Trending Youtube Video Analysis With AWS

TABLE OF CONTENTS

I. Introduction

YouTube (the world-famous video sharing website) maintains a list of the top trending videos on the platform. According to Variety magazine, “To determine the year’s top-trending videos, YouTube uses a combination of factors including measuring users interactions (number of views, shares, comments and likes).

This dataset is a daily record of the top trending YouTube videos. This dataset includes several months (and counting) of data on daily trending YouTube videos. Data is included for the US, GB, DE, CA, and FR regions (USA, Great Britain, Germany, Canada, and France, respectively), with up to 200 listed trending videos per day.

II. Extracting data from source

1. Upload data to S3 Bucket using AWS CLI

aws s3 cp . s3://youtube-deprojet/youtube/raw_statistics_reference_data/ --recursive --exclude "*" --include "*.json"

Then upload the csv file of each region to respective folders:

aws s3 cp CAvideos.csv s3://youtube-deprojet/youtube/raw_statistics/region=ca/

aws s3 cp DEvideos.csv s3://youtube-deprojet/youtube/raw_statistics/region=de/

aws s3 cp FRvideos.csv s3://youtube-deprojet/youtube/raw_statistics/region=fr/

aws s3 cp GBvideos.csv s3://youtube-deprojet/youtube/raw_statistics/region=gb/

aws s3 cp INvideos.csv s3://youtube-deprojet/youtube/raw_statistics/region=in/

aws s3 cp JPvideos.csv s3://youtube-deprojet/youtube/raw_statistics/region=jp/

aws s3 cp KRvideos.csv s3://youtube-deprojet/youtube/raw_statistics/region=kr/

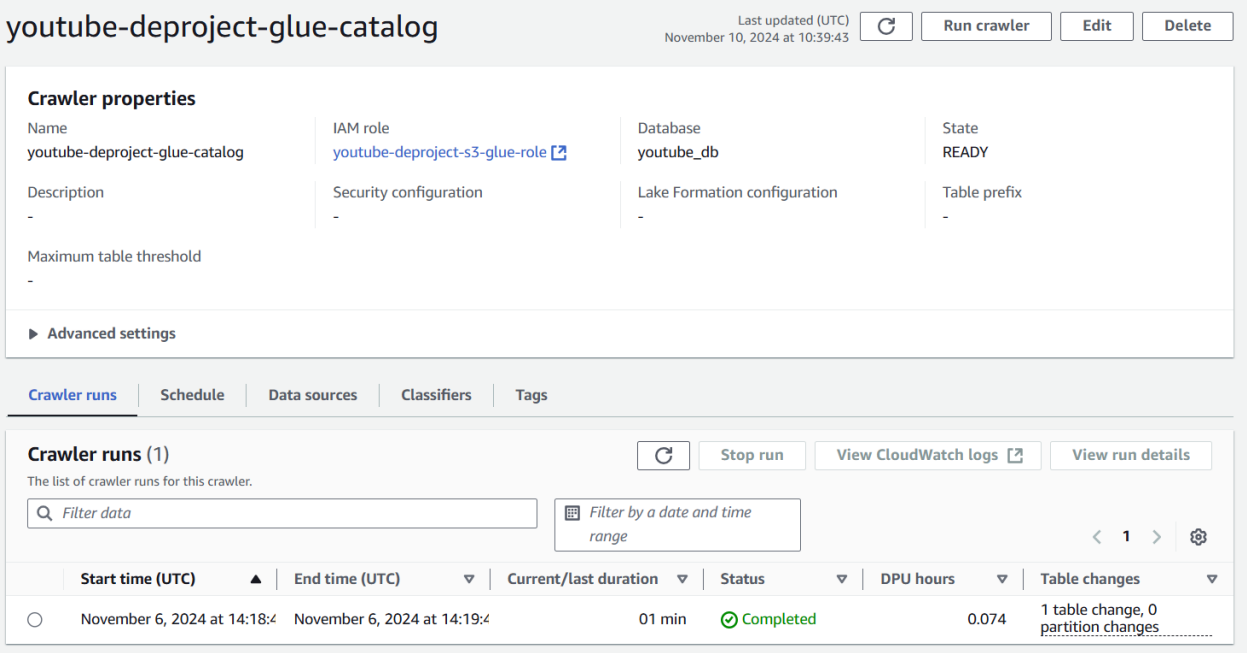
aws s3 cp MXvideos.csv s3://youtube-deprojet/youtube/raw_statistics/region=mx/

