



Cloudera Administrator Training for Apache Hadoop: Hands-On Exercises

General Notes	3
Setup Activity: Configuring Networking.....	7
Hands-On Exercise: Installing Cloudera Manager Server	13
Hands-on Exercise: Creating a Hadoop Cluster	20
Hands-On Exercise: Working With HDFS.....	33
Hands-On Exercise: Running YARN Applications.....	38
Hands-On Exercise: Explore Hadoop Configurations and Daemon Logs	49
Hands-On Exercise: Using Flume to Put Data into HDFS.....	55
Hands-On Exercise: Importing Data with Sqoop	63
Hands-On Exercise: Querying HDFS With Hive and Cloudera Impala.....	68
Hands-On Exercise: Using Hue to Control Hadoop User Access	78
Hands-On Exercise: Configuring HDFS High Availability.....	91
Hands-On Exercise: Using the Fair Scheduler	101
Hands-On Exercise: Breaking The Cluster.....	107
Hands-On Exercise: Verifying The Cluster's Self-Healing Features	109
Hands-On Exercise: Taking HDFS Snapshots.....	112
Hands-On Exercise: Configuring Email Alerts	114
Troubleshooting Challenge: Heap O' Trouble	116

Appendix A: Setting up VMware Fusion on a Mac for the Cloud Training Environment.....	120
Appendix B: Setting up VirtualBox for the Cloud Training Environment.....	124

General Notes

Training Environment Overview

In this training course, you will use a cloud environment consisting of five Amazon EC2 instances. You will also use a local virtual machine (VM) to access the cloud environment.

All of the EC2 instances use the CentOS 6.4 Linux distribution.

Use the **training** user account to do your work. You should not need to enter passwords for the **training** user.

Should you need superuser access, you can use `sudo` as the **training** user without entering a password. The **training** user has unlimited, passwordless `sudo` privileges.

For the training environment:

- You will start the VM, and then use it to connect to your EC2 instances
- The EC2 instances have been configured so that you can connect to them without entering a password
- Your instructor will provide the following details for five EC2 instances per student:
 - EC2 *public* IP addresses – You will run a script that adds the EC2 public IP addresses of your five EC2 instances to the `/etc/hosts` file on your VM
 - EC2 *private* IP addresses – You will use these addresses when you run a script that configures the `/etc/hosts` file on your EC2 instances. EC2 private IP addresses start with the number 10, 172, or 192.168

- The instructor will not provide EC2 internal host names. Please use the host names elephant, tiger, horse, monkey and lion for the five internal host names. **It is important that you use these five host names**, because several scripts, which expect these five host names, have been provided for your use while you perform exercises. The scripts will not work if you use different host names.
- Please write out a table similar to the following with your EC2 instance information:

Host Name	EC2 Public IP Address	EC2 Private IP Address
elephant		
tiger		
horse		
monkey		
lion		

Notational Convention

In some command-line steps in the exercises, you will see lines like this:

```
$ hdfs dfs -put shakespeare \
/user/training/shakespeare
```

The backslash at the end of the first line signifies that the command is not completed, and continues on the next line. You can enter the code exactly as shown (on two lines), or you can enter it on a single line. If you do the latter, you should *not* type in the backslash.

Copying and Pasting from the Hands-On Exercises Manual

If you wish, you can usually copy commands and strings from this Hands-On Exercises manual and paste them into your terminal sessions. However, please note one important exception:

If you copy and paste table entries that exceed a single line, a space may be inserted at the end of each line.

Dash (-) characters are especially problematic and do not always copy correctly. Be sure to delete any pasted dash characters and key them in manually after pasting in text that contains them.

Please use caution when copying and pasting when performing the exercises.

Resetting Your Cluster

You can use the `reset_cluster.sh` script to change the state of your cluster so that you can start with a fresh, correct environment for performing any exercise. Use the script in situations such as the following:

- While attempting one of the exercises, you misconfigure your machines so badly that attempting to do the next exercise is no longer possible.
- You have successfully completed a number of exercises, but then you receive an emergency call from work and you have to miss some time in class. When you return, you realize that you have missed two or three exercises. But you want to do the same exercise everyone else is doing so that you can benefit from the post-exercise discussions.

The script is destructive: any work that you have done is overwritten when the script runs. If you want to save files before running the script, you can copy the files you want to save to a subdirectory under `/home/training`.

Before you attempt to run the script, verify that networking among the five hosts in your cluster is working. If networking has not been configured correctly, you can

rerun the `CM_config_hosts.sh` script to reset the networking configuration prior to running the `reset_cluster.sh` script.

Run the script on `elephant` only. You do not need to change to a directory to run the script; it is in your shell's PATH.

The script starts by prompting you to enter the number of an exercise. Specify the exercise you want to perform *after* the script has run. Then confirm that you want to reset the cluster (thus overwriting any work you have done).

The script will further prompt you to specify if you want to run only the steps that simulate the previous exercise, or if you want to completely uninstall and reinstall the cluster and then catch you up to the specified exercise. Note that choosing to only complete the previous exercise does not offer as strong an assurance of properly configuring your cluster as a full reset would do. It is however a more expedient option.

After you have responded to the initial prompts, the script begins by cleaning up your cluster—terminating Hadoop processes, removing Hadoop software, deleting Hadoop-related files, and reverting other changes you might have made to the hosts in your cluster. Please note that as this system cleanup phase is running, you will see errors such as “unrecognized service” and “No packages marked for removal.” These errors are expected. They occur because the script attempts to remove *anything* pertinent that might be on your cluster. The number of error messages that appear during this phase of script execution depends on the state the cluster is in when you start the script.

Next, the script simulates steps for each exercise up to the one you want to perform.

Script completion time varies from 5 minutes to almost an hour depending on how many exercise steps need to be simulated by the script.

Setup Activity: Configuring Networking

In this preparatory exercise you will configure networking for your five instances.

Task Overview

In this task, you will run scripts to configure networking.

First, you will start the local Get2EC2 VM and run a script to configure the /etc/hosts file on that VM with the addresses of the five EC2 instances and the host names elephant, tiger, horse, monkey and lion.

Next, you will run a script to configure the /etc/hosts file on the EC2 instances, setting the five host names to elephant, tiger, horse, monkey and lion.

Then you will verify that networking has been set up correctly.

Finally, you will run a script that starts a SOCKS5 proxy server on the local VM.

Steps to Complete

1. If you are using VMware Fusion on Mac OS to connect to your five EC2 instances, read 'Appendix A, Setting up VMware Fusion on a Mac for the Cloud Training Environment' and perform any indicated actions. After you have done so, continue to the next step.

2. Start the Get2EC2 VM.

You should find the VM on your Desktop, on the C: drive of your machine, or in the location to which you downloaded it. Double-click the icon whose name ends in .vmx to launch the VM.

After a few minutes, the GNOME interface will appear.

Note: A VirtualBox VM is available for students running Mac OS who are unable to or prefer not to install VMware Fusion. However, we strongly recommend using the VMware version VM. Use VirtualBox for this course *only* if it is your preferred virtualization environment and if you are knowledgeable enough to be self-sufficient to troubleshoot problems you might run into.

If you are using a VirtualBox VM, follow the steps in ‘Appendix B, Setting up VirtualBox for the Cloud Training Environment.’ When you have completed the steps in the Appendix, continue to the next step.

3. Run a script that modifies the /etc/hosts file on your VM by adding your five EC2 instances.

Enter the following command in the terminal window in the VM:

```
$ CM_config_local_hosts_file.sh
```

The CM_config_local_hosts_file.sh script will prompt you to enter the five EC2 public IP addresses for your five EC2 instances. Refer to the table you created when your instructor gave you the EC2 instance information.

4. Log in to the elephant EC2 instance as the training user.

```
$ connect_to_elephant.sh
```

When prompted to confirm that you want to continue connecting, enter yes and then press Enter.

5. Run the `CM_config_hosts.sh` script on `elephant`. This script sets up the `/etc/hosts` file on `elephant`, copies that file to `tiger`, `horse`, `monkey` and `lion`, and then sets the host names for the five hosts to `elephant`, `tiger`, `horse`, `monkey` and `lion`.

Enter the following command on `elephant`:

```
$ CM_config_hosts.sh
```

When the script prompts you to enter IP addresses for your EC2 instances, enter the EC2 private IP addresses (which typically start with 10, 172, or 192.168).

When the script prompts you to confirm that you want to continue connecting, enter `yes` each time and then press Enter.

6. Terminate and restart the SSH session with `elephant`.

```
$ exit
$ connect_to_elephant.sh
```

Note: You exit and reconnect to `elephant` to reset the value of the `$HOSTNAME` variable in the shell after the `CM_config_hosts.sh` script changes the host name.

7. Start SSH sessions with your other four EC2 instances, logging in as the `training user`.

Open four more terminal windows (or tabs) on your VM, so that a total of five terminal windows are open.

In one of the new terminal windows, connect to the `tiger` EC2 instance.

```
$ connect_to_tiger.sh
```

When the script prompts you to confirm that you want to continue connecting, enter `yes` and then press Enter. (You will also need to do this when you connect to `horse` and `monkey`.)

In another terminal window, connect to the horse EC2 instance.

```
$ connect_to_horse.sh
```

In another terminal window, connect to the monkey EC2 instance.

```
$ connect_to_monkey.sh
```

In the remaining terminal window, connect to the lion EC2 instance.

```
$ connect_to_lion.sh
```

8. Verify that you can communicate with all the hosts in your cluster from elephant by using the host names.

On **elephant**:

```
$ ping elephant
$ ping tiger
$ ping horse
$ ping monkey
$ ping lion
```

9. Verify that passwordless SSH works by running the ip addr command on tiger, horse, monkey and lion from a session on elephant.

On **elephant**:

```
$ ssh training@tiger ip addr
$ ssh training@horse ip addr
$ ssh training@monkey ip addr
$ ssh training@lion ip addr
```

The ip addr command shows you IP addresses and properties.

Note: Your environment is set up to allow you to use passwordless SSH to submit commands from a session on elephant to the root and training users on tiger, horse, and monkey, and to allow you to use the scp command to copy files from elephant to the other four hosts. **Passwordless SSH and scp are not required to deploy a Hadoop cluster.** We make these tools available in the classroom environment as a convenience.

10. Run the `uname -n` command to verify that all five hostnames have been changed as expected.

On all five hosts:

```
$ uname -n
```

11. Verify that the value of the `$HOSTNAME` environment variable has been set to the new host name.

On all five hosts:

```
$ echo $HOSTNAME
```

The value elephant, tiger, horse, monkey or lion should appear.

12. Start a SOCKS5 proxy server on the VM. The browser on your VM will use the proxy to communicate with your EC2 instances.

Open one more terminal window on the local VM – *not* a tab in an existing window – and enter the following command:

```
$ start_SOCKS5_proxy.sh
```

When the script prompts you to confirm that you want to continue connecting, enter yes and then press Enter.

13. Minimize the terminal window in which you started the SOCKS5 proxy server.

Important: Do not close this terminal window or exit the SSH session running in it while you are working on exercises. If you accidentally terminate the proxy server, restart it by opening a new terminal window and rerunning the `start_SOCKS5_proxy.sh` script.

At the End of Each Day

At the end of the day, exit all active SSH sessions you have with EC2 instances (including the session running the SOCKS5 proxy server). To exit an SSH session, simply execute the `exit` command.

CAUTION: Do *not* shut down your EC2 instances.

Then suspend the Get2EC2 VM.

When you return in the morning, resume the Get2EC2 VM, reestablish SSH sessions with the EC2 instances, and restart the SOCKS5 proxy server.

To reestablish SSH sessions with the EC2 instances, open a terminal window (or tab), and then execute the appropriate `connect_to` script.

To restart the SOCKS5 proxy server, open a separate terminal window, and then execute the `start_SOCKS5_proxy.sh` script.

**This is the end of the setup activity
for the training environment.**

Hands-On Exercise: Installing Cloudera Manager Server

With Cloudera Manager, you can install your Hadoop cluster using one of two installation options, referred to as *Installation Path A* and *Installation Path B* in the Cloudera Manager documentation. Installation Path A is driven entirely by GUI-based wizards and is appropriate for a demo or proof of concept installation. Installation Path B is command-line driven and is appropriate for production deployments.

For this installation, you will use Installation Path B to install Cloudera Manager Server on the `lion` machine. Before installing Cloudera Manager you will configure an external database (MySQL) to be used by Cloudera Manager and CDH, which you will install in the next exercise.

Complete all steps in this exercise on `lion`.

Verify Existing Environment

1. Verify the Oracle JDK is installed and that `JAVA_HOME` is defined and referenced in the system PATH.

On `lion`:

```
$ java -version  
$ echo $JAVA_HOME  
$ env | grep PATH
```

2. Verify Python is installed. It is a requirement for Hue, which you will install later in the course.

On `lion`:

```
$ sudo rpm -qa | grep python-2.6
```

3. Verify Oracle MySQL Server is installed and running on **lion**.

On **lion**:

```
$ chkconfig | grep mysqld
$ sudo service mysqld status
```

Note that in a true production deployment you would also modify the /etc/my.cnf MySQL configurations and move the InnoDB log files to a backup location.

4. Verify the MySQL JDBC Connector is installed. Sqoop (a part of CDH that you will install in this course) does not ship with a JDBC connector, but does require one.

On **lion**:

```
$ ls -l /usr/share/java
```

Configure the External Database

In keeping with the approach to installation that is appropriate for a production cluster environment, you will use an external database rather than the embedded PostgreSQL database option.

1. Create the required databases in MySQL.

On **lion**, open ~/training_materials/admin/scripts/**mysql-setup.sql** in a text editor and observe the script details. This script creates databases for Cloudera Manager, the Hive Metastore, the Activity Monitor, and the Reports Manager.

Note that if you were going to add the Sentry Server CDH service or install Cloudera Navigator, additional databases would also need to be created.

On **lion**:

```
$ mysql -u root < \
~/training_materials/admin/scripts/mysql-setup.sql
$ mysql -u root
```

Now, at the mysql> prompt on **lion**, issue the following commands:

```
show databases;
exit;
```

The ‘show databases’ command should show that the four databases were created.

Note: in a true production deployment you would also regularly backup your database.

2. Make your MySQL installation secure and set the root user password.

On **lion**:

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

3. Verify the Cloudera Manager local software repository.

Your instances contain a local yum repository of Cloudera Manager 5 software to save download time in this course.

CentOS (and Red Hat) store software repository references in `/etc/yum.repos.d`. Issue the command below to view the settings.

On **lion**:

```
$ cat /etc/yum.repos.d/cloudera-cm5.repo
```

View the software packages in a subfolder of the referenced directory.

On **lion**:

```
ls ~/software/cloudera-cm5/RPMS/x86_64
```

Note that these two locations exist on each of the five machines in your environment and have also been made available on HTTP ports 8050 and 8000 respectively via a Linux service. This setup is specific to the course environment and is not required for Cloudera Manager or CDH installations.

If you wanted to install from the online repository, you would create a reference to the Cloudera repository in your /etc/yum.repos.d directory.

Install Cloudera Manager Server

1. Install Cloudera Manager Server.

On **lion**:

```
$ sudo yum install -y cloudera-manager-daemons \
cloudera-manager-server
```

Note: The **-y** option provides an answer of **yes** in response to an expected confirmation prompt.

2. Set the Cloudera Manager Server service to not start on boot.

On **lion**:

```
$ sudo chkconfig cloudera-scm-server off
```

3. Run the script to prepare the Cloudera Manager database.

On **lion**:

```
$ sudo /usr/share/cmf/schema/scm_prepare_database.sh \
mysql cmserver cmserveruser password
```

After running the command above you should see the message, “All done, your SCM database is configured correctly!”

4. Verify the local software repositories.

Run the following two commands to verify the URLs are working.

On **lion**:

```
$ curl lion:8000  
$ curl lion:8050
```

Each command should show hyperlinks (code) to software repositories. If either command did not successfully contact the web server, discuss with the instructor before continuing.

5. Start the Cloudera Manager Server.

On **lion**:

```
$ sudo service cloudera-scm-server start
```

6. Review the processes started by the Cloudera Manager Server.

On **lion**:

```
$ top
```

Cloudera Manager Server runs as a java process that you can view by using the `top` Linux utility. Notice the CPU usage is in the 90th percentile or above for a few seconds while the server starts. Once the CPU usage drops, the Cloudera Manager browser interface will become available.

On **lion**:

```
$ ps wax | grep [c]loudera-scm-server
```

The results of the `ps` command above show that Cloudera Manager Server is using the JDBC MySQL connector to connect to MySQL. It also shows logging configuration and other details.

7. Review the Cloudera Manager Server log file.

The path to the log file is `/var/log/cloudera-scm-server/cloudera-scm-server.log`. Note that you must use `sudo` to access Cloudera Manager logs because of restricted permissions on the Cloudera Manager log file directories.

