Youtube Data Archiving and Purging Pipeline

Problem Statement:

Set up a data archiving and purging pipeline using Azure Data Factory to move and organize data, and Azure Databricks to analyze and archive historical data based on specified criteria

Project Overview:

This project leverages the "YouTube Trending Video Dataset (updated daily)" dataset from kaggle to build a data archiving and purging pipeline using Azure Data Factory. Azure Data Factory orchestrates the workflow, automating data ingestion, filtration, and storage processes. Filtered data is categorized into two storage containers: archived data for analysis and non-compliant data for purging. Azure Databricks is leveraged for analyzing and visualizing the archived data, enabling insights into video engagement metrics and trends. This solution demonstrates the seamless integration of Azure tools for scalable, automated data management and advanced analytics.

YouTube Trending Video Dataset Overview:

This dataset provides a detailed snapshot of trending YouTube videos from multiple regions, capturing metadata from daily video trends. It includes published features such as video titles, channel names, publish dates, and popularity metrics (views, likes, dislikes, and comments). The dataset spans diverse content categories, offering valuable insights into audience preferences, engagement trends, and content performance. With its rich metadata and multi-dimensional coverage, this dataset is ideal for analyzing patterns in video virality, regional preferences, and engagement dynamics.

Data Description (Json file)

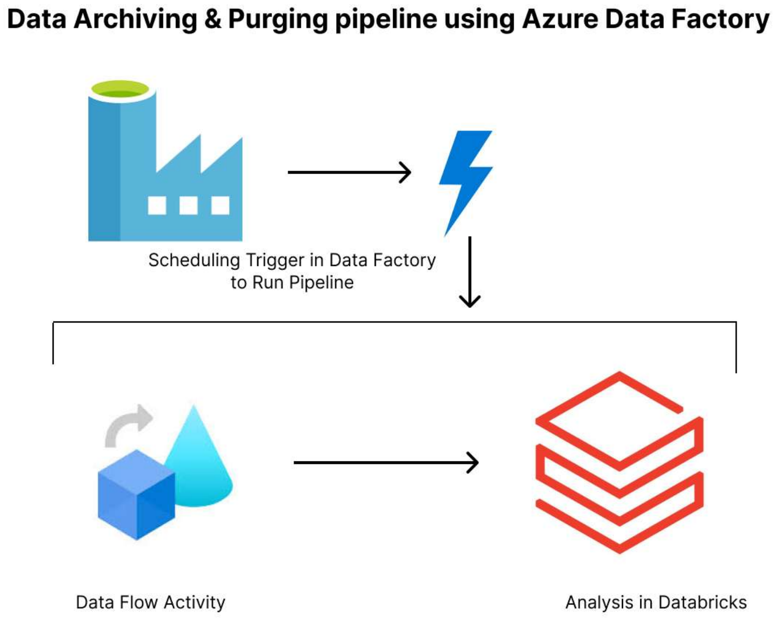
The JSON file represents YouTube video categories, containing metadata such as the category ID (id), title (snippet.title), and whether it is assignable to videos (snippet.assignable). It also includes unique identifiers like etag for version control and channelId linking categories to specific

YouTube channels. The data is organized hierarchically under an items array, with each item providing detailed information about a category

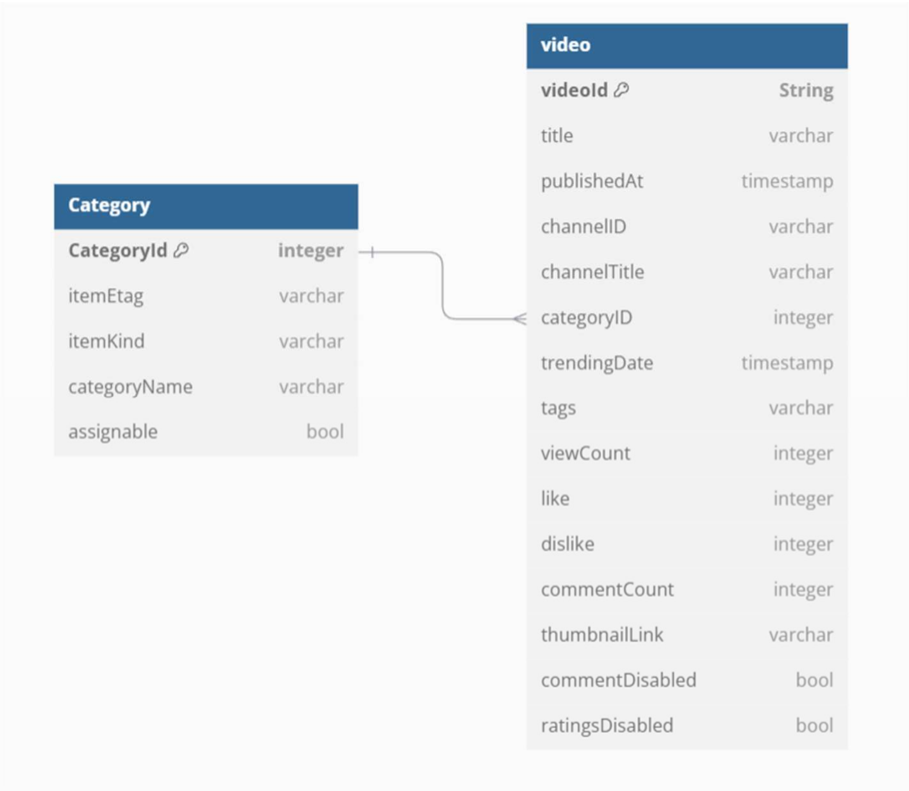
Data Description(Csv file)

