Details

Dataset	Music Brainz dataset https://musicbrainz.org/doc/MusicBrainz_Database
Dataset description	This data includes information about artists, release groups, releases, recordings, works, and labels, as well as the many relationships between them. The database also contains a

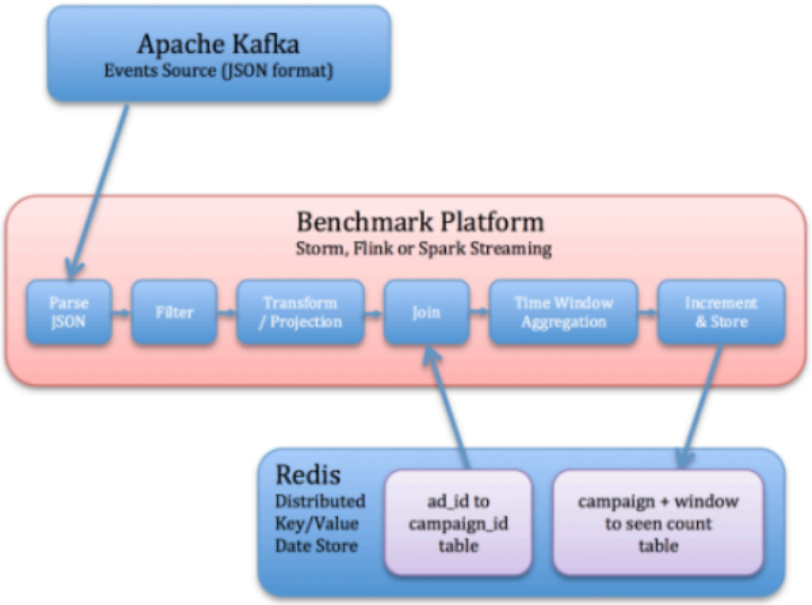
	full history of all the changes that the MusicBrainz community has made to the data

Introduction and motivation for the app and the choice of the technologies

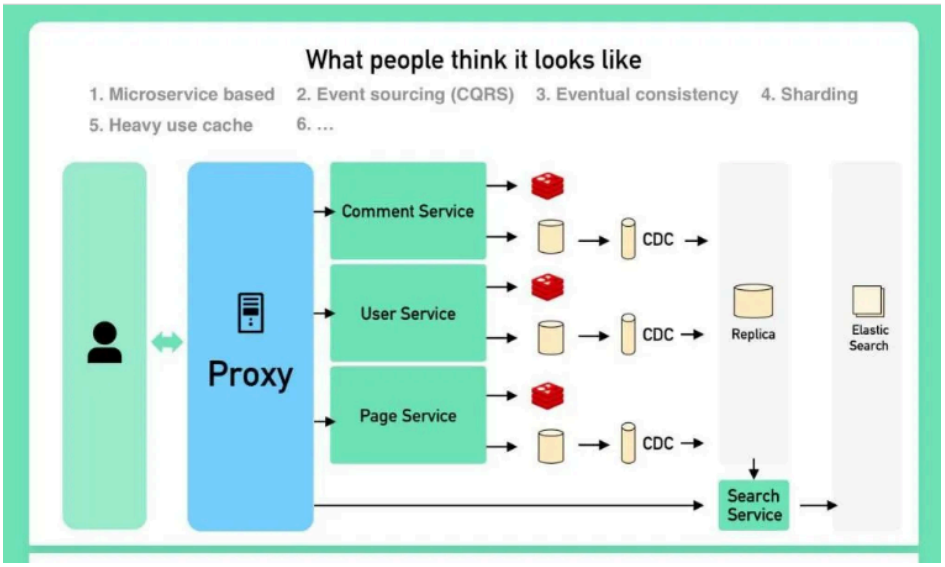
- So here for the initial steps, we identified the yardsticks for the challenge
    a. 1.4 millions music records storage, processing and analysing
    b. Selecting the right cloud service considering the free trials and ease of access within the limited credits
    c. Right database to process and analyse the big data
    d. Right distributed service for performing data visualisation and should be able to store and should readily be available for integrations
    e. Good dev community, in case we stumble upon any issues for us to ask and take the references
- Considering the all the above mentioned yardsticks, our choice has become clearer to opt for
    a. AWS S3, GCP bucket for initial and post storage options
    b. Preprocessing and for major extract, transform tasks
        ■ Athena/Glue for initial processing of schema to understand the nature of data
        ■ pySpark, for cleaning and transforming the data
    c. HDFS/Hadoop EMR cluster, for big data storage
    d. Apache HIVE database, for storing, processing and analysing the dataset
    e. Apache HUE UI, for HIVE, S3, PIG interactions through relatively good user interface rather than overly relying on the CLI tools
    f. Elasticsearch Cloud/Kibana dashboard and integration tools for creating charts, visualisations and reports, hosted on GCP
    g. Python and linux commands to perform miscellaneous tasks

Related work – any related systems/application - Top 3

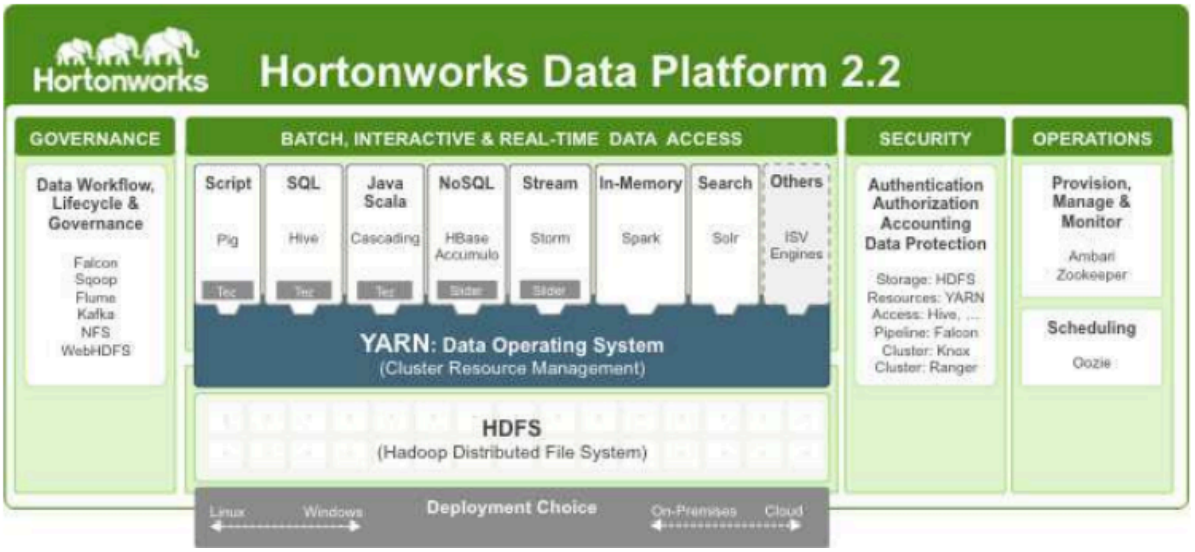
1. Started off with Yahoo architecture and how they have managed to process big data and their architecture around micro batch processing using Apache Kafka and Spark [2]



2. So we took some of the reference from the stackoverflow[1] architecture design where it leveraged the ELK stack sitting on top of monolithic design, we helped them tackle their core issues of storage with eventual consistency and faster search/analytics through ELK (elasticsearch, logstash and kibana) stack.[1] [link]



3. Hortonworks selection of big data architecture, their approach for HDFS interactions [3]



How does our approach differ?

Above mentioned applications are designed to handle scalability with production ready environments, while we had to design the data flow architecture considering the cloud service which are available at our fingertips with minimal to no cost, so with that in mind.

Data -> AWS S3 -> Athena -> Glue -> EMR -> HIVE and Pyspark -> S3 -> Elastic Cloud service (Elasticsearch/Kibana)

With this data flow, we were able to achieve the ultimate goal of creating a good data analytics application with future scope of creating data ingestion pipelines, python endpoints/lambda triggers and handling big data processing with good horizontal scalability.

Description of the dataset

a. Source of data

Music Brainz dataset, https://musicbrainz.org/doc/MusicBrainz_Database

b. Process of extraction/collection
Download the musicbrainz data and upload it to S3 using AWS CLI

```
# to move the files to s3, TODO : use python boto3
aws s3 cp "C:\Users\patel\Downloads\artists.csv" s3://music-5bb01e20
```

c. Data preprocessing – preparation and cleaning
For the preprocessing and preparation, we have selected **AWS Athena** queries after creating the database, table and load the s3 file to the table

```
CREATE EXTERNAL TABLE IF NOT EXISTS
`music_brains`.`music_artists_all` (
  `mbid` string,
  `artist_mb` string,
  `artist_lastfm` string,
  `country_mb` string,
  `country_lastfm` string,
  `tags_mb` string,
  `tags_lastfm` string,
  `listeners_lastfm` string,
  `scrobbles_lastfm` string,
  `ambiguous_artist` string
)
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
WITH SERDEPROPERTIES (
  'separatorChar' = '\t',
  'quoteChar' = '\'',
  'escapeChar' = '\\'
)
STORED AS INPUTFORMAT 'org.apache.hadoop.mapred.TextInputFormat'
OUTPUTFORMAT
'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION 's3://music-951f64e0/csv/'
TBLPROPERTIES ('classification' = 'csv');

SELECT * FROM "music_brains"."music_artists_all" limit 10;
SELECT count(*) FROM "music_brains"."music_artists_all";
SELECT count(*) FROM "music_brains"."music_artists_all" where
ambiguous_artist = 'TRUE';
```

Description of data processing

In two phases,

Phase 1, using Glue to convert it to parquet format, create HIVE tables and fetch the relevant data then export it to S3 [link to gitlab glue file]

```
Create database music;

CREATE EXTERNAL TABLE IF NOT EXISTS `music`.`artists` (
`mbid` string,
`artist_mb` string,
`artist_lastfm` string,
  `country_mb` string,
  `country_lastfm` string,
  `tags_mb` string,
  `tags_lastfm` string,
  `listeners_lastfm` string,
  `scrobbles_lastfm` string,
  `ambiguous_artist` string
)
STORED AS PARQUET
LOCATION 's3://music-d6cb4270/parquet';

--## country/artists count

select * from artists where mb_id in (
select mb_ib, artist_lastfm, count(*) from artists
where ambiguous_artist = 'FALSE'
group by mb_ib, artist_lastfm
having count(*) > 10) grouped;

--## cleaned

select * from artists where
ambiguous_artist is not null
and country_lastfm is not null
and country_mb is not null
and artist_mb is not null
and artist_lastfm is not null
and tags_mb is not null
and tags_lastfm is not null
and mbid != 'mbid';

--#### most listeners

Select
artist_lastfm artist_name,
country_mb country_origin,
split(country_lastfm,';') country_list,
```

```
cast(listeners_lastfm as int) listeners_cnt,
cast(scrobbles_lastfm as int) scrobbles_cnt,
split(tags_mb,';') tags
from artists
where mbid != 'mbid'
and country_lastfm != ''
and artist_mb != ''
order by listeners_cnt desc

hive -e 'Select
artist_lastfm artist_name,
country_mb country_origin,
split(country_lastfm,';') country_list,
cast(listeners_lastfm as int) listeners_cnt,
cast(scrobbles_lastfm as int) scrobbles_cnt,
split(tags_mb,';') tags
from artists
where mbid != 'mbid'
and country_lastfm != ''
and artist_mb != ''
order by listeners_cnt desc' | sed 's/[\t]/,/g'  > /final.csv
```

Phase 2, pyspark, using google collab and perform RDD, SparkSQL operations then make the final file ready for the consumption, this step is just to eliminate all possibilities to surface the refined data, as we have observed pyspark has more support and an excellent developer community than Apache PIG [pyspark notebook]

Development of the application platform

Cloud platform we have used to achieve this step is GCP and that helped us integrate with Elastic cloud service

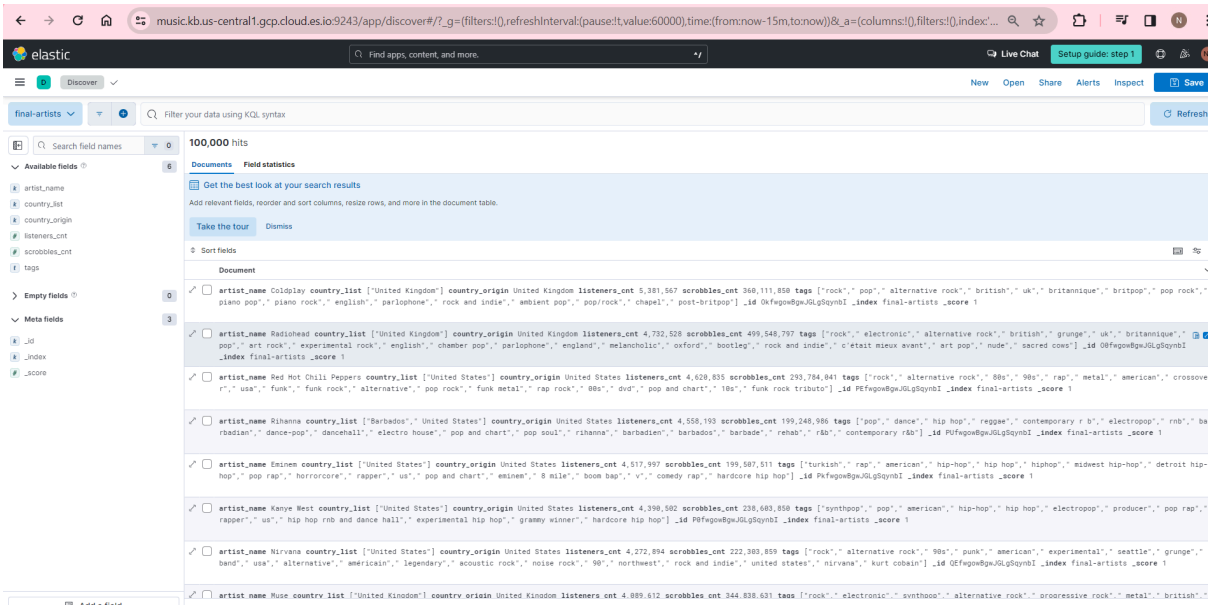
Elastic cloud comes with its integrations agent and GCP dataflow pipelines, however dataflow template pipelines are in beta phase and fails randomly at times, so we have leveraged Elastic cloud data integrations to connect our S3 files and create index using following mappings

```
{
    "known_artists": {
      "mappings": {
        "properties": {
```

```
          "ambiguous_artist": {
            "type": "keyword"
          },
          "artist_lastfm": {
            "type": "keyword"
          },
          "artist_mb": {
            "type": "keyword"
          },
          "country_lastfm": {
            "type": "keyword"
          },
          "country_mb": {
            "type": "keyword"
          },
          "listeners_lastfm": {
            "type": "long"
          },
          "mbid": {
            "type": "keyword"
          },
          "scrobbles_lastfm": {
            "type": "long"
          },
          "tags_lastfm": {
            "type": "text"
          },
          "tags_mb": {
            "type": "text"
          }
        }
      }
    }
  }
#### cluster setting, replica/master nodes
#### text fields are considered analysed by default for performing
### NLP operations

{
    "known_artists": {
      "settings": {
        "index": {
          "routing": {
```

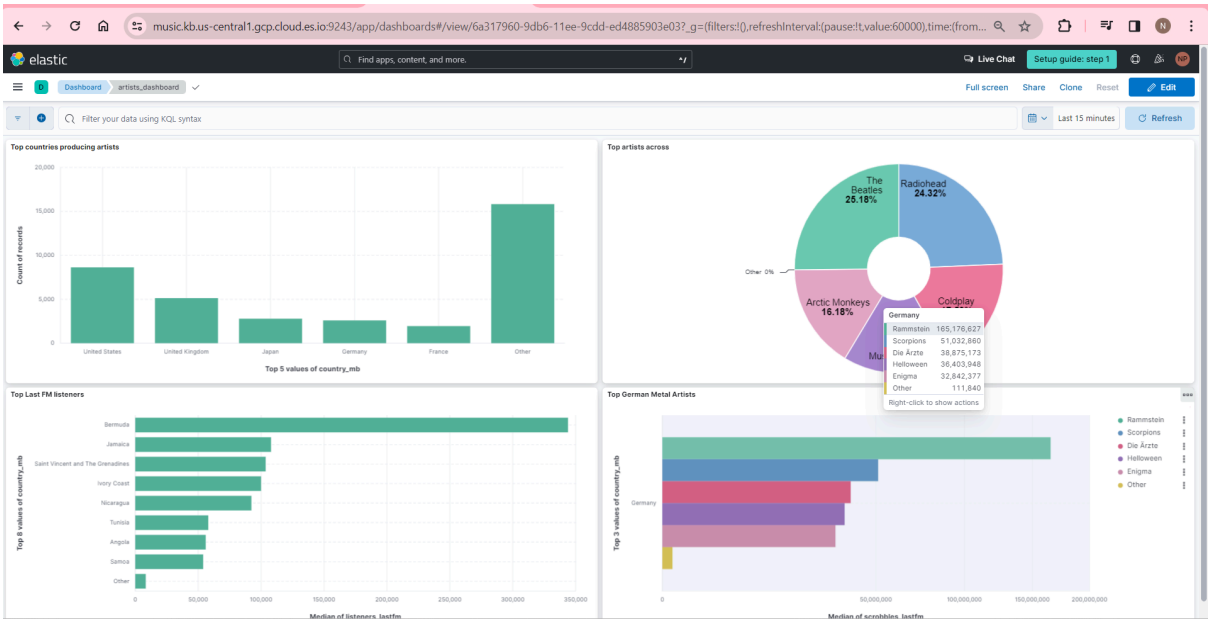
```
            "allocation": {
              "include": {
                "_tier_preference": "data_content"
              }
            }
          },
          "number_of_shards": "1",
          "provided_name": "known_artists",
          "creation_date": "1702911573820",
          "number_of_replicas": "1",
          "uuid": "G7ni20CIRmidn8pnY14Q2A",
          "version": {
            "created": "8500003"
          }
        }
      }
    }
  }
```


Once we successfully integrate the data the data flow would look like this,



Then we can go to the visualization, and start creating the charts to answer the top artists, top artists producing countries and the top german music artists based on their listeners last.fm numbers

Final visualisation looks like this; and this helps us answer who is the most famous artist in Germany is Rammstein!



