

Hands-On Hadoop and AWS

Self-paced exercises

by Leo Heska
June 5, 2013

Table of Contents

Hands-On Hadoop and AWS	1
Table of Contents	2
Introduction	1
Section 1: Logging on to AWS and roaming around	2
Section 2: Elastic MapReduce Jobs	3
Section 2.5 (for Power Users): Starting and Terminating Elastic MapReduce Jobs	6
Section 2.5.1: Starting a job for Pig	11
Section 3: Connecting to your cluster	12
Section 4: Running Hive queries from your Hive job	16
Section 5: Running Pig commands from your Pig job	19
Section 6 (for Power Users): Running and using R	23
Section 7 (for Power Users): Terminating your Hadoop jobs and cluster	24
Section 8 (for Power Users): Moving data using MapR NFS mount	25
Appendix 1 - Advanced/additional exercises and topics	31

Introduction

This document is not intended to be an introduction to Hadoop; see elsewhere for explanations of Hadoop's architecture, history, features, and current and potential usage. The purpose of this document is to guide you through a series of hands-on exercises that will accustom you to working with Hadoop as it is implemented in and supported by Amazon Web Services (AWS).

When you've finished these exercises you will have a good fundamental understanding of:

1. The most important beginning elements of working with, in, and through AWS.
2. The basics of Apache Hive.
3. The basics of Apache Pig.

You may also get some understanding of:

4. Running R with Hadoop.
5. Managing AWS Hadoop clusters, the various choices, and their effects and costs.
6. The MapR Hadoop distribution and usage of its NFS mount feature.

Certain sections of this document can't be done without elevated privileges - these sections are marked "for Power Users." If you want to do one of these, for now, the best way is to find a "power user" and get help from or partner with them.

Section 1: Logging on to AWS and roaming around

Go here:

<https://clientH.signin.aws.amazon.com/console>

to sign in using your [client] corp logon ID and your initial password, which is **[client]1_**

To change your password, click the down arrow by {yourcorpid} @ [client] in the upper right - it looks like this:



then choose "Security Credentials," then enter a new password that satisfies the rules:

- 8 characters minimum
- at least one uppercase letter
- at least one lowercase letter
- at least one number
- at least one non-alphanumeric character

Whether or not you have just changed your password, the screen you are looking at is the AWS management console. It can look different depending on what you are doing; also, certain options do or do not appear, depending on your role. For example, non-administrators cannot review, purge, and/or add user accounts, so you may not see the AIM (AWS Identity and access Management) module in your set of options.

Hunt and click and read around if you want to, but beware: AWS has a lot of parts and you can spend hours (at least) doing this. In fact a person could pretty easily spend a year learning all of what AWS has to offer.

When you are ready, click "Services" in the upper left corner.

Among other choices, you should see "S3" which stands for Amazon's Simple Storage Service. This is Amazon's cloud storage and this is (generally) where data reside, that we will work with and on.

Click on S3 and you should see a window titled "All Buckets" and within it, several folders. One is named

HOH

and if you click it, you'll find subfolders, each containing text files named like their parent folders (but in case you are wondering, the text files don't need to have the same names as the folders that contain them). These files contain information about the nutrient content of foods, courtesy of the United States Department of Agriculture (USDA). In the following directory (directly under "All Buckets"):

APTrainingResources

there is a document (CompositionOfFoods.pdf) that tells you what each of these files means, and all about the data themselves.

Try downloading the PDF document to your machine. Then open it and have a look inside. If you do, you'll probably feel right at home; this document is written in language that only ~~geeks~~ statisticians and analysts could love. Plenty of arcane terminology in there to chew on.

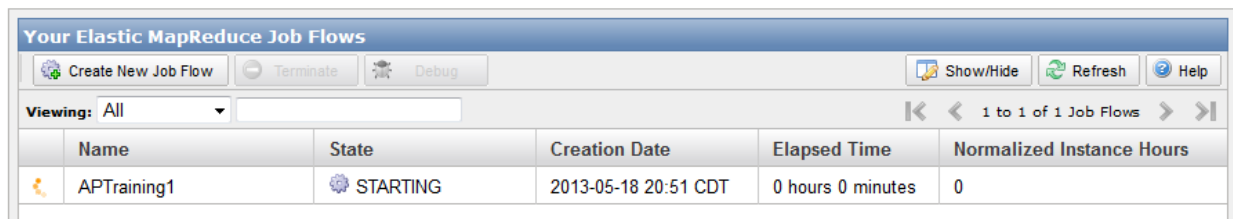
You can experiment with creating directories and uploading information to them but please don't delete or change the existing folders or anything in them. We need them for further exercises.

Section 2: Elastic MapReduce Jobs

If you again click "Services" in the upper left corner of the AWS console you should see an option for Elastic MapReduce. This is Amazon's name for their "big data" cloud-based PaaS (Platform as a Service) that actually does work. By "elastic" Amazon simply means that you rent (or "spin up") only as many machines as you need, for only as long as you need them. [client]'s not bound to pay for a fixed number of processors for a fixed time period.

Don't worry if you don't understand what MapReduce means - it's a misbegotten piece of terminology anyway and not needed to make things work and get things done. I'll explain it to you some time.

Click on Elastic MapReduce and you should see a screen showing some old and current job flows. You will also see a button to allow you Create New Job Flow, but if you try it, you'll find out about halfway through the process that your permissions are insufficient.



The screenshot shows the 'Your Elastic MapReduce Job Flows' interface. At the top, there are buttons for 'Create New Job Flow', 'Terminate', and 'Debug'. On the right, there are buttons for 'Show/Hide', 'Refresh', and 'Help'. Below these is a 'Viewing: All' dropdown and a search bar. A table lists the job flows with columns: Name, State, Creation Date, Elapsed Time, and Normalized Instance Hours. One job flow is listed: 'APTraining1' with state 'STARTING', creation date '2013-05-18 20:51 CDT', elapsed time '0 hours 0 minutes', and normalized instance hours '0'.

Name	State	Creation Date	Elapsed Time	Normalized Instance Hours
APTraining1	STARTING	2013-05-18 20:51 CDT	0 hours 0 minutes	0

You may also get a message about "multiple job flows selected;" you can just ignore it.

The term "Job Flow" is a little bit confusing for we who are going to run interactive commands. We'll be acting like developers, and we don't much care about flows. But, once we write code that works, that gets turned into a production job. Our code typically runs before, after, and/or along with other things; thus the term "job flow."

Some (two or perhaps more) of the Job Flows will be "your" jobs - the ones you will use to do the rest of the hands-on exercises. Find your jobs name and click on their names to find out their details.

The status of your jobs should be "WAITING" but if somebody just started your jobs you, first you need to wait for several (meaning 5 to 10) minutes for everything to really get going, and the job statuses to change from "STARTING" to "RUNNING" (at which point you're still not ready to go) and then, finally, to "WAITING."

If you do look at a job's details while its status is "STARTING" you won't see much. The job hasn't really started yet, so there isn't a lot to report:

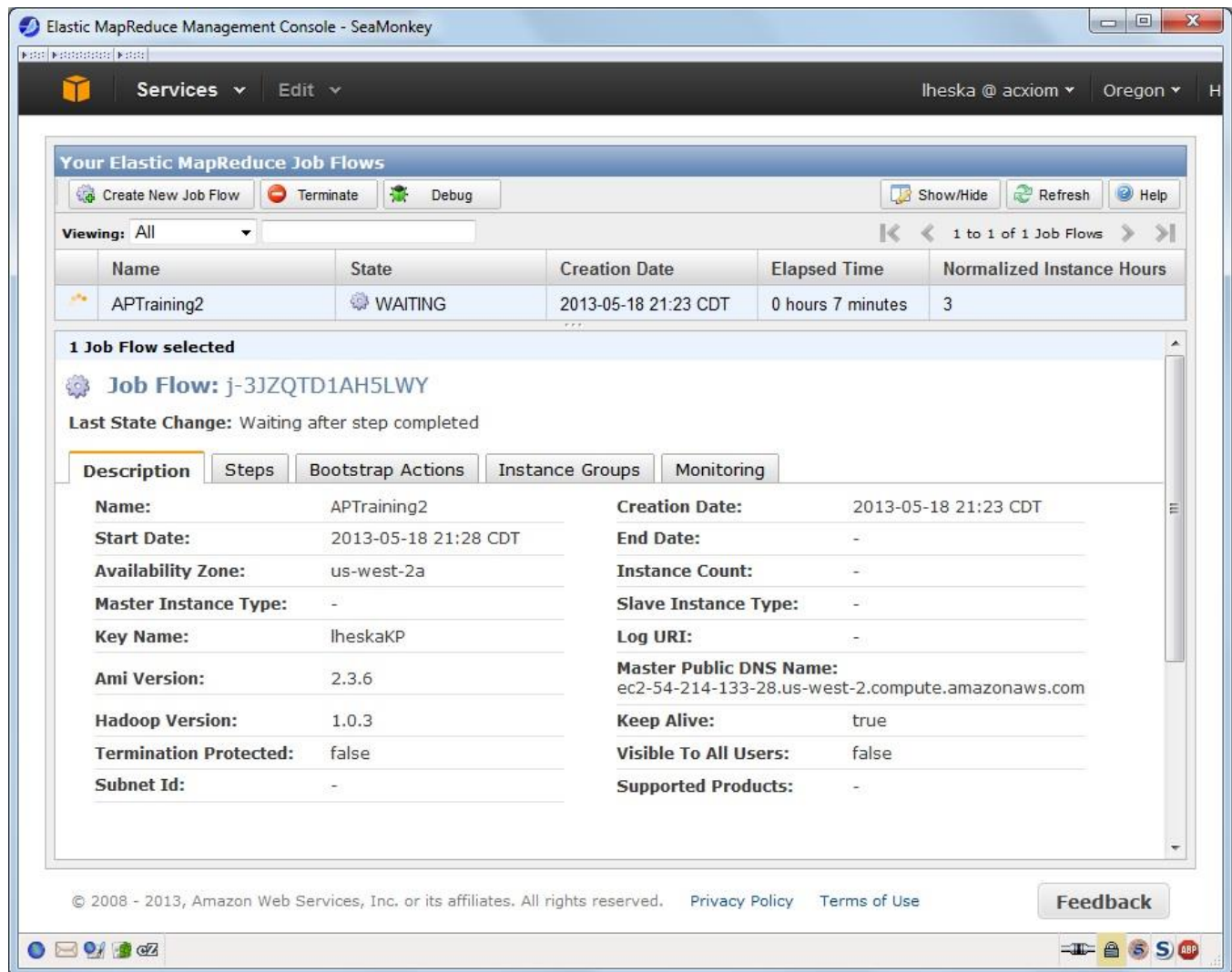
The screenshot shows the Elastic MapReduce Management Console interface. At the top, there's a navigation bar with 'Services' and 'Edit' menus, and a user profile 'lheska @ axiom' in 'Oregon'. The main section is titled 'Your Elastic MapReduce Job Flows'. It includes buttons for 'Create New Job Flow', 'Terminate', and 'Debug', along with 'Show/Hide', 'Refresh', and 'Help' options. A table lists three job flows: 'APTraining3' (STARTING), 'APTraining2' (TERMINATED), and 'APTraining1' (FAILED). The 'APTraining3' job is selected, and its details are shown below. The 'Last State Change' is 'Starting instances'. The 'Description' tab is active, displaying various configuration parameters for the job flow.

Name	State	Creation Date	Elapsed Time	Normalized Instance Hours
APTraining3	STARTING	2013-05-20 17:32 CDT	0 hours 0 minutes	0
APTraining2	TERMINATED	2013-05-18 21:23 CDT	0 hours 60 minutes	3
APTraining1	FAILED	2013-05-18 20:51 CDT	0 hours 0 minutes	0

1 Job Flow selected	
Job Flow: j-2EIF8D18640PP	
Last State Change: Starting instances	
Description	Steps
Name:	APTraining3
Start Date:	-
Availability Zone:	us-west-2a
Master Instance Type:	-
Key Name:	lheskaKP
Ami Version:	2.3.6
Hadoop Version:	1.0.3
Termination Protected:	false
Subnet Id:	-
Creation Date:	2013-05-20 17:32 CDT
End Date:	-
Instance Count:	-
Slave Instance Type:	-
Log URI:	-
Master Public DNS Name:	-
Keep Alive:	true
Visible To All Users:	false
Supported Products:	-

© 2008 - 2013, Amazon Web Services, Inc. or its affiliates. All rights reserved. [Privacy Policy](#) [Terms of Use](#) [Feedback](#)

Once the job status has changed to "WAITING," you can get what you need to proceed. Specifically, the contents of the Master Public DNS Name field. This field is in the Description tab, but it can be hidden. To find it, you may need to enlarge your browser window and/or scroll down within the Amazon "Your Elastic MapReduce Job Flows" window. Once you do, your screen should look something like this:



When I took this screenshot, the Master Public DNS Name was ec2-54-214-133-28.us-west-2.compute.amazonaws.com. Your Master Public DNS Name will be different.

You should have (at least) two different jobs, one for the Hive work you'll be doing, one for Pig.

The Master Public DNS Names of these two jobs are important. You should copy-and-paste them somewhere because you will need them to connect to your Hadoop clusters.

Section 2.5 (for Power Users): Starting and Terminating Elastic MapReduce Jobs

Note: Not everyone will be able to carry out the tasks in this subsection. The privileges that are granted to enable this section, also enable users to spin up very large and long-running Hadoop clusters and jobs, that could potentially cost [client] a lot of money.

If you again click "Services" in the upper left corner of the AWS console you should see an option for Elastic MapReduce. This is Amazon's name for their "big data" cloud-based PaaS (Platform as a Service) that actually does work. By "elastic" Amazon simply means that you rent (or "spin up") only as many machines as you need, for only as long as you need them. [client]'s not bound to pay for a fixed number of processors.

Don't worry if you don't understand what MapReduce means - it's a misbegotten piece of terminology anyway and not immediately critical to making things work and getting things done. I'll explain it to you some time.

Click on Elastic MapReduce and you should be presented with an option to Create New Job Flow.

Click Create New Job Flow and you should see something that looks like this:

Notice that you have your choice of five different types of jobs:

- Hive is a component of Hadoop that treats Hadoop/AWS data like they are stored in a relational database. To use Hive you write queries in the Hive Query Language (HQL), which is an SQL variant.
- Pig is a component of Hadoop that is good for record processing, such as working through billions of lines of a data file. Pig's language is called "Pig Latin" and though it is a relational language like SQL, it is also procedural, contrary to SQL.

- HBase is another Hadoop component that allows relational database-like functionality; we will not use HBase in this workshop.
- Custom JARs (=Java ARchives) are more or less what they sound like. If none of the provided features of Hadoop, nor any Hadoop add-ons, will do the particular job you want, then you can write your own Java code and use Hadoop/AWS to run it. Or you can obtain a JAR to do what you want from someone else.
- Running a "Streaming" job is basically just like running a Custom Jar, but the code is written in a different language. Amazon currently supports Ruby, Perl, Python, PHP, R, Bash, and C++. We may or may not get a chance to try running an R job in AWS during this workshop.

Now I am going to give you some instructions for what to do next. They are as correct as I can make them but you should not expect that every click and action is contained in this document. The time it takes to spin up AWS instances/Hadoop clusters can vary. Some screens and steps can sometimes flash by so quickly you barely notice them; other times, they sit there for minutes. Sometimes you will see a little orange "whirly circle" - other times you won't. Also, sometimes I have omitted mention of certain tasks, such as "Click the CLOSE button."

From the "Create a New Job Flow" screen, fill in the fields/make choices as follows:

Job Flow Name: {Pick something that's neither too long, cryptic, or offensive}

Hadoop Version: **Amazon Distribution** (default)

Make sure that the **Run your own application** radio button is checked.

Choose **Hive Program** from the drop-down list box.

Click **Continue**.

Your "Create a New Job Flow" screen should change to say "SPECIFY PARAMETERS." Ignore the top of the screen for now and click the **Start an Interactive Hive Session** radio button. It's about halfway down the page.

Click **Continue**.

Your "Create a New Job Flow" screen should change to say "CONFIGURE EC2 INSTANCES." The defaults (minimum of everything) should be what you want. However, you may double-check:

Create a New Job Flow Cancel X

DEFINE JOB FLOW SPECIFY PARAMETERS **CONFIGURE EC2 INSTANCES** ADVANCED OPTIONS BOOTSTRAP ACTIONS REVIEW

Specify the master, core and task nodes to run your job flow. For more than 20 instances, complete the [limit request form](#).

Master Instance Group: This EC2 instance assigns Hadoop tasks to core and task nodes and monitors their status.

Instance Type: Small (m1.small) ☐ Request Spot Instance

Core Instance Group: These EC2 instances run Hadoop tasks and store data using the Hadoop Distributed File System (HDFS). Recommended for capacity needed for the life of your job flow.

Instance Count: 2
Instance Type: Small (m1.small) ☐ Request Spot Instances

Task Instance Group (Optional): These EC2 instances run Hadoop tasks, but do not persist data. Recommended for capacity needed on a temporary basis.

Instance Count: 0
Instance Type: Small (m1.small) ☐ Request Spot Instances

[< Back](#) [Continue](#) * Required field

Once your settings match those above, click **Continue**.

Your "Create a New Job Flow" screen should change to say "ADVANCED OPTIONS." The defaults are all fine except one: you have to choose an Amazon EC2 Key Pair. From the drop-down list box, choose the key pair that matches your corp logon (with the letters KP appended). Then click "Continue."

Your "Create a New Job Flow" screen should change to say "BOOTSTRAP ACTIONS." Quoting Amazon:

Bootstrap actions are scripts that are run on the cluster nodes when Amazon EMR launches the cluster. They run before Hadoop starts and before the node begins processing data.

For now we don't need to set up anything to run before Hadoop starts, so just click **Continue**.

Your "Create a New Job Flow" screen should change one last time to say "REVIEW." As you can see, you can edit the various settings you have made so far. You can take a screen shot if you want, for record-keeping or documentation purposes.

Don't worry too much about the note saying you're going to get charged. The amount of work we will be doing falls within Amazon's "free tier" (see [here http://aws.amazon.com/free/faqs/](http://aws.amazon.com/free/faqs/) if you're curious as to the limits).

Once you're done reviewing, click **Create Job Flow**.

You should now see a screen that looks like this. You should read it - it's important:

Create a New Job Flow

Cancel X

✓ Your Job Flow has been created.

Your new job flow will take a few minutes to launch. When its state becomes WAITING you will be able to get the Master Public DNS Name from the job flow details pane. To run your interactive hive session, SSH to this machine as the user "hadoop" and then type 'hive'.

Note that this job flow must be manually terminated when you have finished with it. This can be done by clicking on the job flow in the grid and then clicking the 'Terminate' button.

> [View my job flows and check on job flow status](#)

You may click the link "View my job flows and check on job flow status." One of the important things the above says, is that it can take a little time for AWS to set up your Hadoop cluster and get things ready to go. While that is happening you may get a rather uninformative screen that you've already seen, that makes it look like you've got to start all over again, creating a new job flow. You don't. Just keep refreshing the screen and eventually (hopefully) you will get a screen that looks like this:

Your Elastic MapReduce Job Flows					
Create New Job Flow		Terminate	Debug	Show/Hide	Refresh Help
Viewing: All		1 to 1 of 1 Job Flows			
Name	State	Creation Date	Elapsed Time	Normalized Instance Hours	
APTraining1	STARTING	2013-05-18 20:51 CDT	0 hours 0 minutes	0	

You may or may not also get a message about "multiple job flows selected." Click on your job to see its status and details.

It can take several (meaning 5 to 10) minutes for everything to really get going, and the job status to change from "STARTING" to "RUNNING" (at which point you're still not ready to go) and then, to "WAITING."

If you have a look at the job screen while the job's status is "STARTING" - you'll see that it's mostly empty. The job hasn't started yet, there isn't a lot to report:

The screenshot shows the Elastic MapReduce Management Console interface. At the top, there's a navigation bar with 'Services' and 'Edit' menus, and a user profile 'lheska @ axiom' in 'Oregon'. The main section is titled 'Your Elastic MapReduce Job Flows'. It includes buttons for 'Create New Job Flow', 'Terminate', and 'Debug', along with 'Show/Hide', 'Refresh', and 'Help' options. A 'Viewing: All' dropdown and a pagination indicator '1 to 2 of 2 Job Flows' are also present.

	Name	State	Creation Date	Elapsed Time	Normalized Instance Hours
	APTraining3	STARTING	2013-05-20 17:32 CDT	0 hours 0 minutes	0
	APTraining2	TERMINATED	2013-05-18 21:23 CDT	0 hours 60 minutes	3
	APTraining1	FAILED	2013-05-18 20:51 CDT	0 hours 0 minutes	0

Below the table, a section titled '1 Job Flow selected' shows details for 'Job Flow: j-2EIF8D18640PP'. The 'Last State Change' is 'Starting instances'. There are tabs for 'Description', 'Steps', 'Bootstrap Actions', 'Instance Groups', and 'Monitoring'. The 'Description' tab is active, displaying a list of job flow properties:

Name:	APTraining3	Creation Date:	2013-05-20 17:32 CDT
Start Date:	-	End Date:	-
Availability Zone:	us-west-2a	Instance Count:	-
Master Instance Type:	-	Slave Instance Type:	-
Key Name:	lheskaKP	Log URI:	-
Ami Version:	2.3.6	Master Public DNS Name:	-
Hadoop Version:	1.0.3	Keep Alive:	true
Termination Protected:	false	Visible To All Users:	false
Subnet Id:	-	Supported Products:	-

At the bottom, there's a copyright notice: '© 2008 - 2013, Amazon Web Services, Inc. or its affiliates. All rights reserved.' with links to 'Privacy Policy' and 'Terms of Use'. A 'Feedback' button is located in the bottom right corner.

Once the job status has changed to "WAITING," you can get what you need to proceed. Specifically, the contents of the Master Public DNS Name field. This field is in the Description tab, but it can be hidden. To find it, you may need to enlarge your browser window and/or scroll down within the Amazon "Your Elastic MapReduce Job Flows" window. Once you do, your screen should look something like this:

The screenshot shows the Elastic MapReduce Management Console interface. At the top, there's a navigation bar with "Services" and "Edit" menus. Below that, a section titled "Your Elastic MapReduce Job Flows" contains buttons for "Create New Job Flow", "Terminate", and "Debug". A table lists job flows, with one job flow "APTraining2" in a "WAITING" state. Below the table, the details for the selected job flow "j-3JZQTD1AH5LWY" are shown. The "Description" tab is active, displaying various configuration details for the job flow.

Name	State	Creation Date	Elapsed Time	Normalized Instance Hours
APTraining2	WAITING	2013-05-18 21:23 CDT	0 hours 7 minutes	3

1 Job Flow selected

Job Flow: j-3JZQTD1AH5LWY

Last State Change: Waiting after step completed

Description | Steps | Bootstrap Actions | Instance Groups | Monitoring

Name:	APTraining2	Creation Date:	2013-05-18 21:23 CDT
Start Date:	2013-05-18 21:28 CDT	End Date:	-
Availability Zone:	us-west-2a	Instance Count:	-
Master Instance Type:	-	Slave Instance Type:	-
Key Name:	lheskaKP	Log URI:	-
Ami Version:	2.3.6	Master Public DNS Name:	ec2-54-214-133-28.us-west-2.compute.amazonaws.com
Hadoop Version:	1.0.3	Keep Alive:	true
Termination Protected:	false	Visible To All Users:	false
Subnet Id:	-	Supported Products:	-

When I took this screenshot, the Master Public DNS Name was ec2-54-214-133-28.us-west-2.compute.amazonaws.com. Your Master Public DNS Name will be different. It is important and you should copy-and-paste it somewhere. It is what you will use to connect to your newly "spun up" Hadoop cluster.

Section 2.5.1: Starting a job for Pig

The previous part of this section covered two topics: general considerations and instructions for starting an EMR (Elastic MapReduce) job, and the specifics of starting an EMR job to run Hive. This section will presume you have mastered that material, and rather than repeat it all, give just the few details you need to start Pig jobs. The process is very much like starting Hive jobs.

First, as before, log on to AWS if you are not already on, select Services and Elastic MapReduce. Click on Create New Job Flow.

Give the job some name like "lheska-pig1." Choose "Pig Program" (in the same place as you chose "Hive Program" before).

Click "Start an Interactive Pig Session".

Choose the settings for a small job.

Choose the Amazon EC2 Key Pair for the user you want to allow to connect.

Make sure that "Visible To All IAM Users" is set to "Yes."

Proceed with no Bootstrap options.

Section 3: Connecting to your cluster

At this point you have logged on, roamed around in AWS, and found your Hadoop instances. Now you are ready to use them. This may be done in very many ways but we'll start with two of them; Hive and Pig.

Both Hive and Pig are Hadoop components that are largely command-line oriented. So at this point things will work and look different. Up until now you have been using web graphics, but now you'll be using a classic linux-like (or DOS-like) command window.

If you are a Windows user you may follow these instructions verbatim.

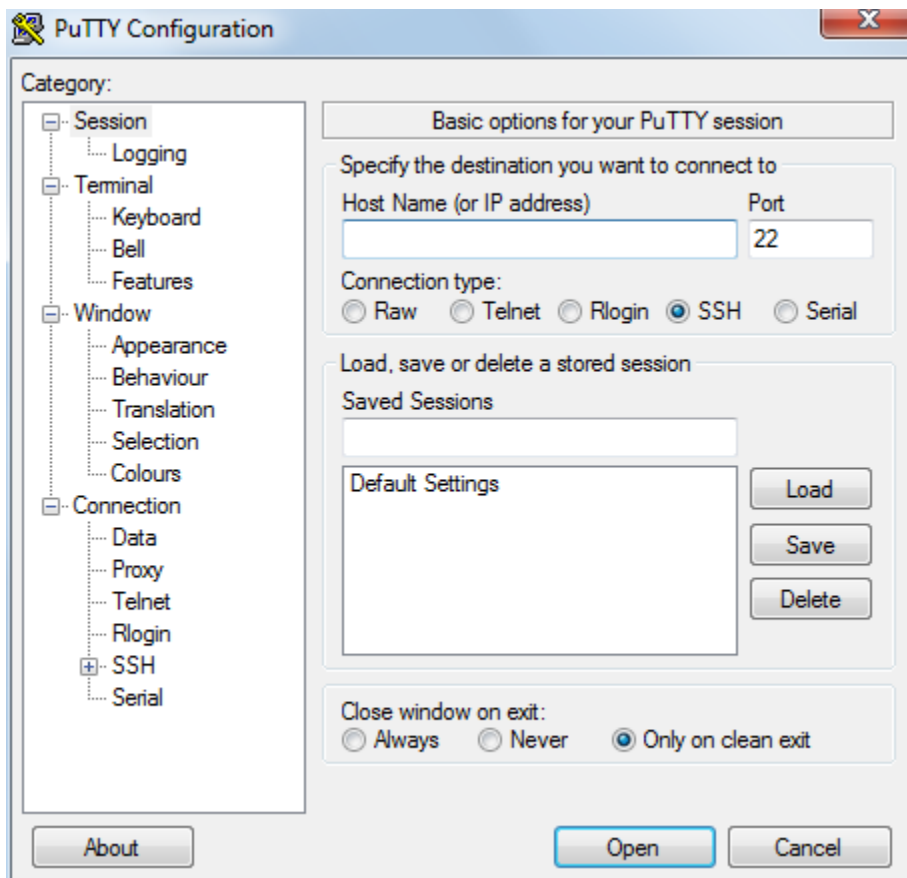
If you are a linux user, PuTTY is available for linux. These instructions may need to be altered slightly depending on how different linux PuTTY is from the Windows version (I have not tested this).

If you are a Mac user, the Terminal program resembles PuTTY. I have not used it so cannot give detailed instructions.

If you have not already downloaded and installed PuTTY, do so now. You can get PuTTY from

<http://www.putty.org/>

Once you have installed PuTTY and started it up, you should see a screen that looks something like this:

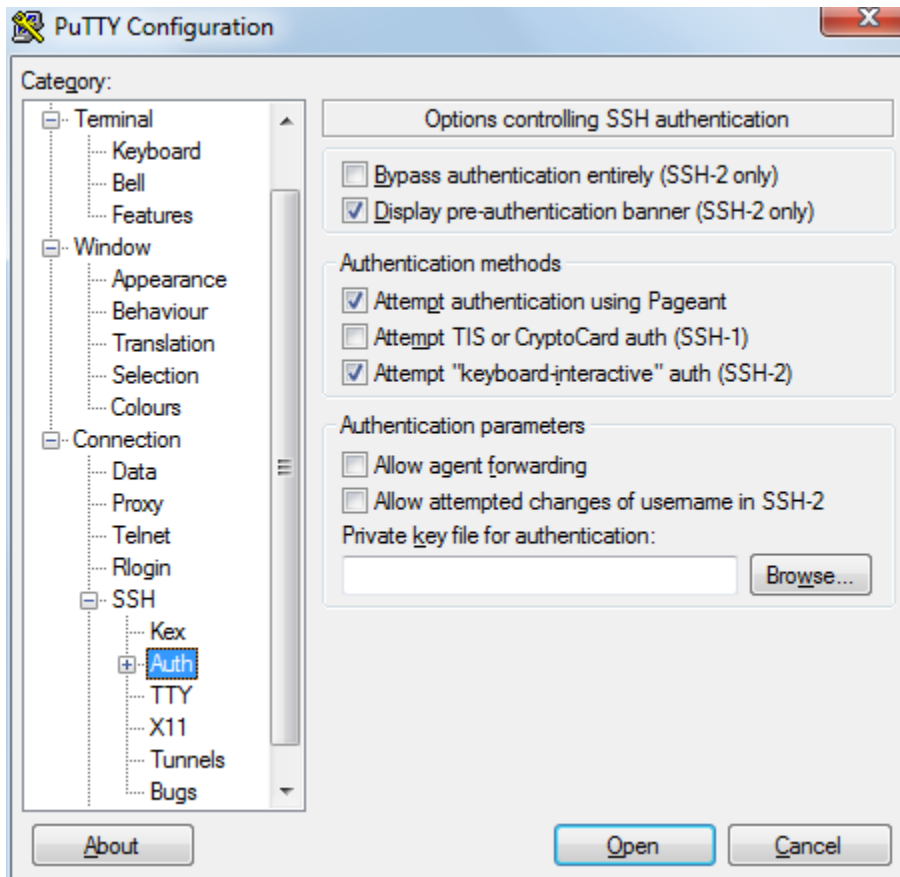


Back in the previous section of this document, I suggested that you record the Master Public DNS Name somewhere. Go to wherever you stored it, and paste it into the box at the top, that says "Host Name (or IP address)." If you neglected to record the Master Public DNS Name you can instead go to the window showing details of your job, and copy-and-paste it now.

Then, look toward the bottom left of the PuTTY window; notice that it says "SSH" there. Click on the plus sign ("+") to expand it.

Then click on the text "Auth" that will appear as a branch in the SSH tree.

At that point your PuTTY screen will change its appearance slightly, looking something like this:



Clicking on the text "Auth" as instructed above enables you to enter your credentials to log on.