On `lion`:

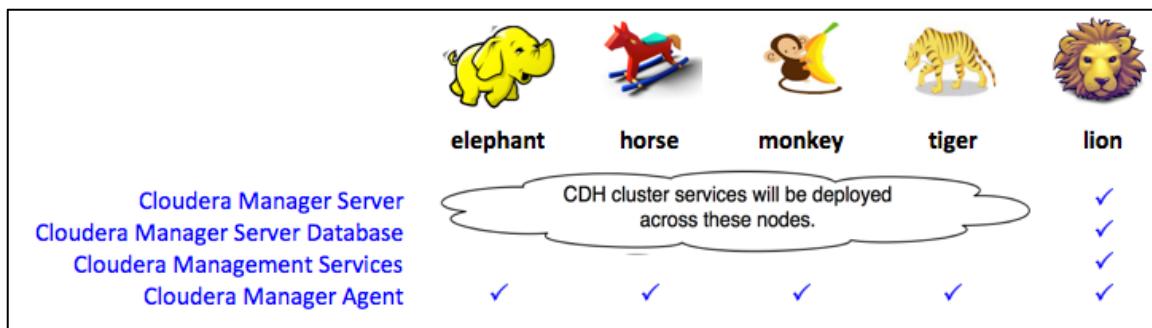
```
$ sudo less /var/log/cloudera-scm-server/cloudera-scm-\\  
server.log
```

Note: You will login to Cloudera Manager in the next exercise.

This is the end of the exercise.

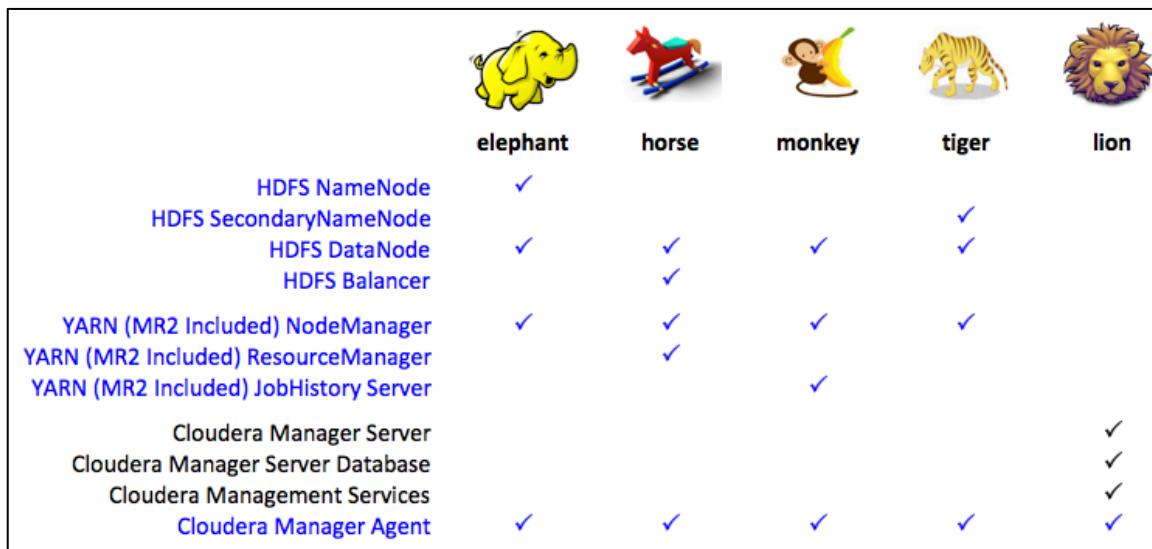
Hands-on Exercise: Creating a Hadoop Cluster

In this task, you will log in to the Cloudera Manager Admin Console and use the Cloudera Manager services wizard to create a Hadoop cluster. The wizard will prompt you to identify the machines to be managed by Cloudera Manager. It will then install the Cloudera Manager Agent on each machine. At that point, your environment will be ready for the CDH software to be installed.



You will then be prompted to choose which CDH services you want to add in the cluster and to which machines you would like to add each service.

At the end of this exercise, you will have Hadoop daemons deployed across your cluster as depicted here (daemons added in this exercise shown in blue).



In subsequent Exercises, you will add more Hadoop services to your cluster.

Because you have only five hosts to work with, you will have to run multiple Hadoop daemons on all the hosts except `lion`, which will be used for Cloudera Manager services only. For example, the NameNode, a DataNode and a NodeManager will all run on `elephant`. Having a very limited number of hosts is not unusual when deploying Hadoop for a proof-of-concept project. Please follow the best practices in the “Planning Your Hadoop Cluster” chapter when sizing and deploying your production clusters.

After completing the installation steps, you will review a Cloudera Manager Agent log file and review processes running on a machine in the cluster. Finally, there is a section at the end of this exercise that provides a brief tour of the Cloudera Manager Admin UI.

IMPORTANT: This exercise builds on the previous one. If you were unable to complete the previous exercise or think you may have made a mistake, run the following command on `elephant` and follow the prompts to prepare for this exercise before continuing:

```
$ ~/training_materials/admin/scripts/reset_cluster.sh
```

Login to Cloudera Manager Admin UI

1. Verify that you are running a SOCKS5 proxy server started with the `start_SOCKS5_proxy.sh` script. The proxy server should be running in a separate terminal window on the Get2EC2 VM.
2. Start the Cloudera Manager Admin Console from a browser. The URL is `http://lion:7180`.
If an Unable to Connect message appears, the Cloudera Manager server has not yet fully started. Wait several moments, and then attempt to connect again.
3. Log in with the username `admin`. The password is `admin`.

The “Welcome to Cloudera Manager” page appears with a table that provides details about editions of Cloudera Manager software.

Install Cloudera Manager Agents

1. A green box highlighting a product edition column should appear.

Select the Cloudera Enterprise Data Hub Edition Trial.

Click Continue.

	Cloudera Express	Cloudera Enterprise Data Hub Edition Trial	Cloudera Enterprise
License	Free	60 Days After the trial period, the product will continue to function as Cloudera Express . Your cluster and your data will remain unaffected.	Annual Subscription Upload License
Node Limit	Unlimited	Unlimited	Unlimited
CDH	✓	✓	✓
Core Cloudera Manager Features	✓	✓	✓
Advanced Cloudera Manager Features		✓	✓
Cloudera Navigator		✓	✓
Cloudera Support			✓

For full list of features available in Cloudera Express and Cloudera Enterprise, [click here.](#)

Continue

2. The “Thank you for choosing Cloudera Manager and CDH” page appears.

Click Continue.

3. The “Specify hosts for your CDH cluster installation” page appears.

Type in the names of all five machines: elephant tiger horse monkey lion

Click Search. All five machines should be found. Ensure they are all selected.

<input checked="" type="checkbox"/>	Expanded Query	Hostname (FQDN)	IP Address	Currently Managed	Result
<input checked="" type="checkbox"/>	elephant	elephant	192.168.123.1	No	Host ready
<input checked="" type="checkbox"/>	horse	horse	192.168.123.3	No	Host ready
<input checked="" type="checkbox"/>	lion	lion	192.168.123.5	No	Host ready
<input checked="" type="checkbox"/>	monkey	monkey	192.168.123.4	No	Host ready
<input checked="" type="checkbox"/>	tiger	tiger	192.168.123.2	No	Host ready

Click Continue.

- The “Cluster Installation – Select Repository” page appears.

Specify the following option:

- Choose Method - Use Parcels

Click the “More Options” button.

- In the “Remote Parcel Repository URLs” area, remove all the current lines by using the minus (-) sign button.
- Once the existing entries are all removed, click on the plus (+) sign to add a new URL.
- In the blank field that appears, type in `http://lion:8000`
- Click OK to return to the Select Repository page.

The ‘Select Repository’ page should now show that `CDH-5.x.x-x.cdh5.x.x.px.xx` (where each x is a version number) is available. Select this version of CDH.

In the “Select the specific release of the Cloudera Manager Agent you want to install on your hosts” area choose “Custom Repository”.

- In the blank field that appears, type in `http://lion:8050`

Click Continue.

- The “Cluster Installation – JDK Installation Options” page appears.

A supported version of the Oracle JDK is already installed.

Verify the box is *unchecked* so that the installation of JDK will be skipped.

Click Continue.

- The “Cluster Installation – Enable Single User Mode” page appears.

Keep the default setting (Single User Mode *not* enabled) and click Continue.

- The “Cluster Installation – Provide SSH login credentials” page appears.

Keep “Login To All Hosts As” set to root.

For “Authentication Method: choose “All hosts accept same private key”

Click the Browse button. In the “Places” column choose “training”. Then, in the “Name” area, right-click (or on a Mac Ctrl-click), and select “Show Hidden Files.” Finally, click into the .ssh directory, select the **id_rsa** file and click Open.

Leave the passphrase fields blank fields and click Continue.

When prompted to continue with no passphrase click OK.

- The “Cluster Installation – Installation in progress” page appears.

Cloudera Manager installs the Cloudera Manager Agent on each machine.

5 of 5 host(s) completed successfully.			
Hostname	IP Address	Progress	Status
elephant	192.168.123.1	<div style="width: 100%;"><div style="width: 100%; background-color: #2e6b2e;"></div></div>	✓ Installation completed successfully. Details
horse	192.168.123.3	<div style="width: 100%;"><div style="width: 100%; background-color: #2e6b2e;"></div></div>	✓ Installation completed successfully. Details
lion	192.168.123.5	<div style="width: 100%;"><div style="width: 100%; background-color: #2e6b2e;"></div></div>	✓ Installation completed successfully. Details
monkey	192.168.123.4	<div style="width: 100%;"><div style="width: 100%; background-color: #2e6b2e;"></div></div>	✓ Installation completed successfully. Details
tiger	192.168.123.2	<div style="width: 100%;"><div style="width: 100%; background-color: #2e6b2e;"></div></div>	✓ Installation completed successfully. Details

Once the installation successfully completes on all machines, click Continue.

Install Hadoop Cluster

- The “Cluster Installation – Installing Selected Parcels” page appears.

The CDH parcel is downloaded, distributed, and activated on all hosts in the cluster.

The “distributing” action may take a few minutes since this involves copying the large CDH parcel file from the lion machine to all of the other machines in the cluster.

When it completes, click Continue.

2. The “Cluster Installation – Inspect hosts for correctness” page appears.

After a moment, output from the Host Inspector appears.

Click Finish.

3. The “Cluster Setup - Choose the CDH 5 services that you want to install on your cluster” page appears.

Click “Custom Services”.

A table appears with a list of CDH service types.

4. Select the HDFS and YARN (MR2 Included) service types.

You will add additional services in later exercises.

Click Continue.

5. The Cluster Setup – Customize Role Assignments page appears.

Assign the following roles.

Role	Node(s)
HDFS	
NameNode	elephant
SecondaryNameNode	tiger
Balancer	horse
HttpFS	Do not specify any hosts
NFS Gateway	Do not specify any hosts
DataNode	elephant, tiger, horse, monkey, but not lion ; the order in which the hosts are specified is not significant
Cloudera Management Service	
Service Monitor	lion
Activity Monitor	lion
Host Monitor	lion
Reports Manager	lion
Event Server	lion
Alert Publisher	lion
YARN (MR2 Included)	
ResourceManager	horse
JobHistoryServer	monkey
NodeManager	Same as DataNode (elephant, tiger, horse, monkey, but not lion)

To assign a role, click the fields with one or more host names in them. For example, the field under NameNode initially has the value `lion`. To change the role assignment for the NameNode to `elephant`, click the field under NameNode (that has `lion` in it). A list of hosts appears. Select `elephant`, and then click OK. The field under NameNode now has the value `elephant` in it.

When you have finished assigning roles, compare your role assignments to the role assignments in the screen shot on the next page.

Verify that your role assignments are correct.

When you are certain that you have the correct role assignments, click Continue and proceed to the step after the screen shot.

6. The “Cluster Setup – Database Setup” page appears.

Fill in the details as show here.

Database Host Name	Database Type	Database Name	Username	Password
lion	MySQL	amon	amonuser	password
lion	MySQL	rman	rmanuser	password

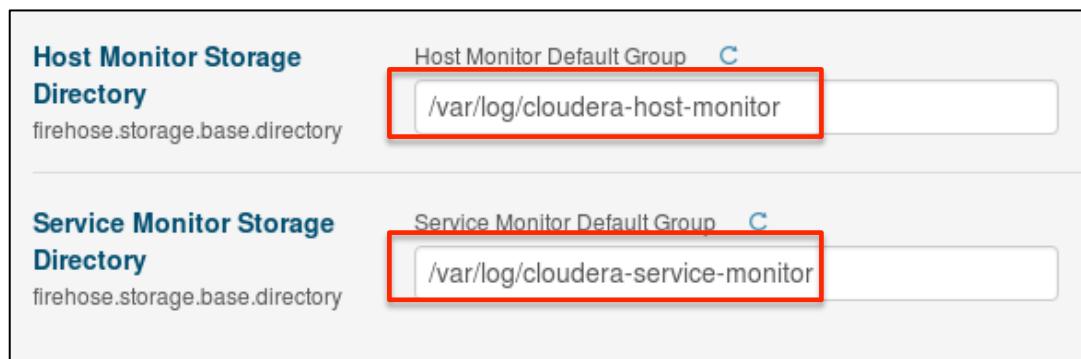
Click “Test Connection” to verify that Cloudera Manager can connect to the MySQL databases you created in an earlier Exercise in this course.

After you have verified that all connections are successful, click Continue.

7. The Cluster Setup – Review Changes page appears.

Specify the following values (change lib to log):

- Host Monitor Storage Directory – /var/log/cloudera-host-monitor
- Service Monitor Storage Directory – /var/log/cloudera-service-monitor



Click Continue.

8. The “Cluster Setup – Progress” page appears.

Progress messages appear while cluster services are being created and starting.

The following is a summary of what Cloudera Manager’s Cluster Setup wizard lists that it does for you:

- Formats the HDFS NameNode
- Starts the HDFS service
- Creates a /tmp directory in HDFS
- Creates a MR2 Job History directory in HDFS
- Creates a YARN NodeManager remote application log directory in HDFS
- Starts the YARN service
- Starts the Cloudera Manager Service service
- Deploys all client configurations

When all the cluster services have started, click Continue.

9. The “Cluster Setup – Congratulations!” page appears.

The page indicates that services have been added and are now configured and running. Click Finish.

The Cloudera Manager Home page appears. Cluster installation is now complete.

10. A note regarding the configuration warnings.

The configuration warnings - highlighted in red in the screen shot below—are expected, and indicate that although the services are in good health, they are operating in low memory conditions. In a production deployment you would want to ensure these warning were addressed. However, in our training environment these warnings can be safely ignored.

The screenshot shows the Cloudera Manager Home page with the following details:

- Cluster Selection:** Cluster 1 (CDH 5.3.0, Parcels)
- Hosts:** 1 warning (highlighted with a red box)
- HDFS:** 1 warning (highlighted with a red box)
- YARN (MR2 Incl...):** 4 warnings (highlighted with a red box)
- Cloudera Management Service:** 4 warnings (highlighted with a red box)

Examine Running Processes on a Cluster Node

1. Review operating system services on a host in the cluster.

On **elephant**:

```
$ chkconfig | grep cloudera
$ sudo service cloudera-scm-agent status
```

You have already added YARN (Including MR2) and HDFS services, yet the only service registered with init scripts at the operating system level is the cloudera-scm-agent service.

With Cloudera Manager managed clusters, the `cloudera-manager-agent` service on each cluster node manages starting and stopping the deployed Hadoop daemons.

2. Review the running Java processes on a host in the cluster.

on **elephant**:

```
$ sudo jps
```

The Hadoop daemons run as Java processes. You should see the NodeManager, NameNode, and DataNode processes running.

Examine the details of one of the running CDH daemons.

```
$ sudo ps -ef | grep NAMENODE
```

You will see details about the process.

If you grep for DATANODE or NODEMANAGER you will see details on those processes. Grep for ‘cloudera’ to see all CDH and Cloudera Manager processes currently running on the machine.

Testing Your Hadoop Installation

We will now test the Hadoop installation by uploading some data. The `hdfs dfs` command you use in this step will be explored in greater detail in the next exercise.

1. Upload some data to the cluster.

The commands below have you unzip a file on the local drive and then upload it to HDFS’ /tmp directory.

On **elephant**:

```
$ cd ~/training_materials/admin/data
$ gunzip shakespeare.txt.gz
$ hdfs dfs -put shakespeare.txt /tmp
```

- Verify that the file is now in HDFS.

In Cloudera Manager choose Clusters > HDFS. Then click on File Browser.

Browse into `tmp` and confirm that `shakespeare.txt` appears.

A Brief Tour of Cloudera Manager

Now that you have the Cloudera Manager managed cluster running, let's briefly explore a few areas in the Admin console.

On the Home page, you will see the cluster named Cluster 1 that you created.

The screenshot shows the Cloudera Manager interface. At the top, there is a navigation bar with tabs: Home, Clusters, Hosts, Diagnostics, Audits, and Charts. The 'Home' tab is selected. Below the navigation bar, the main content area has a title 'Home' with sub-links: Status, All Health Issues, and All Configuration. A list of clusters is displayed, with 'Cluster 1 (CDH 5.2.0, Parcels)' highlighted in green. To the right of this entry is a dropdown menu icon, which is outlined with a red box. Below this, there are three rows of service status indicators: 'Hosts' (green), 'HDFS' (green with a warning icon 'x 1'), and 'YARN (MR2 Incl...)' (green). At the bottom, there is a section titled 'Cloudera Management Service' with a single entry: 'Cloudera Mana...' (green) with a warning icon 'x 4'. The entire screenshot is framed by a black border.

