**December, 2018**



**Immersion Day**

Serverless Data Lake with AWS

# Table of Contents

[Table of Contents 2](#_Toc1030448)

[Revision History 3](#_Toc1030449)

[Disclaimer 3](#_Toc1030450)

[Lab Pre-Requisites 4](#_Toc1030451)

[Expected Costs 4](#_Toc1030452)

[Lab 2.4 Working with a 170GB Public Dataset (Global DB of Events, Language & Tone) 6](#_Toc1030453)

[Summary & Next Steps 18](#_Toc1030454)

[Deleting Lab Resources 18](#_Toc1030455)

[References 19](#_Toc1030456)

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Revision | Date | By | Change Log |
| PA6 | 03-Dec-2018 | @akirmak | First 3 labs reer reviewed by @baya |
| PA7 | 13-Dec-2018 | @akirmak | Initial draft, ready for internal joint review. |
| PA8 | 14-Dec-2018 | @akirmak | Joint reviewed by @halilb and @serdarn |
| PA9 | 16-Dec-2018 | @akirmak | Joint review feedback incorporated. Fixed an error in Lab 2.4 instructions. Shortened Lab 3.1 |

# Disclaimer

The work is provided “as is” without warranties or conditions of any kind, either express or implied, including warranties.

# Lab Pre-Requisites

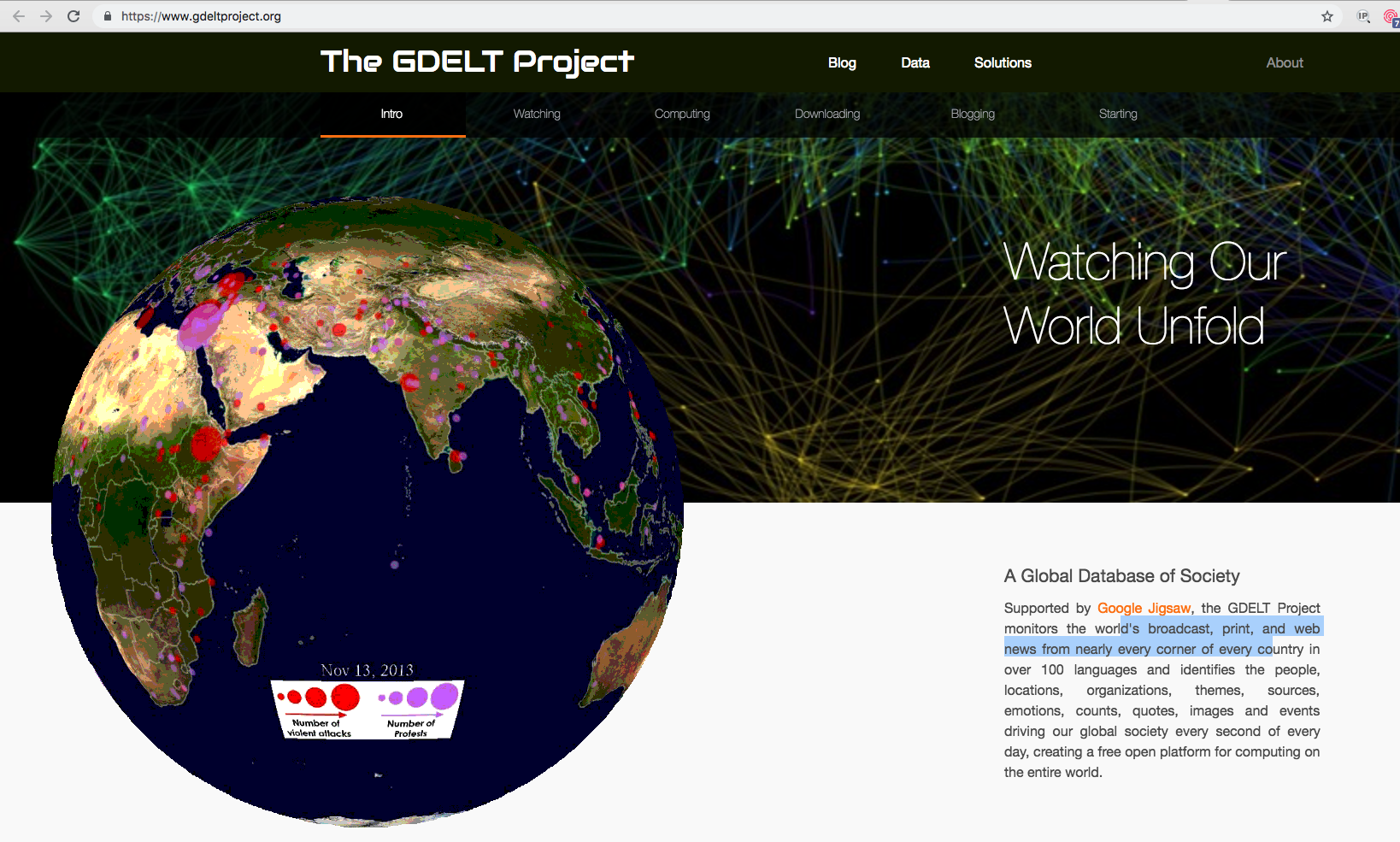
In order to complete this immersion day lab, you need to have an AWS account with Access Glue and Athena services

# Expected Costs

* You are responsible for the cost of the AWS services used while running this lab. As of the date of publication, the baseline cost for running this solution as-is should be around:
  + Athena: < 5$
  + Glue < 1 $
  + S3 < 1$

# Lab 2.4 Working with a 170GB Public Dataset (Global DB of Events, Language & Tone)

In the previous labs, you worked with an extremly small dataset (less than < 10MB) and with a single data source. In this lab, let’s use a public dataset with bigger size and more tables and observe various services.