aws s3 cp RUvideos.csv s3://youtube-deprojet/youtube/raw_statistics/region=ru/

aws s3 cp USvideos.csv s3://youtube-deprojet/youtube/raw_statistics/region=us/

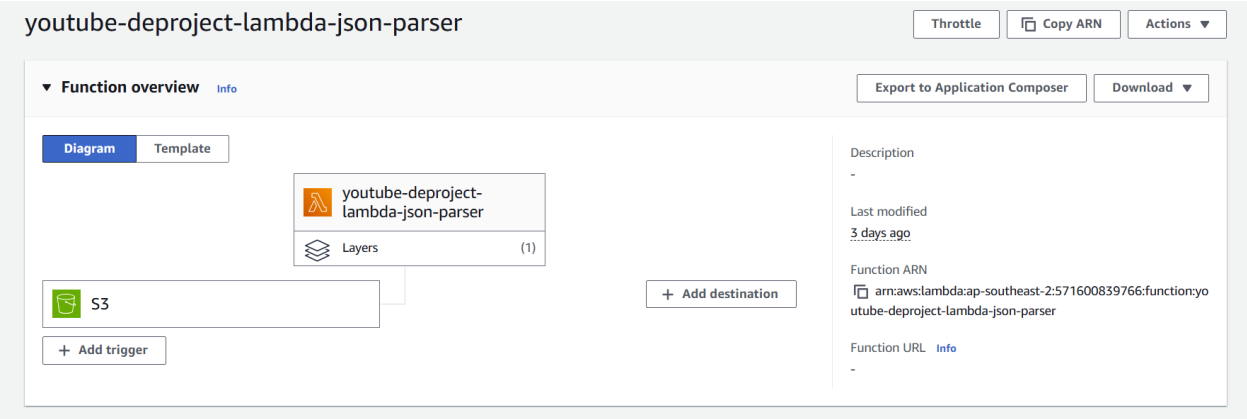
2. Build crawler in AWS Glue

I created the crawler to crawl the data from S3 to a database in Glue.



3. Preprocessing the data using AWS Lambda

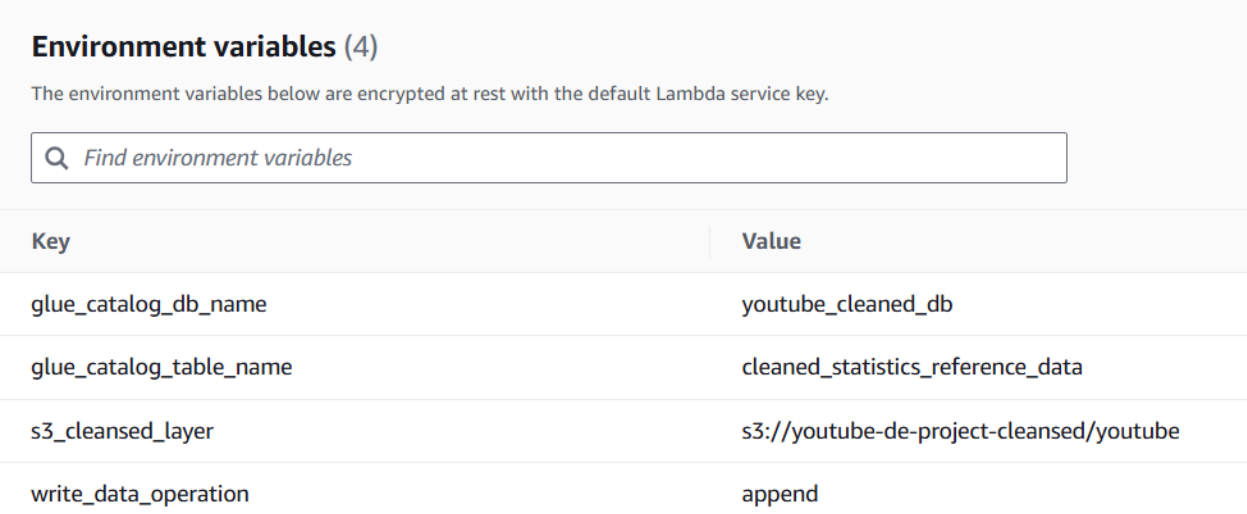
I created a function that will only take the third key in the json data (because that is the only thing we need).



