Big Data on AWS

Lab One: Hosting a Static Website in AMAZON S3

Table of Contents

[Lab One: Hosting a Static Website in Amazon S3 3](#_Toc509604870)

[Task One: Creating a bucket and configuring it as a website 3](#_Toc509604871)

[Task Two: Add a Bucket Policy to Make Your Bucket Content Publicly Available 3](#_Toc509604872)

[Task Three: Uploading an Index Document 4](#_Toc509604873)

[Task Four: Test Your Website 4](#_Toc509604874)

[Task Five: Configure the Data bucket to allow CORS 4](#_Toc509604875)

[Task Four: Use Hive to query the tables in Amazon S3 5](#_Toc509604876)

# Lab One: Hosting a Static Website in Amazon S3

In this lab, you will use Amazon S3 to host a static website. Through this lab, you will perform the following actions:

* Task 1: Creating a Bucket and Configuring It as a Website
* Task 2: Adding a Bucket Policy That Makes Your Bucket Content Publicly Available
* Task 3: Uploading an Index Document
* Task 4: Testing Your Website
* Task 5: Enabling CORS

## Task One: Creating a bucket and configuring it as a website

1. Open Amazon S3 console.
2. Create 2 buckets named like data-[randomNumber] and site-[randomNumber]. Random Number is to ensure unique. Any region of your choice.
3. Download and extract the file from the next URL

<https://s3.ap-south-1.amazonaws.com/training-artifacts-m/static-web.zip>

1. Edit the index.html file and change the line number 131 to include your bucket name such as:

var s3exp\_config = { Bucket: **'test-data-121'**, Prefix: '', Delimiter: '/' };

1. Edit the index.html file and change the line number 136 to include your bucket name such as:
2. AWS.config.region = 'us-west-2';

## Task Two: Add a Bucket Policy to Make Your Bucket Content Publicly Available

1. In the Properties pane for the bucket, choose Permissions.
2. Choose Add Bucket Policy.
3. Copy the following bucket policy, and then paste it in the Bucket Policy Editor.

{

"Version":"2012-10-17",

"Statement":[{

"Sid":"PublicReadForGetBucketObjects",

"Effect":"Allow",

"Principal": "\*",

"Action":["s3:GetObject"],

"Resource":["arn:aws:s3:::example-bucket/\*"

]

}

]

}

1. In the policy, replace example-bucket with the name of your bucket.
2. Choose Save.

## Task Three: Uploading an Index Document

1. Upload the html files from site directory to site-[random] bucket and data files from the data directory to the data-[random] bucket.
2. Choose site-[randomnumber] bucket.
3. Open the bucket **Properties** pane, choose **Static Website Hosting**, and do the following:
   1. Choose **Enable website hosting**.
   2. In the **Index Document** box, type the name of your index document. The name is typically index.html.
   3. In the **Error Document** box, type the name of your error document. The name is typically error.html.
   4. Choose **Save** to save the website configuration.
   5. Write down the **Endpoint**. This is the Amazon S3-provided website endpoint for your bucket. You use this endpoint in the following steps to test your website.

## Task Four: Test Your Website

1. Type the following URL in the browser, replacing example-bucket with the name of your bucket and website-region with the name of the AWS Region where you deployed your bucket.

[http://*example-bucket.*s3-website-*region*.amazonaws.com](http://example-bucket.s3-website-region.amazonaws.com)

1. If your browser displays your index.html page, the website was successfully deployed.
2. However, you may not be able to see the data files listed because by default the site bucket cannot fetch data from the data bucket. We need to enable CORS for that.

## Task Five: Configure the Data bucket to allow CORS

1. Choose the Data bucket and click on permissions.
2. Click on CORS Configurations.
3. Leave the default CORS policy and save.

i.ip

FROM

impressions i

WHERE

i.dt >= '${DAY}-${HOUR}-00' and i.dt < '${NEXT\_DAY}-${NEXT\_HOUR}-00'

;

1. The start of the time period is DAY-HOUR and the end of the period is NEXT\_DAY-NEXT\_HOUR. NEXT\_DAY is the day of the next time period. It differs from ${DAY} only when we're processing the last hour of a day. In this case the time period ends on the next day.

For clicks, we extend the period of time over which we join by 20 minutes. Meaning we accept a click that occurred up to 20 minutes after the impression.

CREATE TABLE tmp\_clicks (

impressionId string

) STORED AS SEQUENCEFILE;

INSERT OVERWRITE TABLE tmp\_clicks

SELECT

impressionId

FROM

clicks c

WHERE

c.dt >= '${DAY}-${HOUR}-00' AND c.dt < '${NEXT\_DAY}-${NEXT\_HOUR}-20'

;

1. Now we combine the impressions and clicks tables using a left outer join. This way any impressions that did not result in a click are preserved. This join also enables us to search for clicks that occurred after the time period. The query also excludes any clicks that did not originate from an impression in the selected time period.

INSERT OVERWRITE TABLE joined\_impressions PARTITION (day='${DAY}', hour='${HOUR}')

SELECT

i.requestBeginTime, i.adId, i.impressionId, i.referrer, i.userAgent, i.userCookie,

i.ip, (c.impressionId is not null) clicked

FROM

tmp\_impressions i LEFT OUTER JOIN tmp\_clicks c ON i.impressionId = c.impressionId

;

All the above could be run in script mode as well.

## Task Six: Use Hive to query the tables in Amazon S3

1. Observe the output in the output S3 bucket from S3 console.
2. Because the joined\_impressions table is located in Amazon S3 this data is now available for other jobs to use.