YouTube Data Analysis

In [13]:
```
import requests
import pandas as pd
import time
```

In [14]:
```
API_KEY = 'AIzaSyCF0FiTq08JiOAsdzlQXlRTGF6fM4uxmgc'
CHANNEL_IDS = ['UC-CSyyi47VX1lD9zyeABW3w', 'UCz4a7agVFr1TxU-mpAP8hkw', 'UC6WzPg6yxF9dQx2_O6R4lww']
```

Function to fetch channel details

In [16]:
```
def get_channel_details(api_key, channel_ids):
    channel_details = []
    for channel_id in channel_ids:
        url = f"https://www.googleapis.com/youtube/v3/channels?part=snippet,statistics,contentDetails&id={chann
        response = requests.get(url)
        if response.status_code == 200:
            data = response.json()
            if 'items' in data and len(data['items']) > 0:
                item = data['items'][0]
                details = {
                    'channel_id': channel_id,
                    'channel_name': item['snippet']['title'],
                    'description': item['snippet']['description'],
                    'custom_url': item['snippet'].get('customUrl', ''),
                    'published_at': item['snippet']['publishedAt'],
                    'view_count': item['statistics']['viewCount'],
                    'subscriber_count': item['statistics'].get('subscriberCount', 0),
                    'video_count': item['statistics']['videoCount']
                }
                channel_details.append(details)
            else:
                print(f"No data found for channel ID: {channel_id}")
        else:
            print(f"Failed to fetch data for channel ID: {channel_id}, status code: {response.status_code}")
    return channel_details
```

Function to fetch video details

In [18]:
```
def get_video_details(api_key, video_ids, channel_id):
    video_details = []
    for video_id in video_ids:
        url = f"https://www.googleapis.com/youtube/v3/videos?part=snippet,statistics,contentDetails&id={video_i
        response = requests.get(url)
        if response.status_code == 200:
            data = response.json()
            if 'items' in data and len(data['items']) > 0:
                item = data['items'][0]
                details = {
                    'video_id': video_id,
                    'channel_id': channel_id,  # Add channel_id to video details
                    'title': item['snippet']['title'],
                    'description': item['snippet']['description'],
                    'tags': ','.join(item['snippet'].get('tags', [])),
                    'published_at': item['snippet']['publishedAt'],
                    'duration': item['contentDetails']['duration'],
                    'view_count': item['statistics'].get('viewCount', 0),
                    'like_count': item['statistics'].get('likeCount', 0),
                    'dislike_count': item['statistics'].get('dislikeCount', 0),
                    'comment_count': item['statistics'].get('commentCount', 0)
                }
                video_details.append(details)
            else:
                print(f"No data found for video ID: {video_id}")
        else:
            print(f"Failed to fetch data for video ID: {video_id}, status code: {response.status_code}")
    return video_details
```

Function to fetch up to 200 video IDs for a channel with pagination

In [20]:
```
def get_recent_video_ids(api_key, channel_id, max_videos=200):
    video_ids = []
    url = f"https://www.googleapis.com/youtube/v3/search?key={api_key}&channelId={channel_id}&part=id&order=dat
    while len(video_ids) < max_videos:
        response = requests.get(url)
        if response.status_code == 200:
            data = response.json()
```

```
            video_ids.extend([item['id']['videoId'] for item in data['items'] if 'videoId' in item['id']])
            if 'nextPageToken' in data and len(video_ids) < max_videos:
                next_page_token = data['nextPageToken']
                url = f"https://www.googleapis.com/youtube/v3/search?key={api_key}&channelId={channel_id}&part=
            else:
                break
        else:
            print(f"Failed to fetch video list for channel ID: {channel_id}, status code: {response.status_code
            break
    return video_ids[:max_videos]
```

Fetch channel details

In [22]:
```
channel_details = get_channel_details(API_KEY, CHANNEL_IDS)
```

Fetch video details for each channel

In [24]:
```
all_video_details = []
for channel_id in CHANNEL_IDS:
    video_ids = get_recent_video_ids(API_KEY, channel_id, max_videos=200)
    video_details = get_video_details(API_KEY, video_ids, channel_id)
    all_video_details.extend(video_details)
    time.sleep(1)
```

Convert data to DataFrames and save to CSV

In [26]:
```
df_channel_details = pd.DataFrame(channel_details)
df_video_details = pd.DataFrame(all_video_details)
```

Save the data to CSV files

In [28]:
```
df_channel_details.to_csv('channel_details.csv', index=False)
df_video_details.to_csv('video_details.csv', index=False)
```

In [ ]:
```

```