1. video_id: A unique identifier for each video on YouTube.
2. title: The title of the video.
3. publishedAt: The date when the video was published on YouTube.
4. channelId: A unique identifier for the YouTube channel that published the video.
5. channelTitle: The name of the YouTube channel.
6. categoryId: Numeric identifier representing the video category.
7. trending_date: The date when the video was trending.
8. tags: List of tags associated with the video for categorization and search optimization.
9. view_count: Total number of views the video has received.
10. likes: Total number of likes the video has received.
11. dislikes: Total number of dislikes the video has received.
12. comment_count: Total number of comments on the video.
13. thumbnail_link: URL of the thumbnail image associated with the video.
14. comments_disabled: Boolean value indicating whether comments are disabled for the video.
15. ratings_disabled: Boolean value indicating whether ratings (likes/dislikes) are disabled for the video.

Architecture Diagram


Data Archiving & Purging pipeline using Azure Data Factory
Scheduling Trigger in Data Factory to Run Pipeline
Data Flow Activity
Analysis in Databricks

ER Diagram



Category	
CategoryId	integer
itemEtag	varchar
itemKind	varchar
categoryName	varchar
assignable	bool

video	
videoId	String
title	varchar
publishedAt	timestamp
channelID	varchar
channelTitle	varchar
categoryID	integer
trendingDate	timestamp
tags	varchar
viewCount	integer
like	integer
dislike	integer
commentCount	integer
thumbnailLink	varchar
commentDisabled	bool
ratingsDisabled	bool

Execution Overview

This project focuses on developing a data processing and analysis pipeline that integrates Azure Data Factory for orchestration and Azure Databricks for data preprocessing, filtration, and visualization. The pipeline categorizes the YouTube Trending Video dataset into archival and purging containers and visualizes insights from the archived data.

1. Data Preparation

1.1. Data Ingestion:

The YouTube Trending Video dataset is ingested and stored in an **Azure Data Lake Storage Gen2** container (Bronze Layer). Azure Data Factory automates the movement of this raw data to Azure Databricks, where it is mounted for seamless access.

1.2. Data Filtration:

Three Filtration techniques are applied:

1. Views Threshold: Videos with views greater than 378000 are considered.
2. Publish Time Frame: Videos getting trended within 3 days.
3. Engagement Metric: Like-to-dislike ratio above 0.04129.

1.3. Data Routing:

- **Compliant Data**: Moved to the **Archive Container** in the Gold Layer.
- **Non-compliant Data**: Moved to the **Purging Container** for deletion or future analysis.

2. Data Analysis and Visualization

The compliant data from the Archive Container is loaded into Azure Databricks for advanced analysis and visualization.

- Engagement metrics such as views, likes, and comments are analyzed.

