Getting Started with AWS

Analyzing Big Data

amazon
web services™

Getting Started with AWS: Analyzing Big Data

Table of Contents

Analyzing Big Data with Amazon Web Services

The following tutorials show you ways to use Amazon Web Services to process big data:

Key AWS Services for Big Data

With AWS, you pay only for the resources you use. Instead of maintaining a cluster of physical servers and storage devices that are standing by for possible use, you can create resources when you need them. AWS also supports popular tools like Hadoop, Hive, and Pig, and makes it easy to provision, configure, and monitor clusters for running those tools.

The following table shows how AWS can help address common big-data challenges.

Challenge	Solution
Data sets can be very large. Storage can become expensive, and data corruption and loss can have far-reaching implications.	Amazon S3 can store large amounts of data, and its capacity can grow to meet your needs. It is highly redundant and secure, protecting against data loss and unauthorized use. Amazon S3 also has an intentionally small feature set to keep its costs low.
Maintaining a cluster of physical servers to process data is expensive and time-consuming.	When you run an application on a virtual Amazon EC2 server, you pay for the server only while the application is running, and you can increase the number of servers — within minutes, not hours or days — to meet the processing needs of your application.

Challenge	Solution
Hadoop and other open-source big-data tools can be challenging to configure, monitor, and operate.	Amazon EMR handles cluster configuration, monitoring, and management. Amazon EMR also integrates open-source tools with other AWS services to simplify large-scale data processing, so you can focus on data analysis and extracting value.

For more information, see Big Data.

Setting Up

Before you use AWS for the first time, complete the following tasks. (These steps prepare you for both of the tutorials in this guide.)

Tasks
- Sign Up for AWS (p. 3)
- Create a Key Pair (p. 3)

Sign Up for AWS

When you sign up for Amazon Web Services (AWS), your AWS account is automatically signed up for all services in AWS and you can start using them immediately. You are charged only for the services that you use.

If you created your AWS account less than 12 months ago, you can get started with AWS for free. For more information, see AWS Free Tier.

If you have an AWS account already, skip to the next step. If you don't have an AWS account, use the following procedure to create one.

To create an AWS account

1. Open http://aws.amazon.com/, and then click **Sign Up**.
2. Follow the on-screen instructions.

   Part of the sign-up procedure involves receiving a phone call and entering a PIN using the phone keypad.

Create a Key Pair

AWS uses public-key cryptography to secure the login information for your instance. A Linux instance has no password; you use a key pair to log in to your instance securely. You specify the name of the key pair when you launch your instance, then provide the private key when you log in using SSH.

If you haven't created a key pair already, you can create one using the Amazon EC2 console.

To create a key pair

1. Open the Amazon EC2 console.
2. From the navigation bar, in the region selector, click **US West (Oregon)**.
3. In the navigation pane, click **Key Pairs**.
4. Click **Create Key Pair**.
5. Enter a name for the new key pair in the **Key pair name** field of the **Create Key Pair** dialog box, and then click **Create**. Choose a name that is easy for you to remember.
6. The private key file is automatically downloaded by your browser. The base file name is the name you specified as the name of your key pair, and the file name extension is `.pem`. Save the private key file in a safe place.

    Important
    This is the only chance for you to save the private key file. You'll need to provide the name of your key pair when you launch an instance and the corresponding private key each time you connect to the instance.

7. Prepare the private key file. This process depends on the operating system of the computer that you're using.

    - If your computer runs Mac OS X or Linux, use the following command to set the permissions of your private key file so that only you can read it.

    ```
    $ chmod 400 my-key-pair.pem
    ```

    - If your computer runs Windows, use the following steps to convert your .pem file to a .ppk file for use with PuTTY.

        a. Download and install PuTTY from http://www.chiark.greenend.org.uk/~sgtatham/putty/. Be sure to install the entire suite.
        b. Start PuTTYgen (for example, from the **Start** menu, click **All Programs > PuTTY > PuTTYgen**).
        c. Under **Type of key to generate**, select **SSH-2 RSA**.
        d. Click **Load**. By default, PuTTYgen displays only files with the extension `.ppk`. To locate your `.pem` file, select the option to display files of all types.
        e. Select your private key file and then click **Open**. Click **OK** to dismiss the confirmation dialog box.
        f. Click **Save private key**. PuTTYgen displays a warning about saving the key without a passphrase. Click **Yes**.
        g. Specify the same name that you used for the key pair (for example, my-key-pair) and then click **Save**. PuTTY automatically adds the `.ppk` file extension.

