```
import pandas as pd
import numpy as np
```

```
youtube_history = pd.read_csv("youtube_history_raw.csv")
```

```
youtube_history.sample(2)
```

	source	name_of_video	url_of_video	name_of_channel	
9705	YouTube	Shawn Mendes - It'll Be Okay (Lyric Video)	https://www.youtube.com/watch?v=q_HNcBjHIPY	ShawnMendesVEVO	
13425	YouTube Music	exile	https://music.youtube.com/watch?v=Nm08yUg38tE	Taylor Swift - Topic	https:/

EST to IST

Considering I was in India before 8th August, 2023, I need to convert the EST time to IST time. Why? Because if I want to know at what time of the day do I consume content the most, I need to consider the time difference.

```
from datetime import datetime
import pytz
```

```
# Convert specific columns to int8
youtube_history['year'] = youtube_history['year'].astype('int16')
youtube_history['hour'] = youtube_history['hour'].astype('int8')
youtube_history['minute'] = youtube_history['minute'].astype('int8')
youtube_history['second'] = youtube_history['second'].astype('int8')

youtube_history['day'] = youtube_history['day'].astype(str).replace('[^0-9]', '', regex=True)
youtube_history['day'] = youtube_history['day'].astype('int8')
```

```
def convert_est_to_ist(timestamp_str):
    # Define time zones
    est = pytz.timezone('US/Eastern')
    ist = pytz.timezone('Asia/Kolkata')

    # Convert string to datetime object
    # Reference: 'Aug 15 2021 8:58:43 AM EST'
    timestamp_str = timestamp_str.replace(' EST', '')
    timestamp = datetime.strptime(timestamp_str, '%b %d %Y %I:%M:%S %p')

    # Set time zone to EST
    timestamp = est.localize(timestamp)

    # Convert to UTC and then to IST
    timestamp_utc = timestamp.astimezone(pytz.utc)
    timestamp_ist = timestamp_utc.astimezone(ist)

    # Return formatted IST timestamp
    return timestamp_ist.strftime('%b %d %Y %I:%M:%S %p')

month_dict = {'Jan': 1, 'Feb': 2, 'Mar': 3, 'Apr': 4, 'May': 5, 'Jun': 6, 'Jul': 7, 'Aug': 8, 'Sep': 9, 'Oct': 10, 'Nov': 11,
# print(month_dict["Aug"])
```

```
# for index, entry in youtube_history.iterrows():
#   print(index)
#   print(entry)
#   print(youtube_history.at[index, "day"])
#   youtube_history.at[index, "day"] = np.random.randint(1, 31)
#   print(youtube_history.at[index, "day"])
#   break
```

```
for index, entry in youtube_history.iterrows():
  if entry['month'] in ['Jan', 'Mar', 'May', 'Jul', 'Aug', 'Oct', 'Dec'] and entry['day'] > 31:
    changed_val = np.random.randint(1, 31)
    youtube_history.at[index, "day"] = changed_val
    entry['day'] = changed_val
  elif entry['month'] == 'Feb' and entry["day"] > 28:
    changed_val = np.random.randint(1, 28)
    youtube_history.at[index, "day"] = changed_val
    entry['day'] = changed_val
  elif entry["day"] > 30:
    changed_val = np.random.randint(1, 30)
    youtube_history.at[index, "day"] = changed_val
    entry['day'] = changed_val

  try:
    val = datetime(entry["year"], month_dict[entry["month"]], entry["day"]) < datetime(2023, month_dict["Aug"], 8)
  except:
    print(entry)

  if datetime(entry["year"], month_dict[entry["month"]], entry["day"]) < datetime(2023, month_dict["Aug"], 8):
    est_timestamp = f"{entry['month']} {entry['day']:02d} {entry['year']} {entry['hour']:02d}:{entry['minute']:02d}:{entry['se
    try:
      temp = convert_est_to_ist(est_timestamp)
    except:
      est_timestamp.replace("  ", " ")
      print(est_timestamp)
      continue
    # Update entry with IST timestamp components
    ist_timestamp = convert_est_to_ist(est_timestamp)
    ist_timestamp = datetime.strptime(ist_timestamp, '%b %d %Y %I:%M:%S %p')
    youtube_history.at[index, 'month'] = ist_timestamp.strftime('%b')
    youtube_history.at[index, 'day'] = ist_timestamp.day
    youtube_history.at[index, 'year'] = ist_timestamp.year
    youtube_history.at[index, 'hour'] = ist_timestamp.hour
    youtube_history.at[index, 'minute'] = ist_timestamp.minute
    youtube_history.at[index, 'second'] = ist_timestamp.second
    youtube_history.at[index, 'AM/PM'] = ist_timestamp.strftime('%p')
    youtube_history.at[index, 'time_zone'] = 'IST'
    # print(f"Original EST Timestamp: {est_timestamp}, Converted IST Timestamp: {ist_timestamp}")
  else:
    continue
    # print("Timestamp after August 8, 2023. No conversion needed.")
```

```
Jul 07 2023 00:07:53 AM EST
Jun 11 2023 00:21:24 AM EST
```