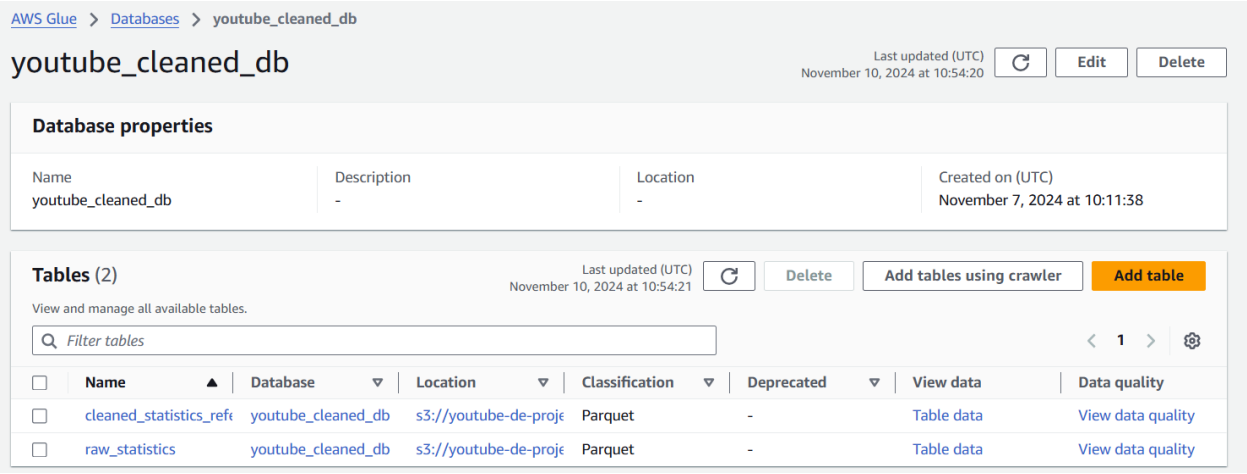
After writing code, I configured the testing event so it will take the data from the data source we want.

```
},
"s3": {
  "s3SchemaVersion": "1.0",
  "configurationId": "testConfigRule",
  "bucket": {
    "name": "youtube-deprojet",
    "ownerIdentity": {
      "principalId": "EXAMPLE"
    },
    "arn": "arn:aws:s3:::youtube-deprojet"
  },
  "object": {
    "key": "youtube/raw_statistics_reference_data/CA_category_id.json",
    "size": 1024,
    "eTag": "0123456789abcdef0123456789abcdef",
    "sequencer": "0A1B2C3D4E5F678901"
  }
}
```

