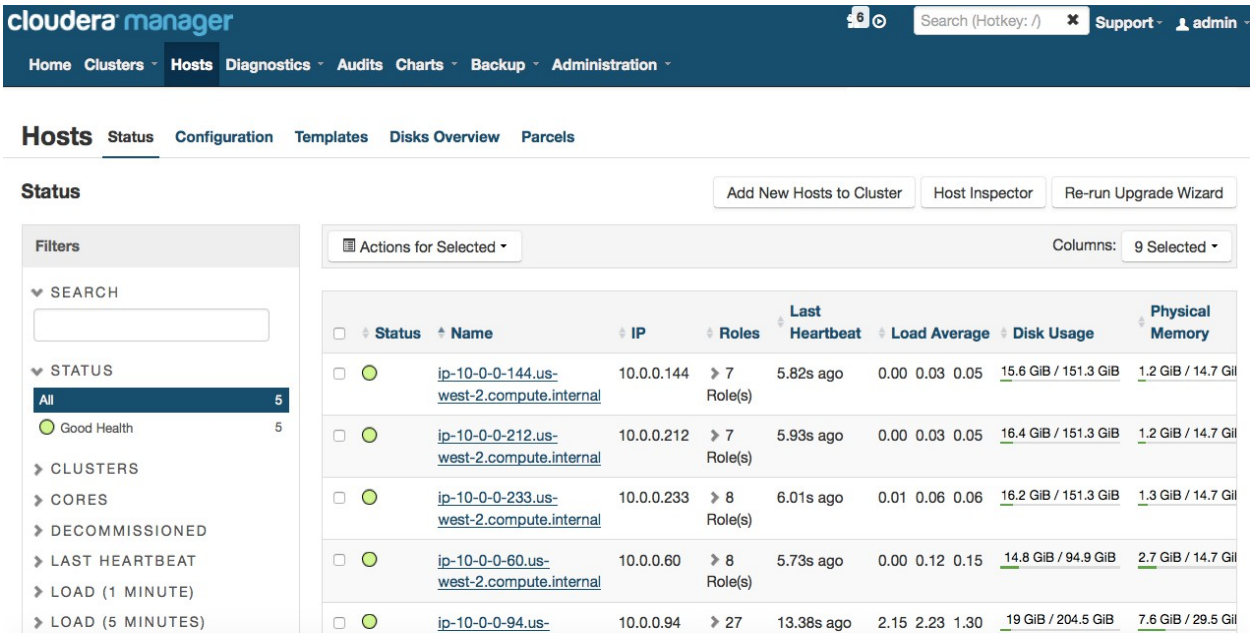


Cloudera Manager

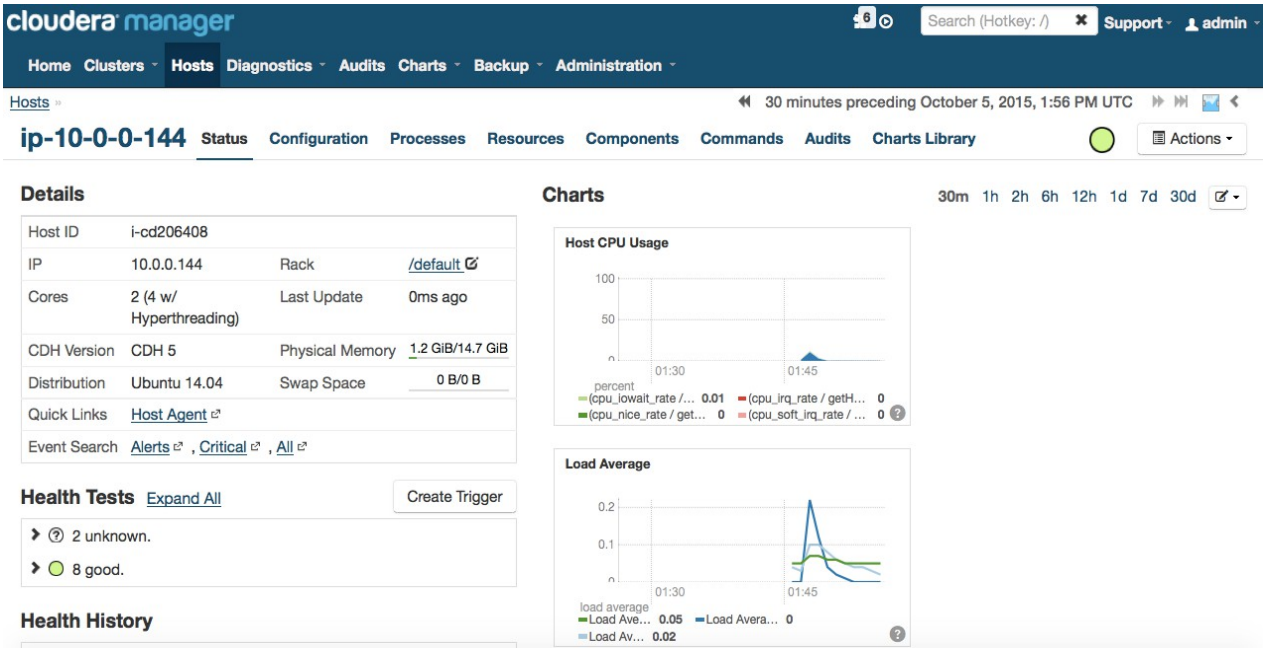
Cloudera Manager: Dashboard



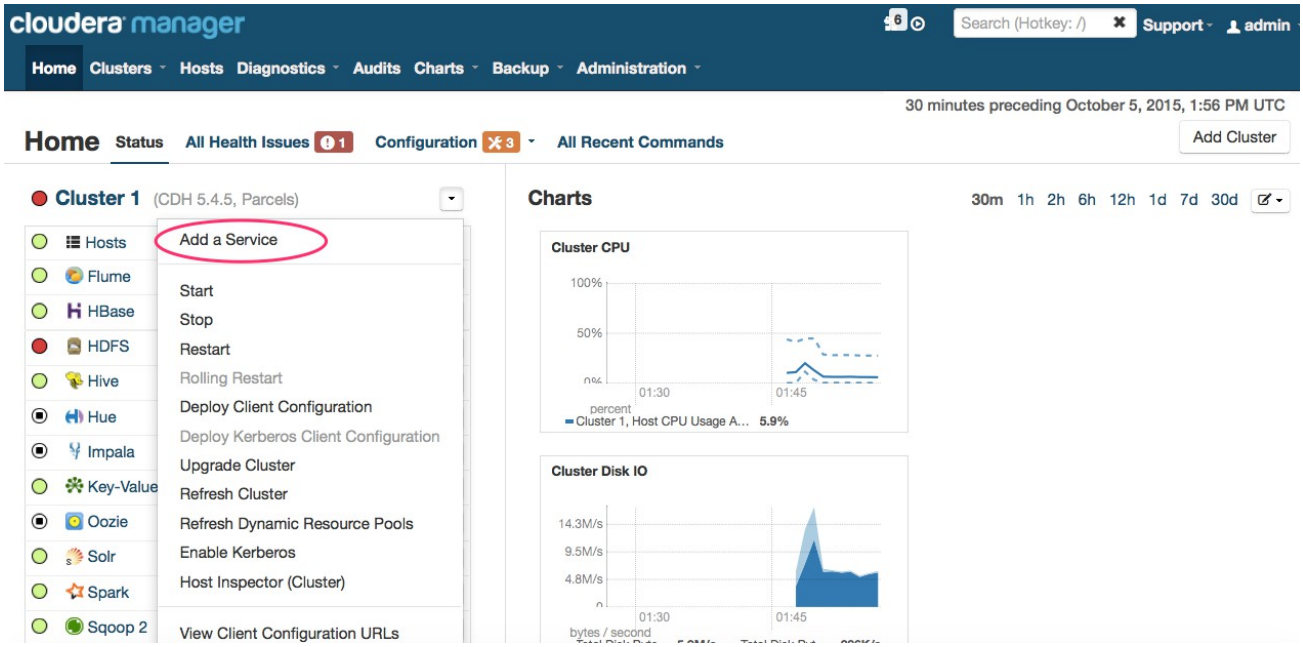
Hosts Management



Hosts Management

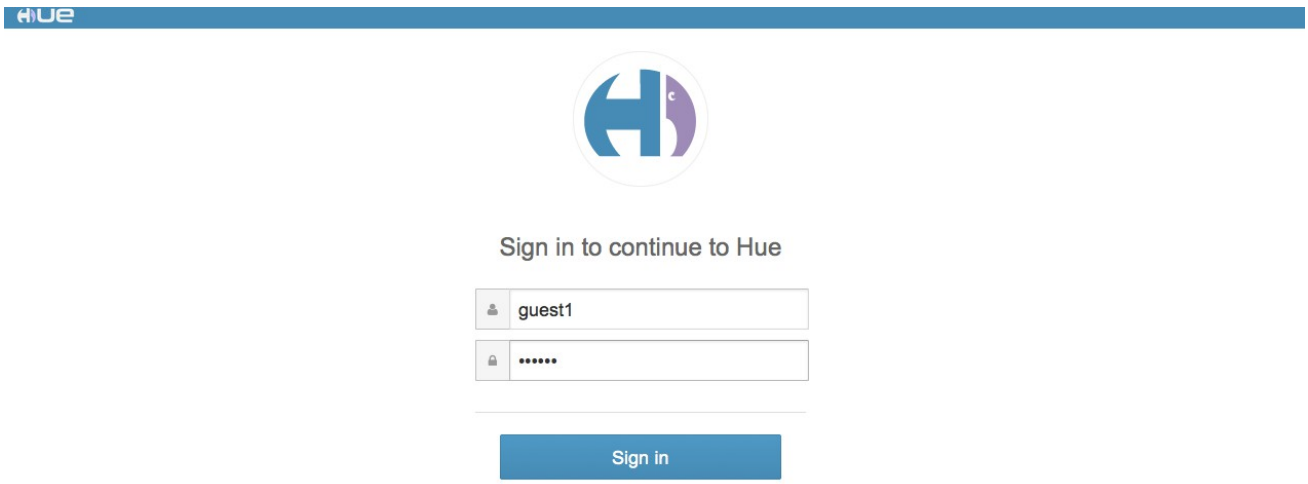


Service Management

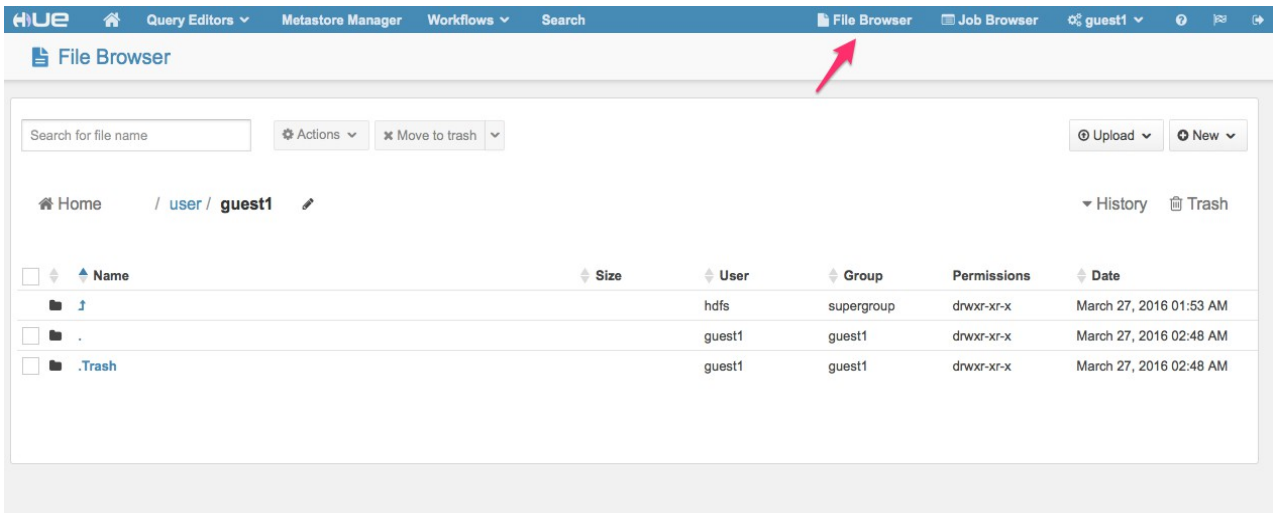


Hands-On: Importing/Exporting Data to HDFS

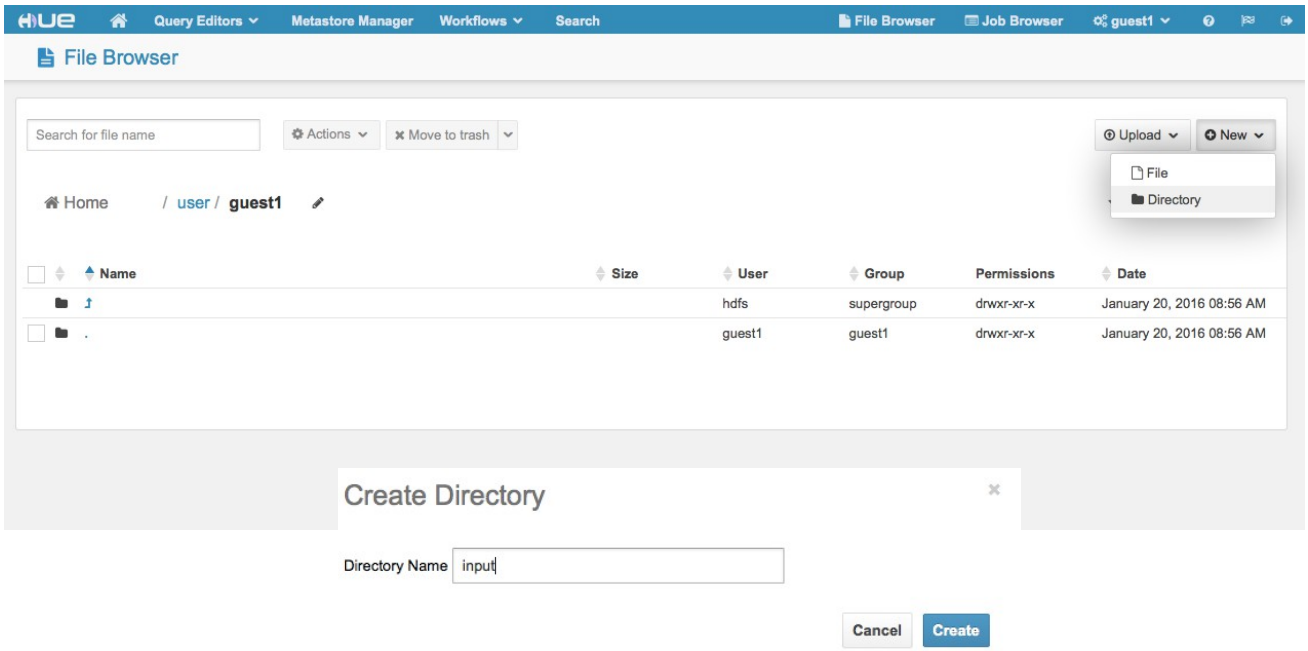
Login Hue as guestx

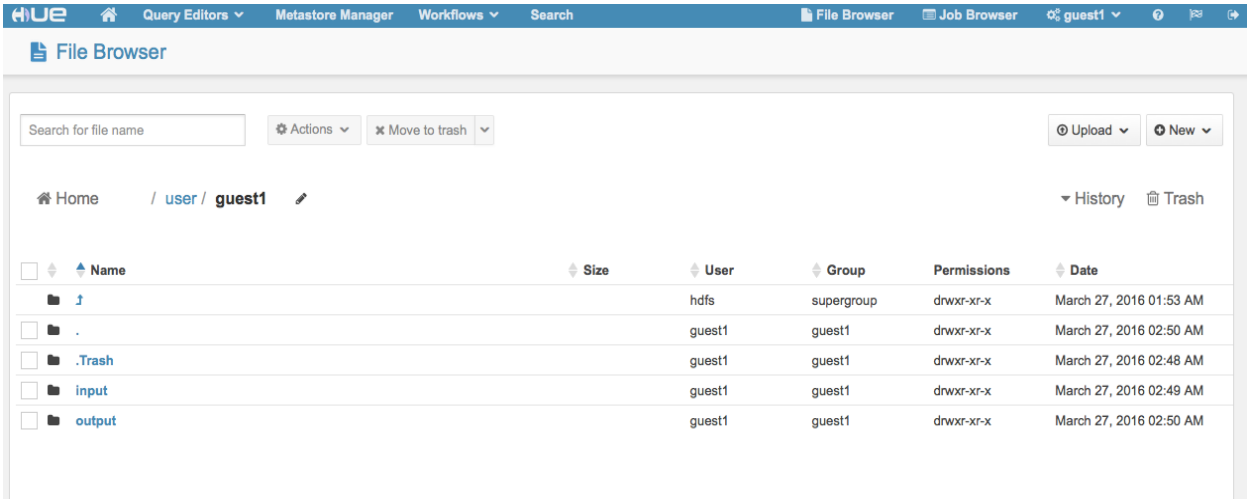


Review file in Hadoop HDFS using File Browse

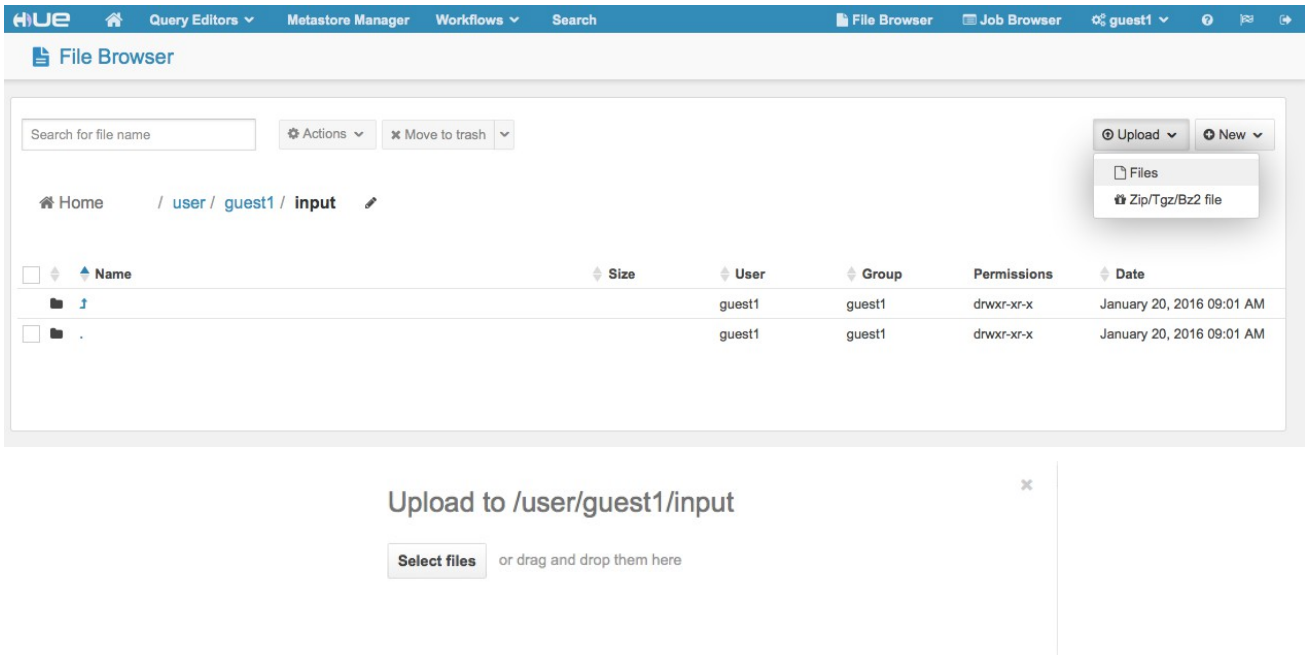


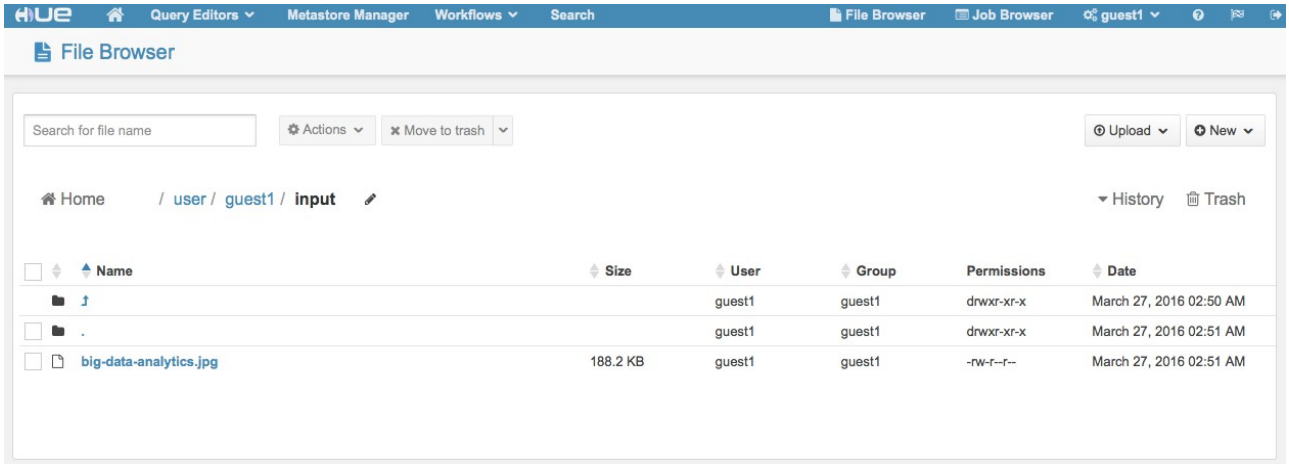
Create a new directory name as: input & output





Upload a local file to HDFS





Hands-On: Connect to a master node via SSH

SSH Login to a master node

```
THANACHARTs-MacBook-Air:elastic-mapreduce-cli THANACHART$ ssh -i "imchadoop.pem" ub
untu@ec2-54-201-147-59.us-west-2.compute.amazonaws.com
Welcome to Ubuntu 14.04.3 LTS (GNU/Linux 3.13.0-74-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

  System information as of Sun Mar 27 09:08:01 UTC 2016

  System load:  0.0                Processes:           135
  Usage of /:   27.6% of 29.40GB   Users logged in:     0
  Memory usage: 24%                IP address for eth0: 172.31.10.53
  Swap usage:   0%

  Graph this data and manage this system at:
    https://landscape.canonical.com/

  Get cloud support with Ubuntu Advantage Cloud Guest:
    http://www.ubuntu.com/business/services/cloud


*** System restart required ***
Last login: Sun Mar 27 09:08:01 2016 from node-io5.pool-125-24.dynamic.totbb.net
ubuntu@ip-172-31-10-53:~$
```


Hadoop syntax for HDFS

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 /`

Download an example text file

Make your own durectory at a master node to avoid mixing with others

```
$mkdir guest1
$cd guest1
$wget https://s3.amazonaws.com/imcbucket/input/pg2600.txt
```

```
--2016-03-27 09:58:48--  https://s3.amazonaws.com/imcbucket/input/pg2600.txt
Resolving s3.amazonaws.com (s3.amazonaws.com)... 54.231.19.187
Connecting to s3.amazonaws.com (s3.amazonaws.com)|54.231.19.187|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3291648 (3.1M) [text/plain]
Saving to: 'pg2600.txt'

100%[======================================>] 3,291,648   3.14MB/s   in 1.0s

2016-03-27 09:58:50 (3.14 MB/s) - 'pg2600.txt' saved [3291648/3291648]
```


Upload Data to Hadoop

Note: you login as ubuntu, so you need to a sudo command to Switch user to hdfs

```
$hadoop fs -ls /user/guest1/input
$sudo -u hdfs hadoop fs -rm /user/guest1/input/*
$sudo -u hdfs hadoop fs -put pg2600.txt /user/guest1/input/
$hadoop fs -ls /user/guest1/input
```

```
ubuntu@ip-172-31-10-53:~/guest1$ hadoop fs -ls /user/guest1/input
Found 1 items
-rw-r--r--   3 hdfs guest1    3291648 2016-03-27 10:04 /user/guest1/input/pg2600.txt
```

Hands-On: Writing you own Map Reduce Program

Example MapReduce: WordCount

```
package org.apache.hadoop.examples;

import java.io.IOException;
import java.util.StringTokenizer;

import org.apache.hadoop.conf.Configuration;
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.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.util.GenericOptionsParser;

public class WordCount {

  public static class TokenizerMapper
       extends Mapper<Object, Text, Text, IntWritable>{

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

    public void map(Object key, Text value, Context context
                    ) throws IOException, InterruptedException {
      StringTokenizer itr = new StringTokenizer(value.toString());
      while (itr.hasMoreTokens()) {
        word.set(itr.nextToken());
        context.write(word, one);
      }
    }
  }
```

```
  public static class IntSumReducer
       extends Reducer<Text,IntWritable,Text,IntWritable> {
    private IntWritable result = new IntWritable();

    public void reduce(Text key, Iterable<IntWritable> values,
                       Context context
                       ) throws IOException, InterruptedException {
      int sum = 0;
      for (IntWritable val : values) {
        sum += val.get();
      }
      result.set(sum);
      context.write(key, result);
    }
  }
```

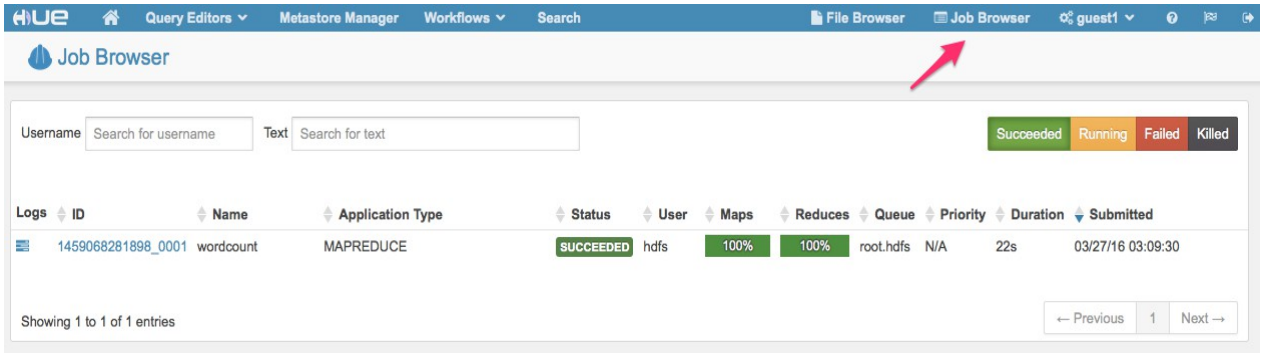
```
  public static void main(String[] args) throws Exception {
    Configuration conf = new Configuration();
    String[] otherArgs = new GenericOptionsParser(conf, args).getRemainingArgs();
    if (otherArgs.length != 2) {
      System.err.println("Usage: wordcount <in> <out>");
      System.exit(2);
    }
    Job job = new Job(conf, "word count");
    job.setJarByClass(WordCount.class);
    job.setMapperClass(TokenizerMapper.class);
    job.setCombinerClass(IntSumReducer.class);
    job.setReducerClass(IntSumReducer.class);
    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(IntWritable.class);
    FileInputFormat.addInputPath(job, new Path(otherArgs[0]));
    FileOutputFormat.setOutputPath(job, new Path(otherArgs[1]));
    System.exit(job.waitForCompletion(true) ? 0 : 1);
  }
}
```


Running Map Reduce Program

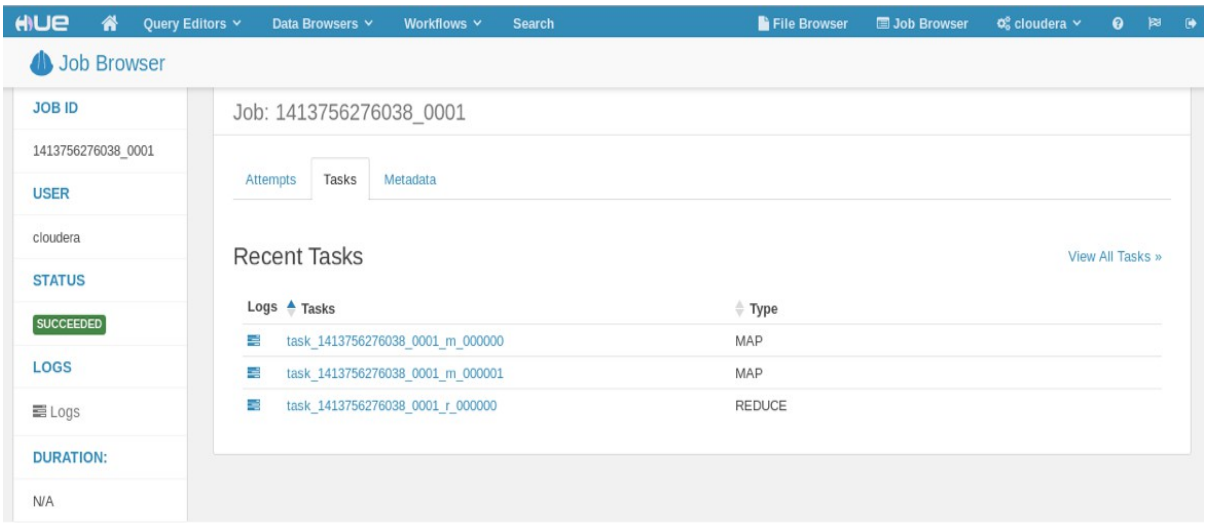
```
[cloudera@quickstart ~]$ cd workspace/
[cloudera@quickstart workspace]$ hadoop jar wordcount.jar org.myorg.WordCount input/*  output/wordcount_output
15/02/08 10:30:31 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
15/02/08 10:30:32 INFO client.RMProxy: Connecting to ResourceManager 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
```

```
$cd /home/ubuntu/guest1
$wget https://dl.dropboxusercontent.com/u/12655380/wordcount.jar
$sudo -u hdfs hadoop jar wordcount.jar org.myorg.WordCount
/user/guest1/input/* /user/guest1/output/wordcount
```