The [Global Database of Events, Language and Tone (GDELT) Project](http://www.gdeltproject.org/) monitors the world's broadcast, print, and web news from nearly every corner of every country in over 100 languages and identifies the people, locations, organisations, counts, themes, sources, emotions, counts, quotes, images and events driving our global society every second of every day. The data set v1.0 is publicly available in S3 in the [Registry of Open Data on AWS](https://aws.amazon.com/public-datasets/gdelt/).

In this lab, you will explore, catalogue, visualize, interact with this data using AWS services.

The data set we will use contains (at the time to writing) thousands of uncompressed CSV files: hundreds of millions of lines,and is about 170+ GB. The Data format is defined [here](http://data.gdeltproject.org/documentation/GDELT-Data_Format_Codebook.pdf). The queries below are from Julien Simon’s blog.

You will use Athena to define columns you needed with the right type and point to the S3 bucket holding all files (in another AWS account). Athena will also be used in later labs to query data hosted in S3. There is no infrastructure to launch or manage, no data preparation, no loading time etc. needed.

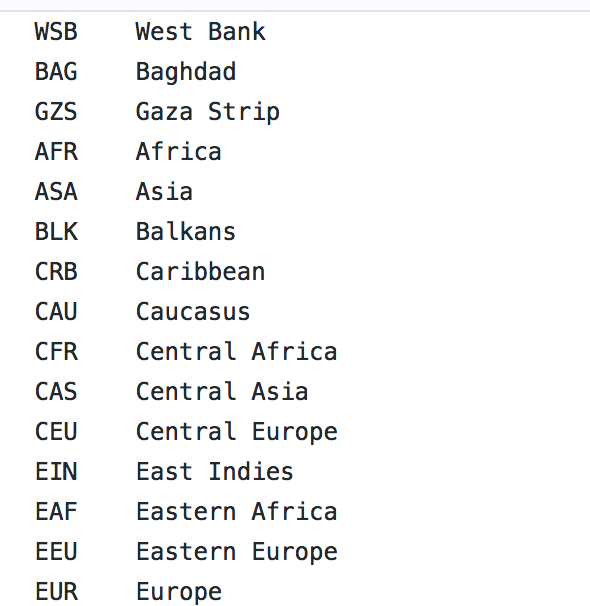
1. Open Athena and enter following HIVE DDL statements below
2. Create a Database in Glue Metadata store

|  |
| --- |
| CREATE DATABASE gdelt; |

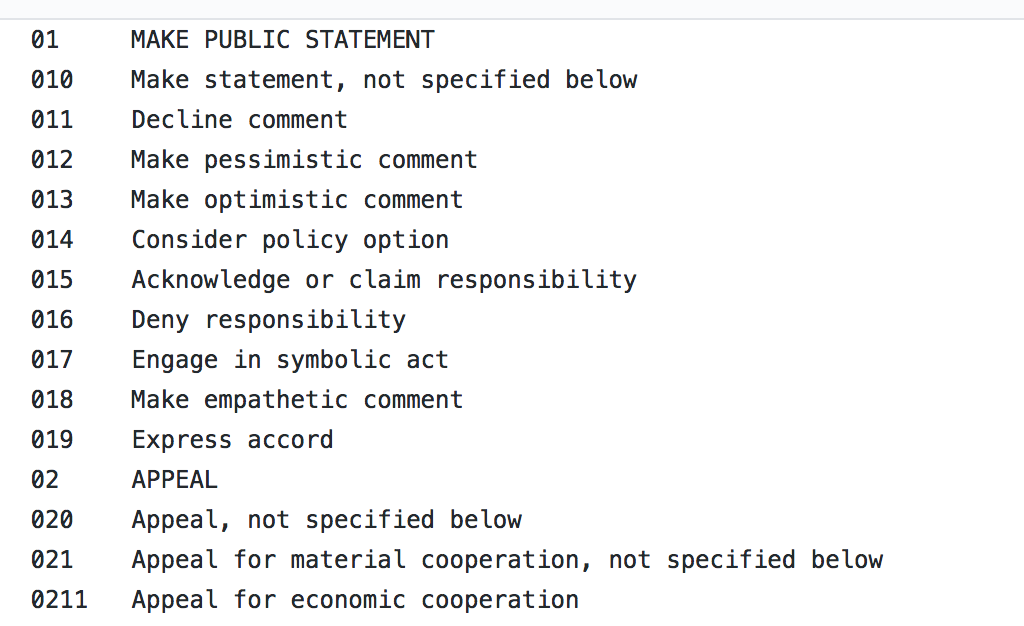
1. Create a Table referring to the S3 bucket holding all files in the AWS account.
   1. Before we proceed, a few remarks:
      1. We are creating a schema definition in our Glue service, in our Data Catalogue.
      2. The actual data is in another AWS, and in another region (if you are working on a region not in us-east-1 (Northern Virginia Region)
      3. You can Access this data, because it is a public dataset located in 's3://gdelt-open-data/events/folder, and is open to everyone.
      4. Although we are creating TABLEs, there is no database. Events table is a representation of thousands of TSV (Tab Seperated Files) files stored in S3. Technologies like Apache HIVE and Presto enables accessing them using SQL like expressions.