Challenges and lessons learned

Challenge 1: Understanding the schema before the process

So we have understood that, schema detection before the process phase is crucial, that's where we had to use the Athena which internally uses the HIVE and helped us figure out the schema, Glue crawler was also useful for schema detection

Challenge 2 : CSV is not big data friendly format, Parquet to the rescue

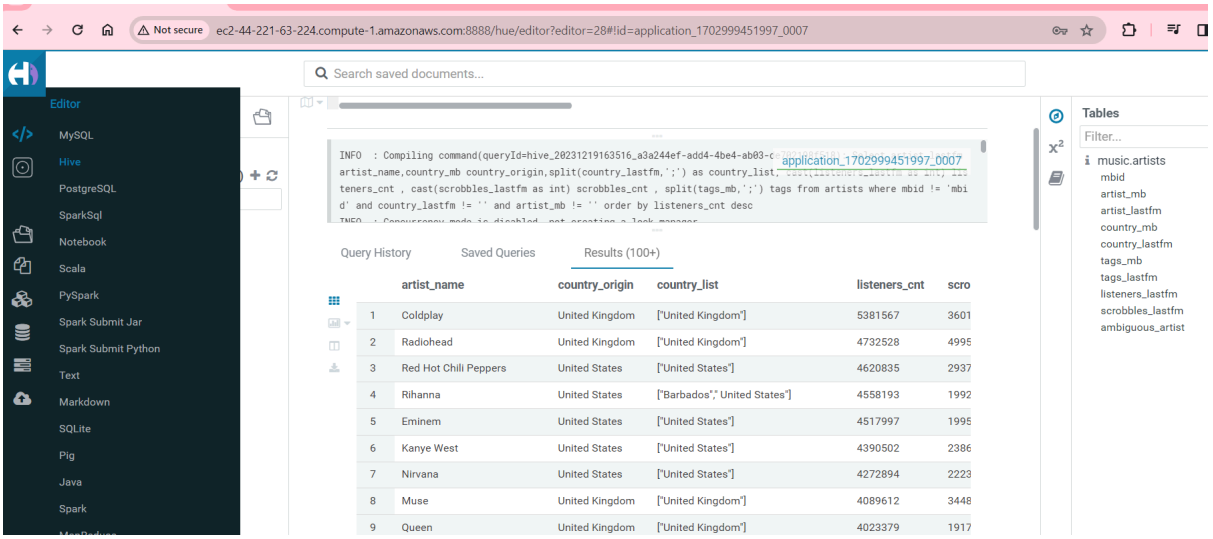
Another challenge, we faced was performing all the operations in CSV, first it consumed space, time and the due to no information on the de-limilitor, we had to perform trial and error just to understand the delimiter with escaping sequence to load, then we understood instead of loading in CSV, parquet and avro seems to be better formats dealing with big data especially in apache tools and technology and that helped us load the data in HIVE easily

Challenge 3 : GCP dataflow template are still in the beta phase

GCP dataflow has defined templates, however to ingest the data into elastic cloud the template was in beta phase and fails randomly, so as fallback options we had to spin the elastic cloud integration agents to get through the ingestion of AWS S3 files to elasticsearch index, however AWS is again the sandbox environment which made things worse in the sync process.

Challenge 4 : EMR hadoop operations are hard to perform on CLI, Apache Hue to our rescue

Since hive, hdfs files, pig and spark streaming, it was a lot to go back and forth so we had the need for the user friendly UI to perform our tasks, stores the queries, visualise the operations and maintain the state, so we had performed SSH tunnelling using proxyswitch omega chrome extension and that ease our burden and helped us immensely in performing processing, hadoop operations.



Our take : Elasticsearch dashboard is very powerful

Elasticseach we understood for the search engineer built on lucene, however the Kibana dashboard seems to be very smooth in fetch the categorical data and intuitive enough to suggest the user which charts and analysis to go for;
Plus, its integration supports AWS, GCP, AZURE . It seems to be a worthy inclusion for all the log, data and NLP operations and high end analytics.

Responsibility section (which group member did what?)

Name	Task	Notes
Vishnu Priya Gandewar	• Dataset cleaning • Preprocessing with Athena • PySpark operations • Schema creations • Glue operations to transform to parquet	
Nirav Patel	• EMR creations • Hive Queries • SSH HUE tunnelling • Elasticseach cloud setup, GCP integrations, dataflow testing	

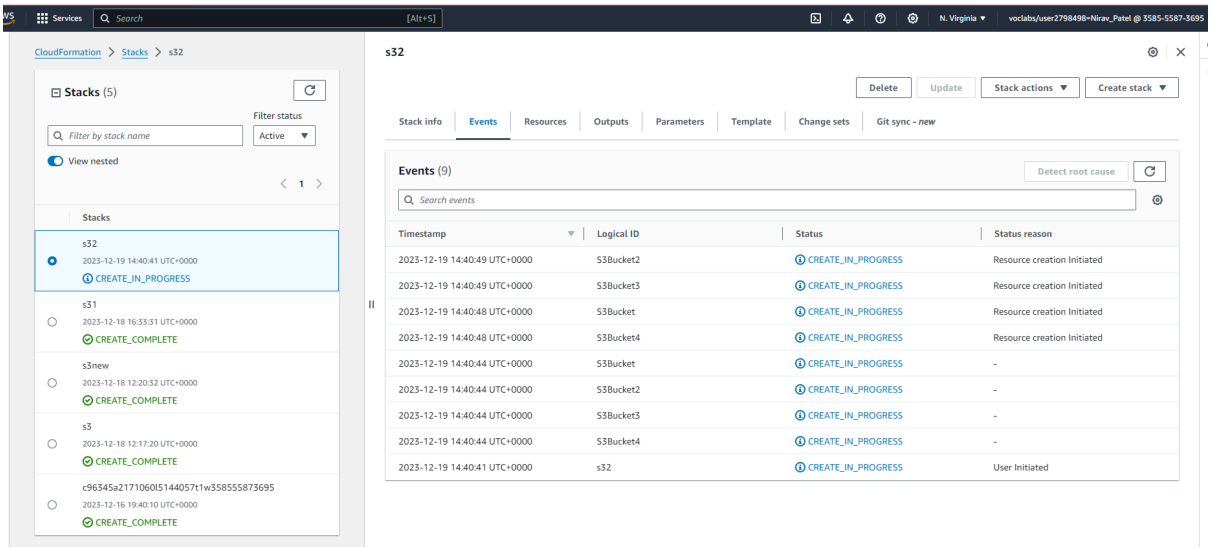
	• Index, replica, cluster settings and dashboard creations • S3 integrations and GCP bigquery, dataflow analysis	

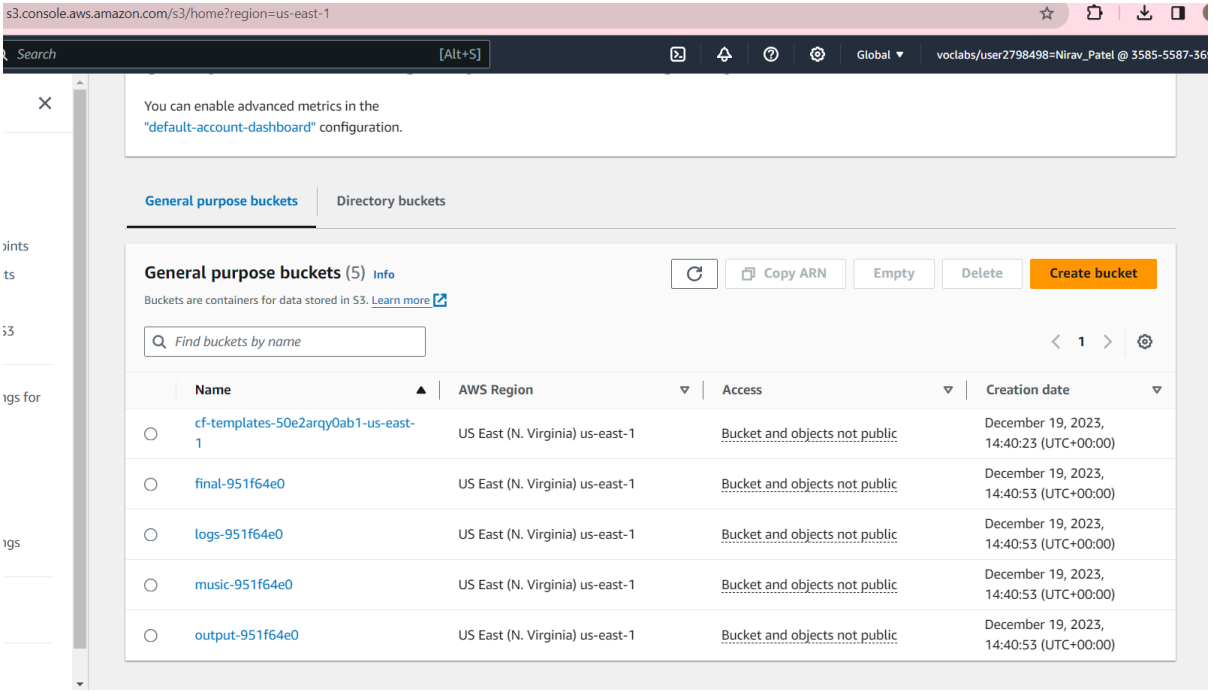
References

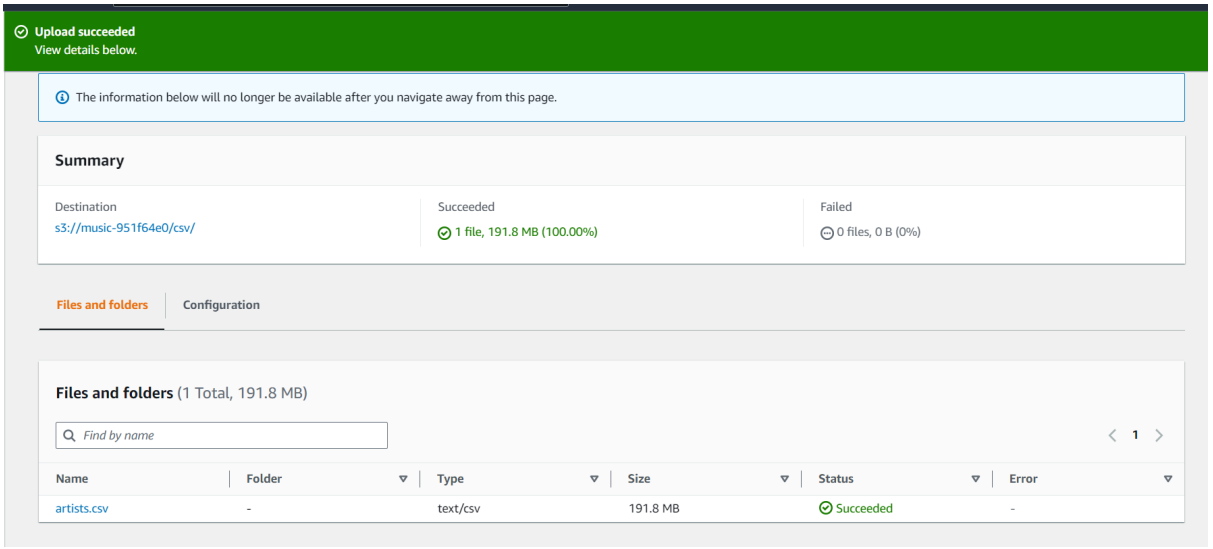
1. Theresa (2022) *EP27: Stack overflow architecture. also...*, *EP27: Stack Overflow Architecture. Also... - by Theresa*. Available at: https://blog.bytebytego.com/p/ep27-stack-overflow-architecture (Accessed: 17 December 2023).
2. Yahoo *et al.* (no date) *Yahoo Engineering*. Available at: https://yahooeng.tumblr.com/page/6 (Accessed: 17 December 2023).
3. *Big Data Reference Architecture - Cloudera*. Available at: https://hortonworks.com/wp-content/uploads/2013/10/Big-Data-Reference-Architecture_4AA5-6136ENW.pdf (Accessed: 17 December 2023).

