Big Data Workshop

Amazon Kinesis, Amazon Redshift, Amazon Elastic Map Reduce

June 9th, 2014

Lab 1: Amazon Kinesis

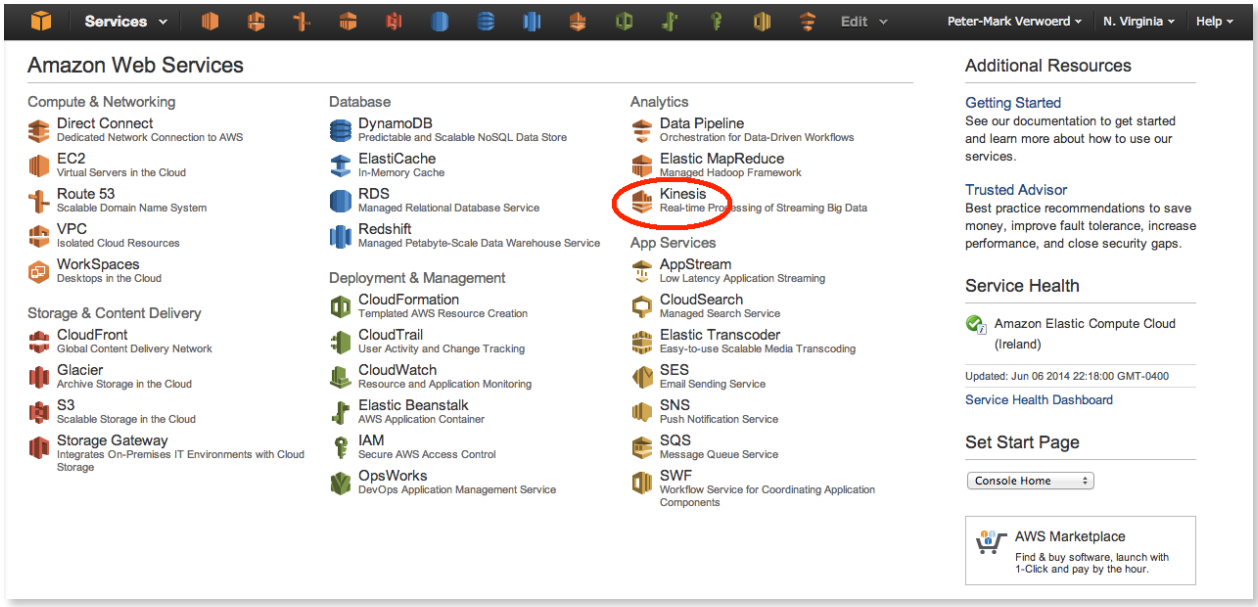
In this first Lab, you will create a Kinesis Stream on the AWS console. Once it is running, you will then use a Cloudformation template to launch an application that will produce random data to feed in to the Kinesis Stream ("producer") as well as an application using the Kinesis Client Library to consume the data from the Kinesis Stream and visualize it ("visualizer").

Step 1: Create the Kinesis Stream

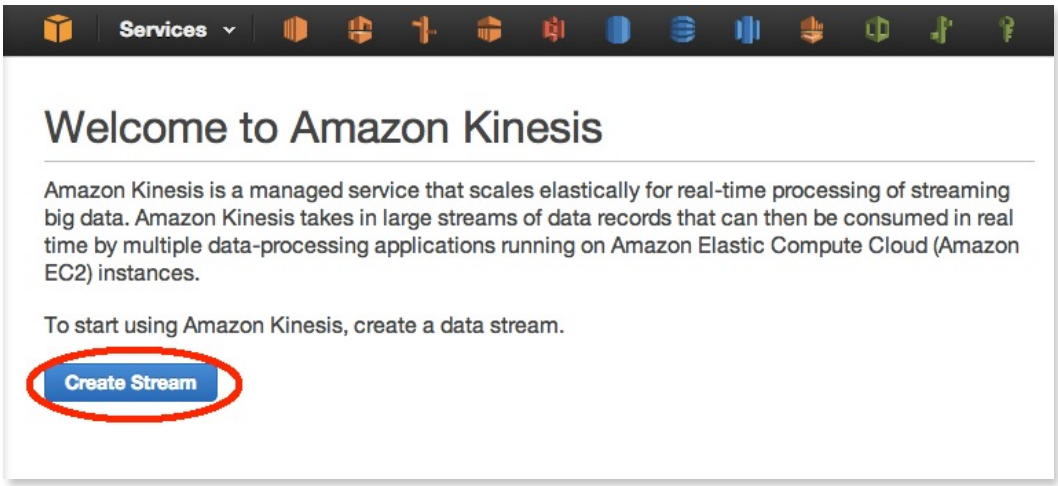
Start at the AWS Console home page:

https://console.aws.amazon.com/console/home

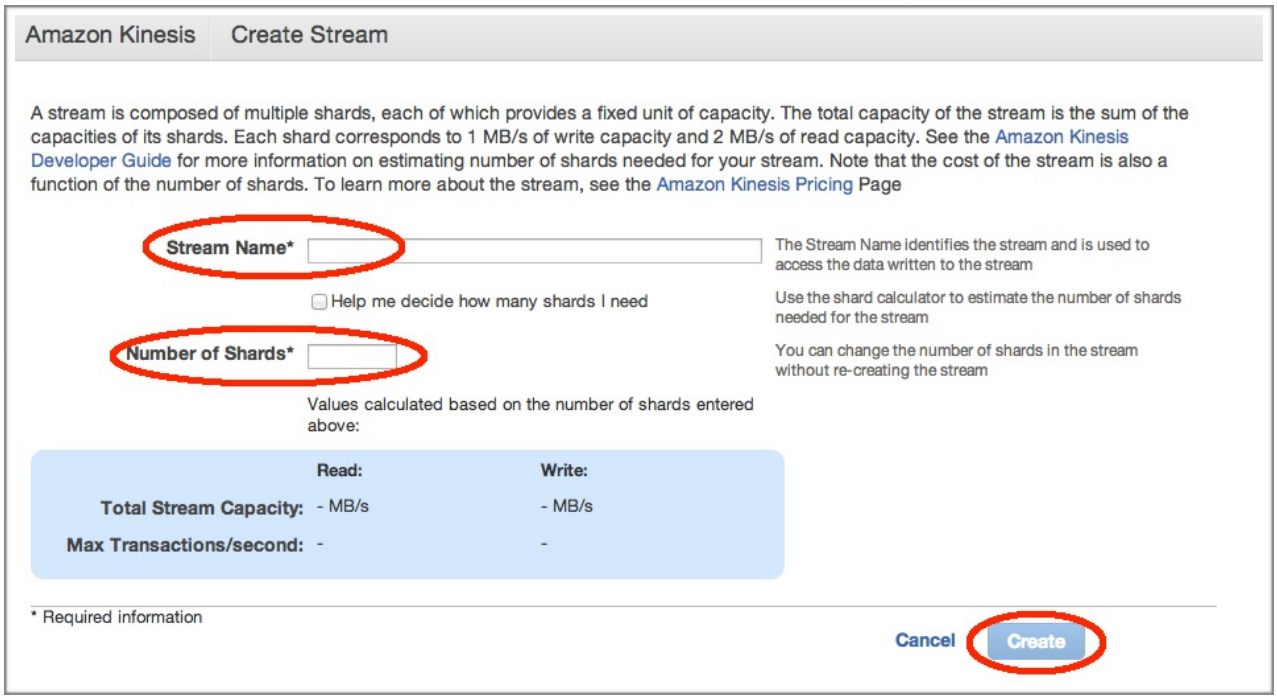
From the AWS Console home page, select Kinesis



Click "Create Stream"



Give it any name you wish (remember it for later). Put in 2 shards and click “Create”.

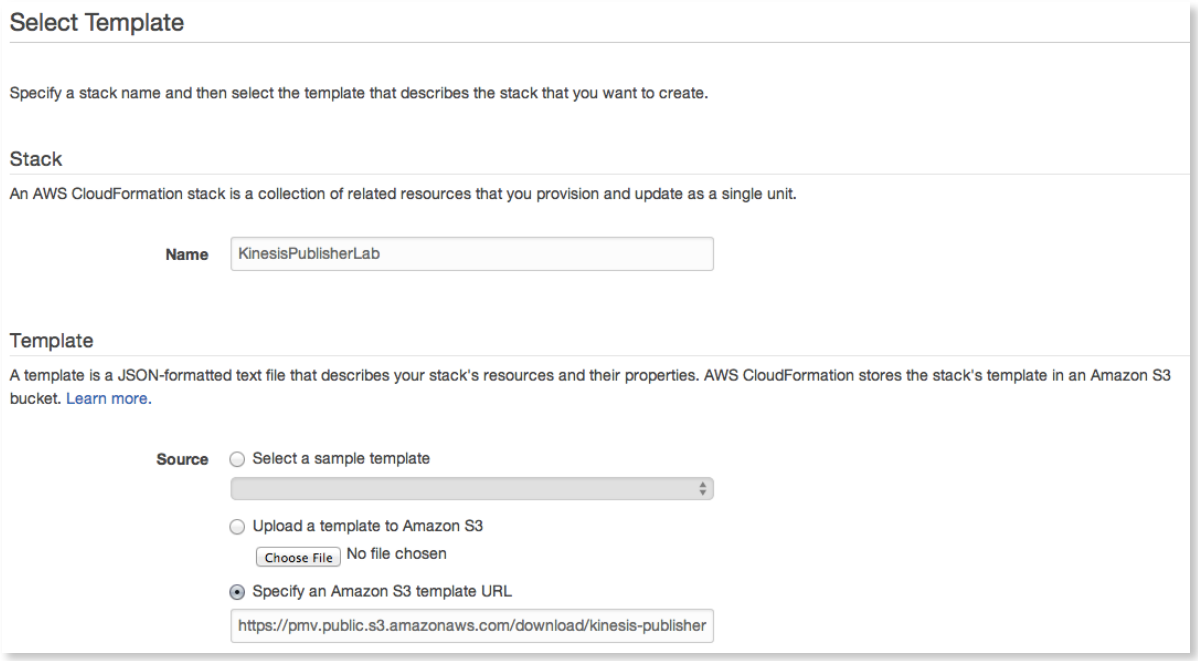


Step 2: Create the Producer and Visualizer

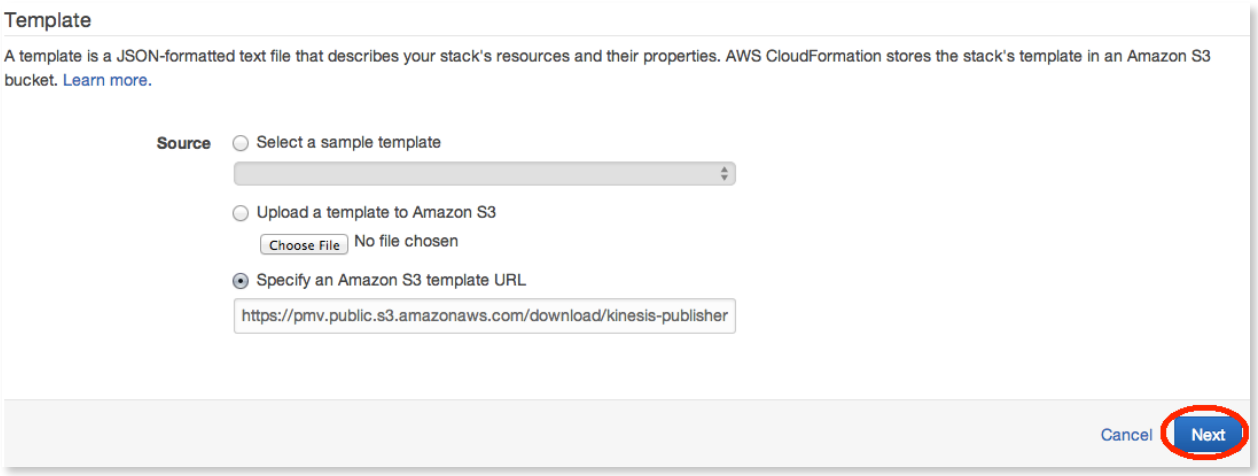
Here you will create the Kinesis producer and consumer using a Cloudformation template. Click on the link below:

KinesisPublisherLab

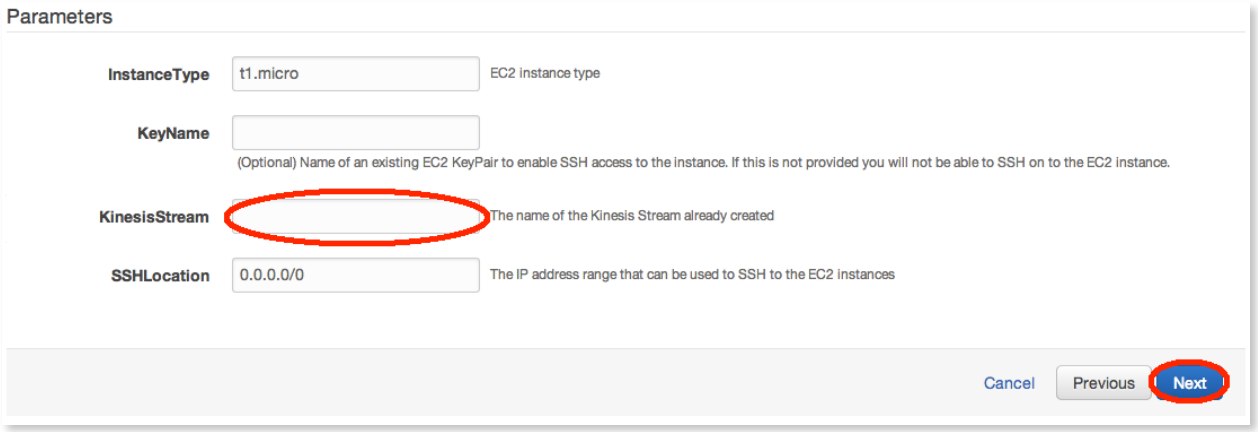
This will open the AWS Console to the Cloudformation page:



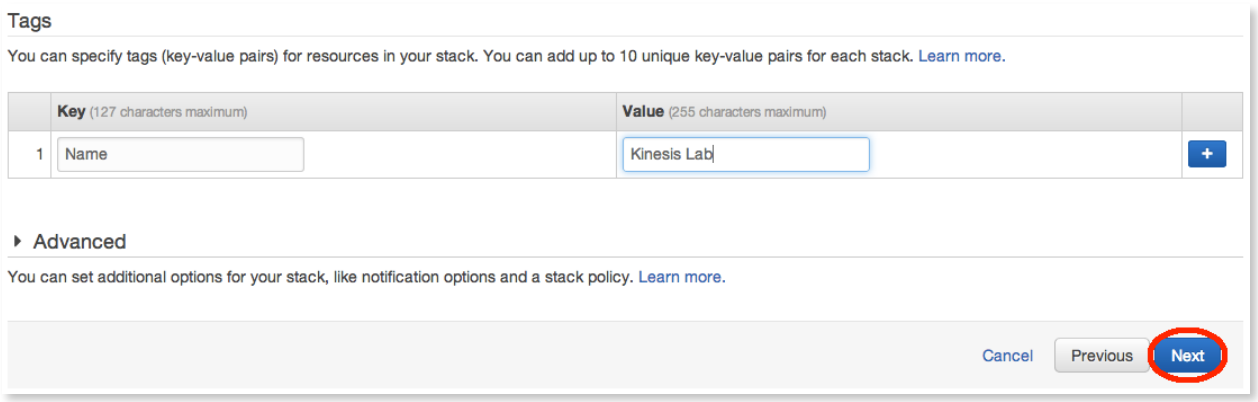
Scroll down and click “Next”:



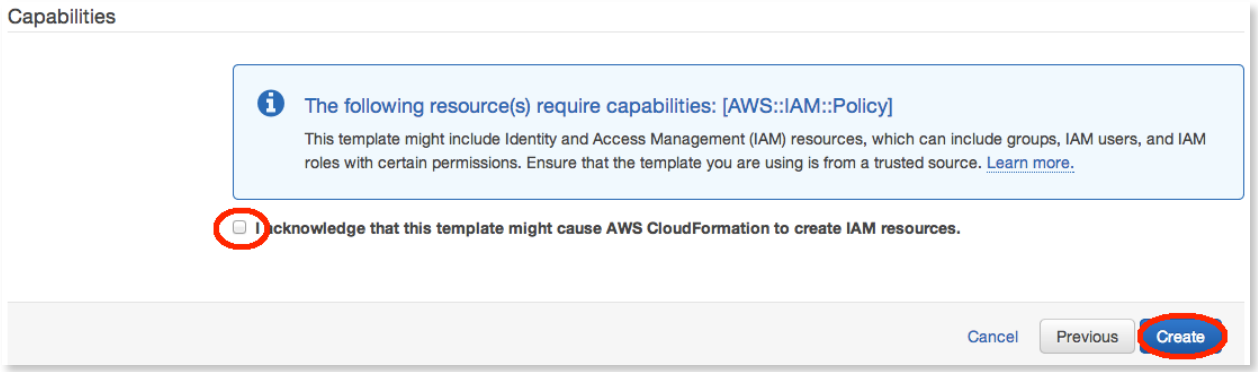
On this page, leave the defaults. You don't need to add a key name at this point. Add the Kinesis Stream name you created in step one and click "Next".