Tutorial: Sentiment Analysis

Sentiment analysis refers to various methods of examining and processing data in order to identify a subjective response, usually a general mood or a group's opinions about a specific topic. For example, sentiment analysis can be used to gauge the overall positivity toward a blog or a document, or to capture constituent attitudes toward a political candidate. Sentiment data is often derived from social media services and similar user-generated content, such as reviews, comments, and discussion groups. The data sets thus tend to grow large enough to be considered "big data."

Amazon EMR integrates open-source data processing frameworks with the full suite of service from AWS. The resulting architecture is scalable, efficient, and ideal for analyzing large-scale sentiment data, such as tweets over a given time period.

Suppose your company recently released a new product and you want to assess its reception among consumers who use Twitter. In this tutorial, you'll launch an AWS CloudFormation stack that provides a script for collecting tweets. You'll store the tweets in Amazon S3 and customize a mapper file for use with Amazon EMR. Then you'll create an Amazon EMR cluster that uses a Python natural language toolkit, implemented with a Hadoop streaming job, to classify the data. Finally, you'll examine the output files and evaluate the aggregate sentiment of the tweets.

This tutorial typically takes less than an hour to complete. You pay only for the resources you use. The tutorial includes a cleanup step to help ensure that you don't incur additional costs.

Prerequisites

Before you begin, make sure that you've completed the steps in Setting Up (p. 3).

Tasks

Step 1: Create a Twitter Developer Account

In order to collect tweets for analysis, you'll need to create an account on the Twitter developer site and generate credentials for use with the Twitter API.

To create a Twitter developer account

1. Go to https://dev.twitter.com/user/login and log in with your Twitter user name and password. If you do not yet have a Twitter account, click the **Sign up** link that appears under the **Username** field.
2. If you have not yet used the Twitter developer site, you'll be prompted to authorize the site to use your account. Click **Authorize app** to continue.
3. Go to the Twitter applications page at https://dev.twitter.com/apps and click **Create a new application**.
4. Follow the on-screen instructions. For the application **Name**, **Description**, and **Website**, you can specify any text — you're simply generating credentials to use with this tutorial, rather than creating a real application.
5. Twitter displays the details page for your new application. Click the <guilabel>Key and Access Tokens</guilabel> tab collect your Twitter developer credentials. You'll see a **Consumer key** and **Consumer secret**. Make a note of these values; you'll need them later in this tutorial. You may want to store your credentials in a text file.
6. At the bottom of the page click **Create my access token**. Make a note of the **Access token** and **Access token secret** values that appear, or add them to the text file you created in the preceding step.

If you need to retrieve your Twitter developer credentials at any point, go to https://dev.twitter.com/apps and select the application that you created for the purposes of this tutorial.

Step 2: Create an Amazon S3 Bucket

Amazon EMR jobs typically use Amazon S3 buckets for input and output data files, as well as for any mapper and reducer files that aren't provided by open-source tools. For the purposes of this tutorial, you'll create an Amazon S3 bucket in which you'll store the data files and a custom mapper.

To create an Amazon S3 bucket using the console

1. Open the Amazon S3 console.
2. Click **Create Bucket**.
3. In the **Create Bucket** dialog box, do the following:

    a. Specify a name for your bucket, such as `mysentimentjob`. To meet Hadoop requirements, your bucket name is restricted to lowercase letters, numbers, periods (.), and hyphens (-).
    b. For the region, select **US Standard**.
    c. Click **Create**.

4. Select your new bucket from the **All Buckets** list and click **Create Folder**. In the text box, specify `input` as the folder name, and then press Enter or click the check mark.
5. Repeat the previous step to create a folder named `mapper` at the same level as the `input` folder.
6. For the purposes of this tutorial (to ensure that all services can use the folders), make the folders public as follows:

    a. Select the `input` and `mapper` folders.

b.   Click **Actions** and then click **Make Public**.

c.   When prompted for confirmation, click **OK**.

Step 3: Collect and Store the Sentiment Data

In this step, you'll use an AWS CloudFormation template to launch an instance, and then use a command-line tool on the instance to collect the Twitter data. You'll also use a command-line tool on the instance to store the data in the Amazon S3 bucket that you created.

Tasks

Launch an Instance Using AWS CloudFormation

We have provided a simple template that launches a single instance using an Amazon Linux AMI, the key pair that you created, and a security group that grants everyone (`0.0.0.0/0`) access using SSH.