|  |
| --- |
| CREATE EXTERNAL TABLE IF NOT EXISTS gdelt.events (  `globaleventid` INT,  `day` INT,  `monthyear` INT,  `year` INT,  `fractiondate` FLOAT,  `actor1code` string,  `actor1name` string,  `actor1countrycode` string,  `actor1knowngroupcode` string,  `actor1ethniccode` string,  `actor1religion1code` string,  `actor1religion2code` string,  `actor1type1code` string,  `actor1type2code` string,  `actor1type3code` string,  `actor2code` string,  `actor2name` string,  `actor2countrycode` string,  `actor2knowngroupcode` string,  `actor2ethniccode` string,  `actor2religion1code` string,  `actor2religion2code` string,  `actor2type1code` string,  `actor2type2code` string,  `actor2type3code` string,  `isrootevent` BOOLEAN,  `eventcode` string,  `eventbasecode` string,  `eventrootcode` string,  `quadclass` INT,  `goldsteinscale` FLOAT,  `nummentions` INT,  `numsources` INT,  `numarticles` INT,  `avgtone` FLOAT,  `actor1geo\_type` INT,  `actor1geo\_fullname` string,  `actor1geo\_countrycode` string,  `actor1geo\_adm1code` string,  `actor1geo\_lat` FLOAT,  `actor1geo\_long` FLOAT,  `actor1geo\_featureid` INT,  `actor2geo\_type` INT,  `actor2geo\_fullname` string,  `actor2geo\_countrycode` string,  `actor2geo\_adm1code` string,  `actor2geo\_lat` FLOAT,  `actor2geo\_long` FLOAT,  `actor2geo\_featureid` INT,  `actiongeo\_type` INT,  `actiongeo\_fullname` string,  `actiongeo\_countrycode` string,  `actiongeo\_adm1code` string,  `actiongeo\_lat` FLOAT,  `actiongeo\_long` FLOAT,  `actiongeo\_featureid` INT,  `dateadded` INT,  `sourceurl` string  )  ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe'  WITH SERDEPROPERTIES (  'serialization.format' = '\t','field.delim' = '\t') LOCATION 's3://gdelt-open-data/events/'; |

1. Create lookup tables.
   1. There are a few tables in the GDELT dataset, and they provide human-friendly descriptions to event codes, country codes etc. in the Gdelt.Events table defined in the previous step. They are also TSV files stored in S3.
      1. The **Countries** file that will be used as a lookup table looks like below: <https://www.gdeltproject.org/data/lookups/CAMEO.country.txt>



* + 1. The **EventCodes** file that will be used as a lookup table looks like below: <https://www.gdeltproject.org/data/lookups/CAMEO.eventcodes.txt>



* + 1. The **Groups** file that will be used as a lookup table looks like below:

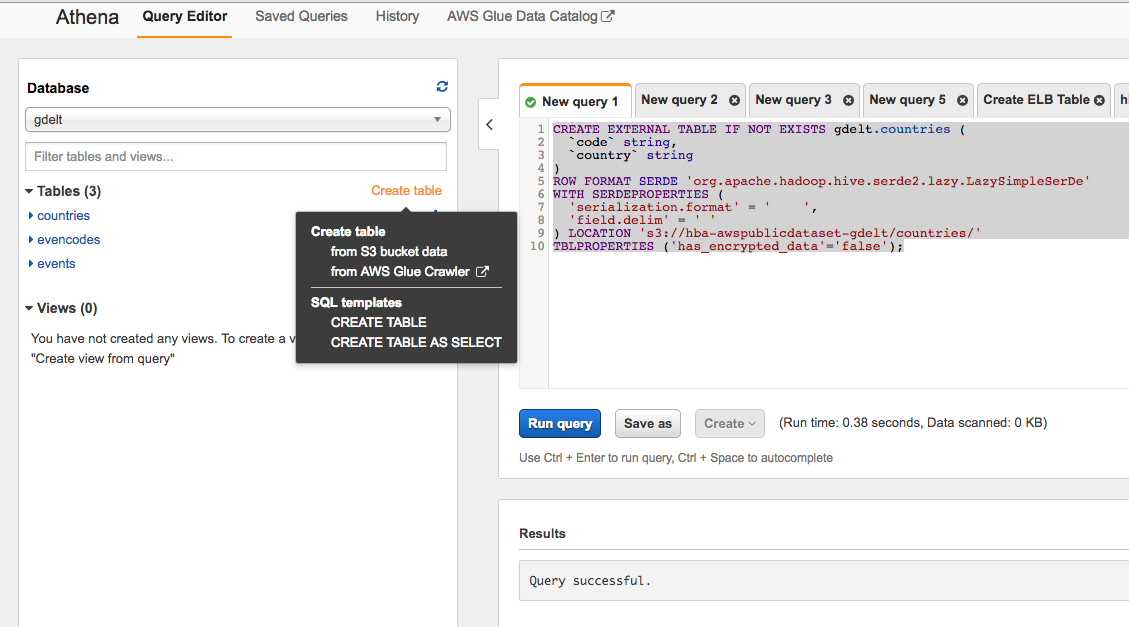


* + 1. The **Types** file that will be used as a lookup table looks like below:

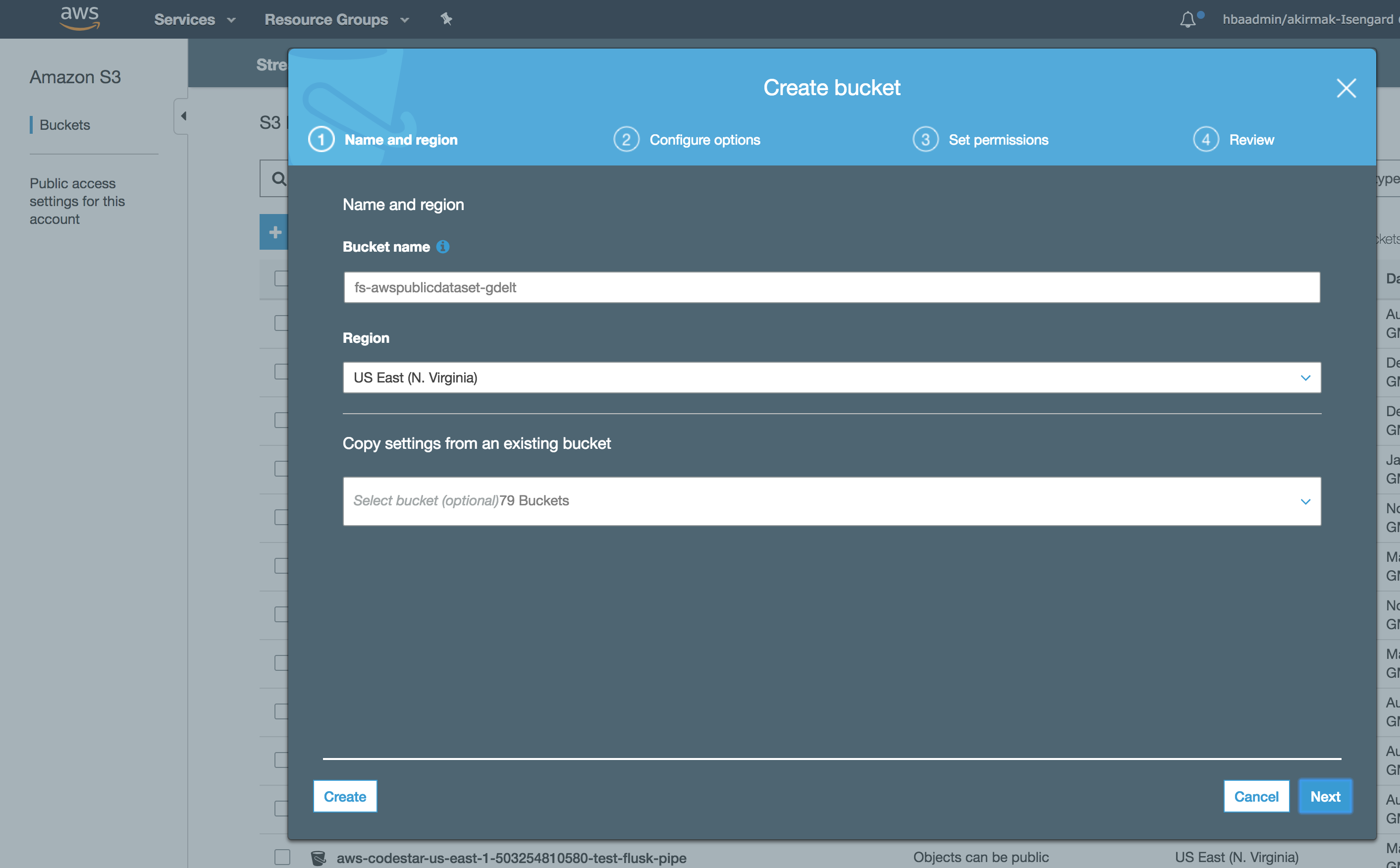


* 1. To maket his exercise interesting, you will store these files in your S3 bucket. The Events table and the lookup tables will be used in a few queries using Athena..

1. Upload lookup table files to your S3 bucket.
   1. **Pro Tip:** There are multiple alternatives to create tables in Glue metadata catalogue. We will use one of them, but if you are interested other options:
      1. First option: You can create a table with CREATE TABLE HIVE DDL command in the Athena Console (You already did in the previous step for the GDELT **Events** table.)
      2. Second option: Another option is to use an AWS Glue Crawler (You already did this for the **raw** table in the previous lab.)
      3. Third option: You can manually create TABLE from S3 bucket data. This option is useful for very simple data such as the lookup tables we will define (They only have 2 columns):