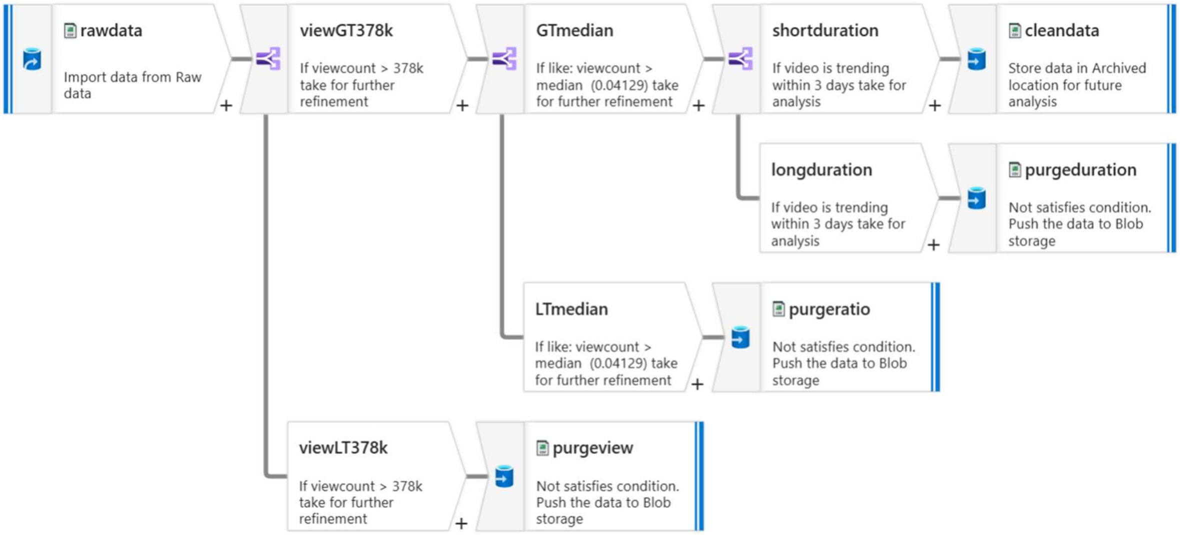
- Insights such as trends in video categories and regional audience preferences are visualized using Databricks notebooks.

Dataflow Pipeline:

Dataflow has two important components. They are Data source and Data sink. Data store represents where the data comes from and data store represents where the data is stored after performing the transformations in the data.

The data flow process in ADF involves the following steps:

1. **Source Data**: Data is ingested from various sources.
2. **Data Integration**: Data flows are created using mapping data flows or pipelines.
3. **Transformations**: Data can be transformed using built-in transformations. This is done in a no-code environment leveraging Azure's compute power.
4. **Sink Data**: After transformation, the data is written to the target destinations (sinks).
5. **Monitoring and Debugging**: The flow is monitored via ADF’s monitoring tools.


rawdata
Import data from Raw data
viewGT378k
If viewcount > 378k take for further refinement
GTmedian
If like: viewcount > median (0.04129) take for further refinement
shortduration
If video is trending within 3 days take for analysis
cleandata
Store data in Archived location for future analysis
longduration
If video is trending within 3 days take for analysis
purgeduration
Not satisfies condition. Push the data to Blob storage
LTmedian
If like: viewcount > median (0.04129) take for further refinement
purgeratio
Not satisfies condition. Push the data to Blob storage
viewLT378k
If viewcount > 378k take for further refinement
purgeview
Not satisfies condition. Push the data to Blob storage

In this project initially the data undergoes the data flow followed by data analysis in databricks. First step is performing the data flow activity. Raw data is ingested from sources where the data storage cost is high. Later the data is moved into an archived or purging container where the storage cost is less. And then the data analysis is performed on the archived data. Using certain conditions the data is undergoing conditional splits.

Splitting conditions:

- Viewcount greater than 25% of total viewcount
- likes to viewcount ratio must be greater than median
- video should get trending within 4 days

If the data satisfies the provided condition it moves for further refinement. If it fails to satisfy then it is moved to the purging container. Finally the data which satisfies all the conditions is stored in an archived container to perform the data analysis.

After the data flow activity succeeded, the data analysis part will start. It will run the databricks notebook and provide the necessary visualization. ADF is scheduled to run every Friday at 5 PM to analyse the youtube trending every week.

Azure Resources Used for this Project:

- Azure Data Lake Storage
- Azure Data Factory
- Azure Databricks
- Azure Blob Storage

Project Requirements:

The requirements for this project are broken down into six different parts which are

Data flow requirements

- Need data source to get input data
- Need data sink to store archived and purged data

1. Data Ingestion Requirements

- Ingest all files into Azure data lake.
- Ingested data must have the same schema applied.
- Ingested data must be stored in CSV or JSON format.
- We must be able to analyze the ingested data via SQL.
- Ingestion Logic must be able to handle the incremental load.

2. Data Conditional split Requirements

- Each record's view count should be greater than 25% or quartile 1 of total.
- Like count and view count ratio should be greater than its median.
- The number of days it took to get trending should be less than 4 days.
- If it satisfies the three conditions, store it in an archived container.
- The data which failed to satisfy is pushed to the purging data sink.

3. Data format Requirements

- Create dataframes from CSV and JSON files.

