Project Description

ETL pipeline on Tube data using Athena, Glue and Lambda

Business Overview

Aim of the Project

This big data project aims to design and build a robust ETL (Extract, Transform, Load) pipeline using Python and various AWS services such as Athena, Lambda, and Glue. The primary objective is to ingest, process, analyze, and visualize data using cloud-native tools to gain insights from Tube trending video data. The project demonstrates how to build scalable and automated pipelines with minimal infrastructure management.

Dataset Description

The dataset used in this project is sourced from Kaggle and provides statistics on daily trending Tube videos for various regions, including Japan. Each region has its own CSV file and an accompanying JSON file that maps video category_id to its corresponding category name. The dataset includes the following fields:

- video_id
- trending_date
- title
- channel_title
- publish_time
- tags
- views
- likes
- dislikes
- comment_count
- category_id
- description

The dataset allows us to analyze viewer engagement trends, content popularity, and category performance over time.

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

Tech Stack

- **Programming Languages**: SQL, Python3
- **AWS Services**:
    - **Amazon S3**: Storage for raw and processed data
    - **AWS Glue**: Data cataloging, transformation, and ETL
    - **AWS Lambda**: Event-driven compute for file processing
    - **AWS Athena**: Serverless querying of data in S3
    - **AWS QuickSight**: Visualization and reporting
    - **AWS IAM**: Access control and permission management

Key Concepts and Components

Data Pipeline Overview

A data pipeline is a structured sequence of processes that move data from source to destination while transforming it along the way. In this project, the pipeline ingests raw video data from S3, processes and enriches it with AWS Glue and Lambda, and stores the results in optimized formats like Parquet for querying via Athena and visualization through QuickSight.

Amazon S3

S3 serves as the backbone for data storage in this project. It is used to:

- Store raw CSV and JSON files
- Store cleaned and transformed Parquet files
- Organize data in a multi-zone lake: raw, clean, and analytics layers

AWS IAM

IAM is used to securely manage access to AWS resources. Roles and policies are defined for Lambda functions, Glue jobs, and Athena queries to ensure least-privilege access.

AWS QuickSight

QuickSight is used to build dashboards and visualizations on top of the analytical data stored in S3. It provides interactive charts and graphs based on Athena queries and enables data-driven decision-making.

AWS Glue

Glue is the primary tool for ETL tasks in this project. Its components include:

- **Glue Crawlers**: Automatically detect schema and create table definitions in the Glue Catalog
- **Glue Studio**: Visual interface for building and running Spark-based ETL jobs
- **Glue Jobs**: Custom Python/Spark scripts to clean, transform, and join data

AWS Lambda

Lambda automates data transformation tasks. It listens for new file uploads to S3 and triggers processing functions that convert JSON to Parquet and update the Glue Catalog.

AWS Athena

Athena is used to run SQL queries directly on Parquet files stored in S3. It supports schema-on-read, allowing exploration without loading data into a traditional database.

Learning Takeaways

From this project:

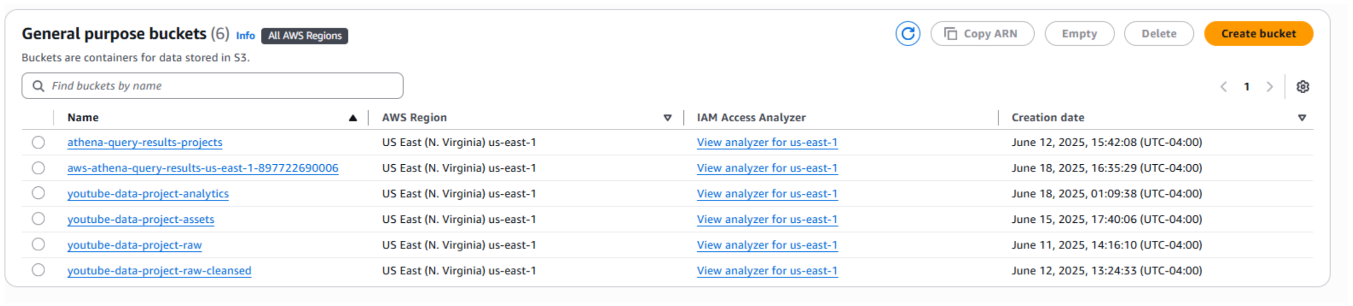
- Learn to build a secure and scalable Data Lake using Amazon S3
- Automate ETL processes using AWS Glue and Lambda
- Understand how to query and analyze data using Athena and SparkSQL
- Design real-time data workflows triggered by S3 events
- Create professional dashboards using Amazon QuickSight

Step-by-Step Solution Approach

Creating AWS S3 Bucket

Create an S3 bucket using best practices and upload:

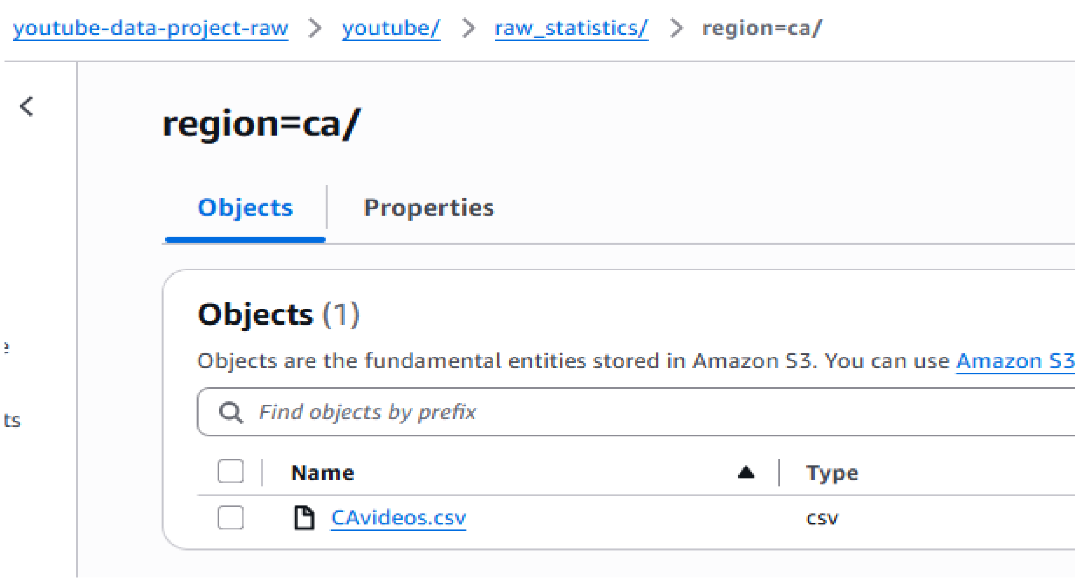
- CSV files containing trending video statistics
- JSON file containing category mappings Organize data into a raw zone structure.



General purpose buckets (6) Info All AWS Regions

Buckets are containers for data stored in S3.