Screenshots

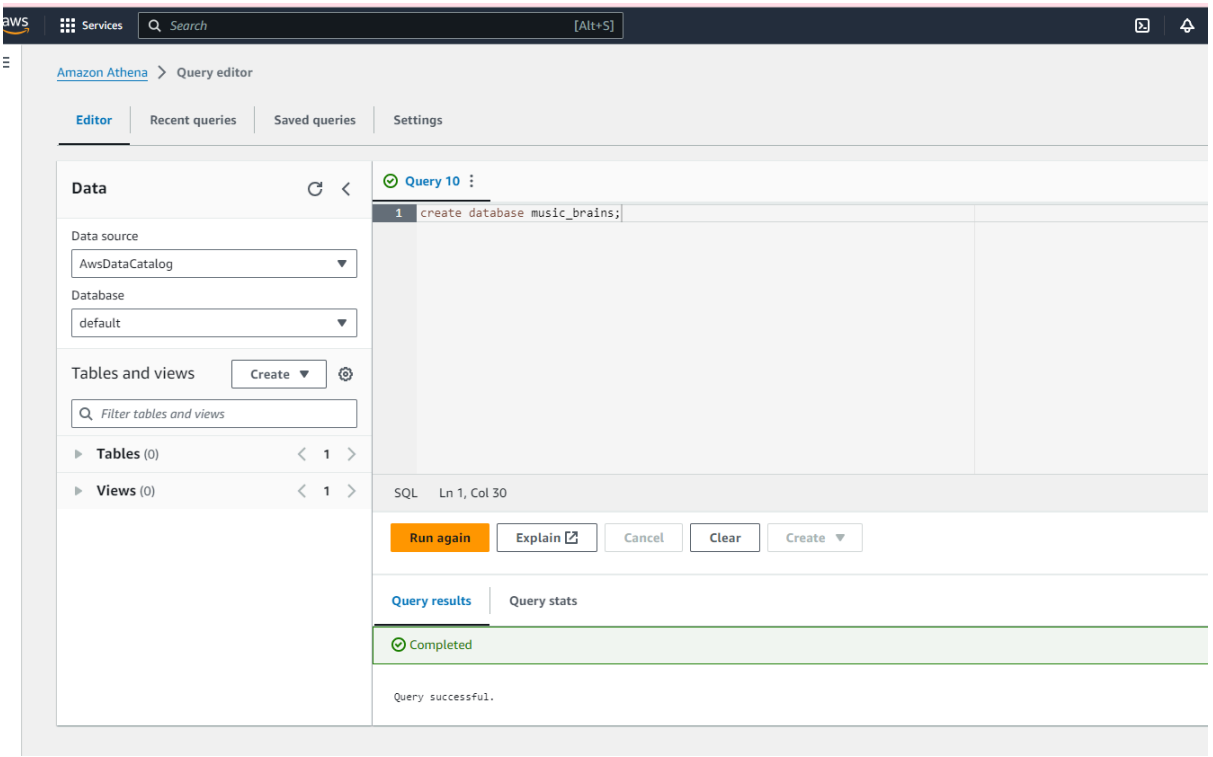
Creating AWS S3 cloudformation

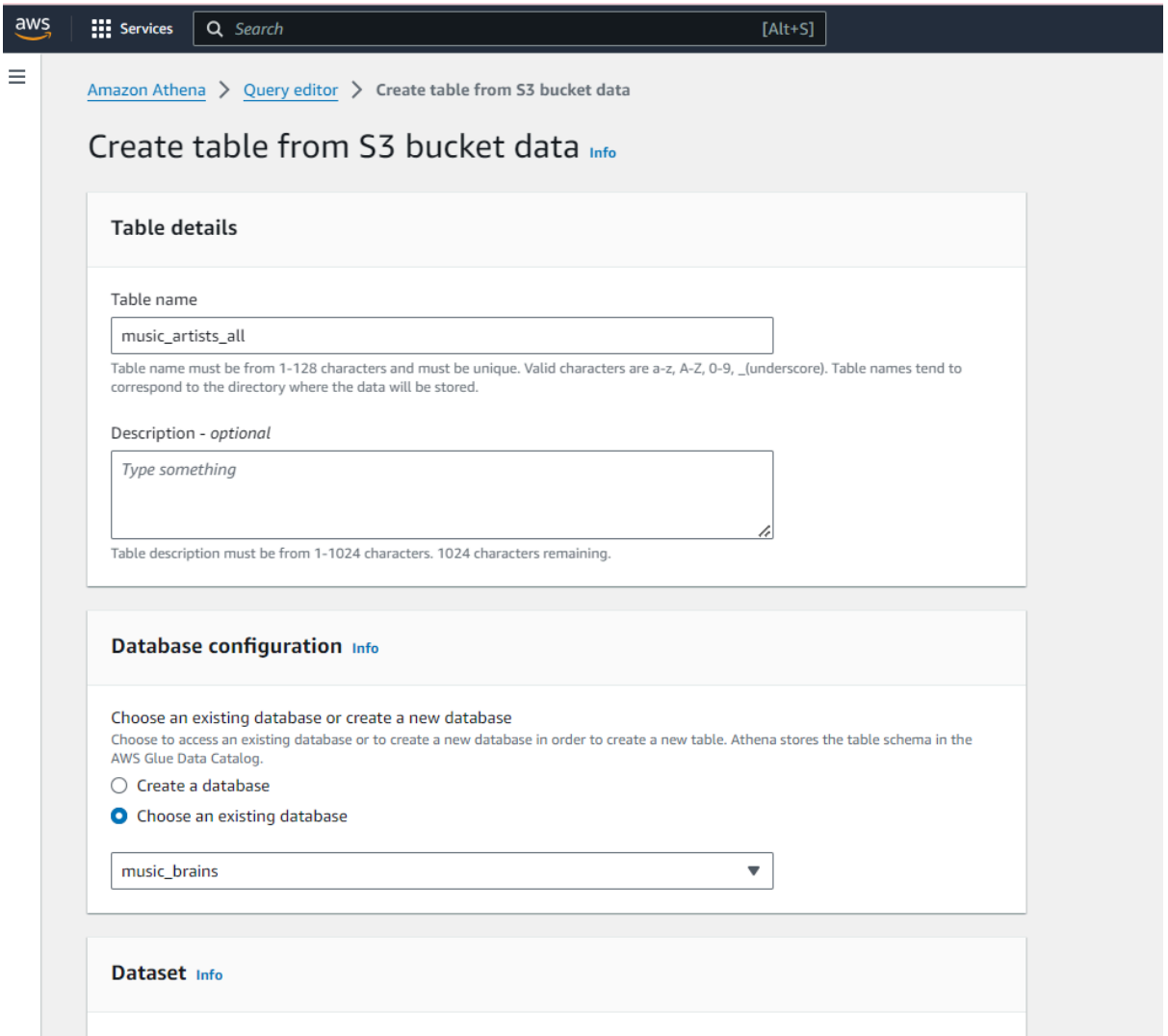


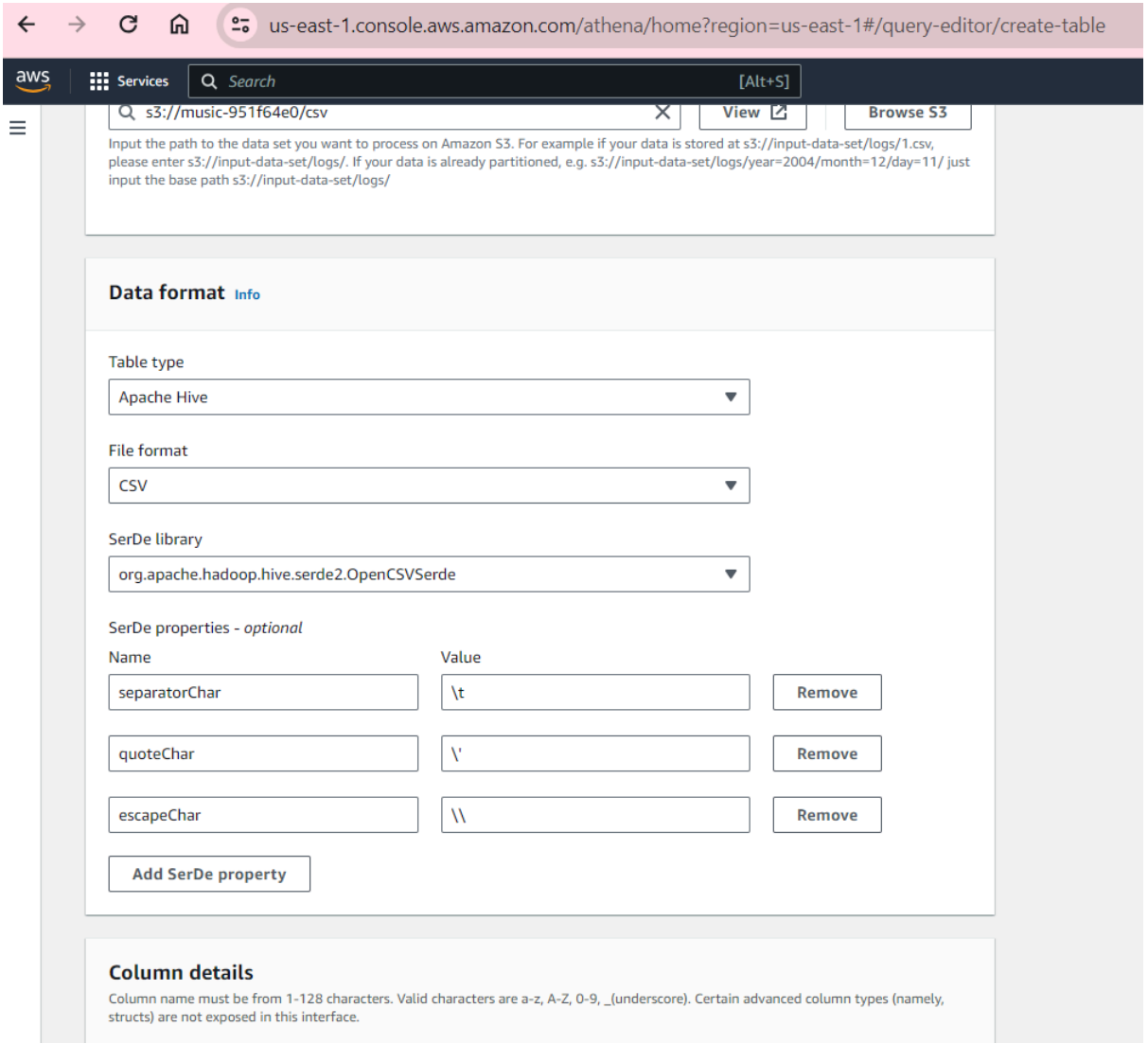


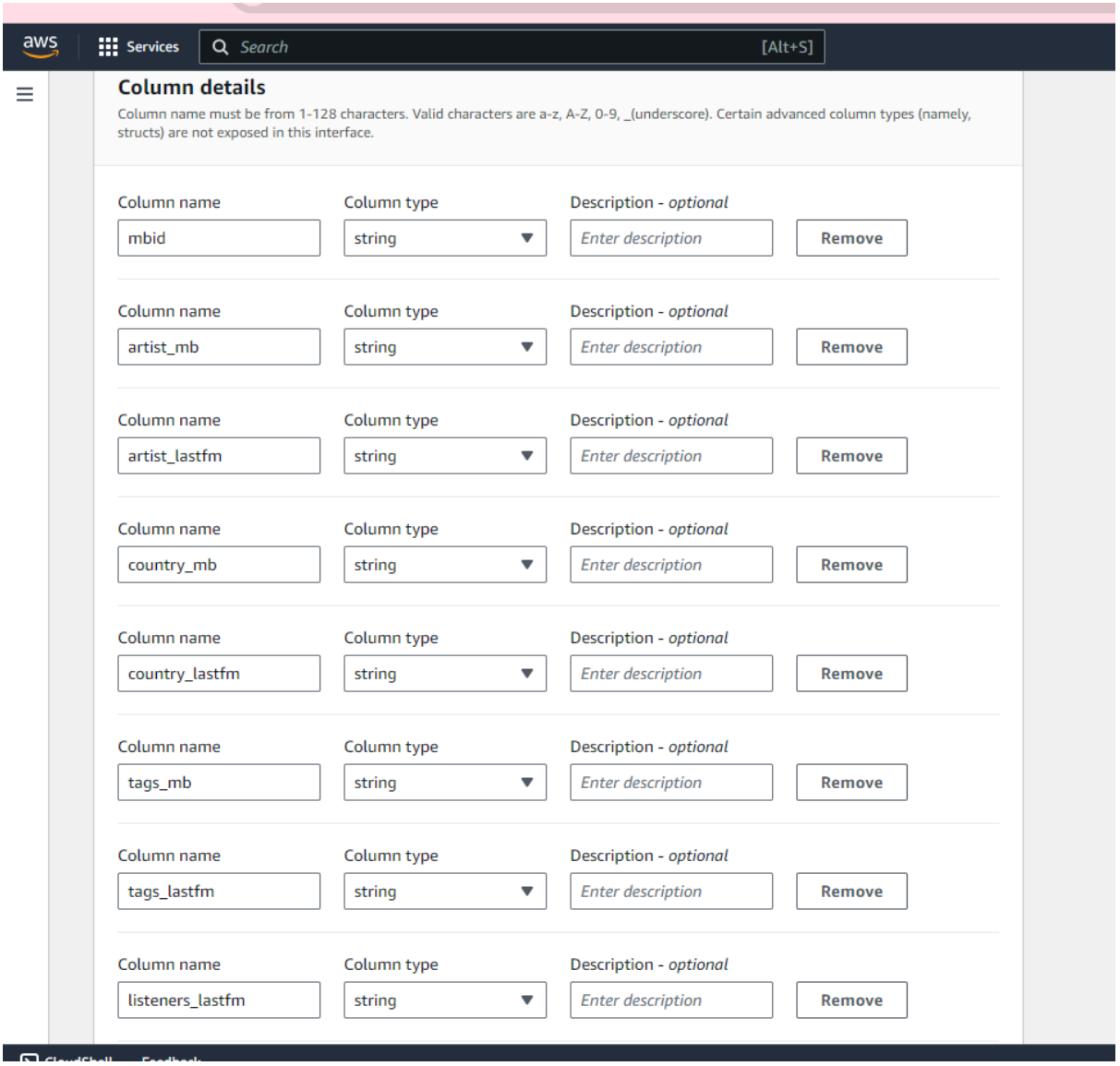


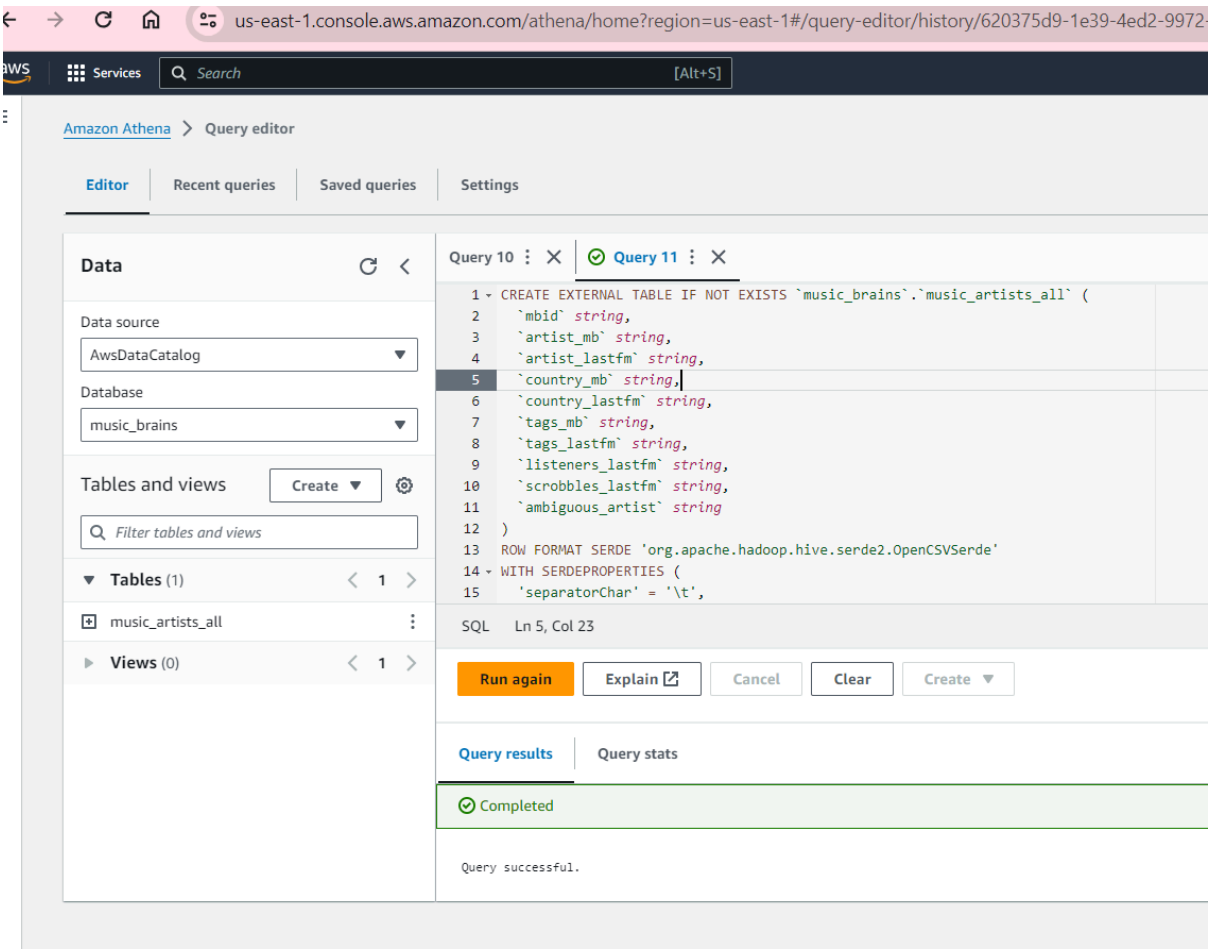
Athena operations

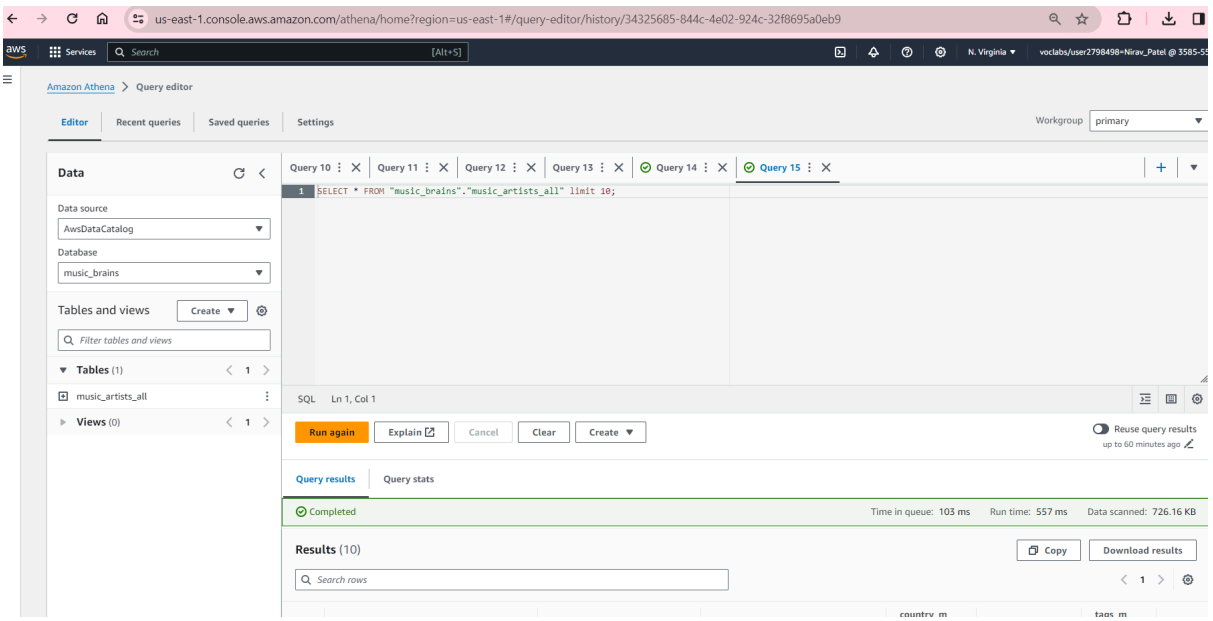

aws
Services
Search
[Alt+S]
Amazon Athena
Query editor
Editor
Recent queries
Saved queries
Settings
Data
Data source
AwsDataCatalog
Database
default
Tables and views
Create
Filter tables and views
Tables (0)
Views (0)
Query 10
1 create database music_brains;
SQL Ln 1, Col 30
Run again
Explain
Cancel
Clear
Create
Query results
Query stats
Completed
Query successful.

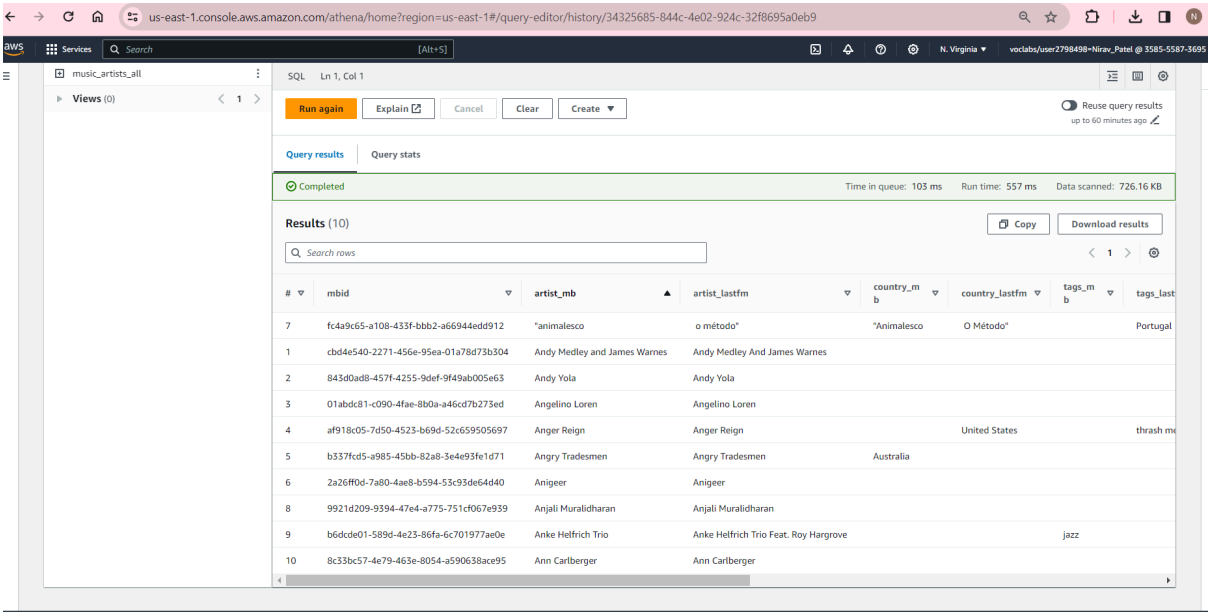

aws
Services
Search
[Alt+S]
Amazon Athena
Query editor
Create table from S3 bucket data
Create table from S3 bucket data Info
Table details
Table name
music_artists_all
Table name must be from 1-128 characters and must be unique. Valid characters are a-z, A-Z, 0-9, _(underscore). Table names tend to correspond to the directory where the data will be stored.
Description - optional
Type something
Table description must be from 1-1024 characters. 1024 characters remaining.
Database configuration Info
Choose an existing database or create a new database
Choose to access an existing database or to create a new database in order to create a new table. Athena stores the table schema in the AWS Glue Data Catalog.
Create a database
Choose an existing database
music_brains
Dataset Info

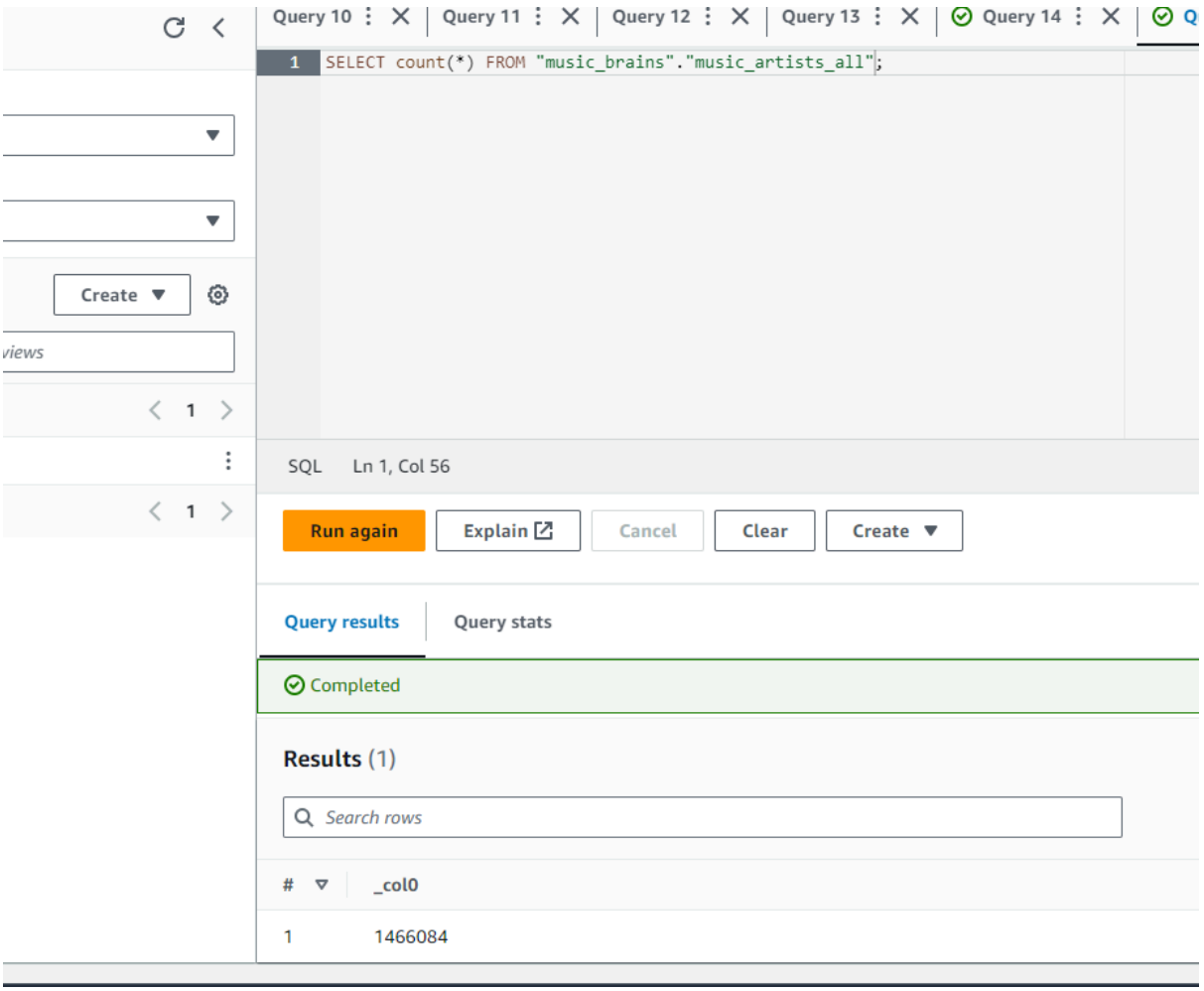

us-east-1.console.aws.amazon.com/athena/home?region=us-east-1#/query-editor/create-table
aws
Services
Search
[Alt+S]
s3://music-951f64e0/csv
View
Browse S3
Input the path to the data set you want to process on Amazon S3. For example if your data is stored at s3://input-data-set/logs/1.csv, please enter s3://input-data-set/logs/. If your data is already partitioned, e.g. s3://input-data-set/logs/year=2004/month=12/day=11/ just input the base path s3://input-data-set/logs/
Data format
Info
Table type
Apache Hive
File format
CSV
SerDe library
org.apache.hadoop.hive.serde2.OpenCSVSerde
SerDe properties - optional
Name
Value
separatorChar
\t
Remove
quoteChar
\'
Remove
escapeChar
\\
Remove
Add SerDe property
Column details
Column name must be from 1-128 characters. Valid characters are a-z, A-Z, 0-9, _(underscore). Certain advanced column types (namely, structs) are not exposed in this interface.

