YouTube Trending Data Engineering and Analytics Using AWS

INTRODUCTION

In today's digital age, online video consumption is not only a major form of entertainment but also a significant indicator of cultural trends and consumer behavior. YouTube, being the world's largest video-sharing platform, generates massive amounts of structured and semi-structured data daily. This project focuses on building a fully serverless data engineering pipeline using Amazon Web Services (AWS) that handles ingestion, transformation, storage, querying, and analysis of YouTube trending video statistics.

RESEARCH QUESTIONS / ANALYTICAL OBJECTIVES

This project addresses the following questions:

- Which video categories attract the highest number of views and likes across different countries?
- How do viewer sentiments vary across regions based on likes and dislikes?
- What are the most engaging video categories in the US, CA, and GB?
- Are there consistent patterns in user engagement by region and category?
- How can a cloud-native architecture support scalable, serverless analytics for streaming video data?

DATA SOURCES

The data used in this project originates from the YouTube Trending Video Statistics dataset available on Kaggle and includes CSV files for various countries:

Country	File Name
United States	USvideos.csv
Great Britain	GBvideos.csv
Canada	CAvideos.csv
Germany	DEvideos.csv
France	FRvideos.csv
India	INvideos.csv

Japan	JPvideos.csv
South Korea	KRvideos.csv
Mexico	MXvideos.csv
Russia	RUvideos.csv

Each file contains trending video data for that specific region, with fields such as video_id, trending_date, title, channel_title, publish_time, views, likes, dislikes, and comment_count. A JSON file maps category_id to human-readable video categories.

METHODOLOGY

This project implements a multi-region ingestion pipeline that processes trending video data across 10 countries. The pipeline includes:

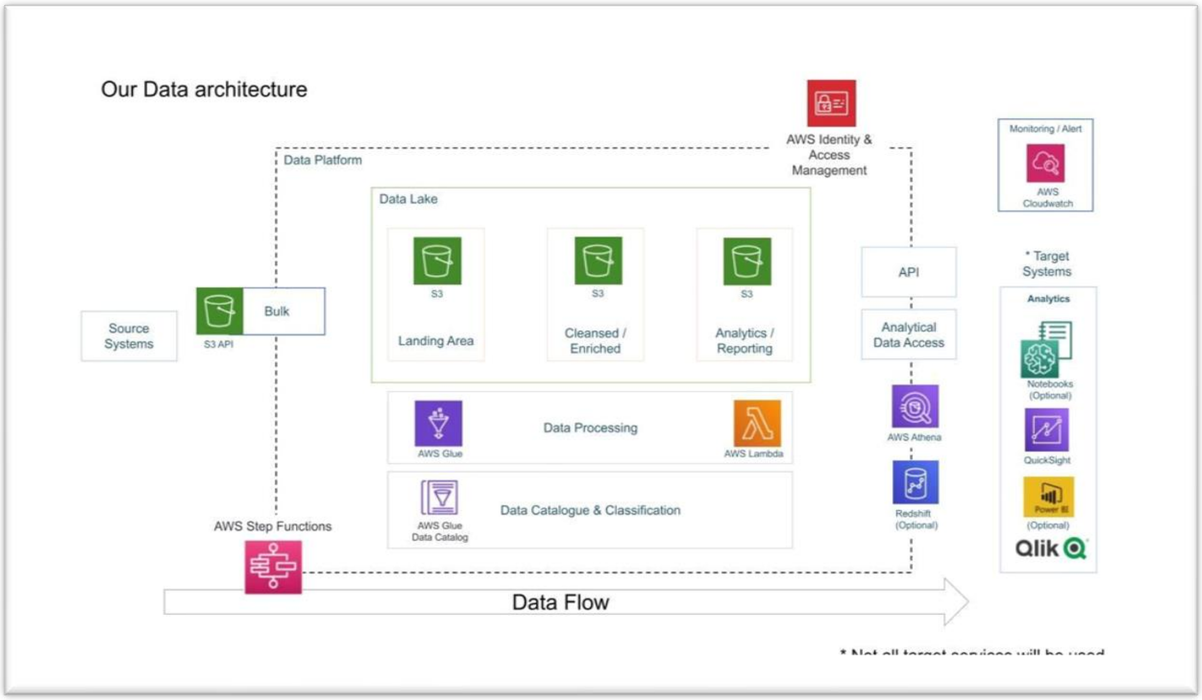


Figure 1: Architecture Diagram of the AWS-based Data Pipeline

1. S3-Based Data Lake Setup: Regional CSVs are stored under region-partitioned folders.

2. Lambda Function: Processes new uploads, normalizes JSON, and writes Parquet data to S3.

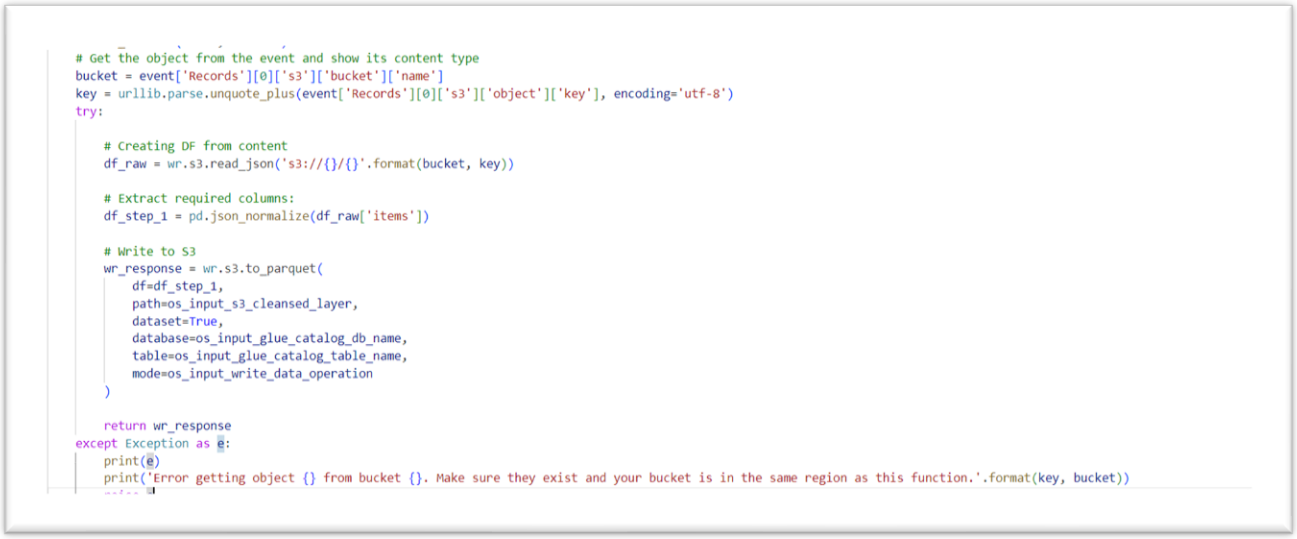
3. Glue Job: Transforms and maps data schema using PySpark and writes back partitioned outputs.

4. Athena SQL Layer: Enables querying and filtering data per region.

5. QuickSight Dashboard: Visualizes views, likes, dislikes, and category patterns.

CODE IMPLEMENTATION

1. AWS Lambda Function – Event-based Data Ingestion



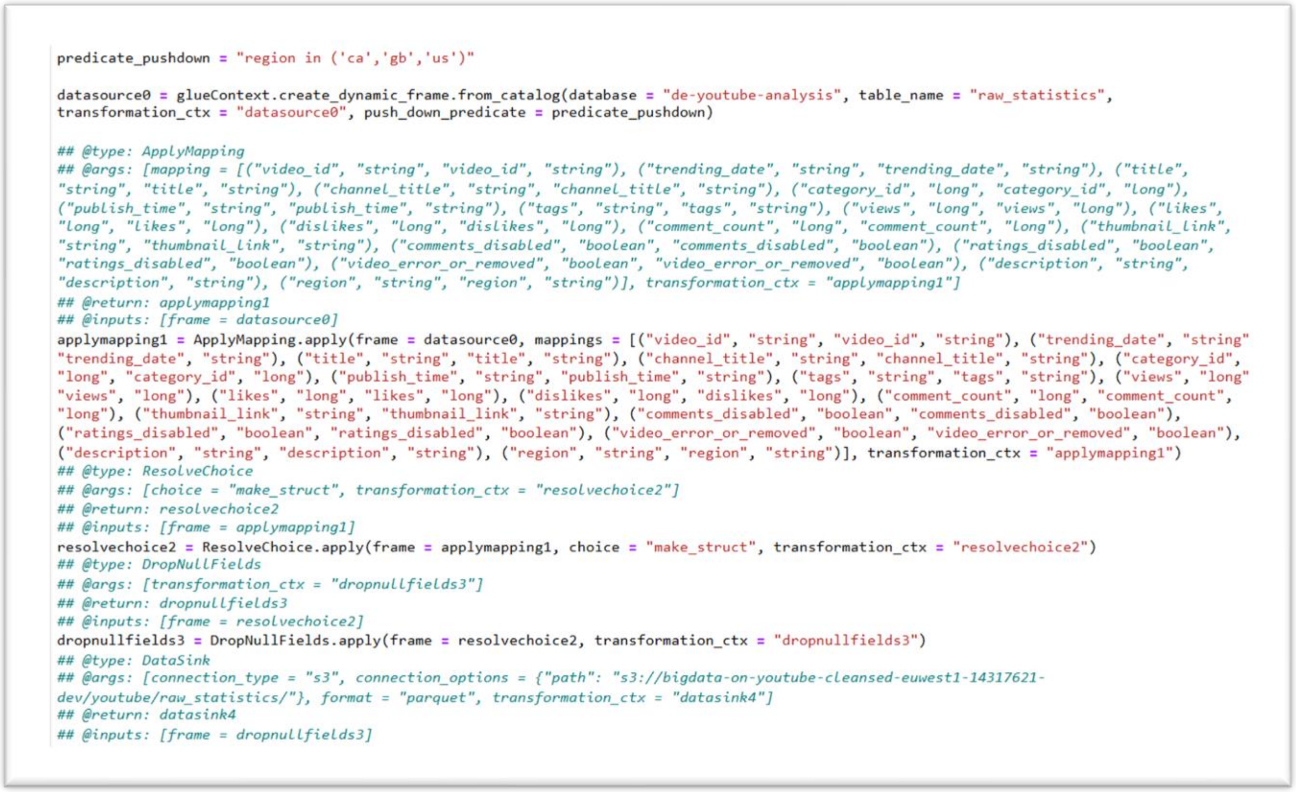
```
# Get the object from the event and show its content type
bucket = event['Records'][0]['s3']['bucket']['name']
key = urllib.parse.unquote_plus(event['Records'][0]['s3']['object']['key'], encoding='utf-8')
try:

    # Creating DF from content
    df_raw = wr.s3.read_json('s3://{}/{}'.format(bucket, key))

    # Extract required columns:
    df_step_1 = pd.json_normalize(df_raw['items'])

    # Write to S3
    wr_response = wr.s3.to_parquet(
        df=df_step_1,
        path=os_input_s3_cleansed_layer,
        dataset=True,
        database=os_input_glue_catalog_db_name,
        table=os_input_glue_catalog_table_name,
        mode=os_input_write_data_operation
    )

    return wr_response
except Exception as e:
    print(e)
    print('Error getting object {} from bucket {}. Make sure they exist and your bucket is in the same region as this function.'.format(key, bucket))
```

This Lambda function is triggered every time a new JSON file is uploaded to the S3 raw layer. It reads the data, flattens nested structures using pandas, and writes it to a 'cleansed' layer in Parquet format. It also registers the dataset into AWS Glue Data Catalog for further querying.

Key Responsibilities:

- Triggered by S3 upload events
- Reads raw YouTube data
- Flattens nested JSON
- Stores clean Parquet data in S3
- Registers schema with AWS Glue Catalog

