Andrea Gili - 20024498

Analyze Big Data on Hadoop using Amazon EMR

⚠ DISCLAIMER #1:

This tutorial is based on qwiklabs.com lab named “Analyze Big Data on Hadoop”. In order to perform this tutorial you either need a personal AWS account with a credit card associated or use a valid qwiklabs account with at least 1 credit. No other methods (i.e. an AWS Educate account) are allowed since the lack of privileges of that kind of account.

🎯GOAL:

The goal of this tutorial is to deploy a fully functional Hadoop cluster aiming to analyze log data from Amazon CloudFront, which are stored in an Amazon S3 bucket, using a HiveQL script and then download the results locally on your computer. We can thereafter divide this tutorial in 5 tasks:

1. Create an Amazon S3 bucket
2. Deploy a Hadoop Cluster within Amazon EMR
3. Process Amazon CloudFront sample data by running a HiveQL script
4. Download and check the Results
5. Terminate your Amazon EMR cluster

🔧TOOLS & SERVICES YOU NEED TO KNOW BEFORE STARTING:

- AMAZON EMR

  Amazon EMR is a managed cluster platform that simplifies running big data frameworks, such as Apache Hadoop and Apache Spark, on AWS to process and analyze vast amounts of data.

- AMAZON S3

  Amazon S3 is object storage built to store and retrieve any amount of data from anywhere. It’s a simple storage service that offers industry leading durability, availability, performance, security, and virtually unlimited scalability at very low costs.

- AMAZON CLOUDFRONT

  Amazon CloudFront is a fast content delivery network (CDN) service that securely delivers data, videos, applications, and APIs to customers globally with low latency, high transfer speeds, all within a developer-friendly environment.

- APACHE HADOOP

  Apache Hadoop is an open source framework that is used to efficiently store and process large datasets ranging in size from gigabytes to petabytes of data. Instead of using one large computer to store and process the data, Hadoop allows clustering multiple computers to analyze massive datasets in parallel more quickly.

- APACHE HIVE

  Apache Hive is a distributed, fault-tolerant data warehouse system, built on top of Hadoop, that enables analytics at a massive scale. A data warehouse provides a central store of information that can easily be analyzed to make informed, data driven decisions. Hive allows users to read, write, and manage petabytes of data using SQL (HiveQL).

⚠ DISCLAIMER #2:

In the following instruction we assume that you own a valid qwiklabs account with at least 1 credit to perform these tasks. Minor changes may be experienced using a personal AWS account with a credit card associated, however we try to stress some of them in this tutorial.

📌TASK 0: Start Lab (ONLY IF YOU USE A QWIKLABS ACCOUNT)

- Follow this Link and press the green button **Start Lab** on the top left of your screen to start the provisioning of your lab resources. If you are prompted for a token, use the credits you have purchased to continue the process by pressing **Launch with 1 credit**. The provisioning can take some minutes.
- When the provisioning ends, press on the blue button **Open Console** on the top left of your screen and you will be automatically redirected and logged in to the AWS management console. Remember to Log out first from your personal AWS if you own one.

📌TASK 1: Create an Amazon S3 bucket

In this task we want to create a bucket to store the output of the HiveQL script we will run on our log files later in this tutorial.

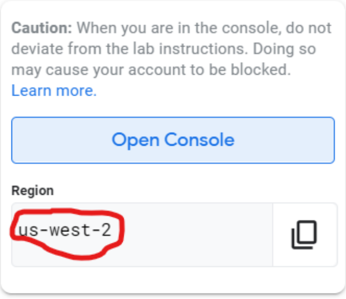
- In the **AWS management console**, on the **services** menu, click on **S3**
- Press on the button **Create Bucket**
- Give a name to your bucket filling **Bucket name** field (e.g. “*hadoop*<NUMBER>” where <NUMBER> could be your student ID number)
- Press then **Create Bucket** on the bottom of the page

The provisioning process may take some time wait until your bucket is successfully created

📌TASK 2: Deploy a Hadoop Cluster within Amazon EMR

In this task we want to create an Hadoop cluster that will be used to process CloudFront data later.

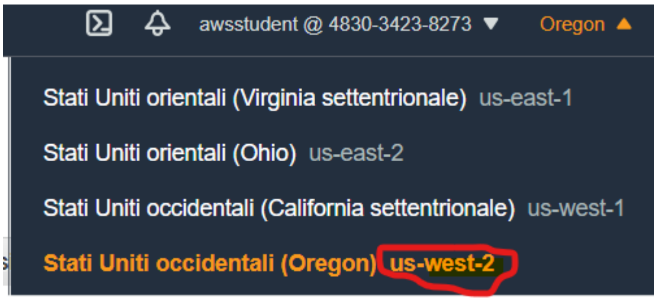
- Go back to **AWS management console** by pressing the **AWS icon** in the navigation bar and, on the **services** menu, select **EMR**
- Check that the **region** located in the top left tab on the **qwiklabs web page** matches the **region** located on top right of your **AWS web page**. If not, change the value of the region on AWS accordingly.



