How Can Fandango MovieClips Maximize Their Youtube Popularity

by Joshua Nguyen

Abstract MovieClips is a *YouTube* channel that uploads and streams videos of movie clips. Movie clips and trailers can originate from **Universal Pictures, Walt Disney Studios, Paramount Pictures, Sony Pictures, DreamWorks** and **Warner Bros, ect.** As movies continue to play a significant part of the entertainment industry in the United States of America and around the world, MovieClips remains one of the most popular *Youtube* channel of all time. The channel was launched on December 3, 2009 and acquired by its current parent company, *Fandango* in 2014. The company changed the official *Youtube* channel name to **Fandango MovieClips.**

Project Overview

1. This project explores different statistical metrics to analyze and assay *Fandango MovieClips* viewership. Additionally, this project will display the results that is easily read and interpreted by the non-technical population who may not easily understand statistical vernacular.
2. This project will explore the relationship between *Fandango MovieClips*'s view, like and comment count since its acquisition in 2014.
3. This project aims to provide conclusions that will help *Fandango MovieClips* maintain and increase its viewership and popularity.
4. This project utilizes **Python** to use API for data extraction, **MySQL** for querying the dataset for valuable information and **Tableau** for data visualizations.

Objectives

The overall goal of this project is to build a report that showcases summarized information about **Fandango MovieClips**' viewership and provide ways on how the company can continue to remain a popular *YouTube* channel.

Questions of Interest

1. To date, how many total views, likes and comments does *Fandango MovieClips* have?
2. How has the view count changed in the past five years? Which month of which year generated the most views? On average, which month produced the most views?
3. What are *Fandango MovieClips*' top three most viewed movie clips? Most liked movie clips?
4. On average, which day should *Fandango MovieClips* upload a video in order to maximize views, likes and comments? Which day will generate the highest like-to-view ratio?
5. Which hour of which day of which month should *Fandango MovieClips* upload a random video in order to maximize the video's popularity in views?
6. The next month will be July. Planning ahead, on which day and time should *Fandango MovieClips* upload their most popular Hollywood movies in order to maximize their viewership potential?

Extraction, Transformation and Preparation

Extracting The Data

This project borrows and credits the Python coding heavily from Jacob Lower's Analytical Project. The original Python coding can be found at his deepnote page.

Import Libraries

```
import os
import requests
import time
from googleapiclient.discovery import build
```

Making API Call

```
API_KEY = 'ENTER_UNIQUE_API_KEY_HERE'
channel_id = 'UC3gNmTGu-TTbFPpfSs5kNkg' #MovieClips' Channel ID
youtube = build('youtube', 'v3', developerKey=API_KEY)

# Getting channel statistics and playlistID

def get_response(youtube, channel_id):
    request = youtube.channels().list(
        part="snippet,contentDetails,statistics",
        id=channel_id
    )
    response = request.execute()

    return response['items']
```

List of VideoID's

```
def get_video_list(youtube, upload_id):
    video_list = []
    request = youtube.playlistItems().list(
        part="snippet,contentDetails",
        playlistId=upload_id,
        maxResults=50
    )
    next_page = True
    while next_page:
        response = request.execute()
        data = response['items']

        for video in data:
            video_id = video['contentDetails']['videoId']
            if video_id not in video_list:
                video_list.append(video_id)

        # Do we have more pages?
        if 'nextPageToken' in response.keys():
            next_page = True
            request = youtube.playlistItems().list(
                part="snippet,contentDetails",
                playlistId=upload_id,
                pageToken=response['nextPageToken'],
                maxResults=50
            )
        else:
            next_page = False

    return video_list
```

List of Video Statistics

```
def get_video_details(youtube, video_list):
    stats_list=[]

    # Can only get 50 videos at a time.
    for i in range(0, len(video_list), 50):
        request= youtube.videos().list(
            part="snippet,id,contentDetails,statistics",
            id=video_list[i:i+50]
        )

        data = request.execute()

        for video in data['items']:
            title=video['snippet']['title']
            published=video['snippet']['publishedAt']

            publisheddate = str(published).split("T")[0]
            publishedtime = str(published).split("T")[1]

            description=len(video['snippet']['description'])
            view_count=video['statistics'].get('viewCount',0)
            like_count=video['statistics'].get('likeCount',0)
            comment_count=video['statistics'].get('commentCount',0)

            stats_dict=dict(title=title,
            description=description,
            publisheddate = publisheddate,
            publishedtime = publishedtime,
            view_count=view_count,
            like_count=like_count,
            comment_count=comment_count)
            stats_list.append(stats_dict)

    return stats_list
```