Name	AWS Region	IAM Access Analyzer	Creation date
athena-query-results-projects	US East (N. Virginia) us-east-1	View analyzer for us-east-1	June 12, 2025, 15:42:08 (UTC-04:00)
aws-athena-query-results-us-east-1-897722690006	US East (N. Virginia) us-east-1	View analyzer for us-east-1	June 18, 2025, 16:35:29 (UTC-04:00)
youtube-data-project-analytics	US East (N. Virginia) us-east-1	View analyzer for us-east-1	June 18, 2025, 01:09:38 (UTC-04:00)
youtube-data-project-assets	US East (N. Virginia) us-east-1	View analyzer for us-east-1	June 15, 2025, 17:40:06 (UTC-04:00)
youtube-data-project-raw	US East (N. Virginia) us-east-1	View analyzer for us-east-1	June 11, 2025, 14:16:10 (UTC-04:00)
youtube-data-project-raw-cleansed	US East (N. Virginia) us-east-1	View analyzer for us-east-1	June 12, 2025, 13:24:33 (UTC-04:00)



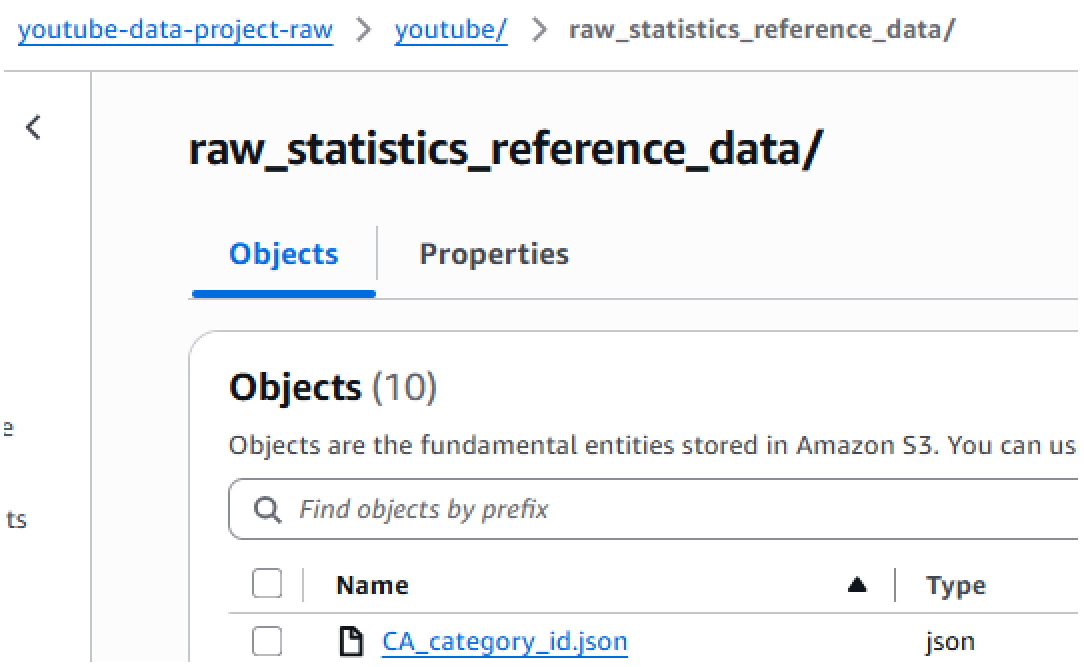
youtube-data-project-raw > youtube/ > raw_statistics/ > region=ca/

region=ca/

Objects (1)

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3

Name	Type
CAvideos.csv	csv


youtube-data-project-raw > youtube/ > raw_statistics_reference_data/

raw_statistics_reference_data/

Objects (10)

Objects are the fundamental entities stored in Amazon S3. You can us

Name	Type
CA_category_id.json	json

Creating AWS Glue Catalog and Crawler

1. Create a database in AWS Glue.
2. Set up a crawler to scan the raw S3 bucket and detect schema.
3. Automatically populate Glue Catalog with table definitions.

AWS Lambda Integration

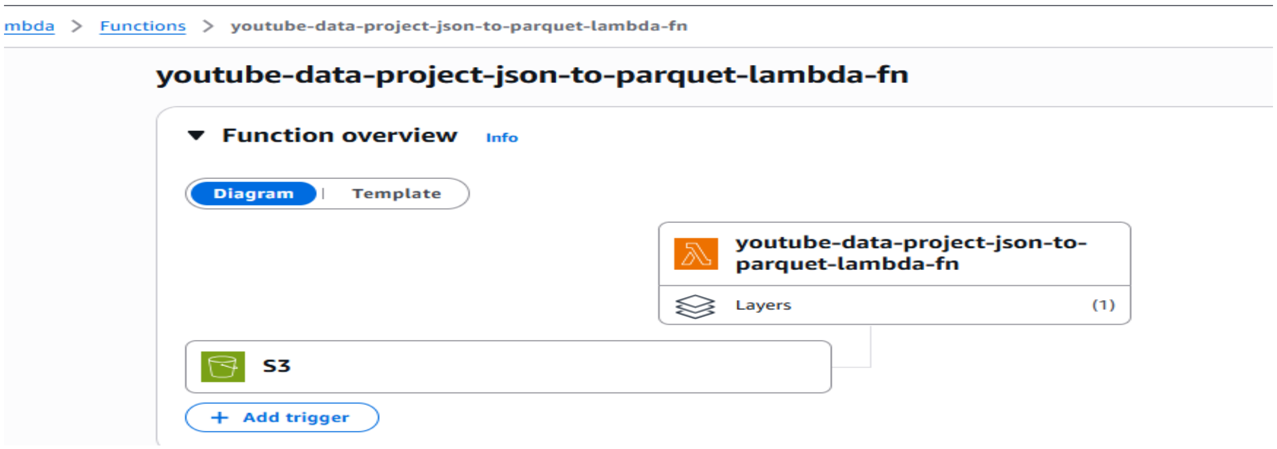
This ETL (Extract, Transform, and Load) process mainly involves using an AWS Lambda function to extract the JSON data from the raw S3 bucket, converting it into Parquet format, pushing that data using AWS Data Wrangler into the clean S3 bucket, and automatically catalog it into the Glue Catalog, where we can query it using Athena SQL statements.

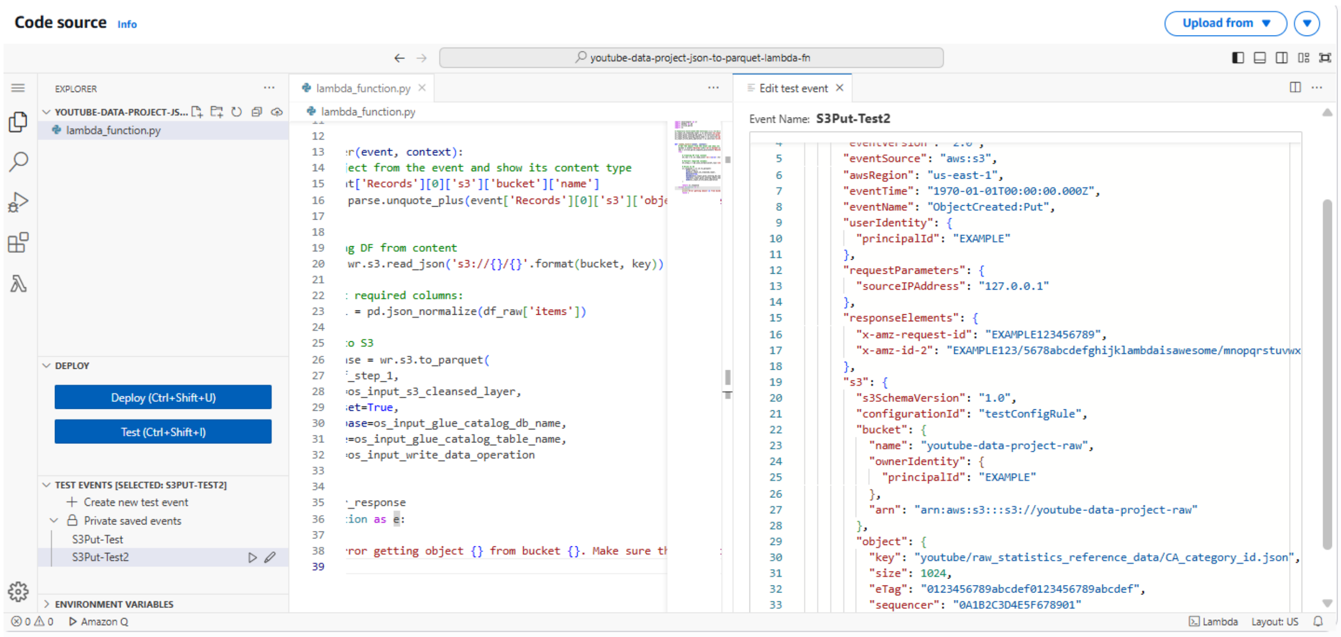
We will use a data wrangler to write the data frame to Parquet in S3 while simultaneously interacting with the Glue Catalog for efficient data cataloging and management. This step also involves creating a new S3 bucket for storing clean and optimized data obtained from the Lambda function, ensuring efficient data organization

Next, it's time to create a new Lambda function that interacts with the S3 bucket and Data Catalog using Data Wrangler, allowing data extraction and manipulation. Configured a test event to simulate and verify the Lambda function's behavior when triggered by an S3 event.