To launch the AWS CloudFormation stack

1.   Open the AWS CloudFormation console.

2.   Make sure that **US East (N. Virginia)** is selected in the region selector of the navigation bar.

3.   Click **Create Stack**.

4.   On the **Select Template** page, do the following:

a.   Under **Stack**, in the **Name** box, specify a name that is easy for you to remember. For example, `my-sentiment-stack`.

b.   Under **Template**, select **Specify an Amazon S3 template URL**, and specify `https://s3.amazonaws.com/awsdocs/gettingstarted/latest/sentiment/sentimentGSG.template` in the text box.

c.   Click **Next**.

5. On the **Specify Parameters** page, do the following:

a. In the **KeyPair** box, specify the name of the key pair that you created in Create a Key Pair (p. 3). Note that this key pair must be in the US East (N. Virginia) region.

b. In the **TwitterConsumerKey**, **TwitterConsumerSecret**, **TwitterToken**, and **TwitterTokenSecret** boxes, specify your Twitter credentials. For best results, copy and paste the Twitter credentials from the Twitter developer site or the text file you saved them in.

> **Note**
> The order of the Twitter credential boxes on the **Specify Parameters** page may not match the display order on the Twitter developer site. Verify that you're pasting the correct value in each box.

c. Click **Next**.

6. On the **Options** page, click **Next**.

7. On the **Review** page, select the **I acknowledge that this template might cause AWS CloudFormation to create IAM resources** check box, and then click **Create** to launch the stack.

8. Your new AWS CloudFormation stack appears in the list with its status set to CREATE_IN_PROGRESS.

> **Note**
> Stacks take several minutes to launch. To see whether this process is complete, click **Refresh**. When your stack status is CREATE_COMPLETE, it's ready to use.

Collect Tweets Using the Instance

The instance has been preconfigured with Tweepy, an open-source package for use with the Twitter API. Python scripts for running Tweepy appear in the sentiment directory.

> **Note**
> You may encounter an issue installing Tweepy. If you do encounter this issue, edit setup.py as follows.
> Remove the following:

```
install_reqs = parse_requirements('requirements.txt', ses
sion=uuid.uuid1())
reqs = [str(req.req) for req in install_reqs]
```

Add the following:

```
import os
from setuptools import setup
with open('requirements.txt') as f: reqs = f.read().splitlines()
```

To collect tweets using your AWS CloudFormation stack

1. Select the **Outputs** tab. Copy the DNS name of the Amazon EC2 instance that AWS CloudFormation created from the EC2DNS key.

Overview	Outputs	Resources	Events	Template	Parameters	Tags	Stack Policy

Key	Value	Description
EC2DNS	ec2-54-161-236-88.compute-1.amazonaws.com	The public DNS for the new EC2 instance

2. Connect to the instance using SSH. Specify the name of your key pair and the user name `ec2-user`. For more information, see Connect to Your Linux Instance.

3. In the terminal window, run the following command:

   ```
   $ cd sentiment
   ```

4. To collect tweets, run the following command, where *term1* is your search term. Note that the collector script is not case sensitive. To use a multi-word term, enclose it in quotation marks.

   ```
   $ python collector.py term1
   ```

   Examples:

   ```
   $ python collector.py kindle
   $ python collector.py "kindle fire"
   ```

5. Press **Enter** to run the collector script. Your terminal window displays the following message:

   ```
   Collecting tweets. Please wait.
   ```

   > **Note**
   > If your SSH connection is interrupted while the script is still running, reconnect to the instance and run the script with nohup (for example, `nohup python collector.py > /dev/null &`).

The script collects 500 tweets, which could take several minutes. If you're searching for a subject that is not currently popular on Twitter (or if you edited the script to collect more than 500 tweets), the script will take longer to run. You can interrupt it at any time by pressing Ctrl+C.

When the script has finished running, your terminal window displays the following message:

```
Finished collecting tweets.
```

Store the Tweets in Amazon S3

Your sentiment analysis stack has been preconfigured with s3cmd, a command-line tool for Amazon S3. You'll use s3cmd to store your tweets in the bucket that you created in Step 2: Create an Amazon S3 Bucket (p. 6).

To store the collected tweets in your bucket

1. In your SSH window, run the following command. (The current directory should still be `sentiment`. If it's not, use `cd` to navigate to the `sentiment` directory.)

    ```
    $ ls
    ```

    You should see a file named `tweets.`*`date-time`*`.txt`, where *`date`* and *`time`* reflect when the script was run. This file contains the ID numbers and full text of the tweets that matched your search terms.

2. To copy the Twitter data to Amazon S3, run the following command, where *`tweet-file`* is the file you identified in the previous step and *`your-bucket`* is the name of your bucket.

    > **Important**
    > Be sure to include the trailing slash, to indicate that `input` is a folder. Otherwise, Amazon S3 will create an object called `input` in your base S3 bucket.