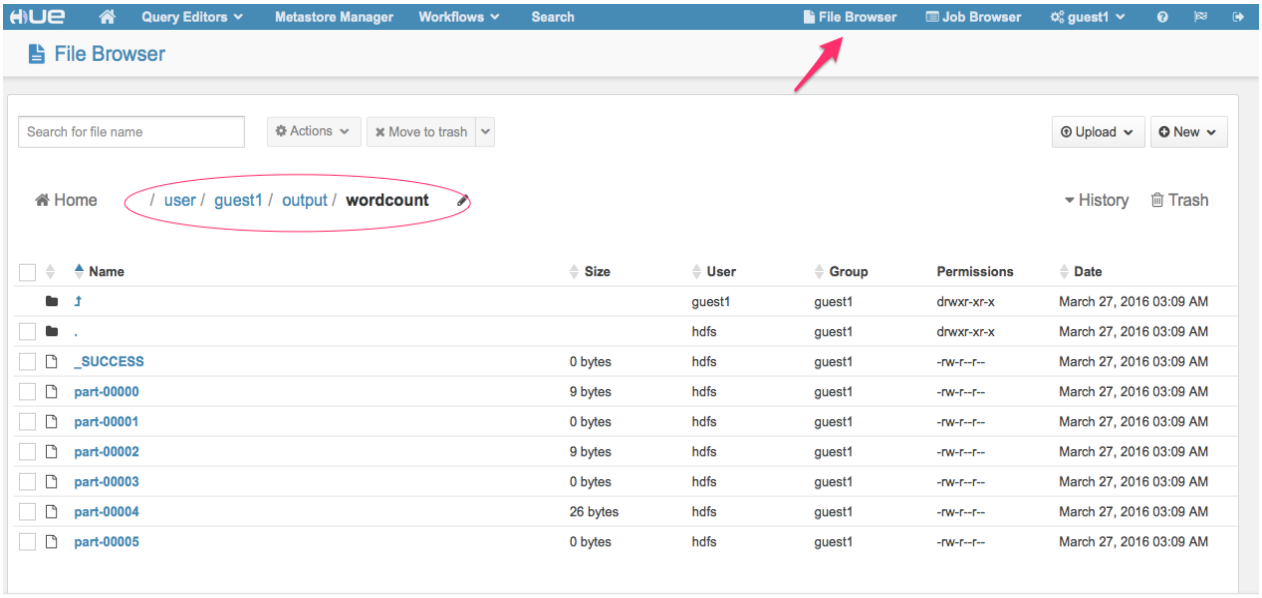
Reviewing MapReduce Job in Hue



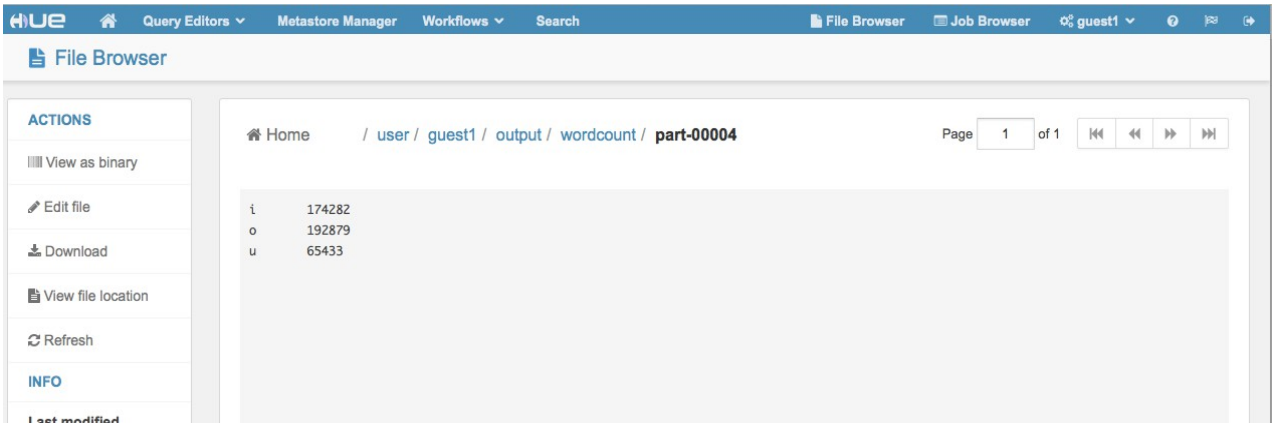
Reviewing MapReduce Job in Hue



Reviewing MapReduce Output Result



Reviewing MapReduce Output Result



Lecture
Understanding Oozie

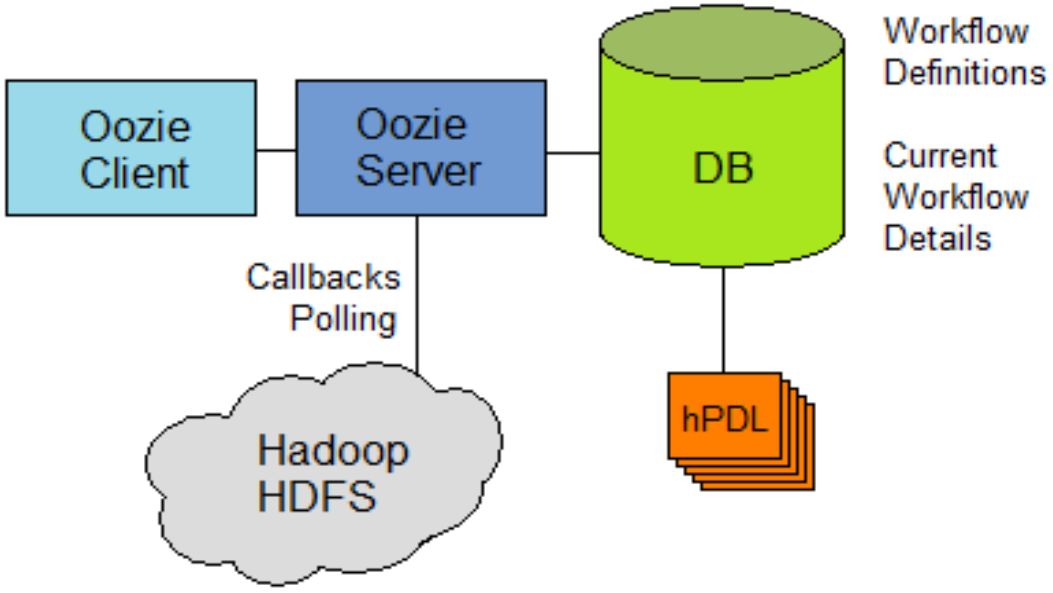
Introduction
Workflow scheduler for Hadoop

Oozie is a workflow scheduler system to manage Apache Hadoop jobs. Oozie is integrated with the rest of the Hadoop stack supporting several types of Hadoop jobs out of the box (such as Java map-reduce, Streaming map-reduce, Pig, Hive, Sqoop and Distcp) as well as system specific jobs (such as Java programs and shell scripts).

What is Oozie?

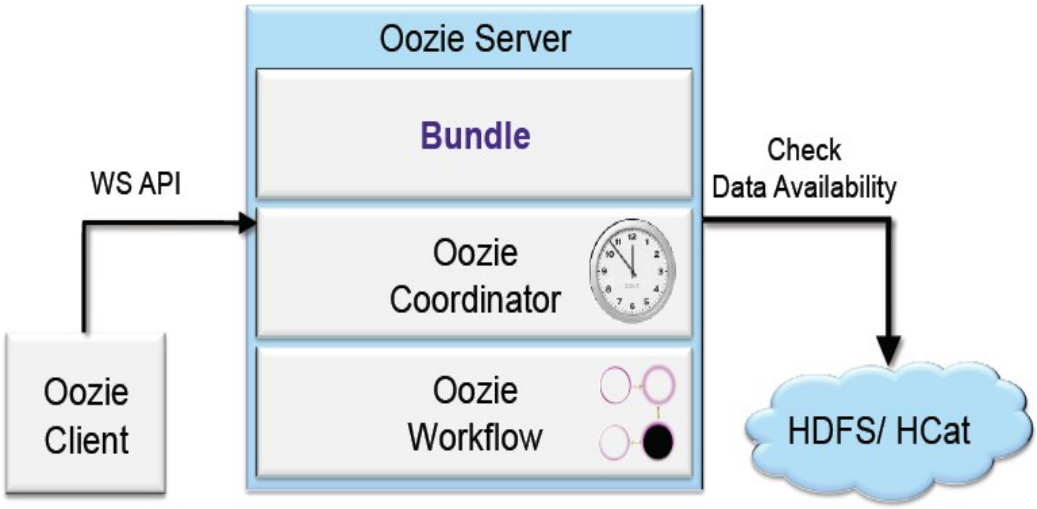
- Work flow scheduler for Hadoop
- Manages Hadoop Jobs
- Integrated with many Hadoop apps i.e. Pig, Hive
- Scaleable
- Schedule jobs
- A work flow is a collection of actions.
- A work flow is
  - Arranged as a DAG (direct acyclic graph)
  - Graph stored as hPDL (XML process definition)

Oozie Architecture



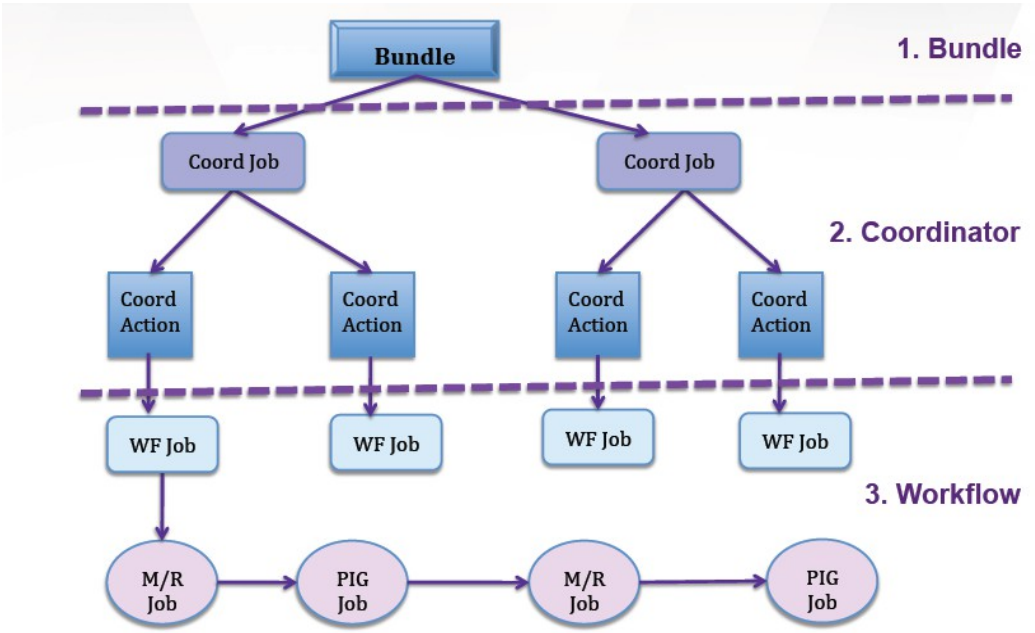
Source: info@semtech-solutions.co.nz

Oozie Server



Source: Oozie – Now and Beyond, Yahoo, 2013

Layer of Abstraction in Oozie



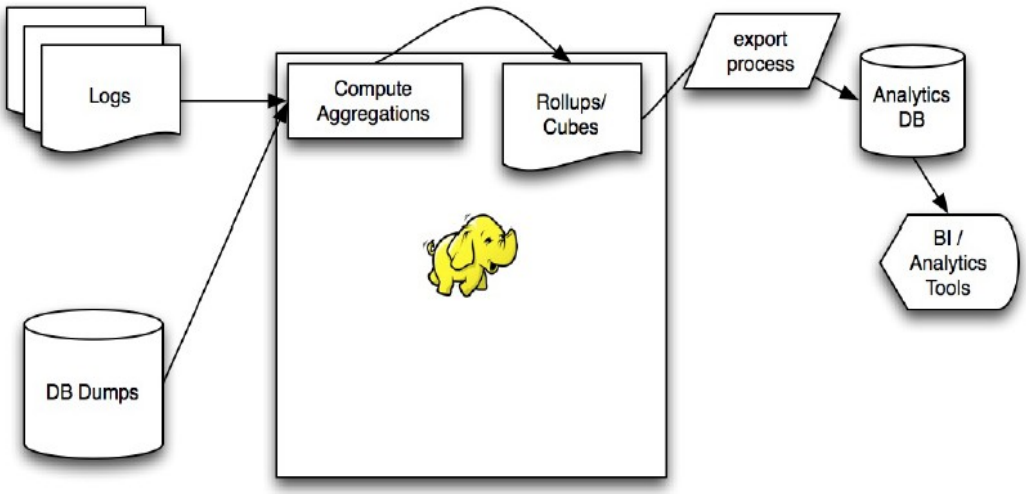
Source: Oozie – Now and Beyond, Yahoo, 2013

Workflow Example: Data Analytics

- Logs => fact table(s)
- Database backup => Dimension tables
- Complete rollups/cubes
- Load data into a low-latency storage (e.g. Hbae, HDFS)
- Dashboard & BI tools

Source: Workflow Engines for Hadoop, Joe Crobak, 2013

Workflow Example: Data Analytics



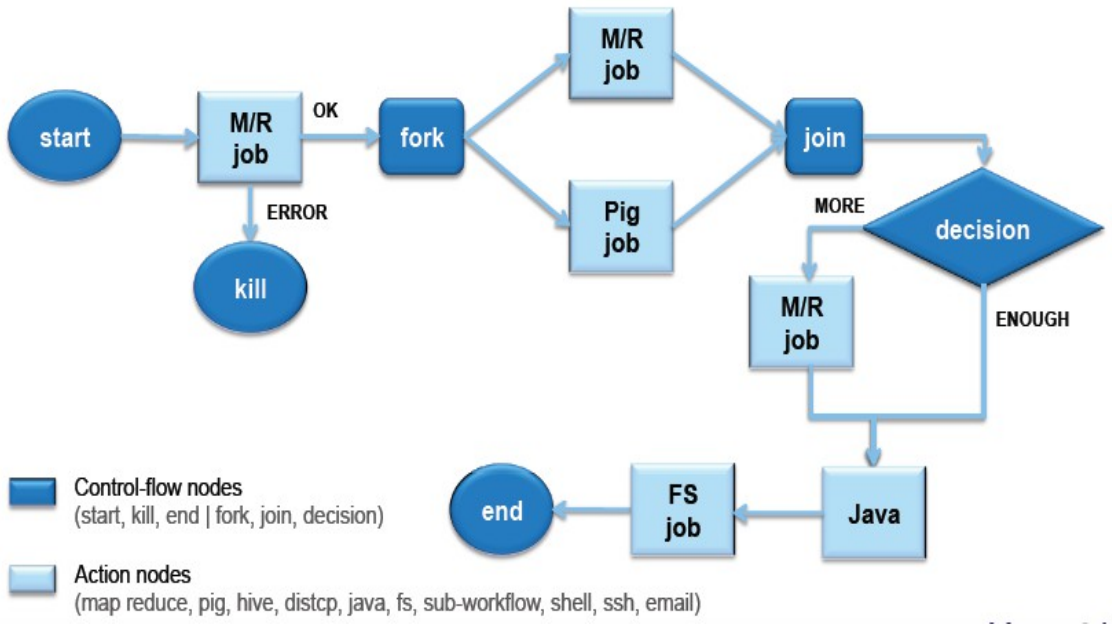
Source: Workflow Engines for Hadoop, Joe Crobak, 2013

Workflow Example: Data Analytics

- What happens if there is a failure?
  - Rebuild the failed day
  - .. and any downstream datasets
- With Hadoop Workflow
  - Possible OK to skip a day
  - Workflow tends to be self-contained, so you do not need to run downstream.
  - Sanity check your data before pushing to production.

Source: Workflow Engines for Hadoop, Joe Crobak, 2013

Oozie Workflow



Source: Oozie – Now and Beyond, Yahoo, 2013

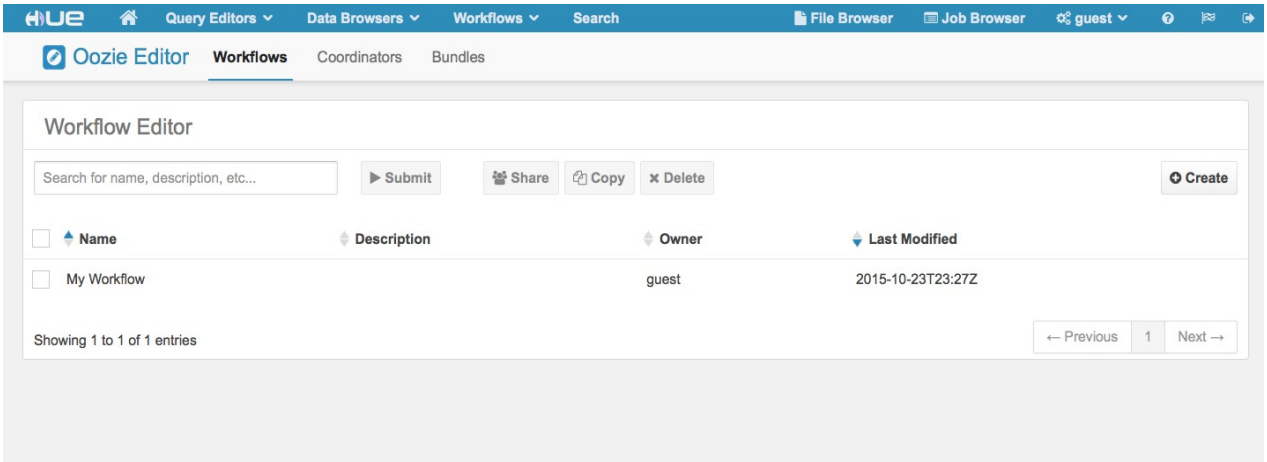
Oozie Use Cases

- Time Triggers
  - Execute your workflow every 15 minutes
- Time and Data Triggers
  - Materialize your workflow every hour, but only run them when the input data is ready (that is loaded to the grid every hour)
- Rolling Window
  - Access 15 minute datasets and roll them up into hourly datasets

Source: Oozie – Now and Beyond, Yahoo, 2013

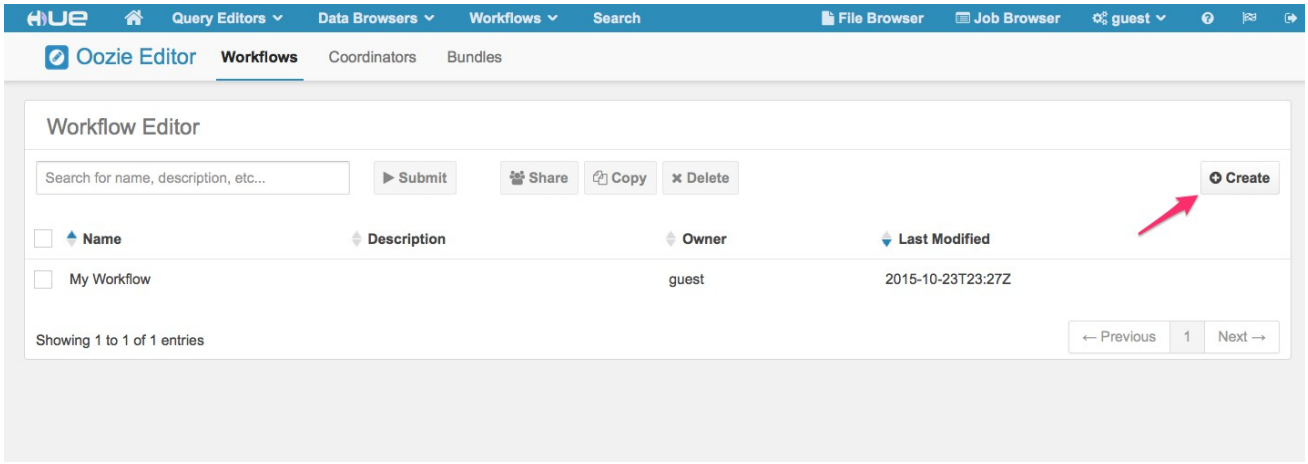
Hands-On: Running Map Reduce using Oozie workflow

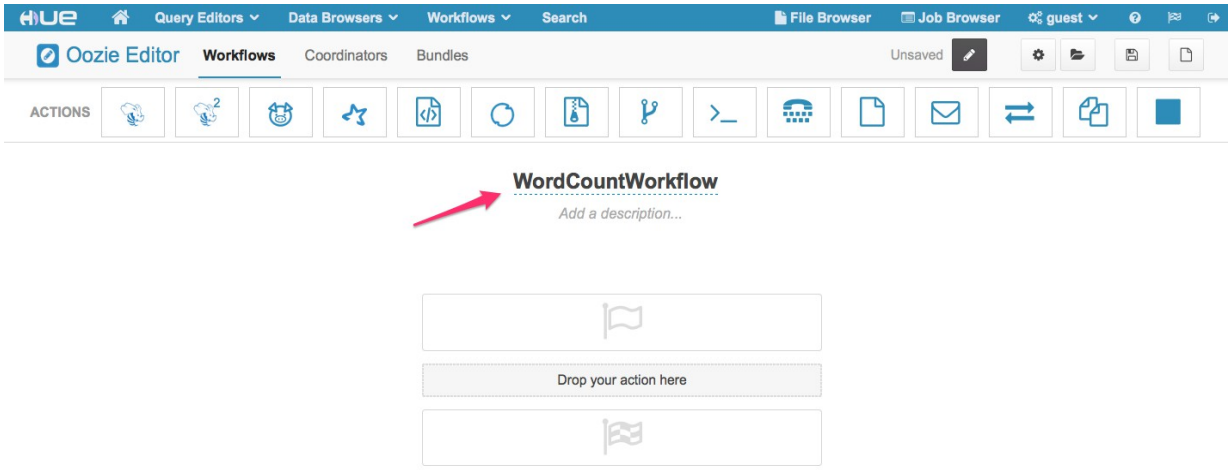
Using Hue: select WorkFlows >> Editors >> Workflows



Create a new workflow

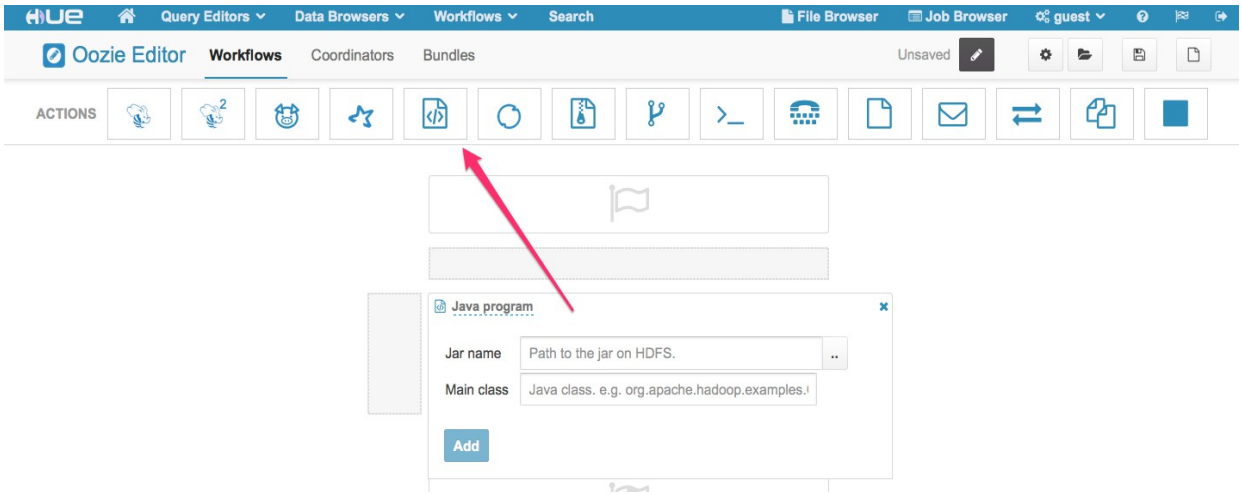
- Click Create button; the following screen will be displayed
- Name the workflow as WordCountWorkflow





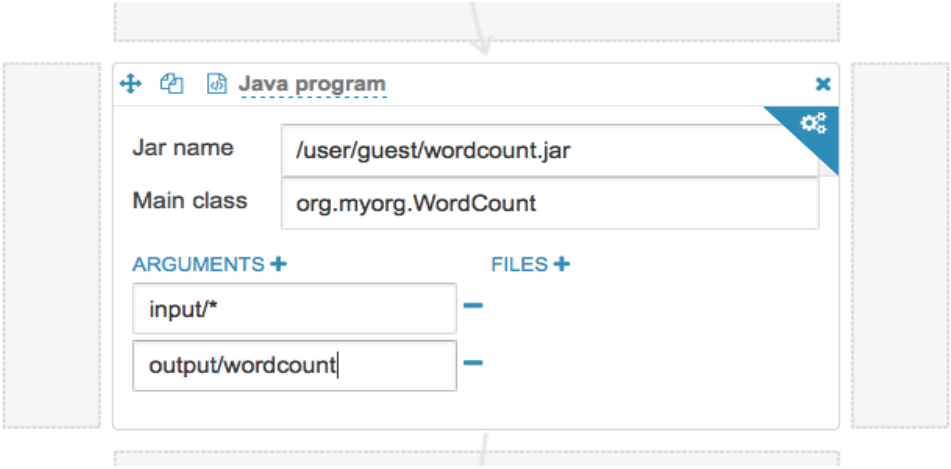
Select a Java job for the workflow

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



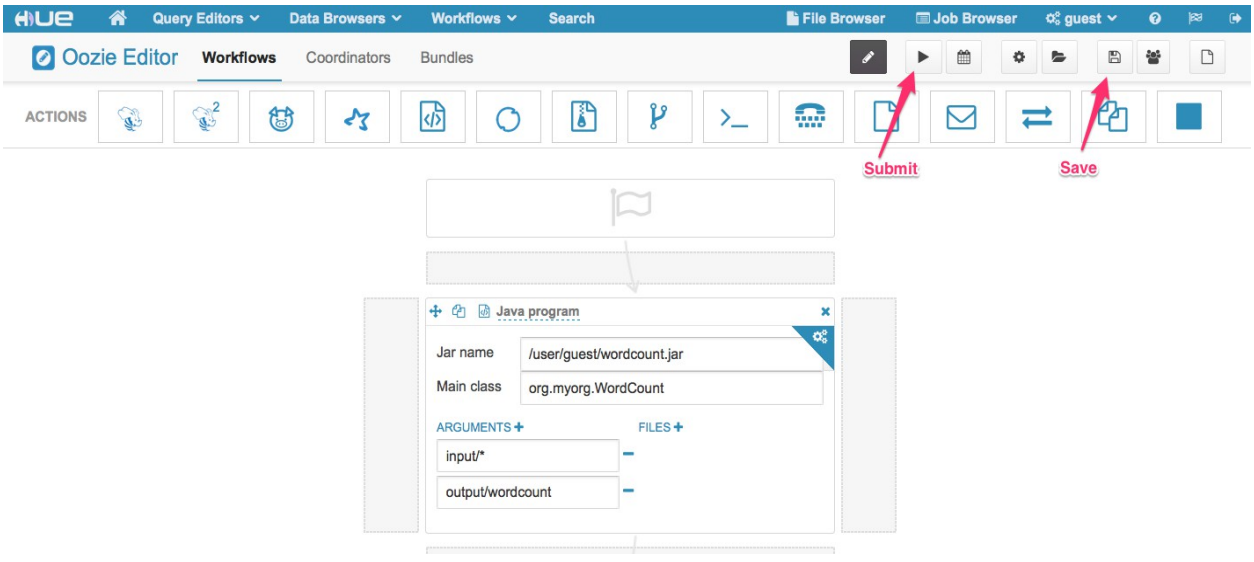
Edit the Java Job

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



Submit the workflow

- Click Done, follow by Save
- Then click submit



Lecture
Understanding Hive

Introduction
A Petabyte Scale Data Warehouse Using Hadoop

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

What Hive is NOT

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

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.apache.org

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.apache.org

System Architecture and Components

Metastore: To store the meta data.

Query compiler and execution engine: To convert SQL queries to a sequence of map/reduce jobs that are then executed on Hadoop.

SerDe and ObjectInspectors: Programmable interfaces and implementations of common data formats and types.