- Click on the drop down menu to the right of Cluster 1 to see many of the actions you can perform on the cluster, such as Add a Service, Stop, and Restart.

2. Click Hosts to view the current status of each managed host in the cluster.
 Clicking on the > icon in the Roles column for a host will reveal which roles are deployed on that host.

	Name	IP	Roles	Load Average	Disk Usage	Physical Memory
<input type="checkbox"/>	 elephant	192.168.123.1	 3 Role(s)	0.00 0.00 0.00	10.3 GiB / 25.6 GiB	614.8 MiB / 1.6 GiB
			 HDFS DataNode			
			 HDFS NameNode			
			 YARN (MR2 Included)			
			NodeManager			
<input type="checkbox"/>	 horse	192.168.123.3	 4 Role(s)	0.00 0.00 0.00	10.5 GiB / 25.6 GiB	622.6 MiB / 1.6 GiB
<input type="checkbox"/>	 lion	192.168.123.5	 6 Role(s)	1.00 0.47 0.30	10.6 GiB / 27.5 GiB	2.6 GiB / 3.5 GiB

In this exercise and the exercises that follow you will discover many other areas of Cloudera Manager that will prove useful for administering your Hadoop cluster(s).

This is the end of the exercise.

Hands-On Exercise: Working With HDFS

In this Hands-On Exercise you will copy a large Web server log file into HDFS and explore the results in the Hadoop NameNode Web UI and the Linux file system. You will then create a snapshot of a directory in HDFS, delete some data and then restore back to its original location in HDFS.

IMPORTANT: This exercise builds on the previous one. If you were unable to complete the previous exercise or think you may have made a mistake, run the following command and follow the prompts to prepare for this exercise before continuing:

```
$ ~/training_materials/admin/scripts/reset_cluster.sh
```

Perform all Command Line steps in this exercise on **elephant**.

Confirm HDFS Processes and Settings

1. Confirm that all HDFS processes are running.

From the Cloudera Manager Home page click on HDFS and then click on the Instances tab.

Notice that the three daemons that support HDFS – the NameNode, SecondaryNameNode, and DataNode daemons – are running. In fact there are DataNodes running on four hosts.

2. Determine the current HDFS replication factor.

From the Cloudera Manager Home page click into the Search box and type “replication”.

Choose “HDFS: Replication Factor” which should be one of the search results.

You are taken to the HDFS Configuration page where you will find the `dfs.replication` setting that has the default value of 3.

3. Similarly, use the search box in the HDFS configuration page, and search for “block size” to discover the HDFS Block Size setting which defaults to 128 MiB.

Add Directories and Files to HDFS

1. As the hdfs user, create a home directory for the training user on HDFS and give the training user ownership of it.

On **elephant**:

```
$ sudo -u hdfs hdfs dfs -mkdir -p /user/training/
$ sudo -u hdfs hdfs dfs -chown training /user/training
```

2. Create an HDFS directory called /user/training/weblog, in which you will store the access log.

On **elephant**:

```
$ hdfs dfs -mkdir weblog
```

3. Extract the `access_log.gz` file and upload it to HDFS in a single step.

On **elephant**:

```
$ cd ~/training_materials/admin/data
$ gunzip -c access_log.gz \
| hdfs dfs -put - weblog/access_log
```

The put command uploaded the file to HDFS. The dash after the put command reads the input from `stdin` and writes it to the destination directory.

Since the file size is 504MB, HDFS will have split it into multiple blocks. Let's explore how the file was stored.

4. Run the `hdfs dfs -ls` command to review the file's permissions in HDFS.

On **elephant**:

```
$ hdfs dfs -ls /user/training/weblog
```

Analyze File Storage in HDFS and On Disk

1. Locate information about the `access_log` file in the NameNode Web UI.
 - a. From the Cloudera Manager Clusters menu, choose HDFS for your cluster.
 - b. In the HDFS Status page, click on “NameNode Web UI (Active)”. This will open the NameNode Web UI at `http://elephant:50070`.
 - c. Choose Utilities > “Browse the file system,” then navigate into the `/user/training/weblog` directory.
 - d. Notice that the permissions shown for the `access_log` file are identical to the permissions you observed when you ran the `hdfs dfs -ls` command.
 - e. Now select the `access_log` file in the NameNode Web UI file browser to bring up the File information window.

Notice that the Block Information drop down list has four entries, Block 0, Block 1, Block 2, and Block 3. This makes sense because as you discovered, the HDFS Block Size on your cluster is 128 MB. The extracted `access_log` file is 559 MB.

Blocks 0, 1, and 2 all show a size of 134217728 (or 128MB) in accordance with the size specified in the HDFS block size setting you observed earlier in this exercise. Block 3 is smaller than the others as you can in the “Size” details if you choose it.

Notice also that each block is available on three different hosts in the cluster. This is what you would expect since three is the current (and default) replication factor in HDFS. Also notice that each block may be replicated to different data nodes than the other blocks that make up the file.

- a. Choose Block 0 and write down the value that appears in the Size field. If Block 0 is not replicated on elephant then chose another block that is replicated on elephant.
-

You will need this value when you examine the Linux file that contains this block.

- f. Select the value of the Block ID field and copy it (Edit menu > Copy). You will need this value for the next step in this exercise.

2. Locate the HDFS block on the `elephant` Linux file system.

- a. In Cloudera Manager's HDFS Configuration page, conduct a search for "Data Directory". You will see that the Data Node Data Directory is `/dfs/dn`.
- b. Let's find the blocks stored in that directory.

On **elephant**:

```
$ sudo find /dfs/dn -name '*BLKID*' -ls
```

where BLKID is the actual Block ID you copied from the NameNode Web UI.

- c. Verify that two files with the Block ID you copied appear in the `find` command output – one file with an extension, `.meta`, and another file without this extension.
- d. Verify in the results of the `find` command output that the size of the file containing the HDFS block is exactly the size that was reported in the NameNode Web UI.

3. Starting any Linux editor with `sudo`, open the file containing the HDFS block. Verify that the first few lines of the file match the first chunk of the `access_log` file content.

Note: You must start your editor with `sudo` because you are logged in to Linux as the `training` user, and this user does not have privileges to access the Linux file that contains the HDFS block.

```
$ sudo head /dfs/dn/path/to/block
```

Note: Replace `/path/to/block` in the command above with the actual path to the block as shown in the results of the `find` command you ran in the previous step.

You can review the `access_log` file content on HDFS as follows:

```
$ hdfs dfs -cat weblog/access_log | head -
```

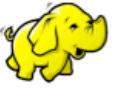
The results returned by the last two commands should match exactly.

This is the end of the exercise.

Hands-On Exercise: Running YARN Applications

In this exercise you will run two YARN applications on your cluster. The first application is a MapReduce job. The second is an Apache Spark application. You will add the Spark service on your cluster before running the Spark application.

After completing this exercise, your cluster will have the following components installed (items installed in this exercise highlighted in blue):

					
	elephant	horse	monkey	tiger	lion
HDFS NameNode	✓				
HDFS SecondaryNameNode				✓	
HDFS DataNode	✓	✓	✓	✓	
HDFS Balancer		✓			
YARN (MR2 Included) NodeManager	✓	✓	✓	✓	
YARN (MR2 Included) ResourceManager		✓			
YARN (MR2 Included) JobHistory Server			✓		
Spark Gateway			✓		
Spark History Server			✓		
Cloudera Manager Server					✓
Cloudera Manager Server Database					✓
Cloudera Management Services					✓
Cloudera Manager Agent	✓	✓	✓	✓	✓

IMPORTANT: This exercise builds on the previous one. If you were unable to complete the previous exercise or think you may have made a mistake, run the following command and follow the prompts to prepare for this exercise before continuing:

```
$ ~/training_materials/admin/scripts/reset_cluster.sh
```

Perform all steps in this exercise on **elephant**.

Submitting a MapReduce Application to Your Cluster

We will now test the Hadoop installation by running a sample Hadoop application that ships with the Hadoop source code. This is WordCount, a classic MapReduce program. We'll run the WordCount program against the Shakespeare data added to HDFS in a previous exercise.

1. Since the code for the application we want to execute is in a Java Archive (JAR) file, we'll use the `hadoop jar` command to submit it to the cluster. Like many MapReduce programs, WordCount accepts two additional arguments: the HDFS directory path containing input and the HDFS directory path into which output should be placed. Therefore, we can run the WordCount program with the following command.

On **elephant**:

```
$ hadoop jar /opt/cloudera/parcels/CDH-5.3.2-1.cdh\  
5.3.2.p0.10/lib/hadoop-mapreduce/hadoop-mapreduce-\  
examples.jar wordcount /tmp/shakespeare.txt counts
```

Verifying MapReduce Job Output

2. Once the program has completed you can inspect the output by listing the contents of the output (`counts`) directory.

On **elephant**:

```
$ hdfs dfs -ls counts
```

3. This directory should show all the data output for the job. Job output will include a `_SUCCESS` flag and one file created by each Reducer that ran. You can view the output by using the `hdfs dfs -cat` command.

On **elephant**:

```
$ hdfs dfs -cat counts/part-r-00000
```

Review the MapReduce Application Details and Logs

In this task you will start by looking at details in the YARN Applications area of Cloudera Manager. The Application Details link in Cloudera Manager will then take you to the HistoryServer Web UI at <http://monkey:19888>.

As you go through the steps below, see if you can reconstruct what occurred where when you ran the MapReduce job, by creating a chart like the one below.

	Node(s)
Application Master	
Map Task(s)	
Reduce Task(s)	

1. Locate your MapReduce application in Cloudera Manager.

In Cloudera Manager, choose Clusters > YARN (MR2 Included), then click on Applications.

In the Results tab that displays, you should see the MapReduce application that just ran.

Note: There will be an entry for each completed MapReduce job that you have run on your cluster within the timeframe of your search. The default search is for applications run within the last 30 minutes.

2. Access the HistoryServer Web UI to discover where the Application Master ran.

From the drop down menu for the “word count” application, choose “Application Details.”

Results Charts Export

12/09/2014 4:48 PM - word count
 12/09/2014 4:49 PM ID: [job_1418165435732_0002](#)

Type: MAPREDUCE
 Pool: root.training
 Mapper: WordCount\$TokenizerMapper
 Reducer: WordCount\$IntSumReducer
 Duration: 34.88s
 CPU Time: 13.83s
 File Bytes Read: 489.3 KIB
 HDFS Bytes Read: 5.2 MIB
 Memory Allocation: 32.9M
 File Bytes Written: 1.3 MIB
 HDFS Bytes Written: 696.8 KIB

This action will open a page in the HistoryServer Web UI with details about the job.

3. Locate where the ApplicationMaster ran and view the log.

Notice the ApplicationMaster section shows which cluster node ran the MapReduce Application Master. Click the “logs” link to view the ApplicationMaster log.

Notice also the number of map and reduce tasks that ran in order to complete the word count job. The number of reducers run by the job should correspond to the number of part-r-##### files you saw when you ran the hdfs dfs -ls command earlier. There are no part-m-##### files because the job ran at least one reducer.

4. Locate where the mapper task ran and view the log.

From the HistoryServer Web UI’s “Job” menu choose “Map tasks”.

From the Map Tasks table, click on the link in the Name column for the task.

The Attempts table displays. Notice the “Node” column shows you where the map task attempt ran.

Click the “logs” link and review the contents of the mapper task log. When done, click the browser back button to return to the previous page.

5. Locate where the reduce tasks ran and view the logs.

From the HistoryServer Web UI’s Job menu choose “Reduce tasks.”

From the Reduce Tasks table, click on the link in the Name column for one of the tasks.

The Attempts table displays. Notice the “Node” column shows you where this reducer task ran.

Click the “logs” link and review the contents of the log. Observe the amount of output in the Reducer task log. When done, click the browser back button to return to the previous page.

6. Determine the log level for Reducer tasks for the word count job.

Expand the Job menu and choose “Configuration.”

Twenty entries from the job configuration that were in effect when the word count job ran appear.

In the Search field, enter `log.level`.

Locate the `mapreduce.reduce.log.level` property. Its value should be `INFO`.

Note: `INFO` is default value for the “JobHistory Server Logging Threshold” which can be found in the Cloudera Manager YARN Configuration page for your cluster.

Run the MapReduce Application with a Custom Log Level Setting

1. Remove the `counts` directory from HDFS and rerun the WordCount program, this time passing it a log level argument.

On `elephant`:

```
$ hdfs dfs -rm -r counts
$ hadoop jar /opt/cloudera/parcels/CDH-5.3.2-1.cdh\
5.3.2.p0.10/lib/hadoop-mapreduce/hadoop-mapreduce-\
examples.jar wordcount \
-D mapreduce.reduce.log.level=DEBUG \
/tmp/shakespeare.txt counts
```

Note: You must delete the `counts` directory before running the WordCount program a second time because MapReduce will not run if you specify an output path which already exists.

MapReduce programs coded to take advantage of the Hadoop ToolRunner allow you to pass several types of arguments to Hadoop, including run-time configuration parameters. The `hadoop jar` command shown above sets the log level for reduce tasks to DEBUG.

Note: The `-D` option, as used in the `hadoop jar` command above, allows you override a default property setting by specifying the property and the value you want to assign.

When your job is running, look for a line in standard output similar to the following:

```
14/12/09 05:47:16 INFO mapreduce.Job: Running job:
job_1391249780844_0004
```

2. After the job completes, locate and view one of the reducer logs.

From Cloudera Manager's YARN Applications page for your cluster, locate the entry for the application that you just ran.

Click on the ID link, and use the information available under the Job menu's "Configuration" and "Reduce tasks" links to verify the following:

- The value of the `mapreduce.reduce.log.level` configuration attribute is DEBUG.

- The Reducer task's logs for this job contain DEBUG log records and the logs are larger than the number of records written to the Reducer task's logs during the previous WordCount job execution.
3. Verify the results of the word count job were written to HDFS using any of the following three methods.

Option 1: In Cloudera Manager, browse to the HDFS page for your cluster, then choose File Browser. Drill down into `/user/training/counts`.

Option 2: Access the HDFS NameNode Web UI at `http://elephant:50070`. Choose Utilities > “Browse the file system”, and navigate to the `/user/training/counts` directory.

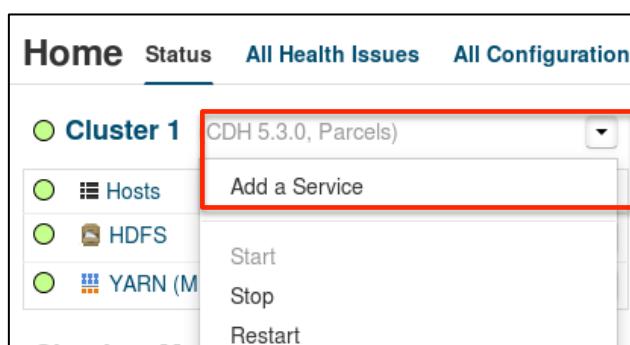
Option 3: on any machine in the cluster that has the DataNode installed (`elephant`, `tiger`, `monkey`, or `horse`) run the following command in a terminal:

```
$ hdfs dfs -tail counts/part-r-00000
```

Add the Apache Spark Service

In this task, you will add the Spark service to your cluster using Cloudera Manager. You will then run a Spark application.

1. In Cloudera Manager, navigate to the Home page.
2. Select the “Add a Service” option from the drop-down menu for Cluster 1.



The Add Service Wizard appears.

3. Select Spark and click Continue.

The “Select the set of dependencies for your new service” page appears.

4. Select the row containing the hdfs and yarn services, then click Continue.

The Customize Role Assignments page appears.

5. Specify host assignments as follows:

- History Server – monkey only
- Gateway – monkey only

Click Continue.

6. Progress messages appear on the Progress page.

When the adding of the service has completed, click Continue.

7. The Congratulations page appears.

Click Finish.

8. The Home page appears.

A status indicator shows you that the Spark service is in good health.

Note: you may notice a configuring warning that appears next to “Hosts” on the Cloudera Manager home page. If you look into it, Cloudera Manager indicates that memory is overcommitted on host monkey. This configuration issue, along with the five other configuration warnings that appeared after the initial cluster installation, would need to be addressed in a true production cluster, however they can be safely ignored in the classroom environment.

Run Spark as a YARN Application

You should complete these steps on monkey since monkey is where you just added the Spark gateway role.

1. Start the Spark shell and connect to the yarn-client spark context on `monkey`.

Recall that the Spark Gateway service was installed on `monkey` so the Spark shell should be run from `monkey`.

On `monkey`:

```
$ spark-shell --master yarn-client
```

The Scala Spark shell will launch. You should eventually see the message, “Spark context available as `sc`.” If necessary, click the <Enter> key on your keyboard to see the `scala>` prompt.

2. Type in the commands below to run a word count application using the `shakespeare.txt` file that is already in HDFS.

This accomplishes something very similar to the job you ran in the MapReduce exercise, but this time the computational framework being used is Spark.

```
scala> val file = sc.textFile(  
  "hdfs://elephant:8020/tmp/shakespeare.txt")  
scala> val counts = file.flatMap(line => line.split(  
  " ")).map(word => (word, 1)).reduceByKey(  
  _ + _).sortByKey()  
scala> counts.saveAsTextFile(  
  "hdfs://elephant:8020/tmp/sparkcount")  
scala> sc.stop()  
scala> sys.exit()
```

3. View the application results written to HDFS by Spark.

On `elephant`:

```
$ hdfs dfs -cat /tmp/sparkcount/part-00000 | less
```

Review Application Details in the Spark History Server

View the Spark application details in Cloudera Manager and the Spark History Server Web UI.