2. AWS Glue Job – Region-wise ETL in PySpark



```
predicate_pushdown = "region in ('ca','gb','us')"

datasource0 = glueContext.create_dynamic_frame.from_catalog(database = "de-youtube-analysis", table_name = "raw_statistics",
transformation_ctx = "datasource0", push_down_predicate = predicate_pushdown)

## @type: ApplyMapping
## @args: [mapping = [("video_id", "string", "video_id", "string"), ("trending_date", "string", "trending_date", "string"), ("title",
"string", "title", "string"), ("channel_title", "string", "channel_title", "string"), ("category_id", "long", "category_id", "long"),
("publish_time", "string", "publish_time", "string"), ("tags", "string", "tags", "string"), ("views", "long", "views", "long"), ("likes",
"long", "likes", "long"), ("dislikes", "long", "dislikes", "long"), ("comment_count", "long", "comment_count", "long"), ("thumbnail_link",
"string", "thumbnail_link", "string"), ("comments_disabled", "boolean", "comments_disabled", "boolean"), ("ratings_disabled", "boolean",
"ratings_disabled", "boolean"), ("video_error_or_removed", "boolean", "video_error_or_removed", "boolean"), ("description", "string",
"description", "string"), ("region", "string", "region", "string")], transformation_ctx = "applymapping1"]
## @return: applymapping1
## @inputs: [frame = datasource0]
applymapping1 = ApplyMapping.apply(frame = datasource0, mappings = [("video_id", "string", "video_id", "string"), ("trending_date", "string"
"trending_date", "string"), ("title", "string", "title", "string"), ("channel_title", "string", "channel_title", "string"), ("category_id",
"long", "category_id", "long"), ("publish_time", "string", "publish_time", "string"), ("tags", "string", "tags", "string"), ("views", "long"
"views", "long"), ("likes", "long", "likes", "long"), ("dislikes", "long", "dislikes", "long"), ("comment_count", "long", "comment_count",
"long"), ("thumbnail_link", "string", "thumbnail_link", "string"), ("comments_disabled", "boolean", "comments_disabled", "boolean"),
("ratings_disabled", "boolean", "ratings_disabled", "boolean"), ("video_error_or_removed", "boolean", "video_error_or_removed", "boolean"),
("description", "string", "description", "string"), ("region", "string", "region", "string")], transformation_ctx = "applymapping1")
## @type: ResolveChoice
## @args: [choice = "make_struct", transformation_ctx = "resolvechoice2"]
## @return: resolvechoice2
## @inputs: [frame = applymapping1]
resolvechoice2 = ResolveChoice.apply(frame = applymapping1, choice = "make_struct", transformation_ctx = "resolvechoice2")
## @type: DropNullFields
## @args: [transformation_ctx = "dropnullfields3"]
## @return: dropnullfields3
## @inputs: [frame = resolvechoice2]
dropnullfields3 = DropNullFields.apply(frame = resolvechoice2, transformation_ctx = "dropnullfields3")
## @type: DataSink
## @args: [connection_type = "s3", connection_options = {"path": "s3://bigdata-on-youtube-cleansed-euwest1-14317621-
dev/youtube/raw_statistics/"}, format = "parquet", transformation_ctx = "datasink4"]
## @return: datasink4
## @inputs: [frame = dropnullfields3]
```

The Glue ETL job runs in batch mode to process datasets from various regions (e.g., US, GB, CA). It applies a schema, filters regions using a push-down predicate, drops null fields, and finally writes the transformed data in partitioned Parquet format to S3.

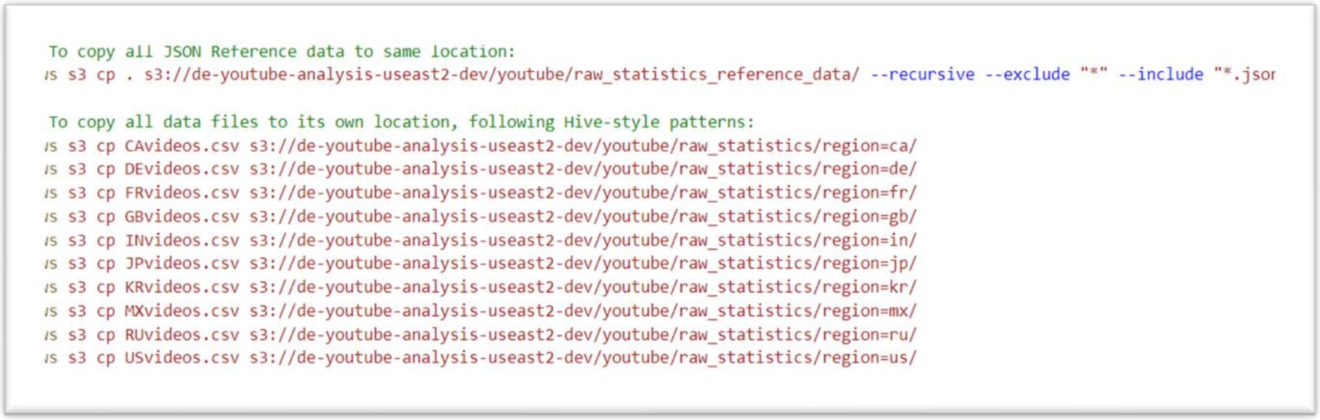
Key Responsibilities:

- Reads data from Glue Data Catalog
- Applies schema mapping using ApplyMapping
- Drops null fields and resolves schema inconsistencies
- Writes partitioned output to S3

3. S3 CLI Script – Data Upload to Raw Layer

This CLI script helps batch upload country-specific video data to S3 using Hive-style directory structure. This format ensures the datasets are immediately queryable by Athena once cataloged.