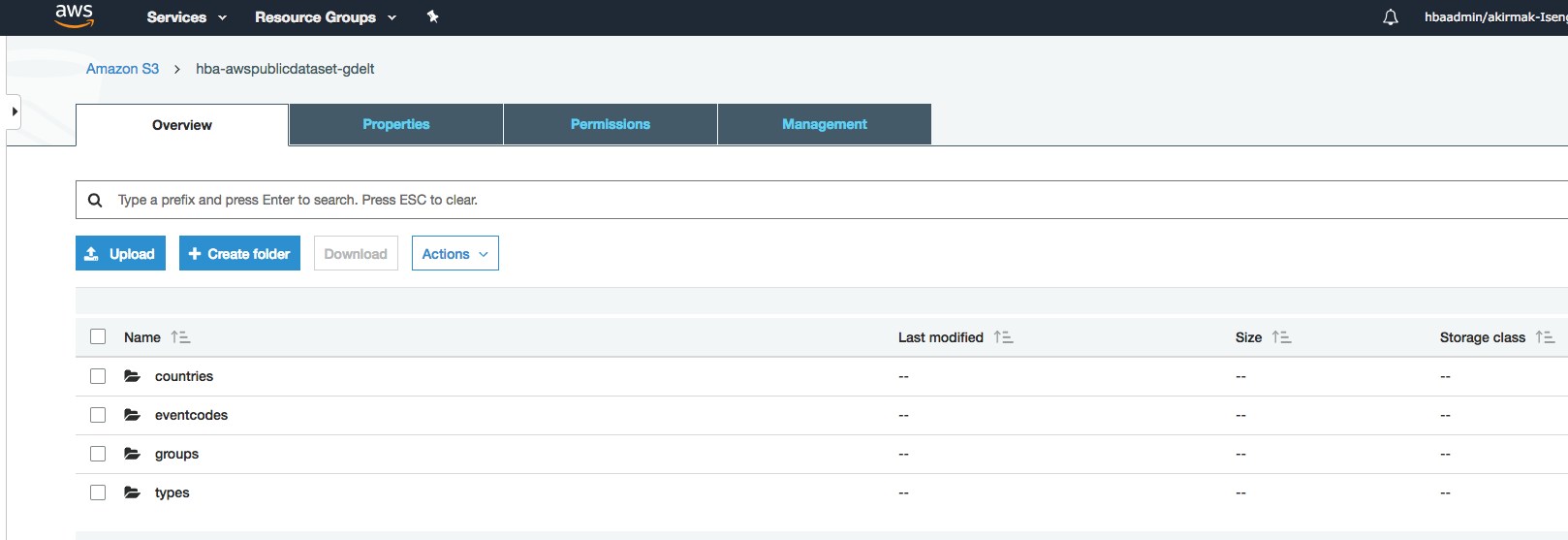


1. Let’s use First option (Athena DDL) fort he following files:
2. First, download the files below to your computer:
   1. Eventcodes: <https://www.gdeltproject.org/data/lookups/CAMEO.eventcodes.txt>
   2. Countries: <https://www.gdeltproject.org/data/lookups/CAMEO.country.txt>
   3. Types: <https://www.gdeltproject.org/data/lookups/CAMEO.type.txt>
   4. Groups: <https://www.gdeltproject.org/data/lookups/CAMEO.knowngroup.txt>
3. Open S3 from the console
4. Create a bucket **<your-initials>-awspublicdataset-gdelt** (e.g. fs-awspublicdataset-gdelt) as below. Use the “**Create**” button on the bottom-left instead of the “**Next**” button on the bottom-right. This option keep default options for everything, and you can create your bucket in one step.



1. Open your new bucket and create the following 4 folders under bucket (all in small letters)
   1. Folder 1: **countries**
   2. Folder 2: **eventcodes**
   3. Folder 3: **groups**
   4. Folder 4: **types**

under your new bucket.



1. Put the corresponding file under each bucket (e.g.
   1. Upload **CAMEO.eventcodes.txt** file from your computer under **eventcodes**
   2. Upload **CAMEO. countries.txt** file from your computer under **countries**
   3. Upload **CAMEO. groups.txt** file from your computer under **groups**
   4. Upload **CAMEO. types.txt** file from your computer under **types**
2. Goto Athena Console. You will add them to your data catalogue
3. Run the following DDL statements from the Athena Console fort he 4 lookup tables. **Important:** Replace YOURINITIALS in bucket name before running the DDL.
4. Add **eventcodes** to the data catalogue by pasting the DDL below to Athena Console, replacing your initials and selecting “Run Query”

|  |
| --- |
| CREATE EXTERNAL TABLE IF NOT EXISTS gdelt.eventcodes (  `code` string,  `description` string  )  ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe'  WITH SERDEPROPERTIES (  'serialization.format' = '\t','field.delim' = '\t')  LOCATION 's3://**<YOUR INITIALS>**-awspublicdataset-gdelt/eventcodes'  TBLPROPERTIES ( "skip.header.line.count"="1") |

1. Add **types** to the data catalogue by pasting the DDL below to Athena Console, replacing your initials and selecting “Run Query”

|  |
| --- |
| CREATE EXTERNAL TABLE IF NOT EXISTS gdelt.types (  `type` string,  `description` string  )  ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe'  WITH SERDEPROPERTIES (  'serialization.format' = '\t','field.delim' = '\t')  LOCATION 's3://**<YOUR INITIALS>**-awspublicdataset-gdelt/types/'  TBLPROPERTIES ( "skip.header.line.count"="1"); |

1. Add **groups** to the data catalogue by pasting the DDL below to Athena Console, replacing your initials and selecting “Run Query”

|  |
| --- |
| CREATE EXTERNAL TABLE IF NOT EXISTS gdelt.groups (  `group` string,  `description` string  )  ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe'  WITH SERDEPROPERTIES (  'serialization.format' = '\t','field.delim' = '\t')  LOCATION 's3://**<YOUR INITIALS>**-awspublicdataset-gdelt/groups/'  TBLPROPERTIES ( "skip.header.line.count"="1"); |