    ```
    $ s3cmd put tweet-file s3://your-bucket/input/
    ```

    For example:

    ```
    $ s3cmd put tweets.Nov12-1227.txt s3://mysentimentjob/input/
    ```

3. To verify that the file was uploaded to Amazon S3, run the following command:

    ```
    $ s3cmd ls s3://your-bucket/input/
    ```

    You can also use the Amazon S3 console to view the contents of your bucket and folders.

Step 4: Customize the Mapper

When you create your own Hadoop streaming programs, you'll need to write mapper and reducer executables as described in Process Data with a Streaming Cluster in the *Amazon Elastic MapReduce Developer Guide*. For this tutorial, we've provided a mapper script that you can customize for use with your Twitter search term.

To customize the mapper

1. On your computer, open a text editor. Copy and paste the following script into a new file, and save the file as `sentiment.py`.

    ```
    #!/usr/bin/python

    import cPickle as pickle
    import nltk.classify.util
    ```

```
from nltk.classify import NaiveBayesClassifier
from nltk.tokenize import word_tokenize
import sys

sys.stderr.write("Started mapper.\n");


def word_feats(words):
    return dict([(word, True) for word in words])


def subj(subjLine):
    subjgen = subjLine.lower()
    # Replace term1 with your subject term
    subj1 = "term1"
    if subjgen.find(subj1) != -1:
        subject = subj1
        return subject
    else:
        subject = "No match"
        return subject


def main(argv):
    classifier = pickle.load(open("classifier.p", "rb"))
    for line in sys.stdin:
        tolk_posset = word_tokenize(line.rstrip())
        d = word_feats(tolk_posset)
        subjectFull = subj(line)
        if subjectFull == "No match":
            print "LongValueSum:" + " " + subjectFull + ": " + "\t" + "1"
        else:
            print "LongValueSum:" + " " + subjectFull + ": " + classifi
er.classify(d) + "\t" + "1"


if __name__ == "__main__":
    main(sys.argv)
```

2. Replace *term1* with the search term that you used in and save the file.

   Important
   Do not change the spacing in the file. Incorrect indentation causes the Hadoop streaming program to fail.

3. Open the Amazon S3 console and locate the `mapper` folder that you created in .

4. Click **Upload** and follow the on-screen instructions to upload your customized `sentiment.py` file.

5. Select the file, click **Actions**, and then click **Make Public**.

Step 5: Create an Amazon EMR Cluster

You can use Amazon EMR to configure a cluster with software, bootstrap actions, and work steps. For this tutorial, you'll run a Hadoop streaming program. When you configure a cluster with a Hadoop streaming
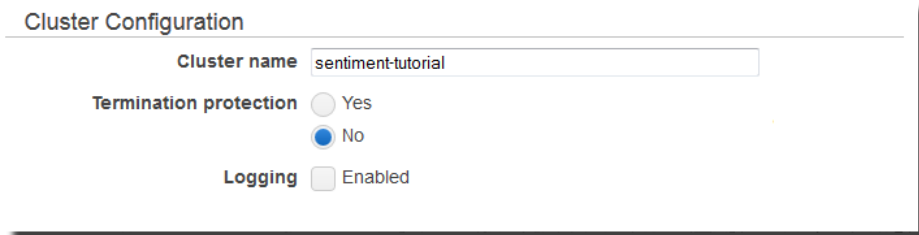
program in Amazon EMR, you specify a mapper and a reducer, as well as any supporting files. The following list provides a summary of the files you'll use for this tutorial.

- For the mapper, you'll use the file you customized in the preceding step.
- For the reducer method, you'll use the predefined Hadoop package `aggregate`. For more information about the aggregate package, see the Hadoop documentation.
- Sentiment analysis usually involves some form of natural language processing. For this tutorial, you'll use the Natural Language Toolkit (NLTK), a popular Python platform. You'll use an Amazon EMR bootstrap action to install the NLTK Python module. Bootstrap actions load custom software onto the instances that Amazon EMR provisions and configures. For more information, see Create Bootstrap Actions in the *Amazon Elastic MapReduce Developer Guide*.
- Along with the NLTK module, you'll use a natural language classifier file that we've provided in an Amazon S3 bucket.
- For the job's input data and output files, you'll use the Amazon S3 bucket you created (which now contains the tweets you collected).

Note that the files used in this tutorial are for illustration purposes only. When you perform your own sentiment analysis, you'll need to write your own mapper and build a sentiment model that meets your needs. For more information about building a sentiment model, see Learning to Classify Text in *Natural Language Processing with Python*, which is provided for free on the NLTK site.