Further enhance Lambda function by incorporating AWSDataWrangler, and creating a database in Glue Catalog will allow us to organize and store the cleansed data extracted by the Lambda function, making it easier to query and analyze. We will also gain insight into visualizing the data in a columnar format (Parquet) using Athena, running SQL queries, and exploring the data further

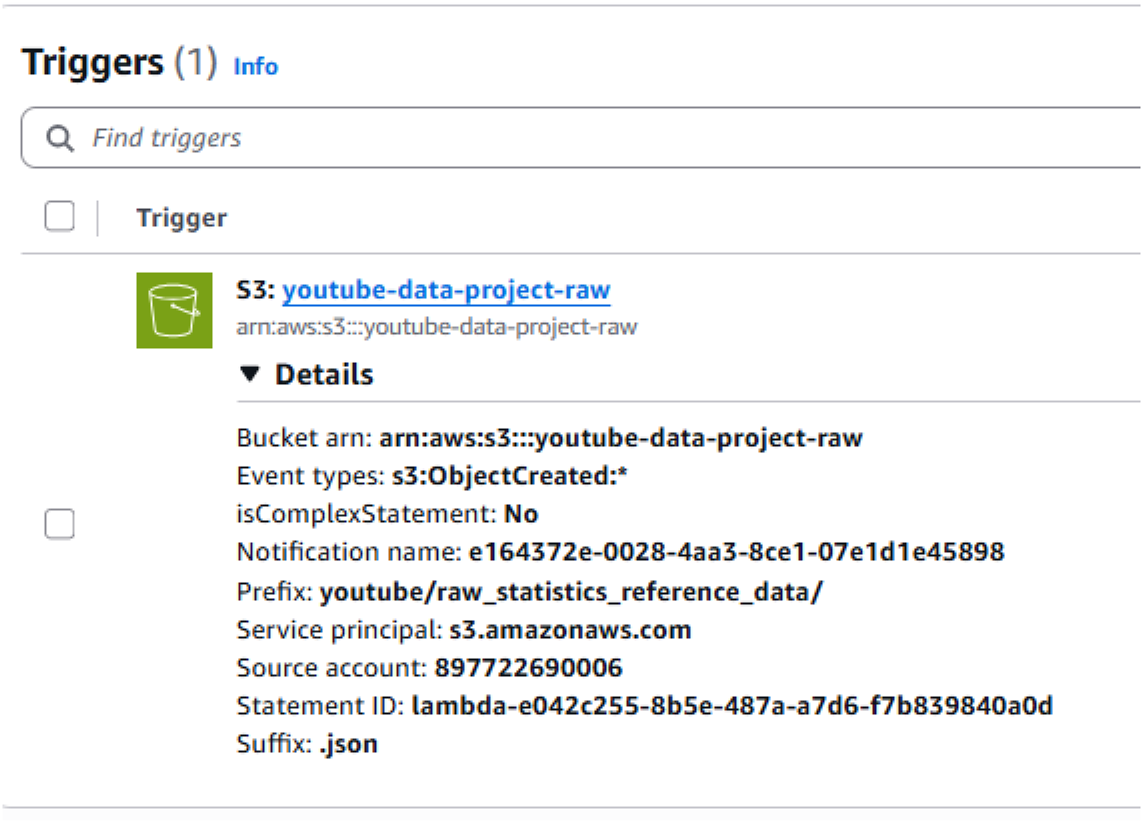
Setting up an S3 crawler allows to easily automate extracting raw_statistics data from the S3 bucket and populating a table in the Glue Catalog. This step also demonstrates how to transform CSV files within the Glue Catalog table into Parquet format data using Glue Studio, optimizing the data for efficiency and querying.


mbda > Functions > youtube-data-project-json-to-parquet-lambda-fn
youtube-data-project-json-to-parquet-lambda-fn
Function overview Info
Diagram
Template
youtube-data-project-json-to-parquet-lambda-fn
Layers (1)
S3
+ Add trigger



```
:r(event, context):
ject from the event and show its content type
ıt['Records'][0]['s3']['bucket']['name']
 parse.unquote_plus(event['Records'][0]['s3']['obje


ıg DF from content
 wr.s3.read_json('s3://{}/{}'.format(bucket, key))

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

:o S3
ıse = wr.s3.to_parquet(
:_step_1,
:os_input_s3_cleansed_layer,
:et=True,
ıase=os_input_glue_catalog_db_name,
:=os_input_glue_catalog_table_name,
:os_input_write_data_operation


:_response
:ion as e:

:ror getting object {} from bucket {}. Make sure th
```

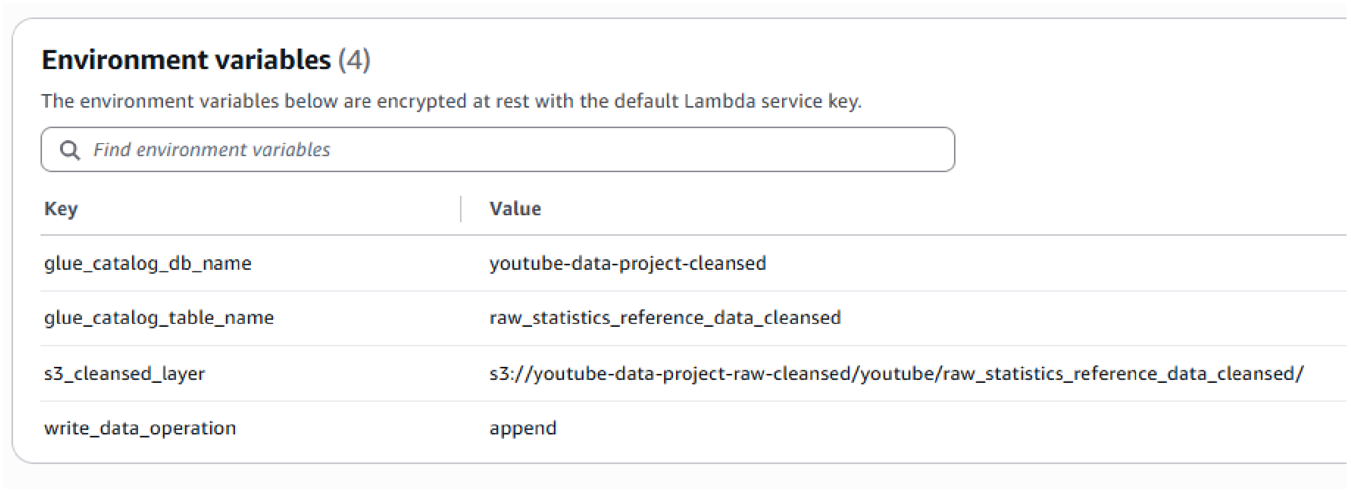
```
"eventSource": "aws:s3",
"awsRegion": "us-east-1",
"eventTime": "1970-01-01T00:00:00.000Z",
"eventName": "ObjectCreated:Put",
"userIdentity": {
  "principalId": "EXAMPLE"
},
"requestParameters": {
  "sourceIPAddress": "127.0.0.1"
},
"responseElements": {
  "x-amz-request-id": "EXAMPLE123456789",
  "x-amz-id-2": "EXAMPLE123/5678abcdefghijklambdaisawesome/mnopqrstuvwx
},
"s3": {
  "s3SchemaVersion": "1.0",
  "configurationId": "testConfigRule",
  "bucket": {
    "name": "youtube-data-project-raw",
    "ownerIdentity": {
      "principalId": "EXAMPLE"
    },
    "arn": "arn:aws:s3:::s3://youtube-data-project-raw"
  },
  "object": {
    "key": "youtube/raw_statistics_reference_data/CA_category_id.json",
    "size": 1024,
    "eTag": "0123456789abcdef0123456789abcdef",
    "sequencer": "0A1B2C3D4E5F678901"
```