FILTERING TO GET ENTRIES OF 2023

```
# Reference: selecting rows based on condition : rslt_df = dataframe[dataframe['Percentage'] > 70]
youtube_history_2023 = youtube_history[youtube_history['year'] == 2023]
```

```
youtube_history_2023.sample(2)
```

	source	name_of_video	url_of_video	name_of_channel	
21329	YouTube Music	Daydreamin' (Live from London)	https://music.youtube.com/watch?v=Tb9G2YIx8w8	Ariana Grande - Topic	https://w
22204	YouTube	Taylor Swift - The Very First Night (Taylor's ...	https://www.youtube.com/watch?v=rVuyi-dPMIc	TaylorSwiftVEVO	https://w

Splitting into YT and YT-Music

I want to analyze both the videos and watch and the music I listen to. Video because it will be funa nd YT does not provide a YT wrapped. Music because I NEED AN ACCURATE WRAPPED FOR MYSELF BECAUSE I USED TWO SEPARATE ACCOUNTS IN 2023 😭

```
channels_df = pd.DataFrame(columns=['name', 'url', 'freq']) # key = (url)
artists_df = pd.DataFrame(columns=['name', 'url', 'freq']) # key = (url)

videos_df = pd.DataFrame(columns=['name_vid', 'url_vid', 'duration', 'freq', 'channel_key']) # key = (url_vid)
# videos_df['freq'] = 0
# videos_df['duration'] = None
music_df = pd.DataFrame(columns=['name_song', 'url_song', 'duration', 'freq', 'artist_key']) # key = (url_song)
# music_df['freq'] = 0
# music_df['duration'] = None

cols = youtube_history.columns

videos_history = pd.DataFrame(columns = cols)
music_history = pd.DataFrame(columns = cols)
# videos_history = pd.DataFrame(columns = ['vid_df_key', 'time_stamp', 'date', 'day', 'month', 'year', 'hour', 'minute', 'secor
# music_history = pd.DataFrame(columns = ['music_df_key', 'time_stamp', 'date', 'day', 'month', 'year', 'hour', 'minute', 'secc
```

```
# for index, entry in youtube_history_2023.iterrows():
#   print(index, entry)
#   break
```

```
# Reference: df.loc[len(df.index)] = ['Amy', 89, 93]
# NOTE use of at : channels_df.at[channel_url, 'freq'] +=  1
for index, row in youtube_history_2023.iterrows():
  if row['source'] == 'YouTube':
    # IT IS A VIDEO
    # Step 01: Update channels_df
    channel_url = row['url_of_channel']
    if channel_url in channels_df['url']:
      channels_df.at[channel_url, 'freq'] +=  1 # NOTE
    else:
      channels_df.loc[channel_url] = [row['name_of_channel'], channel_url, 1] # NOTE
    # Step 02: Update videos_df
    video_url = row['url_of_video']
    if video_url in videos_df['url_vid']:
      videos_df.at[video_url, 'freq'] += 1
    else:
      videos_df.loc[video_url] = [row['name_of_video'], video_url, None, 1, channel_url]
    # Step 03: Update videos_history
    # ['vid_df_key', 'time_stamp', 'date', 'day', 'month', 'year', 'hour', 'minute', 'second', 'AM/PM', 'time_zone']
    videos_history.loc[len(videos_history)] = row
  else:
    # IT IS A SONG
    #STEP 01: Update artists_df
    artist_url = row['url_of_channel']
    if artist_url in artists_df['url']:
      # print(artists_df.at[artist_url, 'freq'])
      artists_df.at[artist_url, 'freq'] +=  1
    else:
      artists_df.loc[artist_url] = [row['name_of_channel'], artist_url, 1]
    # Step 02: Update music_df
    song_url = row['url_of_video']
    if song_url in music_df['url_song']:
      music_df.at[song_url, 'freq'] += 1
    else:
      music_df.loc[song_url] = [row['name_of_video'], song_url, None, 1, artist_url]
    # Step 03: Update music_history
    music_history.loc[len(music_history)] = row
```

```
channels_df[channels_df['freq'] > 30]
```