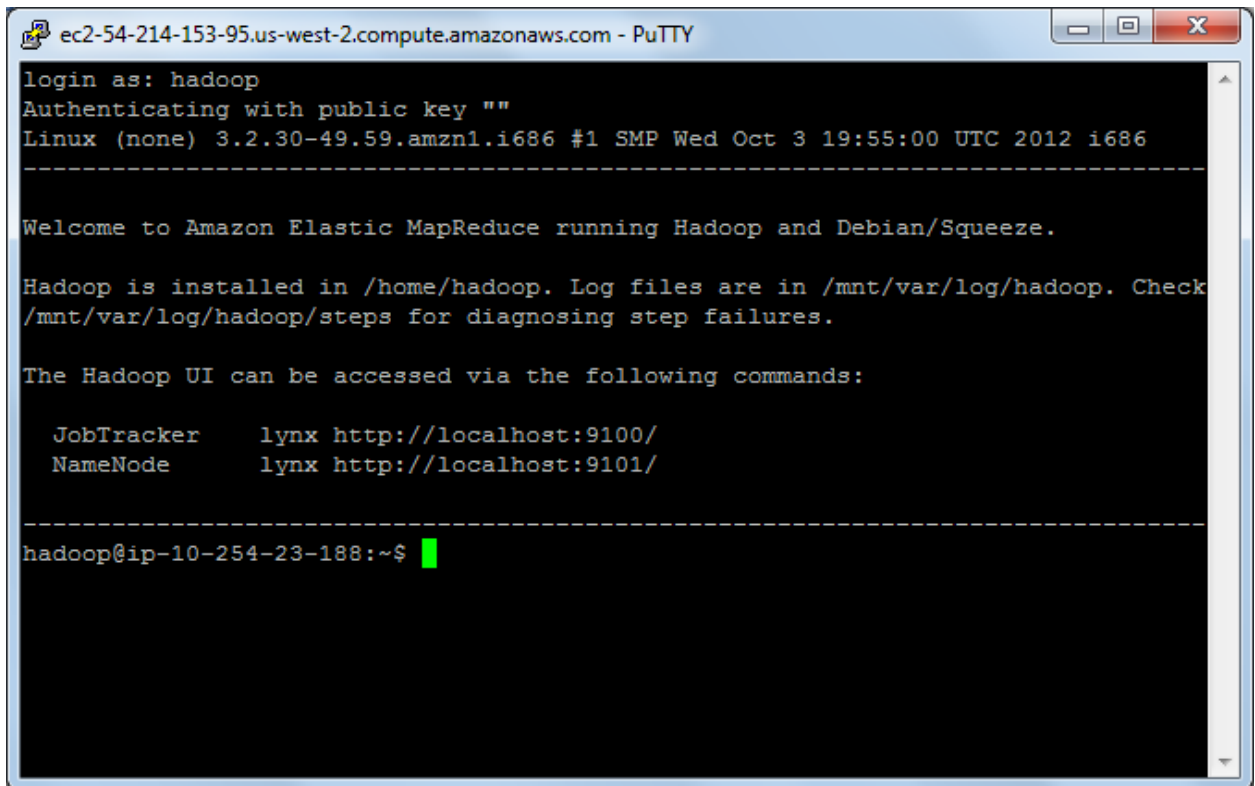
To do this, you need to know the location of the .ppk file that contains your private key. Click the Browse button, browse to your .ppk/private key file, select it, and then click "Open" to open your SSH connection to your AWS Hadoop instance.

At this point you should see a pretty empty black screen asking you to "login as:"

And you may be wondering what to do at this point.

Do you remember the text I mentioned was important to read? The screen titled "Create A New Job Flow?" One thing it told you was, to log in (or SSH) to the instance as the user "hadoop."

Do so now. Type "hadoop" and press <Enter> and you should get a screen that looks something like this:



```
ec2-54-214-153-95.us-west-2.compute.amazonaws.com - PuTTY
login as: hadoop
Authenticating with public key ""
Linux (none) 3.2.30-49.59.amzn1.i686 #1 SMP Wed Oct 3 19:55:00 UTC 2012 i686
-----
Welcome to Amazon Elastic MapReduce running Hadoop and Debian/Squeeze.

Hadoop is installed in /home/hadoop. Log files are in /mnt/var/log/hadoop. Check
/mnt/var/log/hadoop/steps for diagnosing step failures.

The Hadoop UI can be accessed via the following commands:

JobTracker      lynx http://localhost:9100/
NameNode        lynx http://localhost:9101/
-----
hadoop@ip-10-254-23-188:~$
```

Section 4: Running Hive queries from your Hive job

Hive (also known as Apache Hive) is a portion of Hadoop that allows you to treat data in files as if they reside in a relational database (RDBMS). Hive includes a language (Hive Query Language or HQL) that is an SQL variant. If you know SQL you will find HQL very familiar.

A word about case sensitivity:

Once you have connected to PuTTY as instructed in the previous section, you're working with a linux or bash command prompt. Linux is case-sensitive. So, when you go to start Hive, you will need to type "hive", not "Hive", because although the product name is Hive, the linux command is "hive".

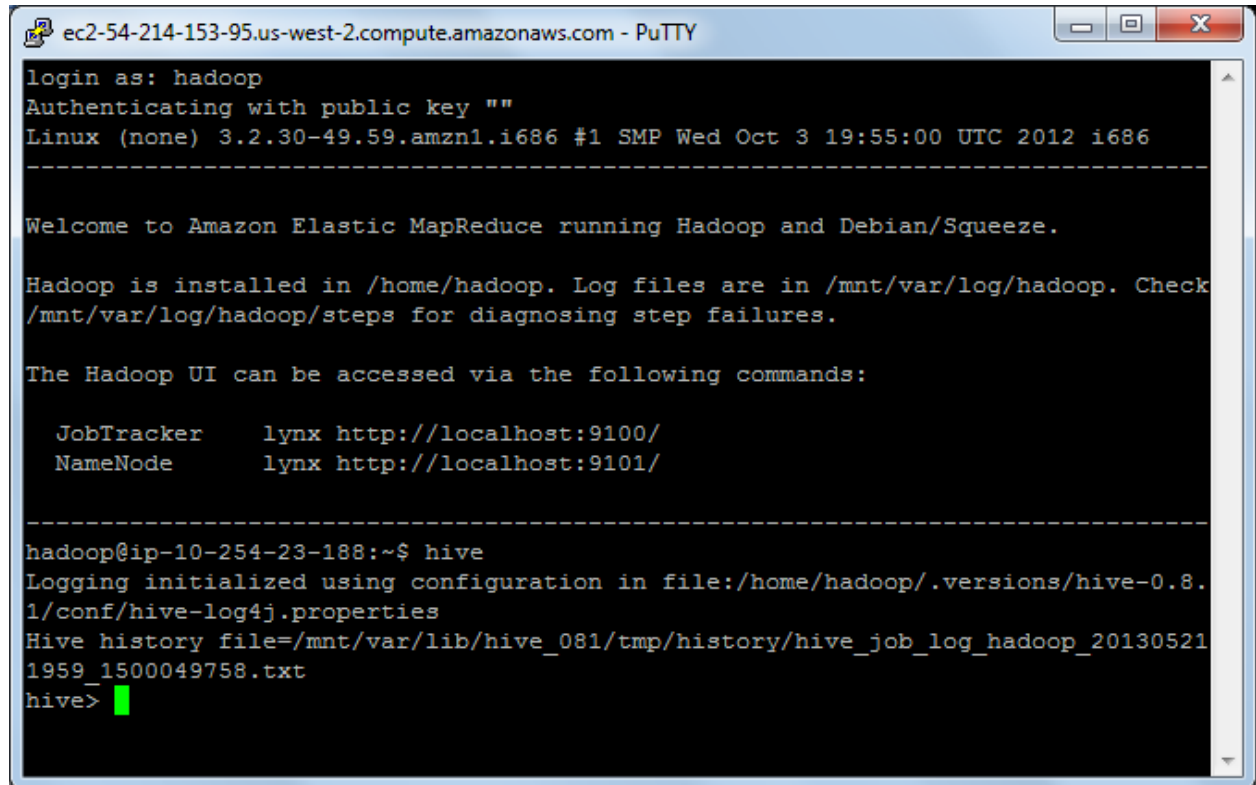
But once you actually start working in Hive, you will be working with HQL, the Hive Query Language. You may note that (unlike linux but like other SQL variants), HQL is generally not case-sensitive, but it can be: in some of your CREATE EXTERNAL TABLE commands (see below) you refer to external files, and the names of these must be correctly cased.

Like all multi-platform, multi-system computing, it can take some time to get used to what is, and what is not, case-sensitive. If you find yourself getting an odd error message (such as "-bash: Hive: command not found") you may want to try checking if you miscased your command.

To start Hive, just type "hive" (all lower-case) on your hadoop command line, and press <Enter>.

Hive and Hadoop in general can seem very slow to do things you would think should be near-instantaneous. The reasons for this are beyond the scope of this document (it has to do with Hadoop's system architecture) but for now, you may find that you need to wait what seems a long time (several seconds) just to get the Hive prompt.

But eventually you should see something like this:



```
ec2-54-214-153-95.us-west-2.compute.amazonaws.com - PuTTY
login as: hadoop
Authenticating with public key ""
Linux (none) 3.2.30-49.59.amzn1.i686 #1 SMP Wed Oct 3 19:55:00 UTC 2012 i686
-----
Welcome to Amazon Elastic MapReduce running Hadoop and Debian/Squeeze.

Hadoop is installed in /home/hadoop. Log files are in /mnt/var/log/hadoop. Check
/mnt/var/log/hadoop/steps for diagnosing step failures.

The Hadoop UI can be accessed via the following commands:

    JobTracker      lynx http://localhost:9100/
    NameNode        lynx http://localhost:9101/
-----
hadoop@ip-10-254-23-188:~$ hive
Logging initialized using configuration in file:/home/hadoop/.versions/hive-0.8.
1/conf/hive-log4j.properties
Hive history file=/mnt/var/lib/hive_081/tmp/history/hive_job_log_hadoop_20130521
1959_1500049758.txt
hive> █
```

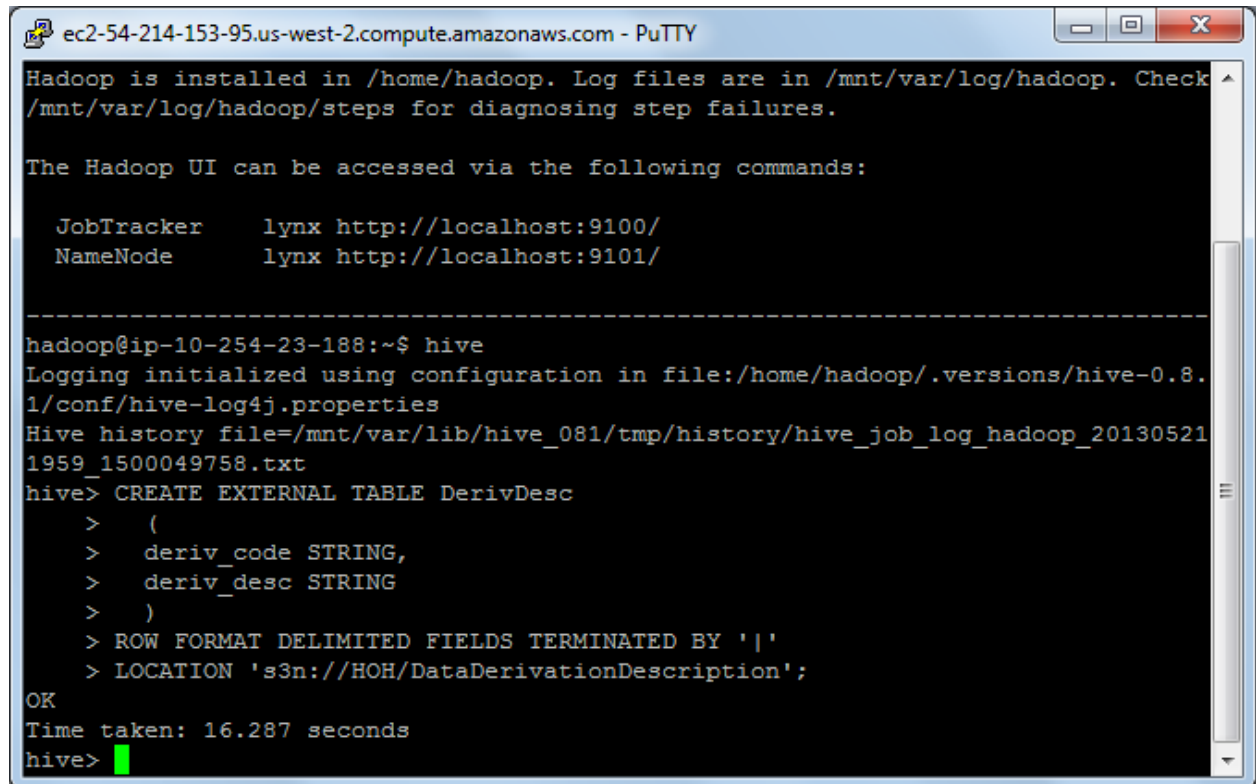
And now, you are "in" Hive. Until you exit Hive you will be working with the Hive command prompt (which is different from the Hadoop command prompt you had at first).