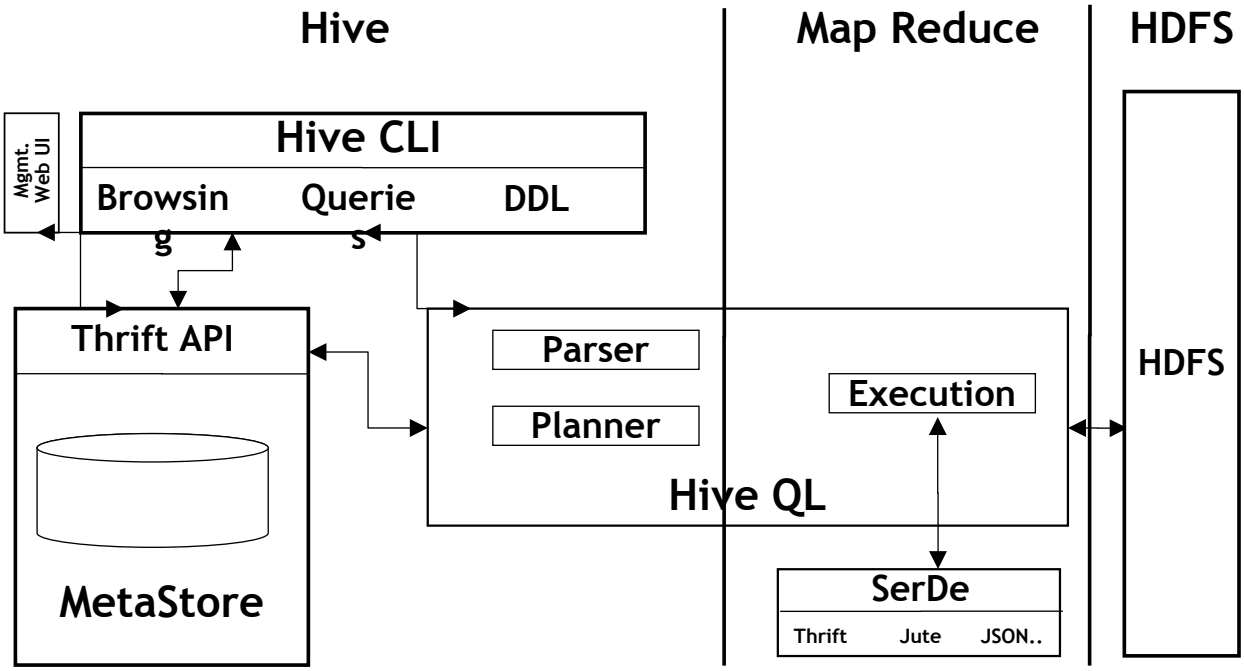
A SerDe is a combination of a Serializer and a Deserializer (hence, Ser-De). The Deserializer interface takes a string or binary representation of a record, and translates it into a Java object that Hive can manipulate. The Serializer, however, will take a Java object that Hive has been working with, and turn it into something that Hive can write to HDFS or another supported system.

UDF and UDAF: Programmable interfaces and implementations for user defined functions (scalar and aggregate functions).

Clients: Command line client similar to Mysql command line.

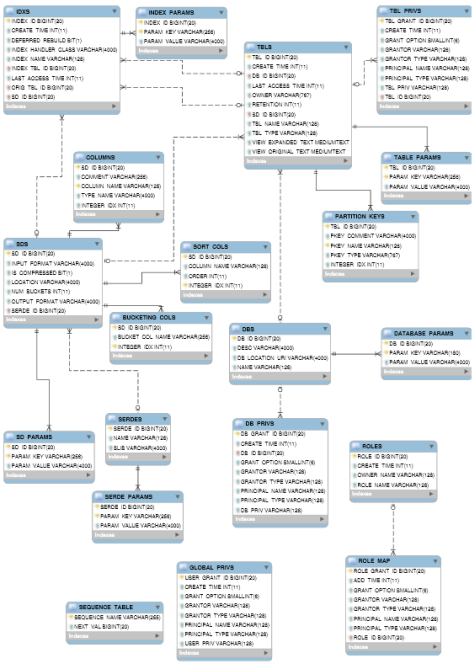
hive.apache.org

Architecture Overview



Hive.apache.org

Hive Metastore



Hive Metastore is a repository to keep all Hive metadata; Tables and Partitions definition.

By default, Hive will store its metadata in Derby DB

Hive Built in Functions

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.apache.org

Hive Aggregate Functions

Return Type	Aggregation Function Name (Signature)	Description
BIGINT	count(*), count(expr), count(DISTINCT expr[, expr_.])	count(*) - Returns the total number of retrieved rows, including rows containing NULL values; count(expr) - Returns the number of rows for which the supplied expression is non-NULL; count(DISTINCT expr[, expr]) - Returns the number of rows for which the supplied expression(s) are unique and non-NULL.
DOUBLE	sum(col), sum(DISTINCT col)	returns the sum of the elements in the group or the sum of the distinct values of the column in the group
DOUBLE	avg(col), avg(DISTINCT col)	returns the average of the elements in the group or the average of the distinct values of the column in the group
DOUBLE	min(col)	returns the minimum value of the column in the group
DOUBLE	max(col)	returns the maximum value of the column in the group

hive.apache.org

Running Hive

Hive Shell

Interactive

hive

Script

hive -f myscript

Inline

*hive -e 'SELECT * FROM mytable'*

Hive.apache.org

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`

ortonworks.com

Hive Tables

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

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

Java JDBC for Hive

```
import java.sql.SQLException;
import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.Statement;
import java.sql.DriverManager;

public class HiveJdbcClient {
  private static String driverName = "org.apache.hadoop.hive.jdbc.HiveDriver";

  public static void main(String[] args) throws SQLException {
    try {
      Class.forName(driverName);
    } catch (ClassNotFoundException e) {
      // TODO Auto-generated catch block
      e.printStackTrace();
      System.exit(1);
    }
    Connection con = DriverManager.getConnection("jdbc:hive://localhost:10000/default", "", "");
    Statement stmt = con.createStatement();
    String tableName = "testHiveDriverTable";
    stmt.executeQuery("drop table " + tableName);
    ResultSet res = stmt.executeQuery("create table " + tableName + " (key int, value string)");
    // show tables
    String sql = "show tables '" + tableName + "'";
    System.out.println("Running: " + sql);
    res = stmt.executeQuery(sql);
    if (res.next()) {
      System.out.println(res.getString(1));
    }
    // describe table
    sql = "describe " + tableName;
    System.out.println("Running: " + sql);
    res = stmt.executeQuery(sql);
    while (res.next()) {
      System.out.println(res.getString(1) + "\t" + res.getString(2));
    }
```


Java JDBC for Hive

```
import java.sql.SQLException;
import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.Statement;
import java.sql.DriverManager;

public class HiveJdbcClient {
  private static String driverName = "org.apache.hadoop.hive.jdbc.HiveDriver";

  public static void main(String[] args) throws SQLException {
    try {
      Class.forName(driverName);
    } catch (ClassNotFoundException e) {
      // TODO Auto-generated catch block
      e.printStackTrace();
      System.exit(1);
    }
    Connection con = DriverManager.getConnection("jdbc:hive://localhost:10000/default", "", "");
    Statement stmt = con.createStatement();
    String tableName = "testHiveDriverTable";
    stmt.executeQuery("drop table " + tableName);
    ResultSet res = stmt.executeQuery("create table " + tableName + " (key int, value string)");
    // show tables
    String sql = "show tables '" + tableName + "'";
    System.out.println("Running: " + sql);
    res = stmt.executeQuery(sql);
    if (res.next()) {
      System.out.println(res.getString(1));
    }
    // describe table
    sql = "describe " + tableName;
    System.out.println("Running: " + sql);
    res = stmt.executeQuery(sql);
    while (res.next()) {
      System.out.println(res.getString(1) + "\t" + res.getString(2));
    }
```

Big Data Hadoop Workshop Danairat T., danairat@gmail.com: Thanachart N., thanachart@imcinstitute.com April 2015

HiveQL and MySQL Comparison

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

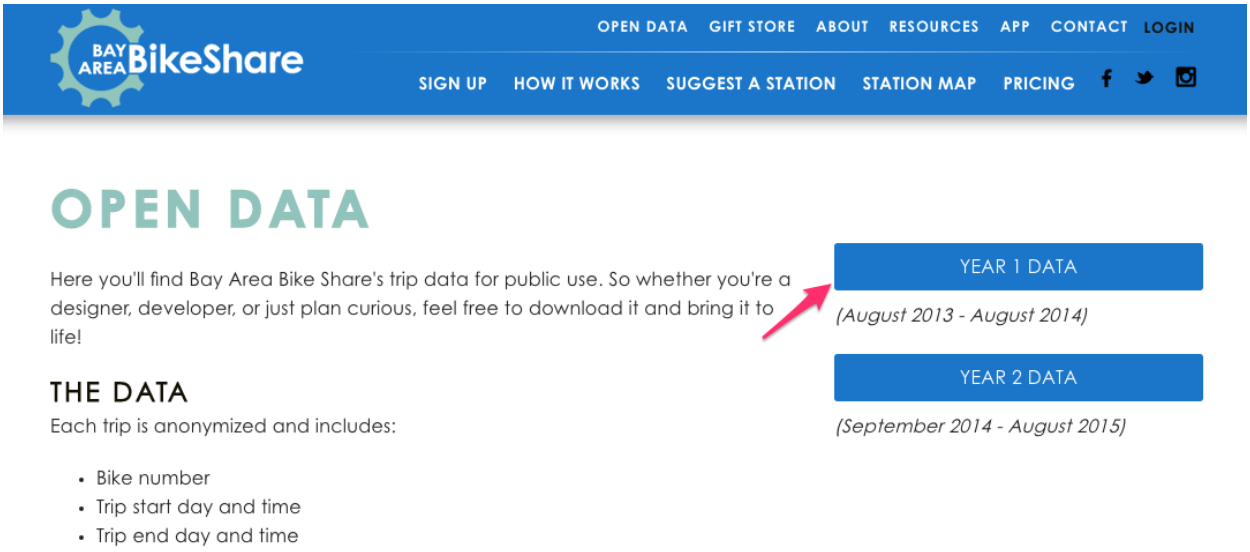
Query

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);`

Hands-On: Loading Data using Hive

Bay Area Bike Share (BABS)

http://www.bayareabikeshare.com/open-data



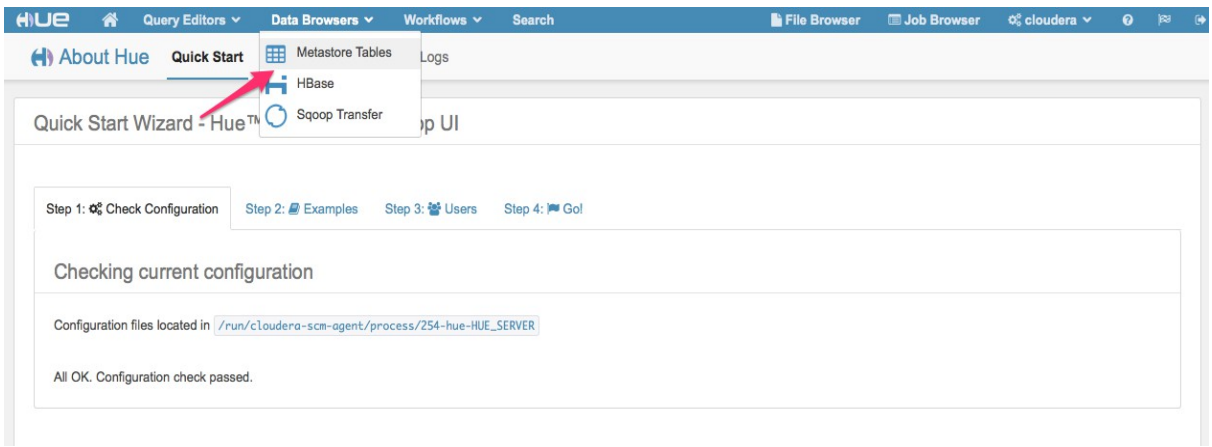
Preparing a bike data

```
$wget https://s3.amazonaws.com/babs-open-data/
babs_open_data_year_1.zip
$unzip babs_open_data_year_1.zip
$cd 201402_babs_open_data/
$sudo -u hdfs hadoop fs -put 201402_trip_data.csv
/user/guest1
$ hadoop fs -ls /user/guest1
```

```
Found 4 items
drwxr-xr-x   - guest1 guest1          0 2016-03-27 09:48 /user/guest1/.Trash
-rw-r--r--   3 hdfs   guest1   17219022 2016-03-27 10:19 /user/guest1/201402_trip_da
ta.csv
drwxr-xr-x   - guest1 guest1          0 2016-03-27 10:04 /user/guest1/input
drwxr-xr-x   - guest1 guest1          0 2016-03-27 10:09 /user/guest1/output
```

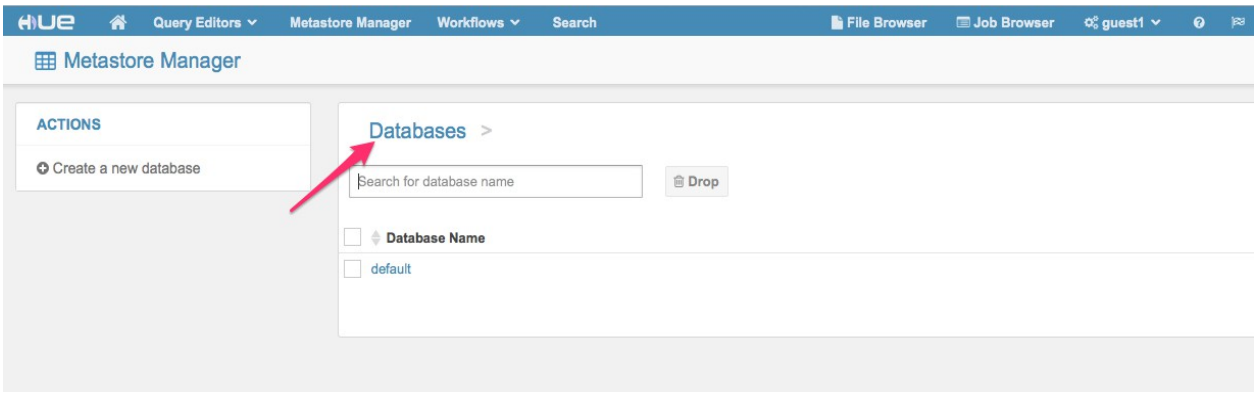

Importing CSV Data with the Metastore App

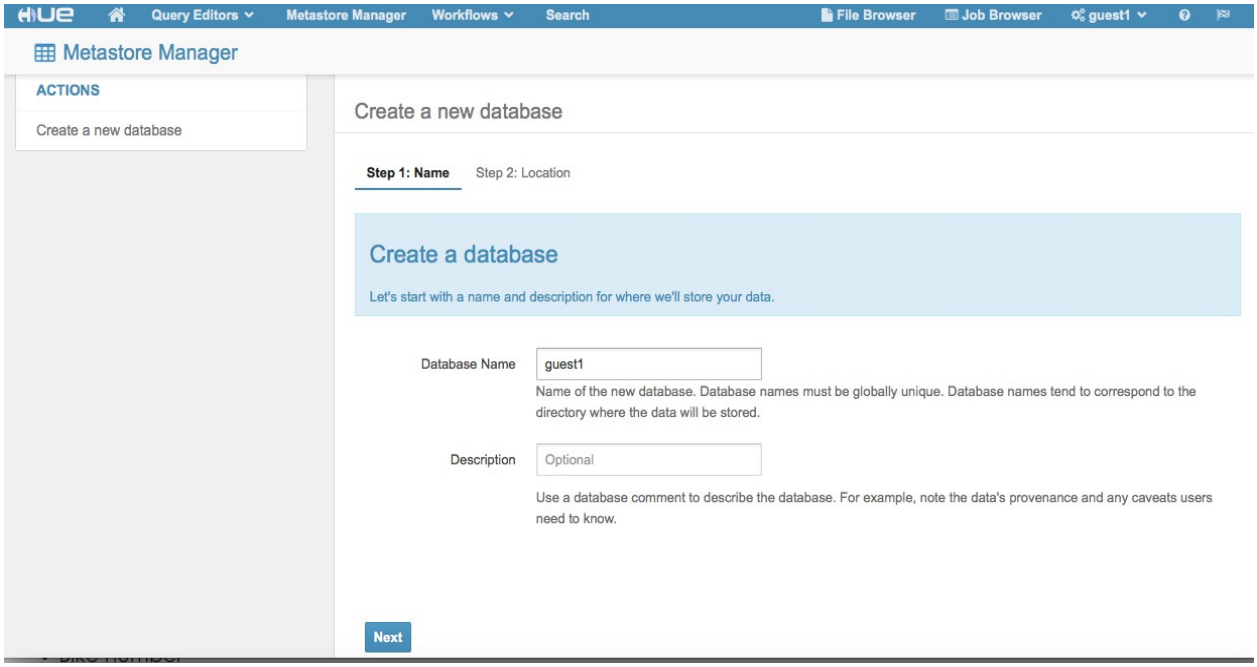
The BABS data set contains 4 CSVs that contain data for stations, trips, rebalancing (availability), and weather. We will import **trips** dataset using Metastore Tables



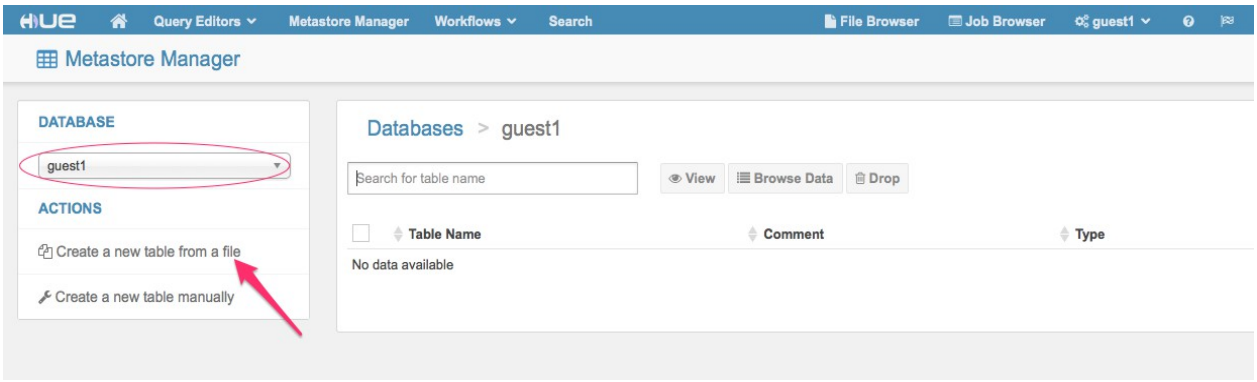
Select Database for create a new database

Name it as guest1

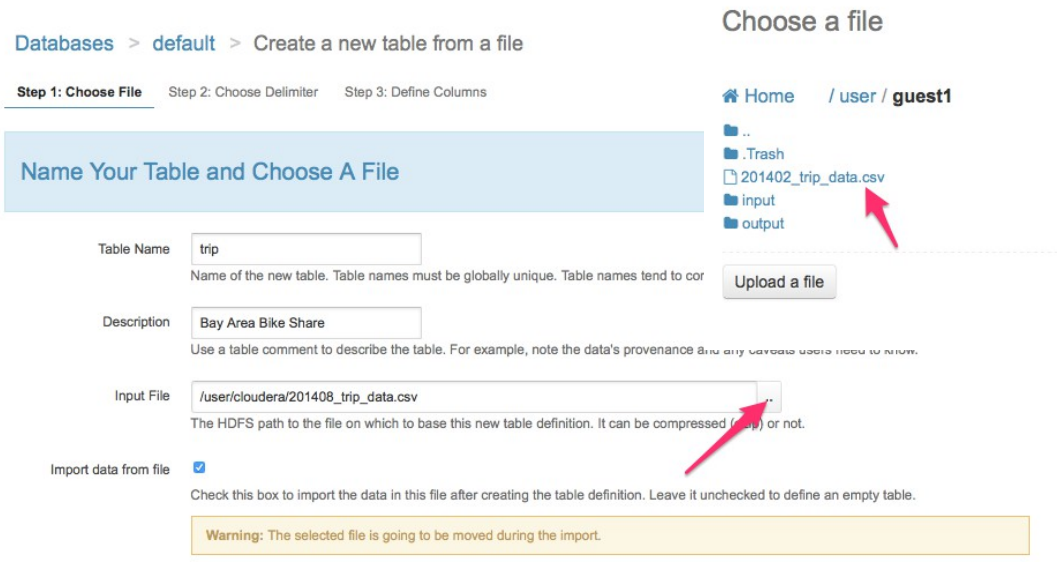




Select: Create a new table from a file



Name a table and select a file



Choose Delimiter

Databases > default > Create a new table from a file

Step 1: Choose File | **Step 2: Choose Delimiter** | Step 3: Define Columns

Choose a Delimiter

Beeswax has determined that this file is delimited by **commas**.

Delimiter: Comma (,) [Preview]

Enter the column delimiter which must be a single character. Use syntax like "\001" or "\t" for special characters.

Table preview

col_1	col_2	col_3	col_4	col_5	col_6	col_7	col_8	col_9	col_10	col_11
Trip ID	Duration	Start Date	Start Station	Start Terminal	End Date	End Station	End Terminal	Bike #	Subscriber Type	Zip Code
432946	406	8/31/2014 22:31	Mountain View Caltrain St…	28	8/31/2014 22:38	Castro Street and El Cami…	32	17	Subscriber	94040
432945	468	8/31/2014 22:07	Beale at Market	56	8/31/2014 22:15	Market at 4th	76	509	Customer	11231

Define Column Types

Databases > default > Create a new table from a file

Step 1: Choose File | Step 2: Choose Delimiter | **Step 3: Define Columns**

Define your columns

Use first row as column names | Bulk edit column names

Column name	Column Type	*Sample Row #1*	*Sample Row #2*
TripID	int	432946	432945
Duration	int	406	468
StartDate	string	8/31/2014 22:31	8/31/2014 22:07
StartStation	string	Mountain View Caltrain Station	Beale at Market
StartTerminal	tinyint	28	56
EndDate	string	8/31/2014 22:38	8/31/2014 22:15

Create Table : Done

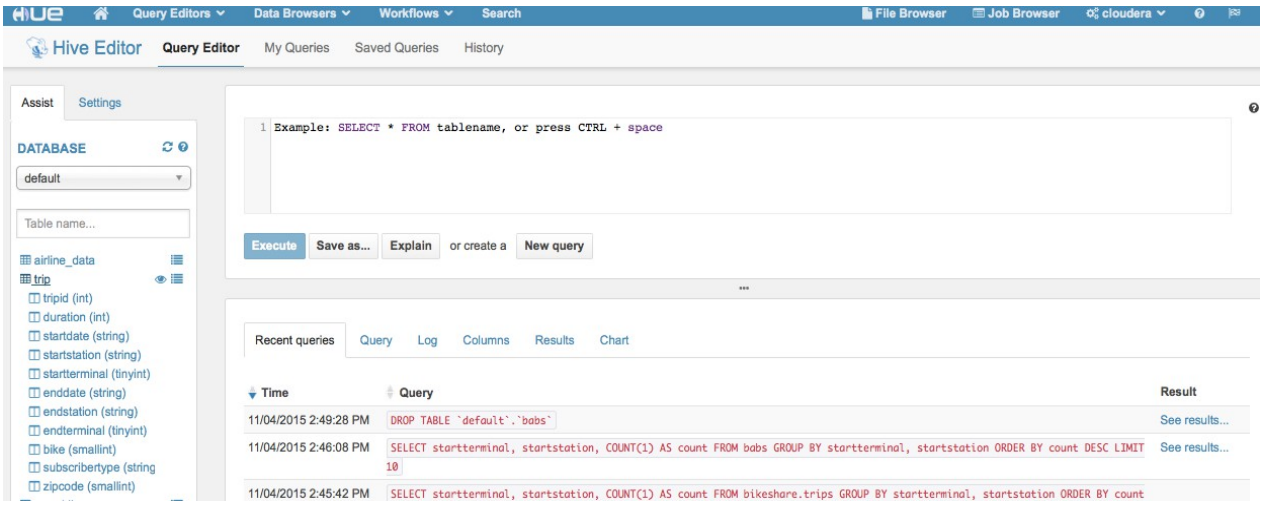
Databases > default > trip

Comment: Bay Area Bike Share

Columns | Sample | Properties

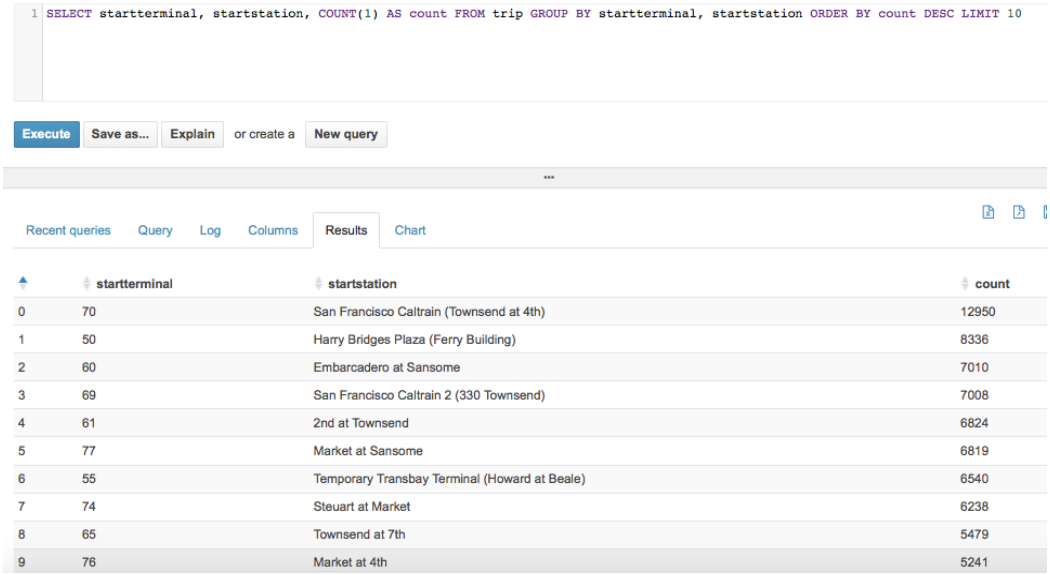
	Name	Type	Comment
0	tripid	int	
1	duration	int	
2	startdate	string	
3	startstation	string	
4	startterminal	tinyint	
5	enddate	string	
6	endstation	string	
7	endterminal	tinyint	
8	bike	smallint	
9	subscribertype	string	
10	zipcode	smallint	

Starting Hive Editor



Find the top 10 most popular start stations based on the trip data

```
SELECT startterminal, startstation, COUNT(1) AS count FROM trip
GROUP BY startterminal, startstation ORDER BY count DESC LIMIT 10
```



	startterminal	startstation	count
0	70	San Francisco Caltrain (Townsend at 4th)	12950
1	50	Harry Bridges Plaza (Ferry Building)	8336
2	60	Embarcadero at Sansome	7010
3	69	San Francisco Caltrain 2 (330 Townsend)	7008
4	61	2nd at Townsend	6824
5	77	Market at Sansome	6819
6	55	Temporary Transbay Terminal (Howard at Beale)	6540
7	74	Steuart at Market	6238
8	65	Townsend at 7th	5479
9	76	Market at 4th	5241

Running Hive from command line

```
$cd /home/ubuntu/guest1
$sudo -u hdfs hive
```

```
Logging initialized using configuration in jar:file:/opt/cloudera/parcels/CDH-5.6.0-
1.cdh5.6.0.p0.45/jars/hive-common-1.1.0-cdh5.6.0.jar!/hive-log4j.properties
WARNING: Hive CLI is deprecated and migration to Beeline is recommended.
hive>
```

Running Hive from command line

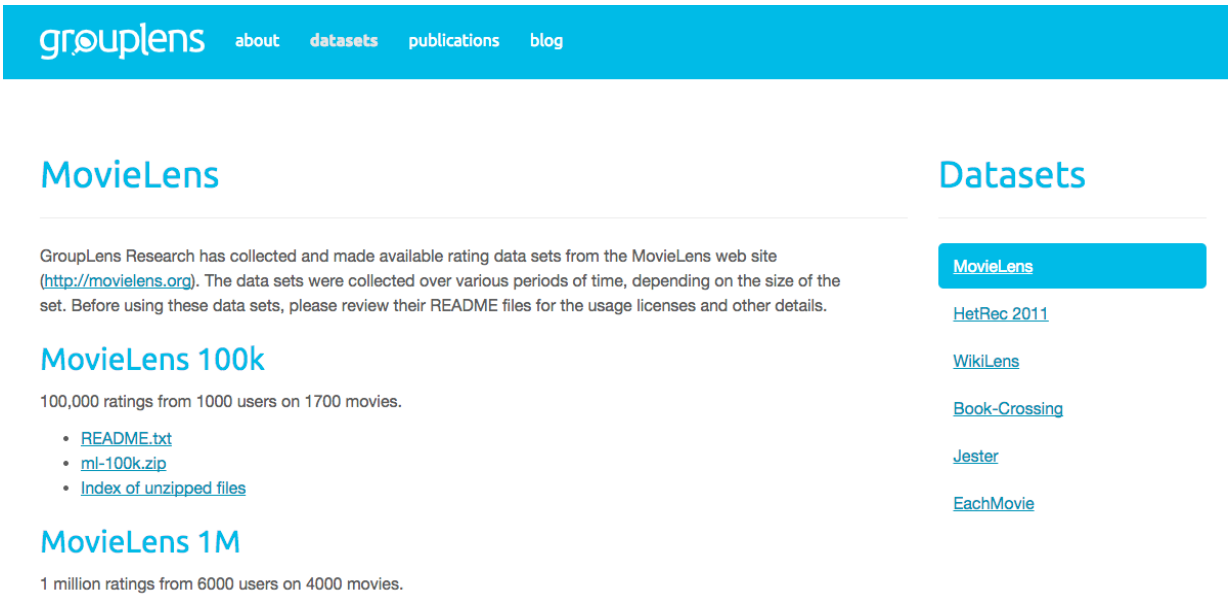
```
hive> use guest1;
OK
Time taken: 0.478 seconds
hive> SELECT startterminal, startstation, COUNT(1) AS count FROM trip GROUP BY start
terminal, startstation ORDER BY count DESC LIMIT 10;


Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 3.85 sec   HDFS Read: 17227138 HD
FS Write: 3162 SUCCESS
Stage-Stage-2: Map: 1  Reduce: 1   Cumulative CPU: 2.71 sec   HDFS Read: 8281 HDFS W
rite: 349 SUCCESS
Total MapReduce CPU Time Spent: 6 seconds 560 msec
OK
70      San Francisco Caltrain (Townsend at 4th)        9838
50      Harry Bridges Plaza (Ferry Building)    7343
60      Embarcadero at Sansome  6545
77      Market at Sansome       5922
55      Temporary Transbay Terminal (Howard at Beale)   5113
76      Market at 4th   5030
61      2nd at Townsend 4987
69      San Francisco Caltrain 2 (330 Townsend) 4976
74      Steuart at Market       4913
65      Townsend at 7th 4493
Time taken: 46.78 seconds, Fetched: 10 row(s)
hive>
```


Find the total number of trips and average duration (in minutes) of those trips, grouped by hour

```
SELECT
    hour,
    COUNT(1) AS trips,
    ROUND(AVG(duration) / 60) AS avg_duration
FROM (
    SELECT
        CAST(SPLIT(SPLIT(t.startdate, ' ')[1], ':')[0] AS INT) AS
hour,
        t.duration AS duration
    FROM `bikeshare`.`trips` t
    WHERE
        t.startterminal = 70
        AND
        t.duration IS NOT NULL
    ) r
GROUP BY hour
ORDER BY hour ASC;
```

Exercise: MovieLens

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



Lecture
Understanding Impala

Introduction

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.