S3: youtube-data-project-raw
arn:aws:s3:::youtube-data-project-raw

▼ Details

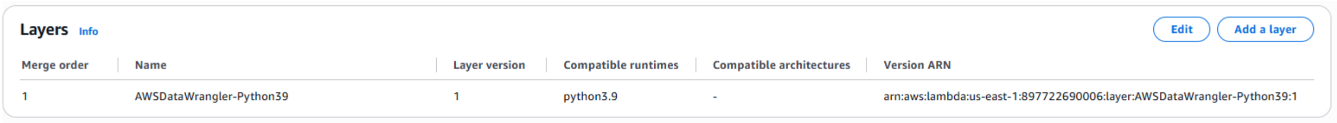
Bucket arn: **arn:aws:s3:::youtube-data-project-raw**
Event types: **s3:ObjectCreated:***
isComplexStatement: **No**
Notification name: **e164372e-0028-4aa3-8ce1-07e1d1e45898**
Prefix: **youtube/raw_statistics_reference_data/**
Service principal: **s3.amazonaws.com**
Source account: **897722690006**
Statement ID: **lambda-e042c255-8b5e-487a-a7d6-f7b839840a0d**
Suffix: **.json**



Environment variables (4)

The environment variables below are encrypted at rest with the default Lambda service key.

Key	Value
glue_catalog_db_name	youtube-data-project-cleansed
glue_catalog_table_name	raw_statistics_reference_data_cleansed
s3_cleansed_layer	s3://youtube-data-project-raw-cleansed/youtube/raw_statistics_reference_data_cleansed/
write_data_operation	append



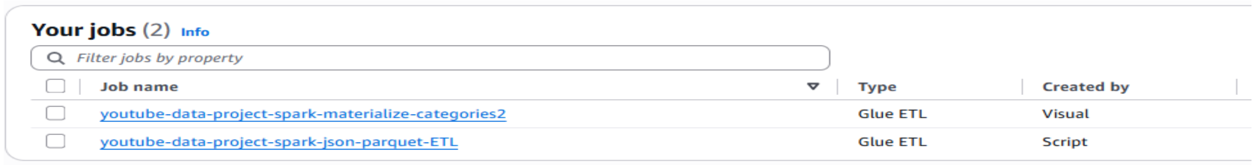
Merge order	Name	Layer version	Compatible runtimes	Compatible architectures	Version ARN
1	AWSDataWrangler-Python39	1	python3.9	-	arn:aws:lambda:us-east-1:897722690006:layer:AWSDataWrangler-Python39:1

AWS Glue ETL Jobs

This step will allow us to leverage AWS Glue's capabilities, particularly the Glue Spark ETL jobs feature, to process and transform data efficiently. will be able to gain insights into creating a Spark job within AWS Glue, understanding how to transform JSON data to Parquet format, and configuring job properties to suit specific requirements.

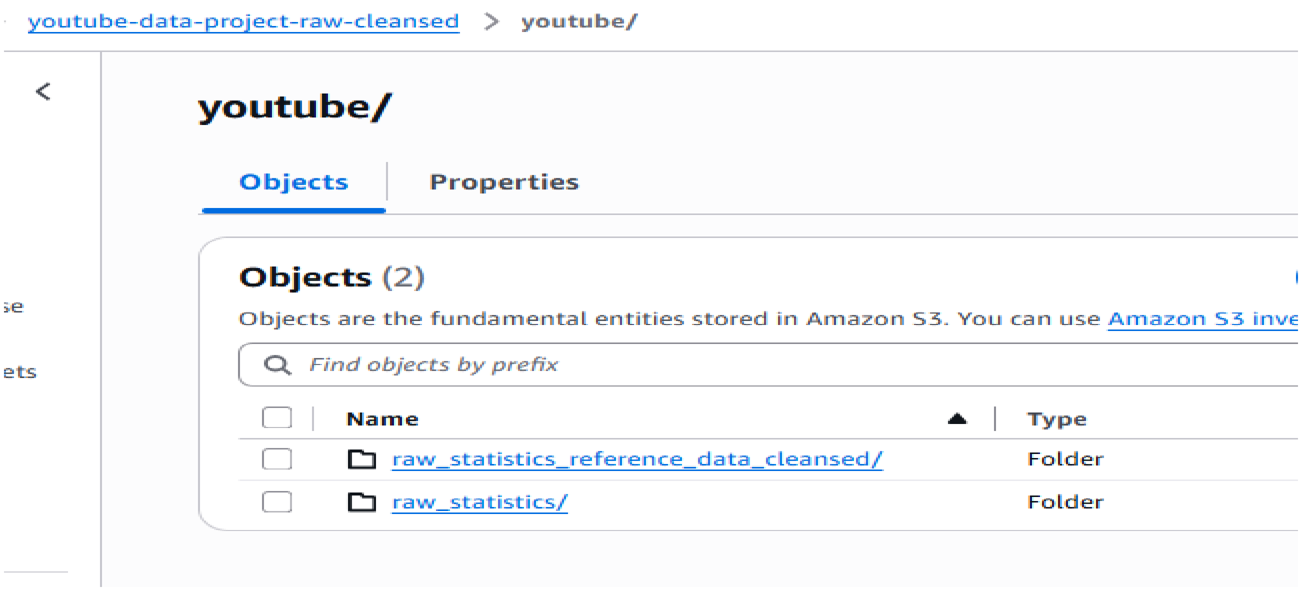
Running the Glue Spark job will allow us to process and transform the data imported and stored in the cleansed S3 bucket. This step helps us understand how to leverage the power of the Spark data processing engine. The new S3 crawler will scan the specified path and partitions in the S3 bucket, extracting data and creating a new table in the Glue Catalog, which will help automate the process of discovering, cataloging, and extracting data.

Once have successfully appended the data from the CSV file and JSON files, it's time to add an S3 trigger to the Lambda function generated earlier. must add an S3 trigger to r Lambda function to automate any further processing of new incoming JSON files, such as video category data, etc., and push that data further into the cleansed layer. can check the performance of the Lambda function using CloudWatch metrics. Now, it's time to create the final layer in S3, i.e., the Analytics/Reporting Layer, that will further help to visualize the data using QuickSight.



Your jobs (2) Info

Job name	Type	Created by
youtube-data-project-spark-materialize-categories2	Glue ETL	Visual
youtube-data-project-spark-json-parquet-ETL	Glue ETL	Script