	name	
https://www.youtube.com/channel/UCP0f9FXZ4g-MxAqSUreHpCw	Taylor Bell	https://w
url_chan	name_chan	
https://www.youtube.com/channel/UC0rE2qq81of4fojo-KhO5rg	Tanmay Bhat	https://
https://www.youtube.com/channel/UCeYUKvRrRY9V3DYdZd1Z_eQ	Strolling The City	https://ww
https://www.youtube.com/channel/UCBkcw8h7epT_bK0QzuY2Bmg	ActionKid	https://w
https://www.youtube.com/channel/UCANLZYMidaCbLQFWXBC95Jg	TaylorSwiftVEVO	https://ww

```
artists_df[artists_df['freq'] > 90].sort_values(by =  'freq', ascending=False)
```

	name	
https://www.youtube.com/channel/UCPC0L1d253x-KuMNwa05TpA	Taylor Swift - Topic	https://www.you
https://www.youtube.com/channel/UCVacQ2t5GUZ2t_J3la9BynA	Harry Styles - Topic	https://www.you
https://www.youtube.com/channel/UCO_sphdxl8_K6mfXzzmkDGQ	Prateek Kuhad - Topic	https://www.yo
https://www.youtube.com/channel/UCpe3TthvXb8kmRiFbkeDGCQ	Louis Tomlinson - Topic	https://www.you

```
videos_df[videos_df['freq'] > 4]
```

	name_vid	url_vid	duration	freq
url_vid	name_vid	url_vid	None	7
https://www.youtube.com/watch?v=L-vptrNAhSs	1B	https://www.youtube.com/watch?v=L-vptrNAhSs	None	5
https://www.youtube.com/watch?v=sjc34vNBNhQ	I Watched all 236 Episodes of Friends	https://www.youtube.com/watch?v=sjc34vNBNhQ	None	6
https://www.youtube.com/watch?v=WEDIj9JBTC8	William Ackman: Everything You Need to Know Ab...	https://www.youtube.com/watch?v=WEDIj9JBTC8	None	10

```
music_df[music_df['freq']>20].sort_values(by='freq', ascending=False)
```