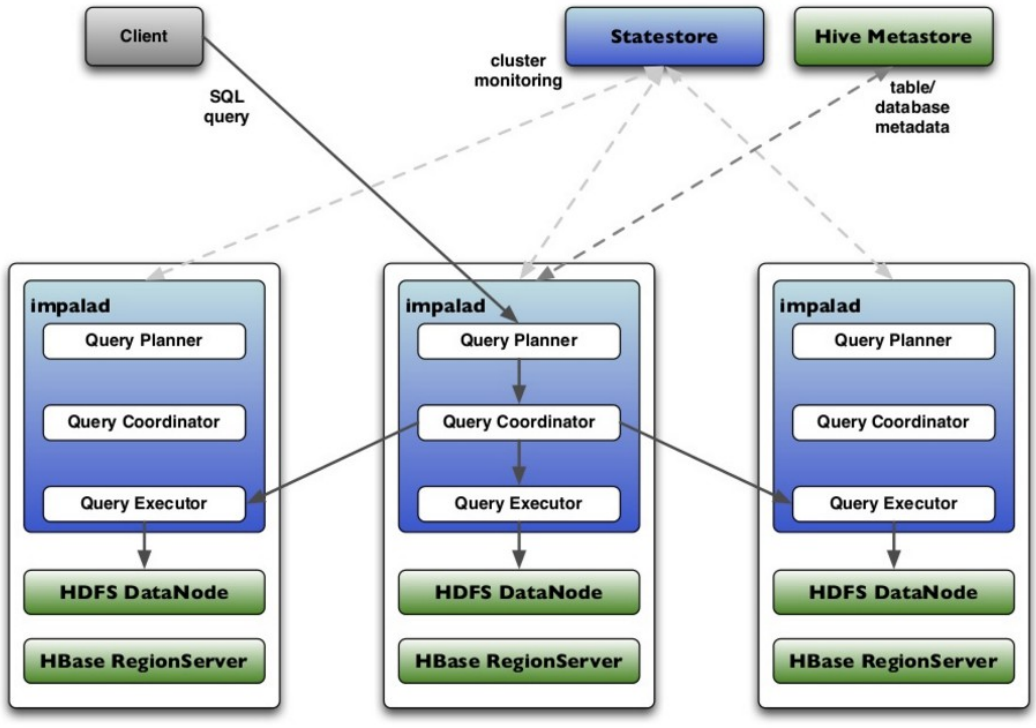
What is Impala?

- 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

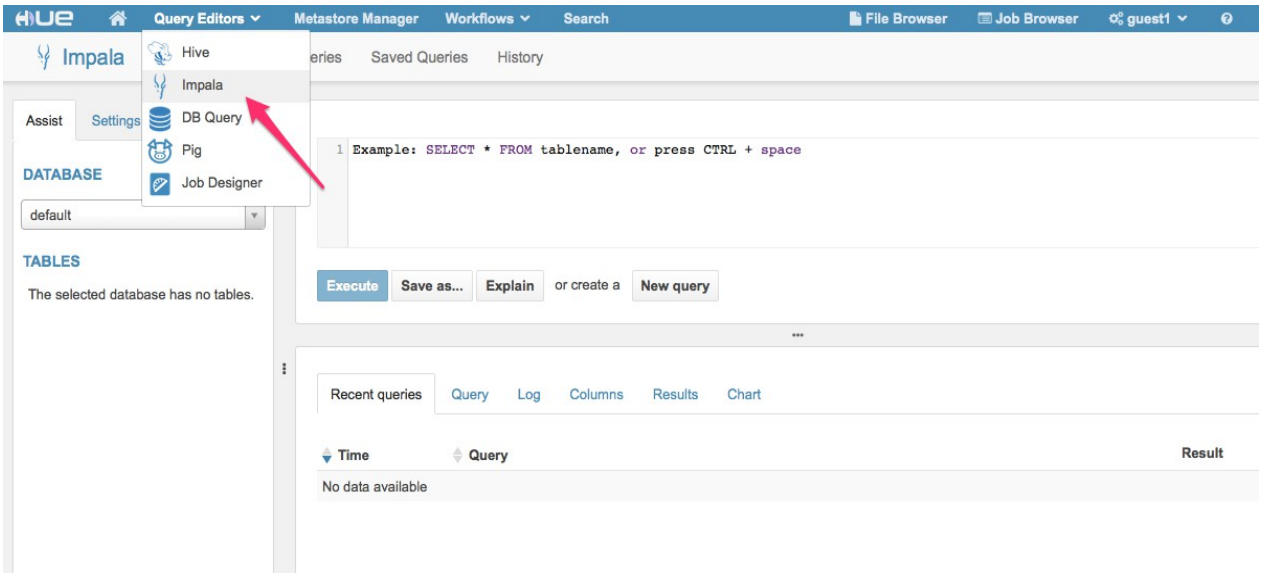
Impala Overview

- Impala daemon run on HDFS nodes
- Statestore (for cluster metadata) v.s. Metastore (for database metastore)
- Queries run on “revelants” nodes
- Support common HDFS file formats
- Submit quries via Hue/Beeswax
- No fault tolerant

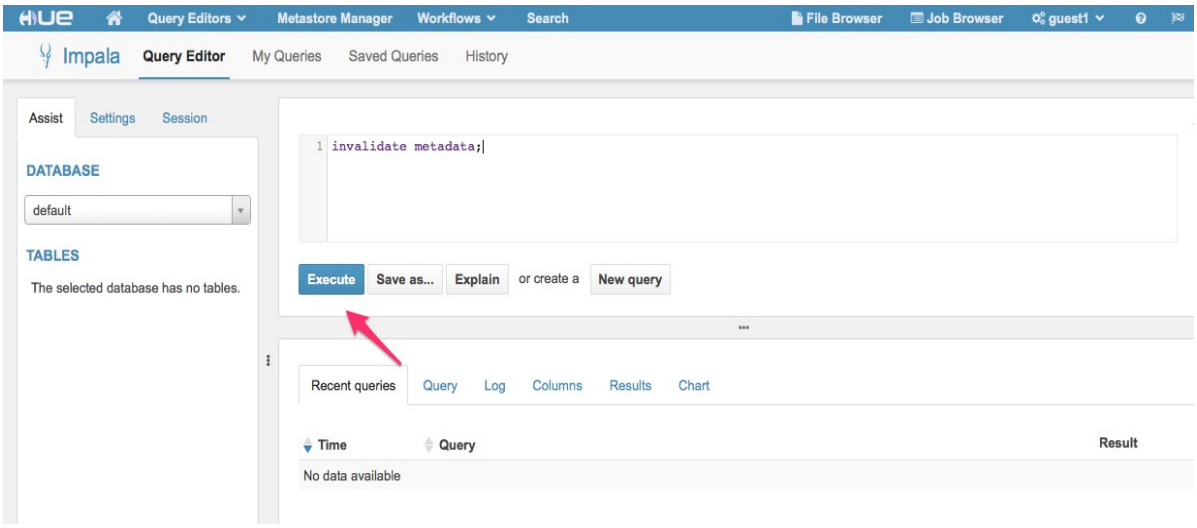
Impala Architecture



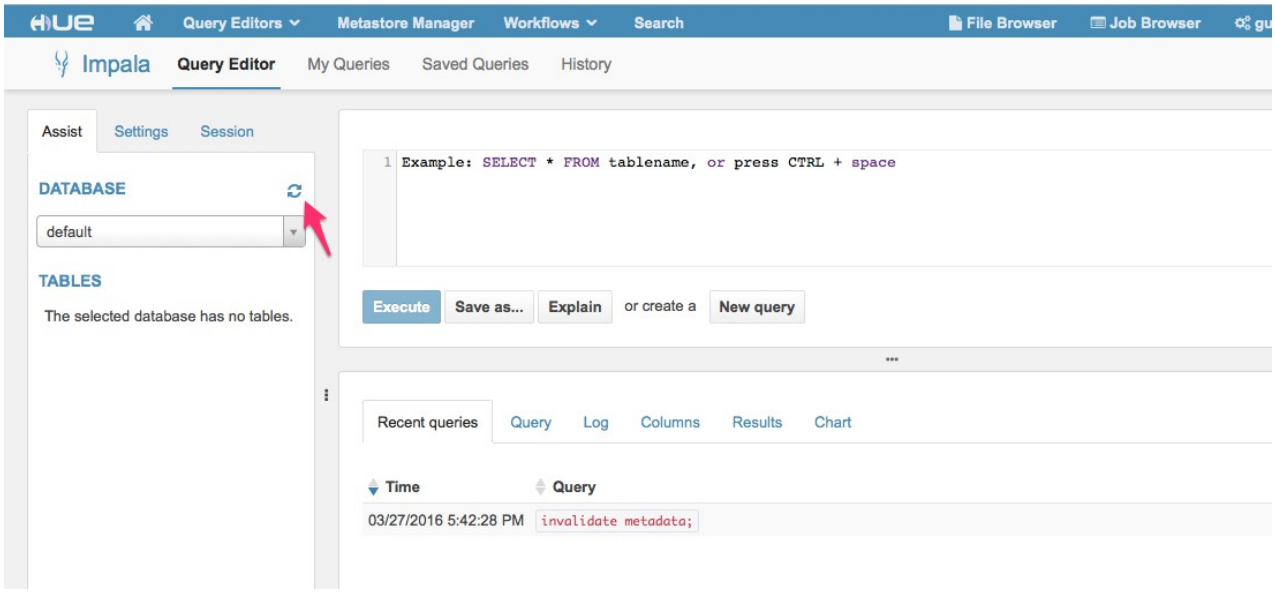
Start Impala Query Editor



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

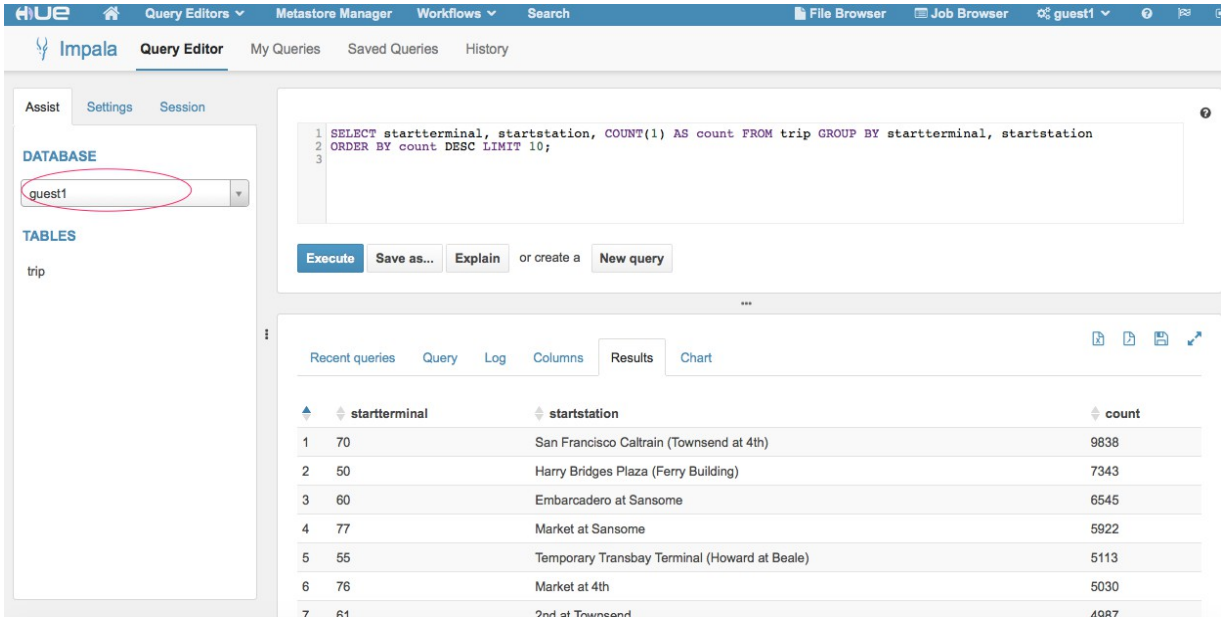


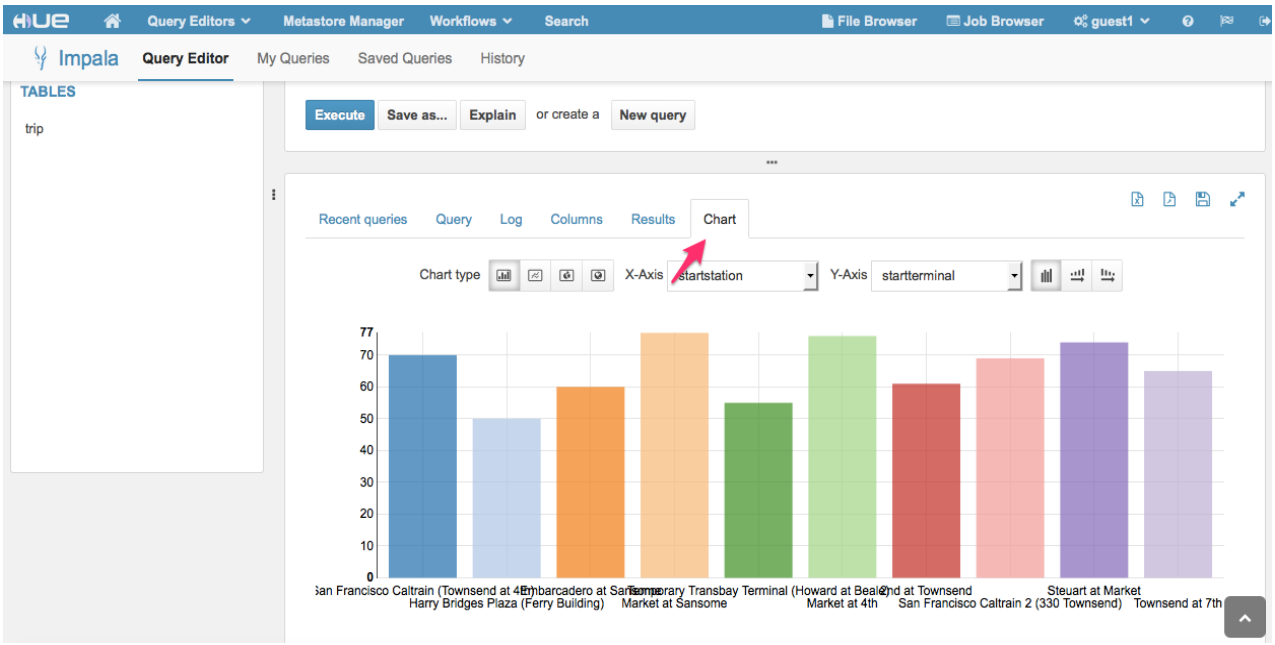
Restart Impala Query Editor and refresh the table list



Find the top 10 most popular start stations based on the trip data: Using Impala

```
SELECT startterminal, startstation, COUNT(1) AS count FROM trip
GROUP BY startterminal, startstation ORDER BY count DESC LIMIT 10
```





Find the total number of trips and average duration (in minutes) of those trips, grouped by hour

```
SELECT
    hour,
    COUNT(1) AS trips,
    ROUND(AVG(duration) / 60) AS avg_duration
FROM (
    SELECT
        CAST(SPLIT(SPLIT(t.startdate, ' ')[1], ':')[0] AS INT) AS
hour,
        t.duration AS duration
    FROM `bikeshare`.`trips` t
    WHERE
        t.startterminal = 70
        AND
        t.duration IS NOT NULL
    ) r
GROUP BY hour
ORDER BY hour ASC;
```

Lecture
Understanding Pig

Introduction

A high-level platform for creating MapReduce programs Using Hadoop

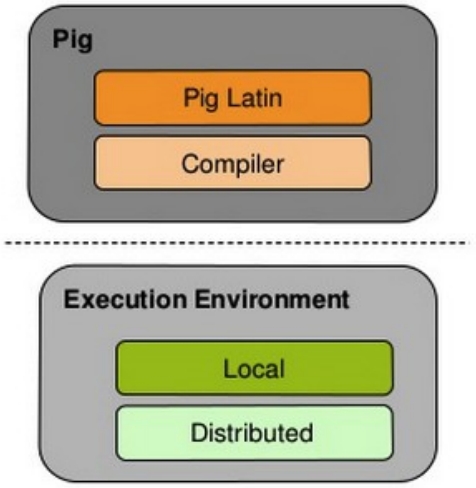
Pig is a platform for analyzing large data sets that consists of a high-level language for expressing data analysis programs, coupled with infrastructure for evaluating these programs. The salient property of Pig programs is that their structure is amenable to substantial parallelization, which in turns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

Pig Components

- **Two Compnents**
  - Language (Pig Latin)
  - Compiler
- **Two Execution Environments**
  - **Local**

    pig -x local
  - **Distributed**

    pig -x mapreduce



Hive.apache.org

Running Pig

- **Script**

  pig myscript
- **Command line (Grunt)**

  pig
- **Embedded**

  Writing a java program

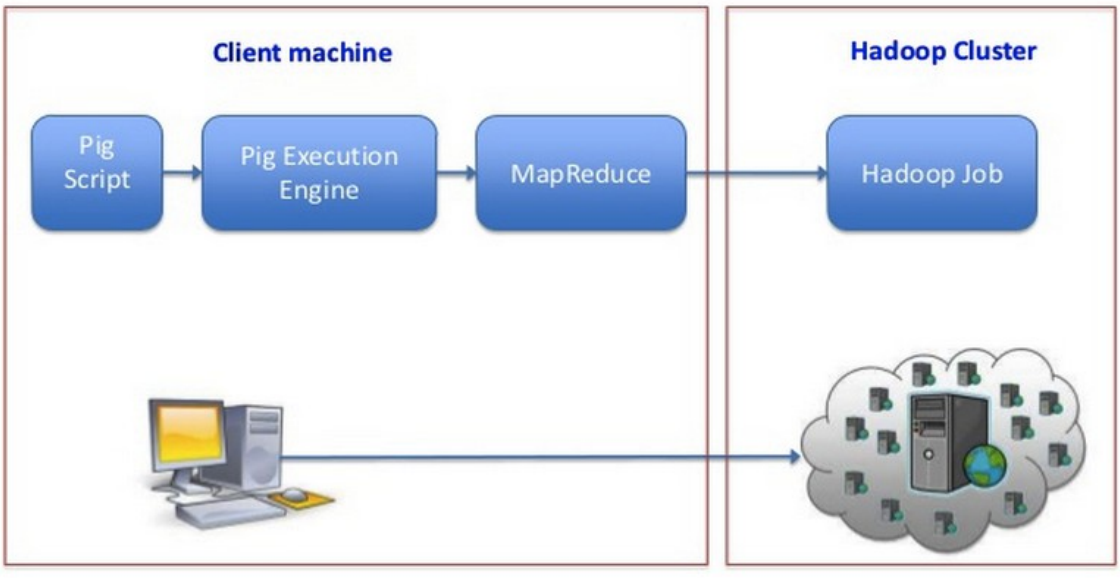
Hive.apache.org

Pig Latin

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

Hive.apache.org

Pig Execution Stages



Source Introduction to Apache Hadoop-Pig: PrashantKommireddi

Hive.apache.org

Why Pig?

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

Source Introduction to Apache Hadoop-Pig: PrashantKommireddi

Hive.apache.org

Pig v.s. Hive

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

Hive.apache.org

Hands-On: Running a Pig script

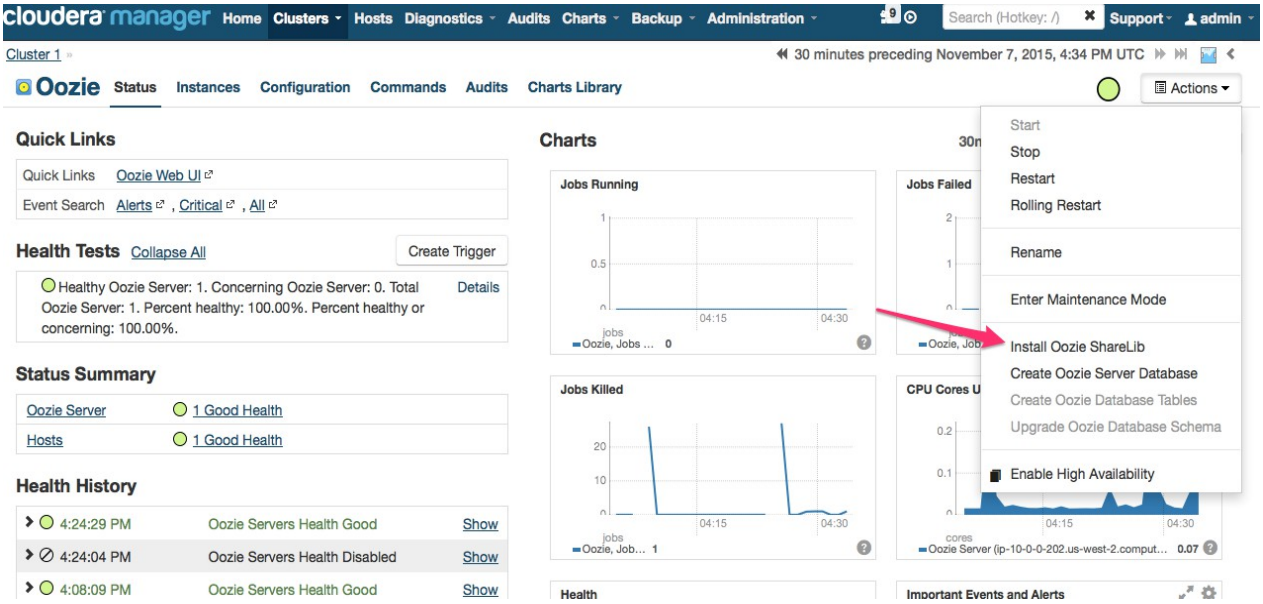
Starting Pig Command Line

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

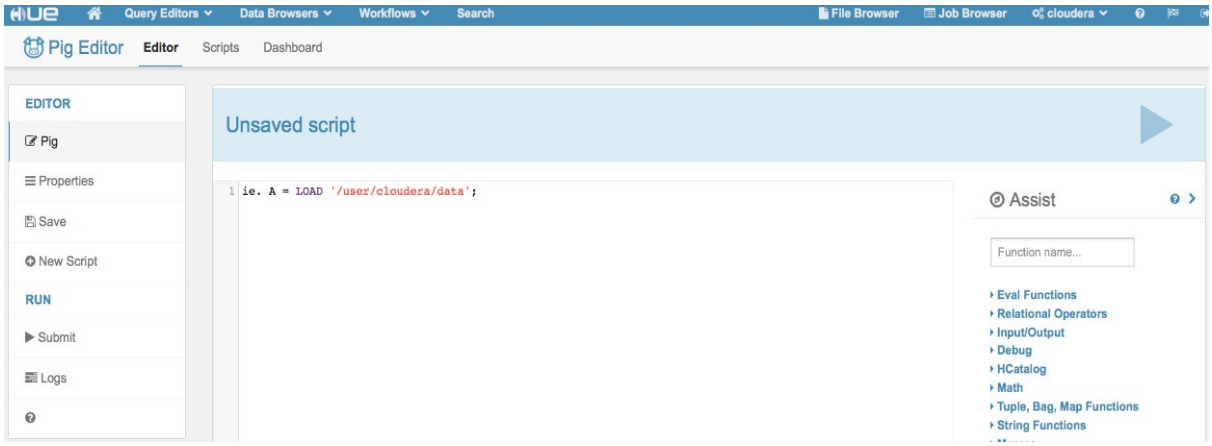
Writing a Pig Script for wordcount

```
A = load './input/*';
B = foreach A generate flatten(TOKENIZE((chararray)$0)) as word;
C = group B by word;
D = foreach C generate COUNT(B), group;
store D into './output/wordcountPig';
```

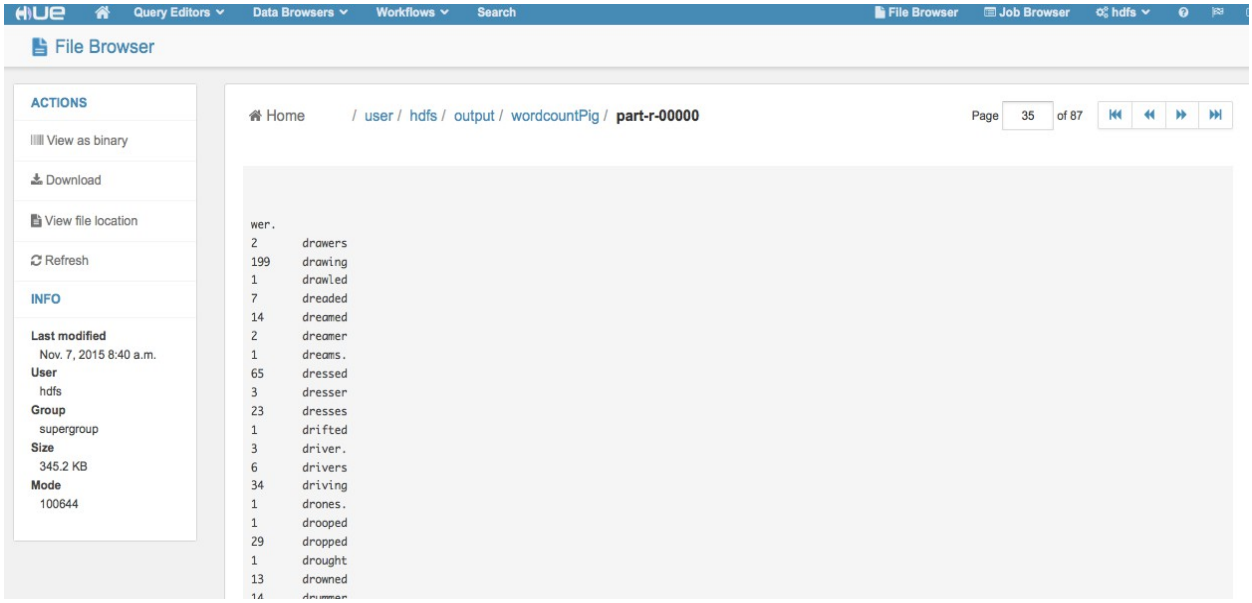

To run Pig from Hue
Install Ozzie Sharelib from Cloudera Manager



Starting Pig from Hue



Starting Pig from Hue



Lecture
Understanding HBase

Introduction

An open source, non-relational, distributed database

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

HBase Features

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

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:

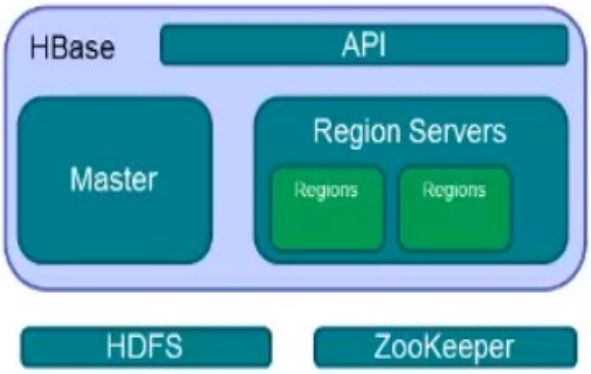
ID (Primary key)	Last name	First name	Password	Timestamp
1234	Smith	John	Hello, world!	20130710
5678	Cooper	Joyce	wysiwyg	20120825
5678	Cooper	Joyce	wisiwig	20130916

- 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

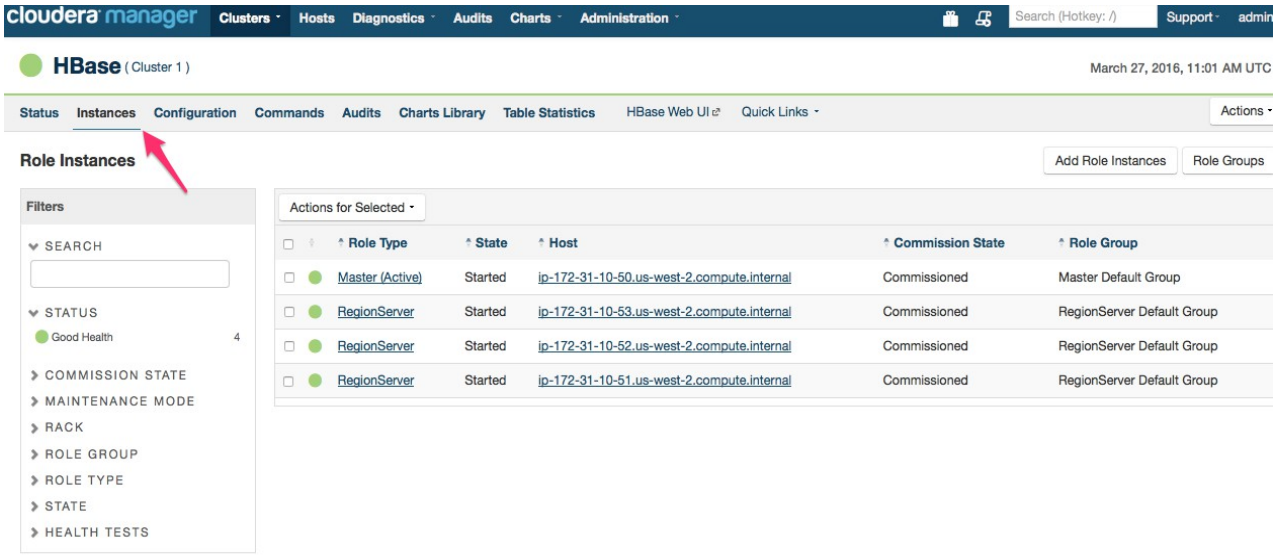
Configure Hue to show Hbase browser

From Cloudera Manager : Add Thrift server

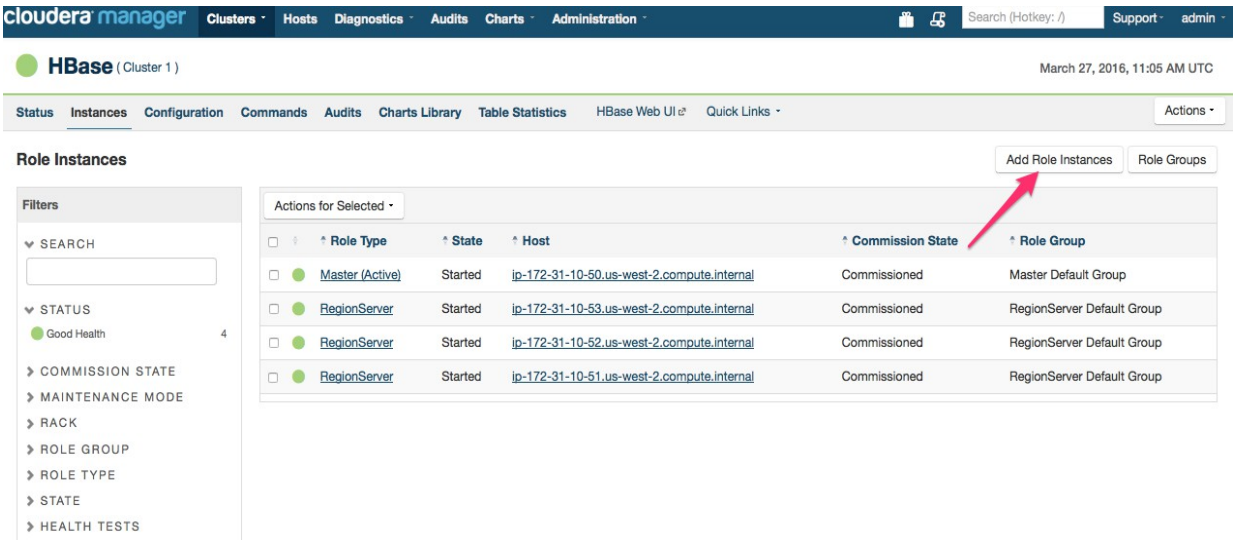
The Thrift Server role is not added by default when you install HBase. To add the Thrift Server role:

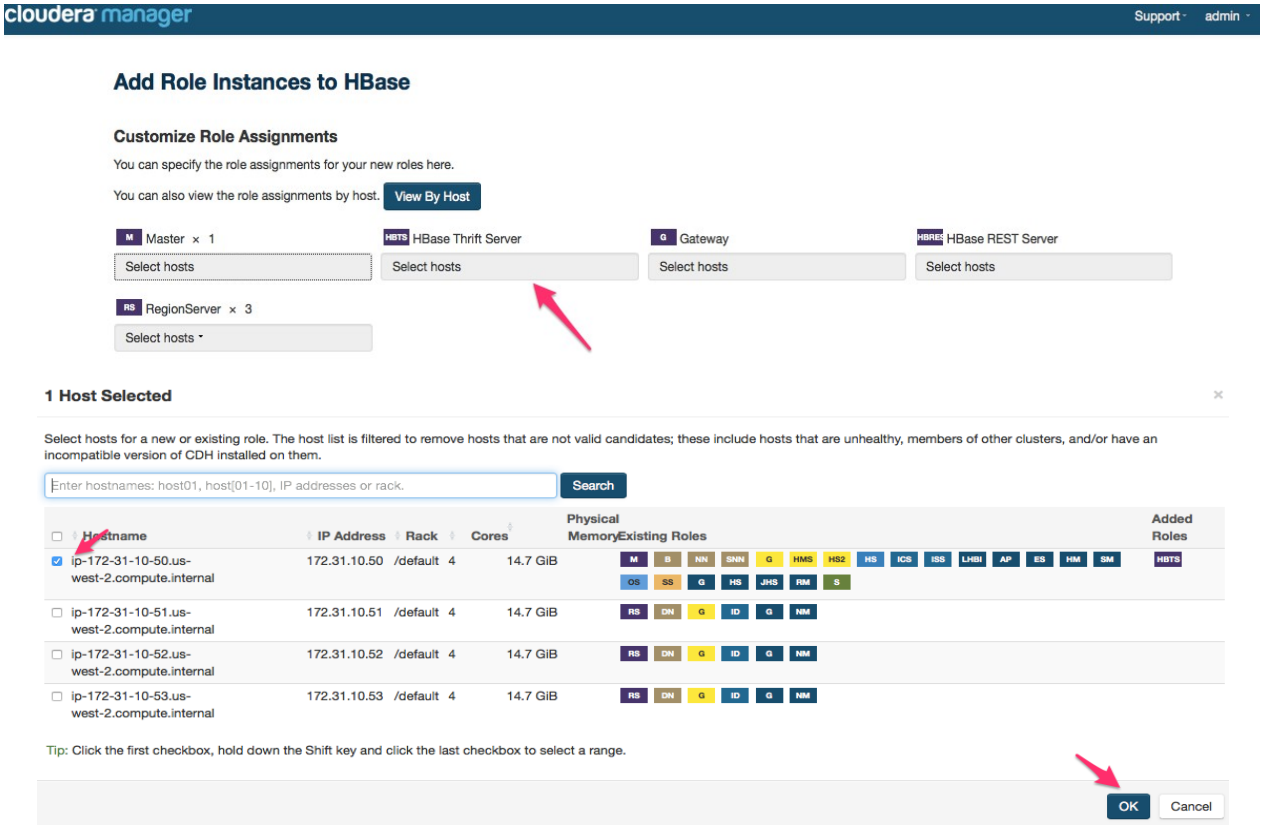
1. Go to the HBase service.
2. Click the **Instances** tab.
3. Click the **Add Role Instances** button.
4. Select the host(s) where you want to add the Thrift Server role (you only need one for Hue) and click **Continue**. The Thrift Server role should appear in the instances list for the HBase server.
5. Select the Thrift Server role instance.
6. Select **Actions for Selected** > **Start**.

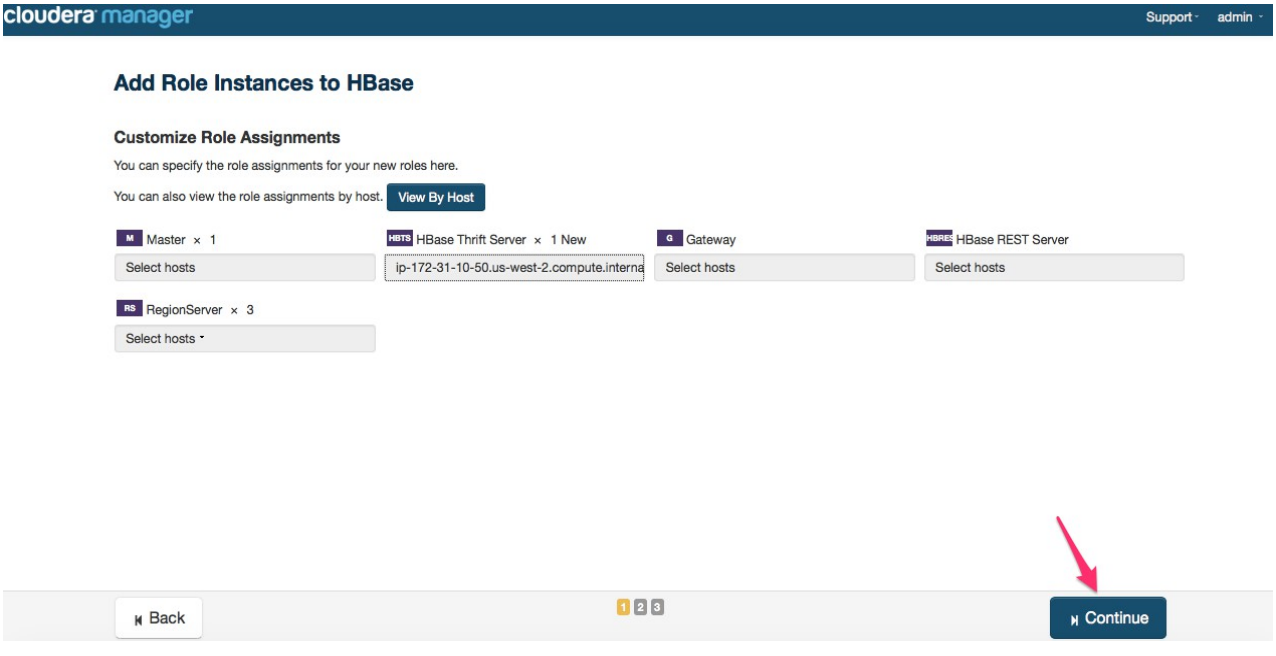
Select Instance

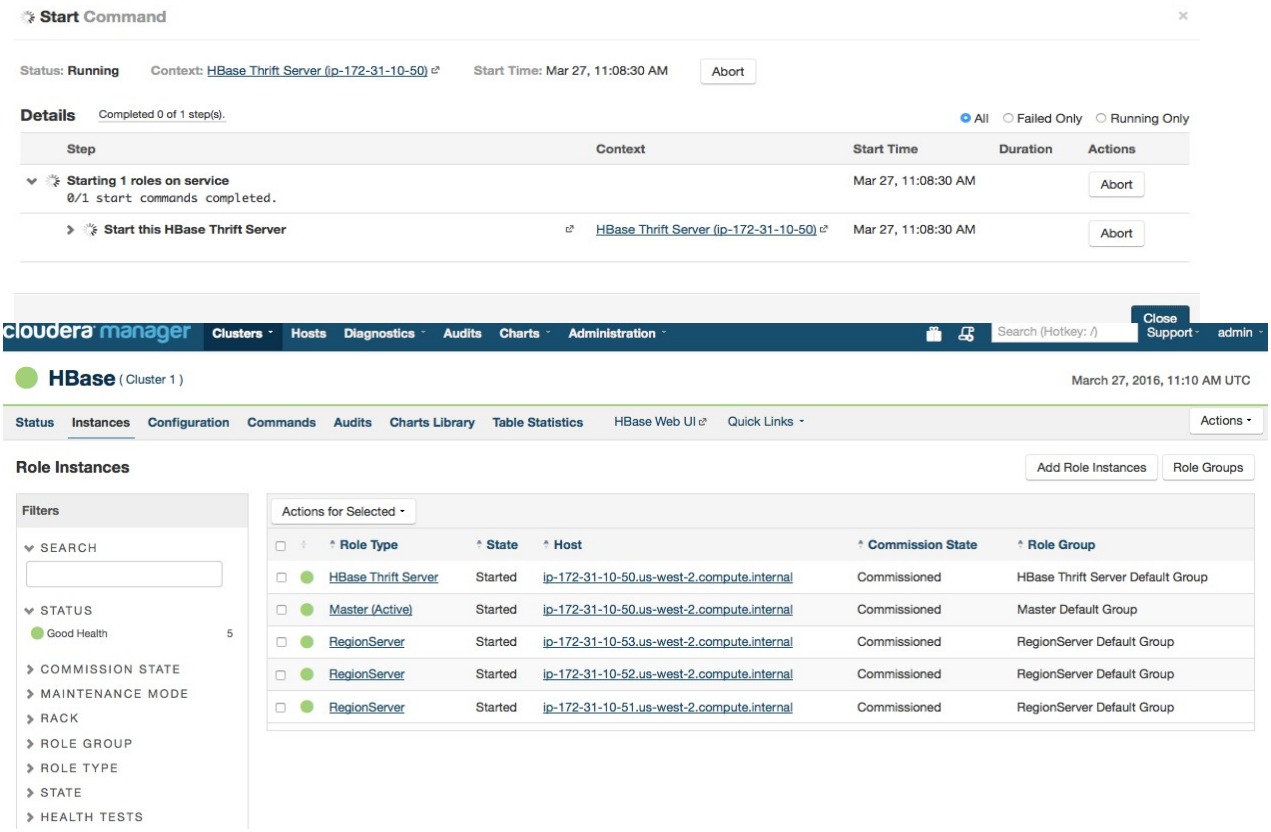


Add Role Instances




cloudera manager
Support
admin
Add Role Instances to HBase
Customize Role Assignments
You can specify the role assignments for your new roles here.
You can also view the role assignments by host.
View By Host
Master × 1
HBase Thrift Server
Gateway
HBase REST Server
Select hosts
RegionServer × 3
1 Host Selected
Select hosts for a new or existing role. The host list is filtered to remove hosts that are not valid candidates; these include hosts that are unhealthy, members of other clusters, and/or have an incompatible version of CDH installed on them.
Enter hostnames: host01, host[01-10], IP addresses or rack.
Search
Hostname
IP Address
Rack
Cores
Physical Memory
Existing Roles
Added Roles
ip-172-31-10-50.us-west-2.compute.internal
172.31.10.50
/default
4
14.7 GiB
ip-172-31-10-51.us-west-2.compute.internal
172.31.10.51
/default
4
14.7 GiB
ip-172-31-10-52.us-west-2.compute.internal
172.31.10.52
/default
4
14.7 GiB
ip-172-31-10-53.us-west-2.compute.internal
172.31.10.53
/default
4
14.7 GiB
Tip: Click the first checkbox, hold down the Shift key and click the last checkbox to select a range.
OK
Cancel


cloudera manager
Support
admin
Add Role Instances to HBase
Customize Role Assignments
You can specify the role assignments for your new roles here.
You can also view the role assignments by host.
View By Host
Master × 1
HBase Thrift Server × 1 New
Gateway
HBase REST Server
Select hosts
ip-172-31-10-50.us-west-2.compute.interna
RegionServer × 3
Back
1 2 3
Continue

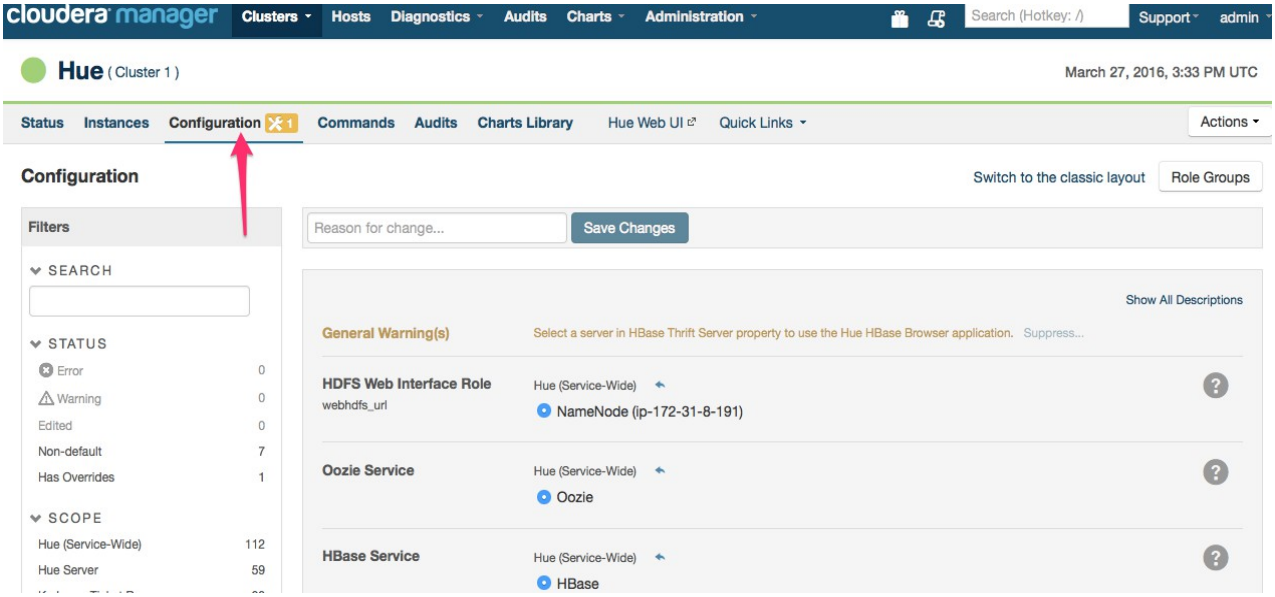


Configure Hue Services for Hbase browser

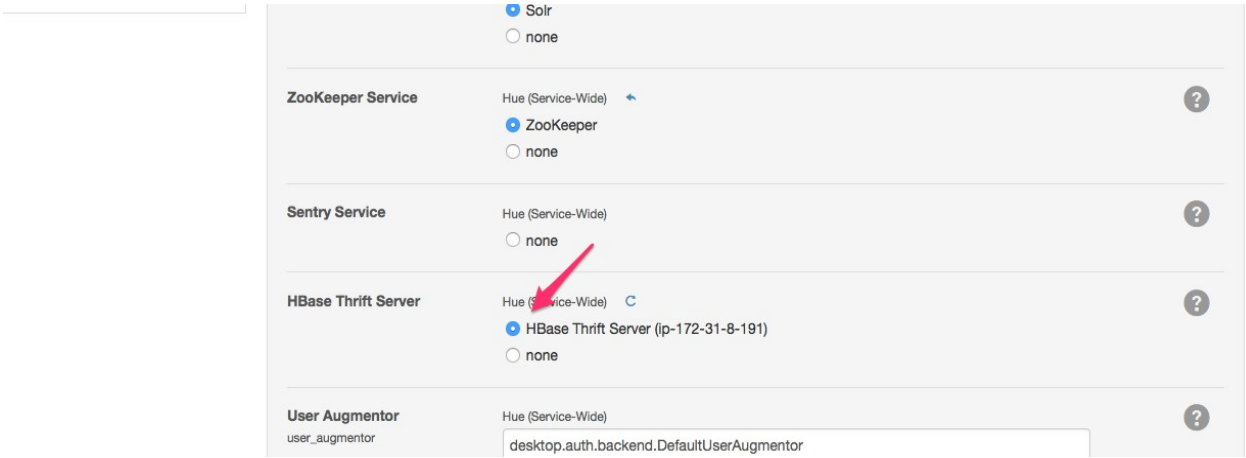
To configure Hue for the HBase Browser:

1. Select the **Hue** service, then under the **Configuration** tab select **View and Edit**.
2. Go to the **Service-Wide** category.
3. For the **HBase Service** property, make sure it is set to the HBase service for which you enabled the Thrift Server role(if you have more than one HBase service instance).
4. In the **HBase Thrift Server** property, click in the edit field and select the Thrift Server role that Hue should use.
5. **Save Changes** to have these configurations take effect.

From Hue Services: select configuration tab

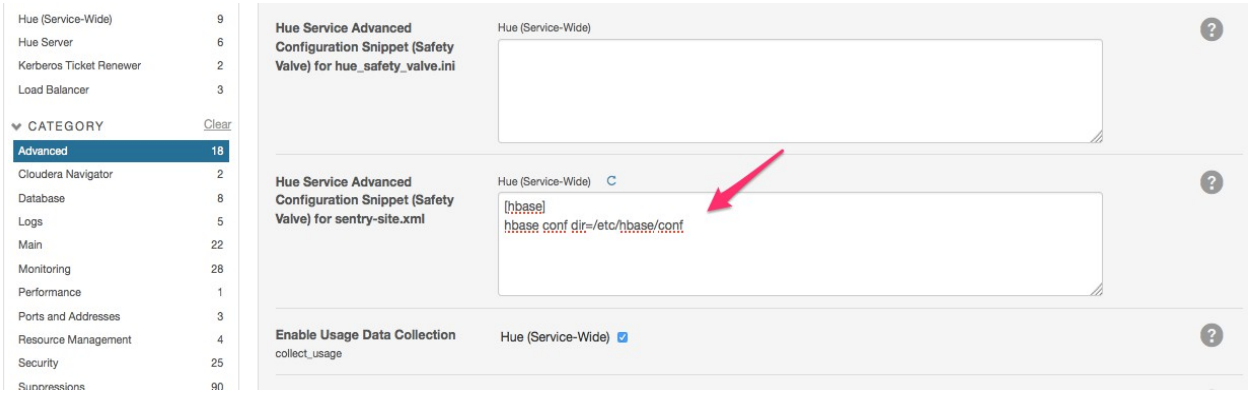


Select the HBase Thrift Server

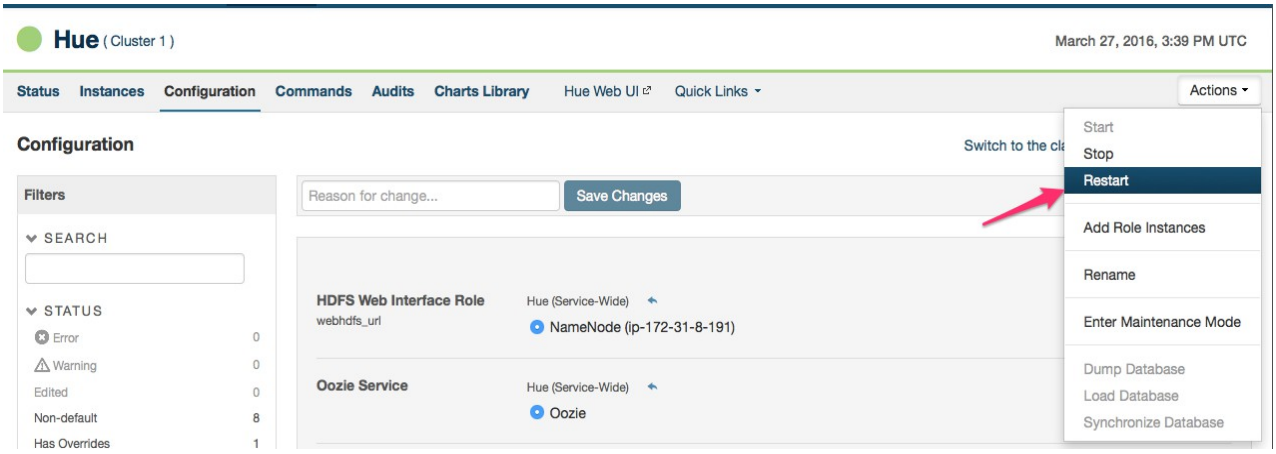


Configure Hue Services

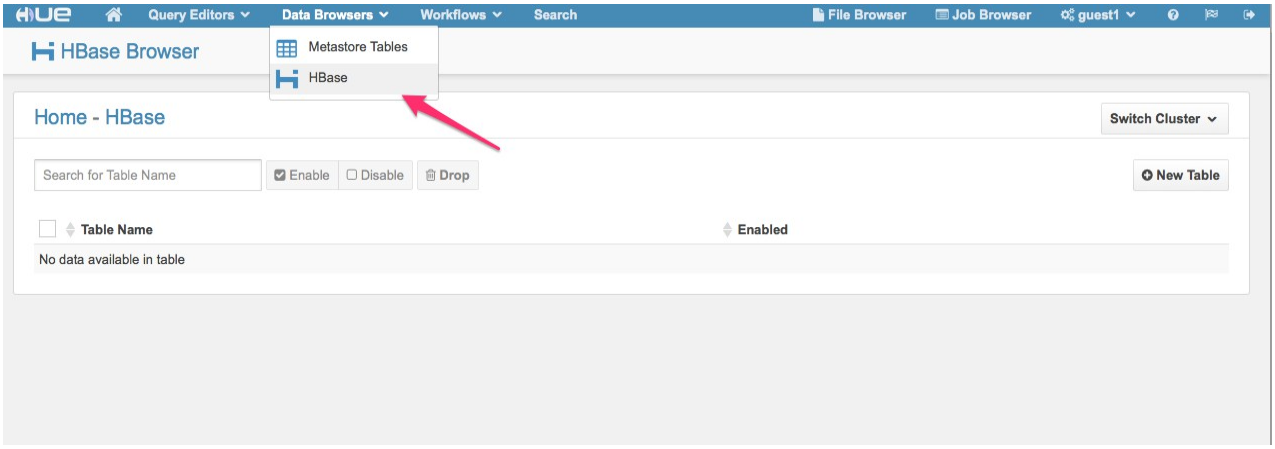
Edit as seen: then click Save change and Restart