youtube-data-project-raw-cleansed > youtube/

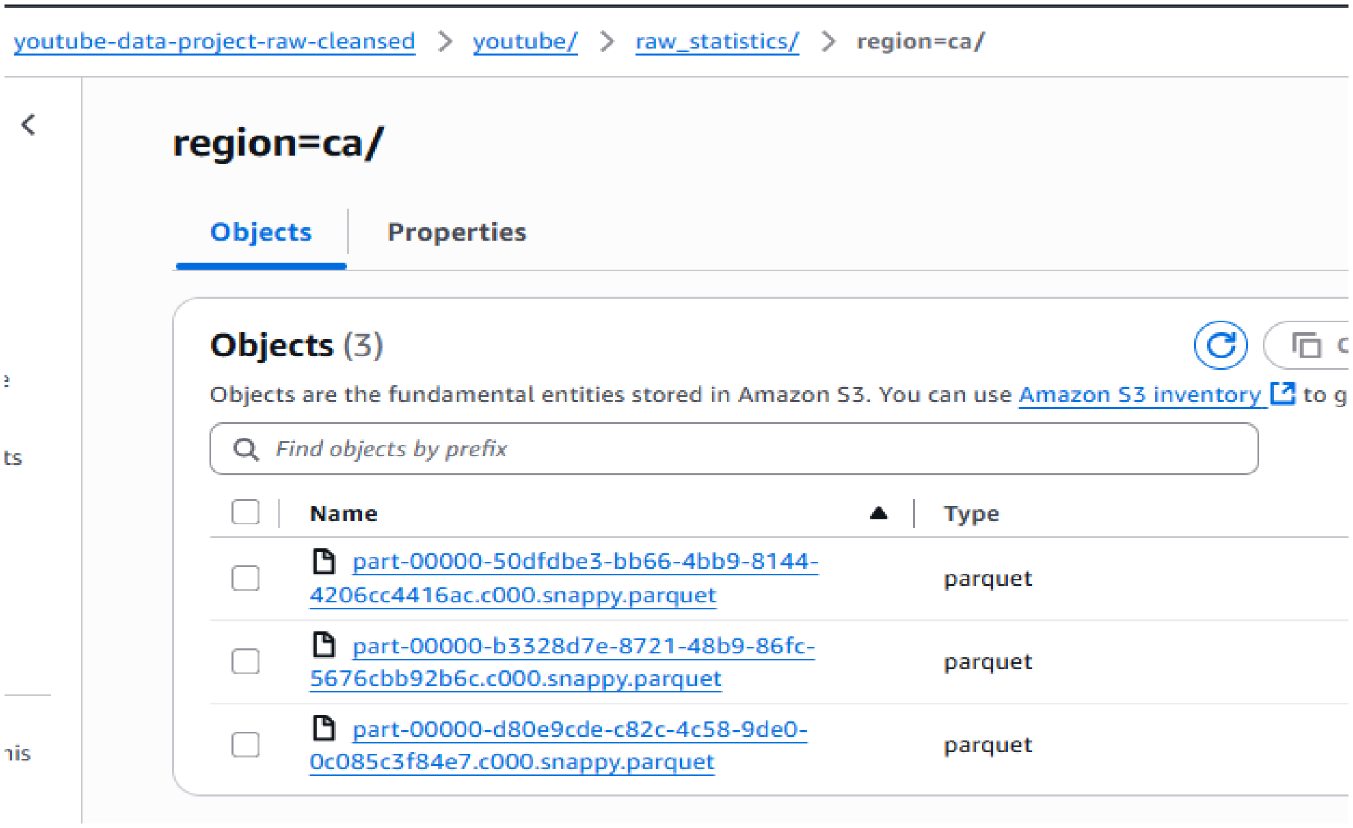
youtube/

Objects (2)

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inve

Name	Type
raw_statistics_reference_data_cleansed/	Folder
raw_statistics/	Folder

CSV to Parquet


youtube-data-project-raw-cleansed > youtube/ > raw_statistics/ > region=ca/

region=ca/

Objects | Properties

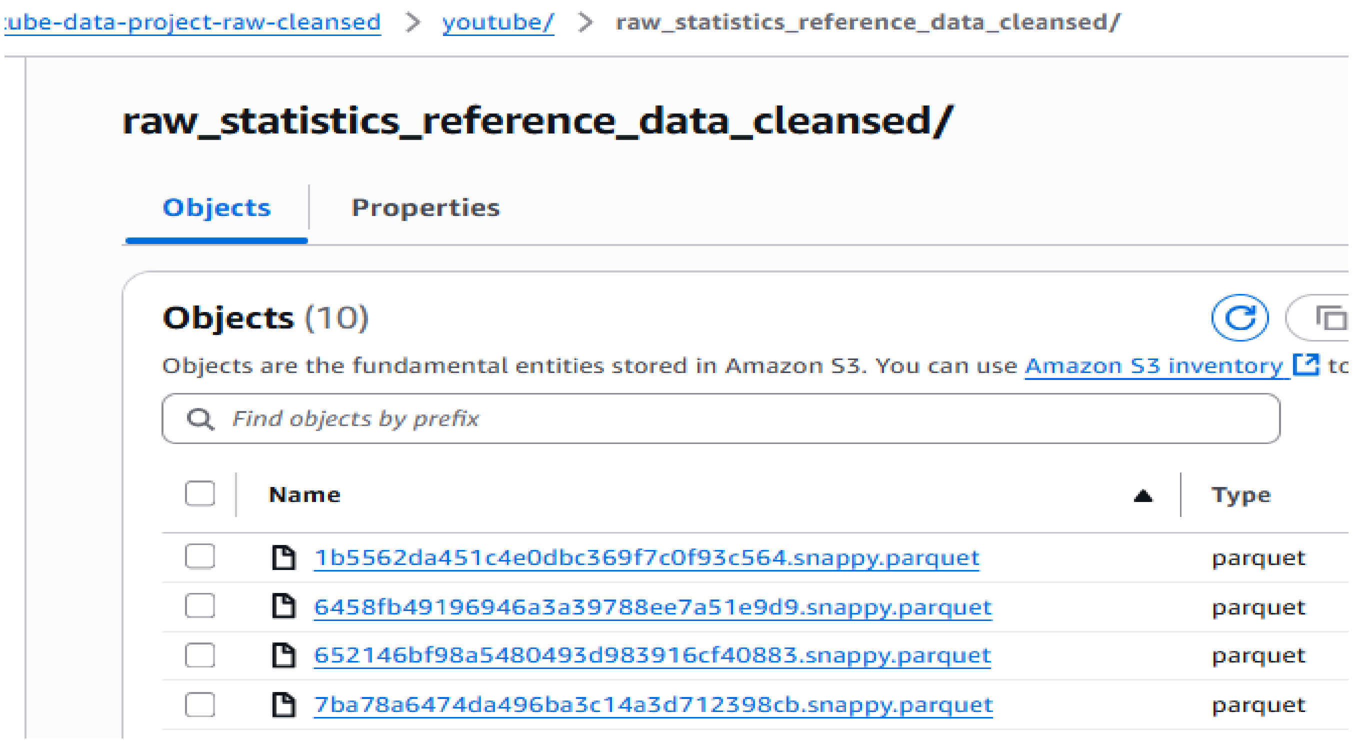
Objects (3)

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to g

Find objects by prefix

Name	Type
part-00000-50dfdbe3-bb66-4bb9-8144-4206cc4416ac.c000.snappy.parquet	parquet
part-00000-b3328d7e-8721-48b9-86fc-5676cbb92b6c.c000.snappy.parquet	parquet
part-00000-d80e9cde-c82c-4c58-9de0-0c085c3f84e7.c000.snappy.parquet	parquet

JSON to Parquet


ube-data-project-raw-cleansed > youtube/ > raw_statistics_reference_data_cleansed/

raw_statistics_reference_data_cleansed/

Objects | Properties

Objects (10)