1. In Cloudera Manager, go to the YARN Applications page for your cluster.

You will see a “Spark shell” application.

2. Click the link in the ID field.

A page in the YARN Resource Manager Web UI opens with details about the application.

3. Click on the History link.

A Spark Jobs page in the Spark History Server Web UI opens.

Notice that this Spark Application consisted of two jobs that are now completed.

4. In the Completed Jobs area, click on the “sortByKey...” link for the first job that ran (Job Id 0).

Notice that this first job consisted of two stages.

5. Click on the “map at...” link for the first stage (Stage 0).

The “Details for Stage 0” page appears.

Here you can see that there were two tasks in this stage and you can see on which executor and on which host each task ran. You can also see tasks details such as duration, input data size, the amount of data written to disk during shuffle operations.

6. Click on the Executors tab to see a summary of all the executors used by the Spark application.

Review the Spark Application Logs

1. Access the Spark application logs from the command line.

First locate the application ID.

On **elephant**:

```
$ yarn application -list -appStates FINISHED
```

Copy the application ID for the Spark application returned by the command above.

Now run this command (where appId is the actual application ID).

On **elephant**:

```
$ yarn logs -applicationId appId | less
```

Scroll through the logs returned by the shell. Notice that the logs for all the containers that ran the Spark executors are included in the results.

These Spark application logs are stored in HDFS under /user/spark/\applicationHistory.

This is the end of the exercise.

Hands-On Exercise: Explore Hadoop Configurations and Daemon Logs

IMPORTANT: This exercise builds on the previous one. If you were unable to complete the previous exercise or think you may have made a mistake, run the following command and follow the prompts to prepare for this exercise before continuing:

```
$ ~/training_materials/admin/scripts/reset_cluster.sh
```

Exploring Hadoop Configuration Settings

In this task, you will explore some of the Hadoop configuration files auto-generated by Cloudera Manager.

1. Go to the directory that contains the Hadoop configuration for Cloudera Manager-managed daemons running on `elephant` and then view the contents.

On `elephant`:

```
$ cd /var/run/cloudera-scm-agent/process  
$ sudo tree
```

Notice how there are separate directories for each role instance running on `elephant` - DataNode, NameNode, and NodeManager. Notice also that some files, such as `hdfs-site.xml` and `core-site.xml`, exist in more than one daemon's configuration directory.

2. Compare the first 20 lines of the NameNode's copy of `hdfs-site.xml` to the NodeManager's copy of the same file.

On `elephant`:

```
$ sudo head -20 nn-hdfs-NAMENODE/core-site.xml
$ sudo head -20 nn-yarn-NODEMANAGER/core-site.xml
```

In the commands above, replace each *nn* with the actual numbers generated by Cloudera Manager.

The entries in these `core-site.xml` files reflect the settings configured in Cloudera Manager for these particular role instances. Some of the settings reflect choices you made when you ran the Cloudera Manager installation wizard. Other entries are optimal initial or default values chosen by Cloudera Manager.

3. Make a configuration change in Cloudera Manager.

In Cloudera Manager, browse to the HDFS page for your cluster and then choose Configuration.

Conduct a search for the word “trash”. The NameNode Default Group “Filesystem Trash Interval” setting appears.

Double-click into the Value area where it currently reads “1 day(s)”.

Change this value to 2 day(s). Click Save Changes.

Notice the  “Stale Configuration - Restart needed” icon that appears on the screen. Click on it.

The “Stale Configurations - Review Changes” screen appears, showing the changes to `core-site.xml` that will be made.

Click “Restart Cluster”. The “Cluster 1 Stale Configurations - Restart Cluster” screen appears.

Click “Restart Now”. The “Stale Configurations - Progress” screen appears. Wait for the child commands to complete successfully. Click Finish.

4. Return to the elephant terminal window and list the contents of the `/var/run/cloudera-scm-agent/process` directory.

```
$ sudo ls -l /var/run/cloudera-scm-agent/process
```

Notice now how there are now two directories each for NameNode, DataNode, and NodeManager. The old settings have been retained, however the new settings have also been deployed and will now be used. The directory with the higher number in the name is the newer one.

5. Find the difference between the old NameNode `core-site.xml` file and the new one.

On **elephant**:

```
$ sudo diff -C 2 nn-hdfs-NAMENODE/core-site.xml nn-hd\  
fs-NAMENODE/core-site.xml
```

The *nn* values above should be replaced with the actual numbers with which the configuration directories are named.

You should see that the `fs.trash.interval` property value change has been deployed to the new NameNode configuration file.

6. Revert the configuration change and restart the cluster.

In Cloudera Manager, go to the HDFS Configuration page for your cluster.

Click the “History and Rollback” button.

The “Configuration and Role Group History” page displays.

Message	Date	User
Current Revision	Mar 3, 2015 9:11:33 PM PST	
(Details...)	Feb 18, 2015 10:54:05 PM PST	admin
Updated service and role type configurations. (Details...)	Feb 6, 2015 10:33:03 PM PST	admin
Express wizard autoconfigured (Details...)		

Under Current Revision click “Details”.

The Revision Details screen displays. Notice the Filesystem Trash Interval property value that you just modified is listed. Choose “Revert Configuration Changes”.

A message appears indicating that the revision was reverted. Click onto the HDFS Status page and notice the “Stale Configuration - Restart needed” icon.

Click on the icon and follow the steps to restart the cluster.

Examining Hadoop Daemon Log Files

In the previous Exercise, you reviewed the application logs from MapReduce and Spark running as YARN applications. Here you will review Hadoop daemon log files, including the HDFS NameNode and YARN ResourceManager log files.

With Cloudera Manager, Hadoop daemons generate a `.log.out` file, a standard error log (`stderr.log`), and a standard output log (`stdout.log`).

In this task, you will examine Hadoop daemon log files using the NameNode Web UI, Cloudera Manager, the ResourceManager Web UI and the NodeManager Web UI.

1. View the NameNode log file using the NameNode Web UI.

Access the NameNode Web UI from the Quick Links on the HDFS Status page in Cloudera Manager.

From the NameNode Web UI, select Utilities > Logs. The list of folders and files in the /var/log/hadoop-hdfs directory on elephant appears.

Open the NameNode log file and review the file.

2. Access the daemon logs directly from Cloudera Manager.

In Cloudera Manager, choose Hosts from the top navigation bar.

Select elephant and then choose the Processes tab.

Locate the row for the NameNode and click “Full log file.”

The Log Details page opens at the tail end of the NameNode log file. Scroll up to view earlier log messages.

Note: If the log file is very large, and you want to see messages near the top, scrolling in the Cloudera Manager UI will be slow. Other tools provide quick access to the entire log file.

Click “Download Full Log” (in the upper right corner of the Log Details page) to download the entire log.

3. Review the NameNode daemons standard error and standard output logs using Cloudera Manager.

Return to the Processes page for elephant.

Click the Stdout link for the NameNode instance. The standard output log appears. Review the file, then return to the Processes page.

Click the Stderr link for the NameNode instance. The standard error log appears. Review the file.

Note: if you want to locate these log files on disk, they can be found on elephant in the /var/log/hadoop-hdfs and /var/run/cloudera-scm-agent/process/nn-hdfs-NAMENODE/logs directories.

4. Using Cloudera Manager, review recent entries in the SecondaryNameNode logs.

To find the log, go to the HDFS Instances page for your cluster, then click on the SecondaryNameNode role type, and in Status page for the `tiger` host, click “Log File”.

5. Access the ResourceManager log file using the ResourceManager Web UI.

Start the ResourceManager Web UI (from Cloudera Manager’s YARN Status page or by specifying the URL `http://horse:8088` in your browser).

Choose “Nodes” from the Cluster menu on the left side of the page.

Click the `horse:8042` link to be taken to the NodeManager Web UI.

Expand the Tools menu on the left side of the page.

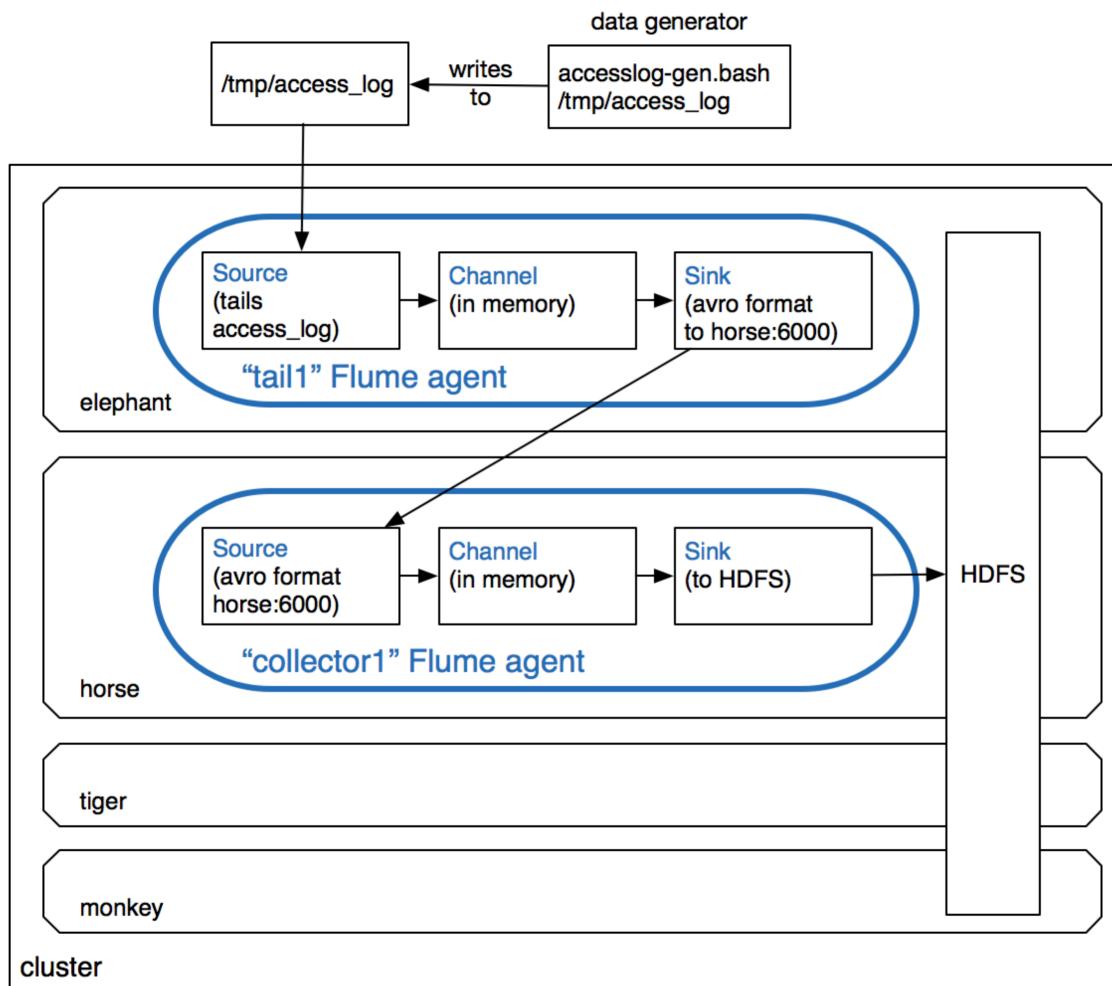
Click “Local logs.”

Finally, click the entry for the ResourceManager log file and review the file.

Hands-On Exercise: Using Flume to Put Data into HDFS

In this Hands-On Exercise you will use Flume to import dynamically generated data into HDFS. A very common use case for Flume is to collect access logs from a large number of Web servers on your network; we will simulate a simple version of this in the following exercise.

The diagram below shows the data flow that will occur once you complete the Exercise.



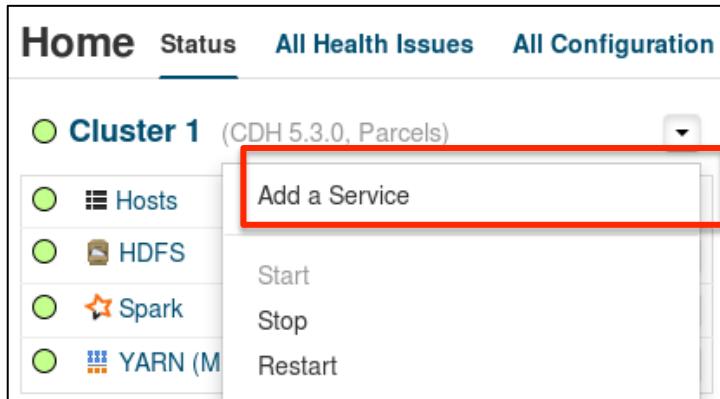
IMPORTANT: This exercise builds on the previous one. If you were unable to complete the previous exercise or think you may have made a mistake, run the following command and follow the prompts to prepare for this exercise before continuing:

```
$ ~/training_materials/admin/scripts/reset_cluster.sh
```

Adding the Flume Service

1. Add the Flume service on elephant and horse.

From the Cloudera Manager Home page, click the down arrow next to your cluster and choose “Add a Service”.



Select “Flume” and click “Continue”. Note that HDFS is a dependency, however the HDFS service is already added. Click “Continue” again.

Use the “Select hosts” button to add the Flume Agent on both elephant and horse then click OK and Continue. At the Congratulations screen click “Finish”.

Note: you may notice that there are now two new ‘Hosts’ configuration issues , this time on elephant and horse that - like the one that appeared on monkey earlier - are related to memory overcommit validation thresholds. In a true production cluster, you would want to address all configuration issues, however you can safely ignore these in the classroom environment.

2. Update the configuration for the Flume agent on elephant.

From the Cloudera Manager Home page, click the Flume link and then click on the Instances tab.

Select the “Agent” that resides on `elephant`, then click Configuration.

Delete the default contents of the two properties listed in the table below entirely and replace with the lines shown below.

Note: the Configuration File lines are also available in

`~/training_materials/admin/scripts/flume-tail1.txt`

Tip: You can expand the Configuration File text area to make it easier to edit in by dragging it out the bottom right corner of the text box.

Property	Value
Agent Name	<code>tail1</code>
Configuration File	<pre> tail1.sources = src1 tail1.channels = ch1 tail1.sinks = sink1 tail1.sources.src1.type = exec tail1.sources.src1.command = tail -F /tmp/access_log tail1.sources.src1.channels = ch1 tail1.channels.ch1.type = memory tail1.channels.ch1.capacity = 500 tail1.sinks.sink1.type = avro tail1.sinks.sink1.hostname = horse tail1.sinks.sink1.port = 6000 tail1.sinks.sink1.batch-size = 1 tail1.sinks.sink1.channel = ch1 </pre>

Click Save Changes.

3. Update the configuration for the Flume agent on `horse`.

From the Cloudera Manager Flume page, click on the Instances tab.

Select the Agent that resides on `horse`, then click Configuration.

Delete the default contents of the two properties listed in the table below entirely and replace with the lines shown below.

Note: the Configuration File lines are also available in
`~/training_materials/admin/scripts/flume-collector1.txt`

Property	Value
Agent Name	collector1
Configuration File	<pre> collector1.sources = src1 collector1.channels = ch1 collector1.sinks = sink1 collector1.sources.src1.type = avro collector1.sources.src1.bind = horse collector1.sources.src1.port = 6000 collector1.sources.src1.channels = ch1 collector1.channels.ch1.type = memory collector1.channels.ch1.capacity = 500 collector1.sinks.sink1.type = hdfs collector1.sinks.sink1.hdfs.path = hdfs://elephant/user/flume/collector1 collector1.sinks.sink1.hdfs.filePrefix = access_log collector1.sinks.sink1.channel = ch1 </pre>

Ensure that “`collector1.sinks.sink1.hdfs.path = hdfs://elephant/user/flume/collector1`” shown above is all on a single line.

Click “Save changes.”

4. Create the /user/flume/collector1 directory in HDFS to store the files.

On **elephant**:

```
$ sudo -u hdfs hdfs dfs -mkdir -p \
/usr/flume/collector1
$ sudo -u hdfs hdfs dfs -chown -R flume /user/flume
```

Starting the Data Generator

1. Open a new terminal window on elephant (or an ssh connection to elephant). In this terminal window, run the accesslog-gen.bash shell script, which simulates a Web server creating log files. This shell script also rotates the log files regularly.

On **elephant**:

```
$ accesslog-gen.sh /tmp/access_log
```

Note: The accesslog-gen.bash script is specific to the training environment and is *not* part of CDH.

2. Open a second new terminal window on elephant (or an ssh connection to elephant). Verify that the log file has been created. Notice that the log file is rotated periodically.

On **elephant**:

```
$ ls -l /tmp/access*
-rw-rw-r-- 1 training training 498 Nov 15 15:12 /tmp/access_log
-rw-rw-r-- 1 training training 997 Nov 15 15:12 /tmp/access_log.0
-rw-rw-r-- 1 training training 1005 Nov 15 15:11 /tmp/access_log.1
```

Starting the Flume Collector Agent

Here you start the Flume agent that will insert the data into HDFS. This agent receives data from the source Flume agent.