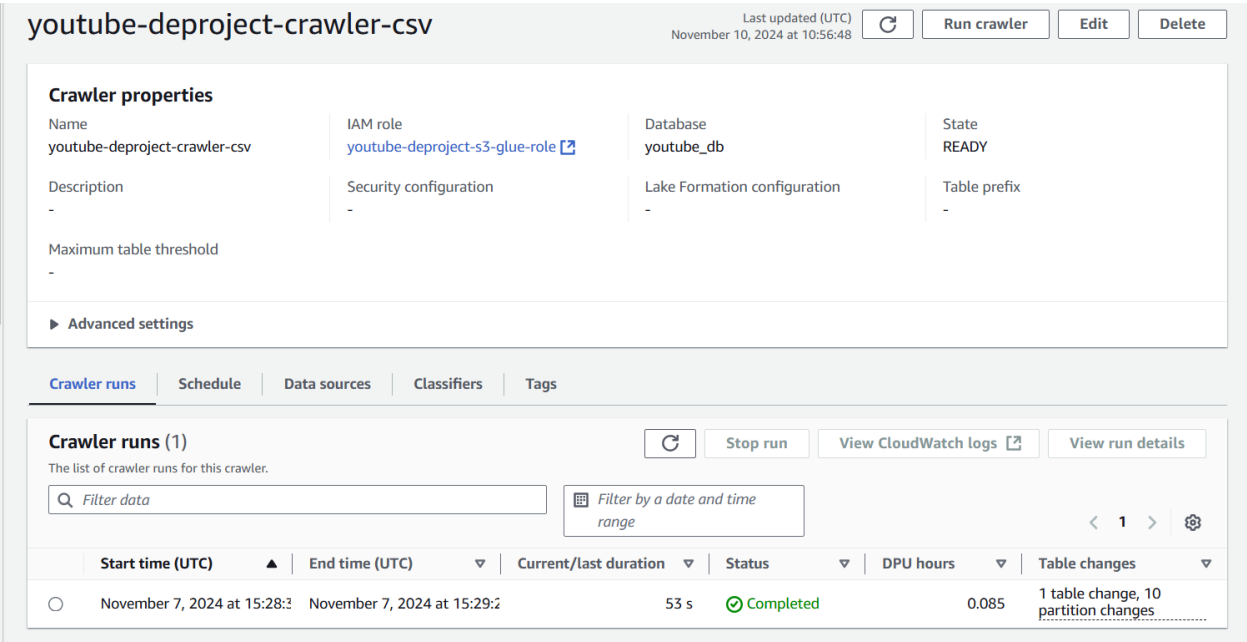

We also need to configure the variables inside the code so it can perform accordingly.



After running, we will have a new cleansed database in our Glue.



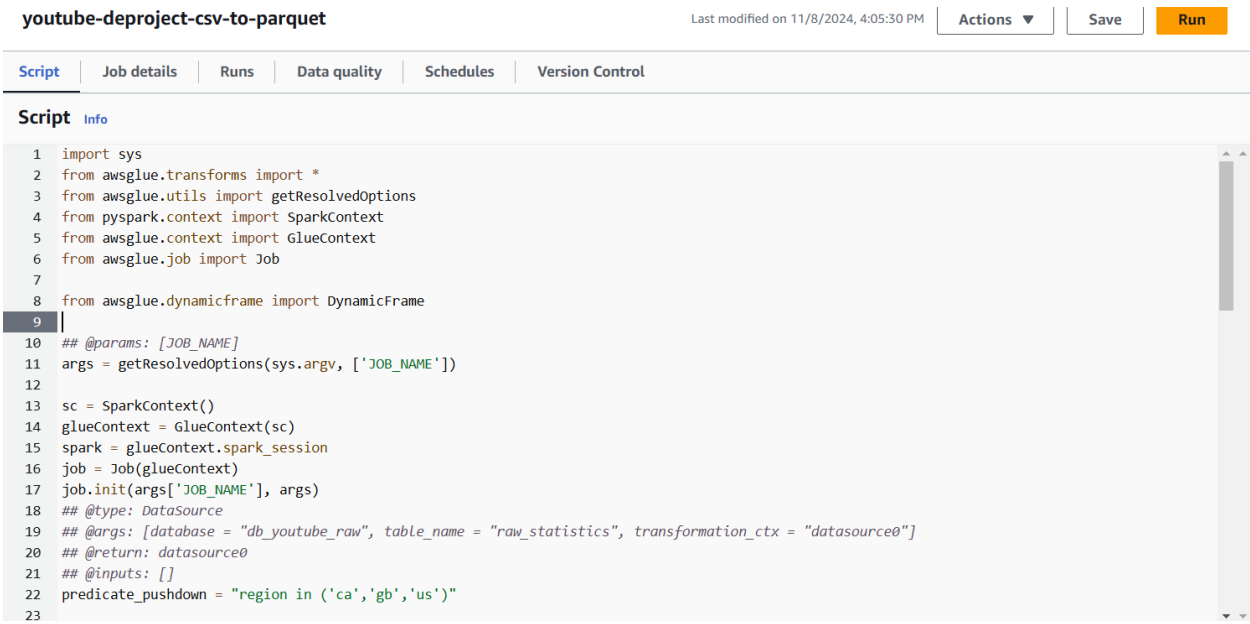
4. Load csv data to the table by using Crawler