- Explode the branching JSON into linear format.
- Select the required columns from the exploded JSON dataframe.
- Perform join operation

4. Data Analysis Requirements

- Total Views per category
- Most commented videos
- Publishing trend over time
- Top channel by total views
- Visualize the Outputs.

5. Scheduling Requirements

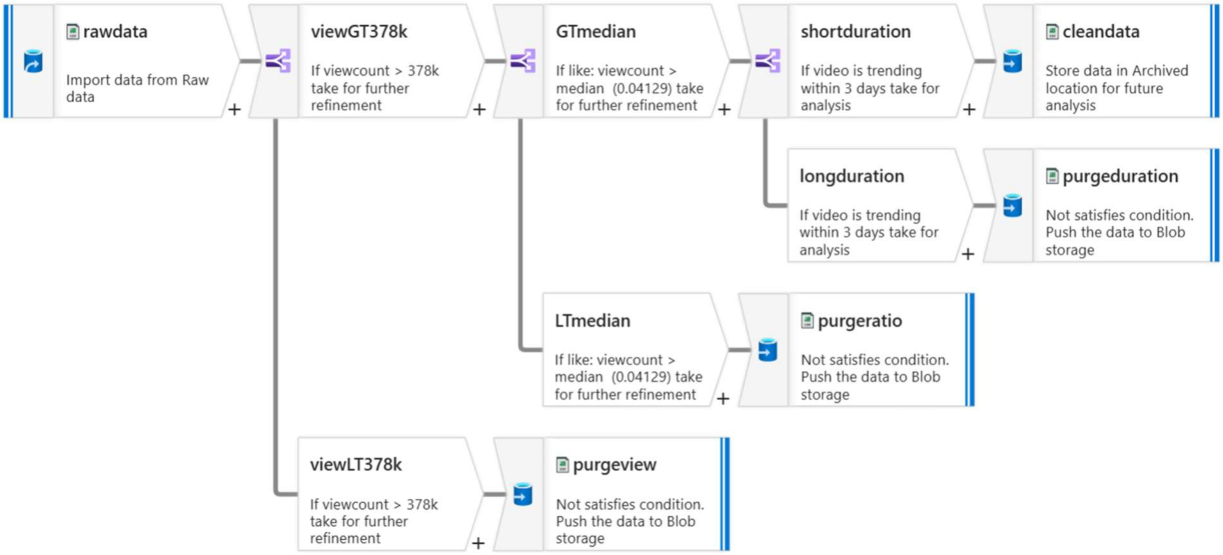
- Scheduled to run every Friday at 5 pm.
- Ability to monitor pipelines.
- Ability to rerun failed pipelines.
- Ability to set up alerts on failures

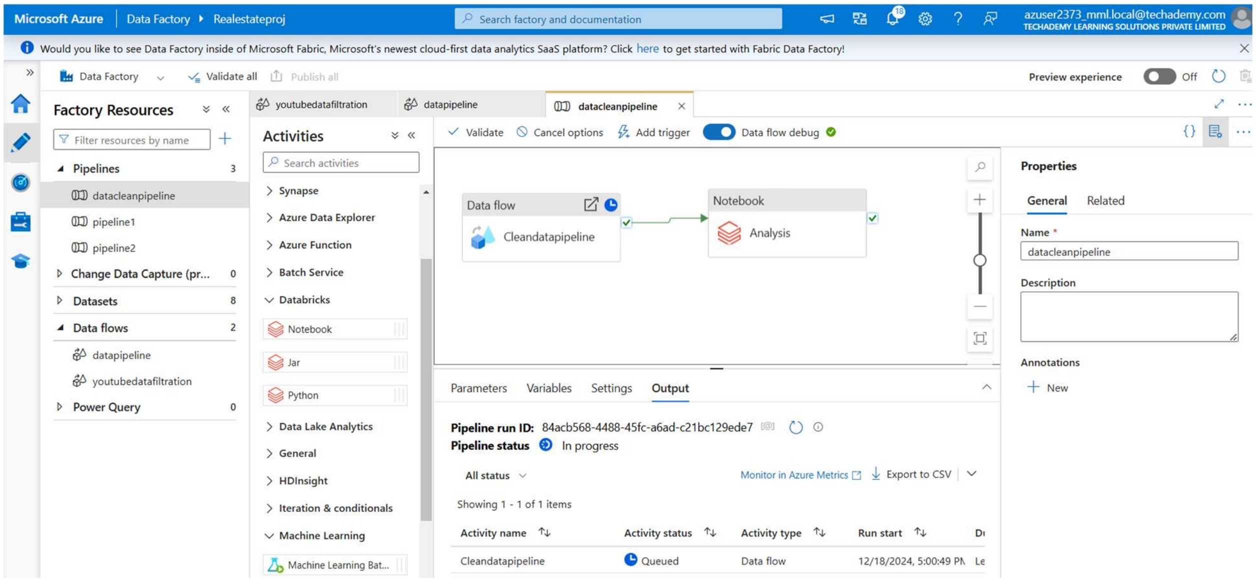
6. Additional Requirements

- Finding threshold values by performing operation on MS-Excel

Results & Analysis:

Data Flow and Pipeline Snapshots


rawdata
Import data from Raw data
viewGT378k
If viewcount > 378k take for further refinement
GTmedian
If like: viewcount > median (0.04129) take for further refinement
shortduration
If video is trending within 3 days take for analysis
cleandata
Store data in Archived location for future analysis
longduration
If video is trending within 3 days take for analysis
purgeduration
Not satisfies condition. Push the data to Blob storage
LTmedian
If like: viewcount > median (0.04129) take for further refinement
purgeratio
Not satisfies condition. Push the data to Blob storage
viewLT378k
If viewcount > 378k take for further refinement
purgeview
Not satisfies condition. Push the data to Blob storage



Would you like to see Data Factory inside of Microsoft Fabric, Microsoft's newest cloud-first data analytics SaaS platform? Click here to get started with Fabric Data Factory!

Name *
datacleanpipeline

Description

Pipeline run ID: 84acb568-4488-45fc-a6ad-c21bc129ede7
Pipeline status In progress

All status

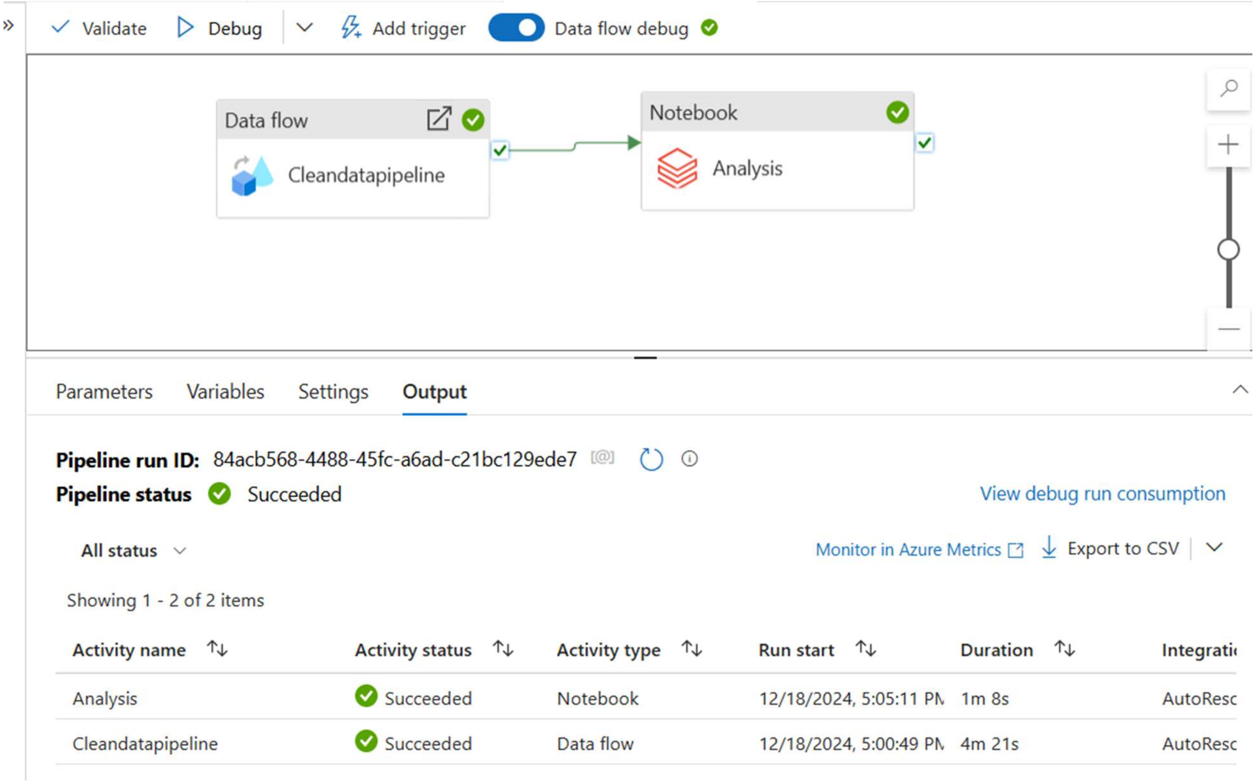
Monitor in Azure Metrics

Export to CSV

Showing 1 - 1 of 1 items

Activity name	Activity status	Activity type	Run start	Di
Cleandatapipeline	Queued	Data flow	12/18/2024, 5:00:49 PI	Le