Main Function

```
def main (youtube, channel_id):
    channel_stats = get_response(youtube, channel_id)
    upload_id = channel_stats[0]['contentDetails']['relatedPlaylists']['uploads']
    video_list = get_video_list(youtube, upload_id)
    video_info = get_video_details(youtube, video_list)

    df = pd.DataFrame(video_info)
    return df
```

The Dataset

```
my_data = main(youtube, channel_id)
my_data.to_csv("MovieClips.csv")
my_data.head()
```

Viewing Dataset in SQL

This project utilized MySQL to extract the data into a csv file. First, import the dataset and select applicable fields of interest.

```
SELECT
        *,
    row_number() OVER(PARTITION BY publisheddate
        ORDER BY title ASC, publishedtime ASC) AS rownum
FROM movieclips;
```

Here, a row number column was added to view the order of movies uploaded ranked by the time a clip was published ordered by the day they were uploaded. **Swimfan (2002) Almost Busted Scene (2/5)** is the first clips uploaded in this dataset. Querying the data with *rownnum = 1*, we can obtain a list of the first movies uploaded by MovieClips on each day.

Our goal to extract a subset of our dataset and import it into **Tableau** in order to obtain visualizations. We will create two types of views that pertains to our questions of interest. The first view will select all fields with the addition of year, month and day groupings. These grouping columns corresponds to the total sum count for views, likes and comments on each day, month and year.

```
-- Selecting fields and create groupings
CREATE VIEW annual_views AS(
WITH CTE AS(
SELECT
    publisheddate,
    YEAR(publisheddate) as years,
    MONTHNAME(publisheddate) as months,
    DAYNAME(publisheddate) as days,
    publishedtime,
    view_count,
    like_count,
    comment_count
FROM movieclips
)
SELECT
    publisheddate, years, months, days,
    IF(GROUPING(years), 'Every year', years) AS gp_year,
    IF(GROUPING(months), 'Every months', months) AS gp_month,
    IF(GROUPING(days), 'Every days', days) AS gp_day,
    SUM(view_count), SUM(like_count), SUM(comment_count)
FROM CTE
    GROUP BY years, months, days WITH ROLLUP
    ORDER BY years, months, days
    );

-- Remove null values
CREATE VIEW annual_views1 AS (
SELECT
    *
FROM annual_views
        WHERE days IS NOT NULL);
```

```
-- Save into CSV file for export
SELECT
    'published_date',
    'years',
    'months',
    'days',
    'gp_year',
    'gp_month',
    'gp_day',
    'SUM(view_count)',
    'SUM(like_count)',
    'SUM(comment_count)'
        UNION ALL
                SELECT
                    *
                FROM annual_views1
                        INTO OUTFILE 'C:\\ProgramData\\MySQL\\MySQL Server
                        ↪  8.0\\Uploads\\movie_clips_annual_views_final.csv'
                                FIELDS TERMINATED BY ','
                                ENCLOSED BY ''
                                LINES TERMINATED BY '\n';
```

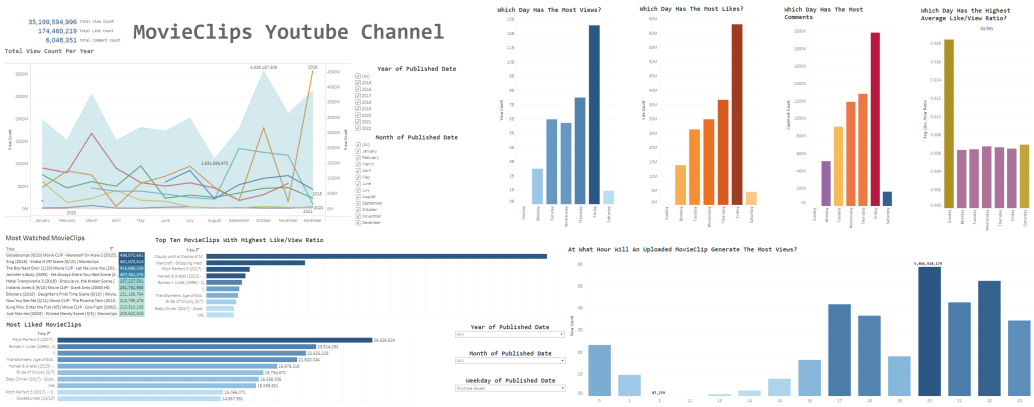
Our last view will select all fields of the original dataset with the addition of an order view rank column. This new column ranks all of the movie clips based on highest view count per year.

```
-- Selecting fields and create rankings
CREATE VIEW with_rnk AS
(
WITH order_views AS(
        SELECT
            title,
            description,
            publisheddate,
            publishedtime,
            HOUR(publishedtime) AS hour_block,
            view_count,
            like_count,
            comment_count,
            RANK() OVER(
                PARTITION BY YEAR(publisheddate)
                ORDER BY view_count DESC
        ) order_view_rank
        FROM movieclips
        )
SELECT
    *
FROM order_views
);
```

```
-- Save into CSV file
SELECT
    'title',
    'description',
    'publisheddate',
    'publishedtime',
    'hour_block',
    'view_count',
    'like_count',
    'comment_count',
    'order_view_rank'
        UNION ALL
            SELECT
                *
            FROM with_rnk
                INTO OUTFILE 'C:\\ProgramData\\MySQL\\MySQL Server
                ↪  8.0\\Uploads\\movie_clips_annual_views_with_rnk.csv'
                FIELDS TERMINATED BY ','
                ENCLOSED BY ''
                LINES TERMINATED BY '\n';
```