- Example Commands:



```
 To copy all JSON Reference data to same location:
/s s3 cp . s3://de-youtube-analysis-useast2-dev/youtube/raw_statistics_reference_data/ --recursive --exclude "*" --include "*.jsor

 To copy all data files to its own location, following Hive-style patterns:
/s s3 cp CAvideos.csv s3://de-youtube-analysis-useast2-dev/youtube/raw_statistics/region=ca/
/s s3 cp DEvideos.csv s3://de-youtube-analysis-useast2-dev/youtube/raw_statistics/region=de/
/s s3 cp FRvideos.csv s3://de-youtube-analysis-useast2-dev/youtube/raw_statistics/region=fr/
/s s3 cp GBvideos.csv s3://de-youtube-analysis-useast2-dev/youtube/raw_statistics/region=gb/
/s s3 cp INvideos.csv s3://de-youtube-analysis-useast2-dev/youtube/raw_statistics/region=in/
/s s3 cp JPvideos.csv s3://de-youtube-analysis-useast2-dev/youtube/raw_statistics/region=jp/
/s s3 cp KRvideos.csv s3://de-youtube-analysis-useast2-dev/youtube/raw_statistics/region=kr/
/s s3 cp MXvideos.csv s3://de-youtube-analysis-useast2-dev/youtube/raw_statistics/region=mx/
/s s3 cp RUvideos.csv s3://de-youtube-analysis-useast2-dev/youtube/raw_statistics/region=ru/
/s s3 cp USvideos.csv s3://de-youtube-analysis-useast2-dev/youtube/raw_statistics/region=us/
```

ANALYSIS & VISUALIZATIONS

Visualization 1: Views by Category

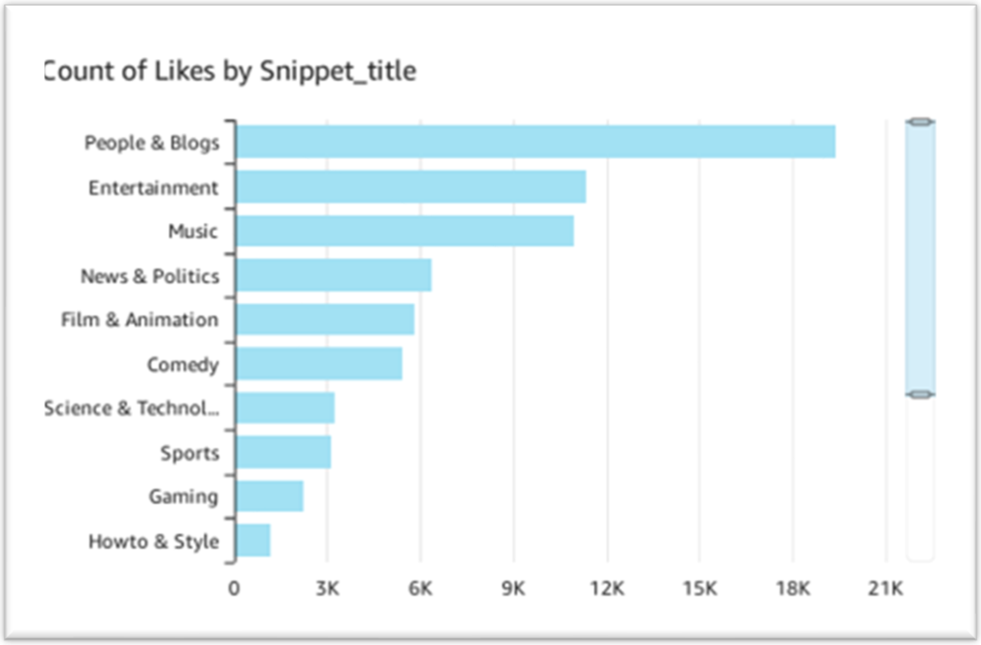


Figure: Screenshot of Visualization 1: Views by Category

This bar chart visualizes the total number of likes received across different YouTube video categories. The 'People & Blogs' category shows the highest engagement with over 20,000 likes, indicating a strong viewer interest in personal storytelling and informal content. It is followed by 'Entertainment' and 'Music', both of which are traditionally strong performers on the platform due to their mass appeal. Categories such as 'News & Politics' and 'Comedy' reflect timely, regional or personality-driven content, while 'How-to & Style' and 'Science & Technology' appeal to niche but dedicated audiences. This visualization helps identify what genres drive the most engagement globally.