Caution: When you are in the console, do not deviate from the lab instructions. Doing so may cause your account to be blocked. Learn more.

Region

us-west-2



awsstudent @ 4830-3423-8273 ▼ Oregon ▲

Stati Uniti orientali (Virginia settentrionale) us-east-1

Stati Uniti orientali (Ohio) us-east-2

Stati Uniti occidentali (California settentrionale) us-west-1

Stati Uniti occidentali (Oregon) us-west-2

- Press on **Create Cluster** button
- Now select **Go to the advanced option**
- First, leave default values in this page and press **next** on the bottom of the screen
- Then, on the **Hardware Configuration** page, set the following parameters:
  - **Network**: LabVPC
  - In the **Master** and **Core** row change the instance type, by clicking on the pencil icon, to **m4.large** and press **save**. If **m4.large** is not temporarily available try with **m5.xlarge**. No other instance types are allowed in this tutorial. (if you use your personal account select an instance without restriction accordingly to your budget)
  - Make sure that **Instance Count** is **1** either for the **Master** node and the **Core** node and that every other instance count is set to **0**.
  - Select then **next** on the bottom of the page.
- Then, on the **General Options** page, configure these parameters:
  - **Cluster name**: *My cluster*
  - **S3 folder**: click on the folder icon, select the **Hadoop bucket** you created earlier (i.e. *hadoop*<NUMBER>) and press on **Select** button
  - **remove** the tick on **Termination protection**
  - Press **next** Button
- Then, on the **Security Options** page, configure these parameters:
  - **Permission**: Custom
  - **EMR role**: EMR_DefaultRole
  - **EC2 instance Profile**: EMR_EC2_DefaultRole
- Finally, launch the EMR Cluster by pressing **Create cluster** on the bottom of the screen

The cluster will take approximately 10 minutes to come online. Wait until your cluster status is Waiting. Try to refresh the page every few minutes and check for the status if the page seems unresponsive.

📌TASK 3: Process Amazon Cloudfront sample data by running a HiveQL script

In this task we will run a Hive script in your Hadoop cluster as a **step** in Amazon EMR to process our CloudFront data sample. A step is one or a series of Hadoop jobs you need to execute on your cluster. This script, in brief, counts the number of Cloudfront requests grouped by operating system in a given time frame. In this tutorial the HiveQL script is already written and placed in a known location. If you want to replicate this tutorial on your personal account or you are simply curious you can find some more information about Cloudfront and how the script should look like. You can find those deepenings in the **EXTRA** section of this tutorial.

- In your EMR cluster page, click the **Steps** tab (alternatively go back to **AWS management console** pressing the **AWS icon** in the navigation bar and, on the **services** menu, select **EMR**. Click on the EMR cluster you just created and then click the **Steps** tab)
- Then select **Add step**
- Now you need to configure the following parameters:
  - **Step type**: Hive program
  - **Name**: *Process logs*
  - **Script S3 location**: `s3://<REGION>.elasticmapreduce.samples/cloudfront/code/Hive_CloudFront.q`
    Where `<REGION>` is the region you can find on qwiklabs page or in the top right corner of your screen on AWS (e.g. `us-west-2`)
  - **Input S3 location**: `s3://<REGION>.elasticmapreduce.samples`
    Where `<REGION>` is the region you can find on qwiklabs page or in the top right corner of your screen on AWS (e.g. `us-west-2`)
  - **Output S3 location**: Click on the folder icon and then select the S3 bucket you created before (i.e. *hadoop*<NUMBER>)
  - **Arguments**: `-hiveconf hive.support.sql11.reserved.keywords=false`
  - Then click on **Add**

The step will take 1-2 minutes to run and its status will change from **Pending** to **Running** to **Completed**. Wait until Completed before you continue with the next task.

📌TASK 4: Download and check the Results

In this task you will download the output, produced by the HiveQL script and stored in the S3 bucket, and check the results.

- Go back to **AWS management console** pressing the **AWS icon** in the navigation bar and, on the **services** menu, click on **S3**
- Click on the **bucket** you created before (i.e. *hadoop*<NUMBER>)
- Choose the **os_reques**t folder, which contains two text files
- **Flag** and **Download** each file, one by one, in the folder
- **Open** these files with a text editor of your choice, you should then see the number of access requests by operating system.