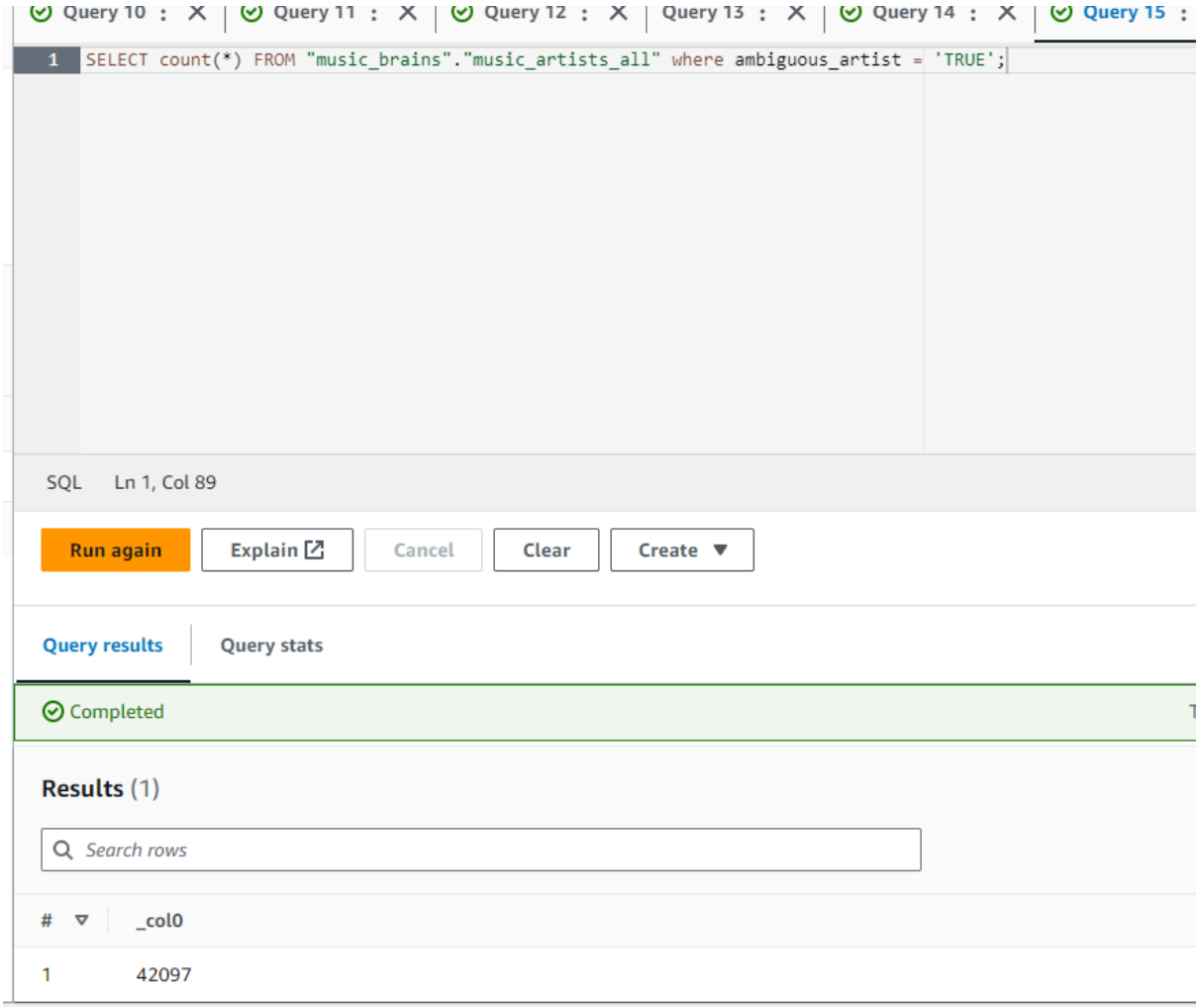

aws
Services
Search
[Alt+S]
Column details
Column name must be from 1-128 characters. Valid characters are a-z, A-Z, 0-9, _(underscore). Certain advanced column types (namely, structs) are not exposed in this interface.
Column name
Column type
Description - optional
mbid
string
Enter description
Remove
artist_mb
artist_lastfm
country_mb
country_lastfm
tags_mb
tags_lastfm
listeners_lastfm


us-east-1.console.aws.amazon.com/athena/home?region=us-east-1#/query-editor/history/620375d9-1e39-4ed2-9972
Services
Search
[Alt+S]
Amazon Athena
Query editor
Editor
Recent queries
Saved queries
Settings
Data
Data source
AwsDataCatalog
Database
music_brains
Tables and views
Create
Filter tables and views
Tables (1)
music_artists_all
Views (0)
Query 10
Query 11
CREATE EXTERNAL TABLE IF NOT EXISTS `music_brains`.`music_artists_all` (
`mbid` string,
`artist_mb` string,
`artist_lastfm` string,
`country_mb` string,
`country_lastfm` string,
`tags_mb` string,
`tags_lastfm` string,
`listeners_lastfm` string,
`scrobbles_lastfm` string,
`ambiguous_artist` string
)
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
WITH SERDEPROPERTIES (
'separatorChar' = '\t',
SQL
Ln 5, Col 23
Run again
Explain
Cancel
Clear
Create
Query results
Query stats
Completed
Query successful.

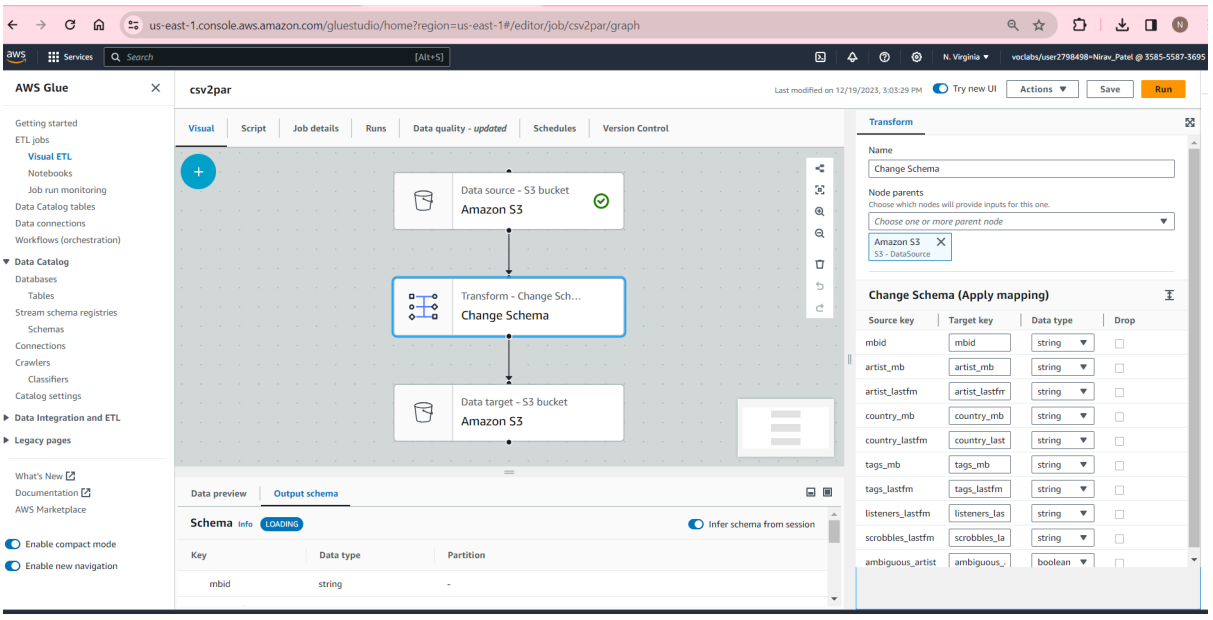

us-east-1.console.aws.amazon.com/athena/home?region=us-east-1#/query-editor/history/34325685-844c-4e02-924c-32f8695a0eb9
Amazon Athena
Query editor
Editor
Recent queries
Saved queries
Settings
Workgroup
primary
Data
Query 10
Query 11
Query 12
Query 13
Query 14
Query 15
SELECT * FROM "music_brains"."music_artists_all" limit 10;
SQL
Ln 1, Col 1
Run again
Explain
Cancel
Clear
Create
Reuse query results
up to 60 minutes ago
Query results
Query stats
Completed
Time in queue: 103 ms
Run time: 557 ms
Data scanned: 726.16 KB
Results (10)
Copy
Download results
Search rows

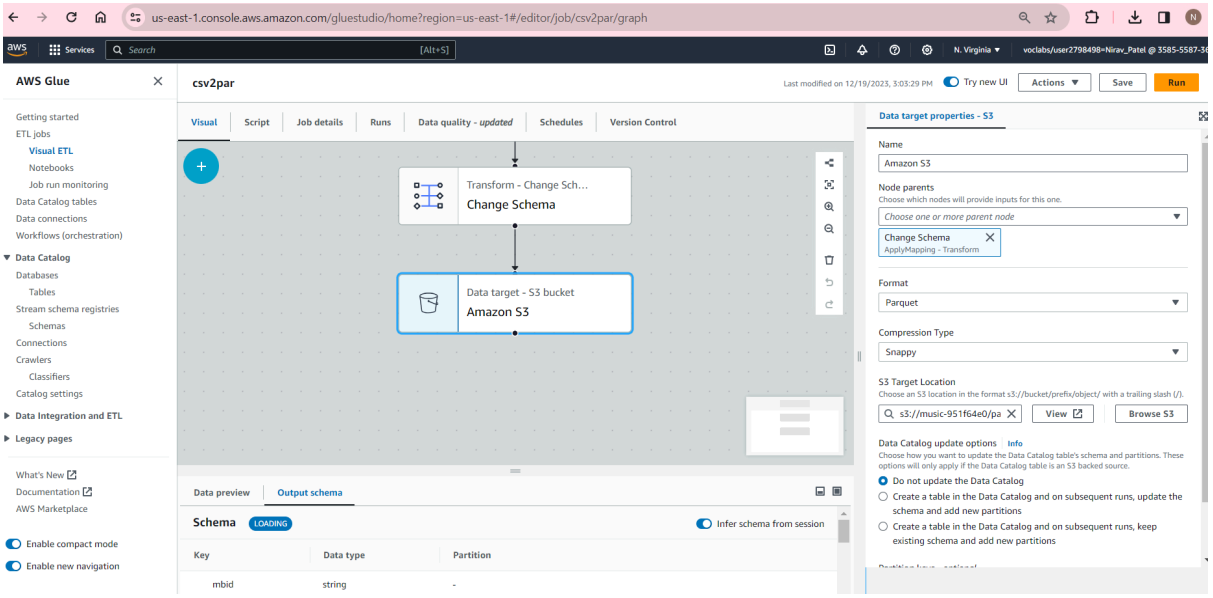

us-east-1.console.aws.amazon.com/athena/home?region=us-east-1#/query-editor/history/34325685-844c-4e02-924c-32f8695a0eb9
music_artists_all
Views (0)
SQL Ln 1, Col 1
Run again
Explain
Cancel
Clear
Create
Reuse query results
up to 60 minutes ago
Query results
Query stats
Completed
Time in queue: 103 ms
Run time: 557 ms
Data scanned: 726.16 KB
Results (10)
Copy
Download results
Search rows
mbid artist_mb artist_lastfm country_mb country_lastfm tags_mb tags_last
7 fc4a9c65-a108-433f-bbb2-a66944edd912 "animalesco o método" "Animalesco O Método" Portugal
1 cbd4e540-2271-456e-95ea-01a78d73b304 Andy Medley and James Warnes Andy Medley And James Warnes
2 843d0ad8-457f-4255-9def-9f49ab005e63 Andy Yola Andy Yola
3 01abdc81-c090-4fae-8b0a-a46cd7b273ed Angelino Loren Angelino Loren
4 af918c05-7d50-4523-b69d-52c659505697 Anger Reign Anger Reign United States thrash me
5 b337fcd5-a985-45bb-82a8-3e4e93fe1d71 Angry Tradesmen Angry Tradesmen Australia
6 2a26ff0d-7a80-4ae8-b594-53c93de64d40 Anigeer Anigeer
8 9921d209-9394-47e4-a775-751cf067e939 Anjali Muralidharan Anjali Muralidharan
9 b6dcde01-589d-4e23-86fa-6c701977ae0e Anke Helfrich Trio Anke Helfrich Trio Feat. Roy Hargrove jazz
10 8c33bc57-4e79-463e-8054-a590638ace95 Ann Carlberger Ann Carlberger

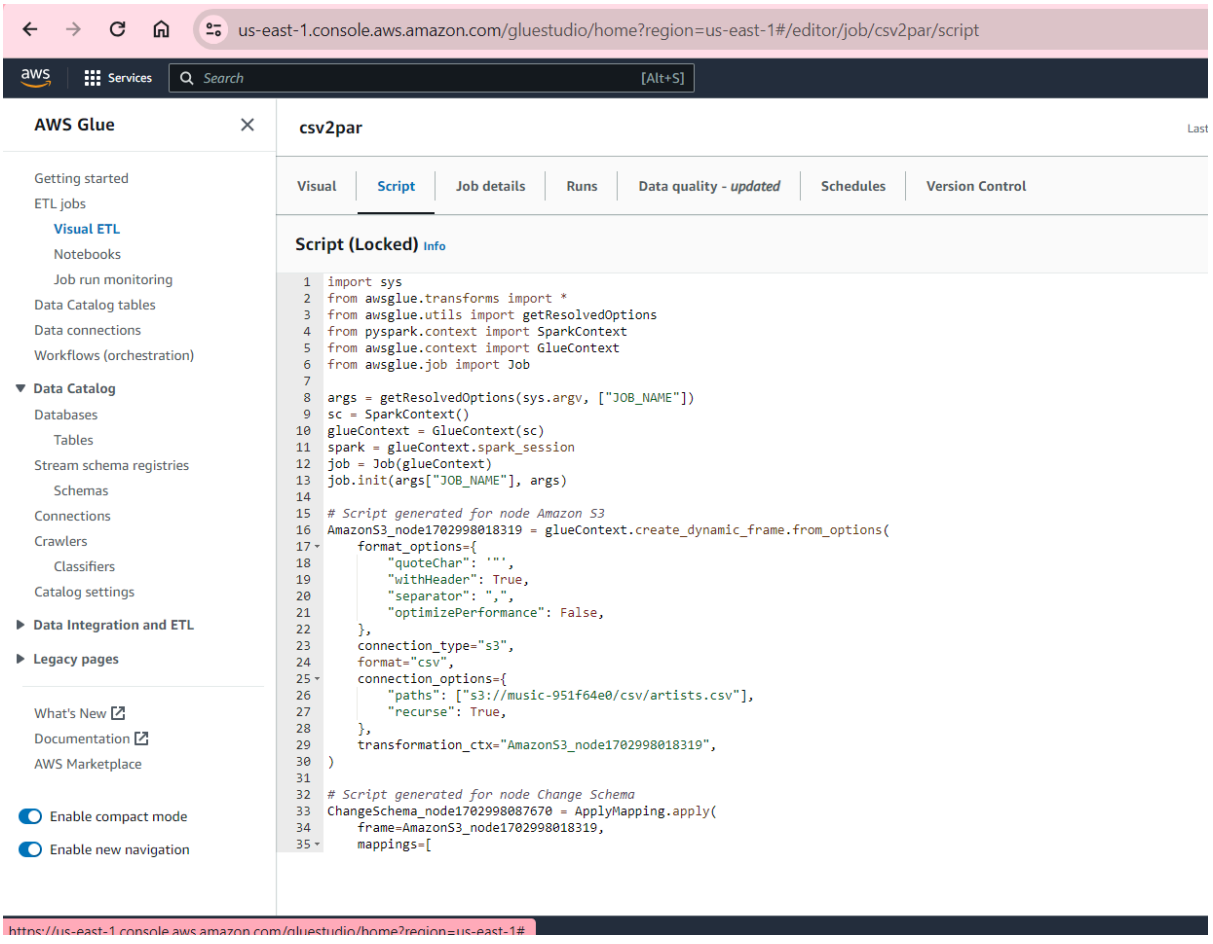

Query 10
Query 11
Query 12
Query 13
Query 14
1 SELECT count(*) FROM "music_brains"."music_artists_all";
SQL Ln 1, Col 56
Run again
Explain
Cancel
Clear
Create
Query results
Query stats
Completed
Results (1)
Search rows
_col0
1 1466084