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to

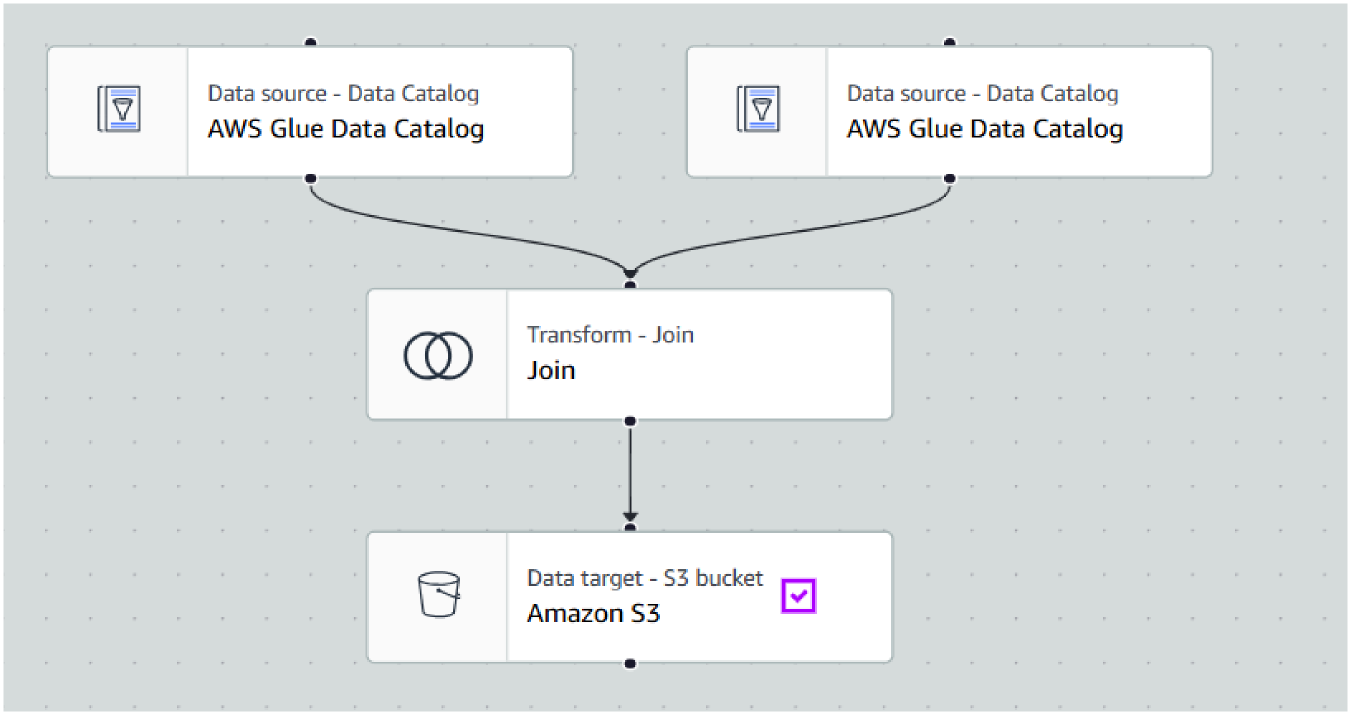
Find objects by prefix

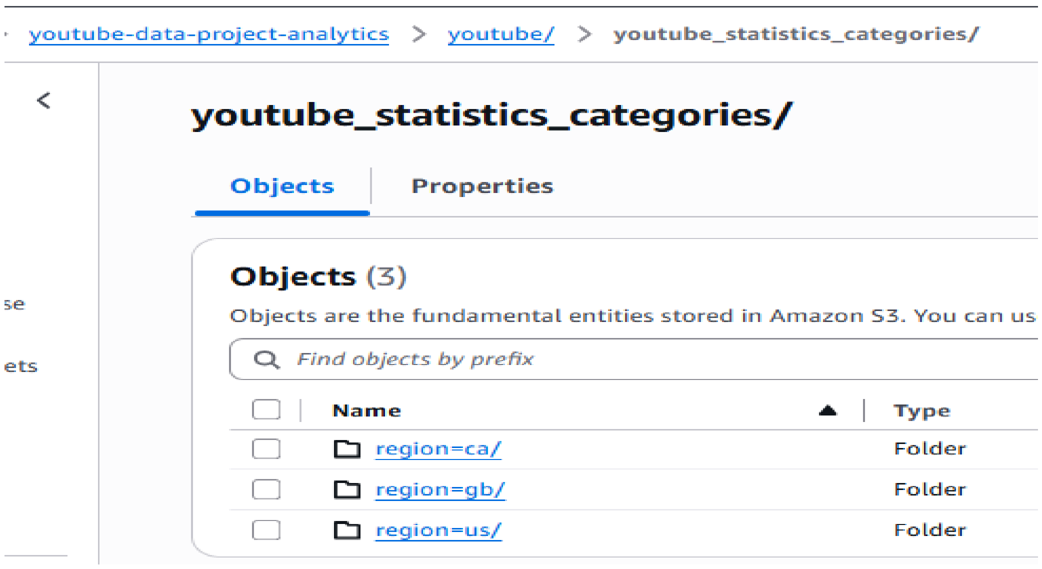
Name	Type
1b5562da451c4e0dbc369f7c0f93c564.snappy.parquet	parquet
6458fb49196946a3a39788ee7a51e9d9.snappy.parquet	parquet
652146bf98a5480493d983916cf40883.snappy.parquet	parquet
7ba78a6474da496ba3c14a3d712398cb.snappy.parquet	parquet

Data Materialization with Glue Studio

The analytics layer aims to provide a materialized view of the cleansed data along with the results of the SQL join query performed earlier to the business users, saving their time and optimizing the cost and performance of the overall business solution.

This step involves creating a new S3 bucket and configuring IAM policies for the necessary privileges. Additionally, I created a new database (youtube-data-project-analytics) and a Glue Studio job (youtube-data-project-spark-materialize-categories2). Performed a join operation on Parquet format datasets and stored the output in an S3 bucket, which can be viewed using Athena SQL statements. Finally, will leverage this view to visualize the data using AWS QuickSight, gaining insights and creating reports.





youtube-data-project-analytics > youtube/ > youtube_statistics_categories/

youtube_statistics_categories/

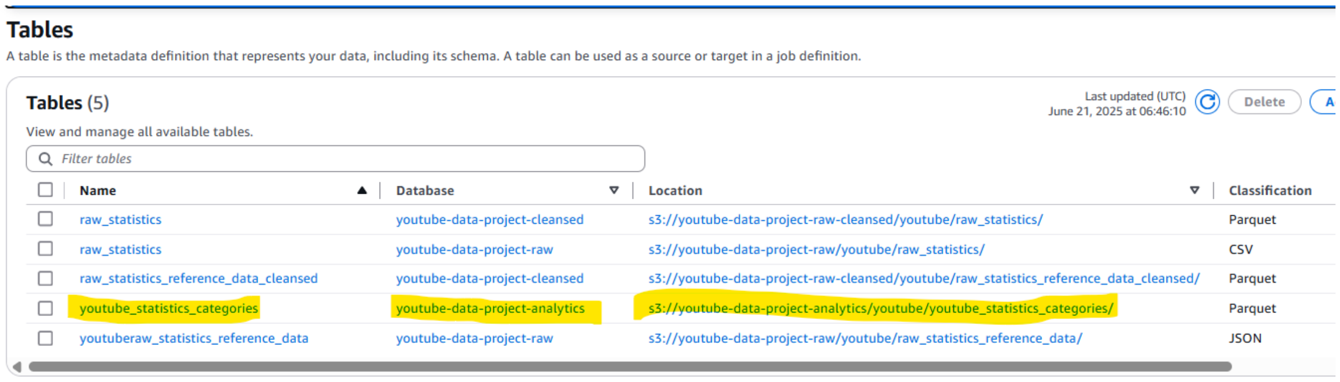
Objects | Properties

Objects (3)

Objects are the fundamental entities stored in Amazon S3. You can us

Find objects by prefix

Name	Type
region=ca/	Folder
region=gb/	Folder
region=us/	Folder



Tables

A table is the metadata definition that represents your data, including its schema. A table can be used as a source or target in a job definition.

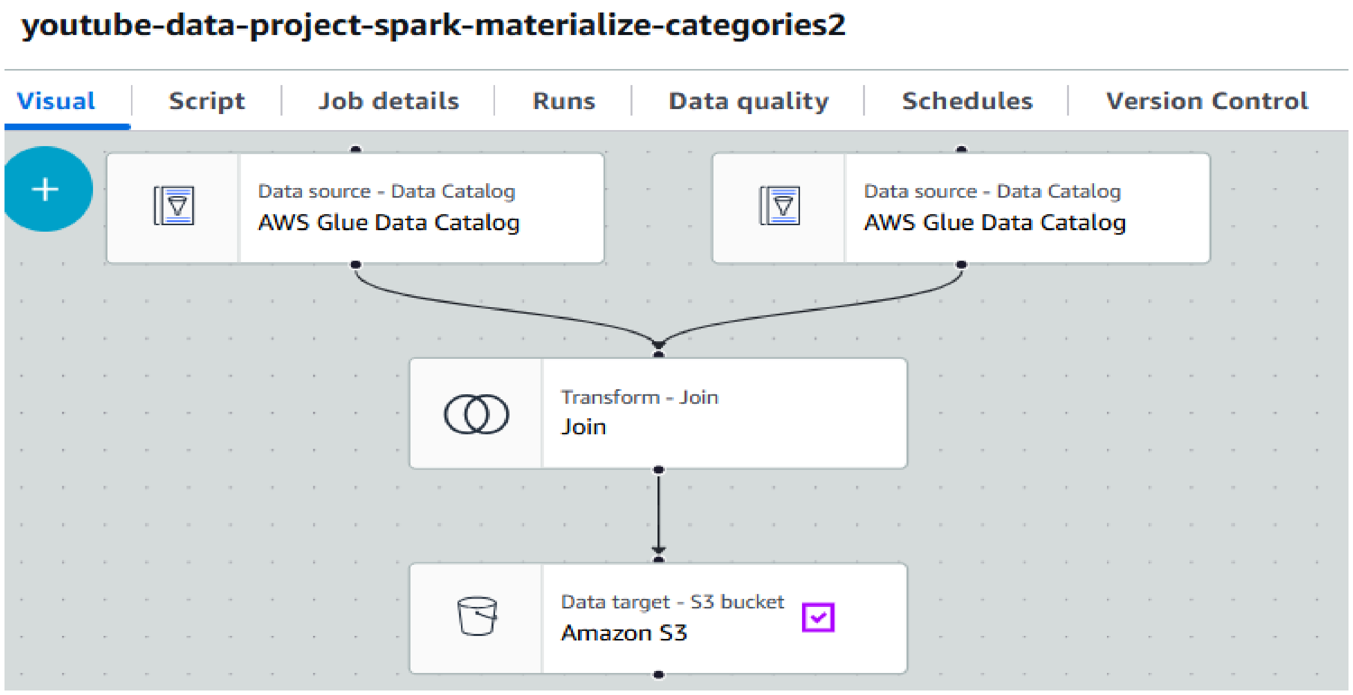
Tables (5)