1. Start the collector1 Flume Agent on horse.

In Cloudera Manager, go to Flume's Instances tab. Select the Agent hosted on horse.

From the "Actions for Selected" menu choose Start. In the "Start" window that appears, click "Start".

In the "Command Details: Start" screen wait for confirmation that the agent started successfully. Click Close.

Starting the Flume Source Agent

Here you start the agent that reads the source log files and passes the data along to the collector agent you have already started.

1. Start the tail1 Flume Agent on elephant.

From the Cloudera Manager Flume Instances tab, select the Agent hosted on elephant.

From the "Actions for Selected" menu choose Start. In the "Start" window that appears, click "Start".

In the "Command Details: Start" screen wait for confirmation that the agent started successfully. Click Close.

Viewing Data in HDFS

1. Confirm the data is being written into HDFS.

In Cloudera Manager browse to the HDFS page for your cluster and click on File Browser.

Drill down into /user/flume/collector1. You should see many access_log files.

2. View Metric Details.

Return to the Flume page and click on the Metric Details tab. Here you can see details related to the Channels, Sinks, and Sources of your running Flume agents. If you are interested, an explanation of the metrics available is at <http://bit.ly/flumetrics>.

Increase the File Size in HDFS (Optional)

These two steps are optional, but may be of interest to some students.

1. Edit the Collector1 agent configuration settings on `horse` by adding these three additional name value pairs:

```
collector1.sinks.sink1.hdfs.rollSize = 2048
collector1.sinks.sink1.hdfs.rollCount = 100
collector1.sinks.sink1.hdfs.rollInterval = 60
```

Click Save Changes.

2. From the Flume Status page, click on the  Stale Configuration - refresh needed icon and follow the prompts to refresh the cluster.
3. Execute `hdfs dfs -ls /user/flume/collector1` in a terminal window and note the file size of the more recent content posted by Flume to HDFS.

Viewing the Logs

1. Check the log files to see messages.

In Cloudera Manager choose Diagnostics > Logs.

Click Select Sources and configure as follows:

- Uncheck all sources except Flume
- Set the Minimum Log Level to INFO
- Leave the timeframe of your search set to 30 minutes

Click Search.

Browse through the logged actions from both Flume agents.

Cleaning Up

1. Stop the log generator by hitting Ctrl-C in the first terminal window.
2. Stop both Flume agents from the Flume Instances page in Cloudera Manager.
3. Remove the generated access log files from the /tmp directory to clear up space on your virtual machine.

On **elephant**:

```
$ rm -rf /tmp/access_log*
```

This is the end of the Exercise.

Hands-On Exercise: Importing Data with Sqoop

For this exercise you will import data from a relational database using Sqoop. The data you load here will be used in a subsequent exercise.

Consider the MySQL database `movielens`, derived from the MovieLens project from University of Minnesota. (See note at the end of this exercise.) The database consists of several related tables, but we will import only two of these: `movie`, which contains about 3,900 movies; and `movierating`, which has about 1,000,000 ratings of those movies.

IMPORTANT: This exercise builds on the previous one. If you were unable to complete the previous exercise or think you may have made a mistake, run the following command and follow the prompts to prepare for this exercise before continuing:

```
$ ~/training_materials/admin/scripts/reset_cluster.sh
```

Perform all steps in this exercise on `elephant`.

Reviewing the Database Tables

First, review the database tables to be loaded into Hadoop.

1. Log on to MySQL.

On `elephant`:

```
$ mysql --user=training --password=training movielens
```

2. Review the structure and contents of the `movie` table.

On `elephant`:

```
mysql> DESCRIBE movie;
. . .
mysql> SELECT * FROM movie LIMIT 5;
```

3. Note the column names for the table.
-

4. Review the structure and contents of the movierating table.

On **elephant**:

```
mysql> DESCRIBE movierating;
. . .
mysql> SELECT * FROM movierating LIMIT 5;
```

5. Note these column names.
-

6. Exit MySQL.

On **elephant**:

```
mysql> quit;
```

Adding the Sqoop 1 Client

1. Add the Sqoop 1 Client gateway on **elephant**.

From the Home page in Cloudera Manager, click the down arrow icon next to Cluster 1 and choose “Add a Service”.

Select **Sqoop 1 Client** and click Continue.

At the “Custom Role Assignments” page, click on the Select hosts box and choose to add the Gateway on **elephant**. Click Continue.

The “Progress” page appears. Once the client configuration has been deployed successfully, click Continue. At the “Congratulations” screen click Finish.

- Using sudo, create a symlink to the MySQL JDBC driver.

On **elephant**:

```
$ sudo ln -s /usr/share/java/mysql-connector-java.jar \
/opt/cloudera/parcels/CDH-5.3.2-1.cdh5.3.2.p0.10\
/lib/sqoop/lib/
```

Now run the command below to confirm the symlink was properly created.

On **elephant**:

```
$ readlink -f /opt/cloudera/parcels/CDH-5.3.2-\
1.cdh5.3.2.p0.10/lib/sqoop/lib/mysql-connector-java.jar
```

If the symlink was properly defined, the command should return the /usr/share/java/mysql-connector-java.jar path.

Importing with Sqoop

You invoke Sqoop on the command line to perform several commands. With it you can connect to your database server to list the databases (schemas) to which you have access, and list the tables available for loading. For database access, you provide a connect string to identify the server, and your username and password.

- Show the commands available in Sqoop.

On **elephant**:

```
$ sqoop help
```

You can safely ignore the warning that Accumulo does not exist since this course does not use Accumulo.

- List the databases (schemas) in your database server.

On **elephant**:

```
$ sqoop list-databases \
--connect jdbc:mysql://localhost \
--username training --password training
```

(Note: Instead of entering --password training on your command line, you may prefer to enter -P, and let Sqoop prompt you for the password, which is then not visible when you type it.)

3. List the tables in the movielens database.

On **elephant**:

```
$ sqoop list-tables \
--connect jdbc:mysql://localhost/movielens \
--username training --password training
```

4. Import the movie table into Hadoop.

On **elephant**:

```
$ sqoop import \
--connect jdbc:mysql://localhost/movielens \
--table movie --fields-terminated-by '\t' \
--username training --password training
```

The --fields-terminated-by '\t' option separates the fields in the HDFS file with the tab character, which is sometimes useful if users will be working with Hive and Pig.

Warnings that packages such as hbase, hive-hcatalog, and accumulo are not installed are expected. It is not a problem that these packages are not installed on your system.

Notice how the INFO messages that appear show that a MapReduce job consisting of four map tasks was completed.

5. Verify that the command has worked.

On **elephant**:

```
$ hdfs dfs -ls movie
$ hdfs dfs -tail movie/part-m-00000
```

6. Import the movierating table into Hadoop using the command in step 4 as an example.

Verify that the `movierating` table was imported using the command in step 5 as an example or by using the Cloudera Manager HDFS page's File Browser.

7. Optionally observe the results in Cloudera Manager's YARN Applications page.

Navigate to the YARN Applications page.

Notice the last two YARN applications that ran (`movie.jar` and `movierating.jar`).

Explore the job details for either or both of these jobs.

This is the end of the Exercise

Note:

This exercise uses the MovieLens data set, or subsets thereof. This data is freely available for academic purposes, and is used and distributed by Cloudera with the express permission of the UMN GroupLens Research Group. If you would like to use this data for your own research purposes, you are free to do so, as long as you cite the GroupLens Research Group in any resulting publications. If you would like to use this data for commercial purposes, you must obtain explicit permission. You may find the full dataset, as well as detailed license terms, at <http://www.grouplens.org/node/73>

Hands-On Exercise: Querying HDFS With Hive and Cloudera Impala

In this exercise, you will add Hive and Cloudera Impala services to your cluster, enabling you to query data stored in HDFS.

You will start by adding the ZooKeeper service to your cluster. ZooKeeper is a prerequisite for HiveServer2, which you will deploy when you add the Hive service.

Then you will add the Hive service, including a Hive Metastore Server and HiveServer2, to your Hadoop cluster, and configure the service.

Next, you will add the Impala service to your cluster and configure the service.

Then you will populate HDFS with data from the movierating table and run queries against it using both Hive and Impala.

At the end of this exercise, you should have daemons deployed on your five hosts as follows (new services added in this exercise are highlighted in blue):

	 elephant	 horse	 monkey	 tiger	 lion
HDFS NameNode	✓				
HDFS SecondaryNameNode				✓	
HDFS DataNode	✓	✓	✓	✓	
HDFS Balancer		✓			
YARN (MR2 Included) NodeManager	✓	✓	✓	✓	
YARN (MR2 Included) ResourceManager		✓			
YARN (MR2 Included) JobHistory Server			✓		
ZooKeeper Server	✓	✓		✓	
Sqoop 1 Client Gateway	✓				
Flume Agent	✓	✓			
HiveServer2	✓				
Hive Gateway	✓				
Hive Metastore Server	✓				
Impala Catalog Server			✓		
Impala StateStore			✓		
Impala Daemon	✓	✓	✓	✓	
Spark Gateway				✓	
Spark History Server				✓	
Cloudera Manager Server					✓
Cloudera Manager Server Database					✓
Cloudera Management Services					✓
Cloudera Manager Agent	✓	✓	✓	✓	✓

IMPORTANT: This exercise builds on the previous one. If you were unable to complete the previous exercise or think you may have made a mistake, run the following command and follow the prompts to prepare for this exercise before continuing:

```
$ ~/training_materials/admin/scripts/reset_cluster.sh
```

Adding the ZooKeeper Service

In this task, you will add a ZooKeeper service to your cluster. A running ZooKeeper service is a prerequisite for many other services so you will add this service now. When you add additional services later in the class, you may notice the Exercise

instructions have you select ZooKeeper as part of the set of dependencies for the new service to use.

1. From the Cloudera Manager Home page, select the 'Add a Service' menu option from the drop-down menu to the right of Cluster 1.

The Add Service Wizard appears.

2. Select ZooKeeper and click Continue.

The Customize Role Assignments page appears.

3. Specify the following host assignments:

- Server - elephant, horse, and tiger but **not** lion or monkey

Click OK and then click Continue.

4. The Review Changes page appears.

Review the default values specified on this page and click Continue.

5. Progress messages appear on the Progress page.

When the adding of the service has completed, click Continue.

6. The Congratulations page appears.

Click Finish.

7. The Cloudera Manager Home page appears.

The `zookeeper` service now appears in the list of services.

A health issue icon may appear next to the new ZooKeeper service temporarily, however this should go away momentarily and the status should change to Good Health.

Note: you may notice that there is now one more ‘Hosts’ configuration issues , this time on tiger that - like the ones that appeared earlier - are related to memory overcommit validation thresholds. In a true production cluster, you would want to address all configuration issues, however you can safely ignore these in the classroom environment.

Adding the Hive Service to Your Cluster

In this task, you will add the Hive service to your cluster using Cloudera Manager.

You will configure the Hive Metastore to use the MySQL metastore database and to have a HiveServer2 instance.

Hive and Impala can both make use of a single, common Hive metastore. Recall that you created a few databases by running a SQL script prior to installing your CDH cluster. One of the databases you created is named metastore, which will be used by Hive and Impala as a common metastore for storing table definitions.

At the end of the task, you will run a simple Hive command to verify that the Hive service has been added.

1. From the Cloudera Manager Home page, select the “Add a Service” option for Cluster 1.

The Add Service Wizard appears.

2. Select Hive and click Continue.

The “Select the set of dependencies for your new service” page appears.

3. Select the row containing the hdfs, yarn, and zookeeper services, then click Continue.

The Customize Role Assignments page appears.

4. Specify host assignments as follows:

- Gateway – elephant only
- Hive Metastore Server – elephant only
- WebHCat Server – Do not select any hosts
- HiveServer2 – elephant only

Verify that you have selected *only* elephant and *not any additional hosts* for the Gateway, Hive Metastore Server, and HiveServer2 roles.

Click Continue.

5. The “Database Setup” page appears.

Specify values for the database as follows:

- Database Host Name – lion
- Database Type – MySQL
- Database Name – metastore
- User Name – hiveuser
- Password – password

Click “Test Connection” and verify that connection to the MySQL database is successful.

Click Continue.

6. The Review Changes page appears.

Review the default values specified on this page and click Continue.

7. Progress messages appear on the Progress page.

When the adding of the service has completed, click Continue.

8. The Congratulations page appears.

Click Finish.

9. The Home page appears.

A status indicator shows you that the `hive` service is in good health.

10. Verify that you can run a Hive command from the Beeline shell.

On **elephant**:

```
$ beeline -u jdbc:hive2://elephant:10000/default \
-n training
```

In the Beeline shell, type the following command:

```
> SHOW TABLES;
```

No tables should appear, because you haven't defined any Hive tables yet, but you should not see error messages.

11. Exit the Beeline shell.

```
> !quit
```

Adding the Impala Service

In this task, you will add the Impala service to your cluster.

Cloudera Manager will automatically configure Impala to use the Hive Metastore service that you created earlier in this exercise.

- From the Cloudera Manager Home page, select the Add a Service option for Cluster 1.

The Add Service Wizard appears.

- Select Impala and click Continue.

The "Select the set of dependencies for your new service" page appears.

- Select the row containing the `hdfs` and `hive` services, then click Continue.

The Customize Role Assignments page appears.

4. Specify host assignments as follows:

- Impala Catalog Server – horse
- Impala StateStore – horse
- Impala Daemon – elephant, horse, monkey and tiger

Click Continue.

Note: When you added the Hive service, you specified elephant as a Gateway host, which caused the Hive client to be added on elephant. With Impala, the Impala client—`impala-shell`—is automatically added on all hosts running Impala.

5. The Review Changes page appears.

Review the default values specified for the Impala Daemon Scratch Directories on this page and click Continue.

6. Progress messages appear on the Progress page.

When the adding of the service has completed, click Continue.

7. The Congratulations page appears.

Click Finish.

8. Restart the HDFS service.

After adding Impala, on the Cloudera Manager home page you will notice that the HDFS service has stale configurations as indicated by the  icon that appears.

Click on the “Stale Configuration: Restart needed” icon. The “Review Changes” page appears.

Click “Restart Cluster” then click “Restart Now”. When the action completes click Finish.

9. Confirm the Impala services started.

Browse to the Impala page for your cluster and click on Instances.

You should see that the Impala Catalog Server, Impala Daemons, and Impala StateStore services have all started successfully.

Running Hive and Impala Queries

In this task, you will define the movierating table in Hive and run a simple query against the table. Then you will run the same query in Impala and compare performance.

Note: You already imported the movierating table into HDFS in the Importing Data With Sqoop exercise.

1. Review the movierating table data imported into HDFS during the Sqoop exercise.

On **elephant**:

```
$ hdfs dfs -cat movierating/part-m-00000 | head
```

2. Start the Beeline shell and connect to HiveServer2.

On **elephant**:

```
$ beeline -u jdbc:hive2://elephant:10000 -n training
```

3. Define the movierating table in Hive.

```
> CREATE EXTERNAL TABLE movierating
> (userid INT, movieid STRING, rating TINYINT)
> ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t'
> LOCATION '/user/training/movierating';
```

4. Verify that you created the movierating table in the Hive metastore.

On **elephant**:

```
> SHOW TABLES;
```

You should see an entry for the movierating table.

5. Run a simple Hive test query that counts the number of rows in the movieratings table.

On **elephant**:

```
> SELECT COUNT(*) FROM movierating;
```

Browse to Cloudera Manager's YARN Applications page to view the MapReduce job that was run when you executed the Hive query. In the Results tab, make note of the amount of time the query takes to execute when running in Hive.

6. Terminate the Beeline shell.

On **elephant**:

```
> !quit
```

7. Start the Impala shell.

On **elephant**:

```
$ impala-shell
```

8. Connect to the Impala Catalog Server running on **horse**.

```
> CONNECT horse;
```

9. Since you defined a new table after starting the Impala server on `horse`, you must now refresh that server's copy of the Hive metadata.

```
> INVALIDATE METADATA;
```

10. In Impala, run the same query against the `movierating` table that you ran in Hive.

```
> SELECT COUNT(*) FROM movierating;
```

Compare the amount of time it took to run the query in Impala to the amount of time it took in Hive.

11. Exit the Impala shell.

```
> quit;
```

12. Explore Impala queries in Cloudera Manager

In Cloudera Manager, from the Clusters menu's Activities section, choose Impala Queries.

Notice that both of the queries you ran from the `impala-shell` appear in the Results panel.

For the 'SELECT' query that you ran, choose "Query Details" from the drop down menu on the right. Browse through the query details noting the information about the query that is available to you.

This is the end of the Exercise.

Hands-On Exercise: Using Hue to Control Hadoop User Access

In this exercise, you will configure a Hue environment that provides business analysts with the following capabilities:

- Submitting Pig, Hive, and Impala queries
- Managing definitions in the Hive Metastore
- Browsing the HDFS file system
- Browsing YARN applications

Users will be able to access their environments by using a Web browser, eliminating the need for administrators to install Hadoop client environments on the analysts' systems.