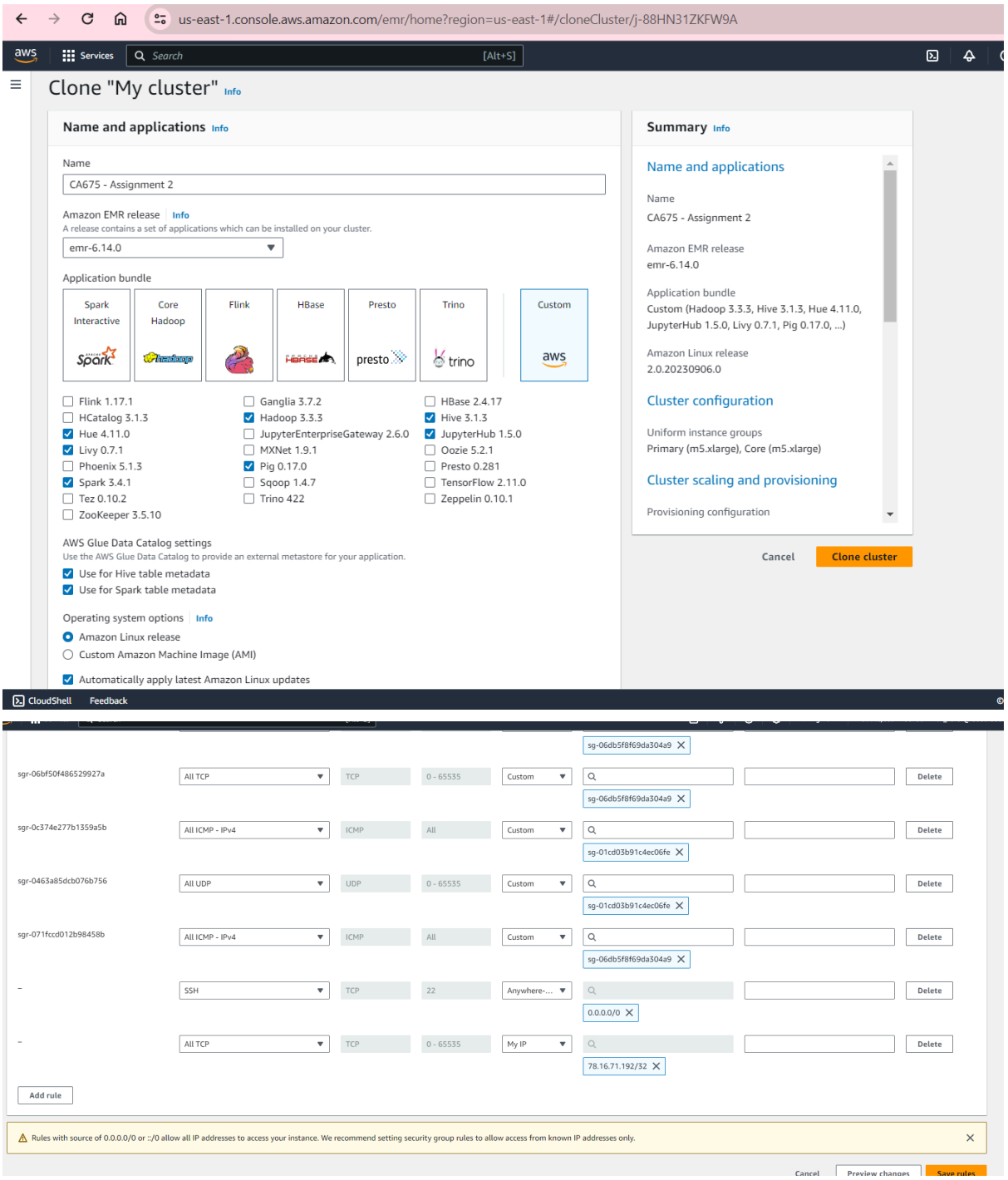
Glue ETL for CSV to Parquet

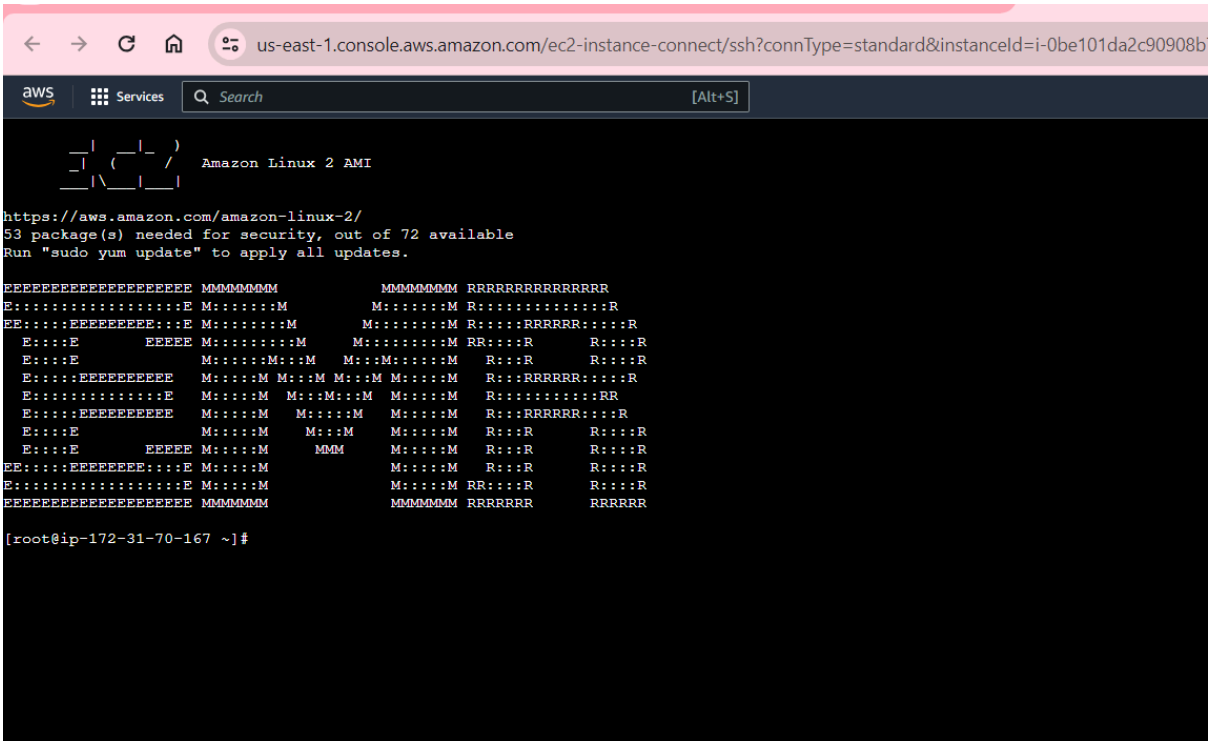



us-east-1.console.aws.amazon.com/gluestudio/home?region=us-east-1#/editor/job/csv2par/graph
AWS Glue
csv2par
Visual
Script
Job details
Runs
Data quality - updated
Schedules
Version Control
Actions
Save
Run
Transform - Change Sch...
Change Schema
Data target - S3 bucket
Amazon S3
Data target properties - S3
Name
Amazon S3
Node parents
Choose one or more parent node
Change Schema
ApplyMapping - Transform
Format
Parquet
Compression Type
Snappy
S3 Target Location
Browse S3
Data preview
Output schema
Schema
LOADING
Infer schema from session
Key
Data type
Partition
mbid
string

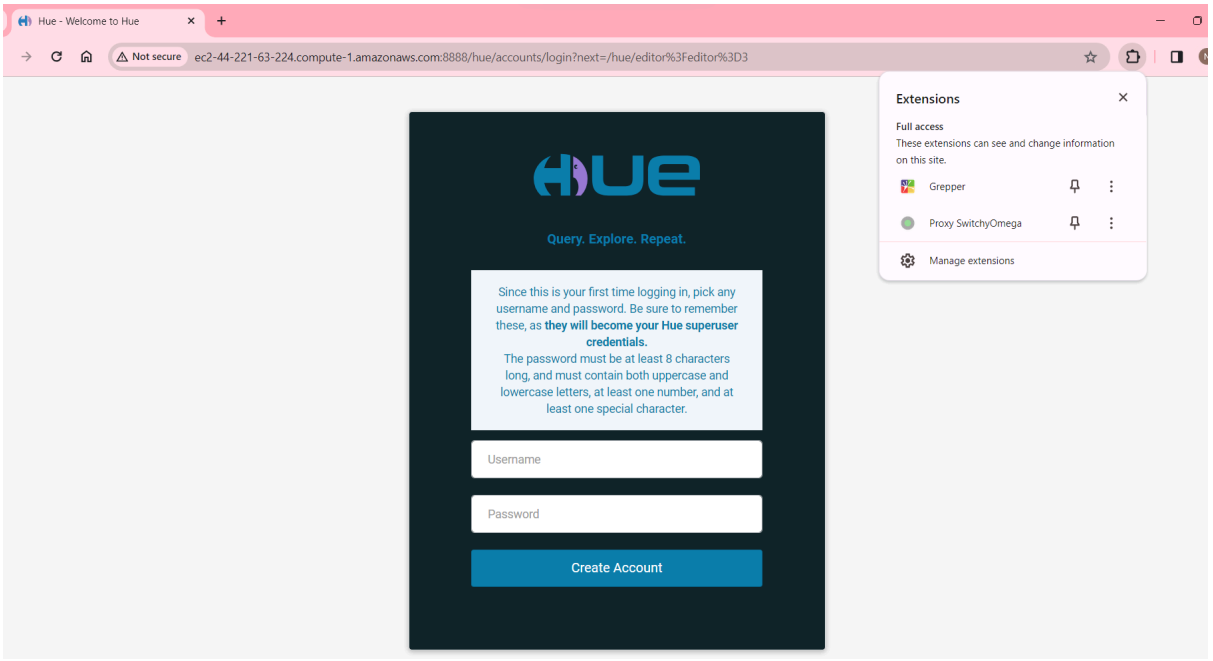

us-east-1.console.aws.amazon.com/gluestudio/home?region=us-east-1#/editor/job/csv2par/script
AWS Glue
csv2par
Visual
Script
Job details
Runs
Data quality - updated
Schedules
Version Control
Script (Locked) Info
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job
args = getResolvedOptions(sys.argv, ["JOB_NAME"])
sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)
job.init(args["JOB_NAME"], args)
# Script generated for node Amazon S3
AmazonS3_node1702998018319 = glueContext.create_dynamic_frame.from_options(
format_options={
"quoteChar": '"',
"withHeader": True,
"separator": ",",
"optimizePerformance": False,
},
connection_type="s3",
format="csv",
connection_options={
"paths": ["s3://music-951f64e0/csv/artists.csv"],
"recurse": True,
},
transformation_ctx="AmazonS3_node1702998018319",
)
# Script generated for node Change Schema
ChangeSchema_node1702998087670 = ApplyMapping.apply(
frame=AmazonS3_node1702998018319,
mappings=[