To create a cluster

1. Open the Amazon EMR console.
2. Click **Create cluster**.
3. Under **Cluster Configuration**, specify a name for the cluster, or leave the default value of **My cluster**. Set **Termination protection** to **No** and clear the **Logging Enabled** check box.

> **Note**
> In a production environment, logging and debugging can be useful tools for analyzing errors or inefficiencies in Amazon EMR steps or programs. For more information, see Troubleshooting in the *Amazon Elastic MapReduce Developer Guide*.

4. Under **Software Configuration**, leave the default **Hadoop distribution** setting: **Amazon** and select **AMI version** 3.4.0. Under **Applications to be installed**, click each X to remove Hive, Pig, and Hue from the list.
5. Under **File System Configuration** and **Hardware Configuration**, leave the default settings.
6. Under **Security and Access**, select your key pair from **EC2 key pair**. Leave the default IAM user access. If this is your first time using Amazon EMR, click **Create Default Role** to create the default EMR role and then click **Create Default Role** to create the default EC2 instance profile.

7. Under **Bootstrap Actions**, do the following:

    a. In the **Add bootstrap action** list, select **Custom action**. You'll add a custom action that installs and configures the Natural Language Toolkit on the cluster.

    b. Click **Configure and add**.

    c. In the **Add Bootstrap Action** dialog box, in the **Amazon S3 location** box, copy and paste the following path:

    ```
    s3://awsdocs/gettingstarted/latest/sentiment/config-nltk.sh
    ```

    Alternatively, copy and paste the following script into a new file, upload that file to an Amazon S3 bucket, and use that location instead:

    ```
    #!/bin/bash
    sudo yum -y install git gcc python-dev python-devel
    sudo pip install -U numpy
    sudo pip install pyyaml nltk
    sudo pip install -e git://github.com/mdp-toolkit/mdp-toolkit#egg=MDP
    sudo python -m nltk.downloader -d /usr/share/nltk_data all
    ```

    d. Click **Add**. The final result is displayed as follows.

8. Under **Steps**, do the following:

    a. Set **Auto-terminate** to **Yes**.

    b. In the **Add step** list, select **Streaming program**, and then click **Configure and add**.

c. In the **Add Step** dialog box, configure the job as follows, replacing *your-bucket* with the name of the Amazon S3 bucket you created earlier, and then click **Add**:

- Set **Name** to `Sentiment analysis`
- Set **Mapper** to s3://*your-bucket*/mapper/sentiment.py
- Set **Reducer** to `aggregate`
- Set **Input S3 location** to s3://*your-bucket*/input/
- Set **Output S3 location** to s3://*your-bucket*/output/ (note that this folder must not exist)
- Set **Arguments** to -cacheFile s3://awsdocs/gettingstarted/latest/sentiment/classifier.p#classifier.p
- Set **Action on failure** to `Continue`

9. At the bottom of the page, click **Create cluster**.

A summary of your new cluster appears, with the status `Starting`. It takes a few minutes for Amazon EMR to provision the EC2 instances for your cluster.

Step 6: Examine the Sentiment Analysis Output

When your cluster's status in the Amazon EMR console is **Waiting: Waiting after step completed**, you can examine the results.

To examine the streaming program results

1. Open the Amazon S3 console and locate the bucket you created in Step 2: Create an Amazon S3 Bucket (p. 6). You should see a new `output` folder in your bucket. You might need to click the refresh arrow in the top right corner to see the new bucket.
2. The job output is split into several files: an empty status file named `_SUCCESS` and several `part-xxxxx` files. The `part-xxxxx` files contain sentiment measurements generated by the Hadoop streaming program.
3. Select an output file, click **Actions**, and then click **Download**. Right-click the link in the pop-up window and click **Save Link As** to download the file.

Repeat this step for each output file.

4. Open the files in a text editor. You'll see the total number of positive and negative tweets for your search term, as well as the total number of tweets that did not match any of the positive or negative terms in the classifier (usually because the subject term was in a different field, rather than in the actual text of the tweet).

   For example:

   ```
   kindle: negative       13
   kindle: positive      479
   No match:               8
   ```

   In this example, the sentiment is overwhelmingly positive. In most cases, the positive and negative totals are closer together. For your own sentiment analysis work, you'll want to collect and compare data over several time periods, possibly using several different search terms, to get as accurate a measurement as possible.

Step 7: Clean Up

To prevent your account from accruing additional charges, you should clean up the AWS resources that you created for this tutorial.