	name_song	url_song	duration	fr
https://music.youtube.com/watch?v=4UAZkexQ7Cg	Little Freak	https://music.youtube.com/watch?v=4UAZkexQ7Cg	None	
https://music.youtube.com/watch?v=3lB7Oh076-8	champagne problems	https://music.youtube.com/watch?v=3lB7Oh076-8	None	
https://music.youtube.com/watch?v=FfBBGDZ4lzk	Satellite	https://music.youtube.com/watch?v=FfBBGDZ4lzk	None	
https://music.youtube.com/watch?v=bqJ9l-3MG1g	Cornelia Street	https://music.youtube.com/watch?v=bqJ9l-3MG1g	None	
https://music.youtube.com/watch?v=Dfiw7EmRTC4	Paris	https://music.youtube.com/watch?v=Dfiw7EmRTC4	None	
https://music.youtube.com/watch?v=BnYokCSJDfw	ADHOORE	https://music.youtube.com/watch?v=BnYokCSJDfw	None	
https://music.youtube.com/watch?v=Or58Kytwxrl	Keep Driving	https://music.youtube.com/watch?v=Or58Kytwxrl	None	

Extracting lengths of videos/songs using API

To find out how many hours/days I have spent consuming content this year, I'll use the YouTube Data API and extract lengths of songs and videos

```
!pip install google-api-python-client
import re
import requests
from googleapiclient.discovery import build
```

```
Requirement already satisfied: google-api-python-client in /usr/local/lib/python3.10/dist-packages (2.84.0)
Requirement already satisfied: httplib2<1dev,>=0.15.0 in /usr/local/lib/python3.10/dist-packages (from google-api-python-client) (0
Requirement already satisfied: google-auth<3.0.0dev,>=1.19.0 in /usr/local/lib/python3.10/dist-packages (from google-api-python-clie
Requirement already satisfied: google-auth-httplib2>=0.1.0 in /usr/local/lib/python3.10/dist-packages (from google-api-python-client
Requirement already satisfied: google-api-core!=2.0.*,!=2.1.*,!=2.2.*,!=2.3.0,<3.0.0dev,>=1.31.5 in /usr/local/lib/python3.10/dist-p
Requirement already satisfied: uritemplate<5,>=3.0.1 in /usr/local/lib/python3.10/dist-packages (from google-api-python-client) (4.1
```

```
Requirement already satisfied: googleapis-common-protos<2.0.dev0,>=1.56.2 in /usr/local/lib/python3.10/dist-packages (from google-ap
Requirement already satisfied: protobuf!=3.20.0,!=3.20.1,!=4.21.0,!=4.21.1,!=4.21.2,!=4.21.3,!=4.21.4,!=4.21.5,<5.0.0.dev0,>=3.19.5
Requirement already satisfied: requests<3.0.0.dev0,>=2.18.0 in /usr/local/lib/python3.10/dist-packages (from google-api-core!=2.0.*
Requirement already satisfied: cachetools<6.0,>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from google-auth<3.0.0dev,>=1.19.0
Requirement already satisfied: pyasn1-modules>=0.2.1 in /usr/local/lib/python3.10/dist-packages (from google-auth<3.0.0dev,>=1.19.0
Requirement already satisfied: six>=1.9.0 in /usr/local/lib/python3.10/dist-packages (from google-auth<3.0.0dev,>=1.19.0->google-api
Requirement already satisfied: rsa<5,>=3.1.4 in /usr/local/lib/python3.10/dist-packages (from google-auth<3.0.0dev,>=1.19.0->google
Requirement already satisfied: pyparsing!=3.0.0,!=3.0.1,!=3.0.2,!=3.0.3,<4,>=2.4.2 in /usr/local/lib/python3.10/dist-packages (from
Requirement already satisfied: pyasn1<0.6.0,>=0.4.6 in /usr/local/lib/python3.10/dist-packages (from pyasn1-modules>=0.2.1->google-a
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0.dev0,>=2.18
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0.dev0,>=2.18.0->google-ap
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0.dev0,>=2.18.0->goo
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0.dev0,>=2.18.0->goo
```

```python
def extract_video_id(url):
    # Extract video ID from YouTube URL
    # pattern = r"(?<=youtu.be\/|watch\?v=|embed\/|youtube.com\/v\/|youtube.com\/embed\/|youtube.com\/watch\?v=|youtube.com\/wat
    # match = re.search(pattern, url)
    vid_id = url.split("=")[-1]
    if vid_id:
        # return match.group(1)
        # print(vid_id)
        return vid_id
    else:
        return None

def get_youtube_video_info(video_id, api_key):
    # Make a request to YouTube Data API to get video details
    api_service_name = "youtube"
    api_version = "v3"
    youtube = build(api_service_name, api_version, developerKey=api_key)

    request = youtube.videos().list(
        part="snippet,contentDetails,statistics",
        id=video_id
    )
    response = request.execute()

    # Extract relevant information from the API response
    if "items" in response and len(response["items"]) > 0:
        video_info = response["items"][0]
        return video_info
    else:
        return None

def parse_duration(duration):
    # Remove the 'PT' at the beginning
    duration = duration[2:]

    # Initialize variables
    hours, minutes, seconds = 0, 0, 0

    # Parse hours if present
    if 'H' in duration:
        hours, duration = duration.split('H')
        hours = int(hours)

    # Parse minutes if present
    if 'M' in duration:
        minutes, duration = duration.split('M')
        minutes = int(minutes)

    # Parse seconds if present
    if 'S' in duration:
        seconds = int(duration.rstrip('S'))

    # Converting duration to minutes
    duration = hours*60 + minutes + seconds/60
    return duration

# Retrieved API Key using GCP
api_key = "YOUR_API_KEY"
```

```
for index, row in videos_df.iterrows():
  video_url = row['url_vid']
  video_id = extract_video_id(video_url)
  if video_id:
    # Get video information using the API key and video ID
    video_info = get_youtube_video_info(video_id, api_key)
    if video_info:
        duration_raw = video_info['contentDetails']['duration']
        duration = parse_duration(duration_raw)
        videos_df.at[video_url, 'duration'] = duration
    else:
        videos_df.at[video_url, 'duration'] = 11.7 # Average length of a Youtube video
  else:
    videos_df.at[video_url, 'duration'] = 11.7 # Average length of a Youtube video
```

```
indexDuration = videos_df[ videos_df['duration'] >= 1000 ].index
videos_df.drop(indexDuration, inplace=True)
videos_df[videos_df['duration'] > 120].sort_values(by = 'duration', ascending = False)
```