At the end of this exercise, you should have daemons deployed on your five hosts as follows (new daemons shown in blue):

	 elephant	 horse	 monkey	 tiger	 lion
HDFS NameNode	✓				
HDFS SecondaryNameNode				✓	
HDFS DataNode	✓	✓	✓	✓	
HDFS Balancer		✓			
HDFS HttpFS			✓		
YARN (MR2 Included) NodeManager	✓	✓	✓	✓	
YARN (MR2 Included) ResourceManager		✓			
YARN (MR2 Included) JobHistory Server			✓		
ZooKeeper Server	✓	✓		✓	
Sqoop 1 Client Gateway	✓				
Flume Agent	✓	✓			
HiveServer2	✓				
Hive Gateway	✓				
Hive Metastore Server	✓				
Impala Catalog Server			✓		
Impala StateStore			✓		
Impala Daemon	✓	✓	✓	✓	
Spark Gateway	✓		✓		
Spark History Server			✓		
Hue Server			✓		
Oozie Server			✓		
Cloudera Manager Server					✓
Cloudera Manager Server Database					✓
Cloudera Management Services					✓
Cloudera Manager Agent	✓	✓	✓	✓	✓

The Hue server will be deployed on monkey. The HttpFS and Oozie servers on monkey will support several Hue applications.

IMPORTANT: This exercise builds on the previous one. If you were unable to complete the previous exercise or think you may have made a mistake, run the following command and follow the prompts to prepare for this exercise before continuing:

```
$ ~/training_materials/admin/scripts/reset_cluster.sh
```

Adding an HttpFS Role Instance to the hdfs Service

In this task, you will use the Cloudera Manager wizard to add an HttpFS role instance to the `hdfs` service. The HttpFS role instance will reside on `monkey`. After adding the role instance, you will run a `curl` command from the command line to verify that HttpFS works correctly.

1. In Cloudera Manager, navigate to the HDFS Instances page.

2. Click Add Role Instances.

The “Add Role Instances to HDFS” page appears.

3. For HttpFS, specify `monkey`.

Click Continue.

4. The `hdfs` Role Instances page reappears. The `HttpFS (monkey)` role instance now appears in the list of role instances.

Notice that the status for this role instance is Stopped.

A status indicator shows you that the HDFS service has good health.

5. Start the HttpFS role instance.

In the HDFS Instances page, check the box next to `HttpFs` and from the Actions for Selected menu choose Start.

In the Start dialog window click Start. When the Command Details window shows that the command completed successfully click Close.

6. To verify HttpFS operation, run the HttpFS LISTSTATUS operation to examine the content in the /user/training directory in HDFS.

On **elephant**:

```
$ ssh training@monkey netstat -tan | grep :14000
$ curl -s "http://monkey:14000/webhdfs/v1/\
user/training?op=LISTSTATUS&user.name=training" \
| python -m json.tool
```

Note: The HttpFS REST API returns JSON objects. Piping the JSON objects to `python -m json.tool` makes the objects easier to read in standard output.

Adding the Oozie Service

In this task, you will add the Oozie service to your cluster. With Cloudera Manager, Oozie is a prerequisites for adding the Hue service. We won't use these services in these exercises, but feel free to explore them on your own if you like.

You will configure the Oozie instance to reside on `monkey` and the Solr instance to run on `tiger`.

1. In Cloudera Manager, navigate to the Home page.
2. Select the Add a Service option for Cluster 1.

The Add Service Wizard appears.

3. Select Oozie and click Continue.

The “Select the set of dependencies for your new service” page appears.

4. Select the row containing the `hdfs`, `yarn`, and `zookeeper` services, then click Continue.

The Customize Role Assignments page appears.

5. Specify host assignments as follows:

- Oozie Server – monkey

Click Continue.

6. The Review Changes page appears.

Review the default values specified on this page and click Continue.

7. Progress messages appear on the Progress page.

When the adding of the service has completed, click Continue.

8. The Congratulations page appears.

Click Finish.

9. The Home page appears.

A status indicator shows you that the `oozie` service is in good health.

Adding the Hue Service

In this task, you will add a Hue service to your cluster, configuring the Hue instance to run on `monkey`.

1. From the Cloudera Manager Home page, select the Add a Service option for Cluster 1.

The Add Service Wizard appears.

2. Select Hue and click Continue.

The “Select the set of dependencies for your new service” page appears.

3. Select the row containing the `hdfs`, `hive`, `impala`, `oozie`, `yarn`, and `zookeeper` services, then click Continue.

The Customize Role Assignments page appears.

4. Specify host assignments as follows:

- Hue Server – monkey

Click Continue.

5. Progress messages appear on the Progress page.

When the adding of the service has completed, click Continue.

6. The Congratulations page appears.

Click Finish.

7. The Home page appears.

A status indicator shows you that the hue service is in good health.

8. Submit a Hadoop WordCount job so that there will be a MapReduce job entry that you can browse in Hue after you start the Hue UI.

On **elephant**:

```
$ hadoop jar /opt/cloudera/parcels/CDH-5.3.2-1.cdh\
5.3.2.p0.10/lib/hadoop-mapreduce/hadoop-mapreduce-\
examples.jar wordcount /tmp/shakespeare.txt test_output
```

9. Install Spark client configuration on elephant.

Previously you ran the Spark shell on monkey. Here you will add the Spark Gateway role on elephant so that you can run the Spark shell from elephant.

Navigate to the Spark Instances page for your cluster and click Add Role Instances.

Add the Gateway role to elephant.

Click on the “Stale configuration - client configuration redeployment needed”  icon and in the “Cluster 1 Stale Configurations” page click “Deploy Client Configuration”.

In the Deploy Client Configuration page click “Deploy Client Configuration”.

In the Progress screen, wait for the commands to complete, then click Finish.

- 10.** Start the Spark Shell so that there will be a Spark job entry in Hue.

On **elephant**:

```
$ spark-shell --master yarn-client
```

Leave the spark-shell open in the terminal for the rest of this exercise.

Exploring the Hue User Interface

In this task, you will log in to the Hue UI as an administrative user and briefly explore the following applications: Hue Home page, Hive UI, Cloudera Impala Query UI, Pig Editor, File Browser, Metastore Manager, Job Browser, Hue Shell, User Admin, and Help.

You will also explore how the Hue UI reports misconfiguration and determine why you cannot use the Job Designer and Oozie Editor/Dashboard applications.

Logging Into Hue

- 1.** Maximize the browser window to give Hue enough space to display as many options as possible on its top menu.
- 2.** View the Hue UI.

Access the “Hue Web UI” from the Cloudera Manager Hue Status page for your cluster or just browse to the URL at <http://monkey:8888>.

- 3.** Login to Hue.

Note: that as the message box in the browser indicates, as the first person to login to this Hue service, you are actually defining the Hue superuser credentials in this step.

Type in `admin` as the user, with the password `training`, then click “Create Account”.

4. The Quick Start Wizard displays.

Click the Home icon.

At the “Did you know?” dialog, click “Got it, prof!”

The “My documents” page appears.

Access Hive Using Hue

1. If you completed the Querying HDFS With Hive and Cloudera Impala exercise, start the Hive Query Editor by selecting Query Editors > Hive.

Enter the following query to verify that Hive is working in Hue:

```
SHOW TABLES;
```

Click Execute. The result of the query should be the `movierating` table.

Enter another query to count the number of records in the `movierating` table:

```
SELECT COUNT(*) FROM movierating;
```

Give it some time to complete. The UI will first show the Log tab contents, then it should eventually show the Results tab. The query should run successfully.

Access Impala Using Hue

1. If you completed the Querying HDFS With Hive and Cloudera Impala exercise, start the Impala Query Editor by selecting Query Editors > Impala.

Enter the following query to verify that Impala is working in Hue:

```
SHOW TABLES;
```

Click Execute. The result of the query should be the movierating table.

Enter another query to count the number of records in the movierating table:

```
SELECT COUNT(*) FROM movierating;
```

The query should run successfully.

Access the Metastore Using Hue

1. Start the Metastore Manager by selecting Data Browsers > Metastore Tables.

The Metastore Manager appears with an entry for the movierating table.

Select the entry for the movierating table.

The schema for the Hive movierating table, which you created in the Hive exercise, appears in the Metastore Manager.

Notice the list of actions available in the ACTIONS menu.

Access Pig Using Hue

1. Start the Pig UI by selecting Query Editors > Pig.

The Pig Query Editor appears.

You can edit and save Pig scripts using Hue's Pig Query Editor in your current Hue deployment.

Access HDFS Using Hue

1. Click on the File Browser (document icon) towards the top right of the Hue UI.

This opens the File Browser application.

Browse the HDFS file system. If you wish, execute some `hdfs dfs` commands from the command line to verify that you obtain the same results from the command line and the Hue File Browser.

Note the Upload menu as well. You could upload files to HDFS through Hue.

There are also Actions available giving the Hue user options to Rename, Move, Download or change permissions on HDFS files (assuming the user has the permissions to do so).

2. In the File Browser, navigate to the `/user/admin` directory.

On the first Hue login, Hue created a superuser – in this case, the `admin` user – and an HDFS path for that user – in this case, the `/user/admin` path.

3. In the File Browser, navigate to the `/user/training/test_output` directory – the output directory of the WordCount job that you ran before starting the Hue UI.
4. Click the entry for the `part-r-00000` file – the output file from the WordCount job.

A read-only editor showing the contents of the `part-r-00000` file appears.

Browse YARN Applications Using Hue

1. Select the Job Browser (list icon) option.

An entry for the Hive job that you ran earlier appears in the Hue Job Browser.

Specify `training` in the Username field.

An entry for the MapReduce word count job you ran in the previous step appears with the status “SUCCEEDED.”

Another entry for the Spark shell you should still have running appears with the status “RUNNING.”

Logs	ID	Name	Status	User	Maps	Reduces	Queue	Priority	Duration	Submitted
	1425355736296_0003	Spark shell	RUNNING	training	10%	20%	root.training	N/A	1m:14s	03/04/15 00:01:09
	1425355736296_0002	word count	SUCCEEDED	training	100%	100%	root.training	N/A	33s	03/04/15 00:00:35

Browse the completed “word count” job details by clicking on the link in the ID column and then looking through the details in the Attempts, Tasks, Metadata, and Counters tabs.

If you are interested, look in Cloudera Manager’s YARN Applications page for your cluster, locate the entry for the same word count job, and follow the “Application Details” link which takes you to the HistoryServer Web UI. Compare the details you find there with the information available in the Hue Job Browser.

2. Back in the terminal window on `elephant`, type `exit` to end the Spark interactive shell.

Browse Users, Documentation, Settings and Logs of Hue

1. Start the User Management Tool by selecting the "admin" menu (cog and wheels icon) and then choosing "Manage Users."

The User Admin screen appears where you can define Hue users and groups and set permissions.

Notice the automatically created entry for the `admin` user.

You will create another user and a group in the next task.

2. Click the Documentation (question mark) icon:

Hue UI user documentation appears.

3. Click the About Hue icon (to the left of the Home icon).

The Quick Start Wizard's Check Configuration tab shows "All OK. Configuration check passed."

Choose the About Hue top-level menu and click into the Configuration tab to examine Hue's configuration.

Click the Server Logs tab to examine Hue's logs.

Setting up the Hue Environment for Business Analysts

Consider a scenario where you have been given a requirement to set up a Hue environment for business analysts. The environment will allow analysts to submit Hive and Impala queries, edit and save Pig queries, browse HDFS, manage table definitions in the Hive Metastore, and browse Hadoop jobs. Analysts who have this environment will not need Hadoop installed on their systems. Instead, they will access all the Hadoop functionality that they need through a Web browser.

You will use the User Admin application to set up the analysts' Hue environment.

1. Verify that you are still logged in to Hue as the `admin` user.
2. Activate the Hue User Management tool by selecting admin > Manage Users.
3. Select Groups.

4. Click Add group, name the new group analysts.

Configure the permissions by selecting the ones listed below:

- about.access
- beeswax.access
- filebrowser.access
- hbase.access
- help.access
- impala.access
- jobbrowser.access
- metastore.write
- metastore.access
- pig.access

Click Add group.

5. Select Users.

6. Add a Hue user named fred with the password training.

In the “Step 2: Names and Groups” tab, make fred a member of the analysts group. However, make sure that fred is *not* a member of the default group.

Click “Add User”.

7. Sign out of the Hue UI (using the arrow icon in the top right of the screen).

8. Log back in to the Hue UI as user fred with password training.

Verify that in the session for fred, only the Hue applications configured for the analysts group appear. For example, the Administration menu does not allow fred to manage users. Fred also has no access to the Workflows, Search, and Security menus that are available to the admin user.

This is the end of the Exercise.

Hands-On Exercise: Configuring HDFS High Availability

In this exercise, you will reconfigure HDFS, eliminating the NameNode as a single point of failure for your Hadoop cluster.

You will start by modifying the `hdfs` service's configuration to enable HDFS high availability.

You will then shut down services that you will no longer use in this exercise or other subsequent exercises.

Next, you will enable automatic failover for the NameNode. Automatic failover uses the ZooKeeper service that you added to your cluster in an earlier exercise. ZooKeeper is a prerequisite for HDFS HA automatic failover.

Once you have enabled HDFS high availability with automatic failover, you will intentionally bring one of the servers down as a test. HDFS services should still be available, but it will be served by a NameNode running on a different host.

At the end of this exercise, you should have daemons deployed and running on your five hosts as follows (new daemons shown in blue):

					
	elephant	horse	monkey	tiger	lion
HDFS NameNode	✓				
HDFS SecondaryNameNode				removed	
HDFS Standby NameNode				✓	
JournalNode	✓	✓		✓	
Failover Controller	✓			✓	
HDFS DataNode	✓	✓	✓	✓	
HDFS Balancer		✓			
HDFS HttpFS			✓		
YARN (MR2 Included) NodeManager	✓	✓	✓	✓	
YARN (MR2 Included) ResourceManager		✓			
YARN (MR2 Included) JobHistory Server			✓		
ZooKeeper Server	✓	✓		✓	
Sqoop 1 Client Gateway	✓				
Flume Agent	stopped	stopped			
HiveServer2	stopped				
Hive Gateway	stopped				
Hive Metastore Server	stopped				
Impala Catalog Server		stopped			
Impala StateStore		stopped			
Impala Daemon	stopped	stopped	stopped	stopped	
Spark Gateway	stopped		stopped		
Spark History Server			stopped		
Hue Server			stopped		
Oozie Server			stopped		
Cloudera Manager Server					✓
Cloudera Manager Server Database					✓
Cloudera Management Services					✓
Cloudera Manager Agent	✓	✓	✓	✓	✓

IMPORTANT: This exercise builds on the previous one. If you were unable to complete the previous exercise or think you may have made a mistake, run the following command and follow the prompts to prepare for this exercise before continuing:

```
$ ~/training_materials/admin/scripts/reset_cluster.sh
```

Bringing Down Unneeded Services

Since you will no longer use Hue, Oozie, Impala, or Hive for the remaining exercises, you can stop their services to improve your cluster's performance.

1. In Cloudera Manager, navigate to the Home page.
2. Stop the `hue` service as follows:

In the row for the `hue` service, click Actions > Stop.

Click Stop in the confirmation window.

Click Close after messages in the Command Details: Stop page indicate that the `hue` service has stopped.

The Home page reappears. The status of the `hue` service should have changed to Stopped.

3. Using steps similar to the steps you followed to stop the `hue` service, stop the `oozie` service.
4. Stop the `impala` service.
5. Stop the `hive` service.
6. Stop the `flume` service if it is running.

Verify that the only services that are still up and running on your cluster are the hdfs, yarn, spark, zookeeper, and mgmt services. All of these services should have good health.

Enabling HDFS High Availability

In this task you will configure your Hadoop configuration to use HDFS high availability.

1. Configure directories for Journal Nodes to store edits directories.

On elephant:

```
$ sudo mkdir /dfs/jn
$ sudo chown hdfs:hadoop /dfs/jn
$ ssh training@horse sudo mkdir /dfs/jn
$ ssh training@horse sudo chown hdfs:hadoop /dfs/jn
$ ssh training@tiger sudo mkdir /dfs/jn
$ ssh training@tiger sudo chown hdfs:hadoop /dfs/jn
```

2. In Cloudera Manager, browse HDFS Configuration page for your cluster.

Select the Service-Wide category.

For ZooKeeper Service, click in the Value area and choose ZooKeeper.

Save Changes.

Now select the Instances tab.

The hdfs Role Instances page appears.

Observe that the hdfs service comprises the following role instances:

- Balancer on horse
- DataNodes on elephant, monkey, horse, and tiger
- The active NameNode on elephant
- The SecondaryNameNode on tiger
- An HttpFS server on monkey

The list of role instances will change after you enable high availability.

3. Click “Enable High Availability”.

The screenshot shows the 'Role Instances' section of the Cloudera Manager interface. On the right, there is a button labeled 'Enable High Availability' which is highlighted with a red box. The table below lists role instances categorized by type (Balancer, DataNode), state (N/A, Started), host (horse, tiger), and role group (Balancer Default Group, DataNode Default Group). There are 8 total instances listed.

Role Type	State	Host	Role Group
Balancer	N/A	horse	Balancer Default Group
DataNode	Started	horse	DataNode Default Group
DataNode	Started	tiger	DataNode Default Group

4. The “Getting Started” page appears.

Change the Nameservice Name to mycluster .

Click Continue.