View and manage all available tables.

Last updated (UTC) June 21, 2025 at 06:46:10

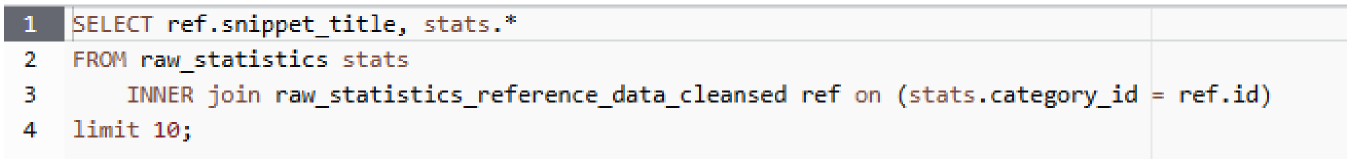
Filter tables

Name	Database	Location	Classification
raw_statistics	youtube-data-project-cleansed	s3://youtube-data-project-raw-cleansed/youtube/raw_statistics/	Parquet
raw_statistics	youtube-data-project-raw	s3://youtube-data-project-raw/youtube/raw_statistics/	CSV
raw_statistics_reference_data_cleansed	youtube-data-project-cleansed	s3://youtube-data-project-raw-cleansed/youtube/raw_statistics_reference_data_cleansed/	Parquet
youtube_statistics_categories	youtube-data-project-analytics	s3://youtube-data-project-analytics/youtube/youtube_statistics_categories/	Parquet
youtuberaw_statistics_reference_data	youtube-data-project-raw	s3://youtube-data-project-raw/youtube/raw_statistics_reference_data/	JSON



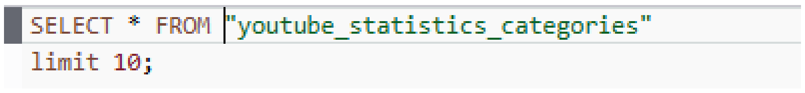
youtube-data-project-spark-materialize-categories2

Using Cleansed Layer



```
SELECT ref.snippet_title, stats.*
FROM raw_statistics stats
    INNER join raw_statistics_reference_data_cleansed ref on (stats.category_id = ref.id)
limit 10;
```