	name_vid	url_vid	duratio
https://www.youtube.com/watch?v=pkYVOmU3MgA	Data Structures and Algorithms in Python - Ful...	https://www.youtube.com/watch?v=pkYVOmU3MgA	750.83333
https://www.youtube.com/watch?v=LHBE6Q9XlzI	Python for Data Science - Course for Beginners...	https://www.youtube.com/watch?v=LHBE6Q9XlzI	739.86666
https://www.youtube.com/watch?v=aLRV8ND5oLw	NEW YORK 🗽 Iconic Soho and Washington Square Park	https://www.youtube.com/watch?v=aLRV8ND5oLw	719.01666
https://www.youtube.com/watch?v=salr45wvt4M	4K NEW YORK Walking Tour 🗽 NIGHT Walk BROOKLYN ...	https://www.youtube.com/watch?v=salr45wvt4M	714.98333
https://www.youtube.com/watch?v=jYHWCbevXkE	New York City NIGHT Walk 🗽 DOWNTOWN MANHATTAN W...	https://www.youtube.com/watch?v=jYHWCbevXkE	714.98333
https://www.youtube.com/watch?v=4n4kfYRJ9ss	New Year's Eve in NEW YORK CITY ☔ RAINY Walki...	https://www.youtube.com/watch?v=4n4kfYRJ9ss	714.96666
https://www.youtube.com/watch?v=oRlzCoj5YzA	NEW YORK Walk in Midtown Manhattan 4K	https://www.youtube.com/watch?v=oRlzCoj5YzA	714.96666
https://www.youtube.com/watch?v=Kxi3M1Qz2Gg	NEW YORK 2023 Manhattanhenge Madness, NYC Walk...	https://www.youtube.com/watch?v=Kxi3M1Qz2Gg	714.93333
https://www.youtube.com/watch?v=iLE8tW9AqR0	New York RAINY Lower Manhattan and SoHo Walk ☔...	https://www.youtube.com/watch?v=iLE8tW9AqR0	714.93333
https://www.youtube.com/watch?v=LOSaABLSlhE	NEW YORK Walking tour - Sunset walk in Manhatt...	https://www.youtube.com/watch?v=LOSaABLSlhE	714.93333
https://www.youtube.com/watch?v=-8qy1xTawaM	8 Hours Walking in NYC Manhattan	https://www.youtube.com/watch?v=-8qy1xTawaM	519.0
https://www.youtube.com/watch?v=7BFqWxDXjwE	St Patrick's Day Parade 2023 in New York City ...	https://www.youtube.com/watch?v=7BFqWxDXjwE	267.5
https://www.youtube.com/watch?v=INfe4jy6O6A	New York Driving Experience #7 • Cloudy day in...	https://www.youtube.com/watch?v=INfe4jy6O6A	266.83333
https://www.youtube.com/watch?v=iKt6bmsfFmY	NYC LIVE Times Square New Year's Eve 2023	https://www.youtube.com/watch?v=iKt6bmsfFmY	241.73333
https://www.youtube.com/watch?v=nKWDJbsSf7c	Jawan Waisi Nahi Jaisa Socha Tha | ScoopCast 61	https://www.youtube.com/watch?v=nKWDJbsSf7c	237.1
https://www.youtube.com/watch?v=-L4gB3q4Rus	FLYING OVER NEW YORK 4K UHD - Relaxing Music W...	https://www.youtube.com/watch?v=-L4gB3q4Rus	231.63333
https://www.youtube.com/watch?v=ZHOJHE1HgUM	CRICOMANIA'22	https://www.youtube.com/watch?v=ZHOJHE1HgUM	222.61666
https://www.youtube.com/watch?v=qR1LRJnEuz8	NEW YORK CITY TRAVEL - WALKING TOUR(5), Broadw...	https://www.youtube.com/watch?v=qR1LRJnEuz8	216.16666
https://www.youtube.com/watch?v=3u7MQz1EyPY	Power BI Full Course - Learn Power BI in 4 Hou...	https://www.youtube.com/watch?v=3u7MQz1EyPY	215.63333
https://www.youtube.com/watch?	NEW YORK CITY TRAVEL - WALKING	https://www.youtube.com/watch?	209.66666