III. Transform the current data

1. Moving csv data to a cleaner database

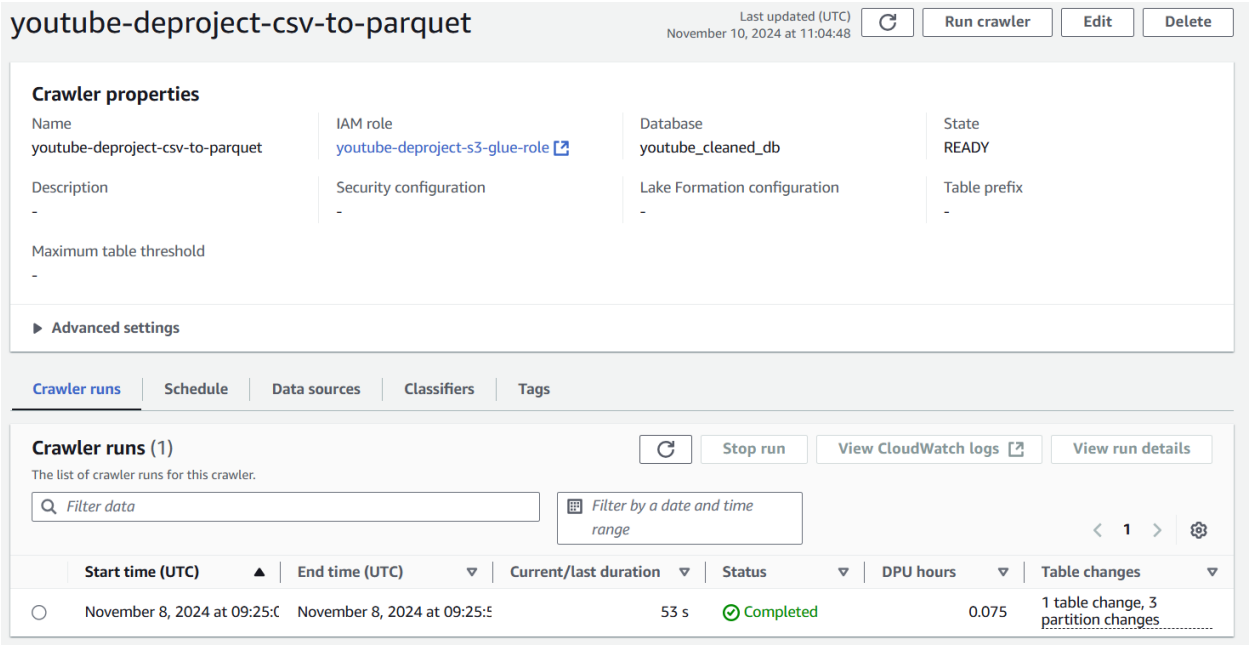
Add a job in AWS Glue.




```
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

from awsglue.dynamicframe import DynamicFrame

## @params: [JOB_NAME]
args = getResolvedOptions(sys.argv, ['JOB_NAME'])

sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)
job.init(args['JOB_NAME'], args)
## @type: DataSource
## @args: [database = "db_youtube_raw", table_name = "raw_statistics", transformation_ctx = "datasource0"]
## @return: datasource0
## @inputs: []
predicate_pushdown = "region in ('ca','gb','us')"