Visualizations

Importing the two datasets into Tableau gives us the following simple one-page dashboard that focuses on visualizing the questions of interests above. Click HERE to access the interactive dashboard.



Conclusion

The data was collected in February 2022 and its first data collection began in June of 2015. During this time span, *Fandango MovieClips* has garnered 35,199,594,996 views. Which is about 4 times the population of the entire world! Similarly, during this time span, across of their videos uploaded, the *YouTube* channel has about 175 million likes and 6 million commnets across all of their videos. Overall, these statistics are on the lower side and the actual values continue to grow each day.

Unfortunately, the channel's view growth has declined over this time span. The year with the most views occurred in 2016 when the channel garnered 10,016,520,107 views. The year with the lowest view count excluding the years (2015 and 2022) where data collection was not complete occured in 2021 when the channel only obtained 273,159,897 views. In fact, since 2016, the viewership has declined each year to date. Moreover, *Fandango MovieClips'* best month occured in December 0f 2016 when it attracted slightly over 3 billion views. In total, the month of October generated the highest view count totalling 4,539,187,506 views. August, on the other hand, generated the least view count out of all 12 months totalling only 1,631,566,473 views.

On average, in any given month, *Fandango MovieClips* should upload their videoes on Friday in

order to maximize viewership, like and comment count. However, it is best to wait until Sunday if *Fandango MovieClips* wants to improve their like-to-view ratios.

From above, October and the day Friday, on average, generated the most view count. Therefore, based on the dashboard, *Fandango MovieClips* should upload decisively around 8 PM on any given Friday in the month of October in order to maximize viewership potential.

Lets say it is summertime and the current month is July. *Fandango MovieClips* should statistically upload their videos at the following times for each day:

Monday	5 PM
Tuesday	9 PM
Wednesday	7 PM
Thursday	5 PM
Friday	8 PM

Therefore, it is best *Fandango MovieClips* to upload during the evening times, particularly on a Friday if they want to maximize view count.

Of course, correlation does not imply causation. Just because a movie clip was uploaded during the month of October or December does not guarantee high viewership. The content of the movie will also contribute to whether a video becomes a viral sensation. In particular, *Fandango MovieClips'* top three most view videos respectively are:

1. Goosebumps (6/10) Movie CLIP - Werewolf On Aisle 2 (2015) HD
2. Sing (2016) - Shake It Off Scene (6/10) | Movieclips
3. Sing (2016) - Shake It Off Scene (6/10) | Movieclips

Each of these three videos has bestowed over 400 million views with Goosebumps almost reaching half a billion views with its 498,072,651 view count.

YouTube success is also not just limited to its view count. A video's like and like-to-view ratio can also be a strong indicator of a video's success. *Fandango MovieClips'* most liked video clip is *Pitch Perfect 3 (2017) - Sit Still* which garnered close to 30 million likes.

Limitations

It is important to know that the data is not inclusive of *Fandango MovieClips'* entire collection of videos. Unfortunately, **YouTube's** API call only allows a collection of 20,000 videos at any given time. There also exists an intrinsic data bias because *Fandango MovieClips* does not upload a significant amount of videos on Sunday or in the morning times. This will of course decrease the viewership during this time span and consequently inflate Sunday's high like-to-view ratio.