v=33hGFMx5TI8	WALKING TOUR(9), Broadw...	v=33hGFMx5TI8	209.66666
https://www.youtube.com/watch?v=zBkIpC0KGxg	To The Point With Preeti Choudhry: 2024 Poll...	https://www.youtube.com/watch?v=zBkIpC0KGxg	203.2
https://www.youtube.com/watch?v=sjc34vNBNhQ	I Watched all 236 Episodes of Friends	https://www.youtube.com/watch?v=sjc34vNBNhQ	193.(
https://www.youtube.com/watch?v=K7ghUiXLef8	Ionic & Capacitor for Building Native Mobile A...	https://www.youtube.com/watch?v=K7ghUiXLef8	190.23333
https://www.youtube.com/watch?v=oYyez2uku1A	NEW YORK CITY TRAVEL 40 - WALKING TOUR MANHATT...	https://www.youtube.com/watch?v=oYyez2uku1A	175.(
https://www.youtube.com/watch?v=UQusM4ryhlU	Driving from Seattle to Vancouver, Canada	https://www.youtube.com/watch?v=UQusM4ryhlU	173.28333
https://www.youtube.com/watch?v=W1zbE_LP8R8	Live NYC Walk: Summer Saturday Morning Stroll ...	https://www.youtube.com/watch?v=W1zbE_LP8R8	162.98333
https://www.youtube.com/watch?v=ydxttMA3vew	Watch MISS AMERICANA with us !! Taylor Swift f...	https://www.youtube.com/watch?v=ydxttMA3vew	162.45
https://www.youtube.com/watch?v=ZhUZhggTD_Q	NEW YORK CITY TRAVEL 41 - WALKING TOUR MANHATT...	https://www.youtube.com/watch?v=ZhUZhggTD_Q	162.38333
https://www.youtube.com/watch?v=mS7wQxtadig	NYC LIVE Exploring Greenwich Village in Spring...	https://www.youtube.com/watch?v=mS7wQxtadig	158.43333
https://www.youtube.com/watch?v=Tf3HFXOdwXE	NYC LIVE Exploring Williamsburg to DUMBO, Broo...	https://www.youtube.com/watch?v=Tf3HFXOdwXE	157.5
https://www.youtube.com/watch?v=MypKoUONUDs	NYC Live - Rainy Day Vibes & Greek Independenc...	https://www.youtube.com/watch?v=MypKoUONUDs	154.23333

```
for index, row in music_df.iterrows():
  video_url = row['url_song']
  video_id = extract_video_id(video_url)
  if video_id:
    # Get video information using the API key and video ID
    video_info = get_youtube_video_info(video_id, api_key)
    if video_info:
        duration_raw = video_info['contentDetails']['duration']
        duration = parse_duration(duration_raw)
        music_df.at[video_url, 'duration'] = duration
    else:
        music_df.at[video_url, 'duration'] = 11.7 # Average length of a Youtube video
  else:
    music_df.at[video_url, 'duration'] = 11.7 # Average length of a Youtube video
```

```
music_df[(music_df['duration'] <= 11) & (music_df['freq'] > 1)].sort_values(by = 'duration', ascending=False).head(20)
```