Visualization 2: Likes vs Dislikes by Region

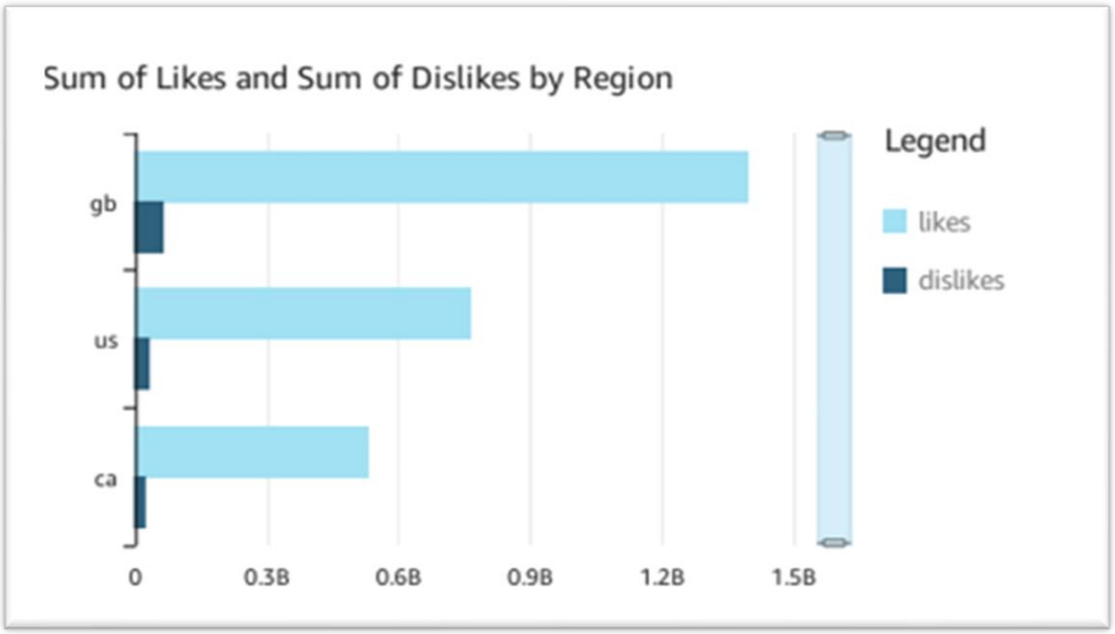


Figure: Screenshot of Visualization 2: Likes vs Dislikes by Region

This grouped bar chart compares the total number of likes and dislikes across three regions: GB (Great Britain), US (United States), and CA (Canada). GB leads in overall likes, with more than 1.5 billion, suggesting higher user interaction levels or broader reach. Dislikes remain relatively low in comparison, suggesting more favorable audience sentiment or less controversial content. The contrast between likes and dislikes in each region also gives an indication of content polarity and regional tolerance toward various genres or topics.