Now enter the following Hive command:

```
CREATE EXTERNAL TABLE DerivDesc
(
    deriv_code STRING,
    deriv_desc STRING
)
ROW FORMAT DELIMITED FIELDS TERMINATED BY '|'
LOCATION 's3n://HOH/DataDerivationDescription';
```

This Hive command does that first part of what I said Hive does - it allows you to treat text data (suitably formatted) as if they were residing in an RDBMS. If you look at the above command carefully you will see how it corresponds to the data file you may have browsed when you were looking in to S3.

The command should work, and you should see a screen that looks like this:



```
ec2-54-214-153-95.us-west-2.compute.amazonaws.com - PuTTY
Hadoop is installed in /home/hadoop. Log files are in /mnt/var/log/hadoop. Check
/mnt/var/log/hadoop/steps for diagnosing step failures.

The Hadoop UI can be accessed via the following commands:

JobTracker      lynx http://localhost:9100/
NameNode        lynx http://localhost:9101/

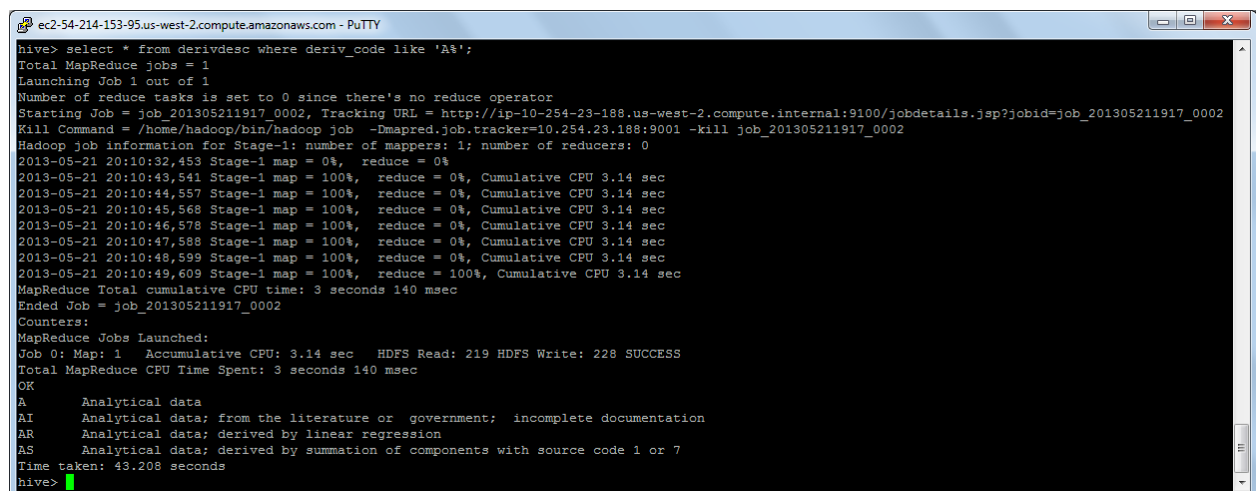
-----
hadoop@ip-10-254-23-188:~$ hive
Logging initialized using configuration in file:/home/hadoop/.versions/hive-0.8.
1/conf/hive-log4j.properties
Hive history file=/mnt/var/lib/hive_081/tmp/history/hive_job_log_hadoop_20130521
1959_1500049758.txt
hive> CREATE EXTERNAL TABLE DerivDesc
> (
>   deriv_code STRING,
>   deriv_desc STRING
> )
> ROW FORMAT DELIMITED FIELDS TERMINATED BY '|'
> LOCATION 's3n://HOH/DataDerivationDescription';
OK
Time taken: 16.287 seconds
hive>
```

Now you can type commands like the following:

```
select * from derivdesc where deriv_code like 'A%';
```

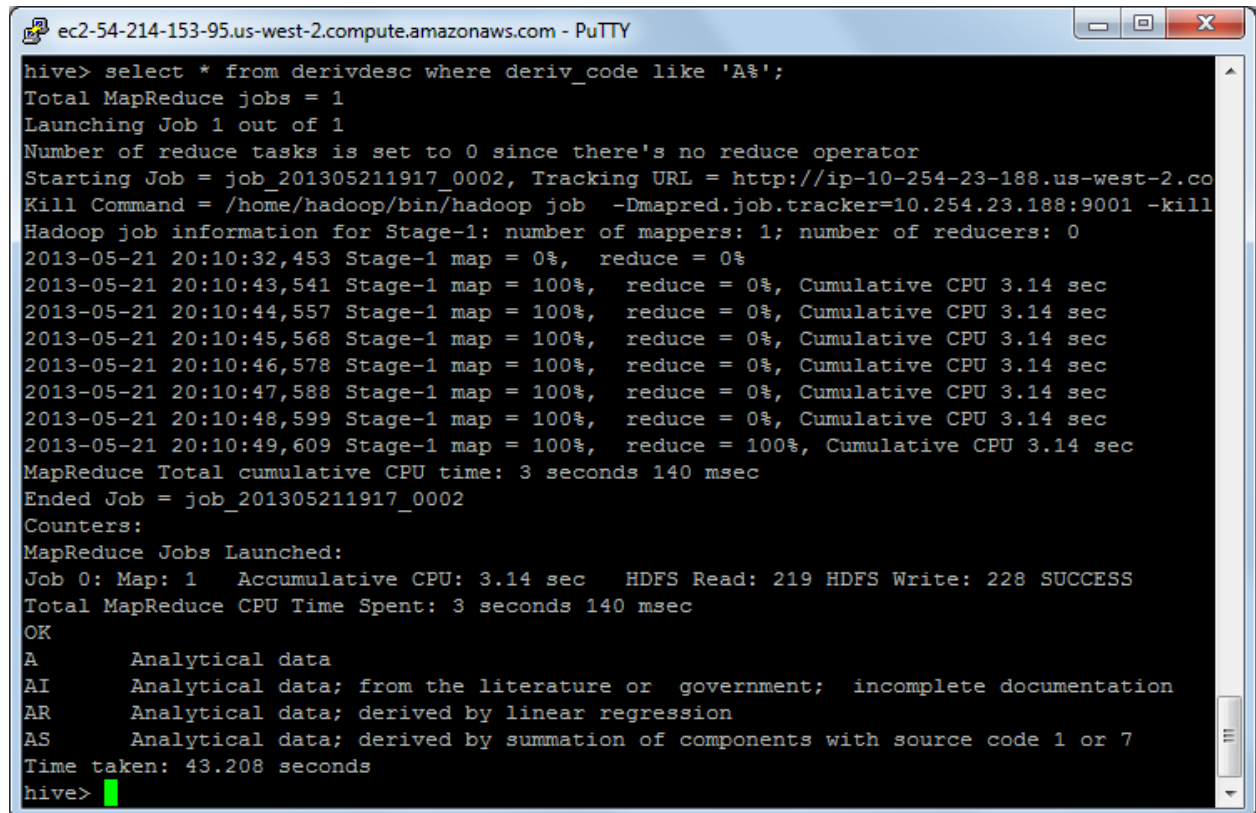
If you are used to other database systems (like Oracle, Access, DB2, Teradata, MySQL, PostgreSQL, and so on) you may be surprised at how long it takes Hive to return the results of your query.

But after a wait, you should see something that looks like this:



```
ec2-54-214-153-95.us-west-2.compute.amazonaws.com - PuTTY
hive> select * from derivdesc where deriv_code like 'A%';
Total MapReduce jobs = 1
Launching Job 1 out of 1
Number of reduce tasks is set to 0 since there's no reduce operator
Starting Job = job_201305211917_0002, Tracking URL = http://ip-10-254-23-188.us-west-2.compute.internal:9100/jobdetails.jsp?jobid=job_201305211917_0002
Kill Command = /home/hadoop/bin/hadoop job -Dmapred.job.tracker=10.254.23.188:9001 -kill job_201305211917_0002
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 0
2013-05-21 20:10:32,453 Stage-1 map = 0%, reduce = 0%
2013-05-21 20:10:43,541 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 3.14 sec
2013-05-21 20:10:44,557 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 3.14 sec
2013-05-21 20:10:45,568 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 3.14 sec
2013-05-21 20:10:46,578 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 3.14 sec
2013-05-21 20:10:47,588 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 3.14 sec
2013-05-21 20:10:48,599 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 3.14 sec
2013-05-21 20:10:49,609 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 3.14 sec
MapReduce Total cumulative CPU time: 3 seconds 140 msec
Ended Job = job_201305211917_0002
Counters:
MapReduce Jobs Launched:
Job 0: Map: 1 Accumulative CPU: 3.14 sec HDFS Read: 219 HDFS Write: 228 SUCCESS
Total MapReduce CPU Time Spent: 3 seconds 140 msec
OK
A Analytical data
AI Analytical data; from the literature or government; incomplete documentation
AR Analytical data; derived by linear regression
AS Analytical data; derived by summation of components with source code 1 or 7
Time taken: 43.208 seconds
hive>
```

Or, with some of the right-hand side cut off to make reading easier:

A screenshot of a PuTTY terminal window titled "ec2-54-214-153-95.us-west-2.compute.amazonaws.com - PuTTY". The terminal displays the output of a Hive query: "hive> select * from derivdesc where deriv_code like 'A%';". The output shows job execution details, including "Total MapReduce jobs = 1", "Launching Job 1 out of 1", and "Number of reduce tasks is set to 0 since there's no reduce operator". It lists the starting job, tracking URL, kill command, and Hadoop job information for Stage-1. Progress updates show Stage-1 map reaching 100% and reduce reaching 100% at 20:10:49,609. Summary statistics include "MapReduce Total cumulative CPU time: 3 seconds 140 msec", "Ended Job = job_201305211917_0002", and "Counters: MapReduce Jobs Launched: Job 0: Map: 1 Accumulative CPU: 3.14 sec HDFS Read: 219 HDFS Write: 228 SUCCESS". The total CPU time spent is "3 seconds 140 msec". The results list shows four entries starting with "A": "Analytical data", "Analytical data; from the literature or government; incomplete documentation", "Analytical data; derived by linear regression", and "Analytical data; derived by summation of components with source code 1 or 7". The time taken is "43.208 seconds". The prompt "hive>" is followed by a green cursor.