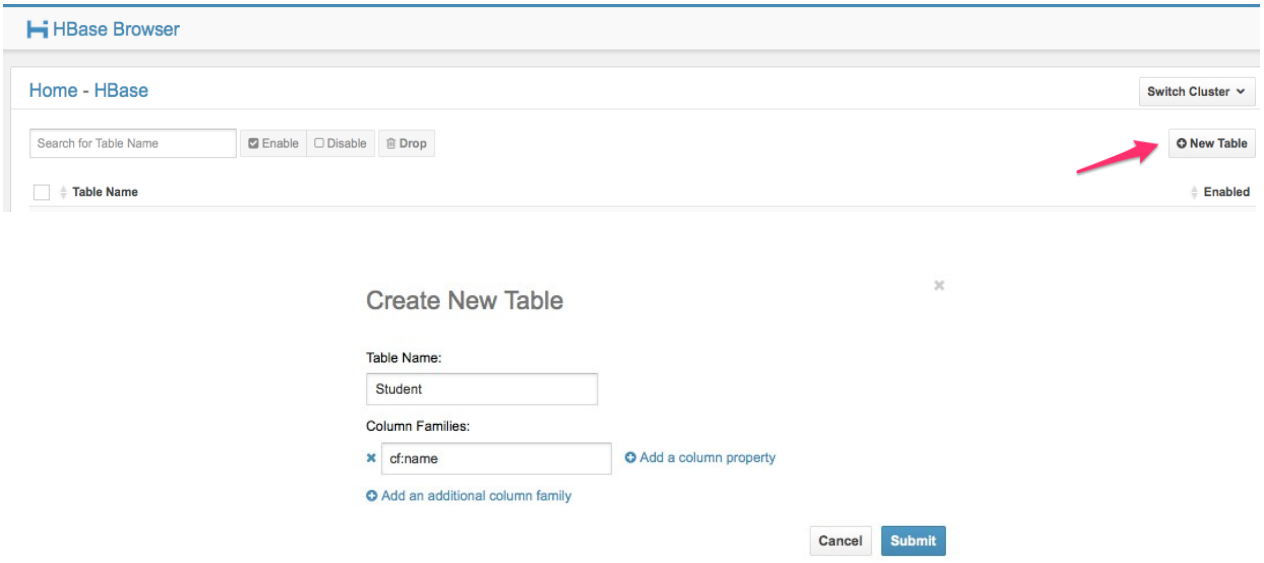
Restart Hue service



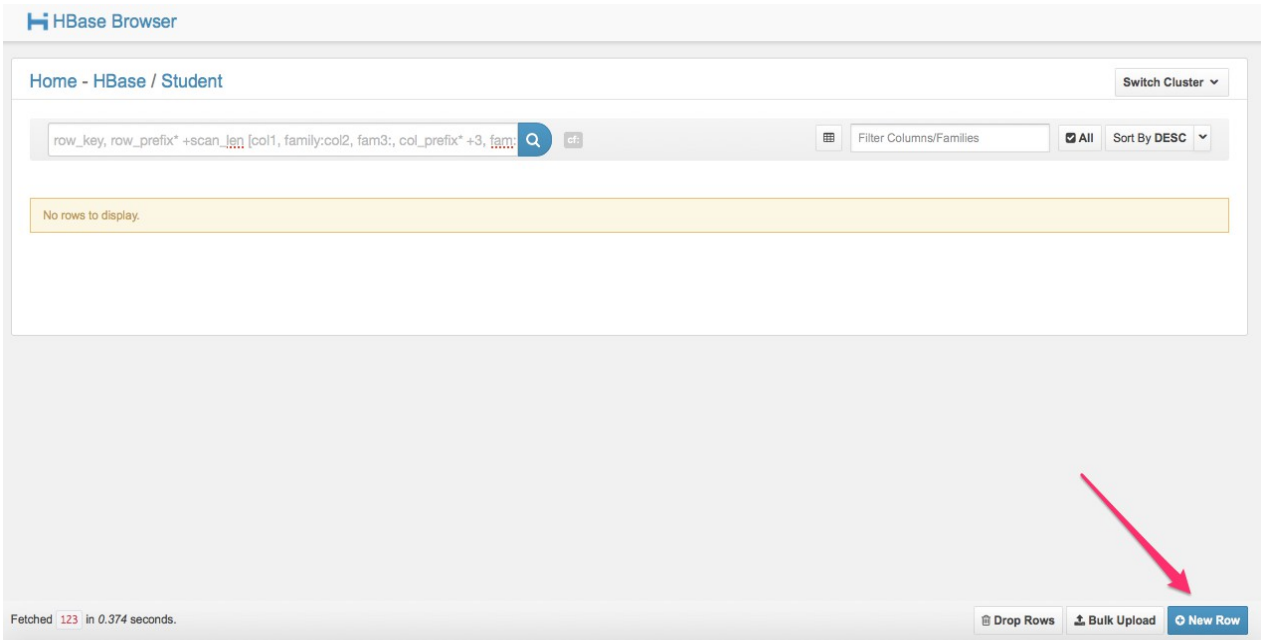
Running HBase Browser



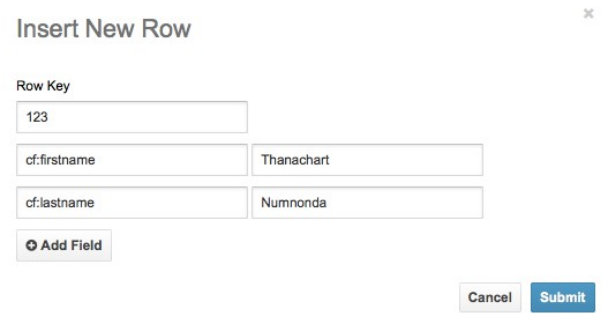
Create a table in HBase

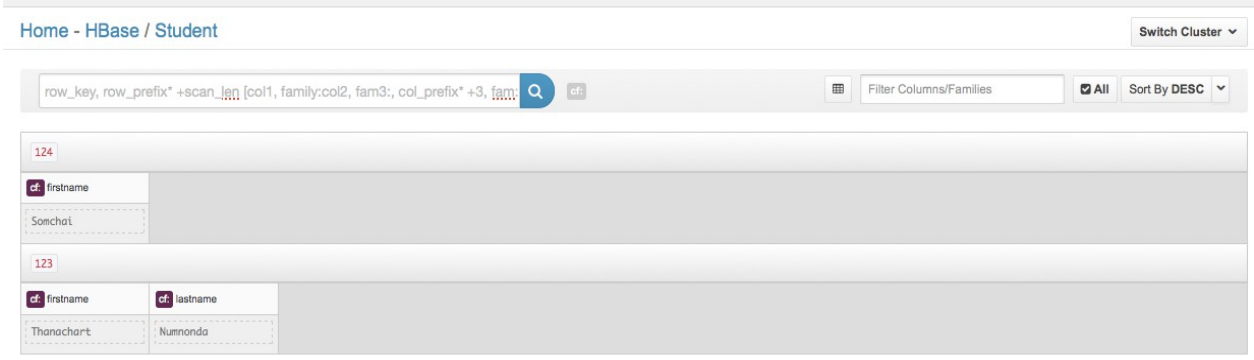


Insert a new row in a table



Add field into a new row





Lecture: Understanding Sqoop

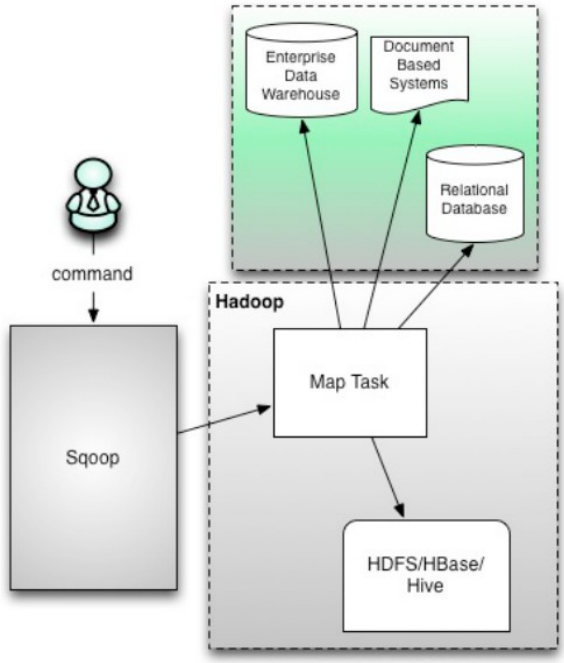
Introduction

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

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

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

Architecture Overview



Hive.apache.org

Sqoop Benefit

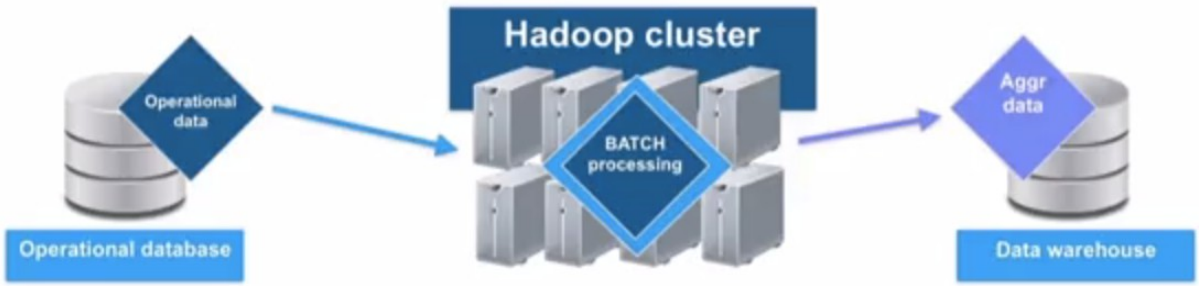
- Leverages RDBMS metadata to get the column data types
- It is simple to script and uses SQL
- It can be used to handle change data capture by importing daily transactional data to Hadoop
- It uses MapReduce for export and import that enables parallel and efficient data movement

Sqoop Mode

- Sqoop import: Data moves from RDBMS to Hadoop
- Sqoop export: Data moves from Hadoop to RDBMS

Use Case #1: ETL for Data Warehouse

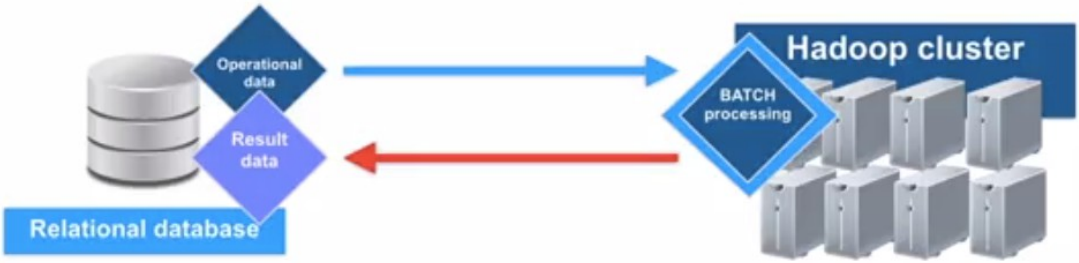
- Transform operational data for data warehouse reports in Hadoop as the batch transformation "engine"



Source: Mastering Apache Sqoop, David Yahalom, 2016

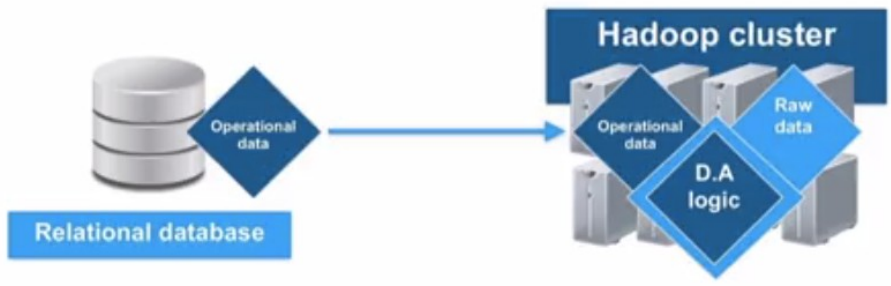
Use Case #2: ELT

- Extract operational data from RDBMS, process in Hadoop, return ***result*** to RDBMS



Use Case #3: Data Analysis

- Copy real-time data from RDBMS, combine with raw data on Hadoop using complex data analysis logic (not just SQL!)



Source: Mastering Apache Sqoop, David Yahalom, 2016

Use Case #4: Data Archival

- Move data from RDBMS after it expires to Hadoop, keeping the RDBMS "clean and lean"

Source: Mastering Apache Sqoop, David Yahalom, 2016

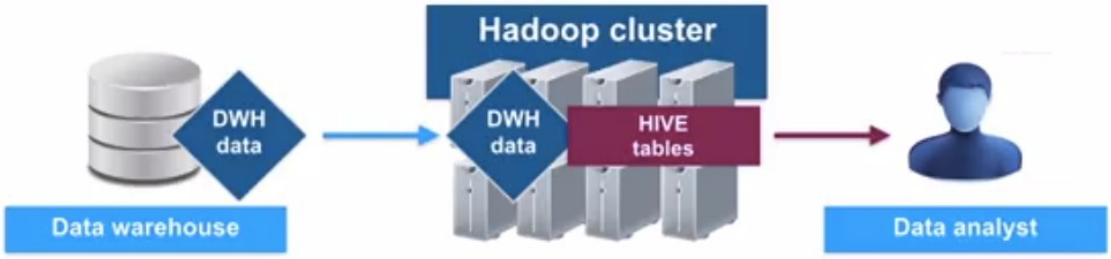
Use Case #5: Data Consolidation

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

Source: Mastering Apache Sqoop, David Yahalom, 2016

Use Case #6: Move reports to Hadoop

- Easily allow traditional data analysis and business intelligence using Hadoop's power

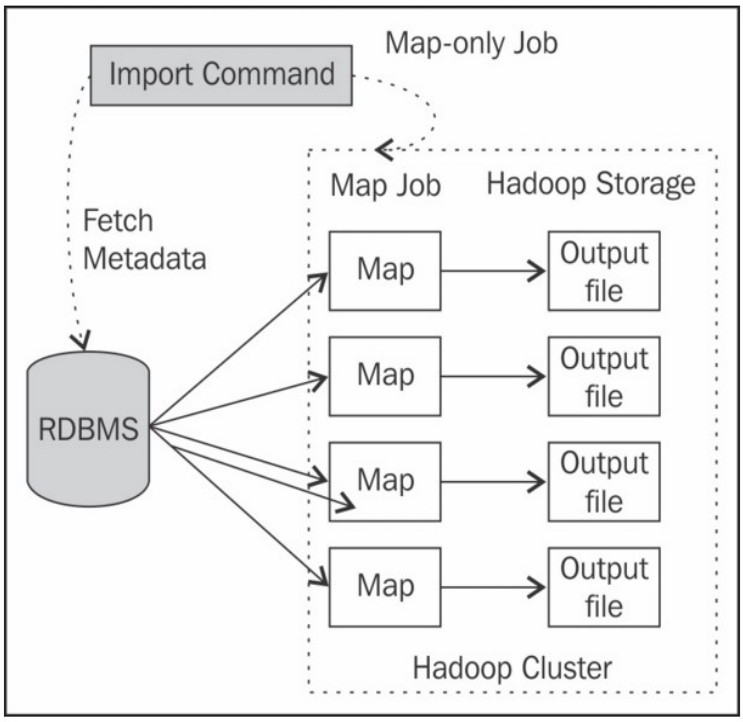


Source: Mastering Apache Sqoop, David Yahalom, 2016

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.

Architecture of the import process



Incremental import

Parameter/argument	Description
`--check-column <column-name>`	The value of this column is used to determine the rows to be imported during the import process.
`--incremental <incremental-type>`	Specifies the type of incremental mode. Possible values are `append` and `lastmodified`.
`--last-value <value>`	Specifies the last value or the maximum value of the `check` column from the previous import. All the records whose `check` column value is greater than the value of the `–last-value` argument will be imported to HDFS.

```
bin/sqoop import -connect jdbc:mysql://localhost:3306/db1 -username root -password password -
table student -target-dir /user/abc/student -columns "student_id,address,name"  --incremental
lastmodified -last-value "2012-11-06 19:01:35"-check-column col4
```

Export Commands

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.

Hands-On: Loading Data from DBMS to Hadoop HDFS

MySQL RDS Server on AWS

A RDS Server is running on AWS with the following configuration

> database: imc_db

> username: admin

> password: imcinstitute

>addr: imcdb.cmw65obdqfnx.us-west-2.rds.amazonaws.com

[This address may change]

DB Instances > imcinstitutedb

Details | Recent Events & Logs

Endpoint: imcinstitutedb.cmw65obdqfnx.us-west-2.rds.amazonaws.com:3306 (authorized)

Configuration Details		Security and Network	
Engine	MySQL 5.6.22	Availability Zone	us-west-2c
License Model	General Public License	VPC	vpc-cd510ca5
Created Time	March 24, 2015 at 9:50:55 PM UTC+7	Subnet Group	default (Complete)
DB Name	imc_db	Subnets	subnet-c0510ca8 subnet-ce510ca6 subnet-cf510ca7
Username	admin	Security Groups	rds-launch-wizard (sg-59dee33c) (active)
Option Group	default:mysql-5-6 (in-sync)	Publicly Accessible	Yes
Parameter Group	default.mysql5.6 (in-sync)	Port	3306
		Certificate Authority	rds-ca-2015 (Mar 5, 2020)

Table in MySQL RDS

Table 1 > country_tbl : Columns: id, country

Table 2 > movie_rating : Columns: userid, movieid, rating, timestamp

Result Set Filter: Search Export: Fetch rows:

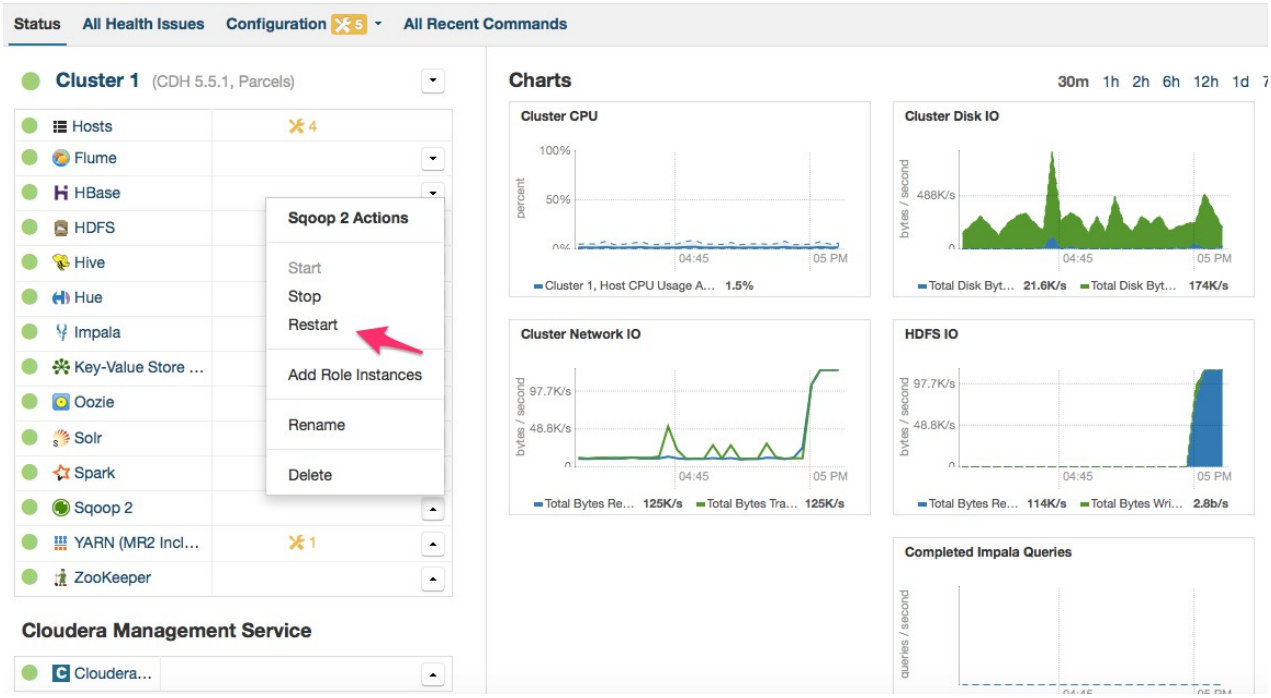
userid	movieid	rating	timestamp
186	302	3	891717742
22	377	1	878887116
244	51	2	880606923
166	346	1	886397596
298	474	4	884182806
115	265	2	881171488
253	465	5	891628467
305	451	3	886324817
6	86	3	883603013
62	257	2	879372434
286	1014	5	879781125

Installing DB driver for Sqoop

```
$ cd /var/lib/sqoop

$ sudo wget
https://www.dropbox.com/s/6zrp5nerrwfixcj/mysql-connector-java-5.1.23-bin.jar

$ ls
```

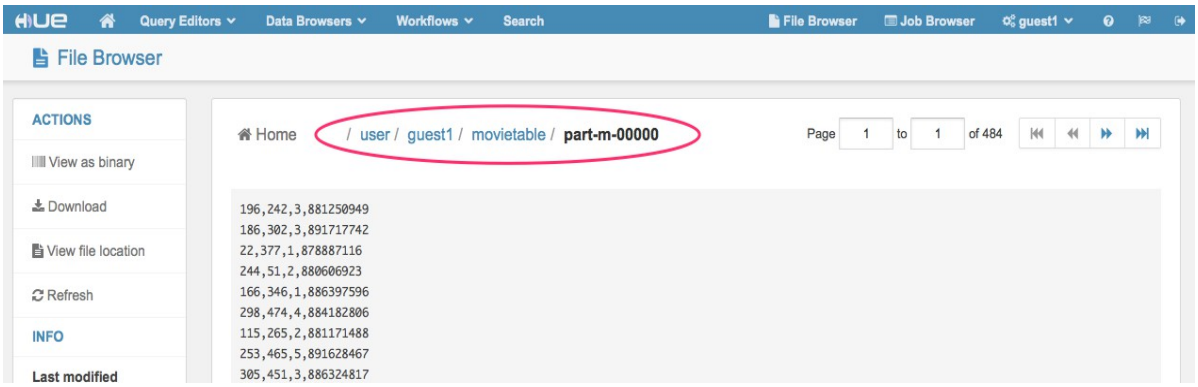
```
ubuntu@ip-10-0-0-52:/var/lib/sqoop$ ls
mysql-connector-java-5.1.23-bin.jar
```


Restart Sqoop2 service



Importing data from MySQL to HDFS

```
$sudo -u hdfs sqoop import --connect
jdbc:mysql://imcdb.cmw65obdqfnx.us-west-
2.rds.amazonaws.com:3306/imc_db  --username admin --password
imcinstitute --table movie_rating --target-dir /user/guest1/movietable
-m 1
```



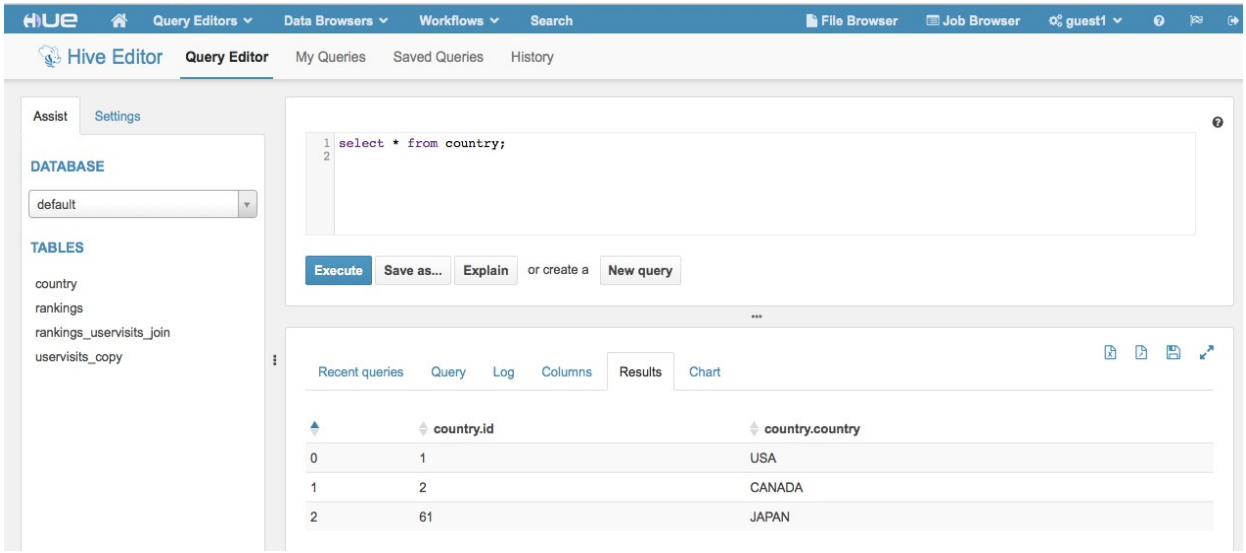
Importing data from MySQL to Hive Table

```
$sudo -u hdfs sqoop import --connect
jdbc:mysql://imcdb.cmw65obdqfnx.us-west-
2.rds.amazonaws.com:3306/imc_db  --username admin --password
imcinstitute --table country_tbl --hive-import --hive-table
thanachart.country -m 1

Warning: /usr/lib/hbase does not exist! HBase imports will fail.
Please set $HBASE_HOME to the root of your HBase installation.
Warning: $HADOOP_HOME is deprecated.

Enter password: <enter here>
```

Reviewing data from Hive Table

```
ubuntu@ip-172-31-12-11:~$ hive

Logging initialized using configuration in jar:file:/usr/local/apache-hive-1.1.0-
bin/lib/hive-common-1.1.0.jar!/hive-log4j.properties
hive> show tables;
OK
country
test_tbl
users
Time taken: 1.122 seconds, Fetched: 3 row(s)
hive> select * from country;
OK
1       USA
93      Afghanistan
66      Thailand
65      Singapore
61      Japan
Time taken: 1.282 seconds, Fetched: 5 row(s)
hive>
```


Running from Hue: Beewax



Importing data from MySQL to HBase

```
$sudo -u hdfs sqoop import --connect
jdbc:mysql://imcdb.cmw65obdqfnx.us-west-
2.rds.amazonaws.com:3306/imc_db  --username admin --password
imcinstitute --table country_tbl --hbase-table
thanachart.hbase_country --column-family hbase_country_cf --hbase-row-
key id --hbase-create-table  -m 1
```

Start HBase

```
$hbase shell
```


Viewing Hbase data

```
hbase(main):003:0> describe 'hbase_country'
Table hbase_country is ENABLED
hbase_country
COLUMN FAMILIES DESCRIPTION
{NAME => 'hbase_country_cf', DATA_BLOCK_ENCODING => 'NONE', BLOOMFILTER => 'ROW
', REPLICATION_SCOPE => '0', VERSIONS => '1', COMPRESSION => 'NONE', MIN_VERSIO
NS => '0', TTL => 'FOREVER', KEEP_DELETED_CELLS => 'FALSE', BLOCKSIZE => '65536
', IN_MEMORY => 'false', BLOCKCACHE => 'true'}
1 row(s) in 0.2520 seconds

hbase(main):004:0> scan 'hbase_country'
ROW                   COLUMN+CELL
 1                    column=hbase_country_cf:country, timestamp=1453743508379,
                      value=USA
 2                    column=hbase_country_cf:country, timestamp=1453743508379,
                      value=CANADA
 61                   column=hbase_country_cf:country, timestamp=1453743508379,
                      value=JAPAN
3 row(s) in 0.0700 seconds

hbase(main):005:0>
```

Lecture: Understanding Flume

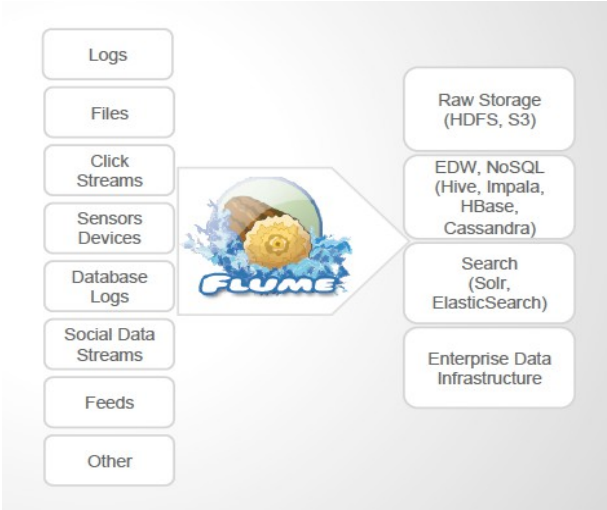
Introduction

Apache Flume is:

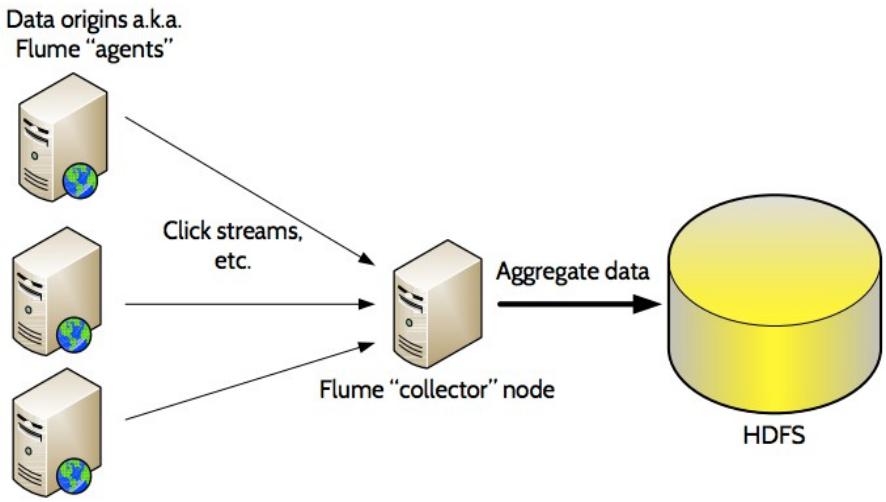
- **A distributed data transport and aggregation system for event- or log-structured data**
- **Principally designed for continuous data ingestion into Hadoop… But more flexible than that**

What is Flume?

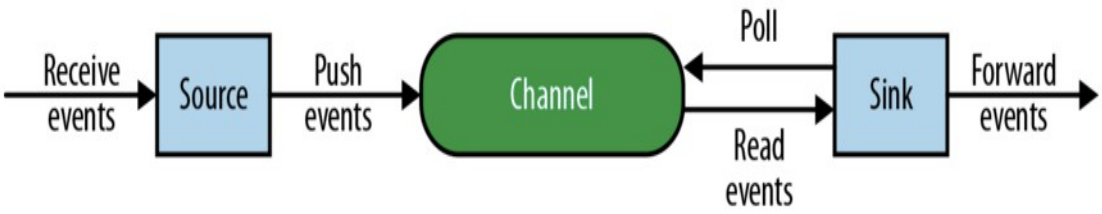
- Apache Flume is a continuous data ingestion system that is...
  - open-source,
  - reliable,
  - scalable,
  - manageable,
  - Customizable,
  - and designed for Big Data ecosystem



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.