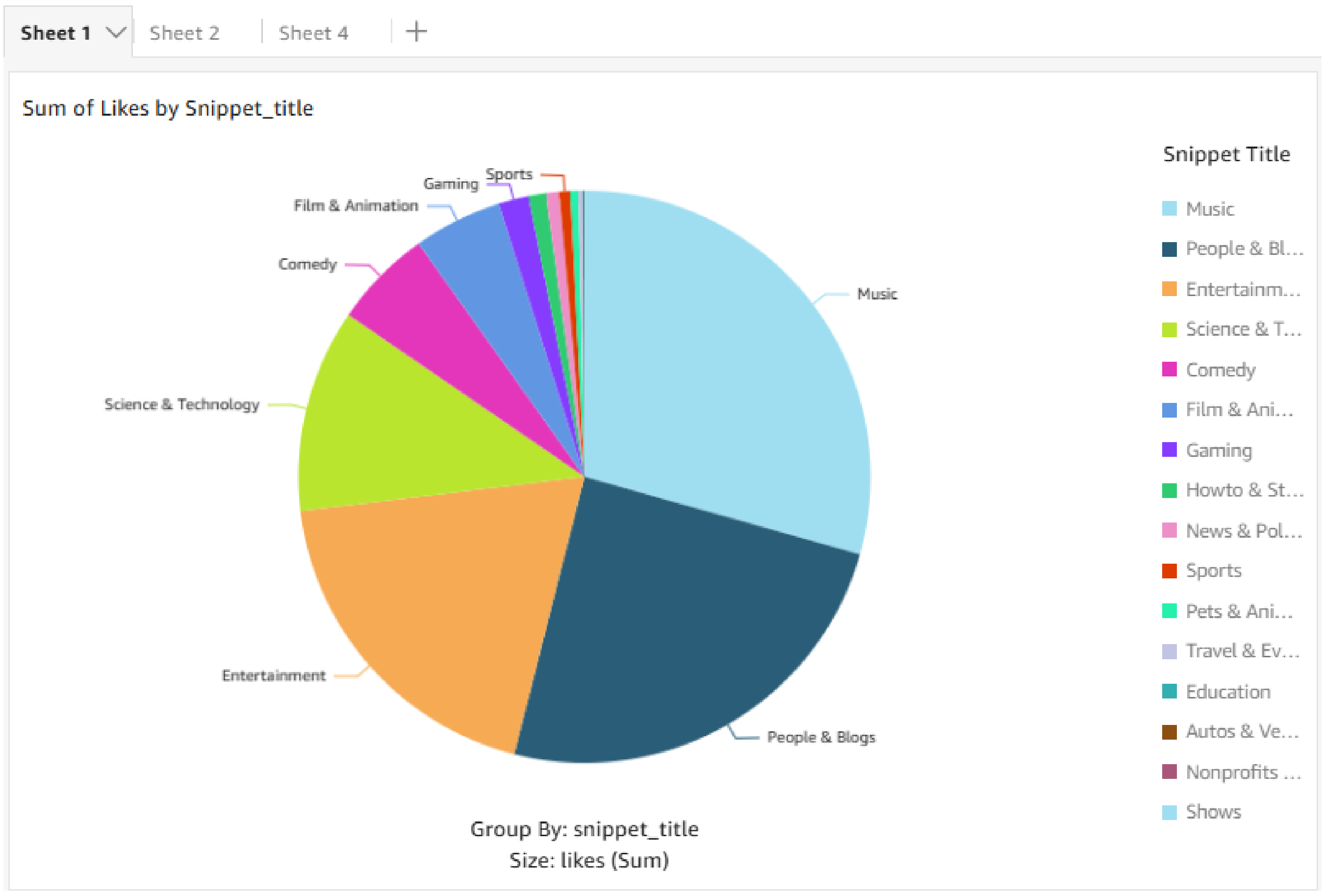
Using Reporting Layer

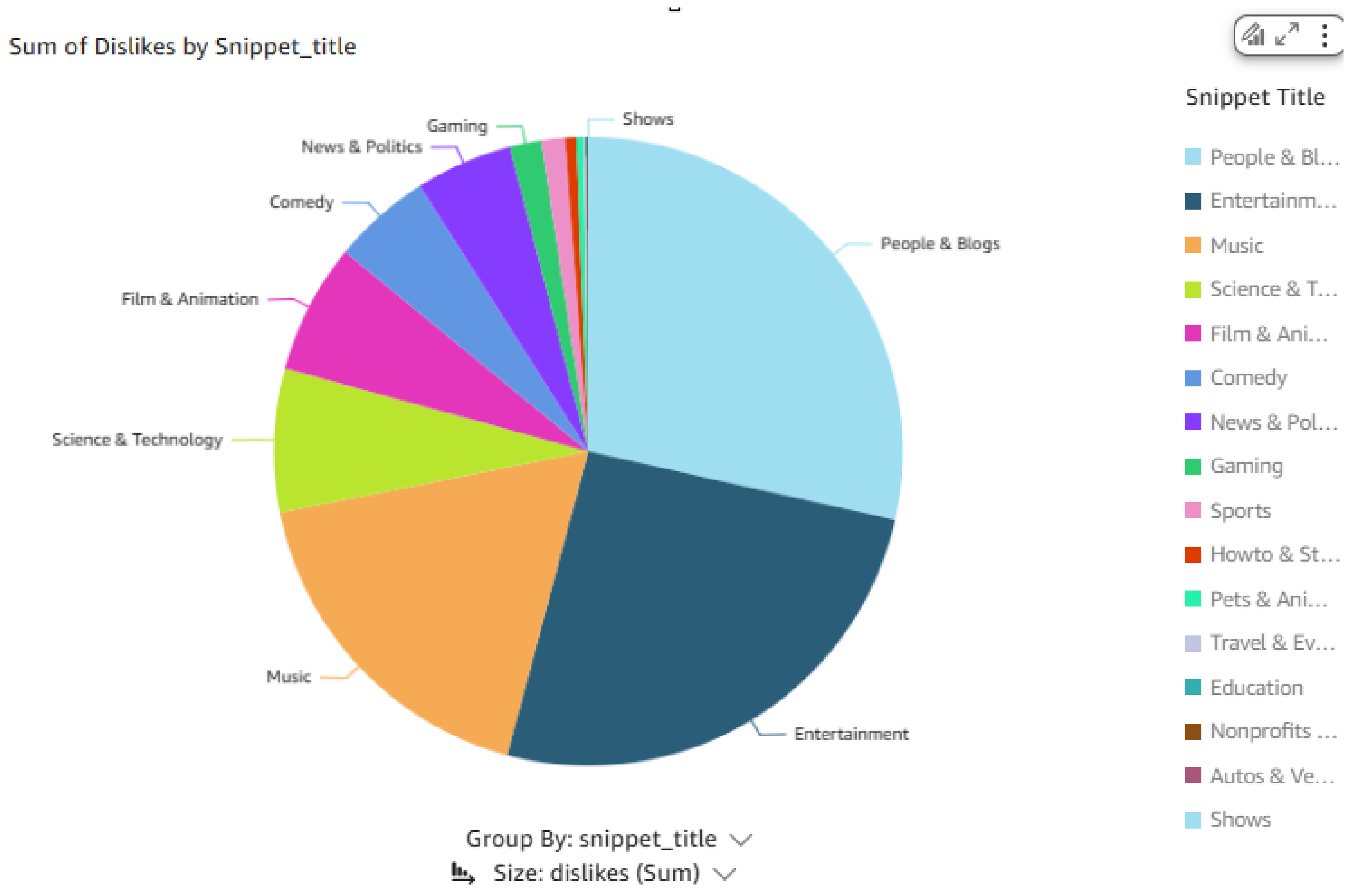


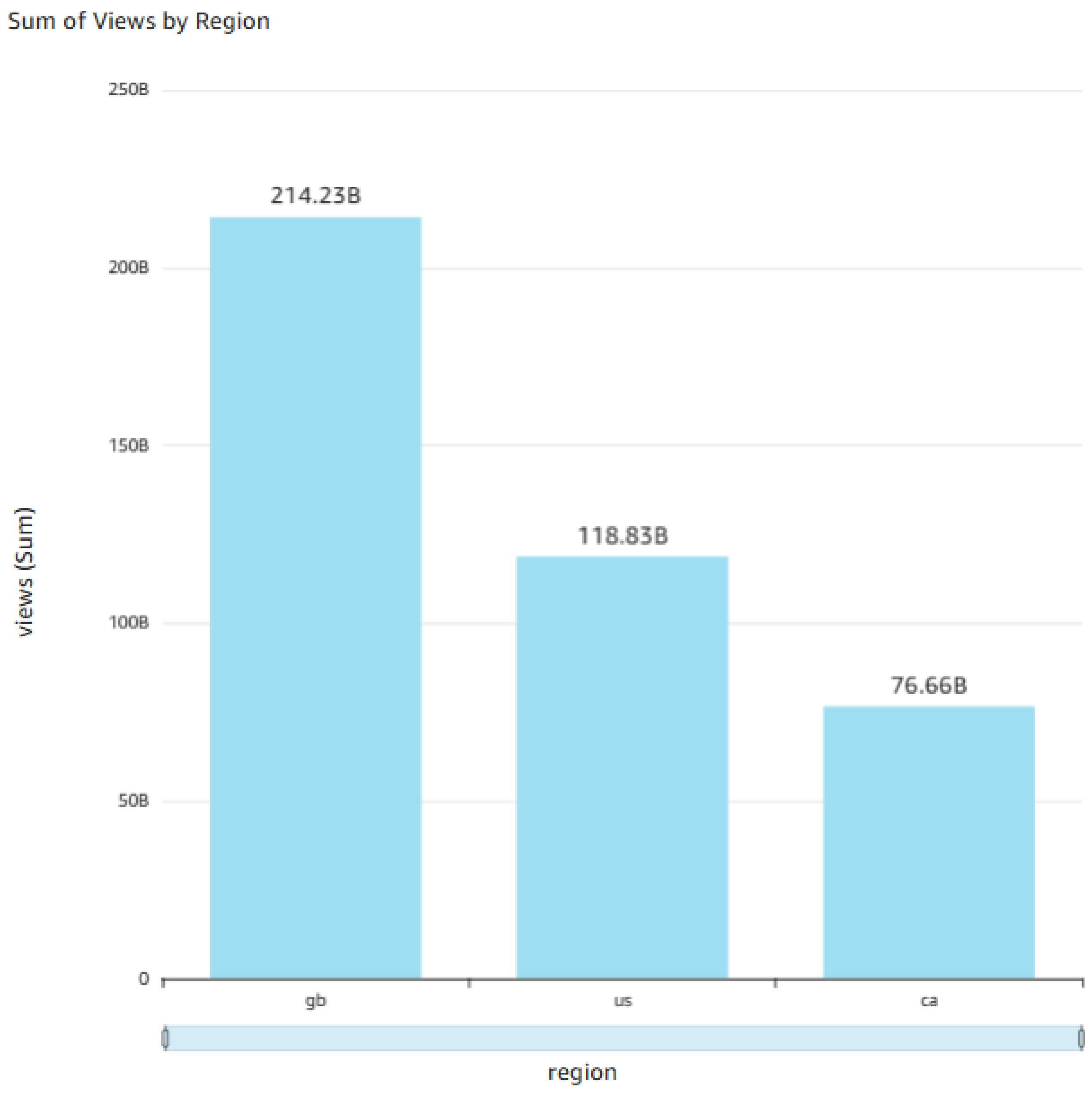
```
SELECT * FROM "youtube_statistics_categories"
limit 10;
```

Visualization with QuickSight

Connect QuickSight to Athena as a data source.
Import datasets from tube-data-project-analytics > tube_statistics_categories




Sum of Dislikes by Snippet_title
Shows
Gaming
News & Politics
Comedy
Film & Animation
Science & Technology
Music
Entertainment
People & Blogs
Snippet Title
People & Bl…
Entertainm…
Music
Science & T…
Film & Ani…
Comedy
News & Pol…
Gaming
Sports
Howto & St…
Pets & Ani…
Travel & Ev…
Education
Nonprofits …
Autos & Ve…
Shows
Group By: snippet_title
Size: dislikes (Sum)



Conclusion

This end-to-end AWS ETL pipeline project using Python showcases the power of cloud native services in managing, transforming, and visualizing large datasets. It serves as a solid foundation for anyone aiming to understand real-world data engineering and analytics on AWS.