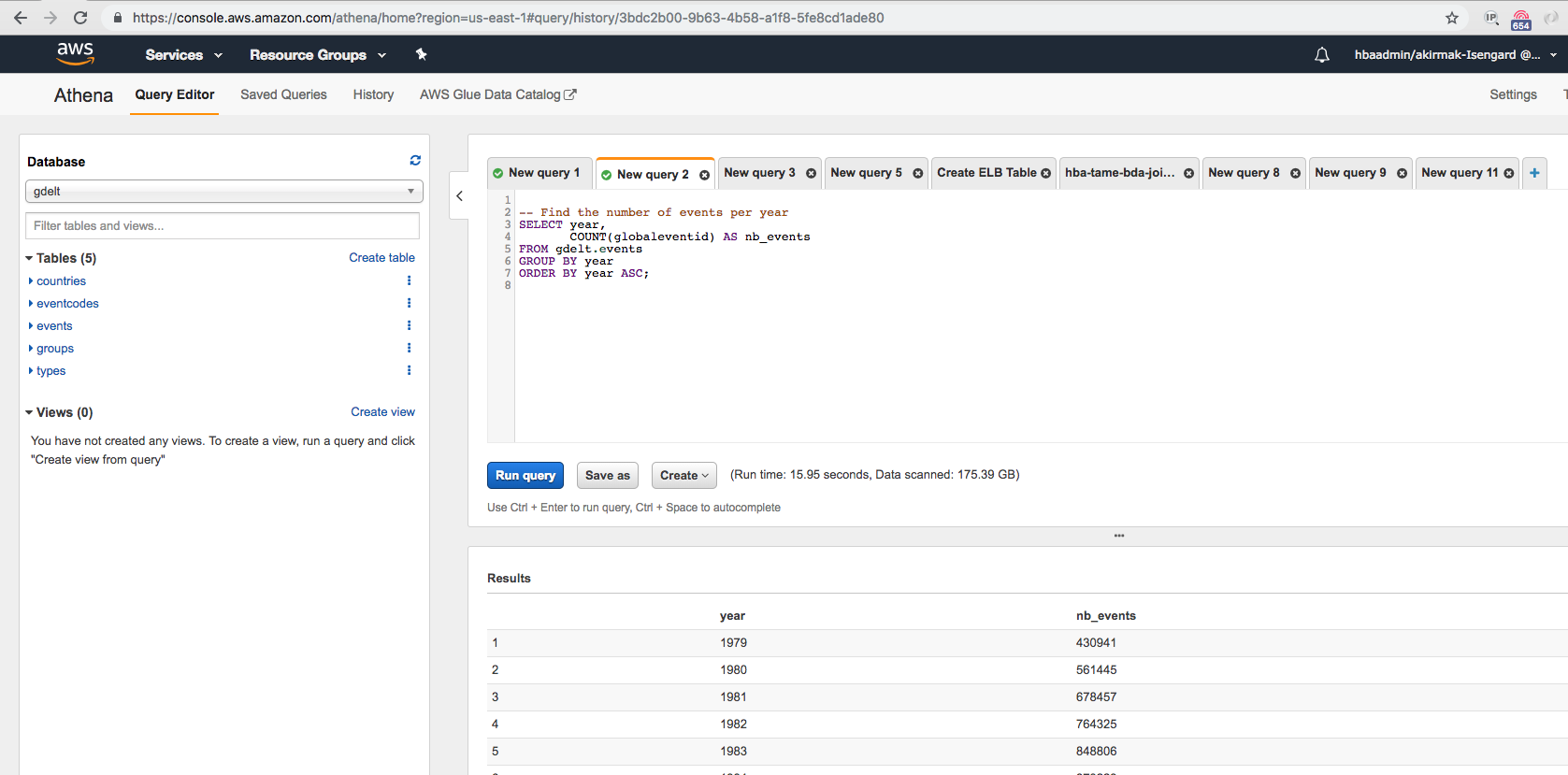
1. Add **countries** to the data catalogue by pasting the DDL below to Athena Console, replacing your initials and selecting “Run Query”

|  |
| --- |
| CREATE EXTERNAL TABLE IF NOT EXISTS gdelt.countries (  `code` string,  `country` string  )  ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe'  WITH SERDEPROPERTIES (  'serialization.format' = '\t','field.delim' = '\t')  LOCATION 's3://**<YOUR INITIALS>**-awspublicdataset-gdelt/ countries/'  TBLPROPERTIES ( "skip.header.line.count"="1"); |

1. Now, explore the data with the queries below.
2. First find the number of events per year from the **Events** table
3. Find the number of events per year

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | |  | -- Find the number of events per year  SELECT year, | |  | COUNT(globaleventid) AS nb\_events | |  | FROM gdelt.events | |  | GROUP BY year | |  | ORDER BY year ASC; | |

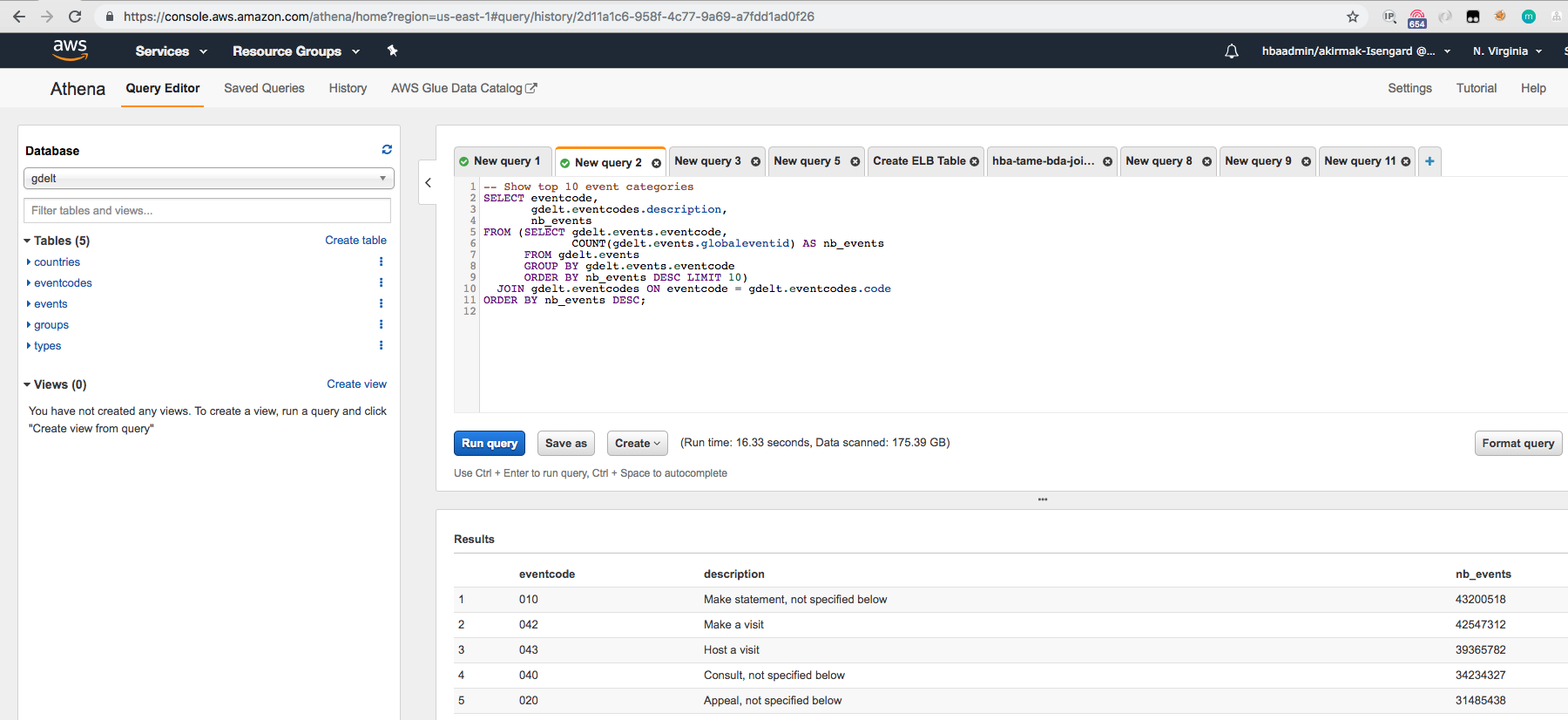
Output:



1. Notice the data amount scanned? The results are returned in less than 30 seconds by scanning 175 GB of data from thousands of uncompressed CSV files on S3. That’s the power of HIVE, Presto and other Hadoop Technologies simplified by Athena Service.
2. Now let’s show the sorted top 10 event categories by joining Events table and the Eventcode lookup table:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | |  | |  | -- Show top 10 event categories  SELECT eventcode, | |  | gdelt.eventcodes.description, | |  | nb\_events | |  | FROM (SELECT gdelt.events.eventcode, | |  | COUNT(gdelt.events.globaleventid) AS nb\_events | |  | FROM gdelt.events | |  | GROUP BY gdelt.events.eventcode | |  | ORDER BY nb\_events DESC LIMIT 10) | |  | JOIN gdelt.eventcodes ON eventcode = gdelt.eventcodes.code | |  | ORDER BY nb\_events DESC; | |

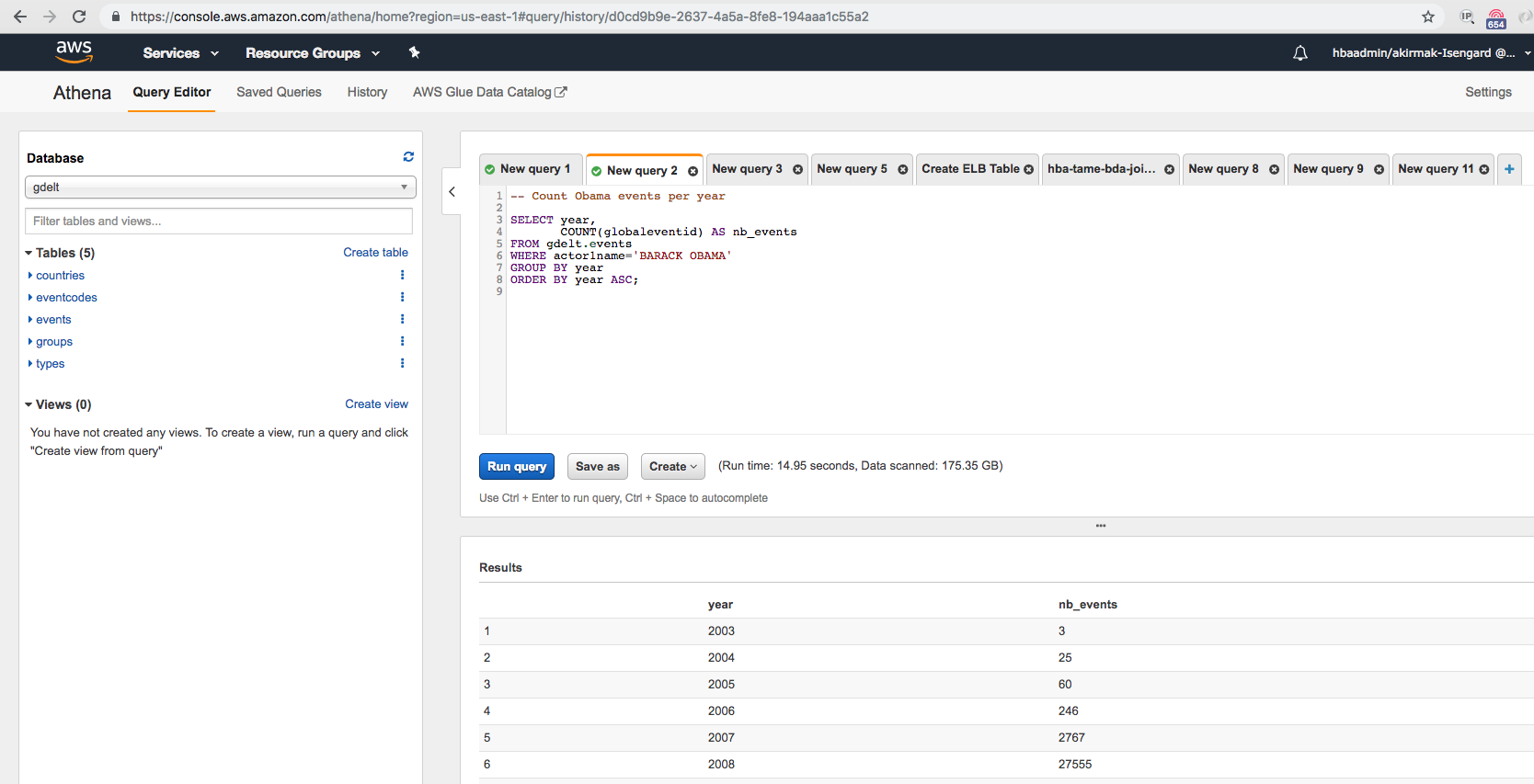
Output:



1. Count US President Obama events per year:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | |  | |  | -- Count Obama events per year  SELECT year, | |  | COUNT(globaleventid) AS nb\_events | |  | FROM gdelt.events | |  | WHERE actor1name='BARACK OBAMA' | |  | GROUP BY year | |  | ORDER BY year ASC; | |