Pipeline Orchestration Results



Pipeline run ID: 84acb568-4488-45fc-a6ad-c21bc129ede7

Pipeline status ✓ Succeeded

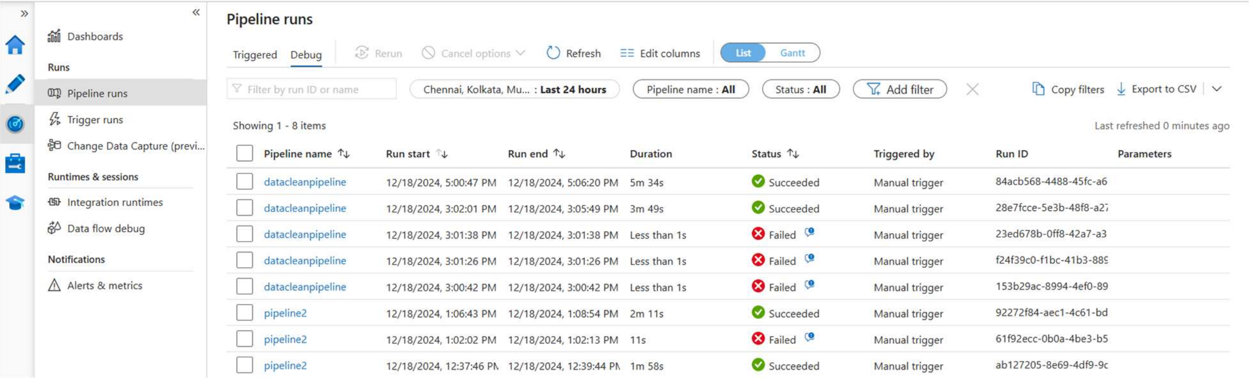
View debug run consumption

All status

Monitor in Azure Metrics | Export to CSV

Showing 1 - 2 of 2 items

Activity name	Activity status	Activity type	Run start	Duration	Integrati
Analysis	✓ Succeeded	Notebook	12/18/2024, 5:05:11 PM	1m 8s	AutoResc
Cleandatapipeline	✓ Succeeded	Data flow	12/18/2024, 5:00:49 PM	4m 21s	AutoResc