As you can see, all the descriptions that begin with the letter "A" are all listed.

Once done with Hive, you can type "exit;" to return to the command prompt, then "exit" to quit PuTTY.

A note of caution:

Though the examples we will be working with all use small data sets, Hadoop is designed to deal with hundreds of billions of rows. You may be tempted to shorten the HQL query you just ran, to show all the rows, not just those beginning with "A". In the case of this particular data file that would not be a problem. But if your data set really does contain 400,000,000,000 rows, it would. Though Hive could handle it, streaming all that text across the network to your command window would really bog things down.

Section 5: Running Pig commands from your Pig job

Pig (also known as Apache Pig) is a portion of Hadoop that allows you to manage (presumably very large) sets of records in a file. Pig makes it relatively easy to sort records, find particular fields or strings inside of records, reformat records, and/or choose only certain fields from records.

Pig does not attempt (like Hive/HQL does) to give you an SQL-like experience. However, Pig is a relational (as opposed to a procedural) language, and parts of it will seem kind of

familiar to a person used to SQL, or with experience in relational algebra (the mathematical discipline/topic that underlies both relational databases and Pig).

The theory that underlies Pig is too extensive to address in this document. However, a few notes may help avoid some confusion and frustration:

- Though Pig Latin (the language used in/for Pig) uses equals signs in every line, don't think that you can write a statement that does something like adding two numbers, like "X = 2 + 3" or for that matter, "X = A + B" where A and B have been set to 2 and 3, respectively, in prior lines of code.
- Neither does Pig support counters, loops, if/then statements, or any similar constructs you may be used to if you have ever programmed using a procedural language like Java, JavaScript, Visual Basic, Perl, PHP, or C/C++/C#.
- Rather, everything that is on the left side of an equals sign in Pig Latin (the programming language used in/with Pig) is a relation. It's pretty close to correct, to say that "a relation is a table" just like you'd see and work with in Oracle, Access, MySQL, or even one Excel sheet or tab.
- Everything to the right of the equals sign in Pig is some sort of manipulation of or action upon one or more relations.

A word about case sensitivity:

Once you have connected to PuTTY as instructed in Section 3, you're working with a linux or bash command prompt. Linux is case-sensitive. So, when you go to start Pig, you will need to type "pig", not "Pig", because although the product name is Pig, the linux command is "pig".

But once you actually start working in Pig, you will be working with Pig Latin, the language Pig uses. Some of what you try may require things to be correctly cased; other times you may be able to get away with alternate casings.

Like all multi-platform, multi-system computing, it can take some time to get used to what is, and what is not, case-sensitive. If you find yourself getting an odd error message (such as "-bash: Pig: command not found") you may want to try checking if you miscased your command.

To start Hive, just type "pig" (all lower-case) on your hadoop command line, and press <Enter>.

You should get a pretty prompt response, and eventually, a new command prompt that says:

```
grunt>
```

Isn't that clever? The inventors/designers of Pig were having lots of fun. From now on for as long as you use Pig interactively you'll be seeing these "grunt" prompts.

For purposes of this training, someone should already have put a file named prince.txt out on the S3 filesystem, in the bucket named "textinput." Have a look out there if you would like, to make sure it's there.

Then, try typing the following Pig Latin commands at the grunt> command prompt:

```
input_lines = LOAD 's3n://textinput/prince.txt' AS (line:chararray);
words = FOREACH input_lines GENERATE FLATTEN(TOKENIZE(line)) AS word;
filtered_words = FILTER words BY word MATCHES '\\w+';
word_groups = GROUP filtered_words BY word;
word_count = FOREACH word_groups GENERATE COUNT(filtered_words) AS count, group AS word;
ordered_word_count = ORDER word_count BY count DESC;
```

If you did the section about Hive you may be thinking around now that Pig is pretty fast. That's an illusion. Pig has not actually done anything yet. It's just sitting there, accepting instructions for what it will do, once you actually kick things off.

Do so by entering one last command. Before you do, you need to identify a subdirectory of the S3 directory named "pigout9" that does not yet exist.

Important note:

It's an integral part of how Pig works, that it writes only to directories, not to files. Also, it will not write to a directory that already exists. So, for the following (and for all write-to-disk-storage commands) you will need to enter a name of a directory that you want pig to:

1. create, and
2. write to.

Since many teams will be writing Pig output to S3 storage, here's a suggestion: incorporate your corp user ID into the directory name you want to create and write to. So, for example, write your output to the S3 directory

pigout9/ssmithtry1

That means that for the following, final Pig Latin command, you can't just type it, you need to customize it first:

```
STORE ordered_word_count INTO 's3n://pigout9/{your-corp-ID}try1';
```

This command (once you customize it right) will actually kick off the Pig job. Generally the job will take several minutes to run. You will get a lot of diagnostic output while it's running. Finally, you should see something like this:

```
HadoopVersion PigVersion  UserId StartedAt  FinishedAt  Features
1.0.3 0.9.2-amzn  hadoop 2013-05-28 00:58:04 2013-05-28 01:01:49 GROUP_BY,ORDER_BY,FILTER
Success!
Job Stats (time in seconds):
JobId  Maps  Reduces MaxMapTime  MinMapTime  AvgMapTime  MaxReduceTime  MinReduceTime  AvgReduceTime  Alias  Feature Outputs
job_201305280017_0005  1  1  12  12  15  15  15
filtered_words,input_lines,word_count,word_groups,words GROUP_BY,COMBINER
job_201305280017_0006  1  1  9  9  18  18  18  ordered_word_count  SAMPLER
job_201305280017_0007  1  3  12  12  21  12  17  ordered_word_count  ORDER_BY  s3n://pigout9/temp1,
Input(s):
Successfully read 2740 records (82703 bytes) from: "s3n://textinput/prince.txt"
Output(s):
Successfully stored 3463 records in: "s3n://pigout9/temp1"
Counters:
```

```
Total records written : 3463
Total bytes written : 0
Spillable Memory Manager spill count : 0
Total bags proactively spilled: 0
Total records proactively spilled: 0
Job DAG:
job_201305280017_0005 -> job_201305280017_0006,
job_201305280017_0006 -> job_201305280017_0007,
job_201305280017_0007
2013-05-28                01:01:49,974                [main]                INFO
org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Success!
```

After all that, you should still be at the `grunt>` command prompt.

If you want to test what I told you here, about how Pig will refuse to write to a directory that already exists, just try running that very last one command, the `STORE` command, again. (Pressing the up arrow is an easy way to retrieve the last command). If you try to re-run it, you should get an error message saying

Output directory {name of directory} already exists

The message comes back right away, without waiting, so that you (and your cluster) don't waste time running a job only to fail when trying to write the output.

There is a lot more to Pig and programming in Pig Latin. This introduction has been intended to give you the start you need, to keep going if you want/need to.

Section 6 (for Power Users): Running and using R

There are many ways to use R in Hadoop and many (including many of the usages we'd most like to be able to use right now) are still being developed. Meaning that no instructions for these functions and/or features can be written because they don't yet exist.

This section is to give you one simple example of how you can run an R script, non-interactively. Like the other sections of this document, it doesn't begin to address the many uses of R in Hadoop; even the ones that exist and are implemented in AWS today. It's just one way to get started.

If you are a "power user" and have already done the other power user sections of this document, you don't need to see the precise screenshots again. Create a new job flow much as you did for Hive and Pig users, with the following differences, settings, and choices:

Services

Elastic MapReduce

First screen: Create a Job Flow*, choose Run your own application and from the "Choose a Job Type" drop-down list box, choose "Streaming."

{next screen}

Input Location: streaminput/wants

{create a new subdirectory in S3 before next step, for example lookmom/2}

Output Location: lookmom/2

Mapper: yourcodehere/mapper.R

Reducer: yourcodehere/reducer.R

{next screen}

Amazon EC2 Key Pair: {not needed for this flow/task/job, so}

Proceed without an EC2 Key Pair

Amazon S3 Log Path: logjam/streaming

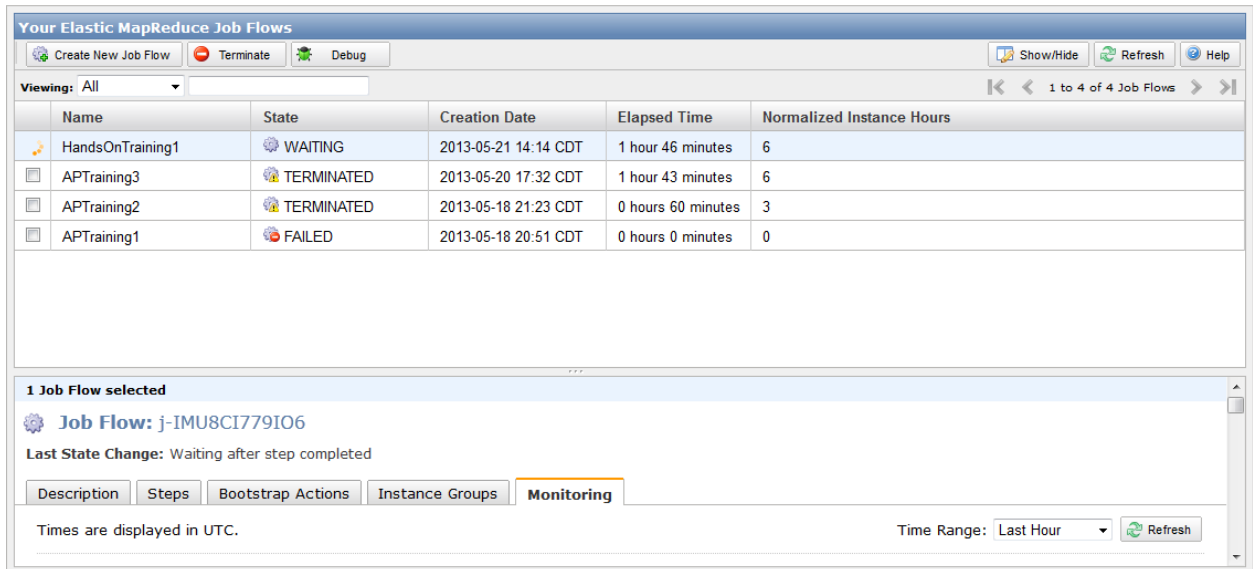
Visible To All IAM Users: **Yes**

Proceed through the final review screen and submit the job.

No need to terminate it afterwards because it will self-terminate when done.

Section 7 (for Power Users): Terminating your Hadoop jobs and cluster

If you have a look back at your AWS management console (clicking on Services/Elastic MapReduce if you need to), you should see a screen that looks something like this:



The screenshot shows the 'Your Elastic MapReduce Job Flows' page in the AWS console. At the top, there are buttons for 'Create New Job Flow', 'Terminate', and 'Debug'. Below these is a 'Viewing: All' dropdown and a search bar. The main table lists four job flows:

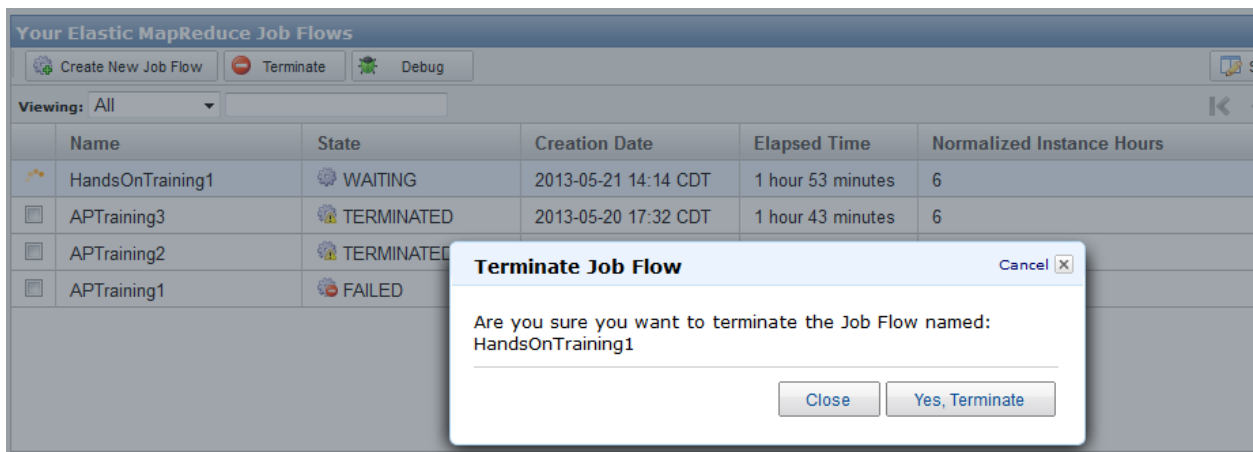
	Name	State	Creation Date	Elapsed Time	Normalized Instance Hours
	HandsOnTraining1	WAITING	2013-05-21 14:14 CDT	1 hour 46 minutes	6
	APTraining3	TERMINATED	2013-05-20 17:32 CDT	1 hour 43 minutes	6
	APTraining2	TERMINATED	2013-05-18 21:23 CDT	0 hours 60 minutes	3
	APTraining1	FAILED	2013-05-18 20:51 CDT	0 hours 0 minutes	0

Below the table, there is a section for '1 Job Flow selected' with details for 'Job Flow: j-IMU8C1779IO6'. It shows the 'Last State Change: Waiting after step completed' and tabs for 'Description', 'Steps', 'Bootstrap Actions', 'Instance Groups', and 'Monitoring'. At the bottom right, there is a 'Time Range: Last Hour' dropdown and a 'Refresh' button.

Note that the job you started up at the beginning, is still in status "WAITING." What that means, is that your Hadoop cluster, running out there on Amazon's servers, is still up, still running, and sitting around waiting for more Hadoop, Hive, or Pig commands.

As long as your instance is up and running, it is potentially costing [client] money. So, it's important to terminate the Hadoop instance when you are done.

To do so is easy. Just click on your job that is in status "WAITING" and then click on the red "Terminate" button at the top of the screen. You should then see a message like this:



The screenshot shows the same AWS Elastic MapReduce console page, but with a 'Terminate Job Flow' dialog box open. The dialog box has a title bar 'Terminate Job Flow' and a 'Cancel' button. The main text asks: 'Are you sure you want to terminate the Job Flow named: HandsOnTraining1'. At the bottom, there are two buttons: 'Close' and 'Yes, Terminate'.

Click "Yes, Terminate." The job status should change to "SHUTTING DOWN" and eventually, to "TERMINATED."

Section 8 (for Power Users): Moving data using MapR NFS mount

In all the previous exercises, when you started jobs, you chose the default Amazon distribution of Hadoop.

In this exercise you will instead "spin up" an instance of MapR's distribution of Hadoop.

Hadoop, being open-source, has been adapted, changed, and/or extended by various companies. MapR is one such company and their distribution, simply called "MapR Hadoop," differs from regular Apache Hadoop primarily in that data are stored in MapR's proprietary file system, not in Hadoop's HDFS file system. That, to analysts, is a technical detail that's not really worth learning.

However, for certain applications, the MapR distribution can be quite useful/handy. One thing about the MapR file system, is that even though it is a distributed file system (therefore it can nicely support Hadoop's distributed processing), it also provides a simple "NFS mount." Just what NFS means (Networked File System, if you care) and how it works, is beyond the scope of this document. From our point of view, NFS makes it easy to copy things into and out of our Hadoop cluster. This permits, among other things, running a "turn Terabytes into Megabytes" task on Hadoop, then downloading the resulting (megabytes of) data to another machine for further analysis and/or use.

This section presumes that you have mastered the preceding tasks/sections, and instructions herein are abbreviated. Accomplishing the tasks in this section, is much like performing the Hive and Pig sections.

* * *

First, as before, log on to AWS if you are not already on, select Services and Elastic MapReduce. Click on Create New Job Flow.

On the first screen (titled "DEFINE JOB FLOW"):

Give the job some name like "lheska-NFS1."

Choose MapR M5 Distribution (different from what you've chosen before)

Choose "Hive Program" (though if you want to and know what you're doing, you could start a Pig program/job here).

On the next screen, click "Start an Interactive Hive Session".

On the next screen, you won't have the option you had before, of a small job. The default settings you see should specify a Large (m1.large) job, like this:

Create a New Job Flow Cancel

DEFINE JOB FLOW SPECIFY PARAMETERS **CONFIGURE EC2 INSTANCES** ADVANCED OPTIONS BOOTSTRAP ACTIONS REVIEW

Specify the master, core and task nodes to run your job flow. For more than 20 instances, complete the [limit request form](#).

Master Instance Group: This EC2 instance assigns Hadoop tasks to core and task nodes and monitors their status.

Instance Type: ☐ Request Spot Instance

Core Instance Group: These EC2 instances run Hadoop tasks and store data using the Hadoop Distributed File System (HDFS). Recommended for capacity needed for the life of your job flow.

Instance Count:

Instance Type: ☐ Request Spot Instances

Task Instance Group (Optional): These EC2 instances run Hadoop tasks, but do not persist data. Recommended for capacity needed on a temporary basis.

Instance Count:

Instance Type: ☐ Request Spot Instances

[< Back](#) * Required field

Continue

Next screen:

- choose the Amazon EC2 Key Pair for the user you want to allow to connect.
- Make sure that "Visible To All IAM Users" is set to "Yes."

Proceed with no Bootstrap options.

Click through to the end, wait for the new job to go through the "STARTING" and "BOOTSTRAPPING" states (I don't know why this state is entered even though you specified no bootstrapping options above), the "RUNNING" state, and, finally, the "WAITING" state.

It takes about 5 minutes to get to the "RUNNING" state; about 7 minutes to get to the "WAITING" state. You may notice that whereas the small jobs you started in previous sections spun up clusters of 3 machines, the job you just started spun up a 12-machine cluster (you can tell by looking at the Normalized Instance Hours).

Connect to the newly spun-up instance just as you did when you started a Hive session; find your Master Public DNS Name, plug it in to PuTTY, specify your private key, log in as user "hadoop".

Once logged on you should see the linux command prompt, something like this:

```
hadoop@ip-10-254-71-178:~$
```

If you'd like, start Hive and run some Hive commands. If you or someone working with you knows how, you may also save some Hive output to the linux MapR filesystem.

If you ran Hive commands, get out of Hive, back to a linux command prompt.

Type the following at the command prompt (prompts not shown):

```
cd /mapr/MapR_EMR.amazonaws.com
mkdir in
echo "This is a test." > in/data.txt
ls -la in
```

Now, you have a data file named

data.txt

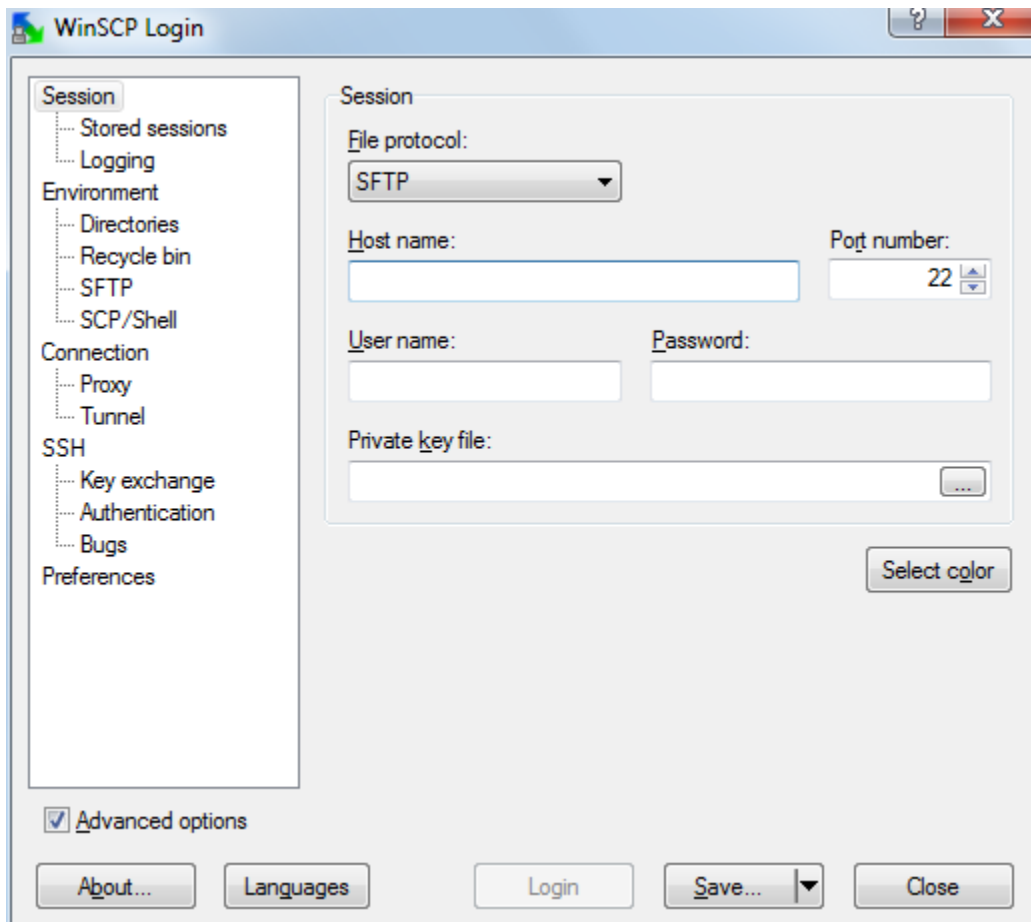
sitting in the directory

/mapr/MapR_EMR.amazonaws.com

which is really an NFS mount provided by MapR.