5. The “Assign Roles” page appears.

Specify the following:

- NameNode Hosts
 - elephant (Current)
 - tiger
- JournalNode Hosts
 - elephant, horse, tiger

Click Continue.

6. The “Review Changes” page appears.

Specify the value **/dfs/jn** in all three **JournalNode Edits Directory** fields.

Enable High Availability for HDFS

Review Changes

Set the following configuration values for your new role(s). Required values are highlighted in red.

Parameter	Group	Value
Service HDFS		
NameNode Data Directories*	elephant	/dfs/nn
dfs.namenode.name.dir		Inherited from: NameNode Default Group
	tiger	/dfs/nn
		Inherited from: NameNode Default Group
JournalNode Edits Directory*		
dfs.journalnode.edits.dir	elephant	/dfs/jn
		Reset to empty default value
	horse	/dfs/jn
		Reset to empty default value
	tiger	/dfs/jn
		Reset to empty default value

Click Continue.

The “Progress” page appears. The messages shown in the screen shot below appear as Cloudera Manager enables HDFS high availability.

Note: Formatting the name directories of the current NameNode will fail. As described in the Cloudera Manager interface, this is expected.

✓ Stop hdfs and its dependent services
Successfully executed command Stop on service HDFS

✓ Creating roles to enable High Availability.
Successfully added new JournalNode to HDFS on horse.

✓ Deleting the SecondaryNameNode role. The checkpoint directories of the SecondaryNameNode will not be deleted.
Successfully deleted role.

✓ Configuring NameNodes and the HDFS service to enable High Availability.
Successfully updated config value.

✓ Check that name directories for the new Standby NameNode either do not exist or are writable and empty. Can optionally clear directories.
Confirmed that directory is empty and writable by an HDFS service.
[Details ↗](#)

✓ Check that edits directories for the nameservice either do not exist or are writable and empty. Can optionally clear directories.
Confirmed that directory is empty and writable by an HDFS service.
[Details ↗](#)

✓ Initializing High Availability state in ZooKeeper.
Successfully initialized High Availability state in ZooKeeper from failovercontroller (elephant)
[Details ↗](#)

✓ Starting the JournalNodes
Supervisor returned RUNNING
[Details ↗](#)

✗ Formating the name directories of the current NameNode. If the name directories are not empty, this is expected to fail.
Command (703) has failed
[Details ↗](#)

✓ Initializing shared edits directory of NameNodes.
Successfully initialized shared edits directory of the NameNode
[Details ↗](#)

✓ Starting the NameNode that will be transitioned to active mode namenode (elephant).
Supervisor returned RUNNING
[Details ↗](#)

✓ Waiting for the Active NameNode to start up.
NameNode started responding to RPCs successfully.
[Details ↗](#)

✓ Bootstrapping Standby NameNode by initializing its name directories.
Successfully bootstrapped Standby NameNode namenode (tiger).
[Details ↗](#)

✓ Starting Standby NameNode
Supervisor returned RUNNING
[Details ↗](#)

✓ Starting the Failover Controller on the host of the Active NameNode.
Supervisor returned RUNNING
[Details ↗](#)

✓ Starting the Failover Controller on the host of the Standby NameNode.
Supervisor returned RUNNING
[Details ↗](#)

✓ Waiting for the Standby NameNode to start up.
NameNode started responding to RPCs successfully.
[Details ↗](#)

✓ Creating HDFS /tmp directory if not already created.
HDFS directory /tmp already exists.
[Details ↗](#)

✓ Start hdfs and its dependent services
Successfully executed command Start on service Spark

✓ Deploying configurations for clients of services in this cluster.
Successfully deployed all client configurations.
[Details ↗](#)

[Back](#) 1 2 3 4 5 [Continue](#)

When the process of enabling high availability has finished, click Continue.

An informational message appears informing you of post-setup steps regarding the Hive Metastore. You will not perform the post-setup steps because you will not be using Hive for any remaining exercises.

Click Finish.

7. The `hdfs` Role Instances page appears.

Observe that the `hdfs` service now comprises the following role instances:

- Balancer on horse
- DataNodes on elephant, tiger, horse, and monkey
- Failover Controllers on elephant and tiger
- An HttpFS server on monkey
- JournalNodes on elephant, tiger, and horse
- The active NameNode on elephant
- The standby NameNode on tiger
- No SecondaryNameNode

Verifying Automatic NameNode Failover

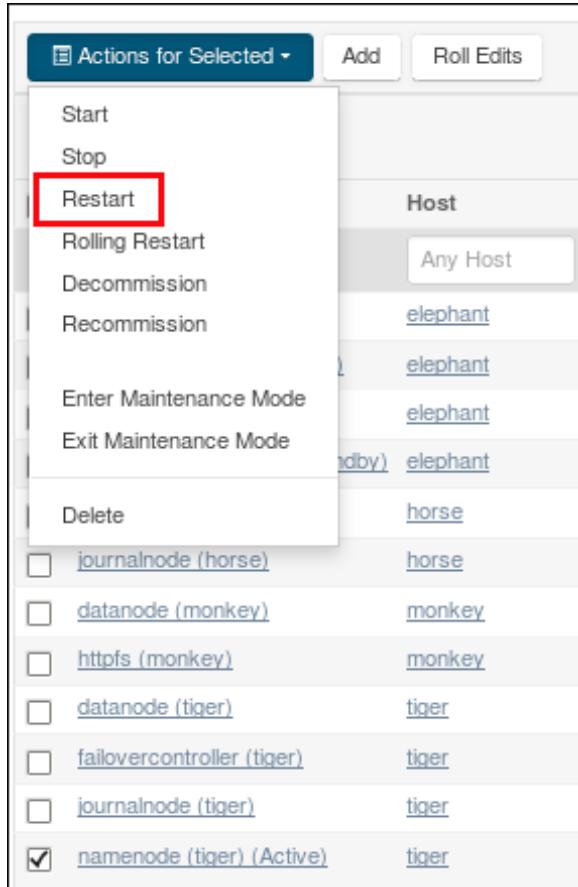
In this task, you will restart the active NameNode, bringing it down and then up. The standby NameNode will transition to the active state when the active NameNode goes down, and the formerly active NameNode will transition to the standby state.

Then you will restart the new active NameNode in order to restore the original states of the two NameNodes.

1. Navigate to the HDFS service's Instances tab.
2. In the "Federation and High Availability" section of the page, observe that the state of one of the NameNodes is active and the state of the other NameNode is standby.
3. Scroll down to the "Role Instances" section.
4. In the Role Instances section of the page, select the check box to the left of the entry for the active NameNode.

5. Click Actions for Selected > Restart.

Click Restart to confirm that you want to restart the instance.



6. Wait for the restart operation to complete. When it has successfully completed click Close.

Verify that the states of the NameNodes have changed—the NameNode that was originally active (elephant) is now the standby, and the NameNode that was originally the standby (tiger) is now active. If the Cloudera Manager UI does not immediately reflect this change, give it a few seconds and it will.

7. Go to Diagnostics > Events and notice the many recent event entries related to the restarting of the NameNode.
8. Back in the HDFS Instances tab, restart the NameNode that is currently the active NameNode.

After the restart has completed, verify that the states of the NameNodes have again changed.

This is the end of the Exercise.

Hands-On Exercise: Using the Fair Scheduler

In this exercise, you will submit some jobs to the cluster and observe the behavior of the Fair Scheduler.

IMPORTANT: This exercise builds on the previous one. If you were unable to complete the previous exercise or think you may have made a mistake, run the following command and follow the prompts to prepare for this exercise before continuing:

```
$ ~/training_materials/admin/scripts/reset_cluster.sh
```

1. Adjust YARN memory setting and restart the cluster.

In order to demonstrate the Fair Scheduler, you will need to increase the amount of memory that YARN containers are permitted to use on your cluster.

In Cloudera Manager, go to the YARN (MR2 Included) Configuration page.

In the search box, search for `yarn.nodemanager.resource.memory-mb`

Change the value of this parameter to 3GB in both of the Role Groups where it appears.

The screenshot shows the Cloudera Manager interface for the YARN service. The top navigation bar includes Status, Instances, Configuration (selected), Commands, Audits, Applications, and Charts Library. A green circular icon indicates the cluster is healthy. The Configuration tab is selected, displaying the 'Configuration' section. A search bar at the top right is set to 'yarn.nodemanager.resource.memory'. Below it are tabs for Role Groups, History and Rollback, Notes, and Save Changes (which is highlighted in blue). The main table lists configuration properties under two categories: 'NodeManager Default Group / Resource Management' and 'NodeManager Group 1 / Resource Management'. Both entries show the property 'Container Memory' (yarn.nodemanager.resource.memory-mb) with a value of '3 GiB'. Each entry has a 'Reset to the default value: 8' link and a 'GiB' unit indicator.

Category	Property	Value	Description
NodeManager Default Group / Resource Management	Container Memory yarn.nodemanager.resource.memory-mb	3 GiB	Amount of physical memory, in MiB, that can be allocated for containers. Reset to the default value: 8 GiB ↴
NodeManager Group 1 / Resource Management	Container Memory yarn.nodemanager.resource.memory-mb	3 GiB	Amount of physical memory, in MiB, that can be allocated for containers. Reset to the default value: 8 GiB ↴

Save the Changes.

Navigate to the YARN Status page and click on the “Stale Configuration: Restart Needed” icon.

Follow the steps to restart the cluster. When the restart has completed, click Finish.

2. Analyze the script you will run in this exercise.

To make it easier to start and stop MapReduce jobs during this exercise, a script has been provided. View the script to gain an understanding of what it does.

On **elephant**

```
$ cd ~/training_materials/admin/scripts
$ cat pools.sh
```

The script will start or stop a job in the pool you specify. It takes two parameters. The first is the pool name and the second is the action to take (start or stop). Each job it will run will be relatively long running and consist of 10 mappers and 10 reducers.

Note: Remember, we use the terms *pool* and *queue* interchangeably.

3. Start three Hadoop jobs, each in a different pool.

On elephant:

```
$ ./pools.sh pool1 start
$ ./pools.sh pool2 start
$ ./pools.sh pool3 start
```

It is recommended that you attempt to go through this exercise by only starting jobs when prompted in the instructions. However, depending on how quickly you complete the steps, a job may have completed earlier than the instructions anticipated. Therefore, please note that at any time during this exercise you can start additional jobs in any pool using any of the three commands you ran in this step.

4. Verify the jobs started.

In Cloudera Manager, browse to Clusters > YARN Applications.

You should notice that the three jobs have started. If the jobs do not yet display in the page, wait a moment and then refresh the page.

Note: Cloudera Manager does refresh pages automatically, however in this exercise you may find it useful to refresh the pages manually to more quickly observe the latest status of pools and jobs running in the pools.

Leave this browser tab open.

5. Observe the status of the pools.

Open another browser tab, and in Cloudera Manager, browse to Clusters > Dynamic Resource Pools.

Analyze the details in the “Resource Pools Usage” table.

If pool1, pool2, and pool3 do not yet display, refresh the browser tab.

The table displays the pools in the cluster. The pools you submitted jobs to should have pending containers and allocated containers.

The table also shows the amount of memory and vcores that have been allocated to each pool.

6. Analyze the Per Pool charts.

On the same page in Cloudera Manager, notice the Per Pool Shares charts. There is one chart for Fair Share Memory and another for Fair Share Vcores.

If nothing is displaying in these charts yet, wait a moment and they will. Optionally refresh the browser page.

Leave this browser tab open.

7. Start another job in another pool.

In the same shell session on elephant:

```
$ ./pools.sh pool4 start
```

8. Back in Cloudera Manager, observe the resource allocation affect of starting a new job in a new pool.

Occasionally refresh the Dynamic Resource Pools page.

Some pools may be initially over their fair share because the first jobs to run will take all available cluster resources.

However, over time, notice that the jobs running over fair share begin to shed resources, which are reallocated to other pools to approach fair share allocation for all pools.

Tip: Mouse over any one of the “Per Pool..” charts and then click the double-arrow icon to expand the chart size.

9. Conduct further experiments.

Stop the job running in pool1.

```
$ ./pools.sh pool1 stop
```

Wait a minute or two, then observe the results in the charts on the Dynamic Resource Pools page.

Start a second job in pool3.

```
$ ./pools.sh pool3 start
```

Again observe the results.

10. Configure a Dynamic Resource Pool for pool2.

In the Dynamic Resource Pools page, click into the Configuration tab and click on the Resource Pools tab.

Click on “Add Resource Pool”.

In the General tab, set the Resource Pool Name to pool2. Keep DRF as the scheduling policy.

In the YARN tab, configure the following settings:

- Weight: **2**
- Virtual Cores Min: **1**
- Virtual Cores Max: **2**
- Memory Min: **2400**
- Memory Max: **5000**

Click OK to save the changes.

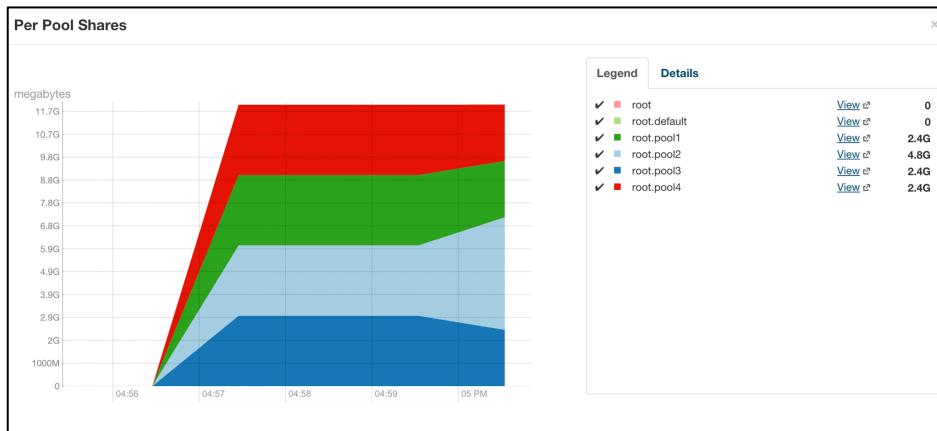
11. Observe the effect of the new Dynamic Resource Pool on the resource allocations within the cluster.

In the YARN applications page, check how many jobs are still running and in which pools they are running.

Use the pools.sh script to start or stop jobs so that there is one job running in each of the four pools.

Return to the Dynamic Resource Pools page's Status tab to observe the affect of the pool settings you defined.

As you continue to observe the Per-Pool Shares chart, you should soon see that pool2 is given a greater share of resources.



12. Clean up.

When you are done observing the behavior of the Fair Scheduler, stop all running jobs either by using the pools.sh script or kill the applications from the YARN Applications page in Cloudera Manager.

This is the end of the Exercise.

Hands-On Exercise: Breaking The Cluster

In this exercise, you will see what happens during failures of portions of the Hadoop cluster.

IMPORTANT: This exercise builds on the previous one. If you were unable to complete the previous exercise or think you may have made a mistake, run the following command and follow the prompts to prepare for this exercise before continuing:

```
$ ~/training_materials/admin/scripts/reset_cluster.sh
```

1. Verify the existence of a large file in HDFS.

In a previous exercise you placed the weblog files in HDFS. Verify the files are there.

On **elephant**:

```
$ hdfs dfs -ls weblog
```

Only if you do not see the access_log file in HDFS, place it there now.

On **elephant**:

```
$ cd ~/training_materials/admin/data
$ hdfs dfs -mkdir weblog
$ gunzip -c access_log.gz \
| hdfs dfs -put - weblog/access_log
```

2. Locate a block that has been replicated on **elephant** as follows:

In the NameNode Web UI, navigate to the /user/training/weblog/access_log file. The File Information window appears.

Locate the Availability section in the File Information window for Block 0. You should see three hosts on which Block 0 is available. If one of the replicas is on `elephant`, note its Block ID. You will need to refer to the Block ID in the next exercise.

If none of Block 0's replicas are on `elephant`, view the replication information for other blocks in the file until you locate a block that has been replicated on `elephant`. Once you have located a block that has been replicated on `elephant`, note its block ID.

We will revisit this block when the NameNode recognizes that one of the DataNodes is a 'dead node' (after 10 minutes).

3. Now, intentionally cause a failure and observe what happens.

Using Cloudera Manager, from the HDFS Instances page, stop the DataNode running on `elephant`.

4. Visit the NameNode Web UI again and click on 'Datanodes'. Refresh the browser several times and notice that the 'Last contact' value for the `elephant` DataNode keeps increasing.
5. Run the HDFS file system consistency check to see that the NameNode currently thinks there are no problems.

On `elephant`:

```
$ sudo -u hdfs hdfs fsck /
```

6. Wait for at least ten minutes to pass before starting the next exercise.

(optional) in any terminal:

```
$ sleep 600 && echo "10 minutes have passed."
```

This is the end of the Exercise.

Hands-On Exercise: Verifying The Cluster's Self-Healing Features

In this exercise, you will see what has happened to the data on the dead DataNode.

IMPORTANT: This exercise builds on the previous one. If you were unable to complete the previous exercise or think you may have made a mistake, run the following command and follow the prompts to prepare for this exercise before continuing:

```
$ ~/training_materials/admin/scripts/reset_cluster.sh
```

1. In the NameNode Web UI, click on Datanodes and confirm that you now have one ‘dead node.’
2. View the location of the block from the `access_log` file you investigated in the previous exercise. Notice that Hadoop has automatically re-replicated the data to another host to retain three-fold replication.
3. In Cloudera Manager, choose Clusters > Reports.