Source: Using Flume, Hari Shreedharan, 2014

Sources

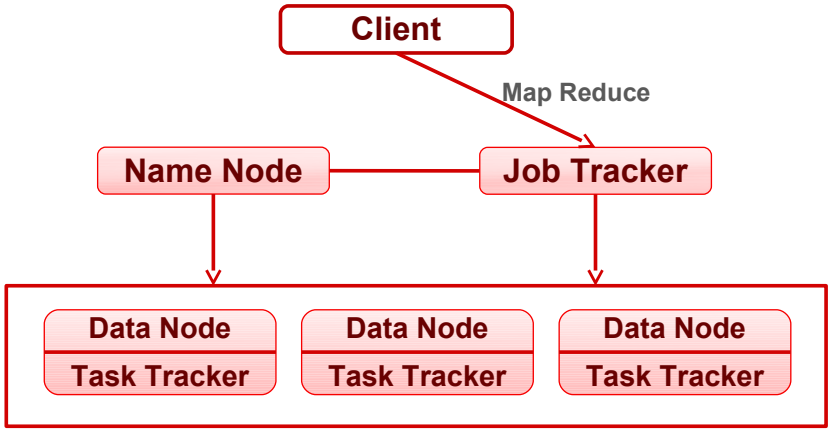
- Different Source types:
- Require at least one channel to function
- Specialized sources for integrating with well-known systems.
  - Example: Spooling Files, Syslog, Netcat, JMS
  - Auto-Generating Sources: Exec, SEQ
  - IPC sources for Agent-to-Agent communication: Avro, Thrift

Channel

- Different Channels offer different levels of persistence:
  - Memory Channel
  - File Channel:
- Eventually, when the agent comes back data can be accessed.
- Channels are fully transactional
- Provide weak ordering guarantees
- Can work with any number of Sources and Sinks

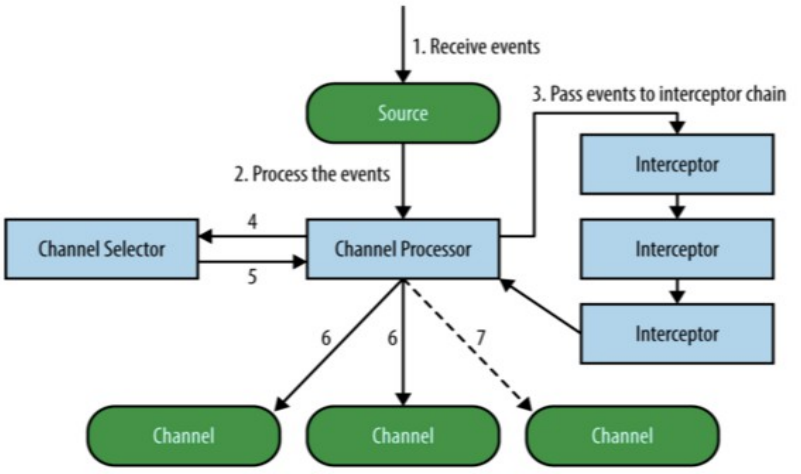
Sink

- Different types of Sinks:
  - Terminal sinks that deposit events to their final destination. For example: HDFS, HBase, Morphline-Solr, Elastic Search
  - Sinks support serialization to user's preferred formats.
  - HDFS sink supports time-based and arbitrary bucketing of data while writing to HDFS.
  - IPC sink for Agent-to-Agent communication: Avro, Thrift
- Require exactly one channel to function



Lecture: Understanding Map Reduce Processing

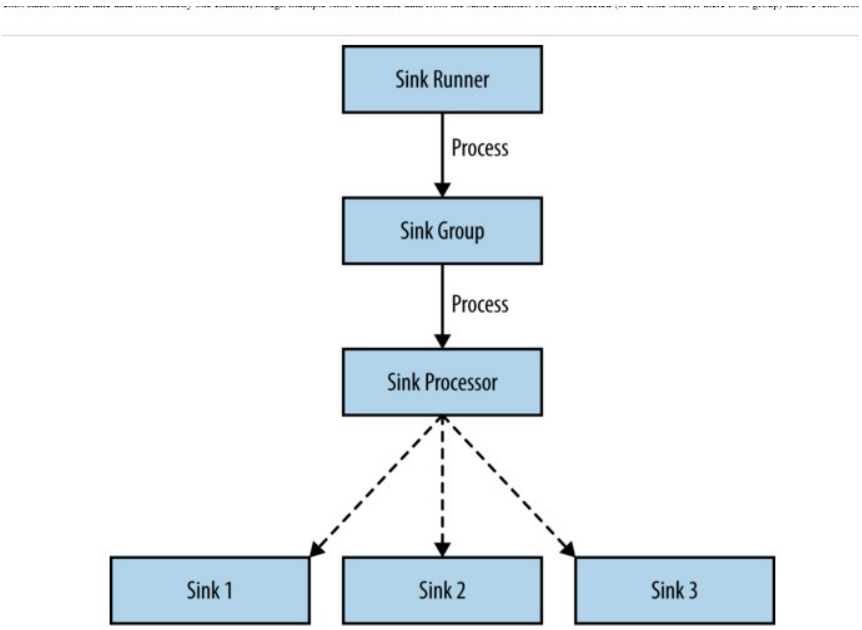
Flume Process



4. Pass each event to channel selector
5. Return list of channels the event is to be written to
6. Write all events that need to go to each required channel. Only one transaction is opened. with each channel, and all events to a channel are written as part of that transaction.
7. Repeat the same with optional channels

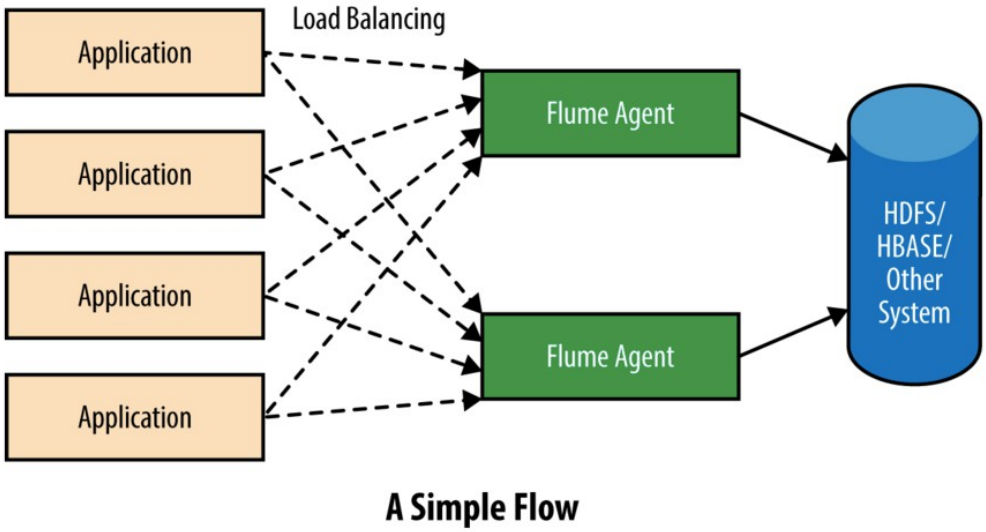
Source: Using Flume, Hari Shreedharan, 2014

Flume Process



Source: Using Flume, Hari Shreedharan, 2014

Flow



A Simple Flow

Source: Using Flume, Hari Shreedharan, 2014

Flume terminology

- 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.

Flume Agent Configuration : Example

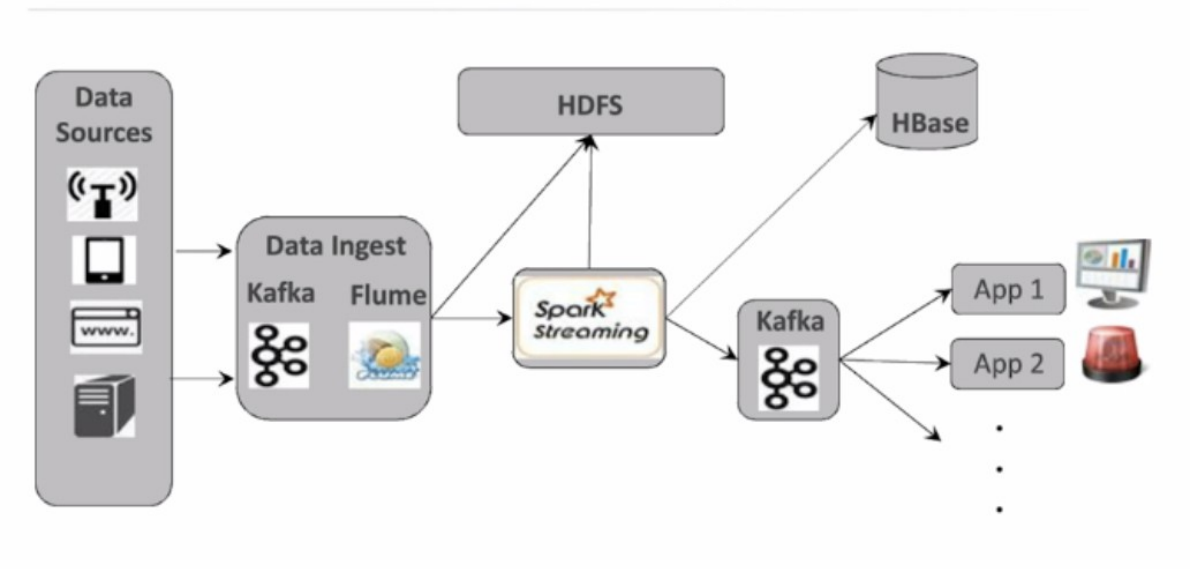
```
agent.sources = httpSrc
agent.channels = memory1 memory2
agent.sinks = hdfsSink hbaseSink

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

```
# 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
```

Odiago

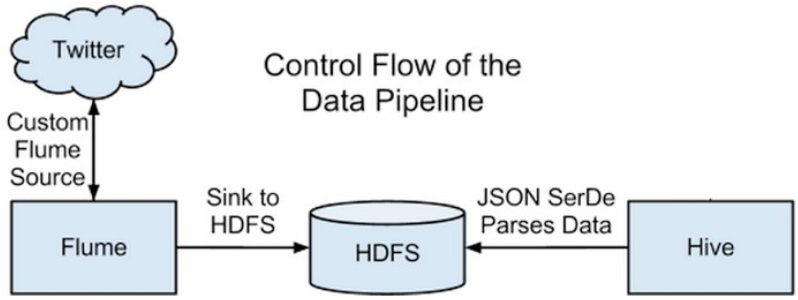
Stream Processing Architecture



Odiago

Hands-On: Loading Twitter Data to Hadoop HDFS

Exercise Overview



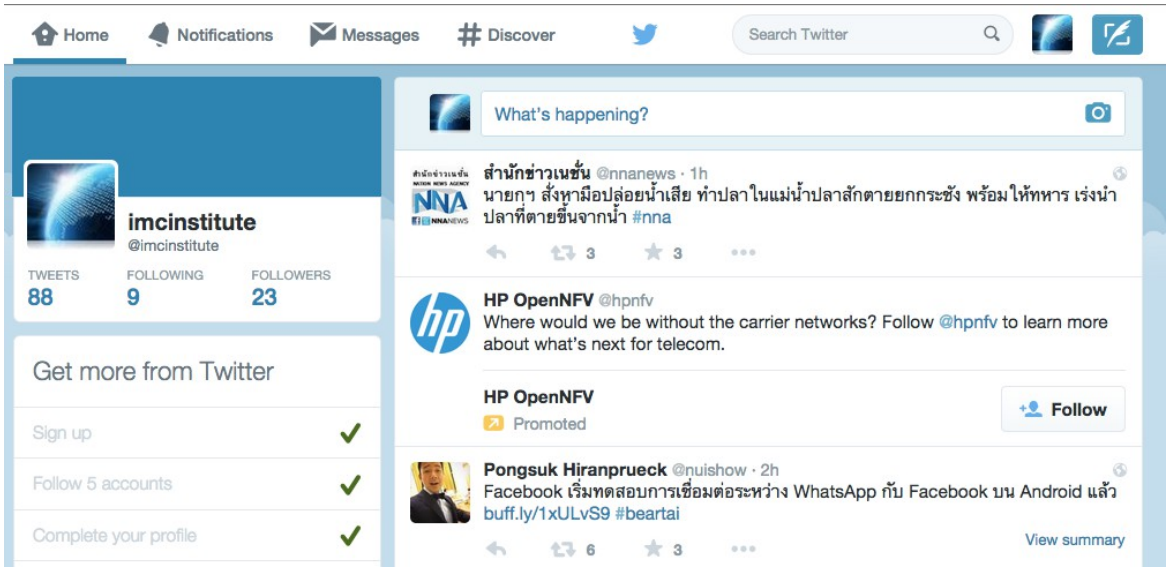
Hive.apache.org

Installing Pre-built version of flume

```
$ wget
http://files.cloudera.com/samples/flume-sources-1.0-SNAPSHOT.jar

$ sudo cp flume-sources-1.0-SNAPSHOT.jar
/opt/cloudera/parcels/CDH-5.5.1-1.cdh5.5.1.p0.11/lib/flume-
ng/lib/

$sudo cp /etc/flume-ng/conf/flume-env.sh.template
/etc/flume-ng/conf/flume-env.sh
```

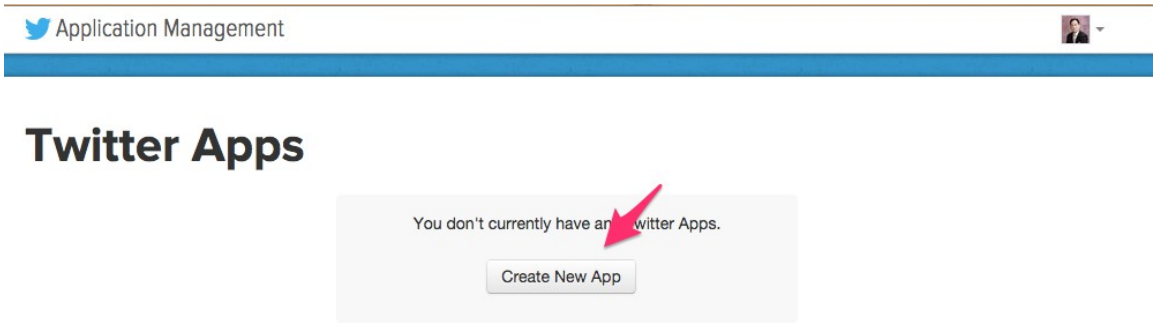

Create a new Twitter App

Login to your Twitter @ twitter.com



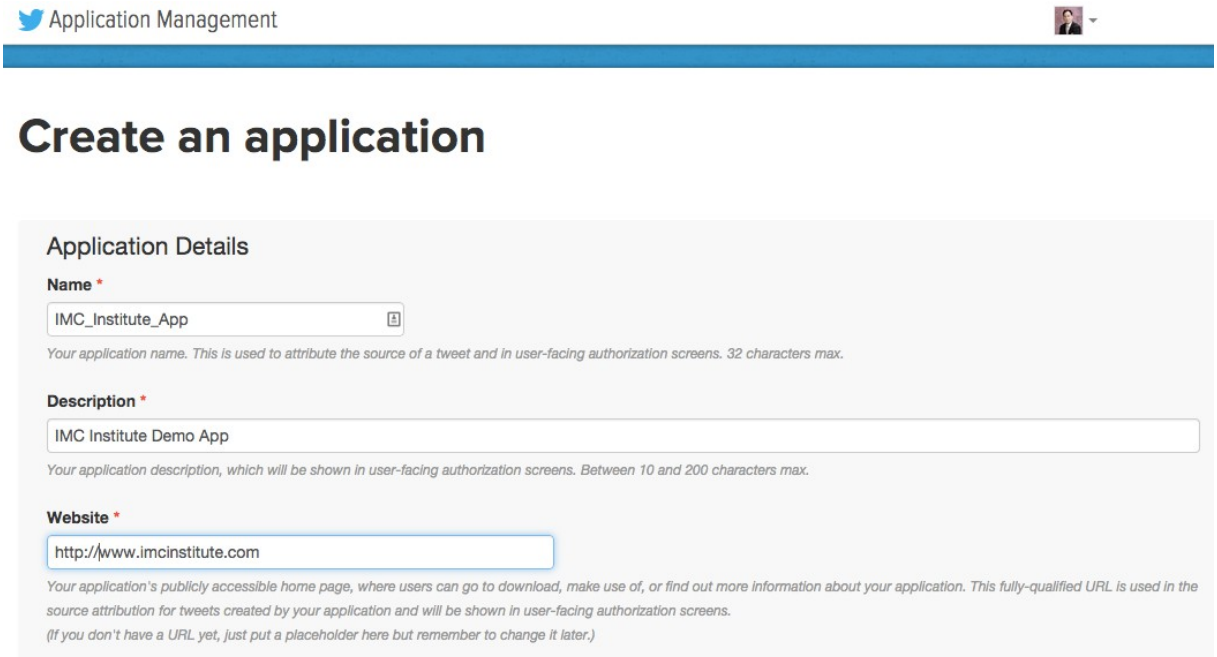
Create a new Twitter App (cont.)

Create a new Twitter App @ apps.twitter.com



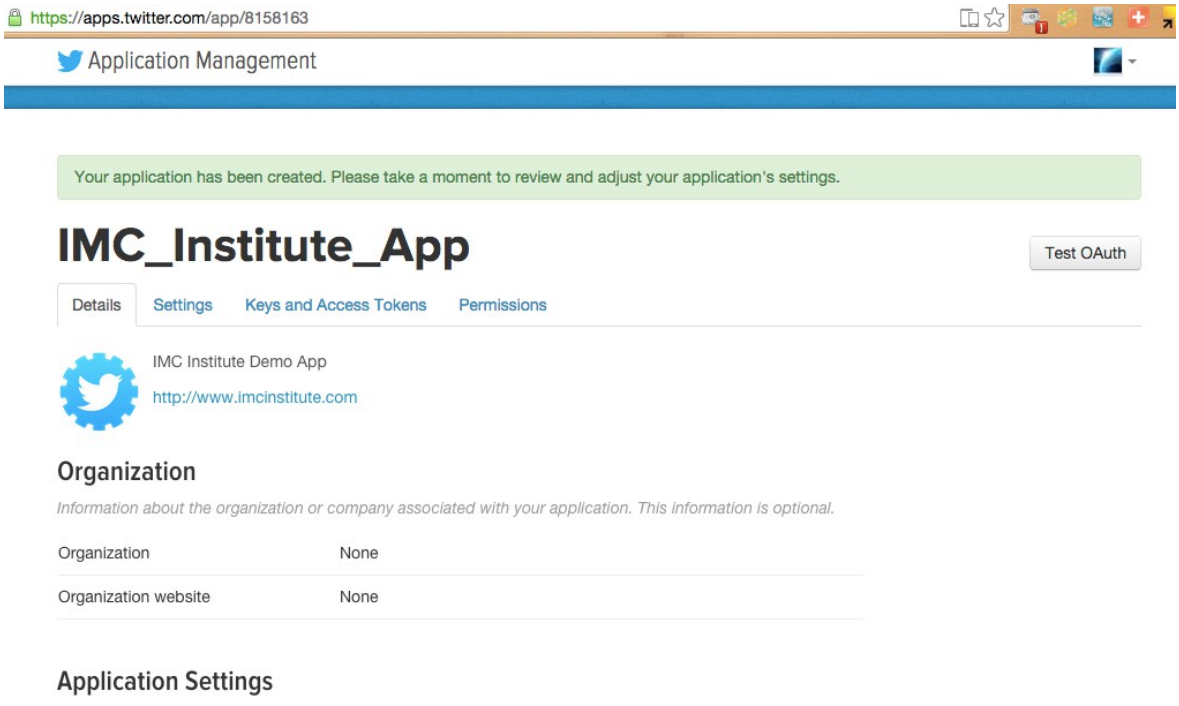
Create a new Twitter App (cont.)

Enter all the details in the application:



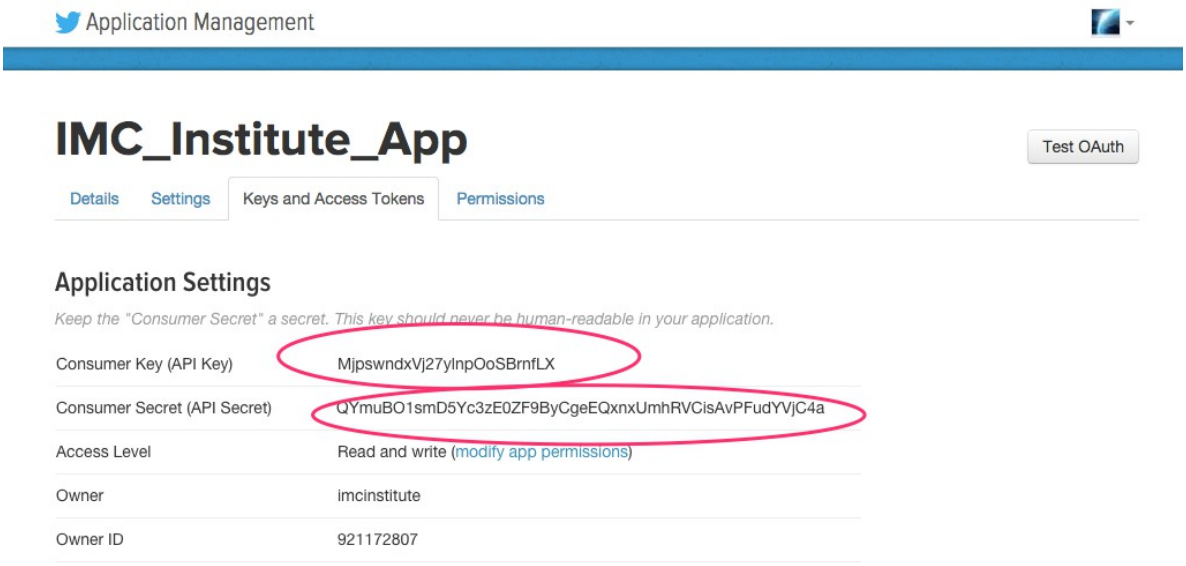
Create a new Twitter App (cont.)

Your application will be created:



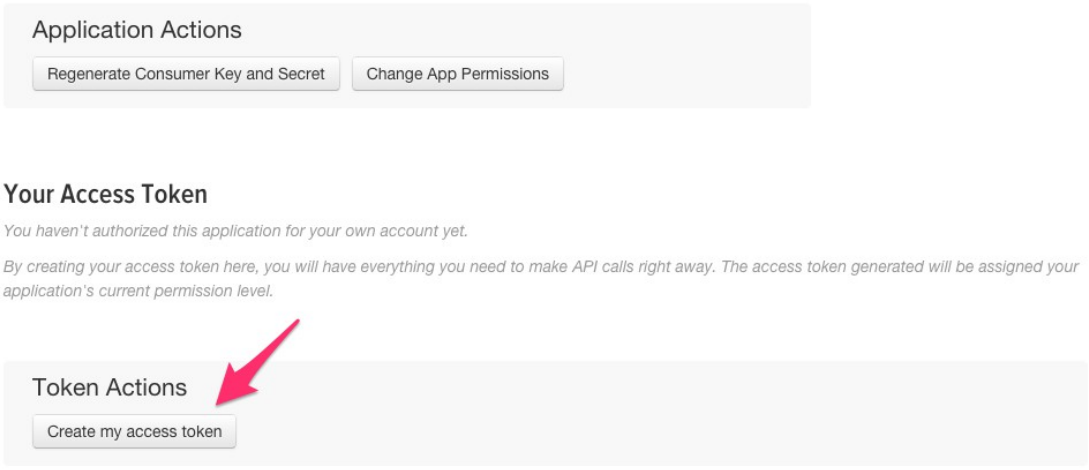
Create a new Twitter App (cont.)

Click on Keys and Access Tokens:



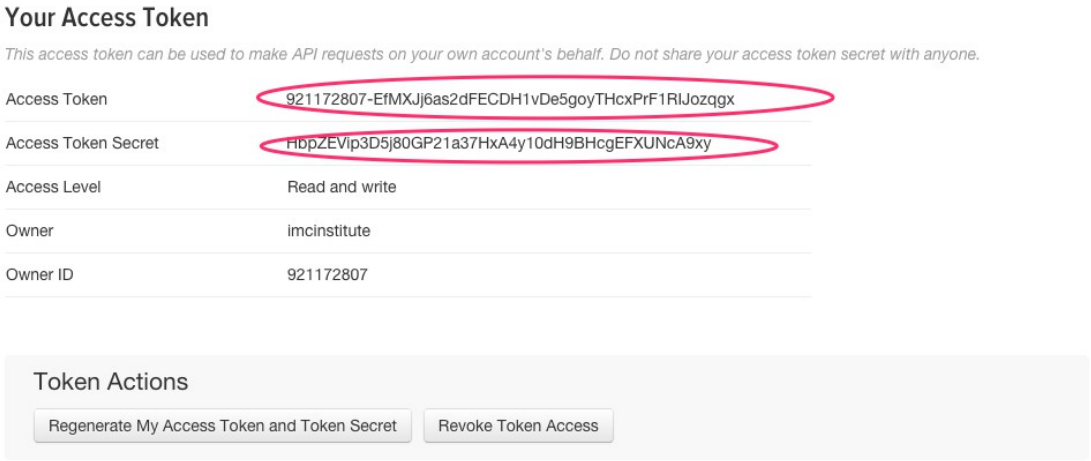
Create a new Twitter App (cont.)

Click on Keys and Access Tokens:



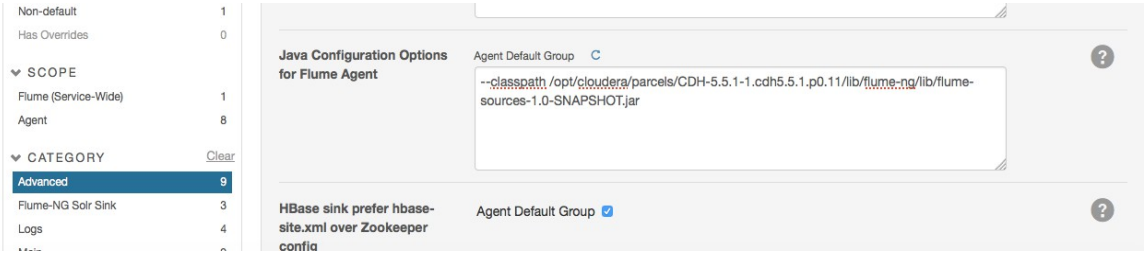
Create a new Twitter App (cont.)

Your Access token got created:

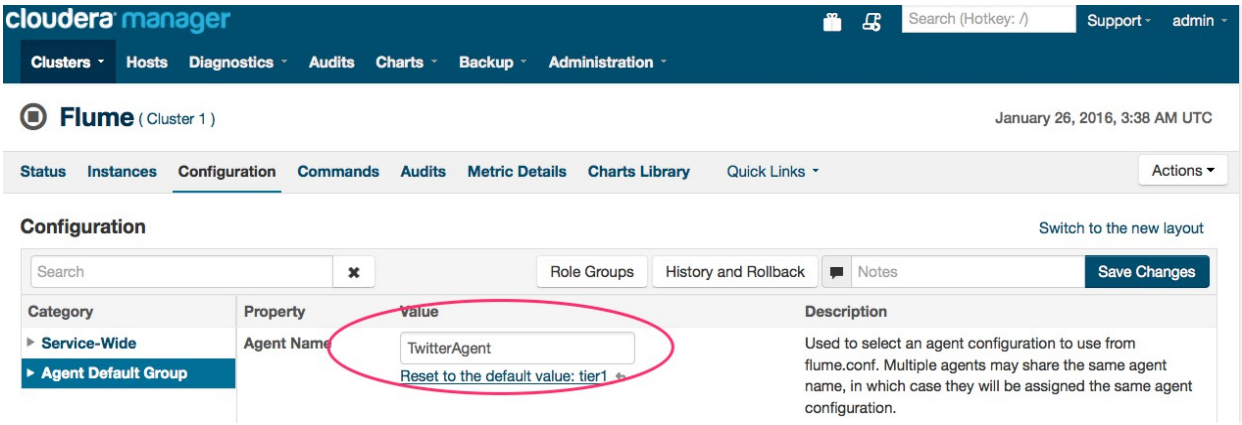


Add classpath in Cloudera Manager

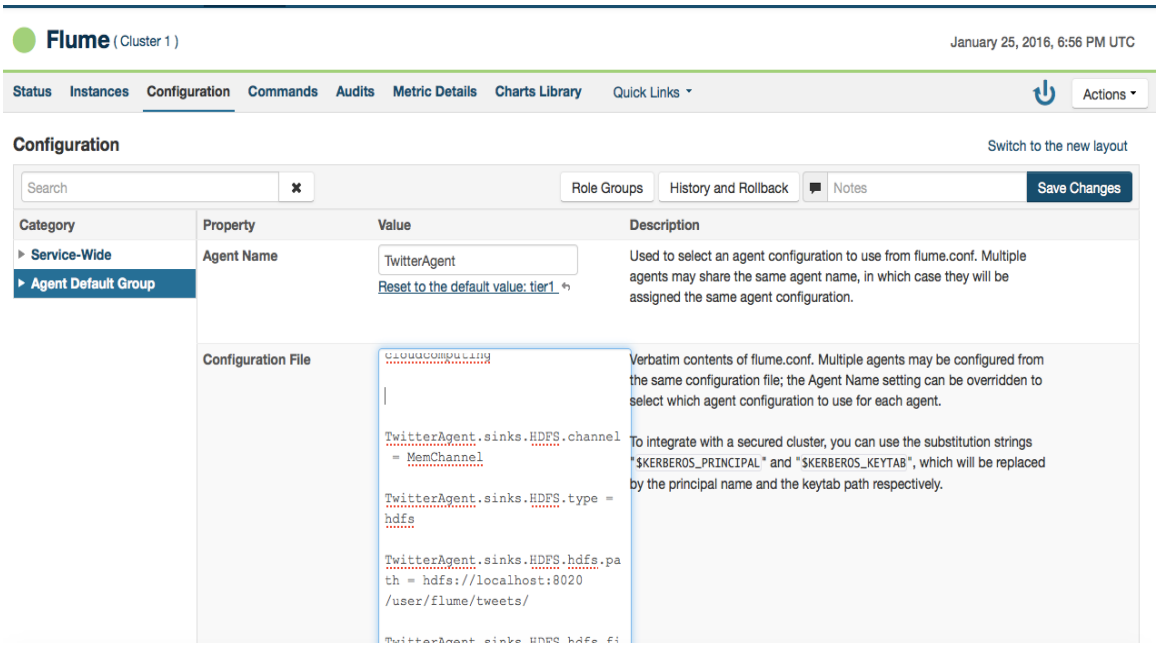
```
"Services" -> "flume1" -> "Configuration" -> -> "Advanced"
-> "Java Configuration Options for Flume Agent", add:


--classpath /opt/cloudera/parcels/CDH-5.5.1-
1.cdh5.5.1.p0.11/lib/flume-ng/lib/flume-sources-1.0-
SNAPSHOT.jar
```



Change the Flume Agent Name



Configuring the Flume Agent



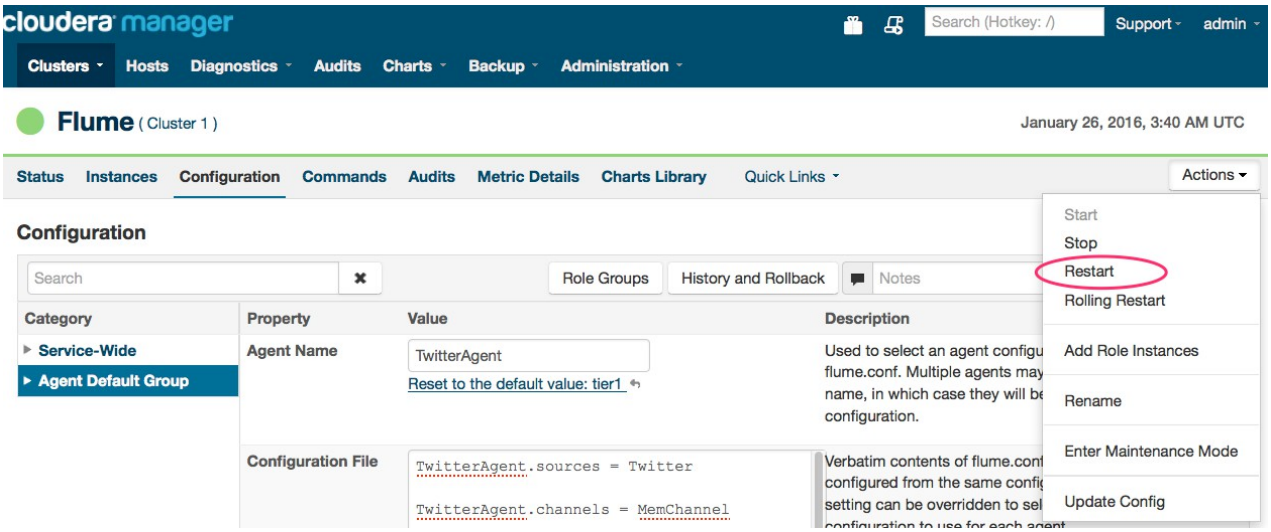
Agent Configuration