To delete the AWS CloudFormation stack

1. Open the AWS CloudFormation console.
2. Select your sentiment stack and click **Delete Stack**.
3. When prompted for confirmation, click **Yes, Delete**.

Because you ran a Hadoop streaming program and set it to auto-terminate after running the steps in the program, the cluster should have been automatically terminated when processing was complete.

To verify that the cluster was terminated

1. Open the Amazon EMR console.
2. Click **Cluster List**
3. In the cluster list, the **Status** of your cluster should be **Terminated**. Otherwise, select the cluster and click **Terminate**.

Tutorial: Web Server Log Analysis

Suppose you host a popular e-commerce website and you want to analyze your Apache web logs to see how people find your site. You want to determine which of your online ad campaigns drive the most traffic to your online store.

The web server logs, however, are too large to import into a MySQL database, and they are not in a relational format. You need another way to analyze them.

Amazon EMR integrates open-source applications such as Hadoop and Hive with Amazon Web Services to provide a scalable and efficient architecture for analyzing large-scale data, such as Apache web logs.

In this tutorial, we'll import data from Amazon S3 and create an Amazon EMR cluster. Then we'll connect to the master node of the cluster, where we'll run Hive to query the Apache logs using a simplified SQL syntax. To learn more about Hive, see http://hive.apache.org/.

This tutorial typically takes less than an hour to complete. You pay only for the resources you use, and the tutorial includes a clean-up step to help ensure that you don't incur unnecessary costs.

Prerequisites

Before you begin, make sure you've completed the steps in Setting Up (p. 3).

Tasks

Step 1: Create an Amazon EMR Cluster

You can use Amazon EMR to configure a cluster with software, bootstrap actions, and work steps. For this tutorial, you'll run a Hadoop streaming program.

To create a cluster

1. Open the Amazon EMR console.

2. Click **Create cluster**.

3. Under **Cluster Configuration**, specify a **Cluster name** or leave the default value of `My cluster`. Set **Termination protection** to **No** and clear the **Logging Enabled** check box.

> **Note**
> In a production environment, logging and debugging can be useful tools for analyzing errors or inefficiencies in Amazon EMR steps or programs. For more information, see Troubleshooting in the *Amazon Elastic MapReduce Developer Guide*.

4. Under **Software Configuration**, leave the default **Hadoop distribution** setting, **Amazon**, and the latest **AMI version**. Under **Applications to be installed**, leave the default **Hive** settings. Click the X to remove Pig from the list.

5. Under **Hardware Configuration**, leave the default settings.

> **Note**
> When you analyze data in a real application, you might want to increase the size or number of these nodes to improve processing power and reduce computational time. You may also

want to use spot instances to further reduce your costs. For more information, see Lowering Costs with Spot Instances in the *Amazon Elastic MapReduce Developer Guide*.

6.   Under **Security and Access**, select your key pair from **EC2 key pair**. Leave the default IAM user access. If this is your first time using Amazon EMR, click **Create Default Role** to create the default EMR role and then click **Create Default Role** to create the default EC2 instance profile.

7.   Leave the default **Bootstrap Actions** and **Steps** settings. Bootstrap actions and steps allow you to customize and configure your application. For this tutorial, we are using Hive, which is already included in our configuration.

8.   At the bottom of the page, click **Create cluster**.

When the summary of your new cluster appears, the status is `Starting`. It takes a few minutes for Amazon EMR to provision the Amazon EC2 instances for your cluster and change the status to `Waiting`.

Step 2: Connect to the Master Node

When the status of the cluster is `Waiting`, you can connect the master node. The way that you connect depends on the operating system of the computer that you're using. For step-by-step directions by operating system, click **SSH**.

When you've successfully connected to the master node, you'll see a welcome message and prompt similar to the following:

```
--------------------------------------------------------------------------

Welcome to Amazon Elastic MapReduce running Hadoop and Amazon Linux.

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:
```

```
  ResourceManager      lynx http://ip-172-16-43-158:9026/
  NameNode             lynx http://ip-172-16-43-158:9101/


--------------------------------------------------------------------------------
[hadoop@ip-172-16-43-158 ~]$
```

Step 3: Start and Configure Hive

Apache Hive is a data warehouse application you can use to query Amazon EMR cluster data with a SQL-like language. Because we select Hive when we configured the cluster, it's ready to use on the master node.

To use Hive interactively to query the web server log data, you'll need to load some additional libraries. The additional libraries are contained in a Java archive file named `hive_contrib.jar` on the master node. When you load these libraries, Hive bundles them with the map-reduce job that it launches to process your queries.