```

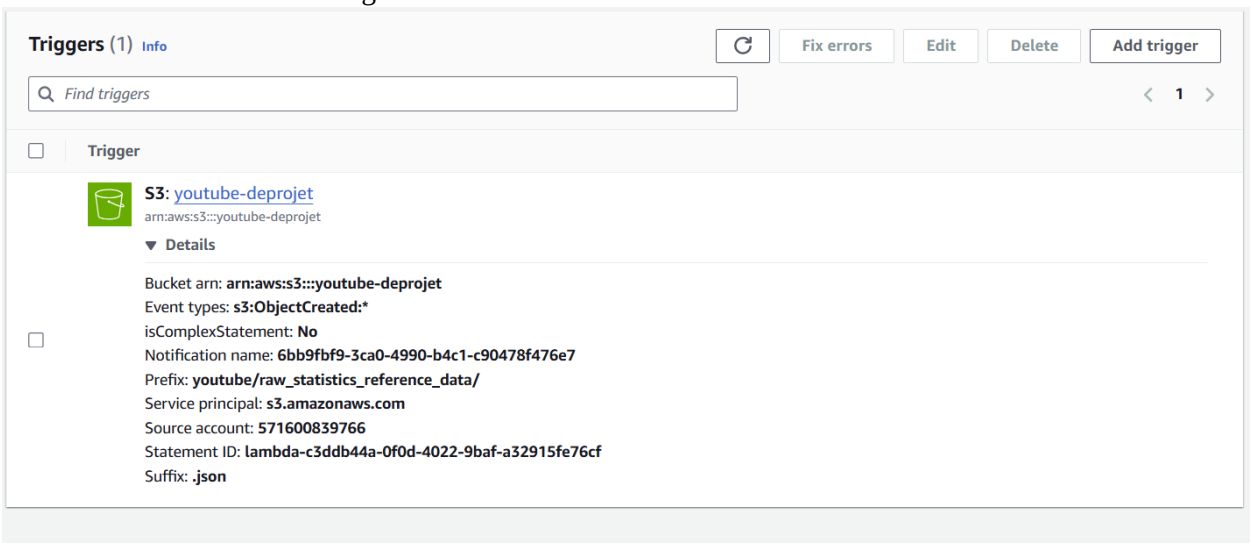
This job will cover through all the data, apply data type based on the schema we specified and store the preprocessed data in a S3 bucket