Visualization 3: Sum of Views by Region

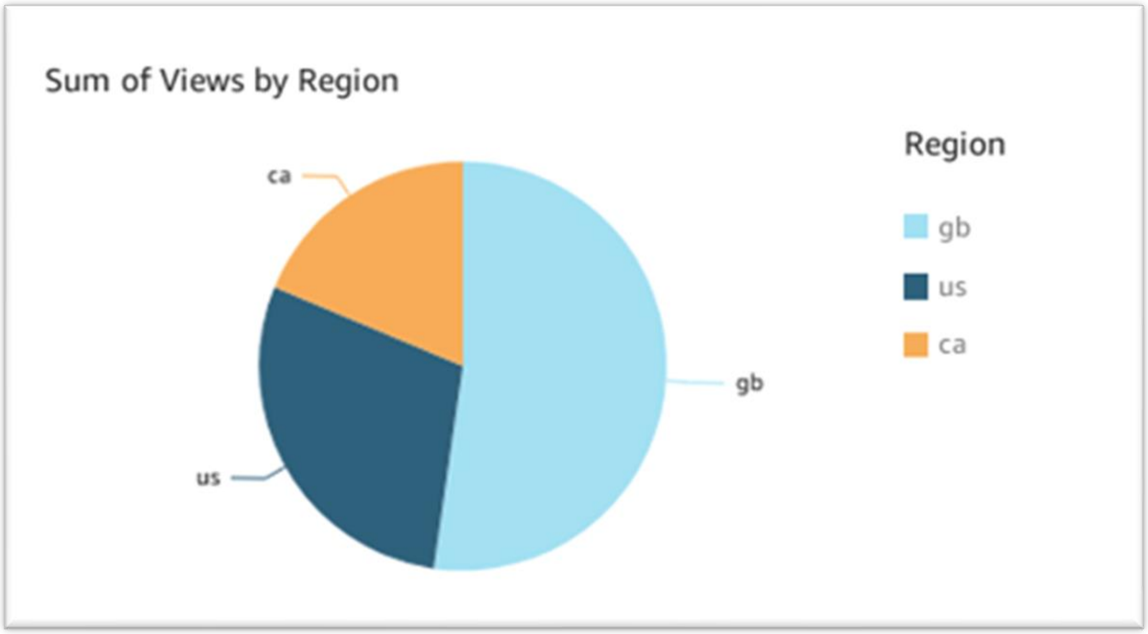


Figure: Screenshot of Visualization 3: Sum of Views by Region

The pie chart displays the total viewership share for YouTube videos by region. Great Britain commands the largest share, followed closely by the US and Canada. This regional distribution may be reflective of population size, content relevance, and market maturity in digital video consumption. It also reveals where YouTube content is resonating most with users and where regional marketing or targeting efforts could be optimized.

Visualization 4: Top Categories by Likes

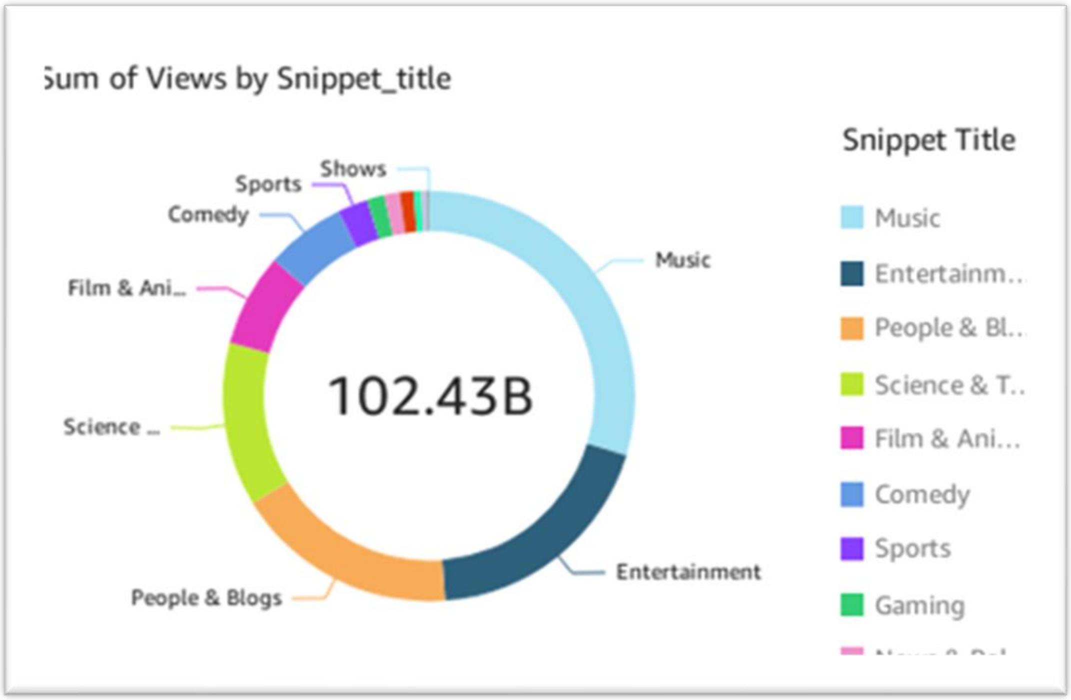


Figure: Screenshot of Visualization 4: Top Categories by Likes

This donut chart shows the distribution of total views across various video categories, with the inner label representing the cumulative view count (102.43B). The largest segments—'Music', 'Entertainment', and 'People & Blogs'—reaffirm that YouTube's audience is primarily drawn to cultural, performance, and lifestyle content. The smaller segments—such as 'Science & Technology', 'Sports', and 'Shows'—suggest specialized interest areas. This view helps platforms or content creators understand where bulk of user attention lies across video types.

DASHBOARD

The following image provide a snapshot of the Amazon QuickSight dashboard used to visualize and analyze the YouTube trending video data.