To start and configure Hive on the master node

1. In the terminal window for the master node, at the `hadoop` prompt, run the following command.

    ```
    [hadoop@ip-172-16-43-158 ~]$ hive
    ```

    Tip
    If the hive command is not found, be sure that you specified **hadoop** as the user name when connecting to the master node, not **ec2-user**. Otherwise, close this connection and connect to the master node again.

2. At the `hive>` prompt, run the following command.

    ```
    hive> add jar /home/hadoop/hive/lib/hive_contrib.jar;
    ```

    Tip
    If `/home/hadoop/hive/hive_contrib.jar` is not found, it's possible that there's a problem with the AMI that you selected. Follow the directions in Clean Up (p. 21) and then start this tutorial again, using a different AMI version.

When the command completes, you'll see a confirmation message similar to the following:

```
Added /home/hadoop/hive/lib/hive_contrib.jar to class path
Added resource: /home/hadoop/hive/lib/hive_contrib.jar
```

Step 4: Create the Hive Table and Load Data into HDFS

In order for Hive to interact with data, it must translate the data from its current format (in the case of Apache web logs, a text file) into a format that can be represented as a database table. Hive does this

translation using a serializer/deserializer (SerDe). SerDes exist for a variety of data formats. For information about how to write a custom SerDe, see the Apache Hive Developer Guide.

The SerDe we'll use in this example uses regular expressions to parse the log file data. It comes from the Hive open-source community. Using this SerDe, we can define the log files as a table, which we'll query using SQL-like statements later in this tutorial. After Hive has loaded the data, the data persists in HDFS storage as long as the Amazon EMR cluster is running, even if you shut down your Hive session and close the SSH connection.

To translate the Apache log file data into a Hive table

1. Copy the following multi-line command.

```
CREATE TABLE serde_regex(
  host STRING,
  identity STRING,
  user STRING,
  time STRING,
  request STRING,
  status STRING,
  size STRING,
  referer STRING,
  agent STRING)
ROW FORMAT SERDE 'org.apache.hadoop.hive.contrib.serde2.RegexSerDe'
WITH SERDEPROPERTIES (
  "input.regex" = "([^ ]*) ([^ ]*) ([^ ]*) (-|\\[[^\\]]*\\]) ([^
\"]*|\"[^\"]*\") (-|[0-9]*) (-|[0-9]*)(?: ([^ \"]*|\"[^\"]*\") ([^
\"]*|\"[^\"]*\"))?",
  "output.format.string" = "%1$s %2$s %3$s %4$s %5$s %6$s %7$s %8$s %9$s"
)
LOCATION 's3://elasticmapreduce/samples/pig-apache/input/';
```

2. At the `hive` command prompt, paste the command (use Ctrl+Shift+V in a terminal window or right-click in a PuTTY window), and then press Enter.

When the command finishes, you'll see a message like this one:

```
OK
Time taken: 12.56 seconds
```

Note that the `LOCATION` parameter specifies the location of a set of sample Apache log files in Amazon S3. To analyze your own Apache web server log files, replace the URL in the example command with the location of your own log files in Amazon S3.

Step 5: Query Hive

You can use Hive to query the Apache log file data. Hive translates your query into a Hadoop MapReduce job and runs it on the Amazon EMR cluster. Status messages appear as the Hadoop job runs.

Hive SQL is a subset of SQL; if you know SQL, you'll be able to easily create Hive queries. For more information about the query syntax, see the Hive Language Manual.

The following are some example queries.

Example 1: Count the number of rows in the Apache webserver log files

```
select count(1) from serde_regex;
```

After the query finishes, you'll see output similar to the following:

```
Total MapReduce CPU Time Spent: 13 seconds 860 msec
OK
239344
Time taken: 86.92 seconds, Fetched: 1 row(s)
```

Example 2: Return all fields from one row of log file data

```
select * from serde_regex limit 1;
```

After the query finishes, you'll see output similar to the following:

```
OK
66.249.67.3    -    -    20/Jul/2009:20:12:22 -0700]    "GET /gal
lery/main.php?g2_controller=exif.SwitchDetailMode
&g2_mode=detailed&g2_return=%2Fgallery%2Fmain.php%3Fg2_itemId%3D15741&g2_return
Name=photo HTTP/1.1"    302    5
"-"    "Mozilla/5.0 (compatible; Googlebot/2.1; +http://www.google.com/bot.html)"
Time taken: 4.444 seconds, Fetched: 1 row(s)
```

Example 3: Count the number of requests from the host with an IP address of `192.168.1.198`

```
select count(1) from serde_regex where host="192.168.1.198";
```

After the query finishes, you'll see output similar to the following:

```
Total MapReduce CPU Time Spent: 13 seconds 870 msec
OK
46
Time taken: 73.077 seconds, Fetched: 1 row(s)
```

Step 6: Clean Up

To prevent your account from accruing additional charges, clean up the following AWS resources that you created for this tutorial.