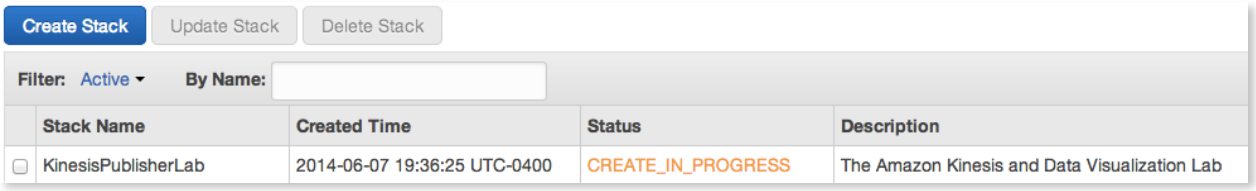
You don't have to add tags at this point, but it's a good best practice to do so. You could do something like "Name" for the Key and "Kinesis Lab" for the Value. Many users will further tag with Keys like "Environment" or "User" with respective Values. Once you have entered a tag (if you want), click "Next".



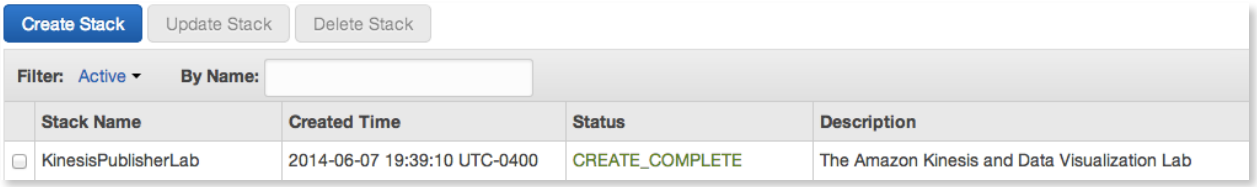
On the final page, scroll down and make sure to select the acknowledgement check box. Then click "Create".



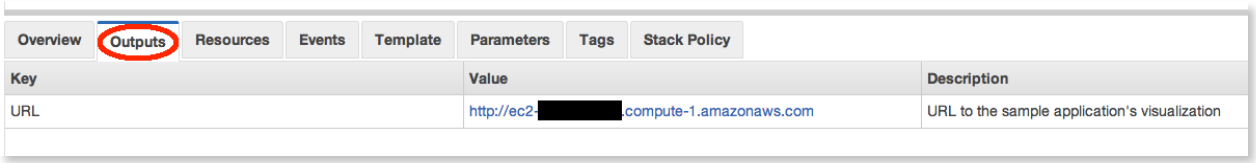
The Cloudformation console will then load and you will see this:



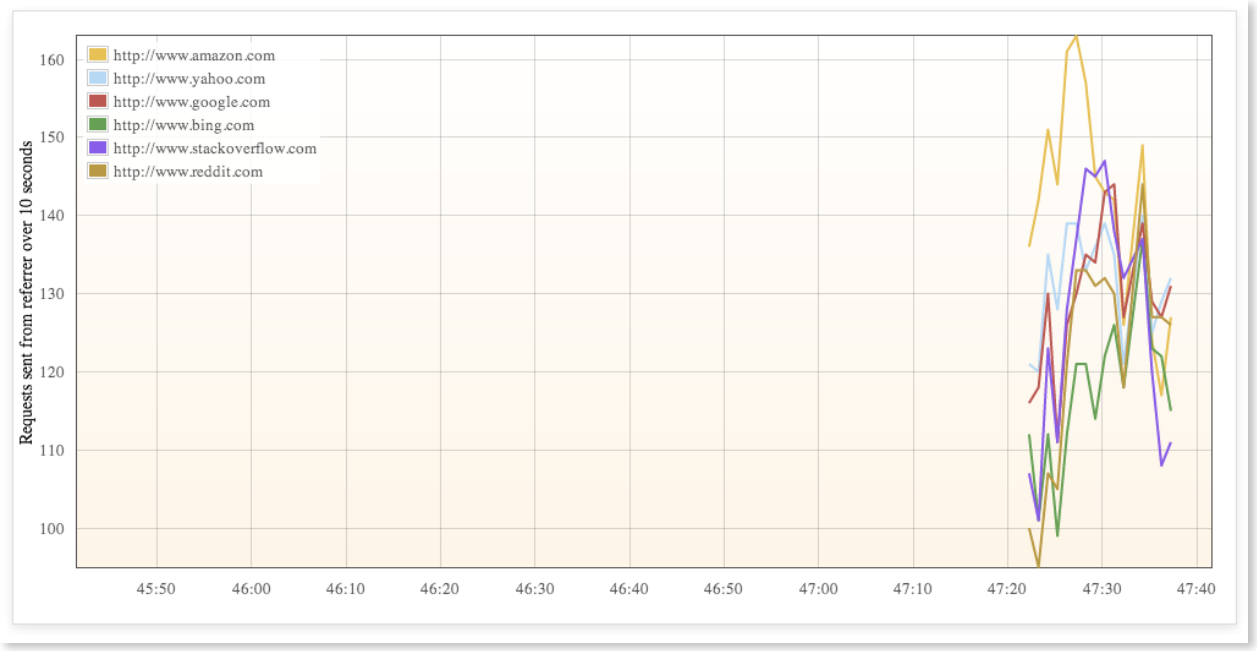
Wait until the status changes from “CREATE_IN_PROGRESS” to “CREATE_COMPLETE” as below:



To verify the Cloudformation stack launched successfully, go to the “Outputs” tab and select the URL in the output and open it in a new browser tab/window.



On the page that you open, you will see a visualization of the data like below:



Congratulations! Kinesis is ingesting data from the producer and the visualizer is reading data from the Kinesis Stream and displaying it.

Lab 2: Amazon Redshift

In this section you will create an Amazon Redshift database cluster, another EC2 Kinesis Client Library application and use the Kinesis Connector library in that application to write to the Redshift database. If you haven't already, make sure you install a client that will be able to communicate with Redshift. SQL Workbench/J is recommended as it will work on any system with Java. The instructions for installing it are here:

http://docs.aws.amazon.com/redshift/latest/mgmt/connecting-using-workbench.html

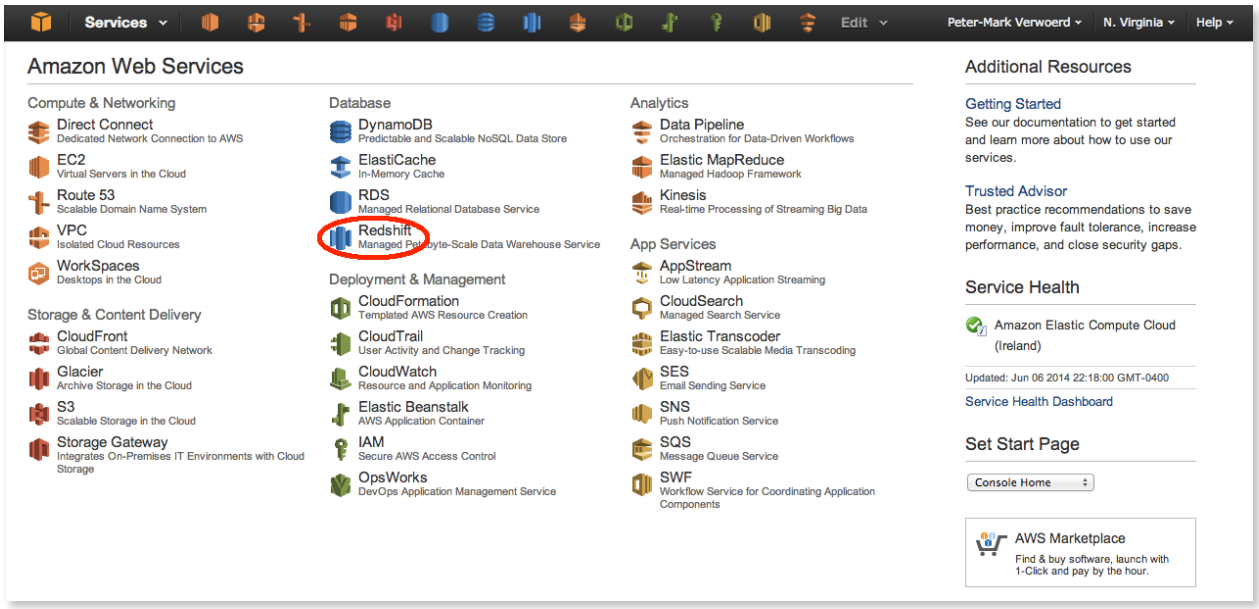
You may be required to download the Postgres JDBC Driver. It is available here:

http://docs.aws.amazon.com/redshift/latest/mgmt/configure-jdbc-connection.html

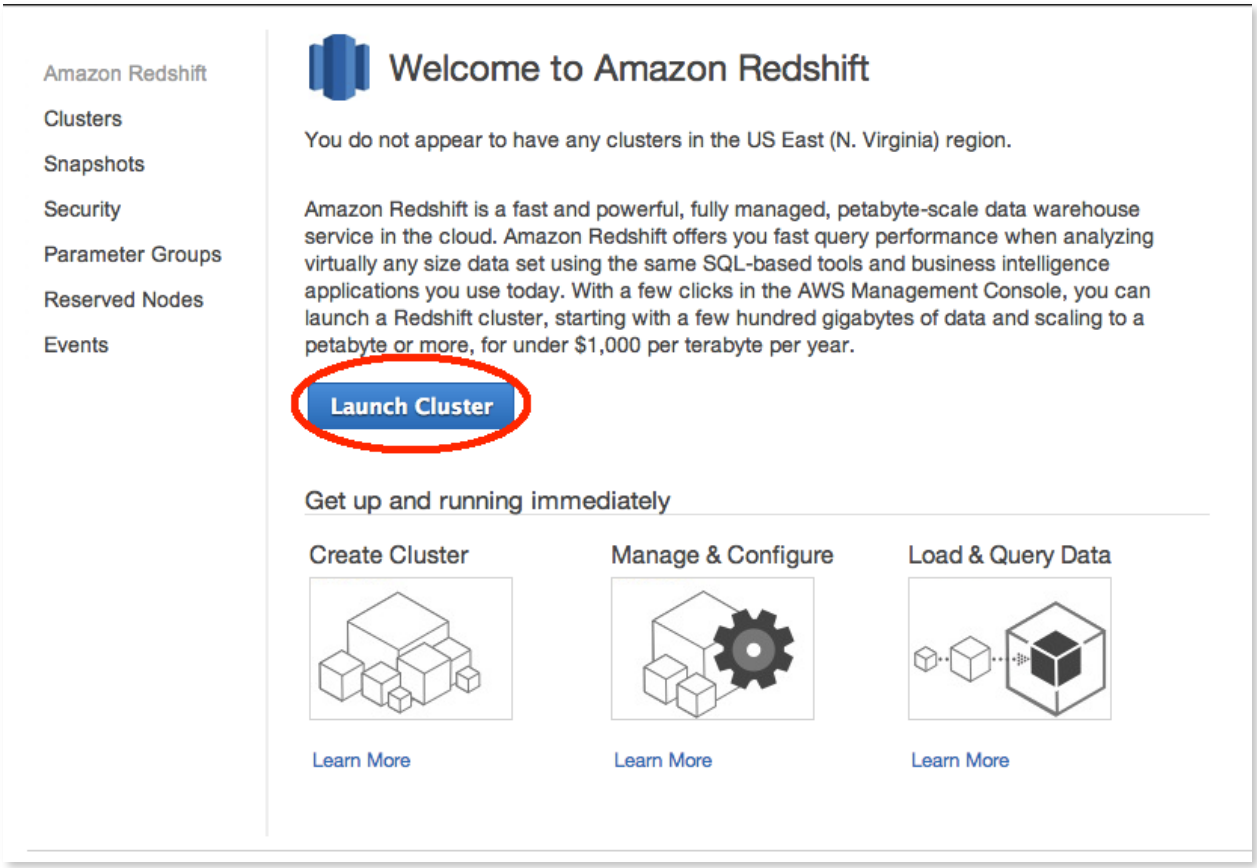
You will be able to test if it has been successfully installed at the end of Step 1.

Step 1: Create the Redshift Cluster

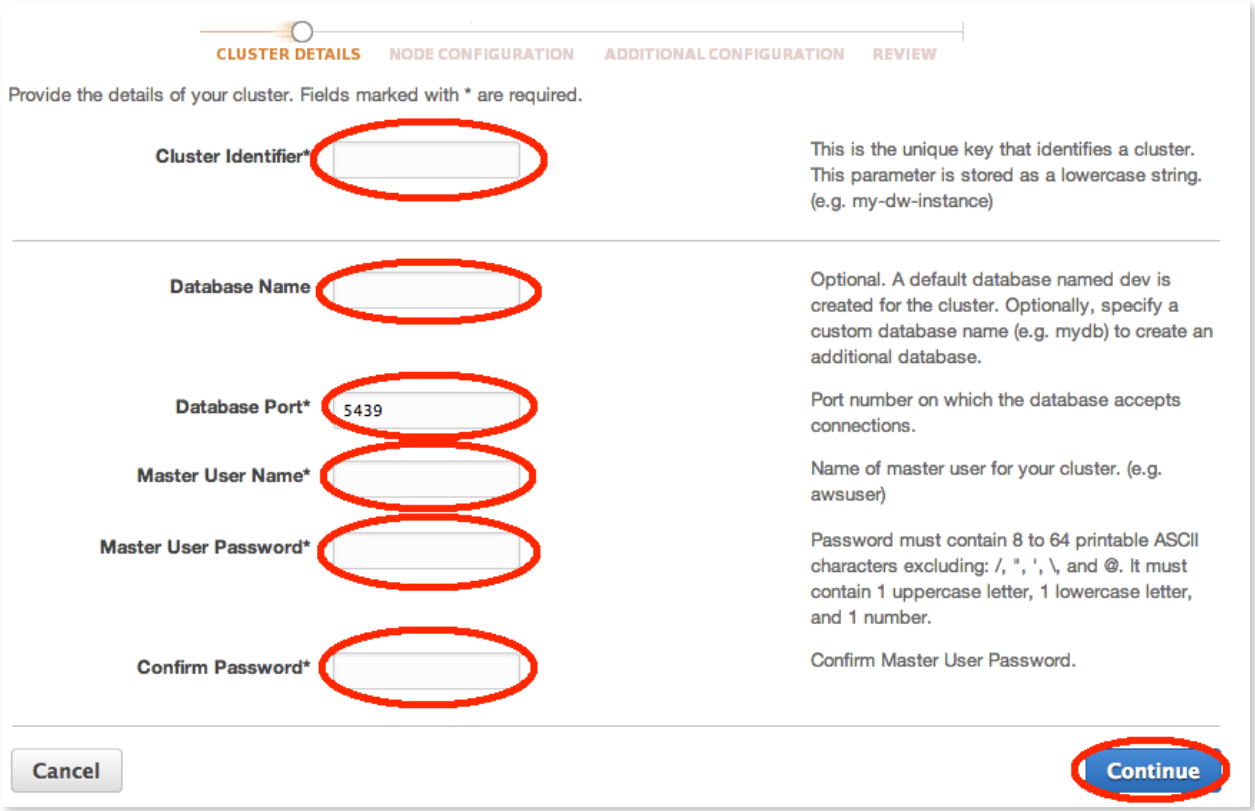
Go back to the Amazon Web Services console main page and select Amazon Redshift:



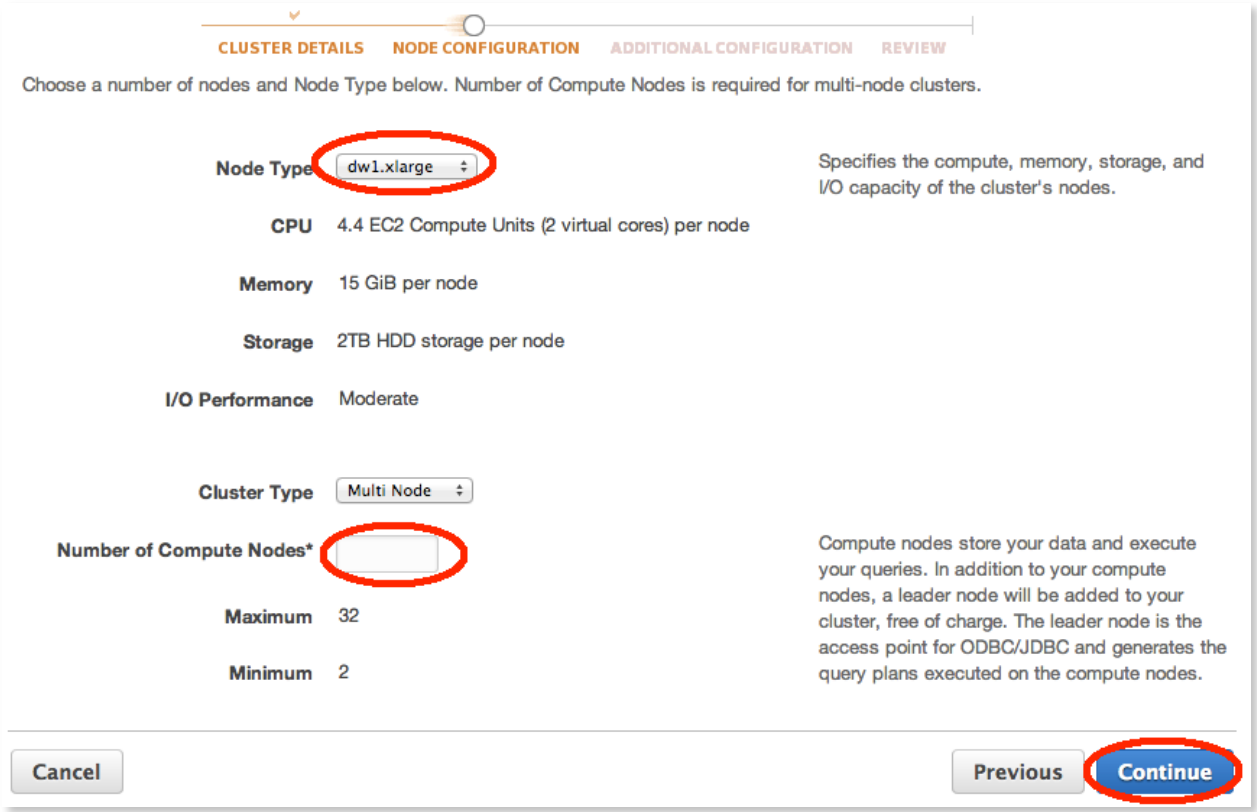
Click “Launch Cluster”



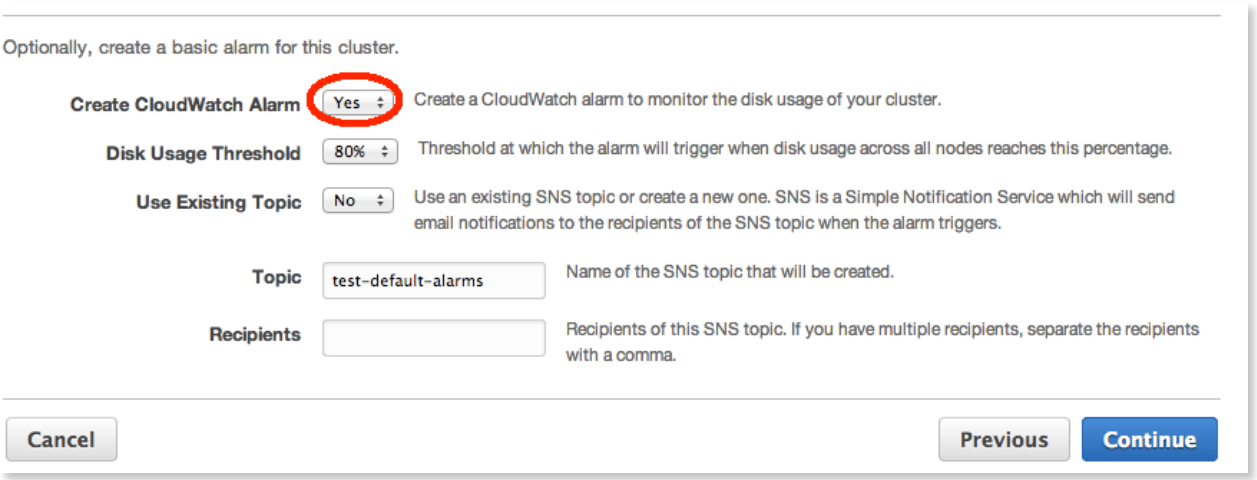
On this page you will set the parameters necessary to start your cluster. Give your cluster an identifier - any will do, within the limits described on the page. Then give a database name. Though it's optional, you may want to give it a name you find it easy to remember. You may change the port from the default port if you wish or if your laptop will block connections on 5439 (this can happen). Port 8192 is a common alternative. Provide the username and password within the limits described. Be sure to remember your password - it won't be displayed anywhere. Click "Continue".



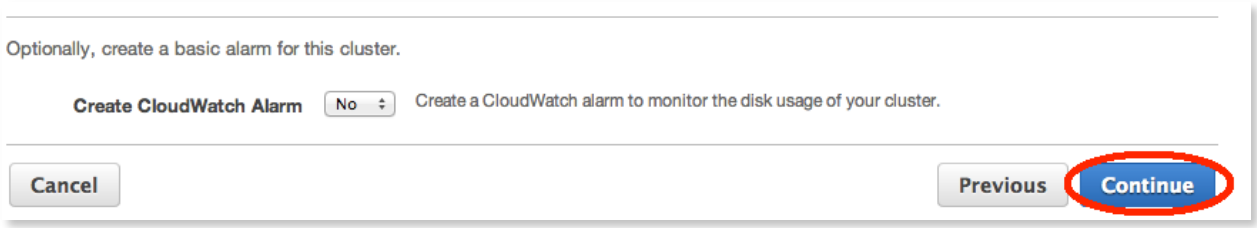
On the next page, you will be able to select the size and type of your cluster. The 4 instance type options are: dw1.xlarge, dw1.8xlarge, dw2.large, dw2.8xlarge. The difference between the dw1 and dw2 types is the type of storage. The dw1 family have magnetic hard drives while the dw2 family have SSD, or flash, hard drives. For this lab, you should select the dw2.large. You will also need to add a number of nodes of the cluster. 2 nodes are more than sufficient for the work you will be doing. Once entered, click “Continue”.



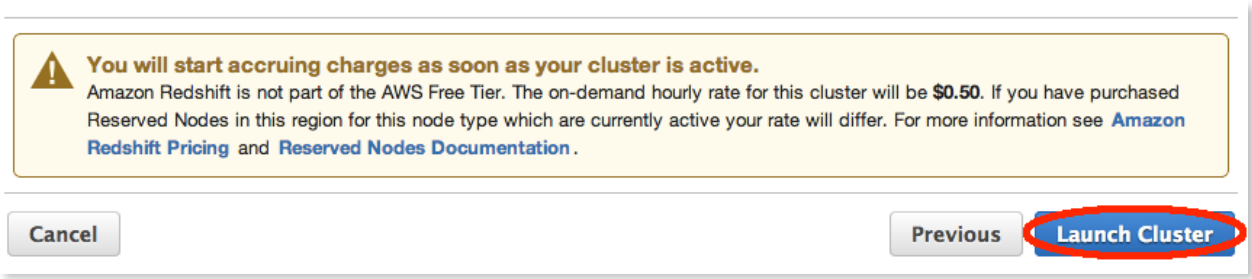
On the next page, scroll down to the bottom. Change the “Create CloudWatch Alarm” option from Yes to No. When using Redshift normally, you should leave this enabled. An alarm that will alert you between when disk usage is between 70% - 80% is a good best practice. But since this lab does not come close to that, and it will slow down the process, deselect if for now.



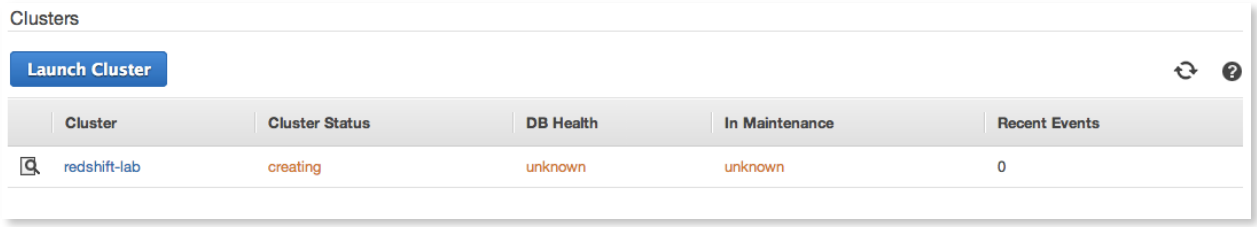
When you've changed it to "No", then click "Continue".



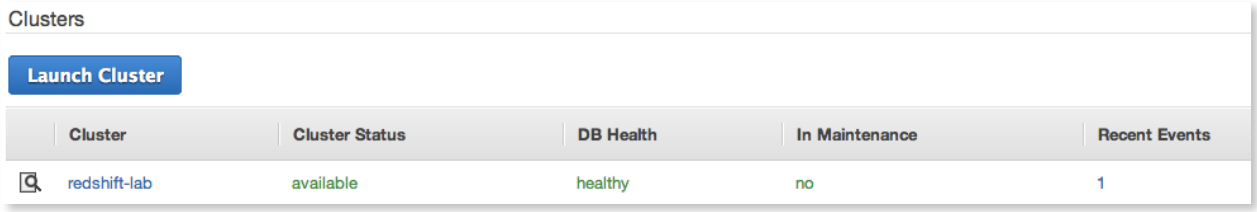
Scroll to the bottom of the next page. If you've selected 2 dw2.large instances for your cluster, you will see the same price as in the screen shot below. Click "Create Cluster".



Once you return to the list of Redshift Clusters, the status of your new cluster will be "creating". The creation process can take several minutes.



Once it becomes "available", as below, finish the walkthrough from the documentation to create a JDBC connection to your cluster. Once you have successfully connected, leave the connection open and proceed to Step 2.



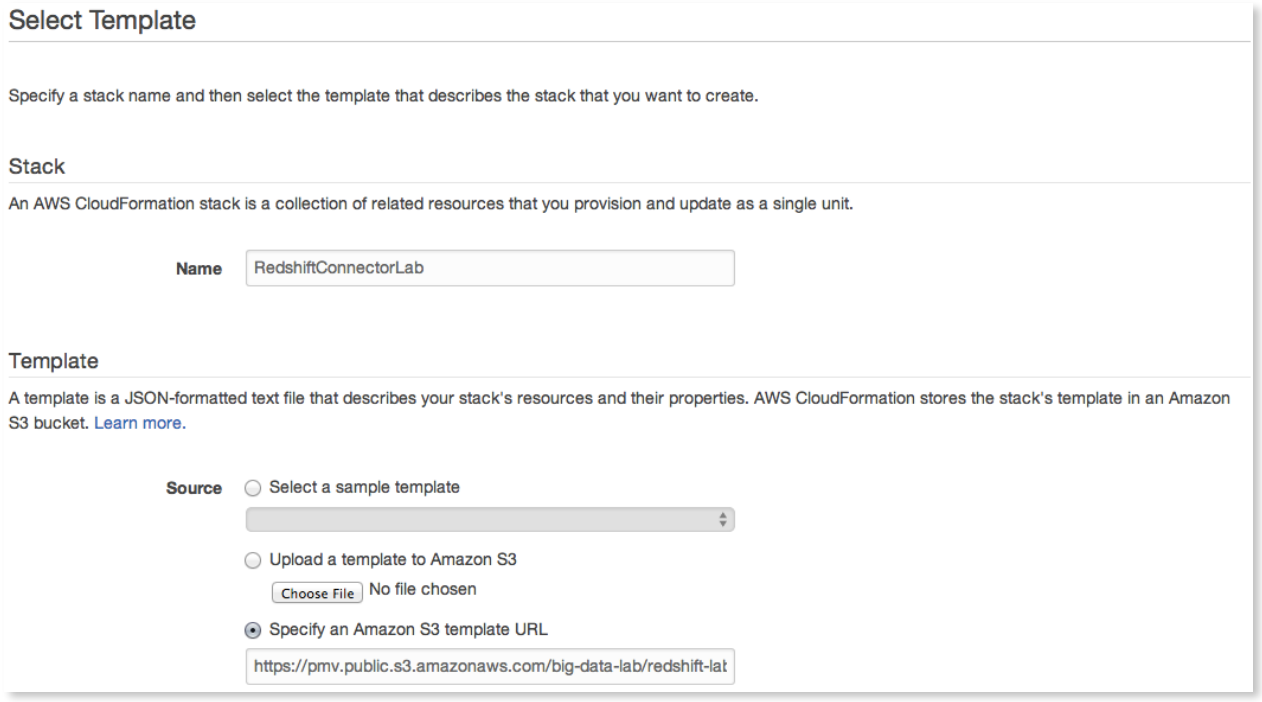
Step 2: Create the Redshift Connector

In this section, as in Step 2 in the first lab, you will use Cloudformation to launch the Redshift connector on EC2. Click on the link below:

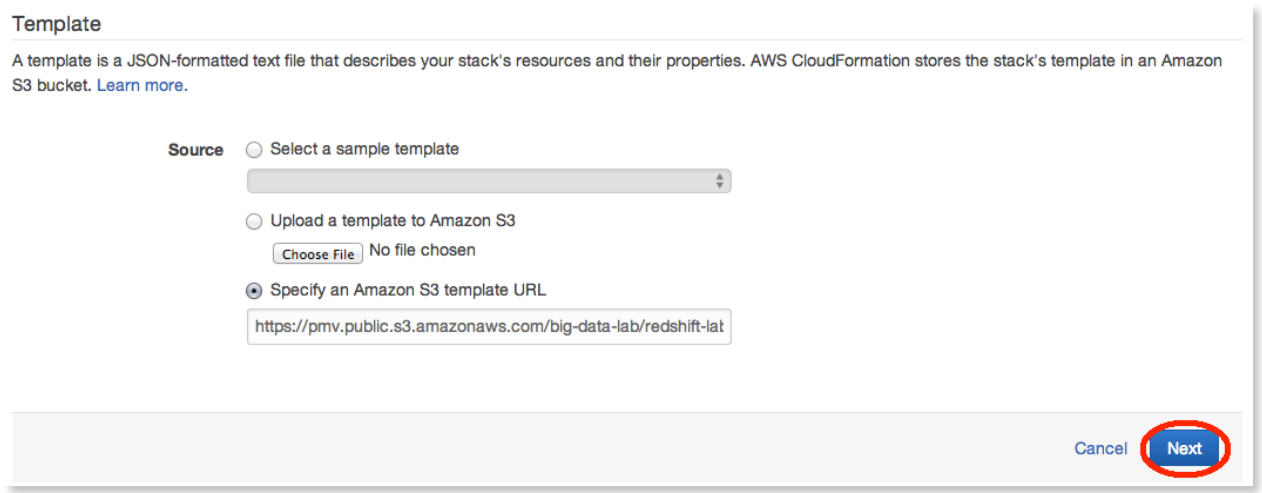
RedshiftConnectorLab

to begin.

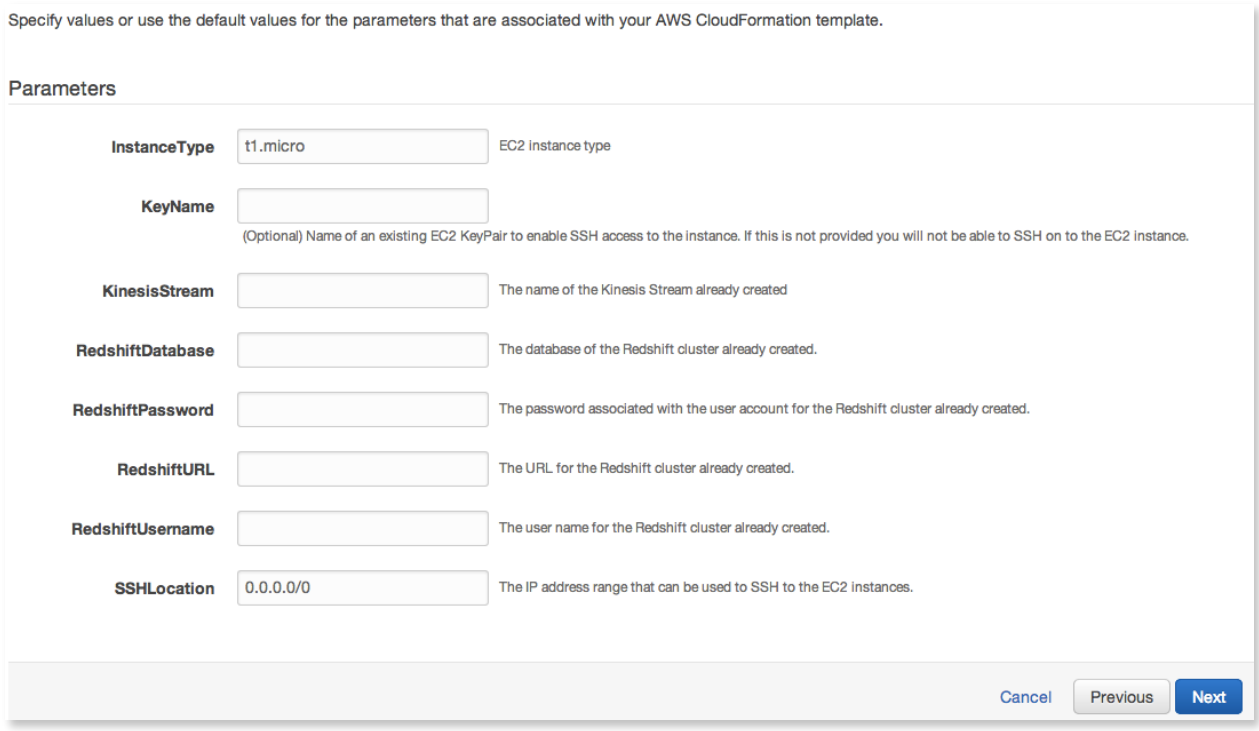
This will open the AWS Console to the Cloudformation page.



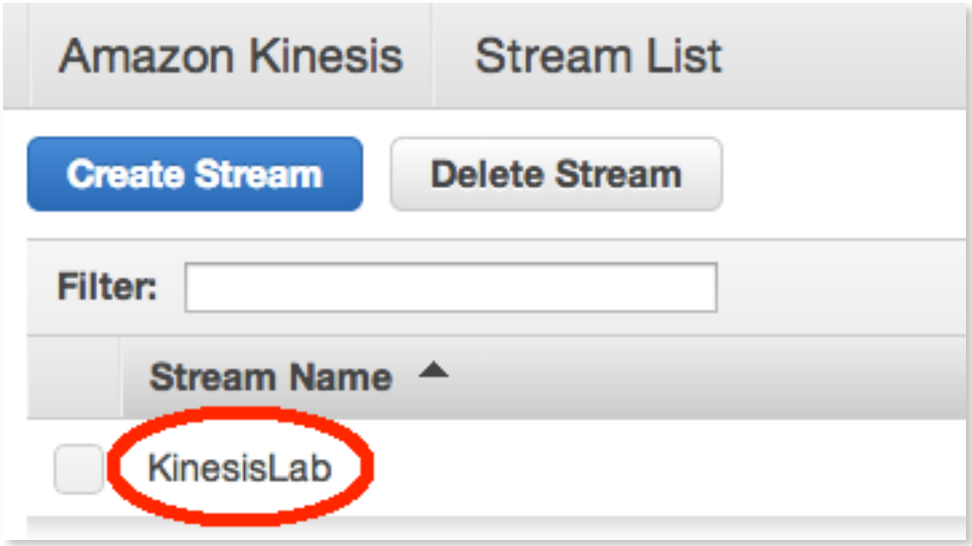
Scroll down and click



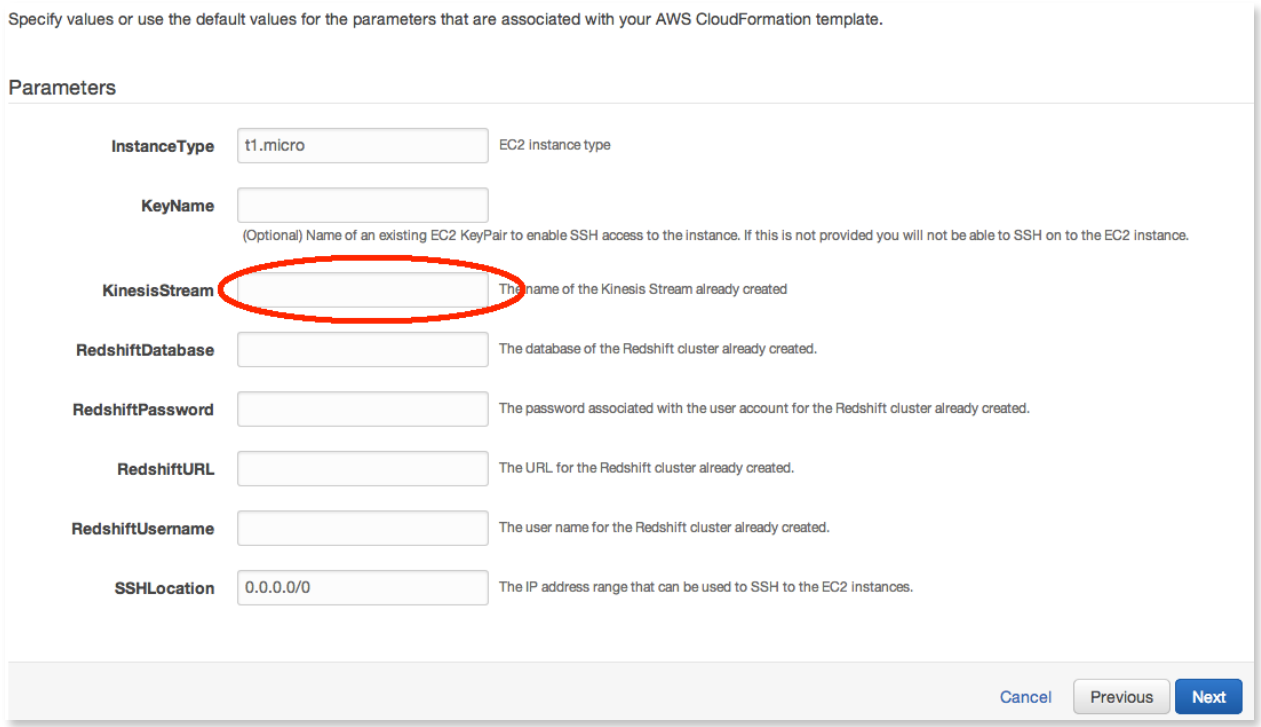
On this page, you will configure everything that the Redshift connector application needs to connect to the Redshift cluster. You will not need to change the Instance Type, add a KeyName. or change the SSH Location. If you are familiar with any of those parameters however, feel free to change them.



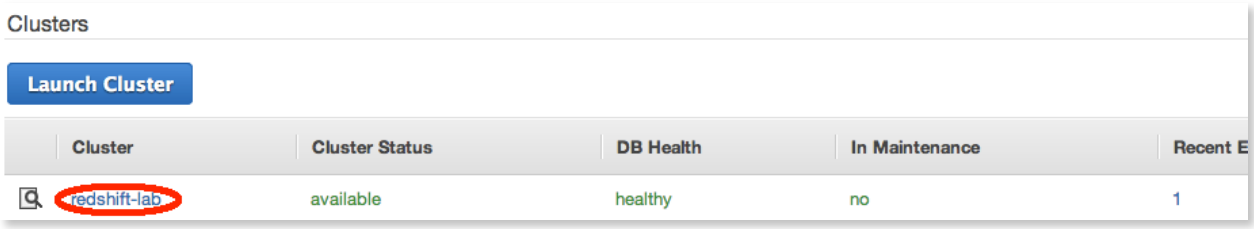
First, add the name of the Kinesis Stream. If don't remember it, you can open the console in a new tab, and navigate to Kinesis again.



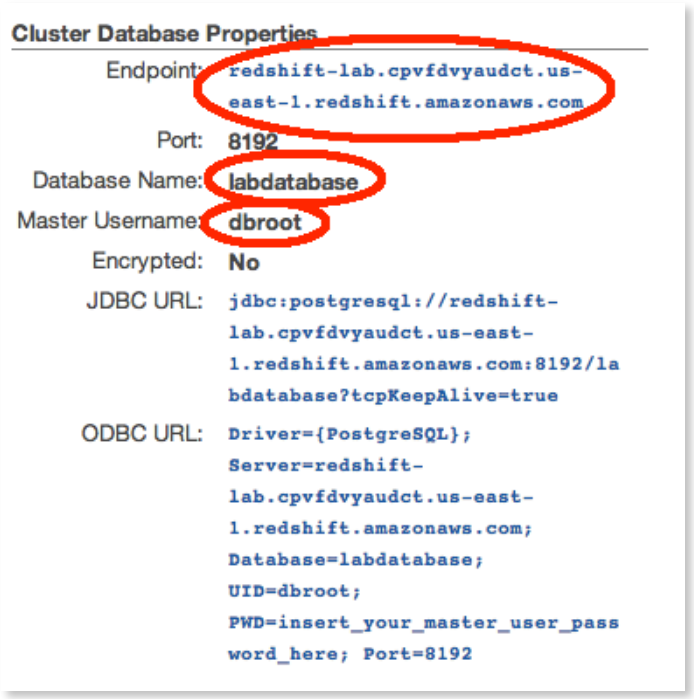
Of course, use the name of the Kinesis Stream you created in the field selected below:

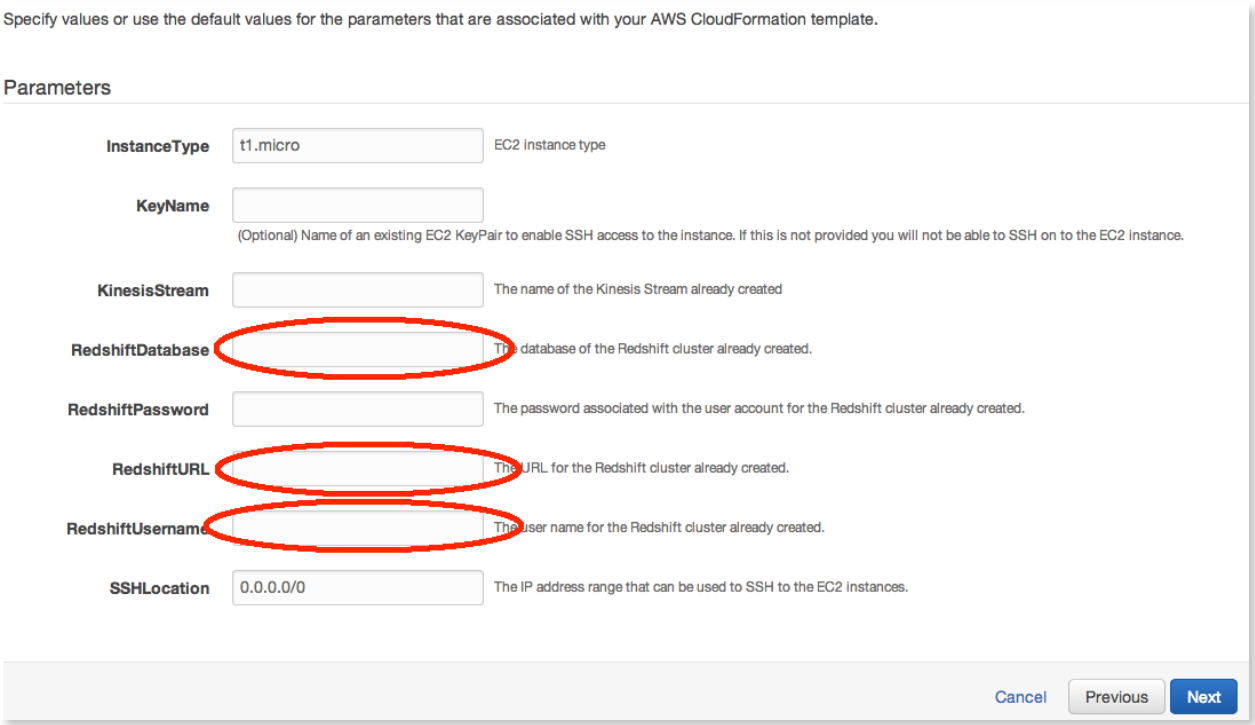


Next, add the information about your Redshift cluster. From the AWS Console home page, navigate to the Redshift page. Select the cluster you created.

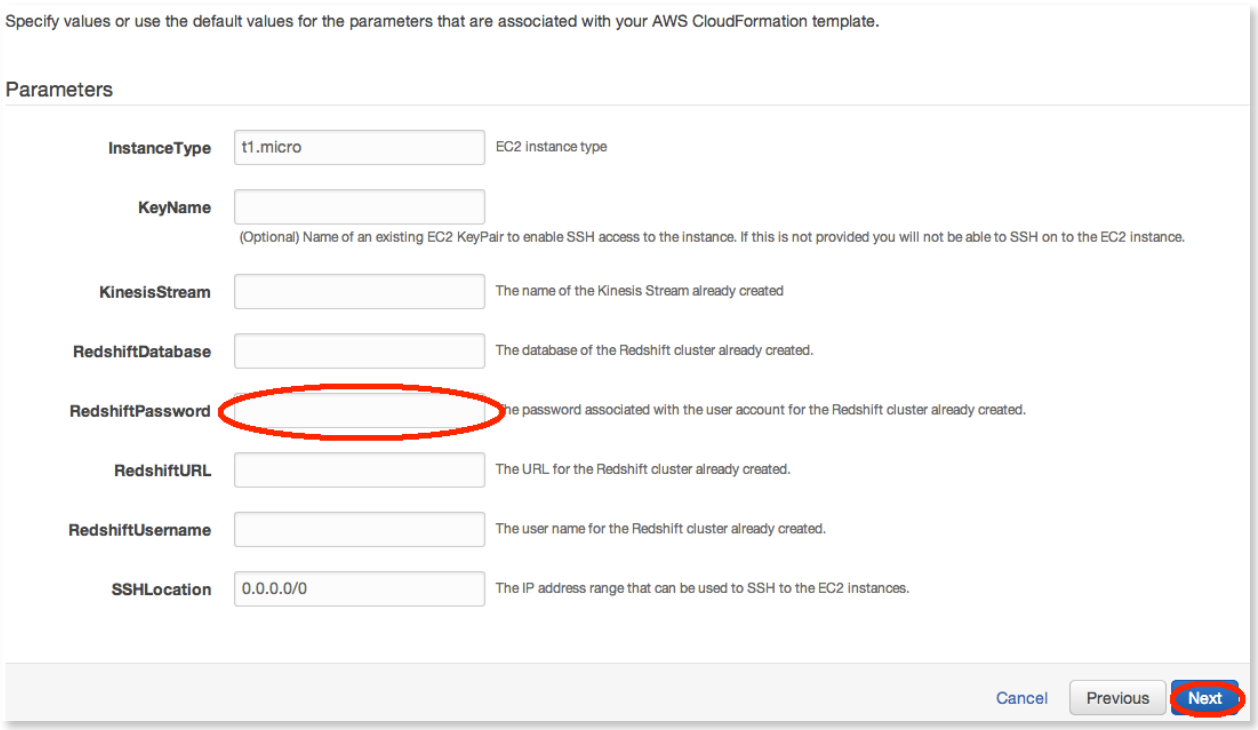


Scroll down to the middle of the page so you can see the “Cluster Database Properties”. The name that you need to add to the Cloudformation template is the “Database Name” - importantly, **not** the Cluster Name. The “Redshift URL” is the “Endpoint”. And the “Redshift Username” is the “Master Username”.

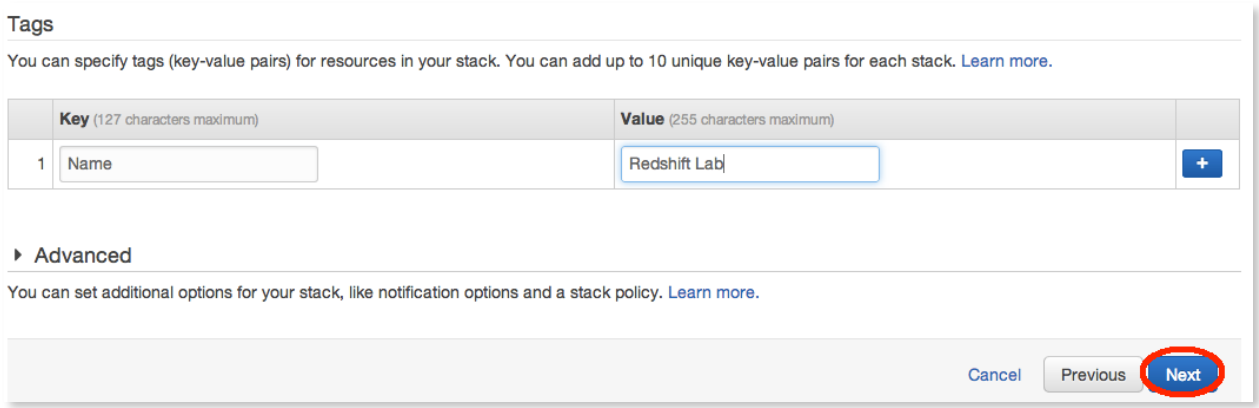




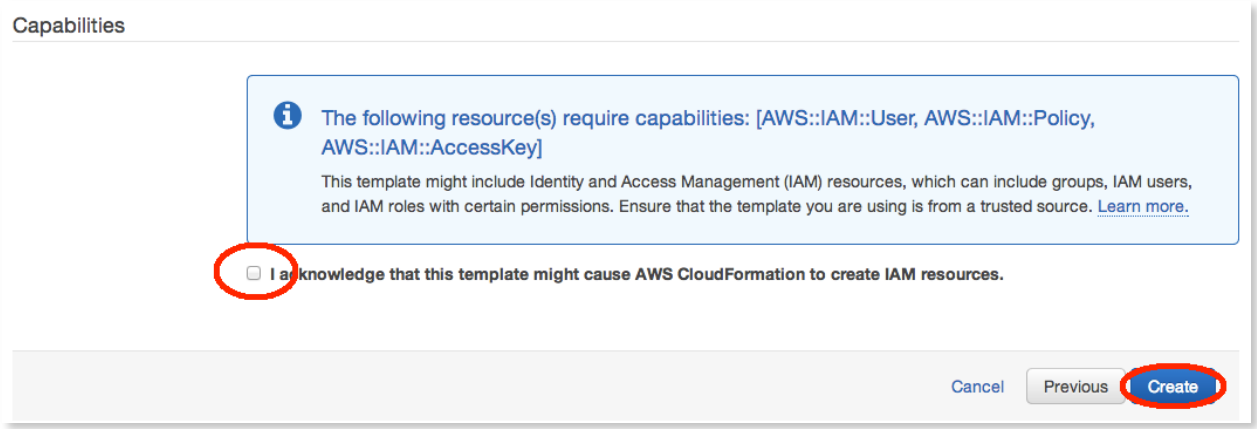
The Redshift Password is the password you provided when you created the Redshift cluster. After you add it, click “Next”.



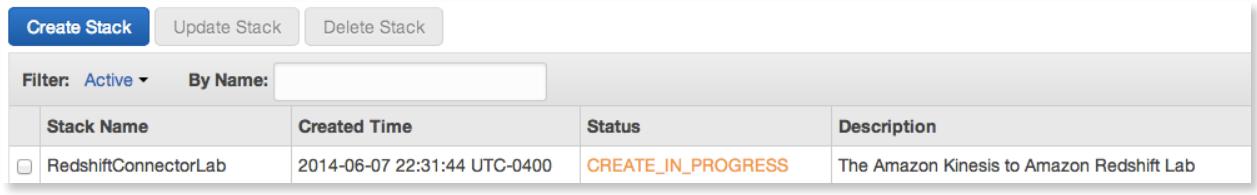
As before, it is not necessary to add tags to this Cloudformation stack, but again, it is a good practice to do so. Once you have done so, or not, click “Next”.



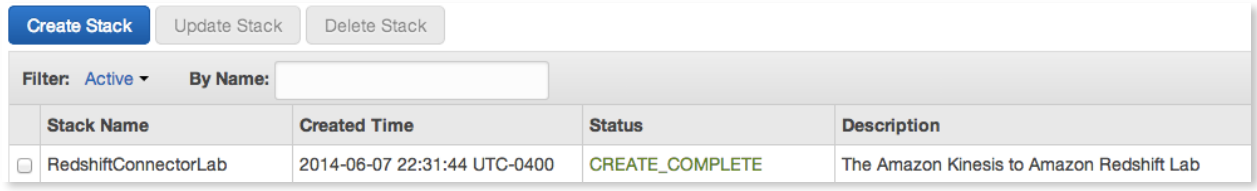
On the final page, don't forget to click to acknowledge the creation of IAM permissions and then click "Create"



You will then return to the Cloudformation console and the status of your Cloudformation stack will be "CREATE_IN_PROGRESS".



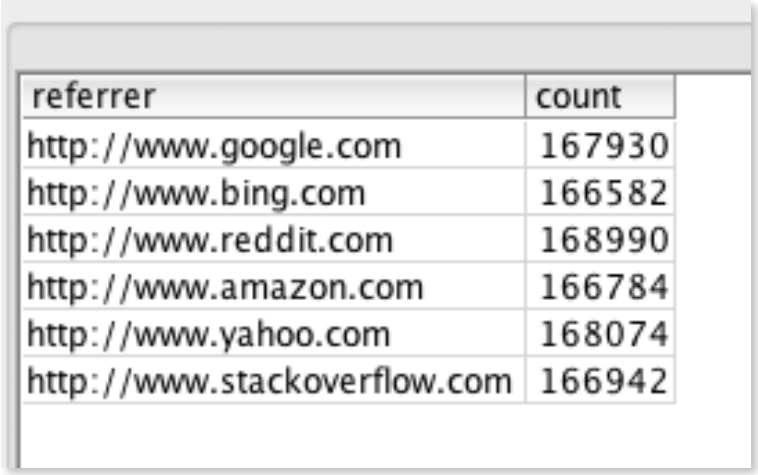
Once the status has gone to "CREATE_COMPLETE", switch to your SQL client.



In the SQL client, run the following statement:

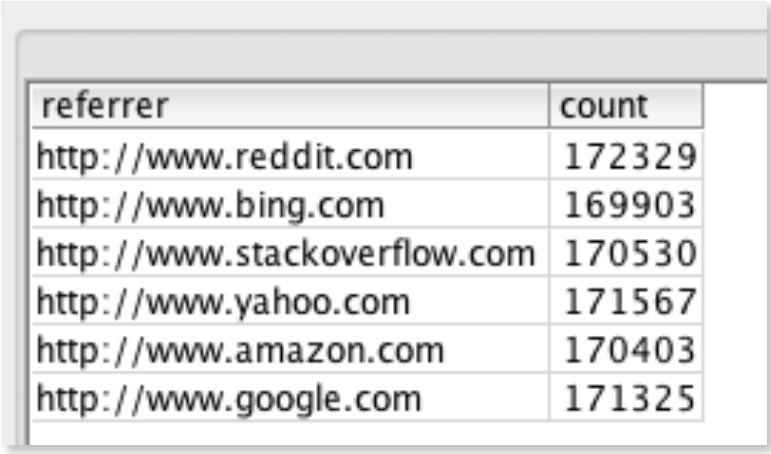
```
SELECT referrer, count(*) FROM Kinesisbasictable WHERE resource LIKE '%index
%' GROUP BY referrer;
```

You should get results in the form of the following:



referrer	count
http://www.google.com	167930
http://www.bing.com	166582
http://www.reddit.com	168990
http://www.amazon.com	166784
http://www.yahoo.com	168074
http://www.stackoverflow.com	166942

Try running the query again a minute or so later to make sure the numbers are increasing:



referrer	count
http://www.reddit.com	172329
http://www.bing.com	169903
http://www.stackoverflow.com	170530
http://www.yahoo.com	171567
http://www.amazon.com	170403
http://www.google.com	171325

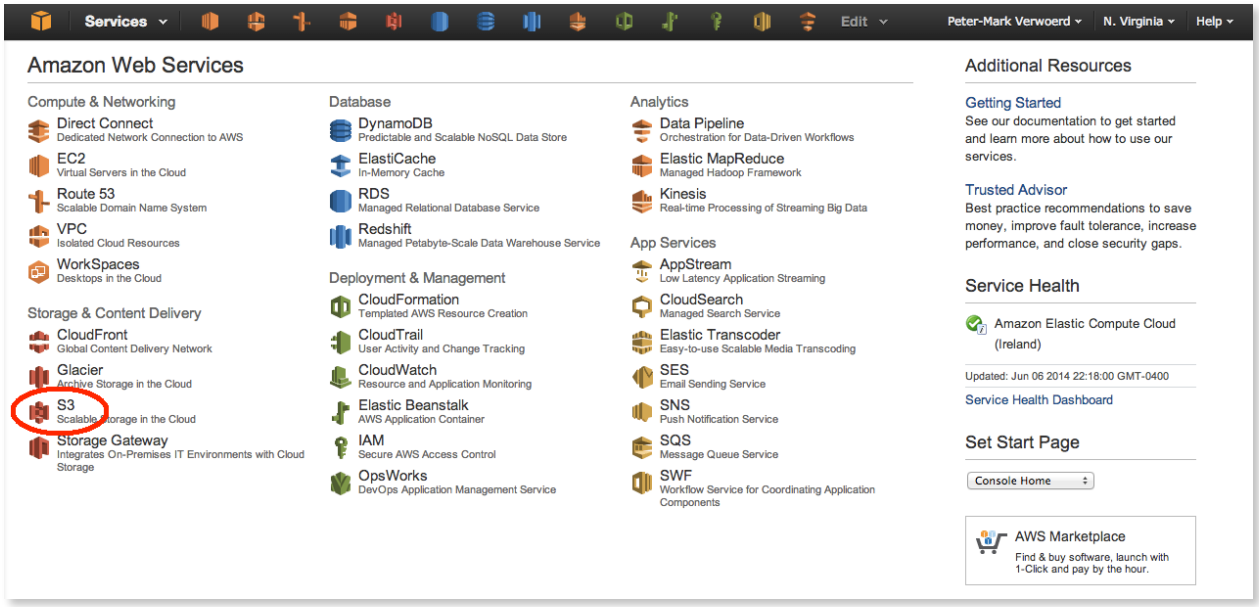
If the numbers are increasing, then congratulations! You now have a producer sending data to a Kinesis Stream, being read by a Kinesis Client Library enabled application, which is using the Kinesis Redshift Connector to write to a Redshift data warehouse. Feel free to experiment with querying the data in the Kinesisbasictable.

Lab 3: Amazon Elastic Map Reduce

In this section you will create an S3 bucket to keep a small amount of fake “user” data. Then you will create an Elastic Map Reduce (EMR) cluster to analyze and transform the user data. Then add the transformed data to your existing Redshift cluster.

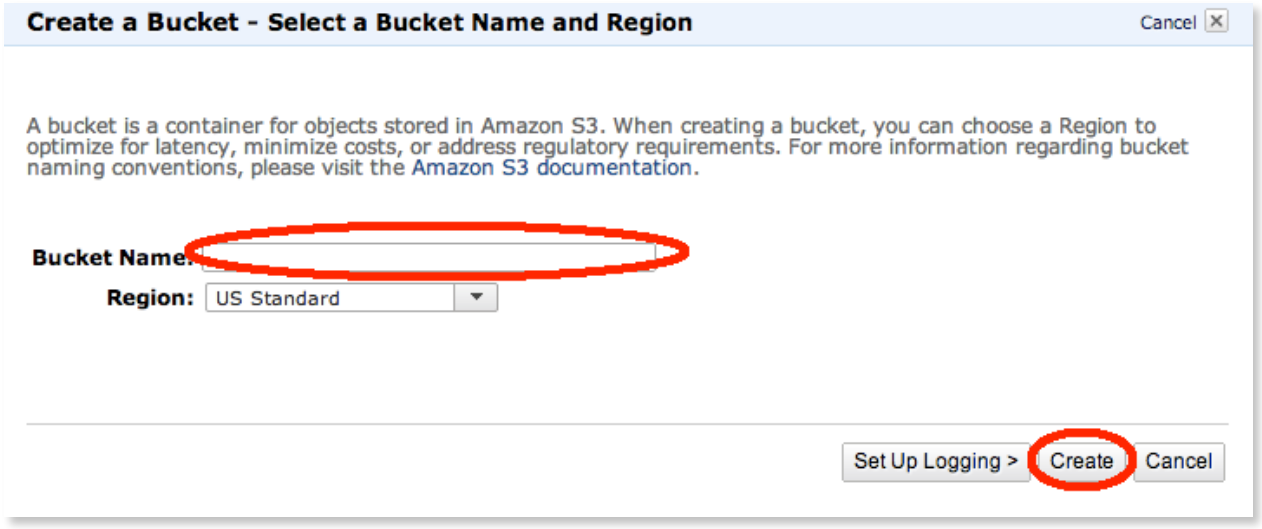
Step 1: Create the S3 bucket

Go to the AWS Console home page and navigate to the S3 page.

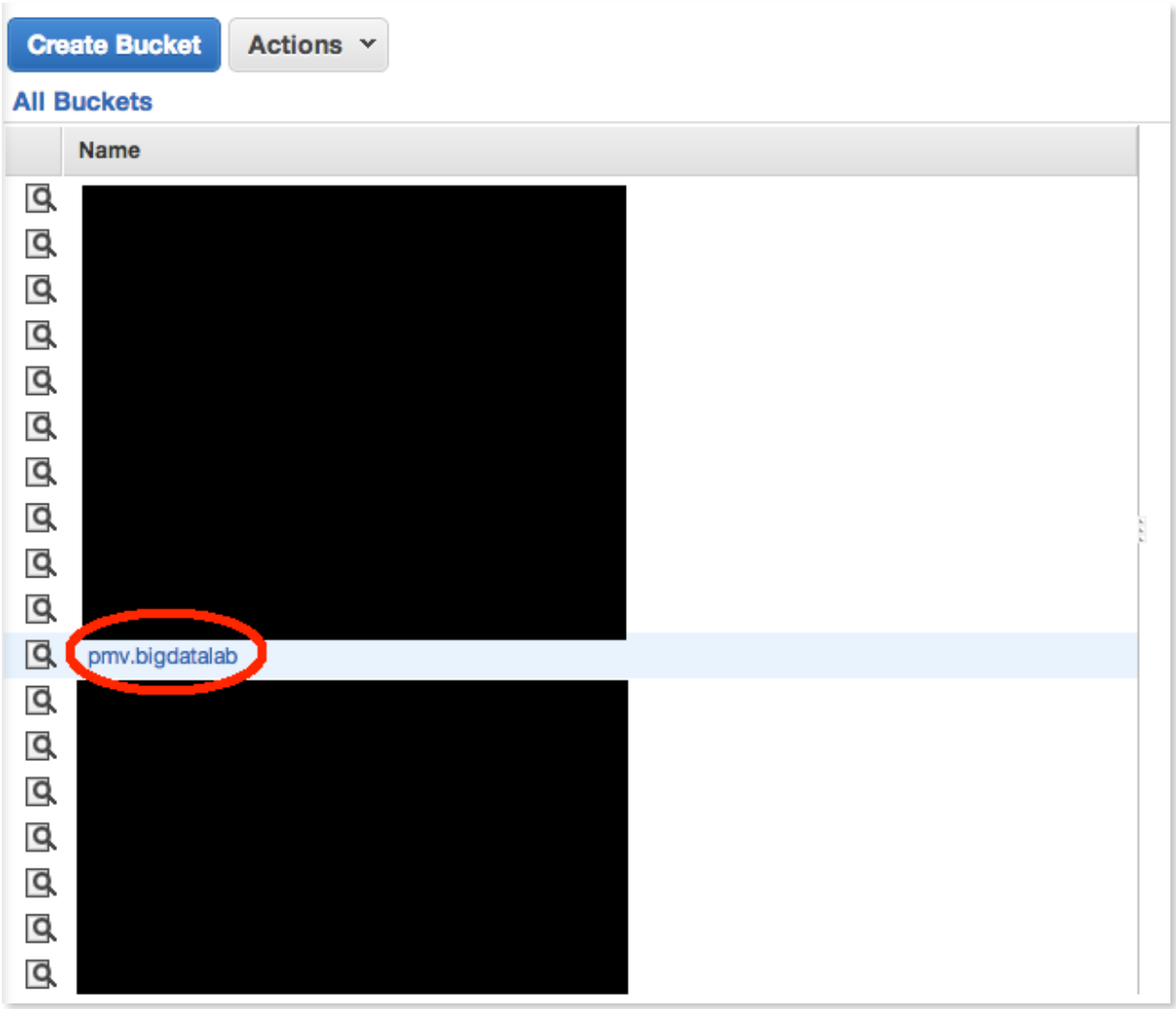


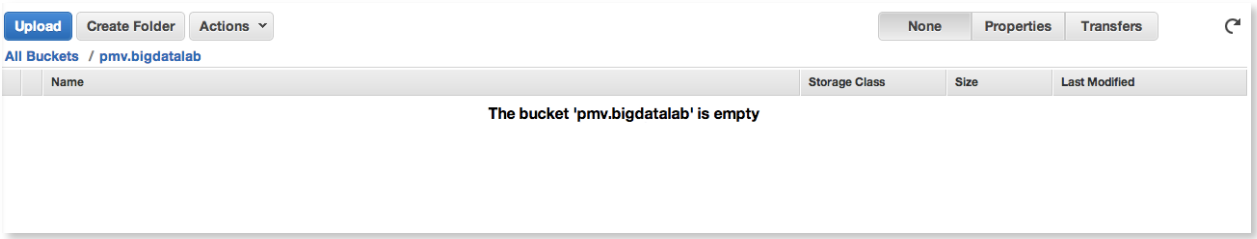
Here, click “Create Bucket”

Give the bucket any name you wish, just make sure it only contains letters, numbers, hyphens or periods. Also make sure the region you select is “US-Standard”. For historical reasons, this is a different name than the rest of the region in US-East, but it is the corresponding S3 region for US East. Also for historical reasons, US Standard allows a wider variety of names for its buckets than the rest of the regions. For the purposes of this lab and for best practices, it’s recommended to use the stricter naming policy. Logging isn’t necessary at this point. Once you’ve named your bucket, click “Create”.

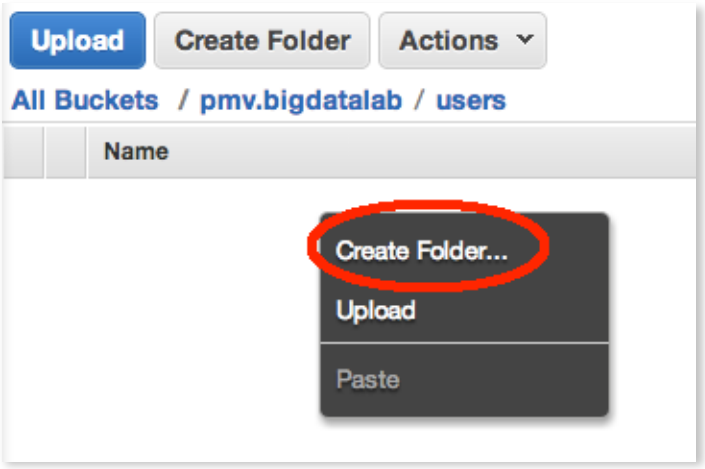


On the S3 console, select the bucket you've just created.

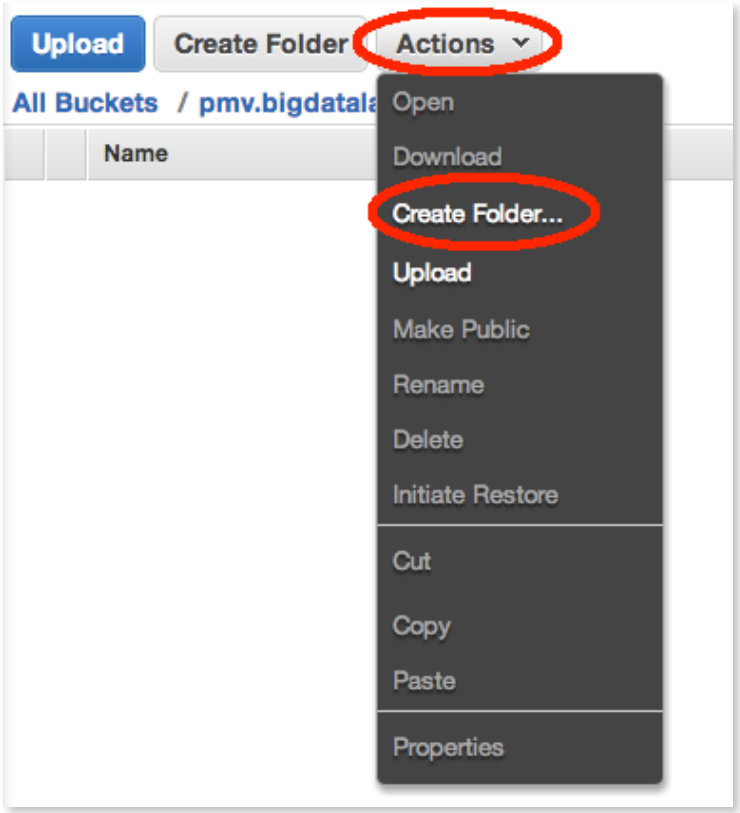




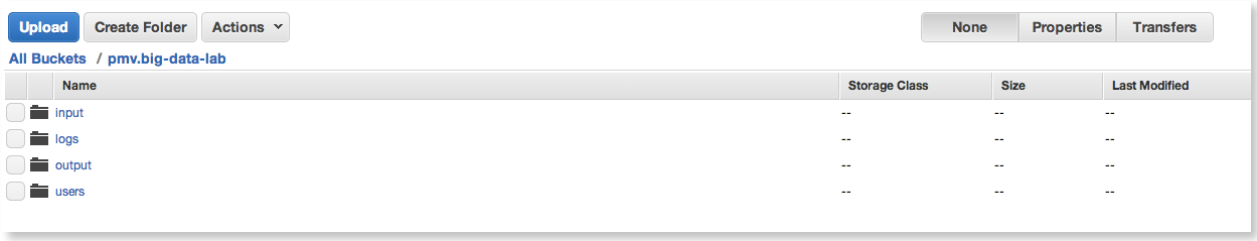
In that bucket, create 4 folders. To create a folder, you can either right click



or use the menu at the top left and select the “Create folder” option.



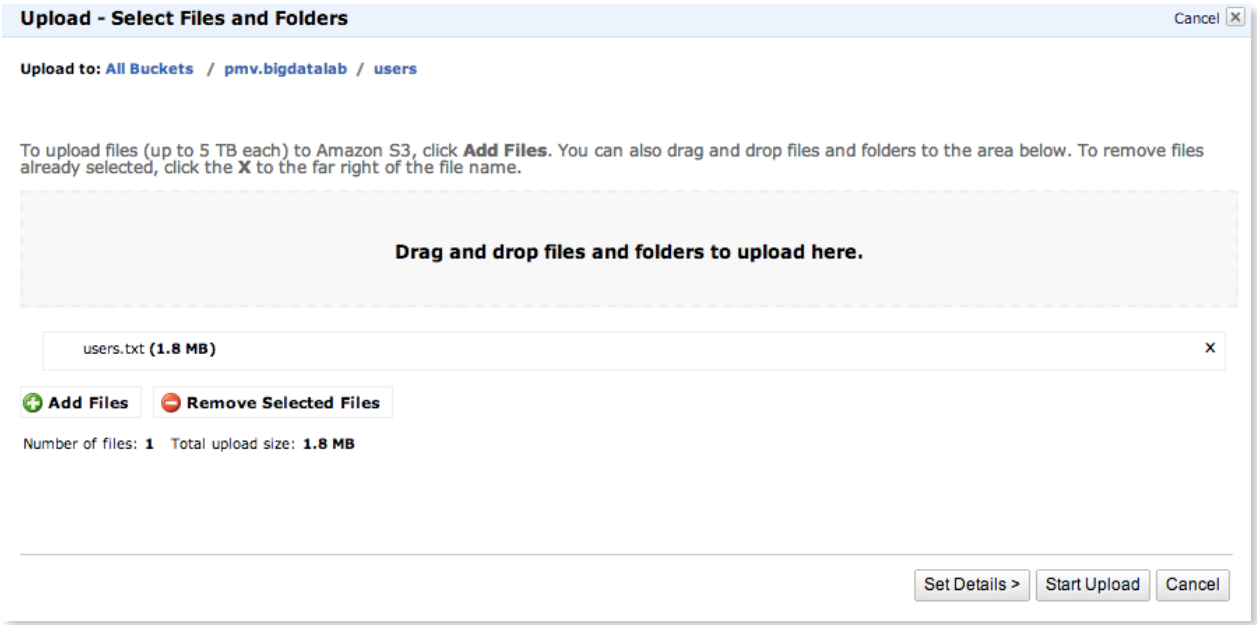
The 4 folders you should make are “input”, “logs”, “users”, and “output”.

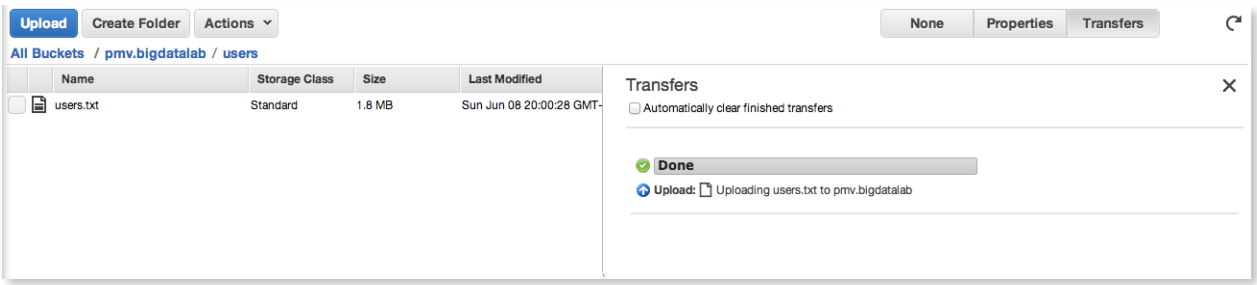


Next, download the following link to your local desktop - it’s the fake user data you will be analyzing in this part:

bit.ly/awsbdlab2

Now, add it to the “users” folder in your S3 bucket.

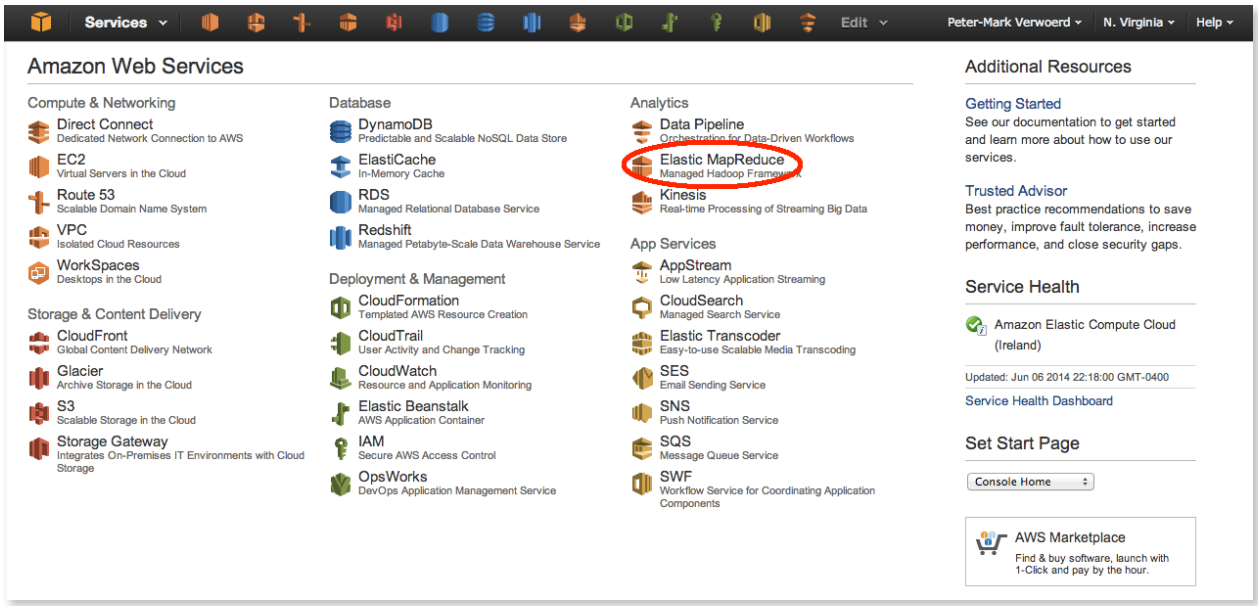




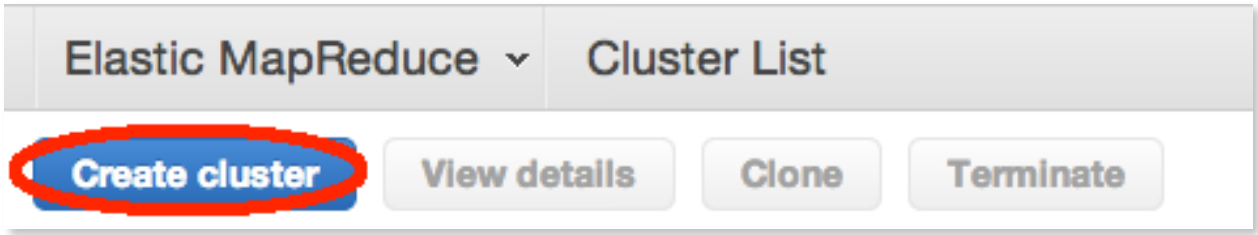
Your S3 bucket is set up and ready to use now.

Step 2: Create the EMR cluster

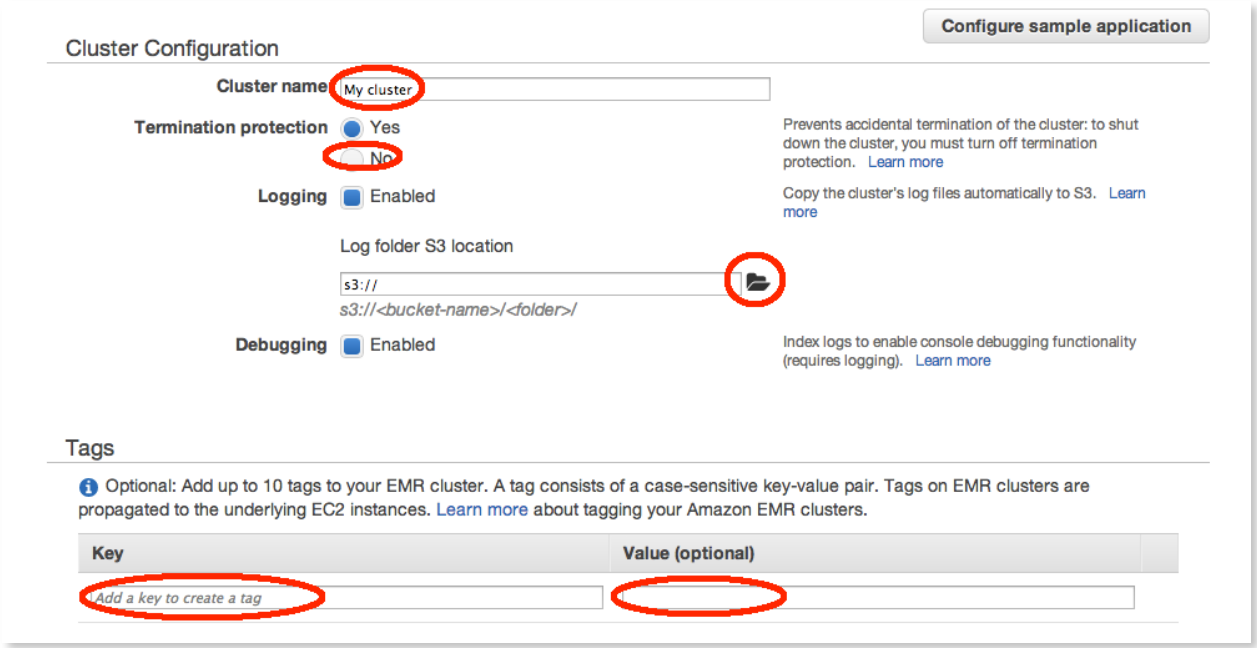
Go to the AWS Console home page and navigate to the EMR page.



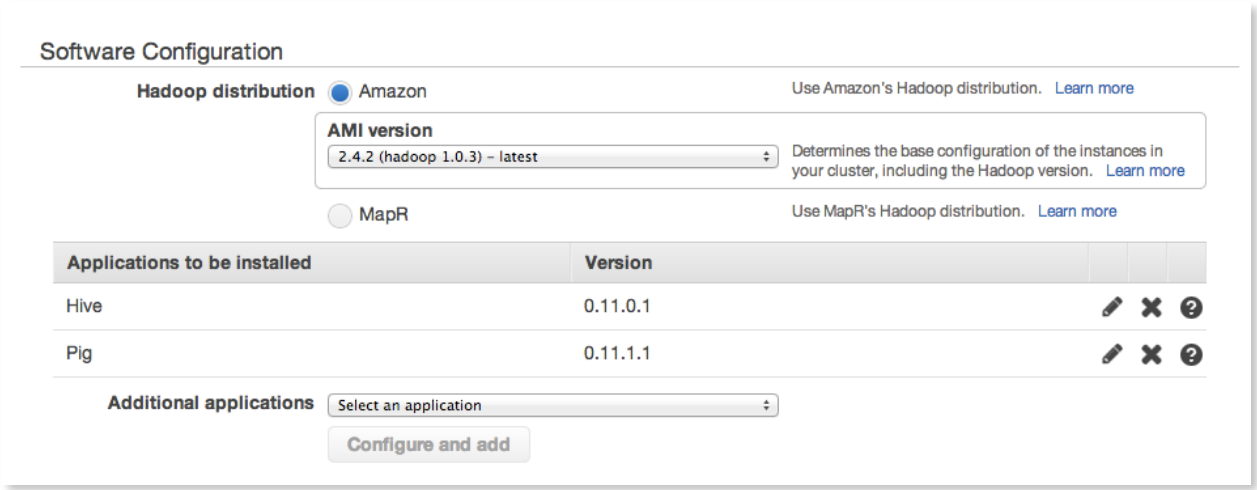
On the EMR console page, click “Create cluster”



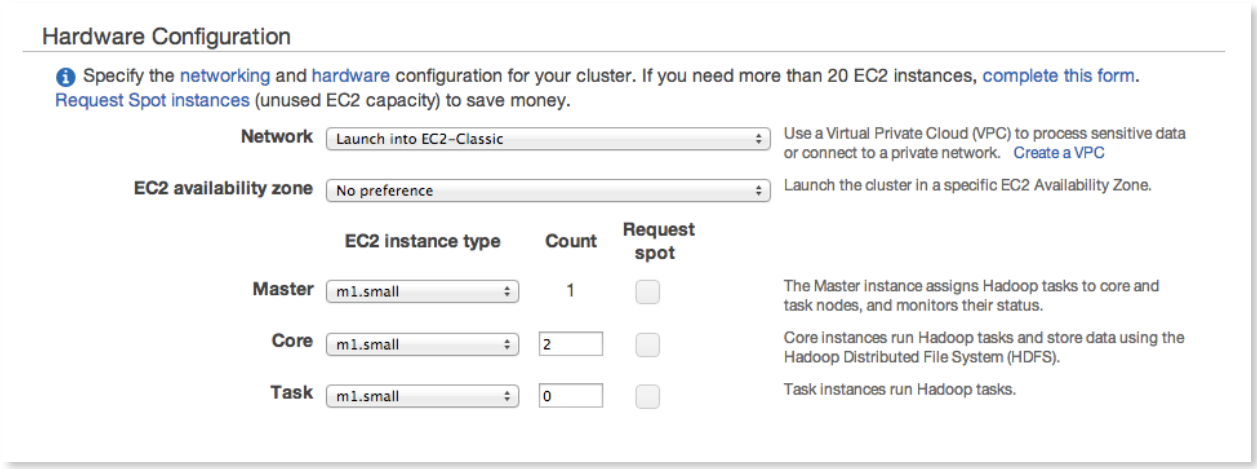
On this page you will define your EMR cluster. Give it any name you wish. You will not need Termination protection for this lab, so you can switch that option. Keep logging enabled - this is a best practice as well as helping debugging, if necessary. You can select the bucket you created in Step 1, in the logging folder. As before, you don't have to use tagging, but it is a best practice that is recommended.



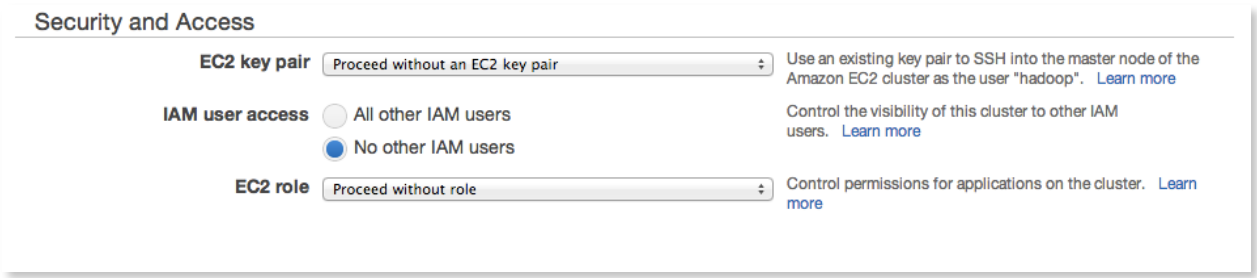
The rest of the setup is fine with the defaults, but just to go over whats there. The first section below Tagging is the software configuration. The AMI version is the Amazon release of each version of Hadoop that we support. Currently, the default is Hadoop 1.0.3, but there is also support for several versions of Hadoop 2. We also support the MapR Hadoop distribution. Hive and Pig are both installed by default on new clusters. Since you will be using them both, leave them as is. You could also optionally add HBase, a database, and Ganglia, distributed monitoring, here, but they aren't necessary. With Hadoop 2, you could also install Impala, a distributed query engine, but this is also not necessary.



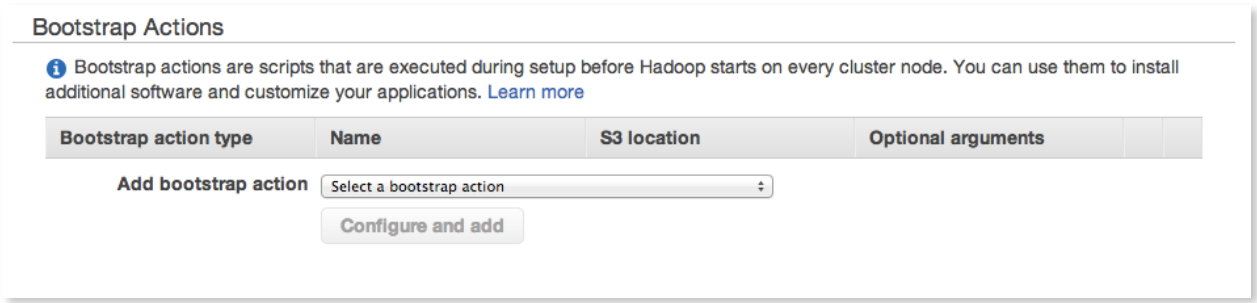
Next you can configure the type of cluster in this section, VPC or not, Availability Zones and which type of instances you want to use for your cluster. The defaults here are fine.