	name_song	url_song	duration
https://music.youtube.com/watch?v=7SqC7_f_-0M	All Too Well (10 Minute Version) (Taylor's Ver...	https://music.youtube.com/watch?v=7SqC7_f_-0M	10.216667
https://music.youtube.com/watch?v=aPnqx56V8-0	All Too Well (10 Minute Version) (Taylor's Ver...	https://music.youtube.com/watch?v=aPnqx56V8-0	10.216667
https://music.youtube.com/watch?v=KC0AG8HfAi4	All Too Well (Sad Girl Autumn Version) - Recor...	https://music.youtube.com/watch?v=KC0AG8HfAi4	9.983333
https://music.youtube.com/watch?v=GipThm50RkU	Star Wars, Episode IV "A New Hope": Throne Roo...	https://music.youtube.com/watch?v=GipThm50RkU	7.966667
https://music.youtube.com/watch?v=_fmA1RoHbzA	Kun Faya Kun	https://music.youtube.com/watch?v=_fmA1RoHbzA	7.85
https://music.youtube.com/watch?v=vTBbi3qIC0s	Yun Hi Chala Chal	https://music.youtube.com/watch?v=vTBbi3qIC0s	7.45
https://music.youtube.com/watch?v=aV5f5S5U4Sw	Beautiful Jesus	https://music.youtube.com/watch?v=aV5f5S5U4Sw	6.533333
https://music.youtube.com/watch?v=3Cu5HLA_9T4	Dance for You	https://music.youtube.com/watch?v=3Cu5HLA_9T4	6.3
https://music.youtube.com/watch?v=BViDG6QDe1w	Fine Line	https://music.youtube.com/watch?v=BViDG6QDe1w	6.3
https://music.youtube.com/watch?v=ZvOrqlB1iLI	Romeo And Juliet	https://music.youtube.com/watch?v=ZvOrqlB1iLI	6.016667
https://music.youtube.com/watch?v=p4664nVBMxg	Saathiya	https://music.youtube.com/watch?v=p4664nVBMxg	5.966667
https://music.youtube.com/watch?v=JlGdhAuNyUc	AAOGE JAB TUM	https://music.youtube.com/watch?v=JlGdhAuNyUc	5.933333
https://music.youtube.com/watch?v=1maBkyzY7IA	Der Lagi Lekin	https://music.youtube.com/watch?v=1maBkyzY7IA	5.933333
https://music.youtube.com/watch?v=ntZONYzAMCo	Brooklyn Baby	https://music.youtube.com/watch?v=ntZONYzAMCo	5.866667
https://music.youtube.com/watch?v=ssF-hrwAHHc	Somebody Else	https://music.youtube.com/watch?v=ssF-hrwAHHc	5.8
https://music.youtube.com/watch?v=ZC6mEJ5i5v8	I Know The End	https://music.youtube.com/watch?v=ZC6mEJ5i5v8	5.75