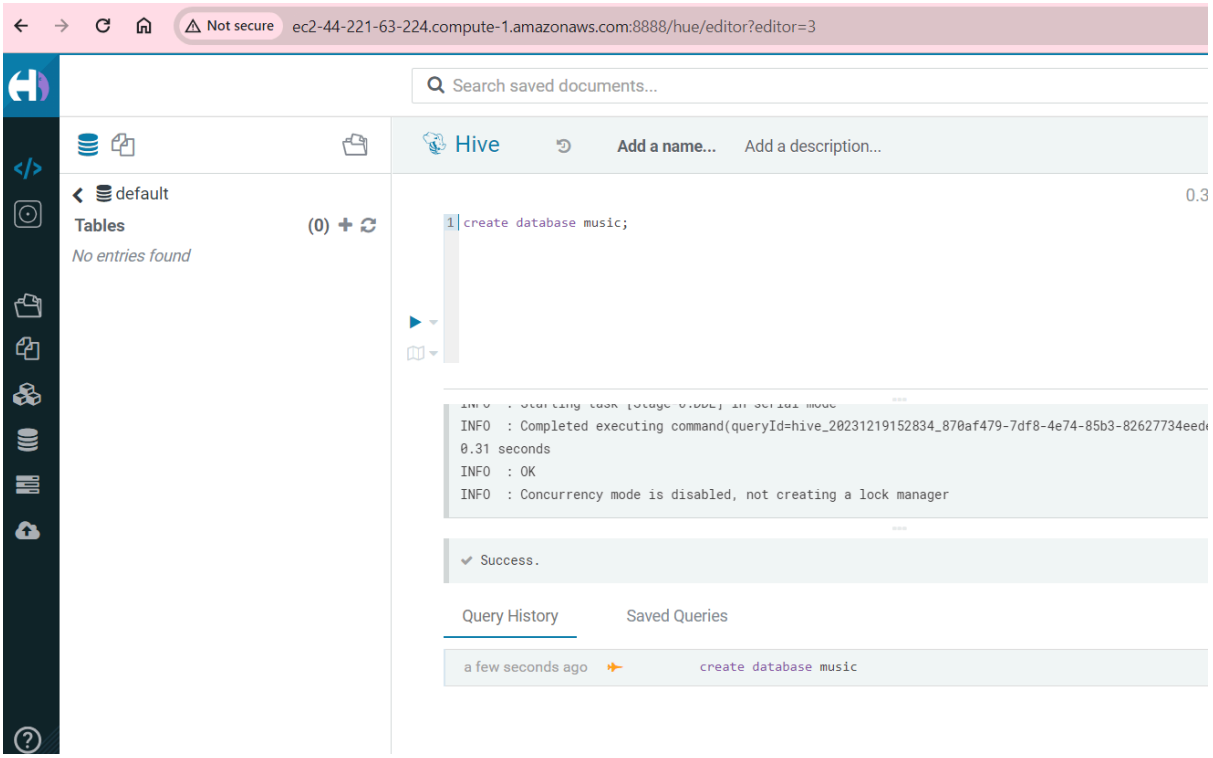
EMR Cluster configs and settings

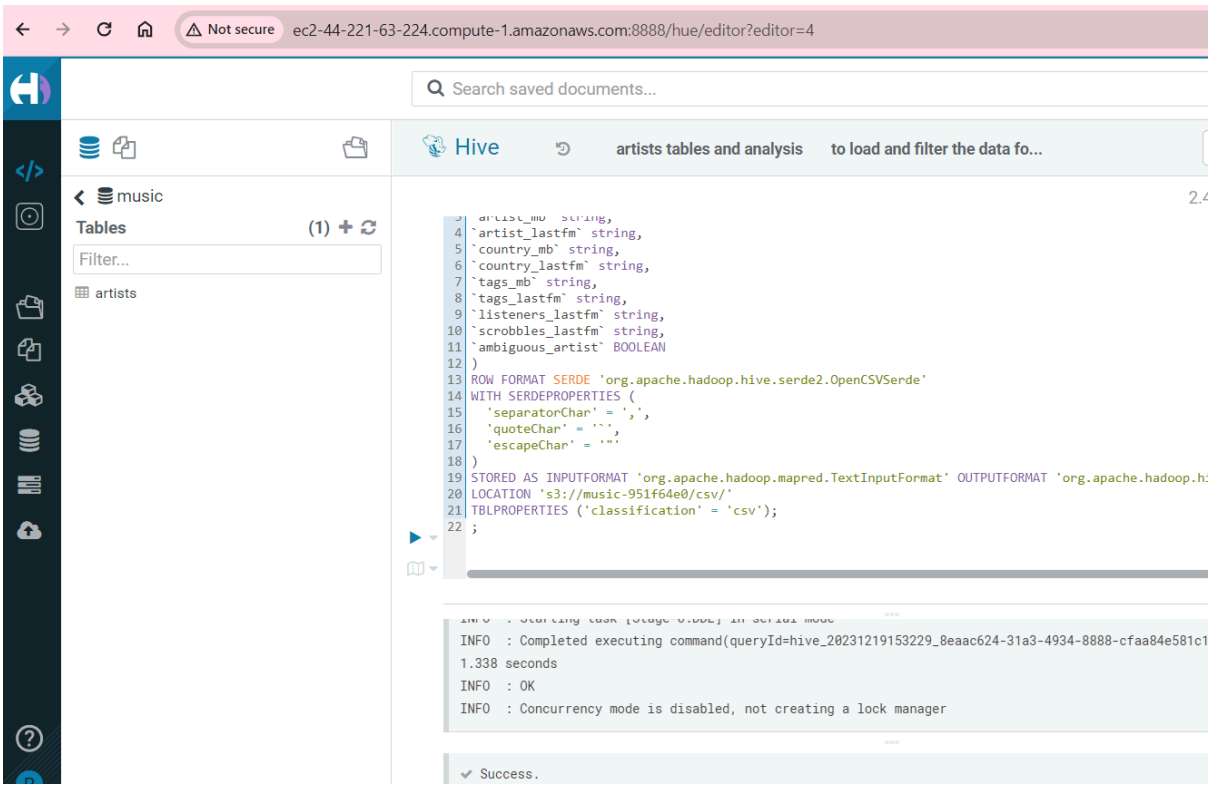


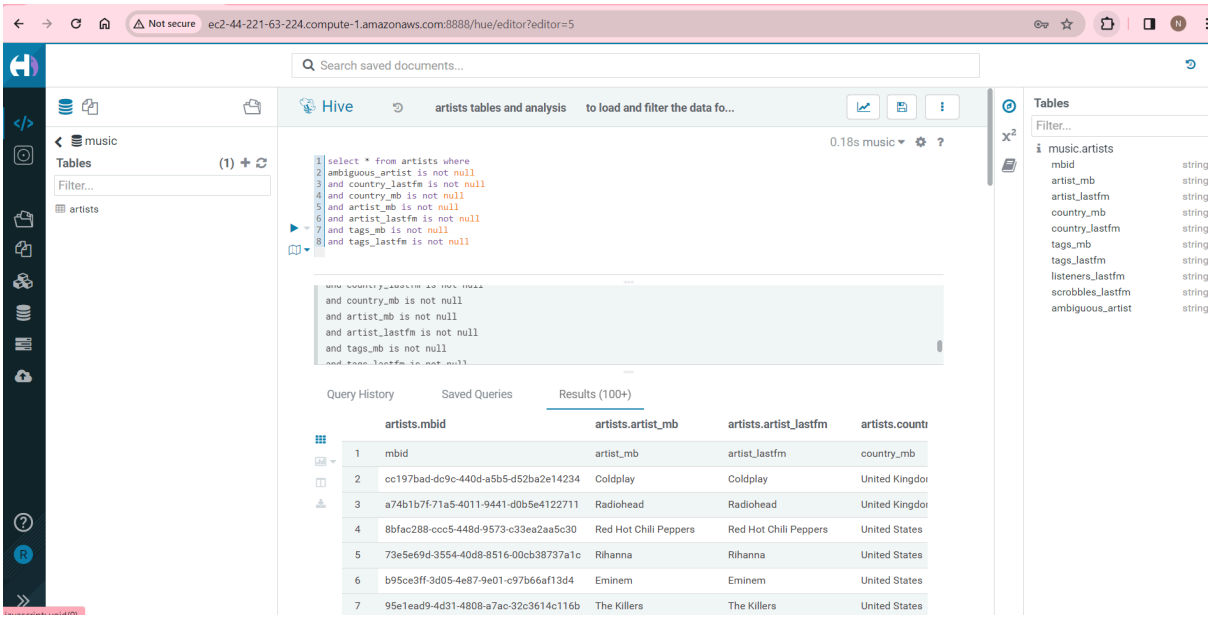


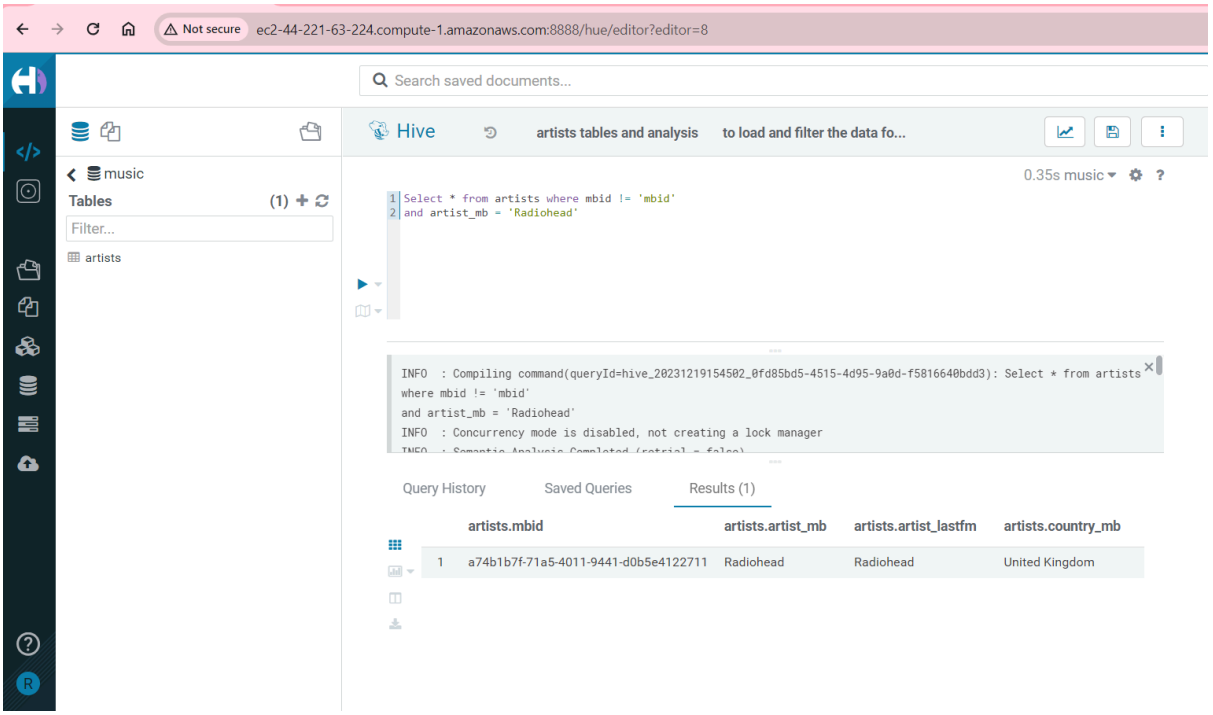
Apache Hue for Hadoop operations UI

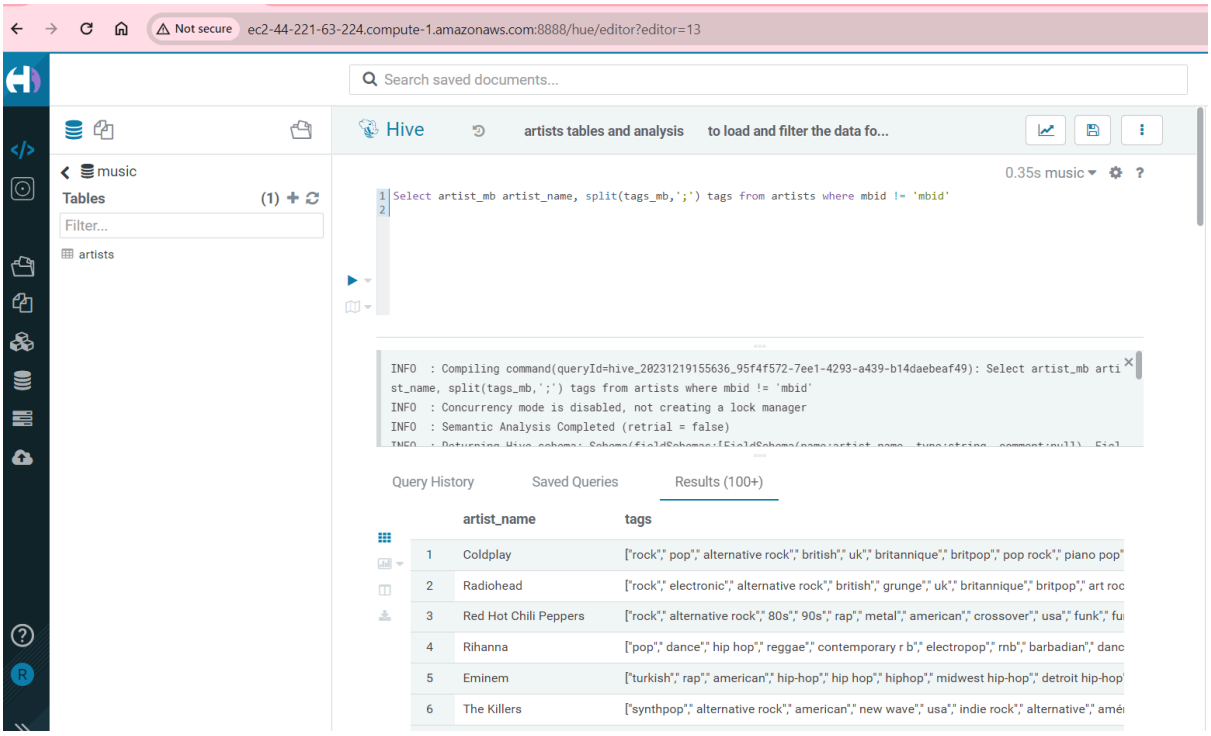


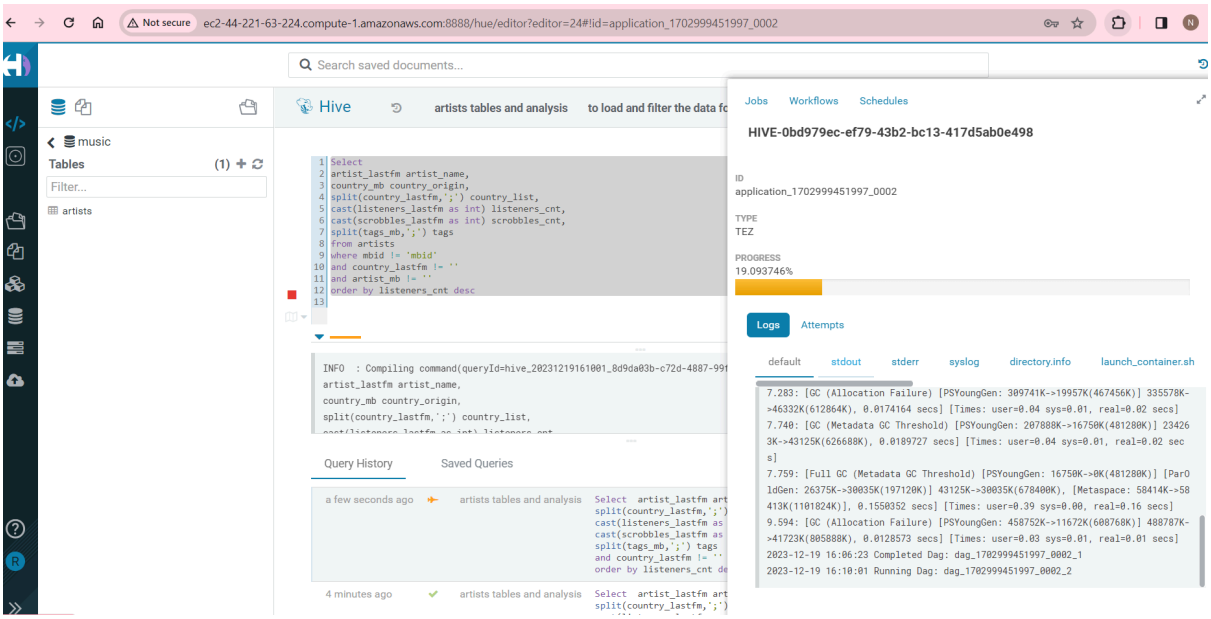

Not secure
ec2-44-221-63-224.compute-1.amazonaws.com:8888/hue/editor?editor=3
Search saved documents...
Hive
Add a name...
Add a description...
default
Tables (0)
No entries found
1 create database music;
INFO : Completed executing command(queryId=hive_20231219152834_870af479-7df8-4e74-85b3-82627734eed
0.31 seconds
INFO : OK
INFO : Concurrency mode is disabled, not creating a lock manager
Success.
Query History
Saved Queries
a few seconds ago
create database music


Not secure
ec2-44-221-63-224.compute-1.amazonaws.com:8888/hue/editor?editor=4
Search saved documents...
Hive
artists tables and analysis
to load and filter the data fo...
music
Tables (1)
Filter...
artists
`artist_lastfm` string,
`country_mb` string,
`country_lastfm` string,
`tags_mb` string,
`tags_lastfm` string,
`listeners_lastfm` string,
`scrobbles_lastfm` string,
`ambiguous_artist` BOOLEAN
)
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
WITH SERDEPROPERTIES (
'separatorChar' = ',',
'quoteChar' = '`',
'escapeChar' = '"'
)
STORED AS INPUTFORMAT 'org.apache.hadoop.mapred.TextInputFormat' OUTPUTFORMAT 'org.apache.hadoop.h
LOCATION 's3://music-951f64e0/csv/'
TBLPROPERTIES ('classification' = 'csv');
;
INFO : Completed executing command(queryId=hive_20231219153229_8eaac624-31a3-4934-8888-cfaa84e581c1
1.338 seconds
INFO : OK
INFO : Concurrency mode is disabled, not creating a lock manager
Success.

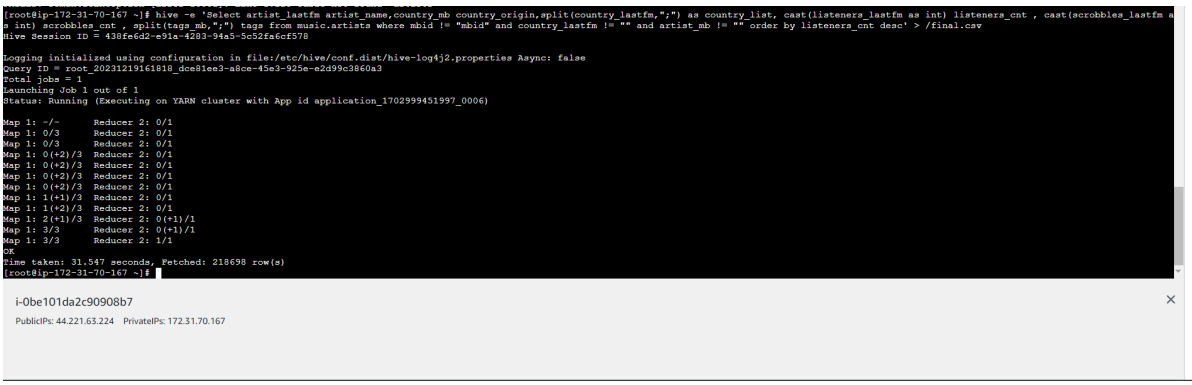

Not secure
ec2-44-221-63-224.compute-1.amazonaws.com:8888/hue/editor?editor=5
Search saved documents...
Hive
artists tables and analysis
to load and filter the data fo...
music
Tables (1)
Filter...
artists
0.18s music
select * from artists where
ambiguous_artist is not null
and country_lastfm is not null
and country_mb is not null
and artist_mb is not null
and artist_lastfm is not null
and tags_mb is not null
and tags_lastfm is not null
Query History
Saved Queries
Results (100+)
artists.mbid
artists.artist_mb
artists.artist_lastfm
artists.countr
1 mbid artist_mb artist_lastfm country_mb
2 cc197bad-dc9c-440d-a5b5-d52ba2e14234 Coldplay Coldplay United Kingdor
3 a74b1b7f-71a5-4011-9441-d0b5e4122711 Radiohead Radiohead United Kingdor
4 8bfac288-ccc5-448d-9573-c33ea2aa5c30 Red Hot Chili Peppers Red Hot Chili Peppers United States
5 73e5e69d-3554-40d8-8516-00cb38737a1c Rihanna Rihanna United States
6 b95ce3ff-3d05-4e87-9e01-c97b66af13d4 Eminem Eminem United States
7 95e1ead9-4d31-4808-a7ac-32c3614c116b The Killers The Killers United States
Tables
Filter...
music.artists
mbid string
artist_mb string
artist_lastfm string
country_mb string
country_lastfm string
tags_mb string
tags_lastfm string
listeners_lastfm string
scrobbles_lastfm string
ambiguous_artist string

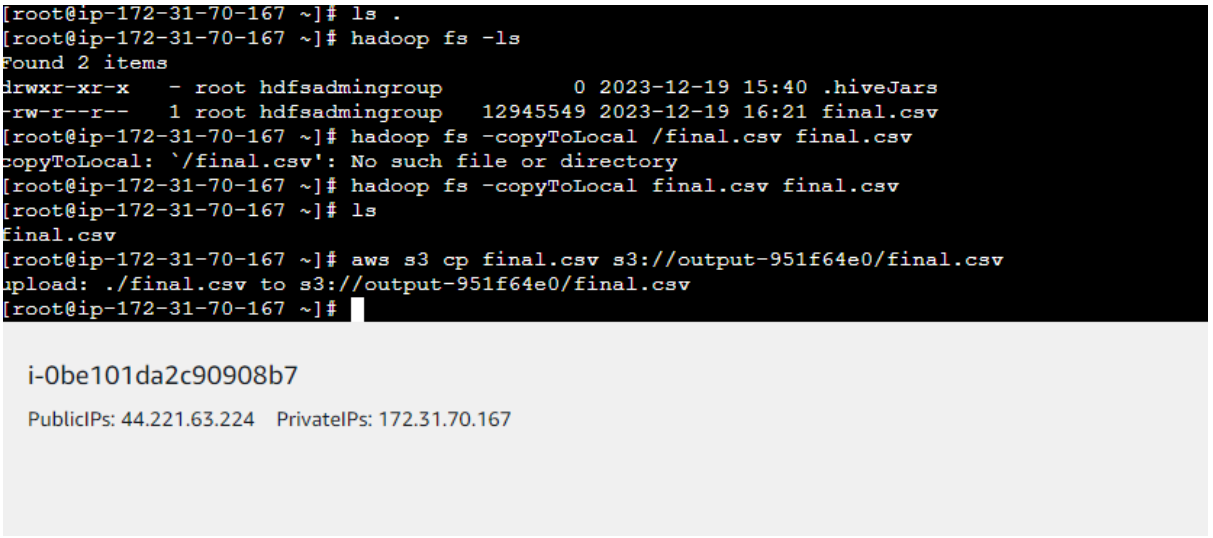

Not secure
ec2-44-221-63-224.compute-1.amazonaws.com:8888/hue/editor?editor=8
Search saved documents...
Hive
artists tables and analysis
to load and filter the data fo...
music
Tables (1)
Filter...
artists
0.35s music
Select * from artists where mbid != 'mbid'
and artist_mb = 'Radiohead'
INFO : Compiling command(queryId=hive_20231219154502_0fd85bd5-4515-4d95-9a0d-f5816640bdd3): Select * from artists
where mbid != 'mbid'
and artist_mb = 'Radiohead'
INFO : Concurrency mode is disabled, not creating a lock manager
Query History
Saved Queries
Results (1)
artists.mbid
artists.artist_mb
artists.artist_lastfm
artists.country_mb
1 a74b1b7f-71a5-4011-9441-d0b5e4122711 Radiohead Radiohead United Kingdom

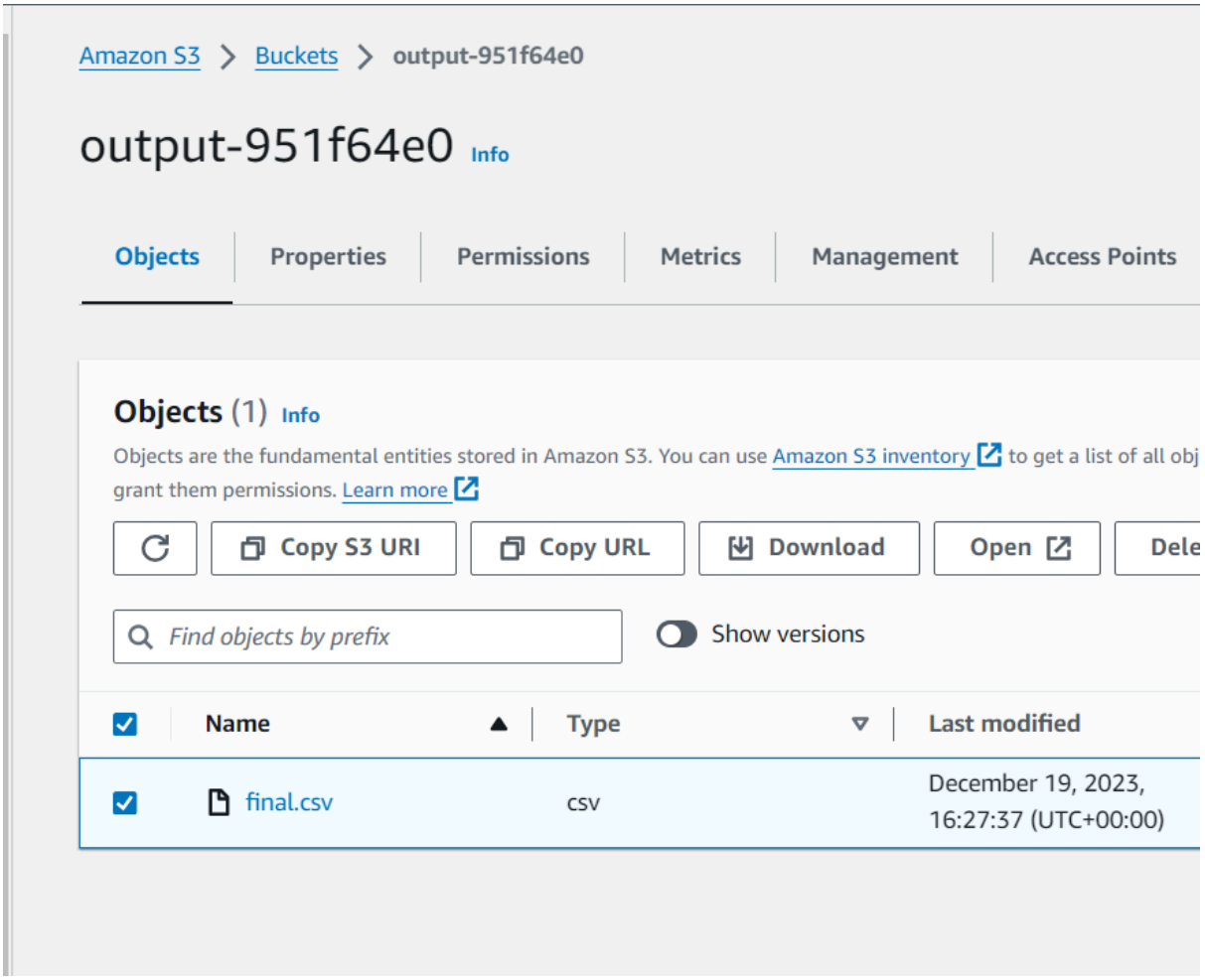




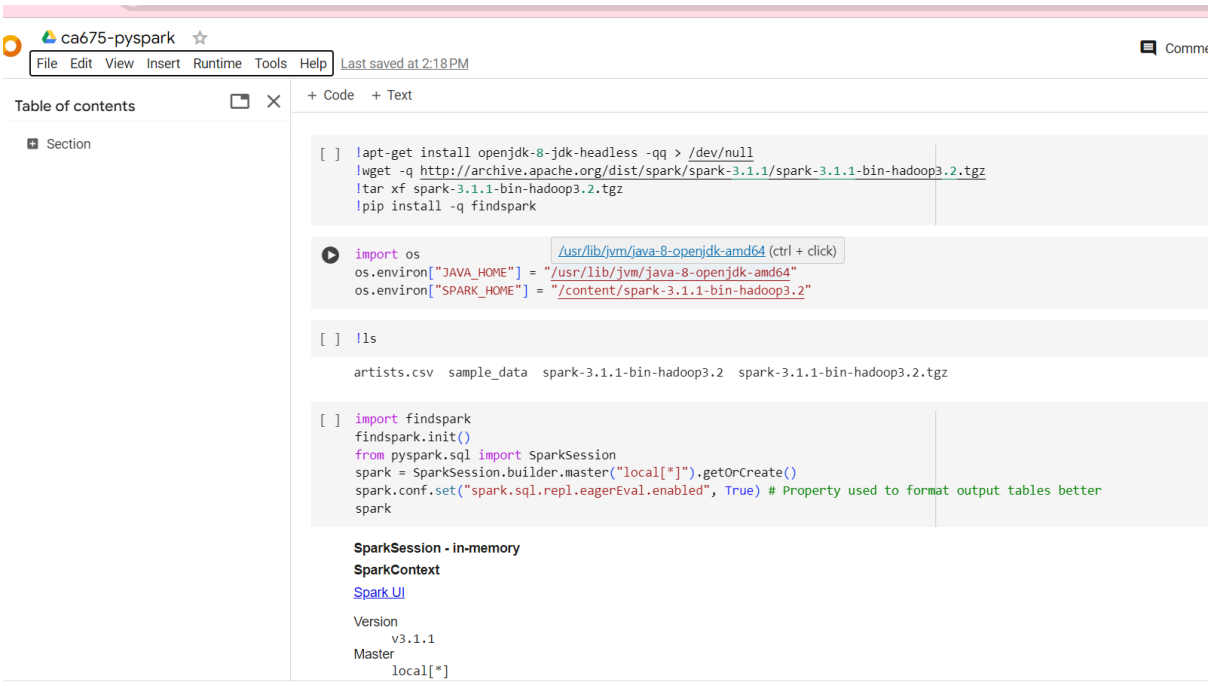
Export Final Report


[root@ip-172-31-70-167 ~]# hive -e 'Select artist_lastfm artist_name,country_mb country_origin,split(country_lastfm,";") as country_list, cast(listeners_lastfm as int) listeners_cnt , cast(scrobbles_lastfm as int) scrobbles_cnt , split(tags_mb,";") tags from music.artists where mbid != "mbid" and country_lastfm != "" and artist_mb != "" order by listeners_cnt desc' > /final.csv
Hive Session ID = 438fe6d2-e91a-4283-94a5-5c52fa6cf578
Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j2.properties Async: false
Query ID = root_20231219161818_dce81ee3-a8ce-45e3-925e-e2d99c3860a3
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1702999451997_0006)
Map 1: -/- Reducer 2: 0/1
Map 1: 0/3 Reducer 2: 0/1
Map 1: 0/3 Reducer 2: 0/1
Map 1: 0(+2)/3 Reducer 2: 0/1
Map 1: 0(+2)/3 Reducer 2: 0/1
Map 1: 0(+2)/3 Reducer 2: 0/1
Map 1: 0(+2)/3 Reducer 2: 0/1
Map 1: 1(+1)/3 Reducer 2: 0/1
Map 1: 1(+2)/3 Reducer 2: 0/1
Map 1: 2(+1)/3 Reducer 2: 0(+1)/1
Map 1: 3/3 Reducer 2: 0(+1)/1
Map 1: 3/3 Reducer 2: 1/1
OK
Time taken: 31.547 seconds, Fetched: 218698 row(s)
[root@ip-172-31-70-167 ~]#
i-0be101da2c90908b7
PublicIPs: 44.221.63.224 PrivateIPs: 172.31.70.167


