CS432/532: Final Project Report

Project Title: YouTube Data Analysis

Team Member(s): Sagar Rajiv Sahani, Akshay Shinde

I. PROBLEM

YouTube is the world-famous video sharing website that maintains a list of the top trending videos on the platform. To determine the year's top-trending videos, YouTube uses a combination of factors including measuring user's interactions (number of views, shares, comments and likes). However, they're not the most-viewed videos overall for the calendar year. Even though YouTube is famous video sharing platform around the world, it has its limitations. There is no categorization of videos based on their statistics. With this project, we plan to analyze YouTube statistics and filter or categorize videos on the basis of those statistics. The proposed system will cover wide range of aspects that YouTube doesn't offer at present. This system will not only provide users to explore more options, but also will help them to make better decisions and set their preferences accordingly.

II. SOFTWARE DESIGN AND IMPLEMENTATION

A. *Software Design and NoSQL-Database and Tools Used*

The project is implemented using python as a front-end and MongoDB as a backend. Python is general purpose, high-level, dynamic and interpreted programming language that supports objected oriented programming and advanced data structures. MongoDB is an open-source, cross platform, document-oriented NoSQL database that provides high performance, availability and automatic scaling. Pymongo was used to establish database connection. MongoDB utility tool was used to import the raw data and visualize the database. Sagar took the responsibility for the backend development and Akshay was responsible for front-end development with validations. GitHub was used as a code hosting platform for version control and collaboration.

The dataset played an important role for the design and implementation of our project. We have fetched a dataset 'Trending YouTube video statistics' from Kaggle. The dataset includes separate CSV files for each country along with all videos and its categories. So, we have implemented different collections for each of these CSV's making it easier to handle data and perform complex queries across multiple collections.

B. *Supported Queries*

Trivial Queries:

1. *How many videos in a category*
   Returns the count of videos from the videos collection by looking up the category id of the inserted category from the categories collection.

2. *How many videos in data by Channel*
   Gets count of videos after extracting the corresponding matches with input under channel_title. $match is used to look up the channels and $sum is used to retrieve the count.

3. *How many videos in all the Channels*
   This query returns the count of videos uploaded in all the channels by particular country dataset.

4. *Videos by particular Date*
   Returns all the videos after user has given upload date as an input.

5. *Videos by specific Tags*
   This query returns all the videos with respect to tag as an input

6. *Filter video category by channel name*
   This query first returns all the categories in the dataset for a country and allows selection for the same. After a selection is made, another query is triggered that retrieves all the channel names under the category. Although this is a two-step process, a variation of this has been executed as one-step processing under Non-trivial queries.

7. *Filter video category by Date*
   The working of this query is same as that of above query, except it also takes date as an input along with the category, executing second query to get videos in a category on a particular date.

8. *Search Tags*
   This query takes a string as an input and looks up every substring for a match under the tag column.

9. *Videos by most Likes*
   Returns the video with most likes.

10. *Videos by most Dislikes*
    Returns the video with most dislikes.

11. *Videos by top likes with most comments*
    Returns the video with most likes and comments. This can be also used as a filter for data analysis.

12. *Videos by top dislikes with comments*
    The working of this query is same as that of above, except it returns the video with most dislikes and comments.

13. *Trending videos on a particular date*
    This query takes date as an input and returns the videos that are trending on that particular date.

14. *Top liked video in country on a particular date*
    This query returns most liked videos on that particular date taking date as an input.

15. *Top viewed video in country on a particular date*
    This query returns most viewed video in the country on a particular date i.e., it performs sorting on the views of the videos and returns the video with the most views.

Non-trivial Queries:

1. *Videos by most likes on a particular trending date*
   This query returns one video with most likes on a particular date. Firstly, it looks up and retrieves the date as an input from the database. Secondly, it sorts with respect to the number of likes. Finally, it returns the video with most likes.

2. *Videos by most views on a particular trending date*
   The query works like the above query, except it returns the videos with most views on a particular date.

3. *Videos by channels in a particular affiliated tag*
   This query takes tag/hashtag as an input and displays channels associated with that tag. It checks each substring in the tag column for match and fetches all the corresponding channel names.

4. *Videos by channels in a particular category*
   This query is triggered on category collections. It looks up for the category id in the collection querying with the videos collection to get the output.

5. *Displays videos from a category for two countries together*
   This query looks up the category id from another collection and returns the videos in that collection for two countries.

6. *Display video of a particular category*
   This is one-step execution of Trivial query 6, returning the videos with category as an input.

III. Project Outcome

GitHub Link:
https://github.com/ashinde3/YouTube-Data-Analysis

References

[1] https://docs.mongodb.com/manual/
[2] https://www.kaggle.com/datasnaek/youtube-new.