To disconnect from the master node

1. In your terminal or PuTTY window, press CTRL+C to exit Hive.
2. At the SSH command prompt, run `exit`.
3. Close the terminal or PuTTY window.

To terminate the cluster

1. Open the Amazon EMR console.
2. Click **Cluster List**.
3. Select the cluster name, and then click **Terminate**. When prompted for confirmation, click **Terminate**.

More Big Data Options

The tutorials in this guide are just two examples of how you can use Amazon EMR to work with big data. This page provides other options that you might want to explore.

- **Analyze Sentiment by Location in Twitter**

  We analyzed only the content of the tweets. You might want to collect geographic data, creating data sets for specific regions or ZIP codes in order to analyze sentiment by location.

  For more information about the geographic aspects of Twitter data, see the Twitter API resource documentation, such as the description of the reverse_geocode resource.

- **Explore Corpora and Data**

  The Natural Language Toolkit includes several corpora, ranging from Project Gutenberg selections to patient information leaflets. The Stanford Large Network Dataset Collection includes various types of sentiment data, as well as Amazon product review data. A wealth of movie review data is available from Cornell. You may find that working with a more focused set of data, rather than general Twitter data, yields more accurate results.

- **Try Other Classifier Models**

  We applied a Naive Bayesian classifier to sentiment data. The Natural Language Toolkit supports several classification algorithms, including maxent (maximum entropy) and support vector machines (SVMs) through the scikitlearn module. Depending on the type of data you're analyzing and the features you choose to evaluate, other algorithms may produce better output. For more information, read Learning to Classify Text in the book *Natural Language Processing with Python*.

- **Script Your Hive Queries**

  Interactively querying data is the most direct way to get results, and interactive queries can help you explore data and refine your approach. After you've created a set of queries that you want to run regularly, you can automate the process by saving your Hive commands as a script and uploading the script to Amazon S3. For more information on how to launch a cluster using a Hive script, see Launch a Hive Cluster in the *Amazon Elastic MapReduce Developer Guide*.

- **Use Pig Instead of Hive to Analyze Your Data**

  Amazon EMR provides access to many open-source tools, including Pig, which uses a language called Pig Latin to abstract map-reduce jobs. For an example of how to analyze log files with Pig, see Parsing Logs with Apache Pig and Amazon Elastic MapReduce.

- **Create Custom Applications to Analyze Your Data**

If you're not able to find an open-source tool that meets your needs, you can write a custom Hadoop map-reduce application and run it on Amazon EMR. For more information, see Run a Hadoop Application to Process Data in the *Amazon Elastic MapReduce Developer Guide*.

Alternatively, you can create a Hadoop streaming job that reads data from standard input. For an example, see Tutorial: Sentiment Analysis (p. 5) in this guide. For more details, see Launch a Streaming Cluster in the *Amazon Elastic MapReduce Developer Guide*.

Related Resources

The following table lists some of the AWS resources that you'll find useful as you work with AWS.

Resource	Description
AWS Products & Services	Information about the products and services that AWS offers.
AWS Documentation	Official documentation for each AWS product, including service introductions, service features, and API reference.
AWS Discussion Forums	Community-based forums for discussing technical questions about Amazon Web Services.
Contact Us	A central contact point for account questions such as billing, events, and abuse. For technical questions, use the forums.
AWS Support Center	The hub for creating and managing your AWS Support cases. Also includes links to other helpful resources, such as forums, technical FAQs, service health status, and AWS Trusted Advisor.
AWS Support	The home page for AWS Support, a one-on-one, fast-response support channel to help you build and run applications in the cloud.
AWS Architecture Center	Provides the necessary guidance and best practices to build highly scalable and reliable applications in the AWS cloud. These resources help you understand the AWS platform, its services and features. They also provide architectural guidance for design and implementation of systems that run on the AWS infrastructure.
AWS Security Center	Provides information about security features and resources.
AWS Economics Center	Provides access to information, tools, and resources to compare the costs of Amazon Web Services with IT infrastructure alternatives.
AWS Technical Whitepapers	Provides technical whitepapers that cover topics such as architecture, security, and economics. These whitepapers have been written by the Amazon team, customers, and solution providers.

Resource	Description
AWS Blogs	Provides blog posts that cover new services and updates to existing services.
AWS Podcast	Provides podcasts that cover new services, existing services, and tips.