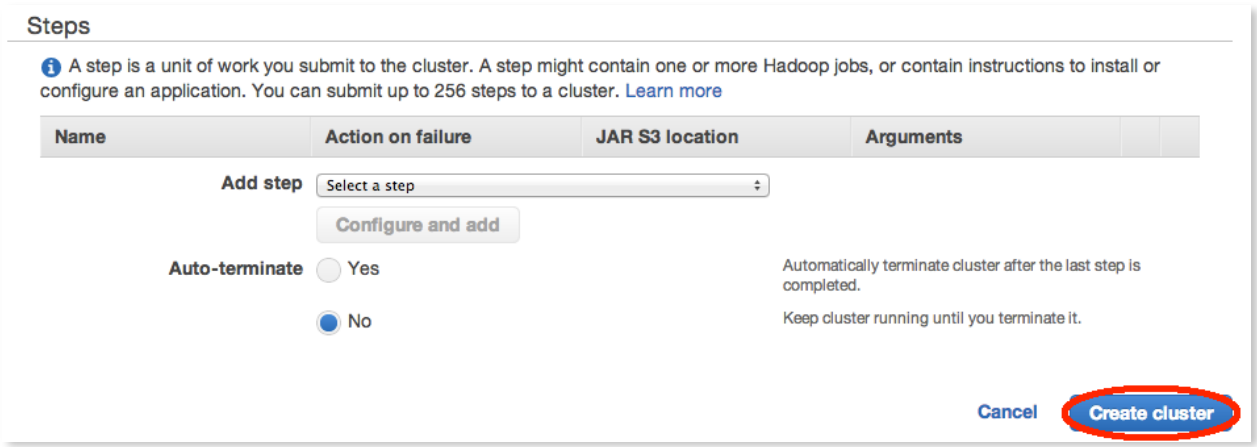
Here is where you could add an EC2 key pair if you wanted to log in to any of the instances in your cluster, or add access to IAM users, or add EC2 role to your instances. All the defaults here are fine.



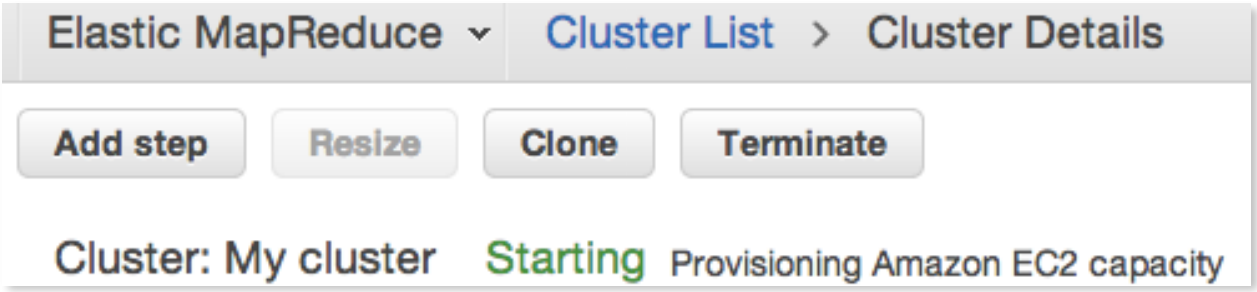
Bootstrap actions are scripts to install on your cluster. We will not need to add anything now.



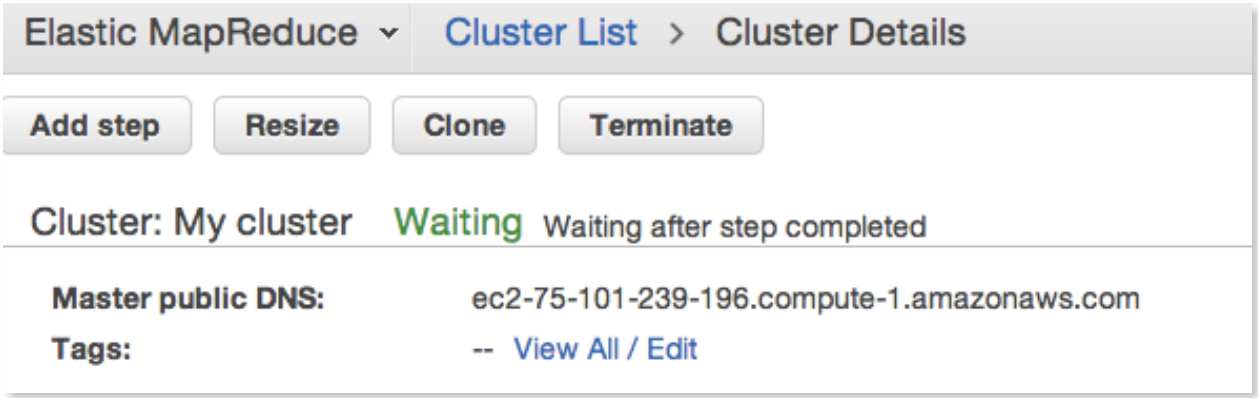
Finally, Steps are scripts you can have Hadoop run when it starts. You will not need any for this lab, so just click “Create cluster”.



Your cluster will be in the “Starting” status.Your cluster will then take a few minutes to start.



It will be ready to go when the cluster status is “Waiting”.



Step 3: Hive

In this step, you will run a Hive script to extract data from the users data file. First, copy the following in to a text editor:

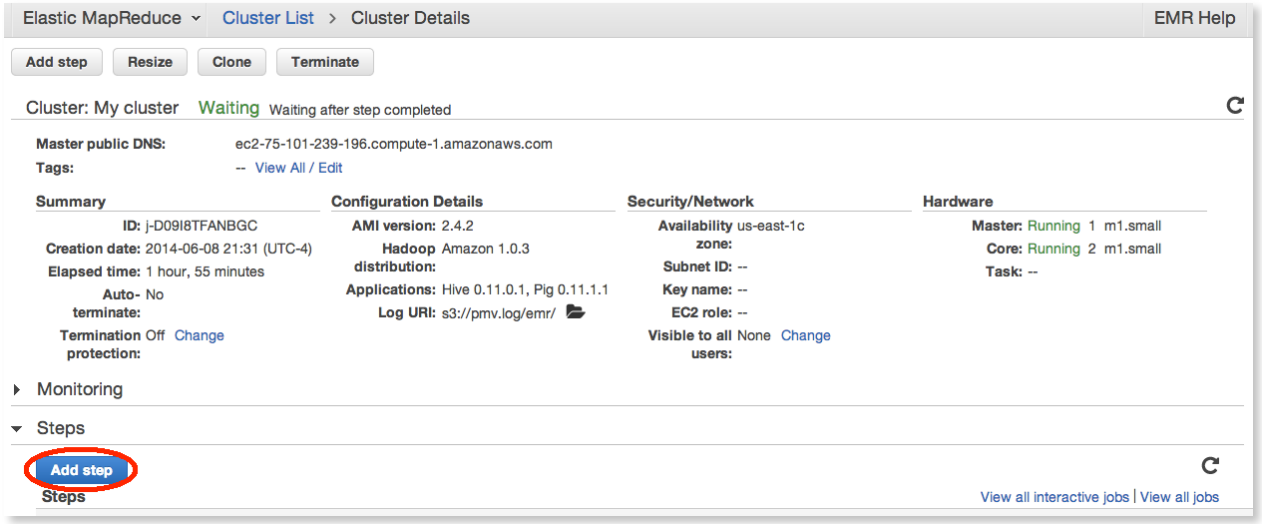
```
ADD JAR s3://pmv.public/emr/jsonserde.jar ;
CREATE EXTERNAL TABLE users (userid int, username string, firstname string,
lastname string, city string, state string,  email string, phone string,
likesports string, liketheatre string, likeconcerts string, likejazz string,
likeclassical string, likeopera string, likerock string, likevegas string,
likebroadway string, likemusicals string)
ROW FORMAT
    serde 'com.amazon.elasticmapreduce.JsonSerde'
    with serdeproperties ( 'paths'='userid, username, firstname, lastname,
city, state, email, phone, likesports, liketheatre, likeconcerts, likejazz,
likeclassical, likeopera, likerock, likevegas, likebroadway, likemusicals' )
LOCATION "${INPUT}" ;
INSERT OVERWRITE DIRECTORY "${OUTPUT}"
SELECT * FROM users WHERE likesports = "true";
```

This script first loads a JSON serializer/deserializer (or serde). Then it creates a table in a location referenced by an input, that will be added shortly. It then writes the output of a select to another referenced argument for the output.

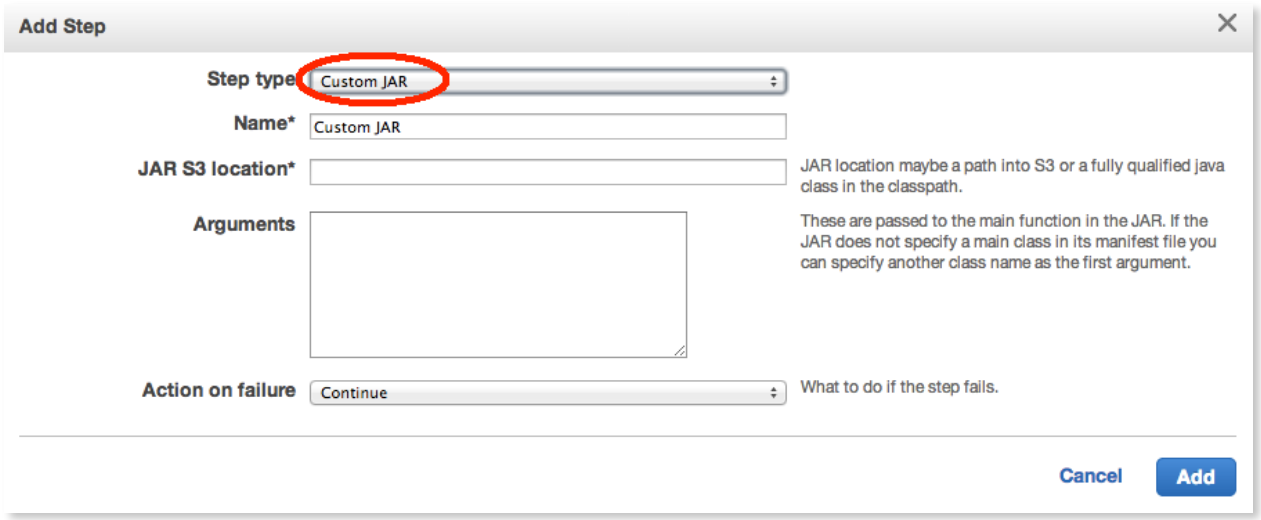
Save the file and give it a name along the lines of“users.sql”.

Upload the file to the “input” folder in the S3 bucket you created for the lab.

Then return to the AWS Console page for the EMR cluster you created. Select”Add step”.



Change the “Step type” to “Hive program”



You need to add 3 locations to this step. The first is the location of the script you just uploaded to S3. The format is:

s3://<YOUR-BUCKET>/input/users.sql

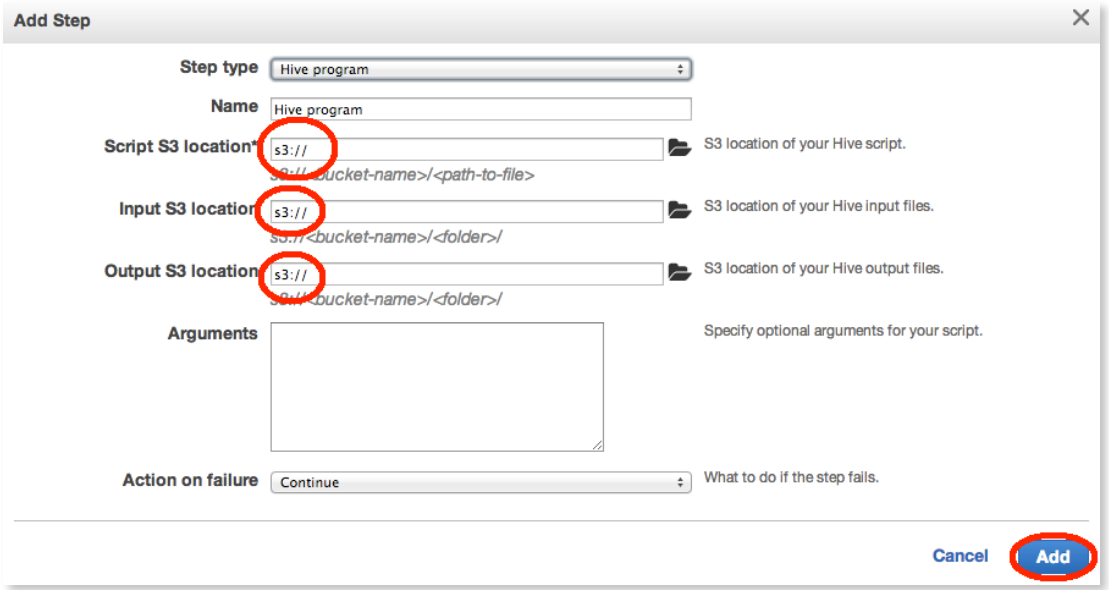
using your bucket name and the name you gave the script. The input location is:

s3://<YOUR-BUCKET>/users/

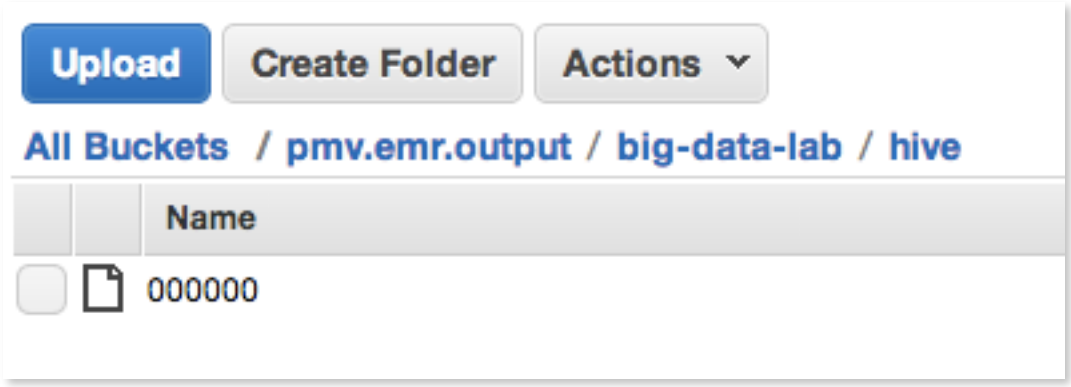
Note that you don’t want to specific the file. Hive reads in folders, not files. The output location is:

<s3://<YOUR-BUCKET>/output/hive/

After you’ve added the information necessary, click “Add”.



The EMR cluster will take a minute or so to run the script. Once the step has completed, you can check the location in S3.



Step 4: Pig

In this step, you will run a Pig script to extract data from the users data file. First, copy the following in to a text editor:

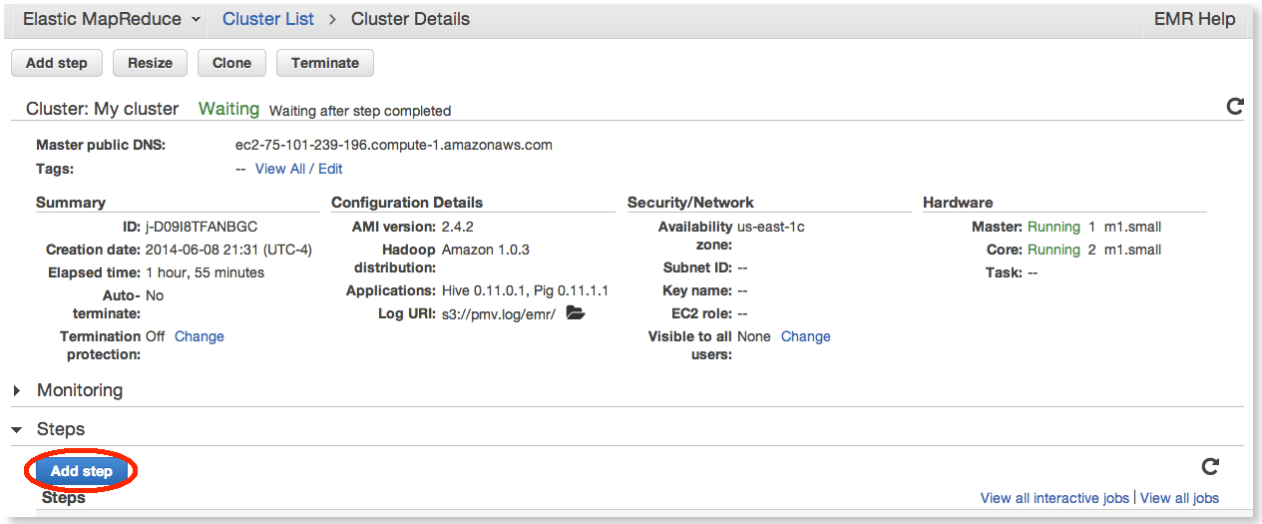
```
USERS = LOAD '$INPUT' USING
JsonLoader('userid:int,username:chararray,firstname:chararray,lastname:charar
ray,city:chararray,state:chararray,email:chararray,phone:chararray,likesports
:chararray,liketheatre:chararray,likeconcert:chararray,likejazz:chararray,lik
eclassical:chararray,likeopera:chararray,likerock:chararray,likevegas:chararr
ay,likebroadway:chararray,likemusicals:chararray');
STORE USERS into '$OUTPUT' USING PigStorage('|');
```

This script loads the data from S3 referenced in the input, mapping the fields in the JSON loader. It will then write to the output location, changing the format to pipe delimited.