Click on “Custom report” at the bottom of the Disk Usage (HDFS) section.

Build a report with the following two filters (use the plus sign to add the second filter):

- Replication < 4
- Group equal to hadoop

Click “Generate Report”. You should see all files match the criteria since the replication factor for your cluster is set to three.

Change the Replication filter from < 4 to < 3 and run generate the report again.

Now there should be no files that match the criteria since data stored on elephant was replicated over to one of the other three DataNodes.

4. View charts that show when replication occurred.

From the Cloudera Manager HDFS page, choose Charts > Charts Library.

In the search box, type in `replication`. This will show you the “Pending Replication Blocks” and “Scheduled Replication Blocks” charts.

Note the spike in activity that occurred after the DataNode went down.

5. View the audit and log trails in Cloudera Manager.

In Cloudera Manager, click on Audits.

Note the timestamp for when the HDFS service was stopped.

Choose Diagnostics > Logs, select sources HDFS only, set the Minimum Log Level to “INFO”, and enter the search term “replicate”.

The screenshot shows the 'Logs' section of the Cloudera Manager interface. At the top, there is a search bar containing 'replicate' with a 'Search' button and an 'Additional Settings' link. Below the search bar are filter options: 'Select Sources' (dropdown), 'Hosts' (text input), 'Enter a host name' (text input), 'Minimum Log Level' (dropdown set to 'INFO'), and a dropdown for 'Additional Settings'. Under these filters, there is a checkbox for 'All Sources'. Below this, the 'Cluster 1' section lists several services with checkboxes: Spark, YARN (MR2 Included), flume, and hdfs (which is checked). To the right of these, under 'Cloudera Management Service', are checkboxes for hive, zookeeper, hue, impala, and oozie. Further to the right, under 'Cloudera Manager', is a checkbox for Cloudera Manager.

Click Search.

Scroll to the bottom. Notice the log messages related blocks being replicated.

6. Run the `hdfs fsck` command again to observe that the filesystem is still healthy.

On **elephant**:

```
$ sudo -u hdfs hdfs fsck /
```

7. Run the `hdfs dfsadmin -report` command to see that one dead DataNode is now reported.

On **elephant**:

```
$ sudo -u hdfs hdfs dfsadmin -report
```

8. Use Cloudera Manager to restart the DataNode on `elephant`, bringing your cluster back to full strength.
9. Run the `hdfs fsck` command again to observe the temporary over replication of blocks.

On `elephant`:

```
$ sudo -u hdfs hdfs fsck /
```

Note that the over replication situation will resolve itself (if it has not already) now that the previously unavailable DataNode is once again running.

If the command above did not show any over replicated blocks, go to Diagnostics > Logs in Cloudera Manager and search the HDFS source for "ExcessRepl". You should find evidence of the temporary over-replication in the log entries.

This is the end of the Exercise.

Hands-On Exercise: Taking HDFS Snapshots

In this exercise, you will enable HDFS snapshots on a directory and the practice restoring data from a snapshot.

IMPORTANT: This exercise builds on the previous one. If you were unable to complete the previous exercise or think you may have made a mistake, run the following command and follow the prompts to prepare for this exercise before continuing:

```
$ ~/training_materials/admin/scripts/reset_cluster.sh
```

1. Enable snapshots on a directory in HDFS.

In Cloudera Manager, go to the HDFS page for your cluster and click File Browser.

Browse to /user/training, then click Enable Snapshots.

In the “Enable Snapshots” windows, keep the Snapshotable Path set to /user/training and click Enable Snapshots.

The command completes. Notice in the message displayed on the “Program:” line that snapshots can also be enabled from the command line using the `hdfs dfsadmin` tool.

Click Close. Notice that there is now a “Take Snapshot” button.

2. Take a snapshot.

Still in the Cloudera Manager File Browser at /user/training, Click “Take Snapshot”. Give it the name snap1 and click OK.

After the snapshot completes click Close.

The snapshot section should now show your “snap1” listing.

3. Delete data from /user/training then restore data from the snapshot.

Now let's see what happens if we delete some data.

On **elephant**:

```
$ hdfs dfs -rm -r weblog  
$ hdfs dfs -ls /user/training
```

The second command should show that the weblog directory is now gone.

However you weblog data is still available, which you can see by running the commands here:

```
$ hdfs dfs -ls /user/training/.snapshot/snap1  
$ hdfs dfs -tail .snapshot/snap1/weblog/access_log
```

Restore a copy of the weblog directory to the original location and then verify it is back in place.

```
$ hdfs dfs -cp .snapshot/snap1/weblog weblog  
$ hdfs dfs -ls /user/training
```

This is the end of the Exercise.

Hands-On Exercise: Configuring Email Alerts

In this exercise, you will configure Cloudera Manager to use an email server to send alerts.

IMPORTANT: This exercise builds on the previous one. If you were unable to complete the previous exercise or think you may have made a mistake, run the following command and follow the prompts to prepare for this exercise before continuing:

```
$ ~/training_materials/admin/scripts/reset_cluster.sh
```

1. Configure Cloudera Manager to send email alerts using the email server on lion.

In Cloudera Manager, choose Clusters > Cloudera Management Service.

Click on Configuration and then choose “Alert Publisher Default Group”.

Confirm the “Alerts: Enable Email Alerts” property is checked.

Configure the following:

- Alerts: Mail Server Username: **training**
- Alerts: Mail Server Password: **training**
- Alerts: Mail Message Recipients: **training@localhost**
- Alerts: Mail Message Format: **text**

Save the changes.

2. Restart the Cloudera Management Service.
3. Send a test alert from Cloudera Manager.

In Cloudera Manager, go to Administration > Alerts. You should see that the recipient(s) of alerts is now set to `training@localhost`.

Click on the “Send Test Alert” button at the top of the page.

4. Confirm emails are being received from Cloudera Manager.

The postfix email server is running on `lion`. Here you use the mail command line client to access the training user's inbox.

On `lion`:

```
$ mail
```

The "Test Alert" email should show as unread (U).

At the & prompt, type in the number that appears to the right of the U and hit the <Enter> key so you can read the email.

After you are done reading the email, type `q` <Enter> to exit the mail client.

This is the end of the Exercise.

Troubleshooting Challenge: Heap O' Trouble

It's 8:30 AM and you are enjoying your first cup of coffee. Keri, who is making the transition from writing RDBMS stored procedures to coding Java MapReduce, shows up in your doorway before you're even halfway through that first cup.

"I just tried to run a MapReduce job and I got an out of memory exception. I heard that there was 32GB on those new machines you bought. But when I run this stupid job, I keep getting out of memory errors. Isn't 32GB enough memory? If I don't fix this thing, I'm going to be in a heap of trouble. I told my manager I was 99% complete with my project but now I'm not even sure if I can do what I want to do with Hadoop."

Put down your coffee and see if you can help Keri get her job running.

IMPORTANT: This exercise builds on the previous one. If you were unable to complete the previous exercise or think you may have made a mistake, run the following command and follow the prompts to prepare for this exercise before continuing:

```
$ ~/training_materials/admin/scripts/reset_cluster.sh
```

Recreating the Problem

1. Confirm files in HDFS.

On **elephant**:

```
$ hdfs dfs -ls /tmp/shakespeare.txt
```

The command above should show that `shakespeare.txt` is in HDFS.

Only if `shakespeare.txt` was not found, run these commands to place the file in HDFS.

On **elephant**:

```
$ cd ~/training_materials/admin/data
$ gunzip shakespeare.txt.gz
$ hdfs dfs -put shakespeare.txt /tmp
```

On **elephant**:

```
$ hdfs dfs -ls weblog/access_log
```

The command above should confirm that the access_log file exists.

Only if access_log was not found, run these commands to place the file in HDFS.

On **elephant**:

```
$ cd ~/training_materials/admin/data
$ hdfs dfs -mkdir weblog
$ gunzip -c access_log.gz \
| hdfs dfs -put - weblog/access_log
```

2. Run the Heap of Trouble program.

on **elephant**:

```
$ cd ~/training_materials/admin/java
$ hadoop jar EvilJobs.jar HeapOfTrouble \
/tmp/shakespeare.txt heapOfTrouble
```

Attacking the Problem

The primary goal of this and all the other troubleshooting exercises is to start to become more comfortable analyzing problem scenarios by using Hadoop's log files and Web UIs. **Although you might be able to determine the source of the problem and fix it, doing so successfully is not the primary goal here.**

Take as many actions as you can think of to troubleshoot this problem. Please write down the actions that you take while performing this challenge so that you can share them with other members of the class when you discuss this exercise later.

Fix the problem if you are able to.

Do not turn to the next page unless you are ready for some hints.

Some Questions to Ask While Troubleshooting a Problem

This list of questions provides some steps that you could follow while troubleshooting a Hadoop problem. All of the steps do not necessarily apply to all Hadoop issues, but this list is a good place to start.

- What is there that is different in the environment that was not there before the problem started occurring?
- Is there a pattern to the failure? Is it repeatable?
- If a specific job seems to be the cause of the problem, locate the task logs for the job, including the ApplicationMaster logs, and review them. Does anything stand out?
- Are there any unexpected messages in the NameNode, ResourceManager, and NodeManager logs?
- How is the health of your cluster?
 - Is there adequate disk space?
 - More specifically, does the `/var/log` directory have adequate disk space?
 - Might this be a swapping issue?
 - Is network utilization extraordinarily high?
 - Is CPU utilization extraordinarily high?
- Can you correlate this event with any of the issues?
- If it seems like a Hadoop MapReduce job is the cause of the problem, is it possible to get the source code for the job?
- Does searching the Web for the error provide any useful hints?

Fixing the Problem

If you have time and are able to, fix the problem so that Keri can run her job.

Post-Exercise Discussion

After some time has passed, your instructor will ask you to stop troubleshooting and will lead the class in a discussion of troubleshooting techniques.

This is the end of the Exercise.

Appendix A: Setting up VMware Fusion on a Mac for the Cloud Training Environment

When performing the Hands-On Exercises for this course, you use a small CentOS virtual machine called Get2EC2. This VM is configured to use NAT networking. You connect to Amazon EC2 instances from the guest OS by starting SSH sessions. The Get2EC2 VM is supported for VMware or VirtualBox.

VMware Fusion, like other hypervisors, runs an internal DHCP server for NAT-ed guests that assigns IP addresses to the guests. From time to time, the internal DHCP server releases and renews the guests' leases. Unfortunately, the internal DHCP server in VMware Fusion does not always assign the same IP address to a guest that it had prior to the release and renew, and the Get2EC2 VM's IP address changes.

Changing the IP address results in problems for active SSH sessions. Sometimes the terminal window in which the client is running will freeze up, becoming unresponsive to mouse and keyboard input, and no longer displaying standard output. At other times, sessions will be shut down with a Broken Pipe error. If this happens, you will have to re-open any failed sessions.

If you are using VMware Fusion on a Mac to perform the Hands-On Exercises for this course, you need to decide whether you would prefer to take action or do nothing:

- If you have administrator privileges on your Mac, you can configure VMware Fusion to use a fixed IP address. The instructions for configuring VMware Fusion to use a fixed IP address appear below.
- If you have VirtualBox installed, you can use the VirtualBox Get2EC2 VM instead of VMware Fusion.
- You can do nothing, in which case you might encounter terminal freezes as described above.

To configure VMware Fusion to use a fixed IP address, perform the following steps:

1. Start VMware Fusion.
2. Create an entry in the Virtual Machines list for the Get2EC2 VM. To create the entry, drag the Cloudera-Training-Get2EC2-VM-1.0.vmx file to the Virtual Machines list.

You should see the Cloudera-Training-Get2EC2-VM-1.0 entry in the Virtual Machines list. We will refer to the Cloudera-Training-Get2EC2-VM-1.0 VM as the Get2EC2 VM.

3. Make sure the Get2EC2 VM is powered down.
4. Click once on the Get2EC2 VM entry in the Virtual Machines list to select the VM.

Note: If you accidentally double-click the entry, you start the VM. Before you proceed to the next step, power down the VM.

5. Click the Settings icon in the VMware Fusion Toolbar (or select Virtual Machines > Settings).
6. Click Network Adapter.
7. Click Advanced Options.

The MAC Address field appears.

8. If the MAC Address field is empty, click Generate to generate a MAC address for the Get2EC2 VM.
9. Copy the MAC address and paste it into a file where you can access it later. You will need to use the MAC address in a subsequent step.

10. Open the following file on your Mac using superuser (`sudo`) privileges:

- VMware Fusion 4 and higher: `/Library/Preferences/VMware Fusion/vmnet8/dhcpd.conf`
- VMware Fusion 3: `/Library/Application Support/VMware Fusion/vmnet8/dhcpd.conf`

Look for the `range` statement. It should have a range of IP addresses. For example:

```
range 172.16.73.128 172.16.73.254;
```

11. Choose an IP address for the Get2EC2 VM. The IP address should have the first three tuples of the IP addresses in the `range` statement, but the fourth tuple should be outside of the addresses in the `range` statement. Given the example of the `range` statement in the previous step, you would choose an IP address that starts with 172.16.73 and ends with a number lower than 128 (but not 0, 1, or 2 – those numbers are reserved for other purposes).

For example, 172.16.73.10.

12. Add four lines to the bottom of the `dhcpd.conf` file as follows:

```
host Get2EC2 {
    hardware ethernet <MAC_Address>;
    fixed-address <IP_Address>;
}
```

Replace `<MAC_Address>` with the MAC address you generated in an earlier step.

Replace `<IP_Address>` with the IP address you chose in the previous step.

Be sure to include the semicolons after the MAC and IP addresses as shown in the example.

13. Save and close the `dhcpd.conf` file.

- 14.** Run the following commands from a terminal window on your Mac:

For VMware Fusion 4 or higher:

```
$ sudo /Applications/VMware\ Fusion.app/Contents/\
Library/vmnet-cli --stop
$ sudo /Applications/VMware\ Fusion.app/Contents/\
Library/vmnet-cli --start
```

For VMware Fusion 3:

```
$ sudo /Library/Application Support/VMware\ Fusion/\
boot.sh --restart
```

- 15.** Start the Get2EC2 VM.

- 16.** After the VM has come up, run the following command in a Linux terminal window:

```
$ ip addr
```

Verify that the IP address that appears is the IP address that you specified in the `dhcpd.conf` file.

This is the end of this Appendix.

Appendix B: Setting up VirtualBox for the Cloud Training Environment

Follow these steps to set up VirtualBox for the cloud training environment if you do not want to install VMware Fusion on your Mac.

VMware Fusion is our preferred hypervisor for students running this course on Mac OS. Please use VMware Fusion if possible. Use VirtualBox for this course *only* if it is your preferred virtualization environment and if you are knowledgeable enough to be self-sufficient to troubleshoot problems you might run into.

This setup activity comprises:

- Creating the GetEC2 VM
- Powering up the VM
- Installing VirtualBox Guest Additions on the VM

This setup requires VirtualBox version 4 or higher.

1. Get the .vmdk file for the class from your instructor and copy it onto the system on which you will be doing the Hands-On Exercises.
2. Start VirtualBox.
3. Select Machine > New.

The Name and Operating System dialog box appears.

4. In the Name and Operating System dialog box, specify Get2EC2 as the Name, Linux as the Type, and Red Hat (*not* Red Hat 64-bit) as the Version. Click Continue.

The Memory Size dialog box appears.

5. In the Memory Size dialog box, accept the default memory size of 512 MB and click Continue.

The Hard Drive dialog box appears.

6. In the Hard Drive dialog box, select “Use an existing virtual hard drive file.”
7. In the field under the “Use an existing virtual hard drive file” selection, navigate to the .vmdk file for the class and click Open.
8. Click Create.

The Oracle VM VirtualBox Manager dialog box appears. The Get2EC2 VM appears on the left side of the dialog box, with the status Powered Off.

9. Click Start to start the Get2EC2 VM.

The VM starts up. After startup is complete, the GNOME interface appears. You are automatically logged in as the `training` user.

Now you are ready to install VirtualBox Guest Additions on the VM.

Note: The version of VirtualBox and Guest Additions must be the same. You must install Guest Additions now to guarantee compatibility between your version of VirtualBox and Guest Additions.

10. Select Devices > Install Guest Additions.

After several seconds, a dialog box appears prompting you to select how you want to install the version of VBOXADDITIONS on your system. Verify that Open Autorun Prompt is selected as the Action, then click OK.

11. Another dialog box appears prompting you to confirm you want to run the Guest Additions installer. Click Run.
12. The Authenticate dialog box appears, prompting you to enter the `root` user’s password. Specify `training` and click Authenticate.
13. Messages appear in the terminal window while VirtualBox is building and installing the Guest Additions.

When installation is complete, the message, “Press Return to close this window” appears in the terminal window.

14. Press Return.

15. Select System > “Log Out training” to log out of your GNOME session.

After you have logged out, you are automatically logged back in as the training user.

You have completed the VirtualBox setup. Please return to the next step in “Configuring Networking on Your Cluster: Cloud Training Environment” and continue the setup activity for the cloud training environment.

This is the end of this Appendix.