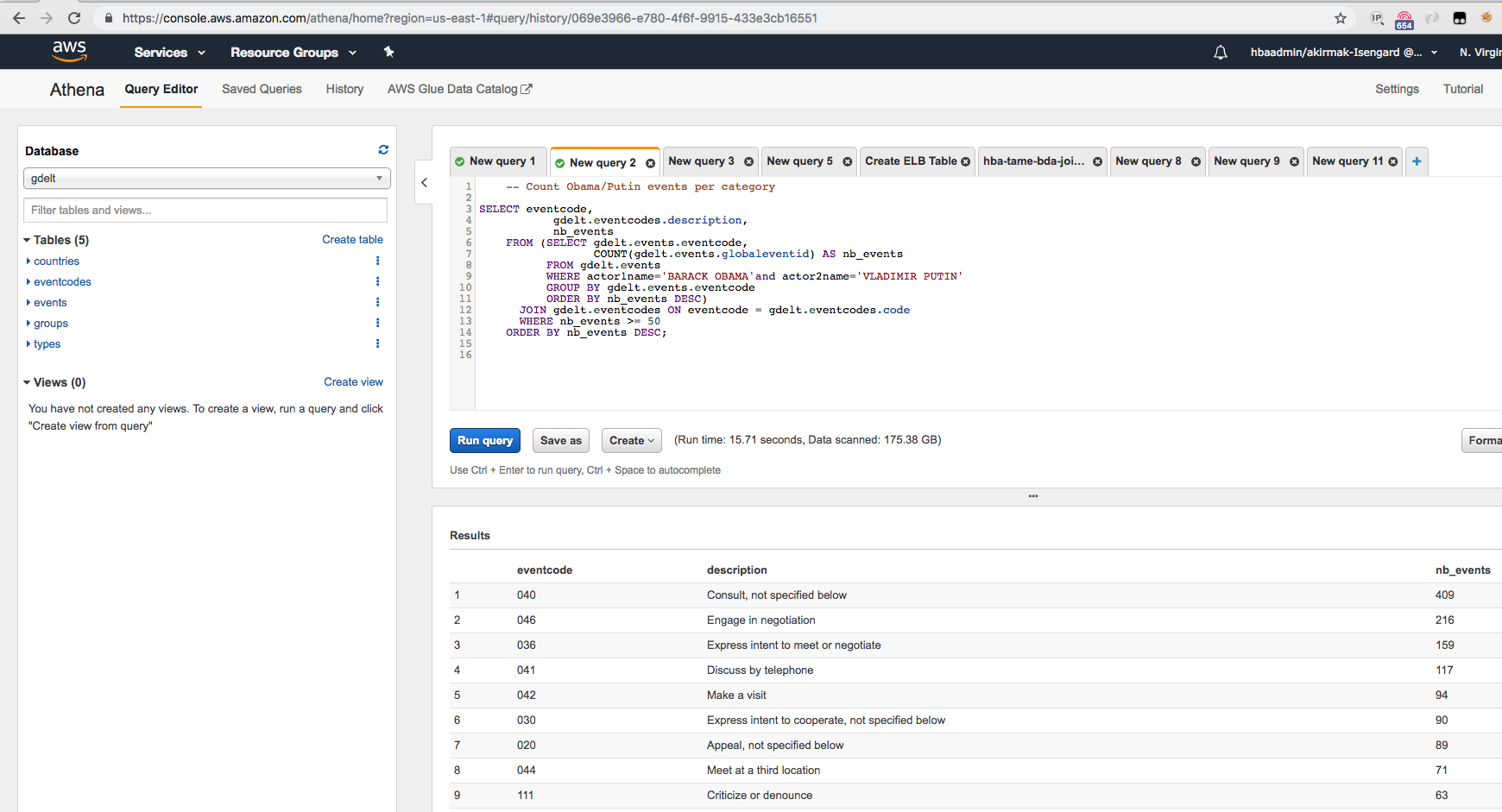
Output:



1. Count Obama/Putin events per category

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | |  | |  | |  | -- Count Obama/Putin events per category  SELECT eventcode, | |  | gdelt.eventcodes.description, | |  | nb\_events | |  | FROM (SELECT gdelt.events.eventcode, | |  | COUNT(gdelt.events.globaleventid) AS nb\_events | |  | FROM gdelt.events | |  | WHERE actor1name='BARACK OBAMA'and actor2name='VLADIMIR PUTIN' | |  | GROUP BY gdelt.events.eventcode | |  | ORDER BY nb\_events DESC) | |  | JOIN gdelt.eventcodes ON eventcode = gdelt.eventcodes.code | |  | WHERE nb\_events >= 50 | |  | ORDER BY nb\_events DESC; | |  |  | |

Output:



1. None of these took more than 30 seconds and that's with uncompressed CSV, the least performing data format possible. Converting the data set columnar formats such as Parquet would yield a massive improvement

# Summary & Next Steps

Congratulations. You have successfully catalogued and queried a big dataset. None of these took more than 30 seconds and that's with uncompressed CSV, the least performing data format possible.

So, what’s next? In the next lab, you will create reports and visualizations of the open dataset, as well as the data you’ve ingested in Lab1 and Lab 2.

# Deleting Lab Resources

If you won’t proceed to the next lab, make sure you terminate the resources below to avoid bills.

* S3:
  + Delete the buckets and the data inside containing the lookup tables.

# References

1. Website, AWS Data Lake, <https://aws.amazon.com/tr/big-data/datalakes-and-analytics/what-is-a-data-lake/>
2. Whitepaper, “Lambda Architecture for Batch and Stream Processing, AWS, October 2018, <https://d0.awsstatic.com/whitepapers/lambda-architecure-on-for-batch-aws.pdf>