Pipeline runs

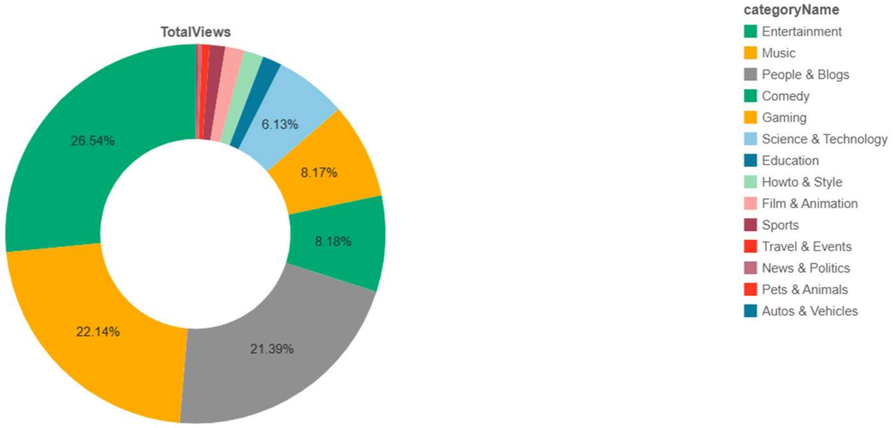
Showing 1 - 8 items

Last refreshed 0 minutes ago

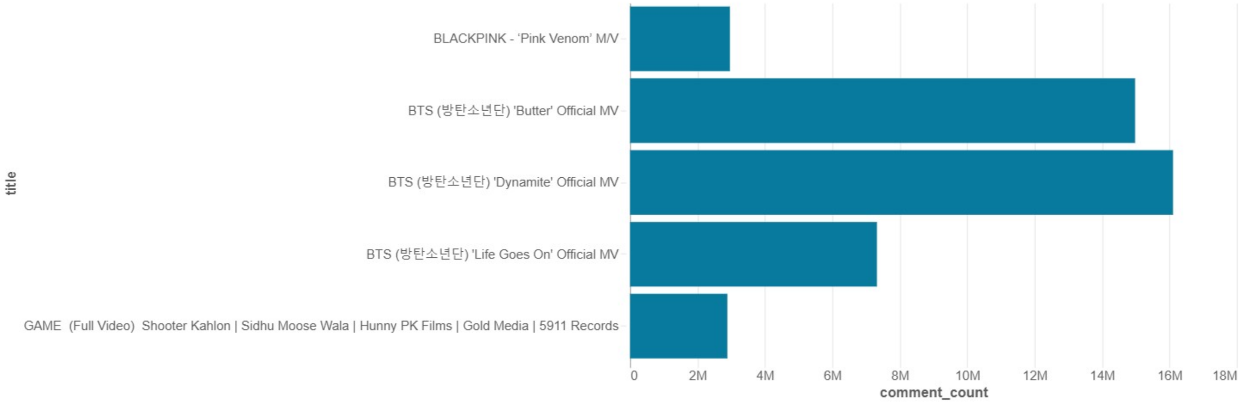
Pipeline name	Run start	Run end	Duration	Status	Triggered by	Run ID	Parameters
datacleanpipeline	12/18/2024, 5:00:47 PM	12/18/2024, 5:06:20 PM	5m 34s	✓ Succeeded	Manual trigger	84acb568-4488-45fc-a6	
datacleanpipeline	12/18/2024, 3:02:01 PM	12/18/2024, 3:05:49 PM	3m 49s	✓ Succeeded	Manual trigger	28e7fcce-5e3b-48f8-a27	
datacleanpipeline	12/18/2024, 3:01:38 PM	12/18/2024, 3:01:38 PM	Less than 1s	✗ Failed	Manual trigger	23ed678b-0ff8-42a7-a3	
datacleanpipeline	12/18/2024, 3:01:26 PM	12/18/2024, 3:01:26 PM	Less than 1s	✗ Failed	Manual trigger	f24f39c0-f1bc-41b3-889	
datacleanpipeline	12/18/2024, 3:00:42 PM	12/18/2024, 3:00:42 PM	Less than 1s	✗ Failed	Manual trigger	153b29ac-8994-4ef0-89	
pipeline2	12/18/2024, 1:06:43 PM	12/18/2024, 1:08:54 PM	2m 11s	✓ Succeeded	Manual trigger	92272f84-aec1-4c61-bd	
pipeline2	12/18/2024, 1:02:02 PM	12/18/2024, 1:02:13 PM	11s	✗ Failed	Manual trigger	61f92ecc-0b0a-4be3-b5	
pipeline2	12/18/2024, 12:37:46 PM	12/18/2024, 12:39:44 PM	1m 58s	✓ Succeeded	Manual trigger	ab127205-8e69-4df9-9c	