[root@ip-172-31-70-167 ~]# ls .
[root@ip-172-31-70-167 ~]# hadoop fs -ls
Found 2 items
drwxr-xr-x   - root hdfsadmingroup   0 2023-12-19 15:40 .hiveJars
-rw-r--r--   1 root hdfsadmingroup   12945549 2023-12-19 16:21 final.csv
[root@ip-172-31-70-167 ~]# hadoop fs -copyToLocal /final.csv final.csv
copyToLocal: `/final.csv': No such file or directory
[root@ip-172-31-70-167 ~]# hadoop fs -copyToLocal final.csv final.csv
[root@ip-172-31-70-167 ~]# ls
final.csv
[root@ip-172-31-70-167 ~]# aws s3 cp final.csv s3://output-951f64e0/final.csv
upload: ./final.csv to s3://output-951f64e0/final.csv
[root@ip-172-31-70-167 ~]#
i-0be101da2c90908b7
PublicIPs: 44.221.63.224 PrivateIPs: 172.31.70.167



pyspark operations



+ Code + Text Connect

```
# header=True means the first row is a header
# sep=';' means the column are seperated using ''
df_artists = spark.read.csv('artists.csv', header=True, sep=",")
df_artists.show(5)
df_artists.count()
```

```
+--------------------+--------------------+--------------------+--------------+--------------------+--------------------+--------------------+---
|                mbid|           artist_mb|       artist_lastfm|    country_mb|      country_lastfm|             tags_mb|         tags_lastfm|lis
+--------------------+--------------------+--------------------+--------------+--------------------+--------------------+--------------------+---
|cc197bad-dc9c-440...|            Coldplay|            Coldplay|United Kingdom|      United Kingdom|rock; pop; altern...|rock; alternative...|
|a74b1b7f-71a5-401...|           Radiohead|           Radiohead|United Kingdom|      United Kingdom|rock; electronic;...|alternative; alte...|
|8bfac288-ccc5-448...|Red Hot Chili Pep...|Red Hot Chili Pep...| United States|       United States|rock; alternative...|rock; alternative...|
|73e5e69d-3554-40d...|             Rihanna|             Rihanna| United States|Barbados; United ...|pop; dance; hip h...|pop; rnb; female ...|
|b95ce3ff-3d05-4e8...|              Eminem|              Eminem| United States|       United States|turkish; rap; ame...|rap; Hip-Hop; Emi...|
+--------------------+--------------------+--------------------+--------------+--------------------+--------------------+--------------------+---
only showing top 5 rows

594907
```

+ Code + Text

```
[ ] #df_artists.columns;
    df_artists.printSchema()
```

```
root
 |-- mbid: string (nullable = true)
 |-- artist_mb: string (nullable = true)
 |-- artist_lastfm: string (nullable = true)
 |-- country_mb: string (nullable = true)
 |-- country_lastfm: string (nullable = true)
 |-- tags_mb: string (nullable = true)
 |-- tags_lastfm: string (nullable = true)
 |-- listeners_lastfm: string (nullable = true)
 |-- scrobbles_lastfm: string (nullable = true)
```

ENG 16:30

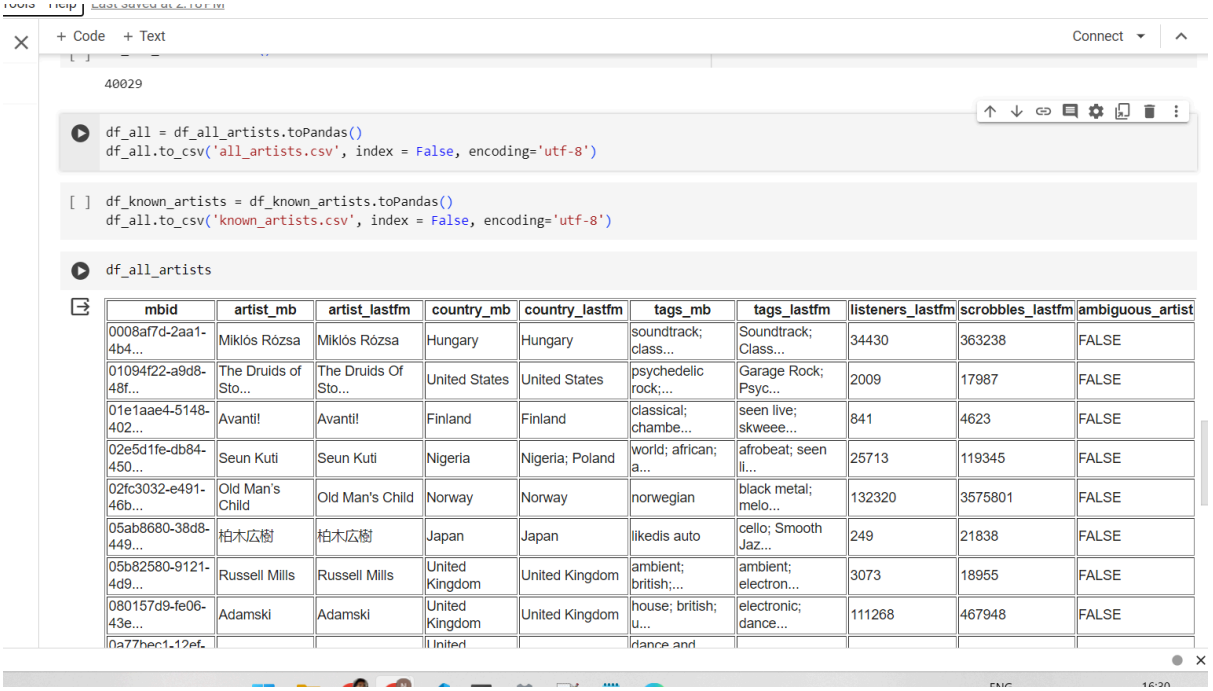
+ Code + Text Connect

```
594907
```

```
[ ] # string cleaning and removing unwanted
    str_cols = [item[0] for item in df_artists.dtypes if item[1].startswith('string')]
    for cols in str_cols:
      df_artists = df_artists.withColumn(cols, trim(df_artists[cols]))
```

```
[ ] # lowercasing the columns and put the underscore

    # create the temp view table for the sql operations
    df_artists.createOrReplaceTempView("artisttbl")
```

```
# to understand the data
# define the categories
# 1. artist_mb
# 2. artist_lastfm
# 3. artist_lastf
# 4. country_mb
# 5. country_lastfm
# 6. tags_mb
# 7. ambiguous_artist bool check True or false

# * indicator_code : to represent the valuation indicator code unit name

query_country_counts = '''select row_number() over (order by country_mb asc) rn, country_mb country, count(*) count
from artisttbl art
where art.country_mb is not null
group by country_mb having count(*) > 1'''

query_artist_counts = '''select row_number() over (order by artist_mb asc) rn, country_mb country, artist_mb artist, count(*) count
```



+ Code + Text Connect

40029

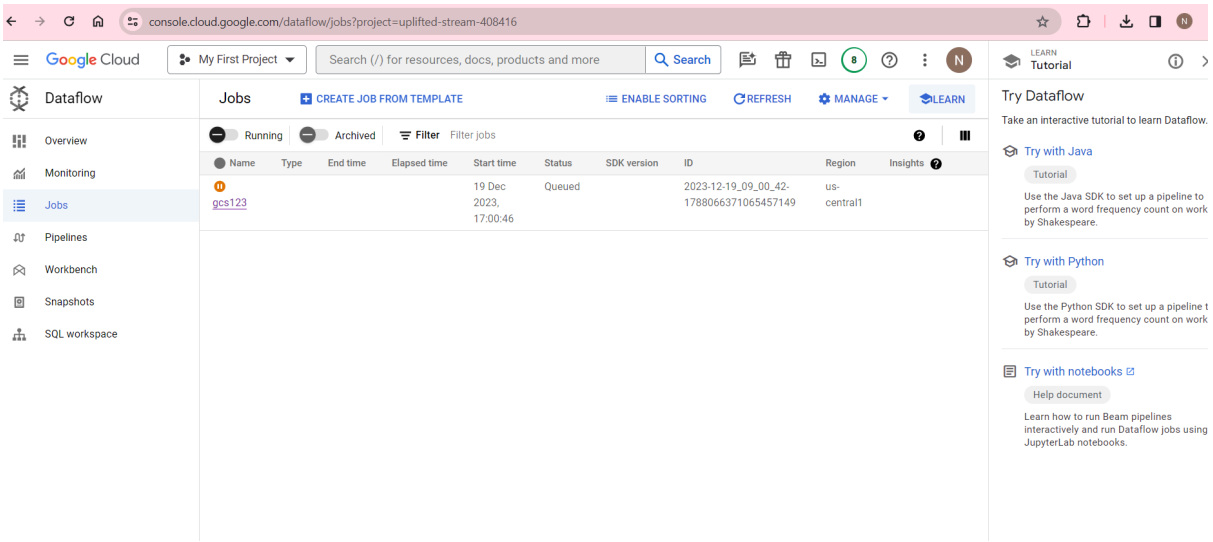
```
df_all = df_all_artists.toPandas()
df_all.to_csv('all_artists.csv', index = False, encoding='utf-8')
```

```
[ ] df_known_artists = df_known_artists.toPandas()
    df_all.to_csv('known_artists.csv', index = False, encoding='utf-8')
```

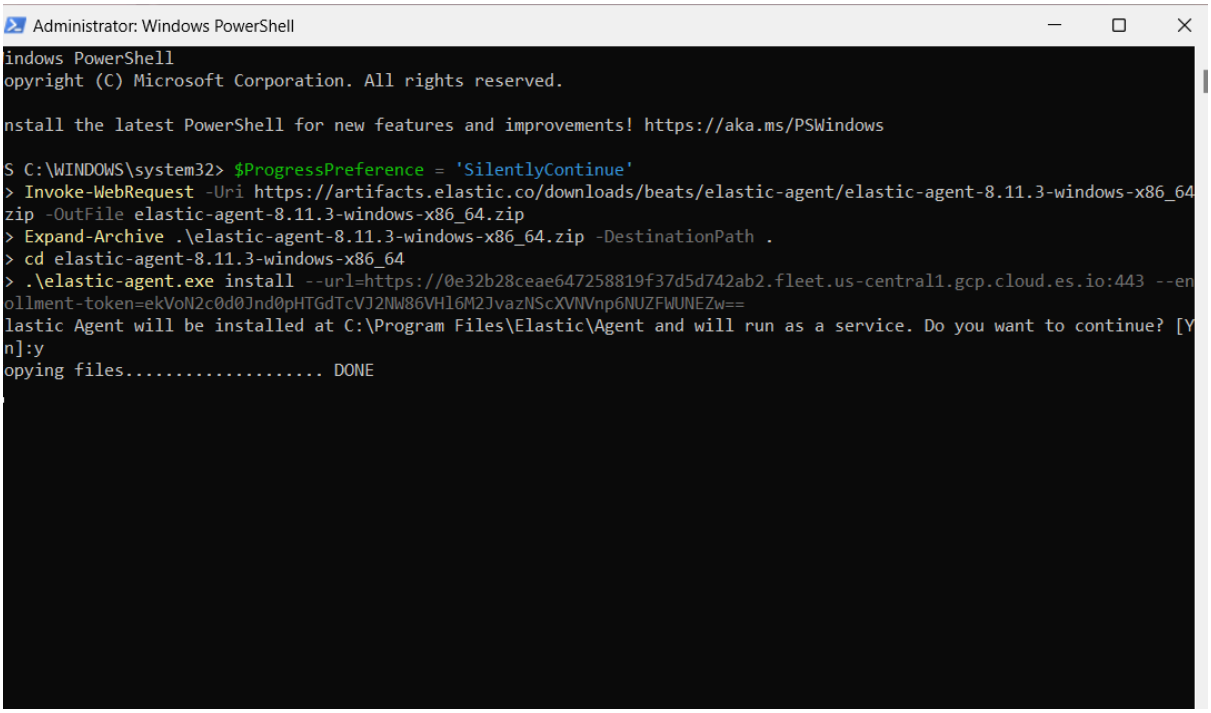
```
df_all_artists
```

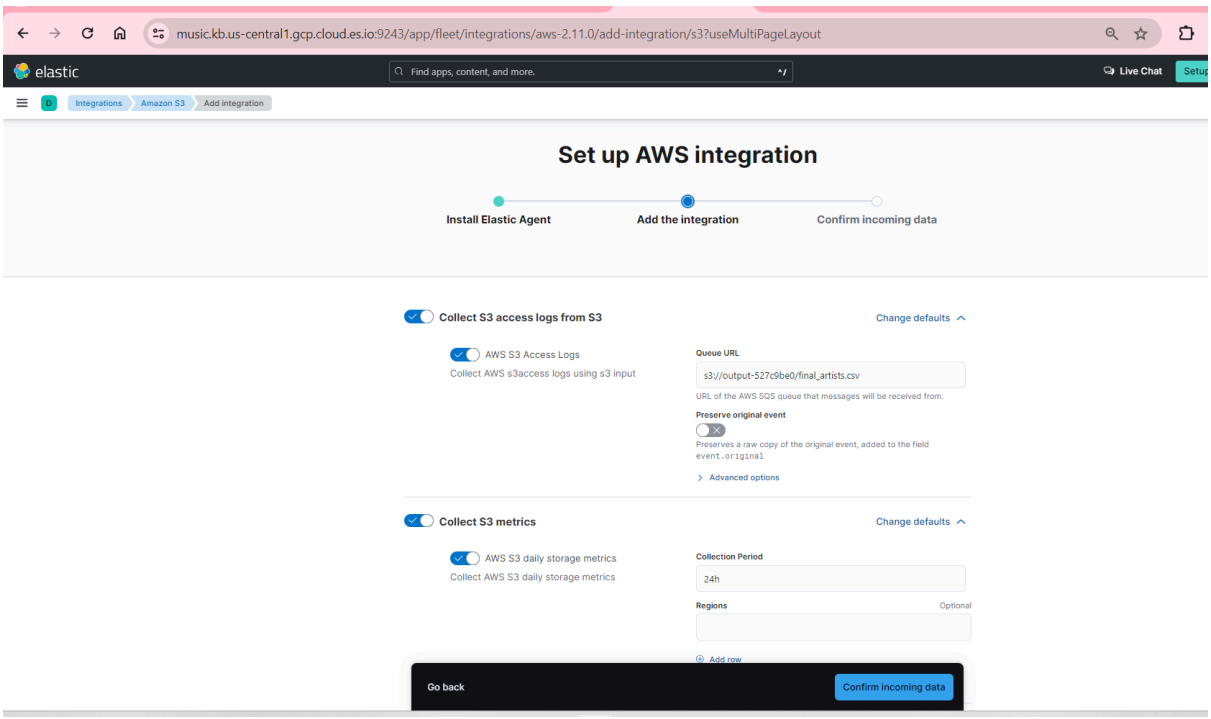
mbid	artist_mb	artist_lastfm	country_mb	country_lastfm	tags_mb	tags_lastfm	listeners_lastfm	scrobbles_lastfm	ambiguous_artist
0008af7d-2aa1-4b4...	Miklós Rózsa	Miklós Rózsa	Hungary	Hungary	soundtrack; class...	Soundtrack; Class...	34430	363238	FALSE
01094f22-a9d8-48f...	The Druids of Sto...	The Druids Of Sto...	United States	United States	psychedelic rock;...	Garage Rock; Psyc...	2009	17987	FALSE
01e1aae4-5148-402...	Avanti!	Avanti!	Finland	Finland	classical; chambe...	seen live; skweee...	841	4623	FALSE
02e5d1fe-db84-450...	Seun Kuti	Seun Kuti	Nigeria	Nigeria; Poland	world; african; a...	afrobeat; seen li...	25713	119345	FALSE
02fc3032-e491-46b...	Old Man's Child	Old Man's Child	Norway	Norway	norwegian	black metal; melo...	132320	3575801	FALSE
05ab8680-38d8-449...	柏木広樹	柏木広樹	Japan	Japan	likedis auto	cello; Smooth Jaz...	249	21838	FALSE
05b82580-9121-4d9...	Russell Mills	Russell Mills	United Kingdom	United Kingdom	ambient; british;...	ambient; electron...	3073	18955	FALSE
080157d9-fe06-43e...	Adamski	Adamski	United Kingdom	United Kingdom	house; british; u...	electronic; dance...	111268	467948	FALSE
0a77bec1-12ef-...			United		dance and				

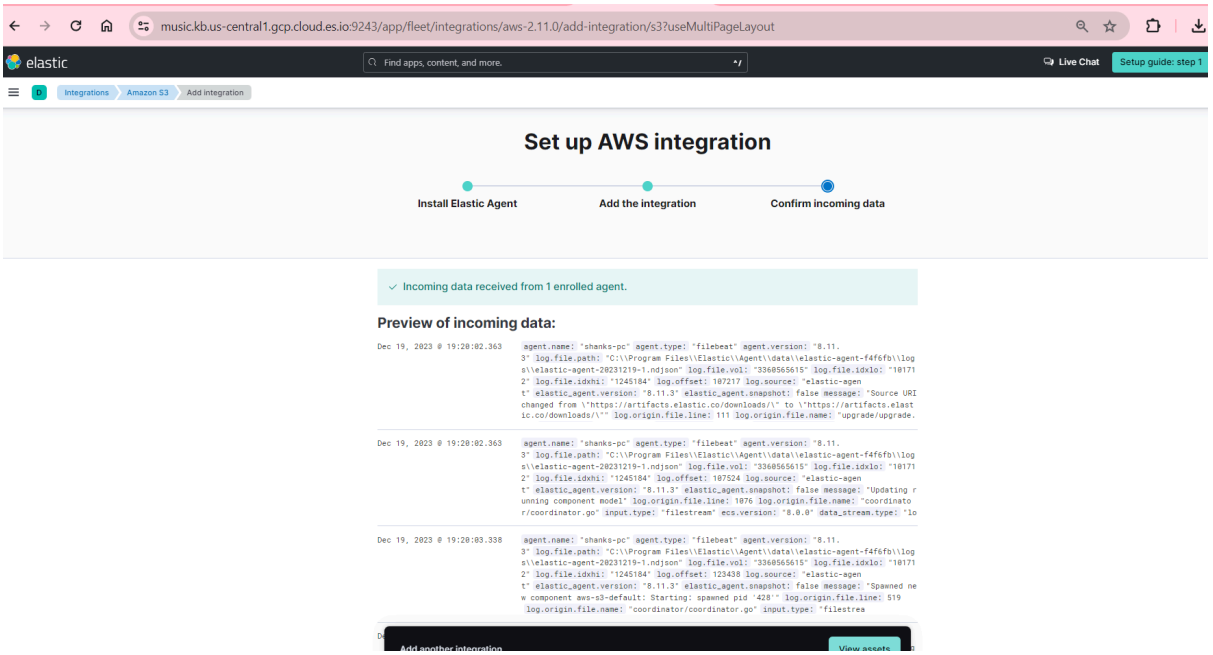
GCP/ElasticCloud integrations dataflow jobs