📌TASK 5: Terminate your Amazon EMR cluster

Last but not least in this task you have to terminate your EMR cluster because we don't need it any more. Normally you should also delete your S3 bucket but in this tutorial (IF YOU USE A QWIKLABS ACCOUNT) it's not needed.

- Go back to the **AWS management console**, on the **services** menu, click on **EMR**
- **Flag** the **Hadoop cluster** you created before (i.e. *My cluster*)
- Click on **Terminate** button
- In the **Terminate cluster** dialog click again on **Terminate** and wait until the cluster is being terminated
- **Sign out** from the AWS account from the navigation bar on top of the screen
- Then (IF YOU USE A QWIKLABS ACCOUNT), on qwiklab webpage, press **End Lab** and then **OK**

🔎EXTRA:

In this tutorial we used a relatively small sample of CloudFront log data and a HiveQL script that performed some computation over those data without paying too much attention on how the data look like, nor the script.

The help you understand better these two element we'll provide in the following sections some deepenings, in particular we'll show you:

1. How Cloudfront log data look like and what are their main fields
2. What the HiveQL script is supposed to do and some code snippets taken from the original script of this tutorial

- **CLOUDFRONT LOG DATA STRUCTURE**

  This is an example of a cloudfront data log:

```
2017-07-05 20:05:47 SEA4 4261 10.0.0.15 eabcd12345678.cloudfront.net
/test-image-2.jpeg
```

```
Mozilla/5.0%20(MacOS;%20U;%20Windows%20NT%205.1;%20en-US;%20rv:1.9.0.9)%20Gecko/2009040821%20Chrome/3.0.9
```

Let's summarize its parts with this table:

FIELD	SAMPLE DATA	DEFINITION
Date	2017-07-05	The date on which the event occurred
Time	20:05:47	The time when Cloudfront server finished responding the the request (in UTC)
Edge Location	SEA4	The edge location that served the request identified by a 3 letter code and an arbitrary number
Bytes	4261	The total number of bytes that Cloudfront served to the viewer in response to request, including headers
IP	10.0.0.15	The IP address of the viewer that made the request.
Method	GET	The HTTP access method
Host	abcd.cloudfront.net	The domain main of the Cloudfront distributions
URI	/test-image-2.jpeg	The portions of the URI that identifies the path and object
Status	200	The HTTP status Code
Referrer	-	The name of the domain that originated the request
User Agent	Mozilla/5.0...	The user agent header that identifies the source of the request. **This field contains the operating system needed for the script**

- ## WHAT THE SCRIPT IS SUPPOSED TO DO

This is a high level explanation of what the original HiveQL script does, complimented by some piece of code taken from that script:

➔ Creates a **Hive Table** named *cloudfront_logs*

```
CREATE EXTERNAL TABLE IF NOT EXISTS cloudfront_logs (
DateObject Date,
Time STRING,
Location STRING,
Bytes INT,
RequestIP STRING,
Method STRING,
Host STRING,
Uri STRING,
Status INT,
Referrer STRING,
OS String,
Browser String,
BrowserVersion String
)
```

➔ Reads the **Cloudfront log files** from amazon S3 and parses the files using the Regular Expression Serializer/Deserializer (RegEx SerDe)

```
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.RegexSerDe'
WITH SERDEPROPERTIES (
  "input.regex" = "^(?!#)([^ ]+)\\s+([^ ]+)\\s+([^ ]+)\\s+([^
]+)\\s+([^ ]+)\\s+([^ ]+)\\s+([^ ]+)\\s+([^ ]+)\\s+([^ ]+)\\s+([^
]+)\\s+[^\(]+[\(](https://s3.us-west-2.amazonaws.com/us-west-2-aws-tra
ining/awsu-spl/spl-166/1.0.11.prod/instructions/en_us/[^\;]+).*\%20([^
\/]+)[\/](https://s3.us-west-2.amazonaws.com/us-west-2-aws-training/aw
su-spl/spl-166/1.0.11.prod/instructions/en_us/.*)$"
) LOCATION '${INPUT}/cloudfront/data/';
```

➔ Writes the parsed results to the cloudfront_logs Hive table

➔ Submits a HiveQL query against the data to retrieve the **total requests per operating system for a given time frame**

```
INSERT OVERWRITE DIRECTORY '${OUTPUT}/os_requests/'
SELECT
  os,
  COUNT(*) count
FROM cloudfront_logs
WHERE dateobject
BETWEEN '2014-07-05' AND '2014-08-05'
GROUP BY os;
```

- ➔ Writes the query results to your Amazon S3 output bucket

🚨TROUBLESHOOTING

- ***“The Step I add to the cluster have terminated prematurely”***
  Wait some minutes, then check if you entered correctly the S3 script location and S3 input location and eventually use the copy feature on qwiklabs page to paste the region in the path. Then try to run the step again.