Archived Data Analysis Results

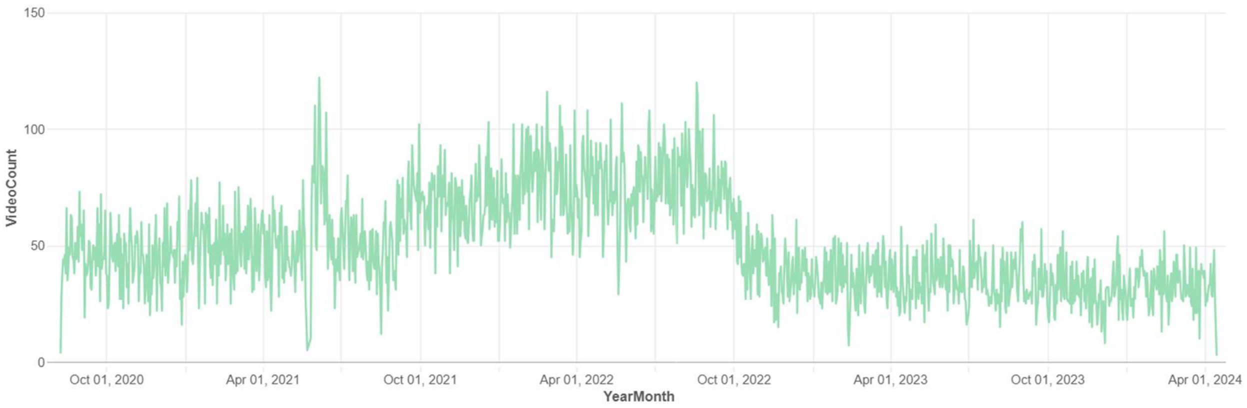
Total Views per category



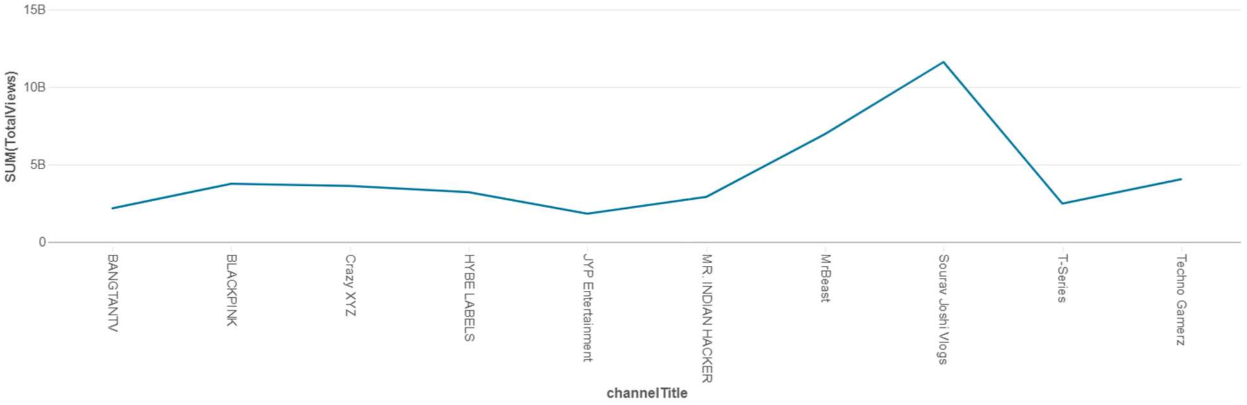
Most commented Videos



Publishing trend overtime



Top channel by Total Views



Tasks Performed:

- Designed and implemented a solution architecture for a data engineering workflow using Azure Data Factory and Azure Data Lake Storage Gen2.
- Developed conditional data processing steps for filtering and transforming raw data based on thresholds such as view counts and medians.

- Used **branching** and **logical conditions** in Azure Data Factory pipelines for data refinement and routing.
- Applied filters such as "greater than median" and "short/long duration" criteria to classify data for further analysis or archival.
- The **Views Threshold** filters videos with more than 378,000 views, chosen as 25% of the total view count to identify significant content.
- The like-to-view ratio was calculated, and its median was analyzed in MS - Excel to determine the engagement threshold.
- The **Publish Time Frame** considers videos that became trending within 3 days, calculated by finding the median difference between the published date and trending date.
- The Engagement Metric evaluates videos based on a like-to-dislike ratio above 0.04129, ensuring content with high engagement is prioritized.
- Archived refined datasets into Azure Data Lake for further analysis.
- Directed non-relevant data to **Azure Blob Storage** for efficient purging and cost management.
- Managed complex data flows with **pipeline activities** like "Copy Data," "Filter," and "Storage Operations."
- Configured **data retention and purge mechanisms** to manage data efficiently.
- Automated and monitored data workflows to ensure efficient data processing and transformation.
- The filtered and archived data is imported into **Azure Databricks** for advanced analysis and visualization.
- The analysis focuses on metrics like views, engagement, and trending timelines to uncover valuable insights.

- These visualizations support data-driven decision-making and enhance understanding of the archived content's performance.

Spark (Only PySpark and SQL)

- Spark architecture, Data Sources API, and Dataframe API.
- PySpark - Ingested CSV into the data lake as parquet files/ tables.
- PySpark - Transformations such as Filter, Simple Aggregations, GroupBy, Window functions etc.
- PySpark - Created global and temporary views.
- Spark SQL - Created databases, tables, and views.
- Spark SQL - Transformations such as Filter, Join, Simple Aggregations, GroupBy, etc.
- Spark SQL - Created local and temporary views.
- Implemented full refresh and incremental load patterns using partition

Azure Data Factory

- Created pipelines to execute Databricks notebooks.
- Designed robust pipelines to deal with unexpected scenarios such as missing files.
- Created dependencies between activities as well as pipelines.
- Scheduled the pipelines using data factory triggers to execute at regular intervals.
- Monitored the triggers/ pipelines to check for errors/ outputs.

About the Project:

Folders:

- Pynb - folder contains youtube tending data analysis notebooks in ipynb format.

- Html - folder contains youtube tending data analysis notebooks in html format to view visualizations.

Technologies/Tools Used:

- Pyspark
- Spark SQL
- Azure Databricks
- Azure Data Factory
- Azure Data Lake Storage Gen2
- Azure Blob Storage
- Microsoft Excel (optional)

Presented By,

DE115 - Divya Sree Murali

DE120 – Jatin J

DE138 – Sivaprakash V