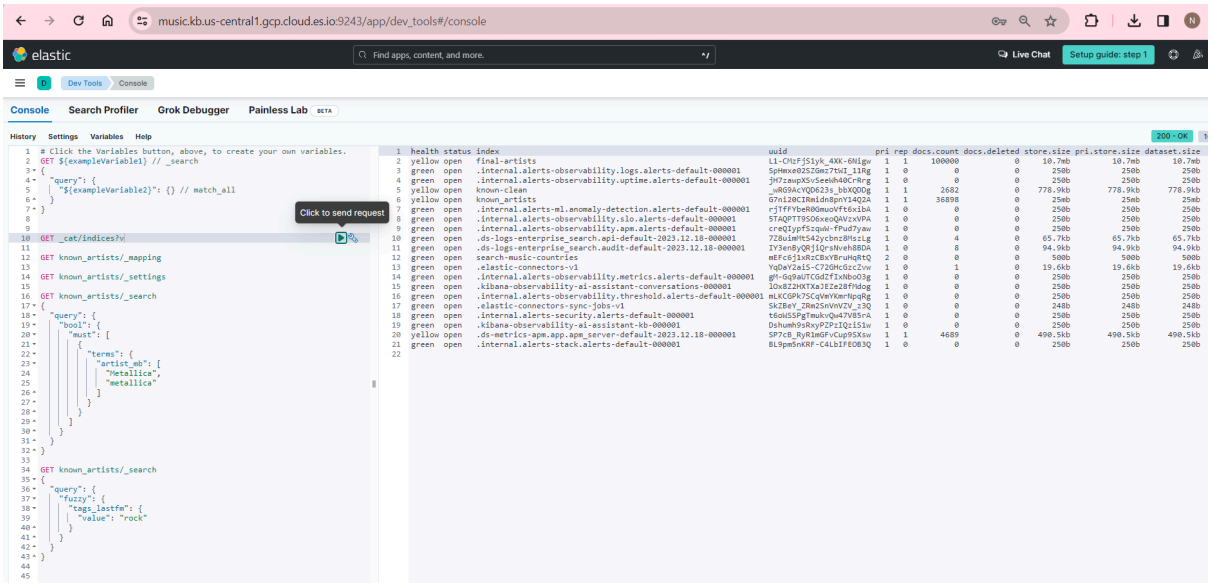
Installing agents for the elastic cloud S3 integrations

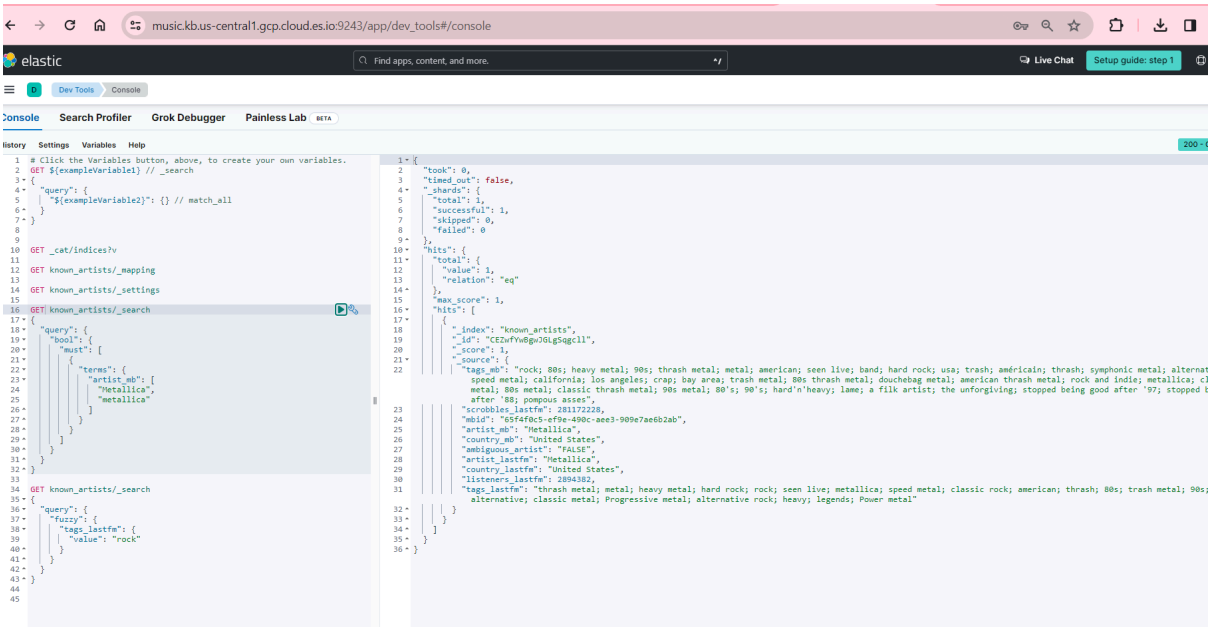

Administrator: Windows PowerShell
indows PowerShell
opyright (C) Microsoft Corporation. All rights reserved.
nstall the latest PowerShell for new features and improvements! https://aka.ms/PSWindows
S C:\WINDOWS\system32> $ProgressPreference = 'SilentlyContinue'
> Invoke-WebRequest -Uri https://artifacts.elastic.co/downloads/beats/elastic-agent/elastic-agent-8.11.3-windows-x86_64
zip -OutFile elastic-agent-8.11.3-windows-x86_64.zip
> Expand-Archive .\elastic-agent-8.11.3-windows-x86_64.zip -DestinationPath .
> cd elastic-agent-8.11.3-windows-x86_64
> .\elastic-agent.exe install --url=https://0e32b28ceae647258819f37d5d742ab2.fleet.us-central1.gcp.cloud.es.io:443 --en
ollment-token=ekVoN2c0d0Jnd0pHTGdTcVJ2NW86VHl6M2JvazNScXVNVnp6NUZFWUNEZw==
lastic Agent will be installed at C:\Program Files\Elastic\Agent and will run as a service. Do you want to continue? [Y
n]:y
opying files..... DONE

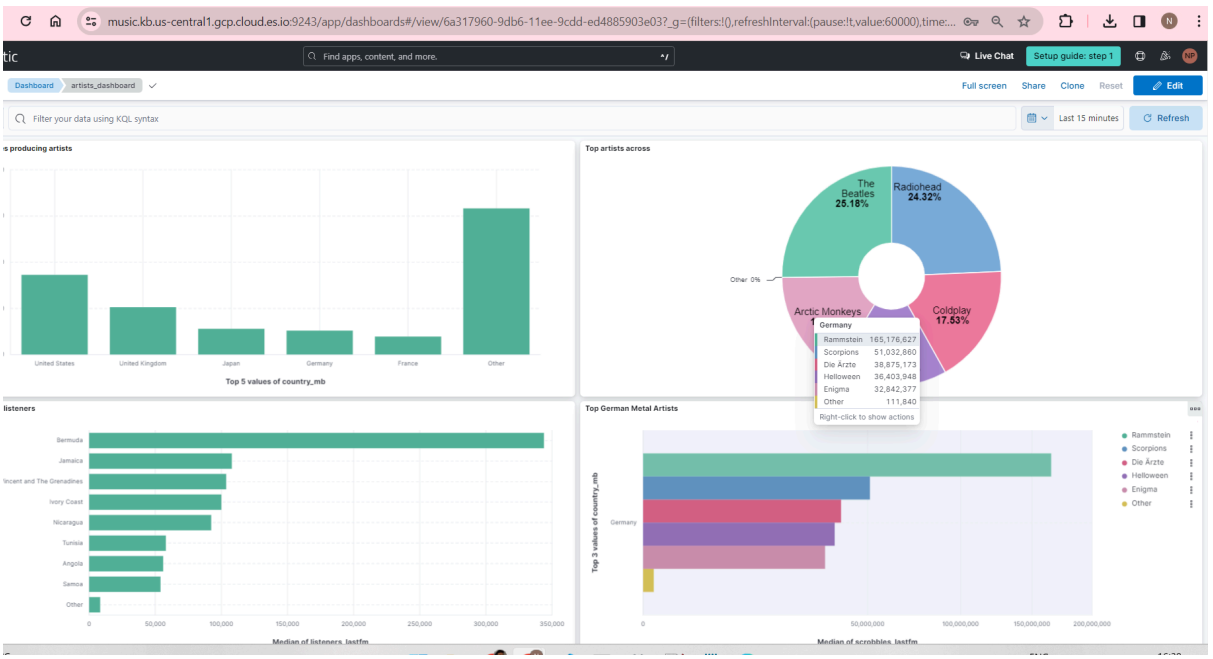

music.kb.us-central1.gcp.cloud.es.io:9243/app/fleet/integrations/aws-2.11.0/add-integration/s3?useMultiPageLayout
elastic
Find apps, content, and more.
Live Chat
Integrations
Amazon S3
Add integration
Set up AWS integration
Install Elastic Agent
Add the integration
Confirm incoming data
Collect S3 access logs from S3
Change defaults
AWS S3 Access Logs
Collect AWS s3access logs using s3 input
Queue URL
s3://output-527c9be0/final_artists.csv
URL of the AWS SQS queue that messages will be received from.
Preserve original event
Preserves a raw copy of the original event, added to the field event.original
Advanced options
Collect S3 metrics
Change defaults
AWS S3 daily storage metrics
Collect AWS S3 daily storage metrics
Collection Period
24h
Regions
Optional
Add row
Go back
Confirm incoming data

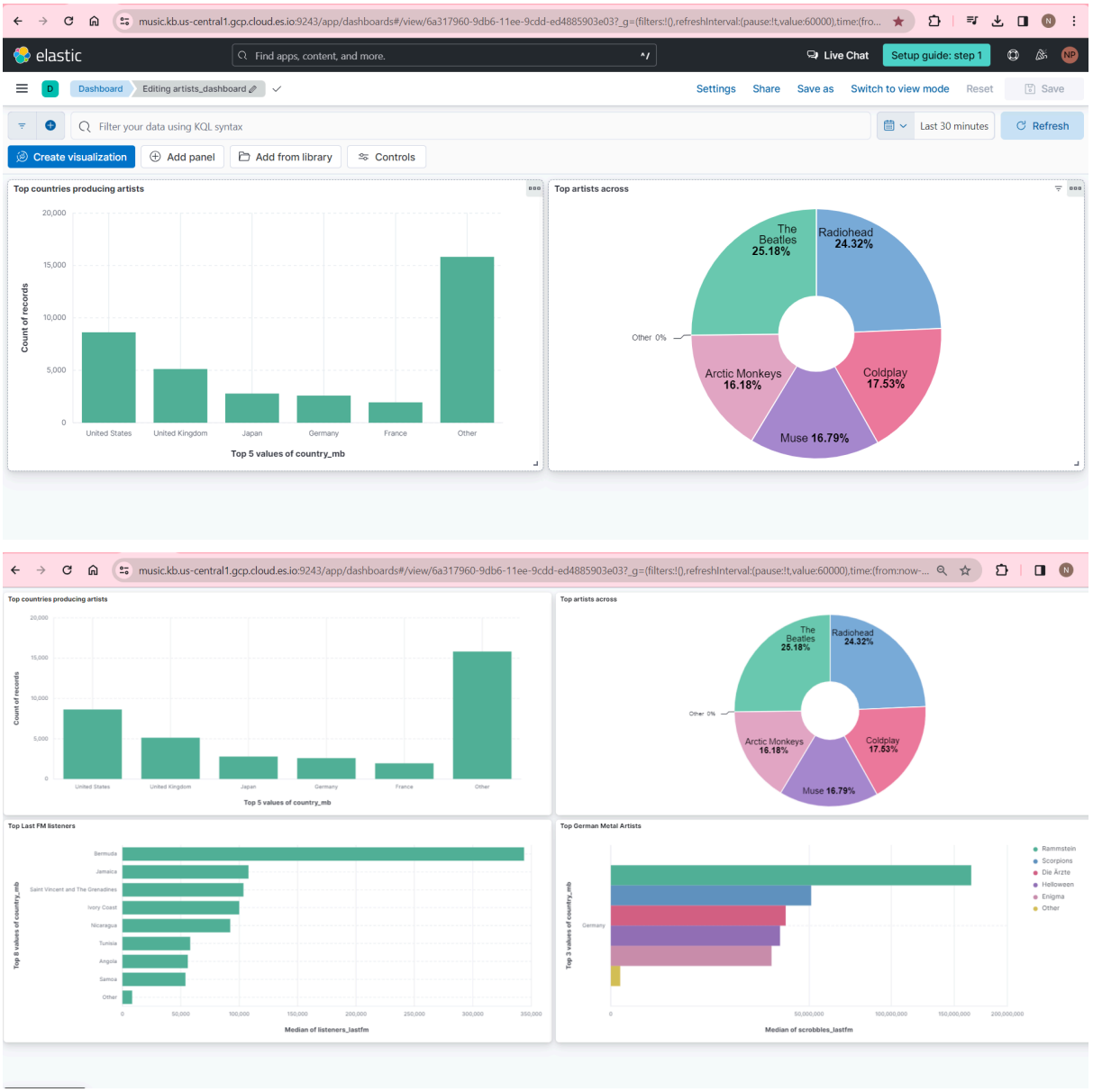


Elasticsearch cloud configs and final kibana dashboard




music.kb.us-central1.gcp.cloud.es.io:9243/app/dev_tools#/console
elastic
Find apps, content, and more.
Live Chat
Setup guide: step 1
Dev Tools
Console
Console
Search Profiler
Grok Debugger
Painless Lab
BETA
History
Settings
Variables
Help
200
# Click the Variables button, above, to create your own variables.
GET ${exampleVariable1} // _search
{
"query": {
"${exampleVariable2}": {} // match_all
}
}
GET _cat/indices?v
GET known_artists/_mapping
GET known_artists/_settings
GET known_artists/_search
{
"query": {
"bool": {
"must": [
{
"terms": {
"artist_mb": [
"Metallica",
"metallica"
]
}
}
]
}
}
}
GET known_artists/_search
{
"query": {
"fuzzy": {
"tags_lastfm": {
"value": "rock"
}
}
}
}
{
"took": 0,
"timed_out": false,
"_shards": {
"total": 1,
"successful": 1,
"skipped": 0,
"failed": 0
},
"hits": {
"total": {
"value": 1,
"relation": "eq"
},
"max_score": 1,
"hits": [
{
"_index": "known_artists",
"_id": "CEZwfYwBgwJGLgSqgCll",
"_score": 1,
"_source": {
"tags_mb": "rock; 80s; heavy metal; 90s; thrash metal; metal; american; seen live; band; hard rock; usa; trash; américain; thrash; symphonic metal; alternat
speed metal; california; los angeles; crap; bay area; trash metal; 80s thrash metal; douchebag metal; american thrash metal; rock and indie; metallica; cl
metal; 80s metal; classic thrash metal; 90s metal; 80's; 90's; hard'n'heavy; lame; a filk artist; the unforgiving; stopped being good after '97; stopped b
after '88; pompous asses",
"scrobbles_lastfm": 281172228,
"mbid": "65f4f0c5-ef9e-490c-aee3-909e7ae6b2ab",
"artist_mb": "Metallica",
"country_mb": "United States",
"ambiguous_artist": "FALSE",
"artist_lastfm": "Metallica",
"country_lastfm": "United States",
"listeners_lastfm": 2894382,
"tags_lastfm": "thrash metal; metal; heavy metal; hard rock; rock; seen live; metallica; speed metal; classic rock; american; thrash; 80s; trash metal; 90s;
alternative; classic metal; Progressive metal; alternative rock; heavy; legends; Power metal"
}
}
]
}
}


music.kb.us-central1.gcp.cloud.es.io:9243/app/dashboards#/view/6a317960-9db6-11ee-9cdd-ed4885903e03?_g=(filters:!(),refreshInterval:(pause:!t,value:60000),time:...
Find apps, content, and more.
Live Chat
Setup guide: step 1
Dashboard
artists_dashboard
Full screen
Share
Clone
Reset
Edit
Filter your data using KQL syntax
Last 15 minutes
Refresh
s producing artists
United States
United Kingdom
Japan
Germany
France
Other
Top 5 values of country_mb
Top artists across
The Beatles 25.18%
Radiohead 24.32%
Coldplay 17.53%
Arctic Monkeys
Other 0%
Germany
Rammstein 165,176,627
Scorpions 51,032,860
Die Ärzte 38,875,173
Helloween 36,403,948
Enigma 32,842,377
Other 111,840
Right-click to show actions
listeners
Bermuda
Jamaica
Ivory Coast
Nicaragua
Tunisia
Angola
Samoa
Other
Median of listeners_lastfm
Top German Metal Artists
Top 3 values of country_mb
Germany
Rammstein
Scorpions
Die Ärzte
Helloween
Enigma
Other
Median of scrobbles_lastfm


music.kb.us-central1.gcp.cloud.es.io:9243/app/dashboards#/view/6a317960-9db6-11ee-9cdd-ed4885903e03?_g=(filters:!(),refreshInterval:(pause:!t,value:60000),time:(fro...
elastic
Find apps, content, and more.
Live Chat
Setup guide: step 1
Dashboard
Editing artists_dashboard
Settings
Share
Save as
Switch to view mode
Reset
Save
Filter your data using KQL syntax
Last 30 minutes
Refresh
Create visualization
Add panel
Add from library
Controls
Top countries producing artists
Count of records
20,000
15,000
10,000
5,000
0
United States
United Kingdom
Japan
Germany
France
Other
Top 5 values of country_mb
Top artists across
The Beatles 25.18%
Radiohead 24.32%
Other 0%
Arctic Monkeys 16.18%
Coldplay 17.53%
Muse 16.79%

music.kb.us-central1.gcp.cloud.es.io:9243/app/dashboards#/view/6a317960-9db6-11ee-9cdd-ed4885903e03?_g=(filters:!(),refreshInterval:(pause:!t,value:60000),time:(from:now-...
Top countries producing artists
Count of records
United States
United Kingdom
Japan
Germany
France
Other
Top 5 values of country_mb
Top artists across
The Beatles 25.18%
Radiohead 24.32%
Other 0%
Arctic Monkeys 16.18%
Coldplay 17.53%
Muse 16.79%
Top Last FM listeners
Top 8 values of country_mb
Bermuda
Jamaica
Saint Vincent and The Grenadines
Ivory Coast
Nicaragua
Tunisia
Angola
Samoa
Other
0
50,000
100,000
150,000
200,000
250,000
300,000
350,000
Median of listeners_lastfm
Top German Metal Artists
Top 3 values of country_mb
Germany
0
50,000,000
100,000,000
150,000,000
200,000,000
Median of scrobbles_lastfm
Rammstein
Scorpions
Die Ärzte
Helloween
Enigma
Other