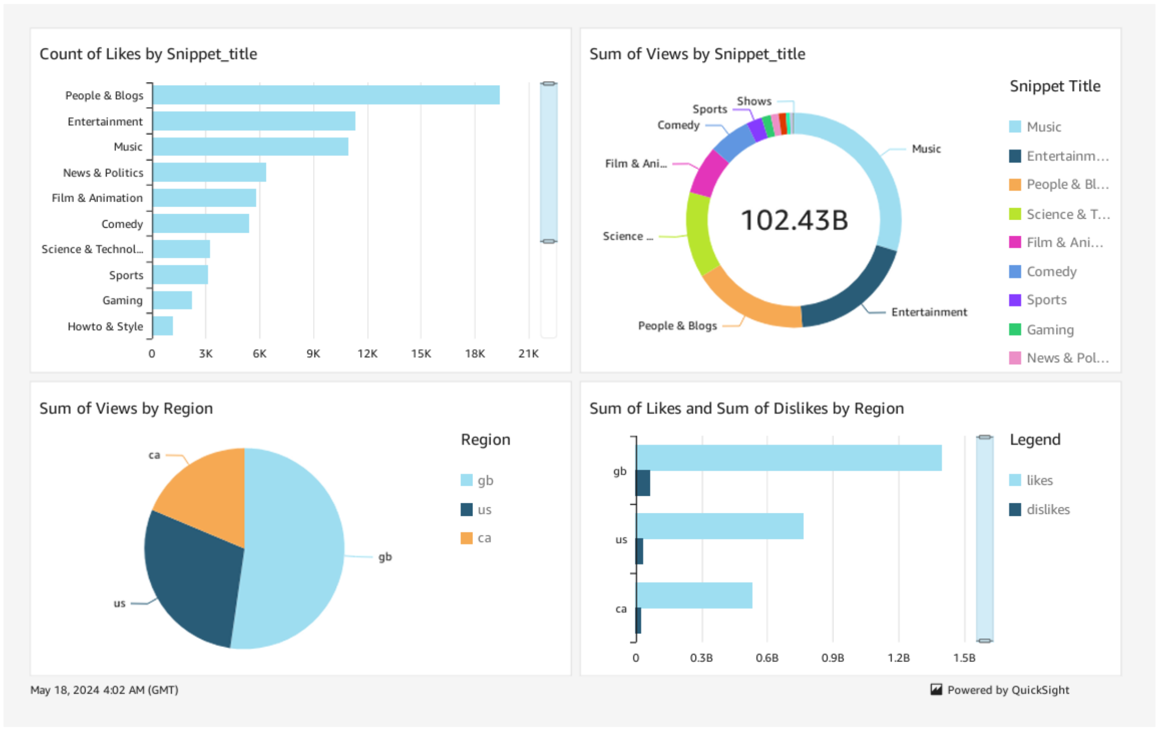


Figure 1: YouTube Analysis Dashboard (Amazon QuickSight)

The Amazon QuickSight dashboard provides a comprehensive view of the analyzed YouTube data. It includes metrics such as total views and likes by video category, likes vs dislikes by region, and region-wise engagement trends.

KEY INSIGHTS

- Music and Entertainment videos dominate global user attention across regions.
- The US region consistently outperforms others in engagement and content volume.
- Categories like How-to & Style, News & Politics, and Comedy show region-specific performance differences.
- Sentiment (likes/dislikes) offers deeper understanding of user response beyond views.

CONCLUSION

This project demonstrates the potential of building an end-to-end data analytics pipeline on the cloud using fully serverless AWS services. It processes YouTube trending video data, transforms it efficiently, and extracts valuable business insights. The solution is scalable, flexible, and cost-effective—ideal for streaming or batch-based analytics involving structured or semi-structured datasets.

FUTURE ENHANCEMENTS

- Add sentiment analysis using YouTube comments or subtitles.
- Extend the dashboard to include more countries and categories.
- Automate scheduled ETL jobs for continuous updates.
- Integrate with Amazon Redshift or OpenSearch for deeper analytics.

REFLECTION AND LEARNING

- This project allowed me to strengthen my skills in cloud-native data engineering using AWS services like Lambda, Glue, Athena, and QuickSight.
- I gained hands-on experience in managing semi-structured data, automating ETL pipelines, and creating insightful dashboards.
- Understanding user engagement trends across regions through real YouTube data provided a meaningful context to apply both technical and analytical skills.

KEY TAKEWAYS

- Designed and deployed a serverless, scalable data pipeline using AWS
- Processed and visualized 10+ regional datasets in Parquet format
- Interpreted viewer behavior across genres and regions with visual analytics
- Improved data storytelling and dashboard design with QuickSight