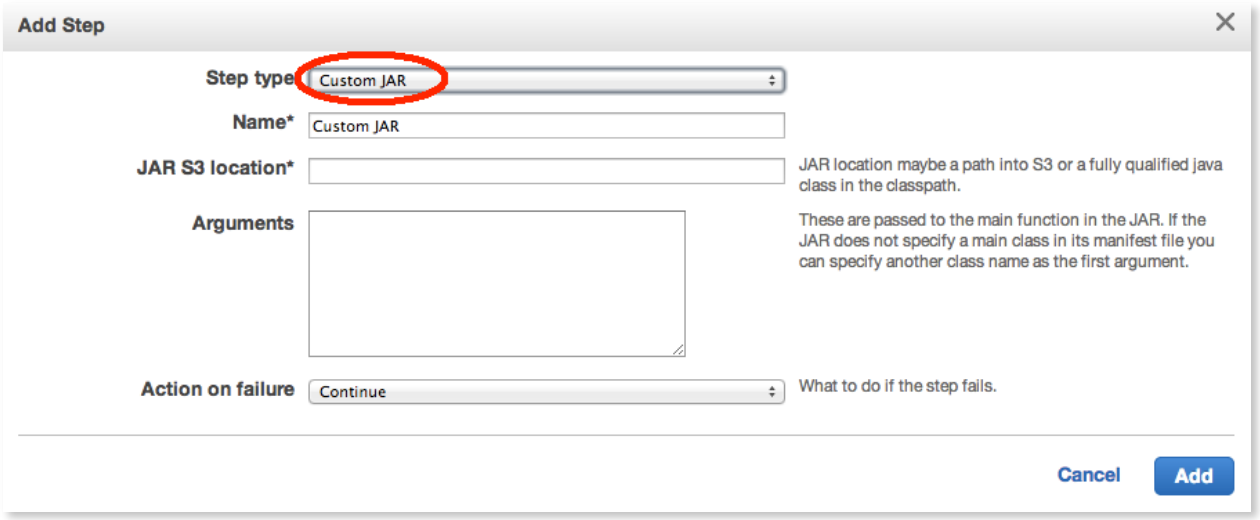
Save the file and give it a name along the lines of“users.pig”.

Upload the file to the “input” folder in the S3 bucket you created for the lab.

Then return to the AWS Console page for the EMR cluster you created. Select”Add step”.



Change the “Step type” to “Pig program”



You need to add 3 locations to this step. The first is the location of the script you just uploaded to S3. The format is:

s3://<YOUR-BUCKET>/input/users.pig

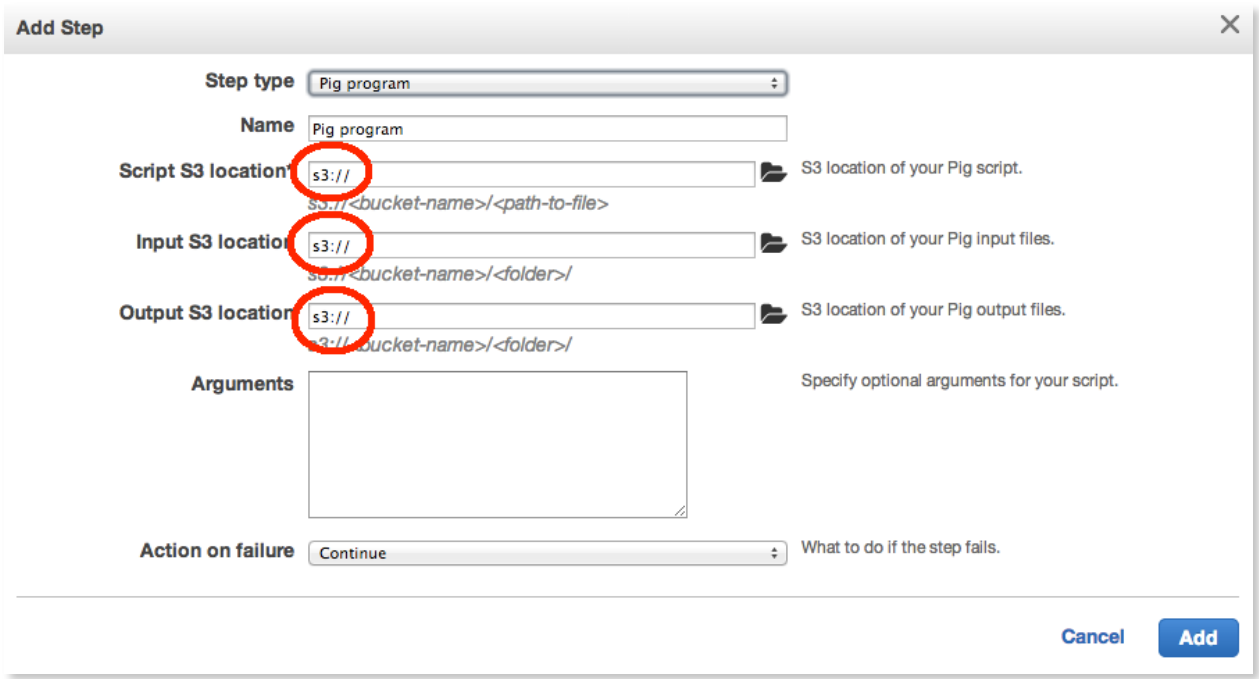
using your bucket name and the name you gave the script. The input location is:

s3://<YOUR-BUCKET>/users/users.txt

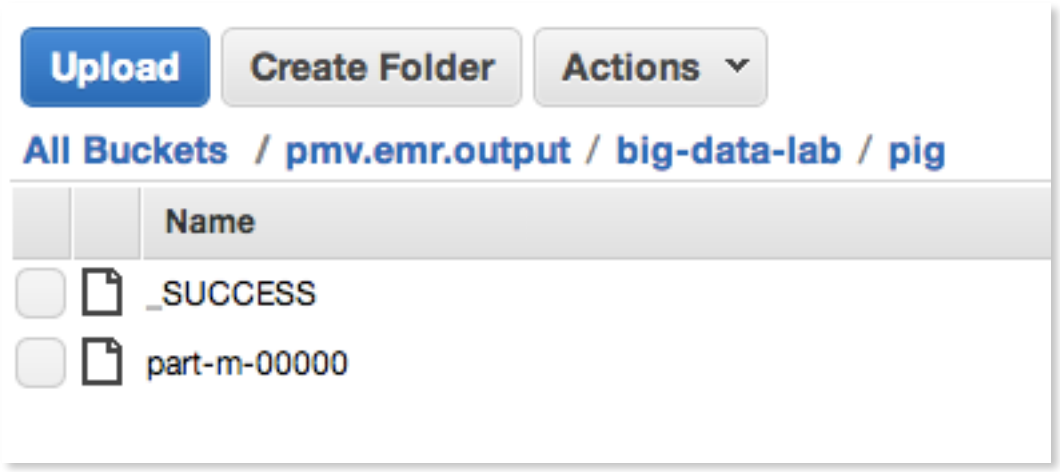
Note unlike Hive, you need to reference the specific file. The output location is:

<s3://<YOUR-BUCKET>/output/pig/

After you’ve added the information necessary, click “Add”.



The EMR cluster will take a minute or so to run the script. Once the step has completed, you can check the location in S3.



Step 5: COPY to Redshift

In this final step, you will copy the users data that was converted by Pig in to your existing Redshift cluster.

In the SQL client connected to the Redshift cluster, create the table to hold the user data:

```
CREATE TABLE users (
    userid int NOT NULL PRIMARY KEY,
    username varchar(100) NOT NULL DISTKEY,
    firstname varchar(100) NOT NULL SORTKEY,
    lastname varchar(100) NOT NULL,
    city varchar(100) NOT NULL,
    state varchar(100) NOT NULL,
    email varchar(100) NOT NULL,
    phone varchar(100) NOT NULL,
    likesports varchar(10) NOT NULL,
    liketheatre varchar(10) NOT NULL,
    likeconcert varchar(10) NOT NULL,
    likejazz varchar(10) NOT NULL,
    likeclassical varchar(10) NOT NULL,
    likeopera varchar(10) NOT NULL,
    likerock varchar(10) NOT NULL,
    likevegas varchar(10) NOT NULL,
    likebroadway varchar(10) NOT NULL,
    likemusicals varchar(10) NOT NULL
);
```

Once it's been created, run the COPY command in the SQL client:

```
COPY USERS FROM 's3://pmv.emr.output/big-data-lab/pig/out/part-m-00000'
CREDENTIALS 'aws_access_key_id=<YOUR-ACCESS-KEY>;aws_secret_access_key=<YOUR-
SECRET-ACCESS-KEY>' delimiter '|' COMPUPDATE ON;
```

Make sure you add your own AWS access key and secret key.
Using this command, Redshift will use its storage nodes to reach out to S3 in parallel and copy the data to the table. Using the "COMPUPDATE" option will make Redshift optimize the compression of the data on the table as it's being loaded.

The command should finish in a few minutes. Once it's done, you can try running some select statements. You could run the same command that was run in Hive:

```
select * from users where likesports = 'true';
```

This will return the same data set that Hive did. If you run the same statement again, you'll notice that it returns much more quickly. This is because Redshift compiles queries the first time they're run so that the each subsequent time the query is run, they are executed directly on the storage nodes.

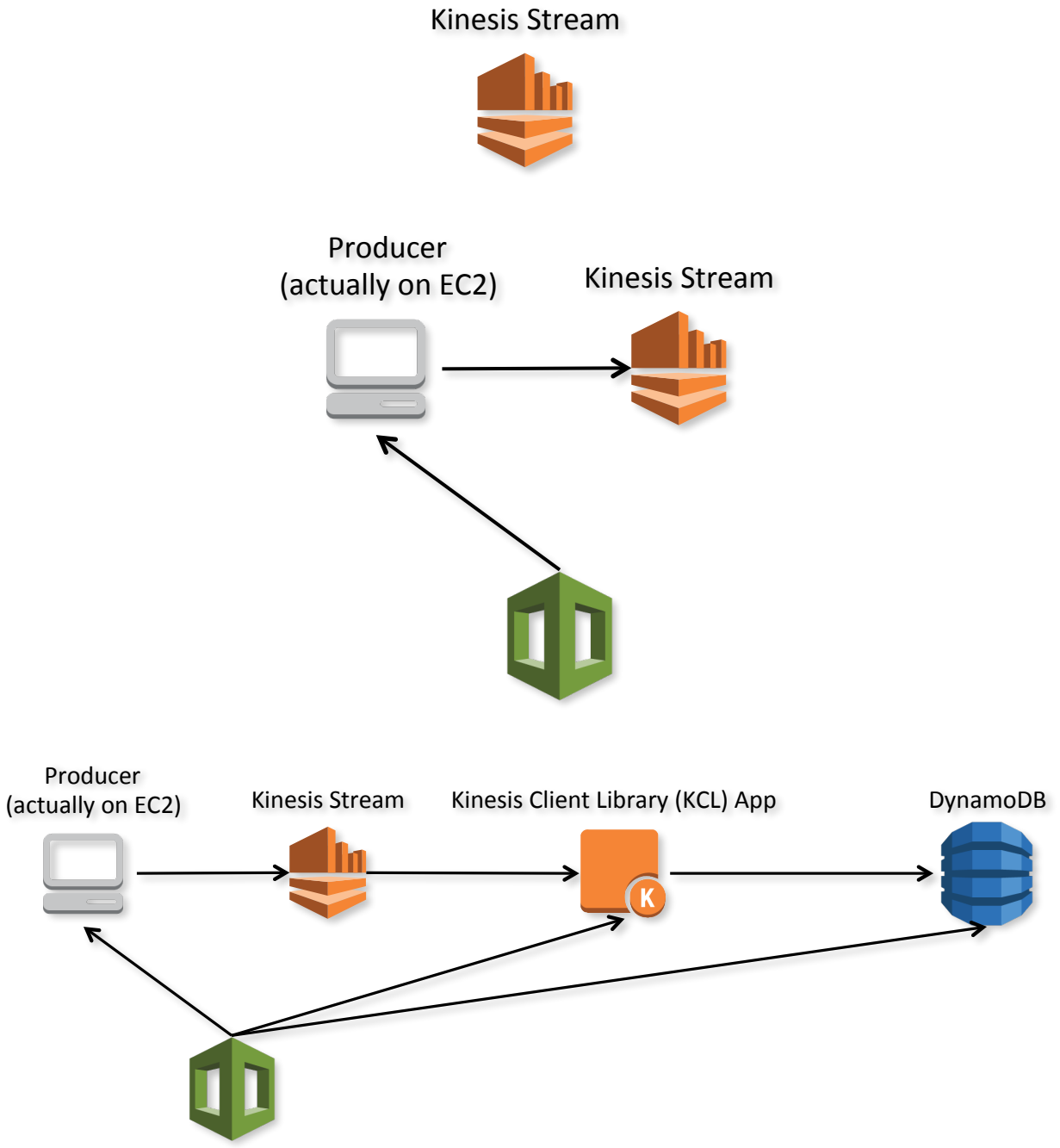
If your queries are returning successfully, congratulations! You've taken data in 1 format on S3, transformed it using EMR and loaded it in to your Redshift data warehouse.

Appendix A: Taking it further

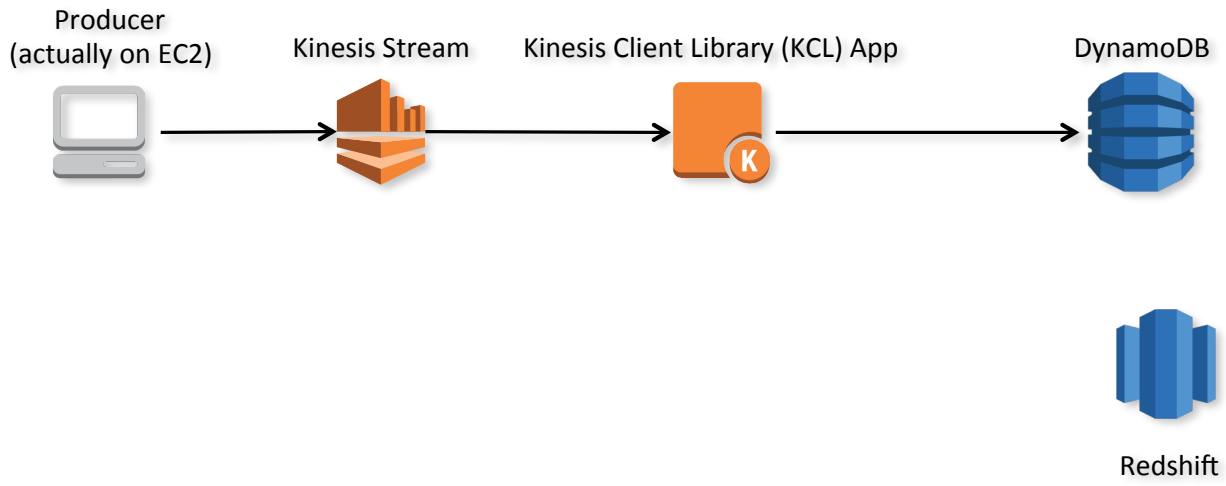
1. Re-write the consumer application from part 1 to write to S3 instead of Redshift directly.
2. Try to run a Business Intelligence (BI) tool on top of your Redshift cluster to visualize query results.
3. Connect to your EMR cluster via the command line and:
   A. run the Hive script in interactive mode and/or make other queries.
   B. run the Pig script in interactive mode and/or make other changes to the users.txt file.
4. Use Data Pipeline to create a pipeline that will launch an EMR cluster, run the Pig script and copy the data to the Redshift cluster.
5. Using DynamoDB
   C. Query the tables using the API
   D. Query the tables using Hive in EMR
   E. Export the tables to Redshift using the COPY command.

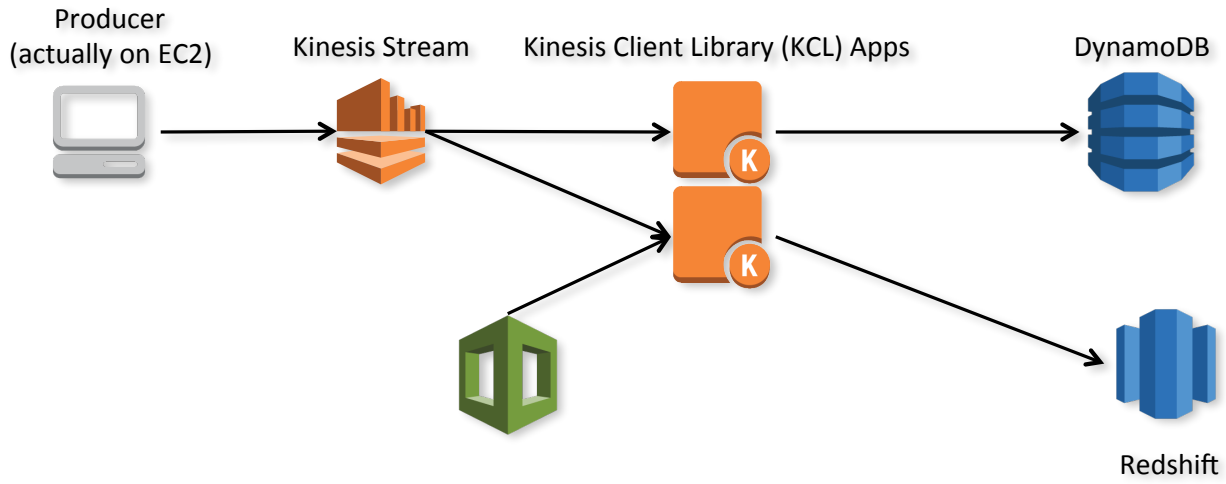
Appendix B: Architecture Diagrams

Amazon Kinesis



Amazon Redshift





Amazon Elastic Map Reduce