To access this data file remotely, you need a program such as WinSCP, which stands for "Windows Secure CoPy." If you are using linux you may simply use scp; and there are plenty of other scp equivalents.

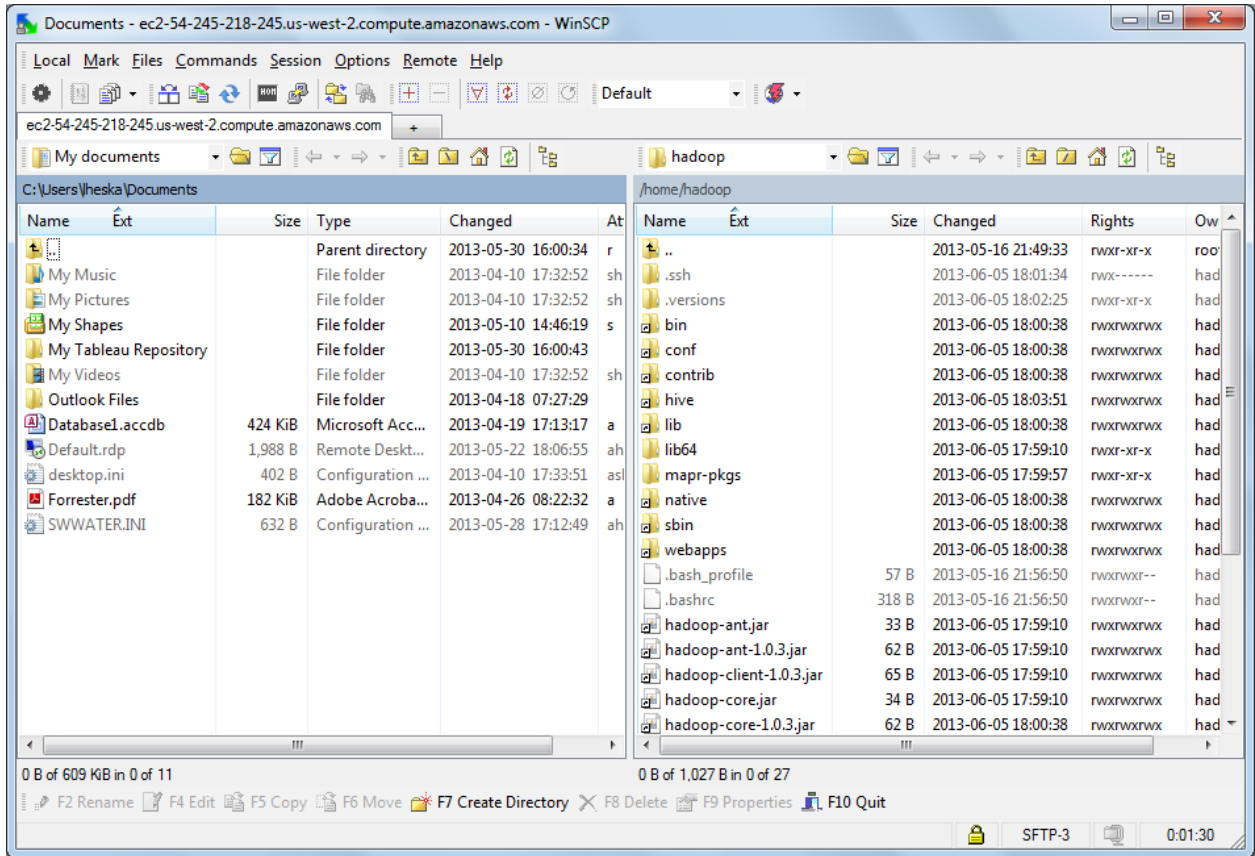
The WinSCP interface is similar to PuTTY:



Fill in the host name and browse to your .ppk file, then click Login.

After some messages (you may have to click a couple of things regarding accepting keys) you should be asked for a login - type "hadoop".

At this point you should get a screen that looks something like this:



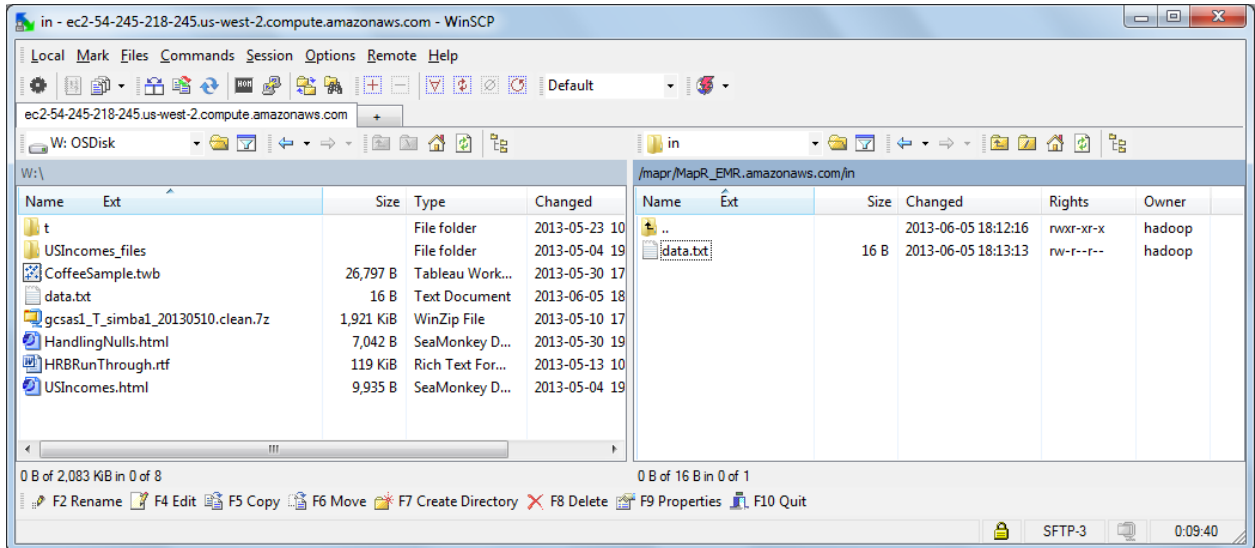
which is just a standard split-screen file management window showing your Windows machine's filesystem on the left and Hadoop's on the right.

Note that toward the upper right of this screen, it's telling you that your current working directory is `/home/hadoop`. That's not where the test file is located, that you created earlier. Using the window's navigation features (there are a variety of ways to do this), navigate your way to

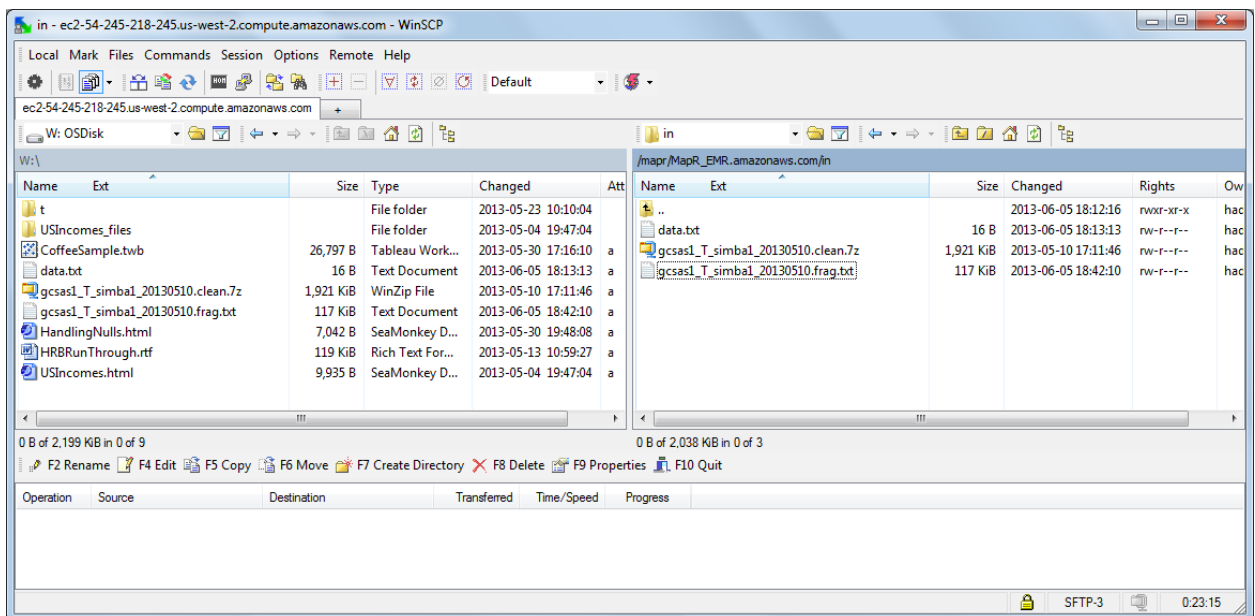
`/mapr/MapR_EMR.amazonaws.com/in`

Once there, you should see the file named `data.txt` that you created earlier. Navigate to a directory on your Windows filesystem, then drag and drop `data.txt` to it.

The result should look something like this:



Also, you may copy files from your Windows filesystem to the MapR NFS mount. When I did that, I got a screen like this:



As you may see, there are a couple of data files out there now on AWS.

The point of the NFS mount is to make it look like data stored on the MapR filesystem (and therefore available for Hadoop processing) are on the linux filesystem.

So when you transfer data to

```
/mapR/MapR_EMR.amazonaws.com/in
```

it looks like your data file is sitting there in a directory with a kind of funny name, in the linux filesystem.

To make sure that you really have gotten your data on the MapR filesystem, you can go to your PuTTY session, start Hive, and do something like this:

```
CREATE EXTERNAL TABLE dummy
(
  field1 STRING,
  field2 STRING,
  field3 STRING
)
ROW FORMAT DELIMITED FIELDS TERMINATED BY '|'
LOCATION '/mapr/MapR_EMER.amazonaws.com/in';

select * from dummy where field2 like 'jvan%';
```

(obviously you would need to tailor your HQL query to your data)

If you have written your Hive query correctly, it should return results to you, proving that the data/your data file really do/does reside on the MapR file system.

Note that using command-line scp inside a batch job would make it rather easy to harvest the result of a "turn Terabytes into Megabytes" job run on AWS.

Also not that in linux it is (said to be) possible to accomplish all steps of this process using no visual tools at all; purely command-line tools, using the Amazon Elastic MapReduce Ruby Client.

Once done, be sure to go back to the AWS console and terminate the Hive job.

Appendix 1 - Advanced/additional exercises and topics

If you are really interested in map/reduce and want to give it a go, you may try following the directions here:

<http://commoncrawl.org/mapreduce-for-the-masses/>

but personally I wouldn't trust the site or the document. The title of the page includes the words "Five Minutes" and there's no way a typical person is going to get all this document's steps done in that time, or even close. If you are already expert in every one of the skills mentioned, and have every one of the necessary tools installed, configured, and ready to go, and are also expert in all those tools, then maybe you could do this in less than an hour.