```
TwitterAgent.sources = Twitter
TwitterAgent.channels = MemChannel
TwitterAgent.sinks = HDFS

TwitterAgent.sources.Twitter.type =
org.apache.flume.source.twitter.TwitterSource
TwitterAgent.sources.Twitter.channels = MemChannel
TwitterAgent.sources.Twitter.consumerKey =
MjpswndxVj27ylnpOoSBrnfLX
TwitterAgent.sources.Twitter.consumerSecret =
QYmuBO1smD5Yc3zE0ZF9ByCgeEQxnxUmhRVCisAvPFudYVjC4a
TwitterAgent.sources.Twitter.accessToken = 921172807-
EfMXJj6as2dFECDH1vDe5goyTHcxPrF1RIJozqgx
TwitterAgent.sources.Twitter.accessTokenSecret =
HbpZEVip3D5j80GP21a37HxA4y10dH9BHcgEFXUNcA9xy
```

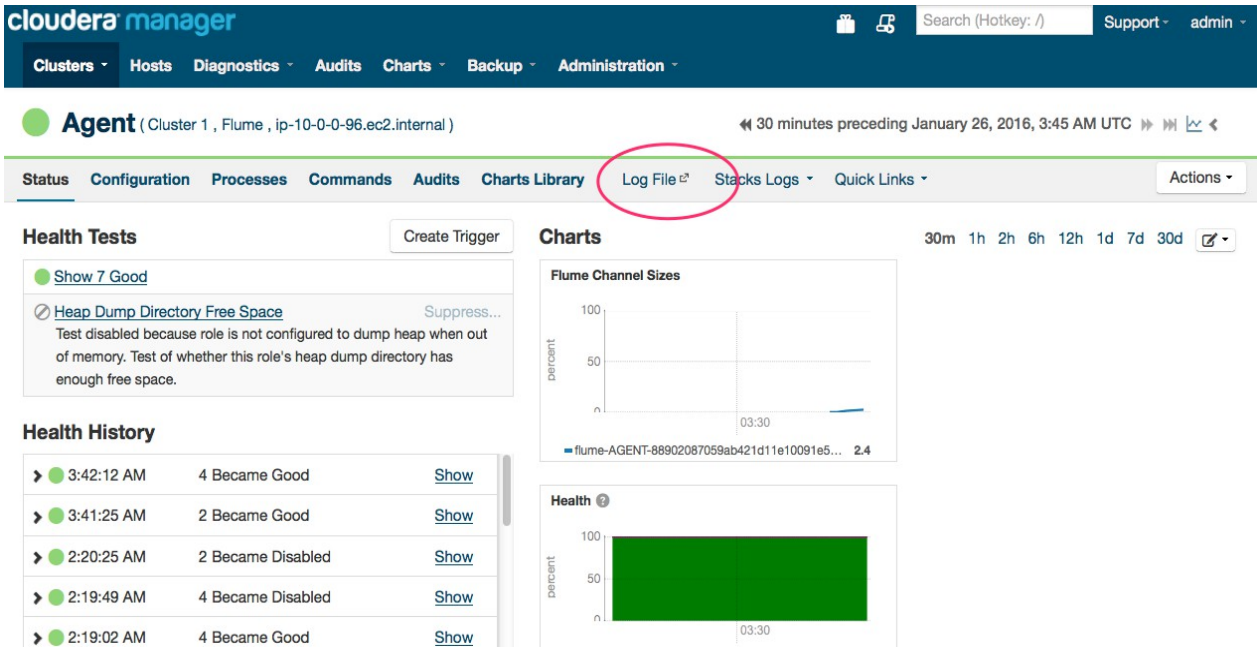
Agent Configuration

```
TwitterAgent.sources.Twitter.keywords = hadoop, big data,
analytics, bigdata, cloudera, data science, data scientiest,
business intelligence, mapreduce, data warehouse, data
warehousing, mahout, hbase, nosql, newsql,
businessintelligence, cloudcomputing
TwitterAgent.sinks.HDFS.channel = MemChannel
TwitterAgent.sinks.HDFS.type = hdfs
TwitterAgent.sinks.HDFS.hdfs.path =
hdfs://xx.xx.xx.xx:8020/user/flume/tweets/
TwitterAgent.sinks.HDFS.hdfs.fileType = DataStream
TwitterAgent.sinks.HDFS.hdfs.writeFormat = Text
TwitterAgent.sinks.HDFS.hdfs.batchSize = 1000
TwitterAgent.sinks.HDFS.hdfs.rollSize = 0
TwitterAgent.sinks.HDFS.hdfs.rollCount = 10000
TwitterAgent.channels.MemChannel.type = memory
TwitterAgent.channels.MemChannel.capacity = 10000
TwitterAgent.channels.MemChannel.transactionCapacity = 100
```

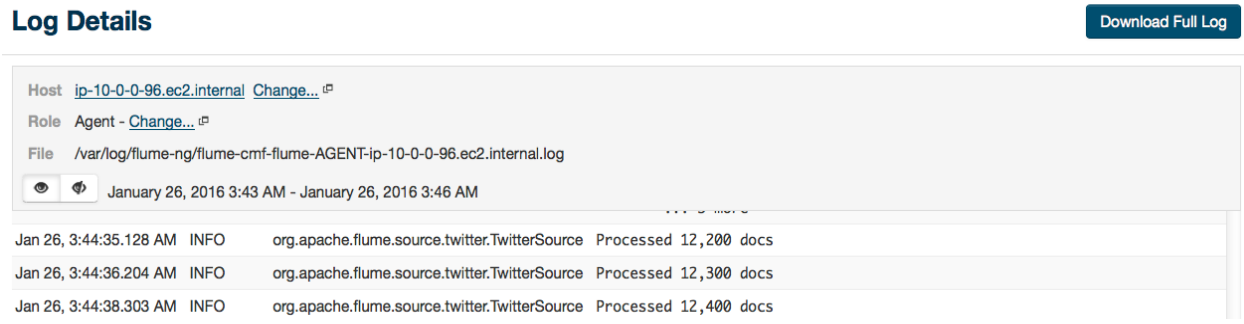

Restart Flume



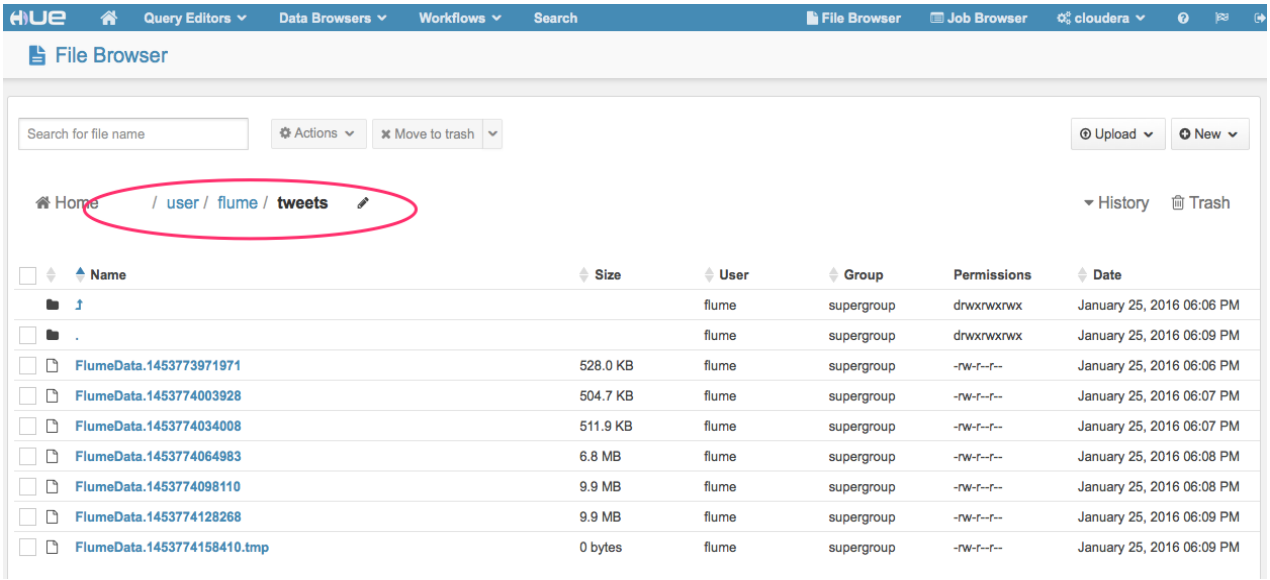
View an agent log file



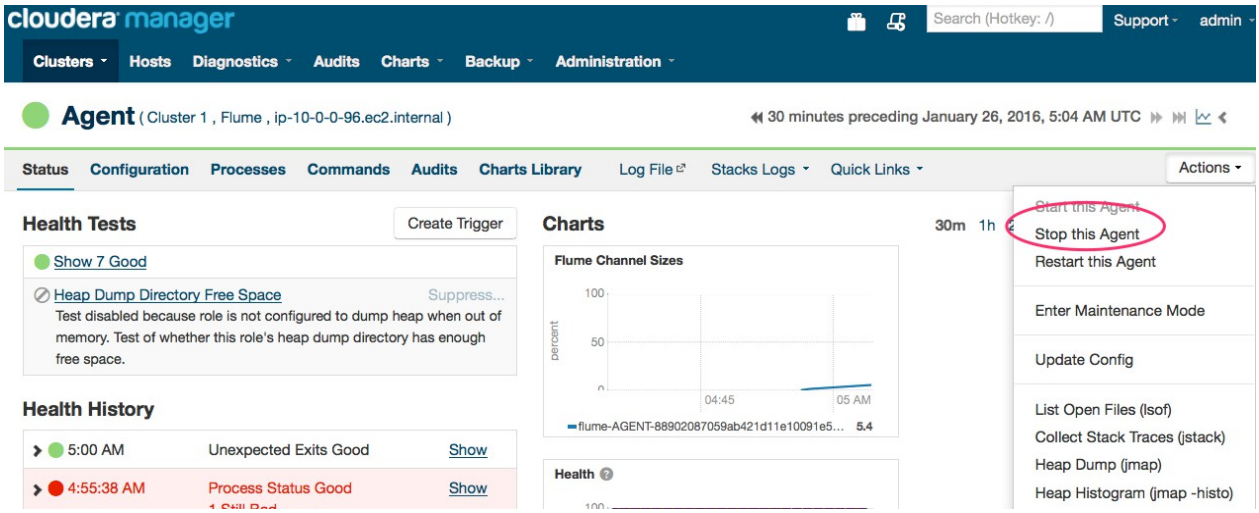
View an agent log file



View a result using Hue



Stop the agent



7. Analyse data using Hive

Get a Serde Jar File for parsing JSON file

```
$ wget
http://files.cloudera.com/samples/hive-serdes-1.0-SNAPSHOT.jar

$ mv hive-serdes-1.0-SNAPSHOT.jar  /usr/local/apache-hive-
1.1.0-bin/lib/

$ hive
```

Register the Jar file.

```
hive> ADD JAR /usr/local/apache-hive-1.1.0-bin/lib/hive-
serdes-1.0-SNAPSHOT.jar;
```


¸Analyse data using Hive (cont.)

Running the following hive command

```
CREATE EXTERNAL TABLE tweets (
   id BIGINT,
   created_at STRING,
   source STRING,
   favorited BOOLEAN,
   retweet_count INT,
   retweeted_status STRUCT<
      text:STRING,
      user:STRUCT<screen_name:STRING,name:STRING>>,
   entities STRUCT<
      urls:ARRAY<STRUCT<expanded_url:STRING>>,
      user_mentions:ARRAY<STRUCT<screen_name:STRING,name:STRING>>,
      hashtags:ARRAY<STRUCT<text:STRING>>>,
   text STRING,
   user STRUCT<
      screen_name:STRING,
      name:STRING,
      friends_count:INT,
      followers_count:INT,
      statuses_count:INT,
      verified:BOOLEAN,
      utc_offset:INT,
      time_zone:STRING>,
   in_reply_to_screen_name STRING
)
ROW FORMAT SERDE 'com.cloudera.hive.serde.JSONSerDe'
LOCATION '/user/flume/tweets';
```

http://www.thecloudavenue.com/2013/03/analyse-tweets-using-flume-hadoop-and.html

Analyse data using Hive (cont)

Finding user who has the most number of followers

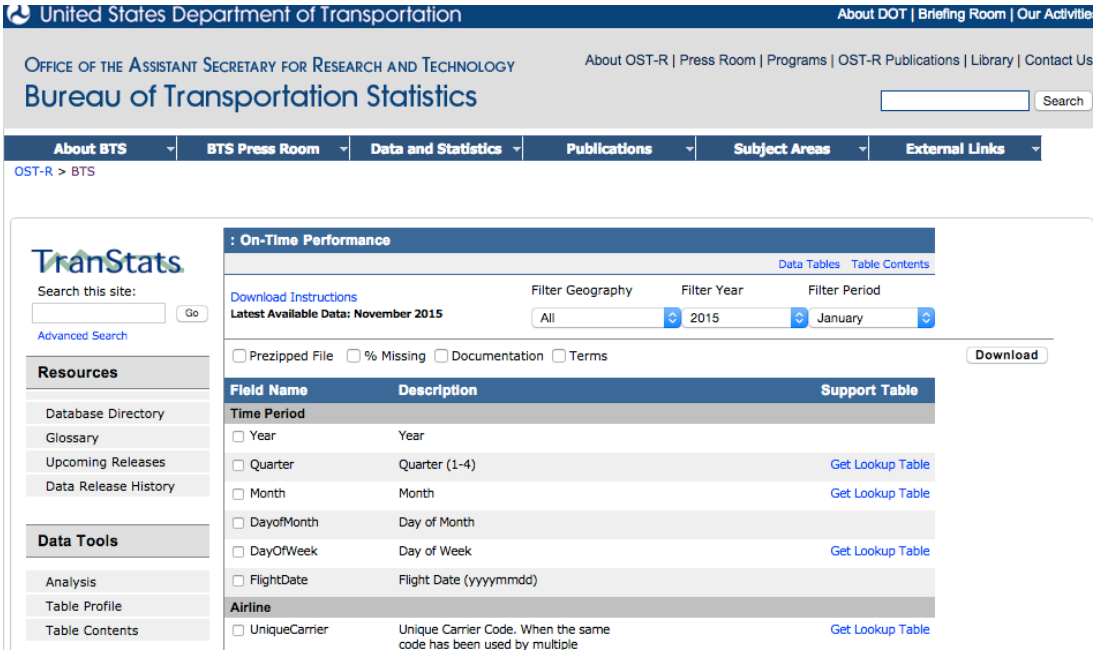
```
hive> select user.screen_name, user.followers_count c from
tweets order by c desc;
```

```
Starting Job = job_201504051617_0010, Tracking URL = http://localhost:50030/jobdetails.jsp?jobid=job_201504051617_0010
Kill Command = /usr/local/hadoop/libexec/../bin/hadoop job  -kill job_201504051617_0010
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2015-04-06 15:37:27,782 Stage-1 map = 0%,  reduce = 0%
2015-04-06 15:37:31,837 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 1.27 sec
2015-04-06 15:37:39,899 Stage-1 map = 100%,  reduce = 33%, Cumulative CPU 1.27 sec
2015-04-06 15:37:40,908 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 2.42 sec
MapReduce Total cumulative CPU time: 2 seconds 420 msec
Ended Job = job_201504051617_0010
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 2.42 sec   HDFS Read: 170686 HDFS Write: 687 SUCCESS
Total MapReduce CPU Time Spent: 2 seconds 420 msec
OK
vinnaum 11523
navchatterji    5485
HCITExpert      4751
NWDCScoop       4097
7wdata  3005
MotivasiMariaP  2007
WesleyBackelant 1977
IFTTTMarketing  1307
jonathangibs    968
ephraimcohen    914
feshob  716
DKajouri        713
```


Project: Flight

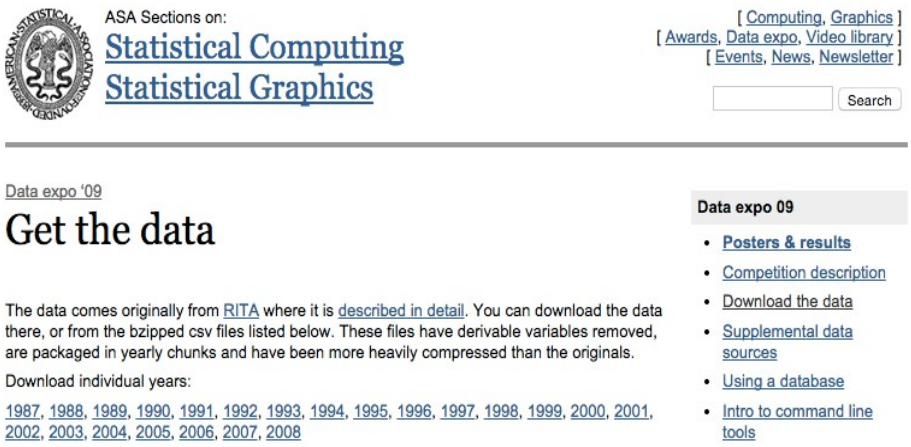
Flight Details Data

http://www.transtats.bts.gov/DL_SelectFields.asp?Table_ID=236



Flight Details Data

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



Data Description

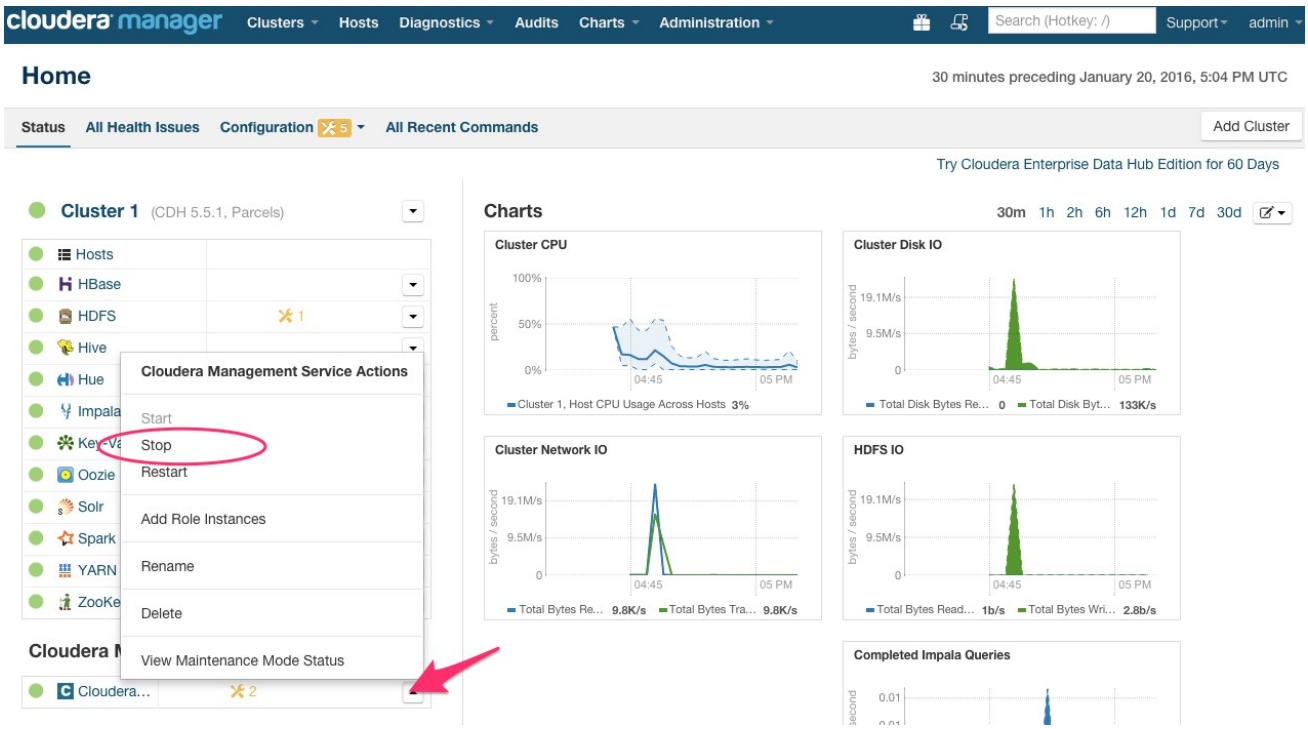
	Name	Description
1	Year	1987-2008
2	Month	1-12
3	DayofMonth	1-31
4	DayOfWeek	1 (Monday) - 7 (Sunday)
5	DepTime	actual departure time (local, hhmm)
6	CRSDepTime	scheduled departure time (local, hhmm)
7	ArrTime	actual arrival time (local, hhmm)
8	CRSArrTime	scheduled arrival time (local, hhmm)
9	UniqueCarrier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

Snapshot of Dataset

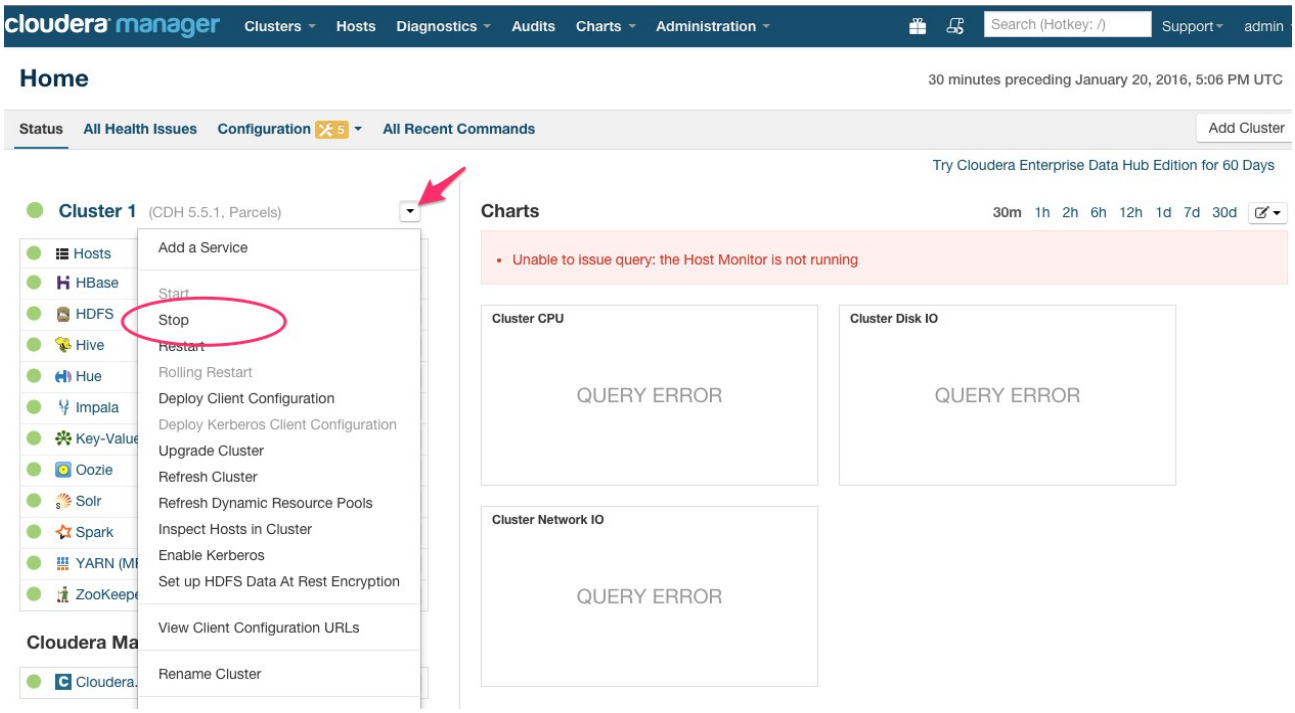
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1	Year	Month	DayofMor	DayOfWe	DepTime	CRSDepTi	ArrTime	CRSArrTin	UniqueCa	FlightNun	TailNum	ActualElaj	CRSElapse	AirTime	ArrDelay	DepDelay	Origin	Dest	Distance	TaxiIn	TaxiO
2	2008	1	5	6	2243	1415	45	1625	WN	1684	N347SW	62	70	41	500	508	SAN	PHX	304	2	
3	2008	1	5	6	1940	1220	2111	1350	WN	1684	N347SW	91	90	64	441	440	SFO	SAN	447	5	
4	2008	1	7	1	111	1845	308	2045	WN	405	N644SW	117	120	103	383	386	MDW	JAN	666	4	
5	2008	1	7	1	2213	1700	2317	1655	WN	1827	N759GS	124	55	75	382	313	IND	MDW	162	10	
6	2008	1	7	1	2143	1720	26	1820	WN	1430	N644SW	163	60	83	366	263	STL	MDW	251	24	
7	2008	1	7	1	117	2020	302	2135	WN	490	N651SW	105	75	87	327	297	STL	TUL	351	5	
8	2008	1	7	1	2358	1855	105	2000	WN	490	N651SW	67	65	50	305	303	MDW	STL	251	4	
9	2008	1	3	4	2245	1730	2354	1850	WN	186	N792SW	69	80	59	304	315	JAN	HOU	359	3	
10	2008	1	7	1	2219	1730	35	1935	WN	2474	N710SW	76	65	67	300	289	MDW	CMH	284	2	
11	2008	1	5	6	2129	1620	2246	1750	WN	1924	N408WN	77	90	56	296	309	SFO	LAS	414	4	
12	2008	1	3	4	1615	1130	1623	1135	WN	10	N617SW	68	65	56	288	285	MAF	ABQ	332	4	
13	2008	1	3	4	1736	1305	2031	1555	WN	1837	N761RR	295	290	268	276	271	MDW	SFO	1855	4	
14	2008	1	5	6	2236	1805	2400	1930	WN	646	N283WN	84	85	71	270	271	LAX	SFO	337	6	
15	2008	1	3	4	2021	1700	2303	1835	WN	2005	N302SW	162	95	73	268	201	LAS	SFO	414	4	
16	2008	1	3	4	2059	1620	2216	1750	WN	1924	N761RR	77	90	60	266	279	SFO	LAS	414	6	
17	2008	1	7	1	2348	2105	307	2250	WN	3137	N358SW	259	165	244	257	163	MCO	MDW	989	1	
18	2008	1	3	4	2255	1820	509	55	WN	1924	N761RR	194	215	176	254	275	LAS	IND	1591	9	
19	2008	1	9	3	1458	1040	1725	1315	WN	2556	N501SW	87	95	76	250	258	BNA	BWI	588	4	
20	2008	1	7	1	2300	1835	113	2105	WN	2804	N420WN	253	270	240	248	265	MDW	PDX	1751	5	
21	2008	1	5	6	47	2040	151	2145	WN	505	N435WN	64	65	51	246	247	BWI	PVD	328	5	
22	2008	1	5	6	1558	1225	14	2010	WN	505	N442WN	316	285	250	244	213	SAN	BWI	2295	5	
23	2008	1	5	6	1931	1540	2104	1705	WN	1179	N718SW	93	85	77	239	231	SAN	OAK	446	7	
24	2008	1	4	5	1822	1425	2003	1605	WN	753	N726SW	101	100	88	238	237	PDX	OAK	543	6	

Shut down Cluster

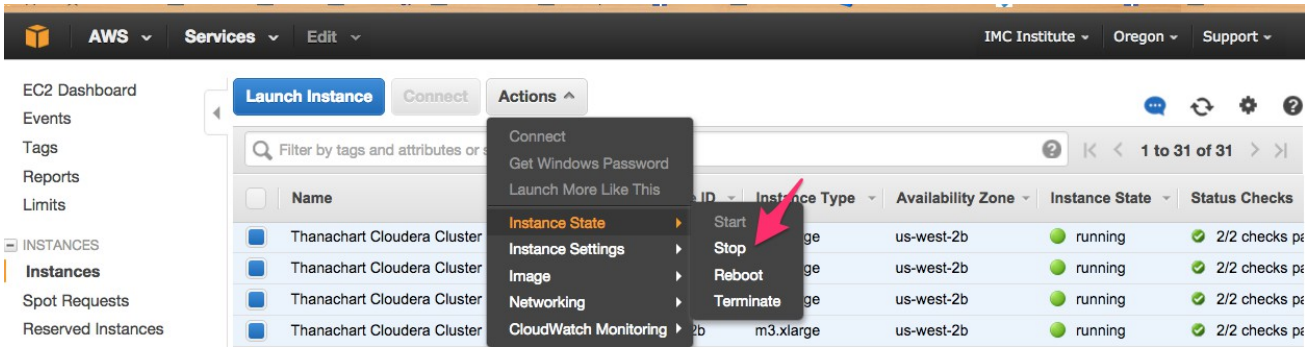
Finish: Stop All Services



Finish: Stop Cluster



Finish: Stop EC2 instances



Shell Script for Complete stopping Hadoop's all services

```
#! /usr/bin/env bash
/etc/init.d/zookeeper-server stop
/etc/init.d/hadoop-hdfs-datanode stop
/etc/init.d/hadoop-hdfs-journalnode stop
/etc/init.d/hadoop-hdfs-namenode stop
/etc/init.d/hadoop-hdfs-secondarynamenode stop
/etc/init.d/hadoop-httpfs stop
/etc/init.d/hadoop-mapreduce-historyserver stop
/etc/init.d/hadoop-yarn-nodemanager stop
/etc/init.d/hadoop-yarn-resourcemanager stop
/etc/init.d/hbase-master stop
/etc/init.d/hbase-rest stop
/etc/init.d/hbase-thrift stop
/etc/init.d/hive-metastore stop
/etc/init.d/hive-server2 stop
/etc/init.d/sqoop2-server stop
/etc/init.d/spark-history-server stop
/etc/init.d/hbase-regionserver stop
/etc/init.d/hue stop
/etc/init.d/impala-state-store stop
/etc/init.d/oozie stop
/etc/init.d/solr-server stop
/etc/init.d/impala-catalog stop
/etc/init.d/impala-server stop
```

Thank you

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