Merging frequencies for duplicate song entries

```
final_music_df = music_df.groupby('name_song').agg({'freq': 'sum', 'artist_key': 'first','name_song' : 'first', 'url_song': 'f
```

```
len(final_music_df)
len(final_music_df['name_song'].unique())
```

```
2067
```

```
final_music_df.sort_values(by = 'freq', ascending=False).head(5)
```

	freq	artist_key	name_song	url_song	duration
name_song					
champagne		champagne		https://music.youtube.com/watch?...	

Exporting data frames to csv files

```
channels_df.to_csv("files/YTchannels.csv")
artists_df.to_csv("files/MusicArtists.csv")
videos_df.to_csv("files/YTvideos.csv")
final_music_df.to_csv("files/Songs.csv")
videos_history.to_csv("files/videos_history.csv")
music_history.to_csv("files/music_history.csv")
youtube_history_2023.to_csv("files/youtube_history_2023.csv")
```

REFERENCE CODE

```
# Extract video ID from the YouTube URL
# youtube_url = 'https://www.youtube.com/watch?v=WEDIj9JBTC8'
# youtube_url = 'https://music.youtube.com/watch?v=7SqC7_f_-0M'
# youtube_url = 'https://music.youtube.com/watch?v=u7hfwX81bY4'
video_id = extract_video_id(youtube_url)

if video_id:
    # Get video information using the API key and video ID
    video_info = get_youtube_video_info(video_id, api_key)

    if video_info:
        # Print relevant information
        for key, val in video_info.items():
          print(key, val)
        # print(f"Title: {video_info['snippet']['title']}")
        # print(f"Duration: {video_info['contentDetails']['duration']}")
        # print(f"Views: {video_info['statistics']['viewCount']}")
    else:
        print("Unable to fetch video information.")
else:
    print("Invalid YouTube URL.")
```

```
kind youtube#video
etag 7pciQGowdVjJNqCtYSC7jzSiG_8
id WEDIj9JBTC8
snippet {'publishedAt': '2012-11-27T19:42:33Z', 'channelId': 'UCvQECJukTDE2i6aCoMnS-Vg', 'title': 'William Ackman: Everything You Ne
contentDetails {'duration': 'PT43M57S', 'dimension': '2d', 'definition': 'hd', 'caption': 'true', 'licensedContent': True, 'contentF
statistics {'viewCount': '11785168', 'likeCount': '260903', 'favoriteCount': '0', 'commentCount': '11547'}
```

```
!pip install youtube-search-python
```

```
from youtubesearchpython import VideosSearch

def get_album_name(song_name):
    try:
        # Searching for the song on YouTube
        videos_search = VideosSearch(song_name, limit = 1)
        results = videos_search.result()

        # Extracting album name from the search results
        video_info = results['result'][0]
```