2. Build a crawler to crawl the cleansed csv data from S3 bucket to AWS Glue database



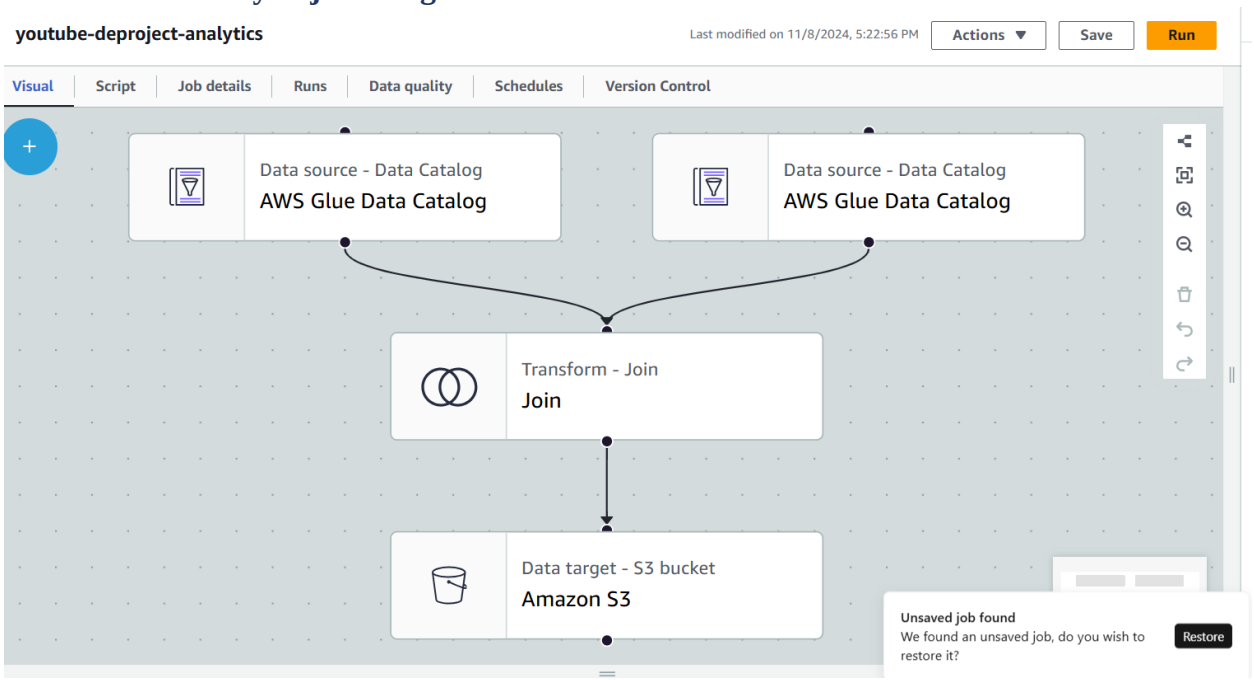
3. Add Trigger event in Lambda

A trigger is created to make this pipeline automated. When a new file is added to the original S3 bucket, the lambda function will run right after for that data.



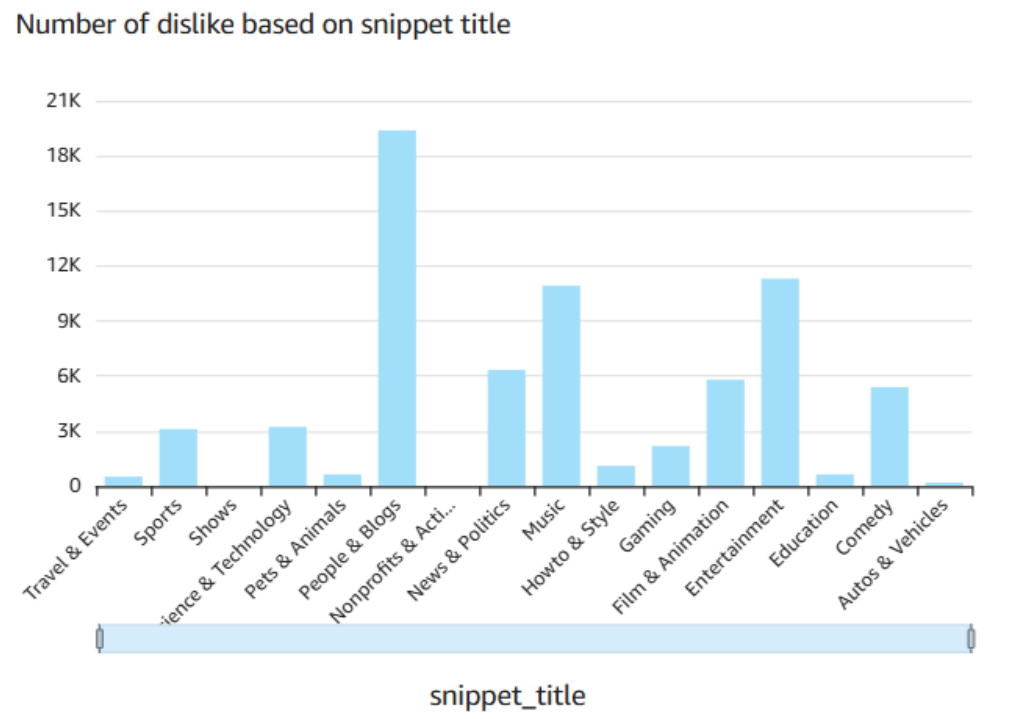
IV. Build a pipeline to get data for analytics

1. Create an analytic job using visual ETL in AWS Glue



2. Use the newly data to analyze with Quicksight on AWS

Build a simple visual to look through the key insight inside the data.



V. Conclusion

- Challenges:
    - Handling inconsistencies in the text data, such as special characters and stopwords, was challenging.
    - Learning the architecture of AWS and the services it offers.
    - Visualize and contextualize the overall work pipeline.
- Summary:
    - This project is to implement a simple data pipeline using AWS that help contribute to my understanding the real word data process.
    - Learned effective techniques for implementing and optimizing the services.

VI. References

https://www.kaggle.com/datasets/datasnaek/youtube-new

https://www.youtube.com/watch?v=yZKJFKu49Dk&list=PLBJe2dFI4sgvQTNNkI3ETYJgNPR4CBpFd