# **ETL PROJECT REPORT**

|  |  |
| --- | --- |
| **Topic:** | Happiness World Report |
| **Date:** | December, 2021 |
|  |  |

**Table of Contents**

[Objective 3](#_Toc39004730)

[Problem Statement 3](#_Toc39004731)

[Data Cleanup and Analysis 3](#_Toc39004732)

[Conclusion 4](#_Toc39004733)

[Additional Links 5](#_Toc39004734)

# Objective

The objective of this report is to provide a detailed analysis of the ETL(Extract-Transform-Load) process by reading the dataset of Trending YouTube videos obtained from Kaggle, cleaning the dataset in the desired form and load into a database for storage.

# Problem Statement

We have been provided with the dataset of YouTube videos that have been trending in Canada and US along with the dataset of various categories they are belonging to. We are required to perform an ETL on the datasets which can be uploaded into production to determine which categories or video are most or least popular after depending on number of views, likes, dislikes and comments.

# Data Cleanup and Analysis

We have performed the ETL process in three steps:

**Step1: Extraction**

The data has been downloaded from public platform Kaggle with the following link -<https://www.kaggle.com/datasnaek/youtube-new>, the files used for extraction were originally formatted as .csv and .json-

* <https://www.kaggle.com/datasnaek/youtube-new#CAvideos.csv>
* <https://www.kaggle.com/datasnaek/youtube-new#USvideos.csv>
* <https://www.kaggle.com/datasnaek/youtube-new#US_category_id.json>

The .csv and .json files are then read into a Pandas data frame in Jupyter notebook.

**Step 2: Transformation**

We used Python as the tool to perform the transformation on the datasets using Pandas library, described in the following steps:

* The CAvideos.csv and USvideos.csv were first read into the Pandas data frame, videoCA\_df and videoUS\_df, using read\_csv.
* The columns that were not relevant were then dropped from the data frames by selecting the columns, namely video\_id, title, category\_id, publish\_time, views, likes, dislikes, comment\_count into the new\_videoCA\_df and new\_videoUS\_df.
* In these new data frames, the columns publish\_time and comment\_count were renamed to publish\_datetime and comments respectively.
* The duplicated video titles were dropped using drop\_duplicates(), and the last record was based on the assumption that data was structured in a way that the last record in the duplication contained the latest information about a trending video.
* We imported langdetect library and used it to drop the video titles that were in a language other than English were dropped.
* The US\_category\_id.json was then read into a Pandas data frame, category\_df, using read\_json().
* Based on the structure of json file, we only needed the category\_id and category\_title which had to be extracted from the item column of the category\_df.
* The item column from category\_df was stored in a list format, by iterating over this list the id and title were extracted and stored in separate lists which were then used in defining the new data frame, new\_category\_df.
* The new\_videoCA\_df and new\_videoUS\_df were merged separately with the new\_category\_df on category\_id and stored into a new data frames, mergedCA\_df and mergedUS\_df and a column for country was added to each storing “CA” and “US”.

**Step 3: Loading**

* We have used relational database, Postgres SQL to load our data.
* Then we establish a connection between SQL database youtube\_video\_db using create\_engine() in SQLAlchemy library, where we have created four tables video\_CA, video\_US, category, video\_category\_CA and video\_category\_US and load the data frames we have in the tables to be used for production.

# Conclusion

Using the database youtube\_video\_db, which consists of five tables namely video\_CA, video\_US, category, video\_category\_CA and video\_category\_US we can run multiple queries on it like getting the most popular category by total number of views, likes and comments for the different countries, also compare which video is most liked, if it is the same for both countries by joining the tables for the two countries etc. Depending on what we want to learn from our datasets, additional queries can be implemented using persistent nature of SQL database.

# Additional Links

1. Vaishali Tanwar’s Github Repository: <https://github.com/Vaishali91tanwar/ETL-Project>
2. Shihui Guo’s Github Repository: <https://github.com/Charrcoal/ETL-project>

We had a problem to read the data because we were using capital letters in the pgAdmin, we had to use cascade to automatically drop objects that it is pin in the table

Create a table first using lower case

* Copy the tabled code in the VC and paste in yhe pgAdmin and refresh, then you can try to connect in VC

Best practise

SELECT \* FROM country

Problem was unknown values that we had to duplicate and insert indez for every simple table.

Graphical user interface, text, application, Word

Description automatically generated

Used Pandas to transport the data

Schedule

Graphical user interface, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated