n-youtube-240159142165240159147138

November 27, 2024

1 Exploring Data Science and Analytics Trends on YouTube!

This dataset contains detailed information about videos from various YouTube channels that specialize in data science and analytics. It includes metrics such as views, likes, comments, and publication dates. The dataset consists of 22862 rows, providing a robust sample for analyzing trends in content engagement, popularity of topics over time, and comparison of channels' performance.

2 Import Library

[3]:
```
import pandas as pd
```

[123]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import seaborn as sns
```

3 Uploading Csv fle

[7]:
```
df = pd.read_csv(r"C:\Users\Syed␣
 ↪Arif\OneDrive\Desktop\Youtube_dataset_all_dataScience_channels.csv")
```

4 Data Preprocessing

5 head()

head is used show to the By default = 5 rows in the dataset

[12]:
```
df.head()
```

[12]:
```
     Channel_Name                                          Title  \
0  Rishabh Mishra  POWER BI Full PROJECT for Data Analysis with P…
1  Rishabh Mishra  AI Revolution - Future of Data Analyst Jobs & …
2  Rishabh Mishra  Reality of Data Analyst Courses and Data Scien…
3  Rishabh Mishra  Personal Portfolio Website for Beginners | How…
4  Rishabh Mishra  How To Create LinkedIn Profile in 2024 | Linke…
```

```
  Published_date     Views  Like_count  Comment_Count
0     2024-04-20  157284.0      5575.0          632.0
1     2024-03-23   78155.0      2712.0          245.0
2     2024-03-16   43627.0      1285.0          267.0
3     2024-01-21  129956.0      3462.0          358.0
4     2024-01-17  143309.0      3389.0          192.0
```

6 .tail()

tail is used to show rows by Descending order

[16]:
```
df.tail()
```

[16]:
```
           Channel_Name                                              Title  \
22997  Alex The Analyst  Data Analyst Resume | Reviewing My Resume! | F…
22998  Alex The Analyst  Working at a Big Company Vs Small Company | To…
22999  Alex The Analyst      Data Analyst Salary | 100k with No Experience
23000  Alex The Analyst  Truth About Big Companies | Told by a Fortune …
23001  Alex The Analyst                        Top 3 Data Analyst Skills in 2020

      Published_date    Views  Like_count  Comment_Count
22997     2020-01-30  72678.0      1679.0           64.0
22998     2020-01-25  15225.0       412.0           22.0
22999     2020-01-23  64571.0      2196.0          229.0
23000     2020-01-21   8943.0       328.0           19.0
23001     2020-01-17  29074.0      1406.0          140.0
```

7 shape

It show the total no of rows & Column in the dataset

[20]:
```
df.shape
```

[20]:
```
(23002, 6)
```

8 Columns

It show the no of each Column

[24]:
```
df.columns
```

[24]:
```
Index(['Channel_Name', 'Title', 'Published_date', 'Views', 'Like_count',
       'Comment_Count'],
      dtype='object')
```

9 .dtypes

This Attribute show the data type of each column

[28]:
```
df.dtypes
```

[28]:
```
Channel_Name       object
Title              object
Published_date     object
Views             float64
Like_count        float64
Comment_Count     float64
dtype: object
```

[40]:
```
# Replace NaN values with 0 (or any other appropriate default value)
df['Views'] = df['Views'].fillna(0)
df['Like_count'] = df['Like_count'].fillna(0)
df['Comment_Count'] = df['Comment_Count'].fillna(0)

# Convert the columns to integers
df['Views'] = df['Views'].astype('int64')
df['Like_count'] = df['Like_count'].astype('int64')
df['Comment_Count'] = df['Comment_Count'].astype('int64')
```

10 .unique()

In a column, It show the unique value of specific column.

[42]:
```
df["Channel_Name"].unique()
```

[42]:
```
array(['Rishabh Mishra', 'StatQuest with Josh Starmer',
       'Nicholas Renotte', 'Leila Gharani', 'Ryan Nolan Data',
       'WsCube Tech', 'codebasics', 'Ken Jee', 'Kaggle', 'Luke Barousse',
       'DeepLearningAI', 'Dataquest', 'Keith Galli', 'Darshil Parmar',
       'Rob Mulla', 'Andrej Karpathy', 'Tina Huang', 'sentdex', 'CampusX',
       'Chandoo', 'freeCodeCamp.org', 'Socratica', 'Kevin Stratvert',
       'Tableau Tim', 'Thu Vu data analytics', 'Guy in a Cube',
       'Krish Naik', 'techTFQ', 'ExcelIsFun', 'Alex The Analyst'],
      dtype=object)
```

11 .nuique()

It will show the total no of unque value from whole data frame

[46]:
```
df.nunique()
```

[46]:
```
Channel_Name         30
Title             22906
Published_date     4333
Views             19376
Like_count         5312
Comment_Count      1115
dtype: int64
```

12 .describe()

It show the Count, mean , median etc

[36]:
```
df.describe()
```

[36]:
```
              Views    Like_count  Comment_Count
count  2.300000e+04  2.296700e+04   22975.000000
mean   1.135679e+05  2.419010e+03     102.550120
std    5.401123e+05  1.256386e+04     431.296342
min    0.000000e+00  0.000000e+00       0.000000
25%    6.447750e+03  9.600000e+01       8.000000
50%    1.911850e+04  3.600000e+02      25.000000
75%    6.556725e+04  1.277000e+03      77.000000
max    4.426709e+07  1.041273e+06   44343.000000
```

13 .value_counts

It Shows all the unique values with their count

[48]:
```
df["Channel_Name"].value_counts()
```

[48]:
```
Channel_Name
WsCube Tech                     5167
ExcelIsFun                      3698
Krish Naik                      1836
freeCodeCamp.org                1674
sentdex                         1254
CampusX                         1051
Guy in a Cube                   1039
codebasics                       881
Kevin Stratvert                  858
Socratica                        661
Leila Gharani                    584
Tableau Tim                      510
Chandoo                          480
DeepLearningAI                   435
Kaggle                           380
```

```
Alex The Analyst               308
Nicholas Renotte               308
Ken Jee                        287
StatQuest with Josh Starmer    279
Tina Huang                     222
Luke Barousse                  159
Rob Mulla                      157
Ryan Nolan Data                157
Darshil Parmar                 152
techTFQ                        136
Keith Galli                     89
Rishabh Mishra                  88
Thu Vu data analytics           87
Dataquest                       50
Andrej Karpathy                 15
Name: count, dtype: int64
```

14 isnull()

It shows the how many null values

[52]:
```
df.isnull()
```

[52]:
```
       Channel_Name  Title  Published_date  Views  Like_count  Comment_Count
0             False  False           False  False       False          False
1             False  False           False  False       False          False
2             False  False           False  False       False          False
3             False  False           False  False       False          False
4             False  False           False  False       False          False
…                …      …               …      …           …              …
22997         False  False           False  False       False          False
22998         False  False           False  False       False          False
22999         False  False           False  False       False          False
23000         False  False           False  False       False          False
23001         False  False           False  False       False          False

[23002 rows x 6 columns]
```

15 .info()

To Show Data type of each column

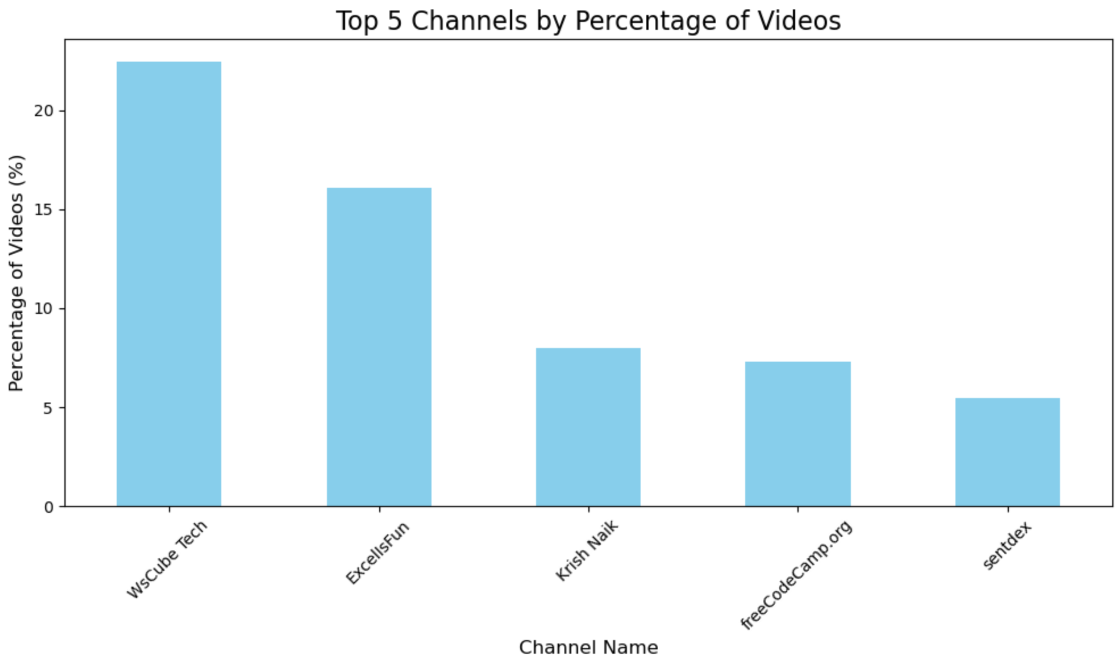
[56]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 23002 entries, 0 to 23001
```

```
Data columns (total 6 columns):
 #   Column          Non-Null Count  Dtype 
---  ------          --------------  ----- 
 0   Channel_Name    23002 non-null  object
 1   Title           23002 non-null  object
 2   Published_date  23002 non-null  object
 3   Views           23002 non-null  int64 
 4   Like_count      23002 non-null  int64 
 5   Comment_Count   23002 non-null  int64 
dtypes: int64(3), object(3)
memory usage: 1.1+ MB
```

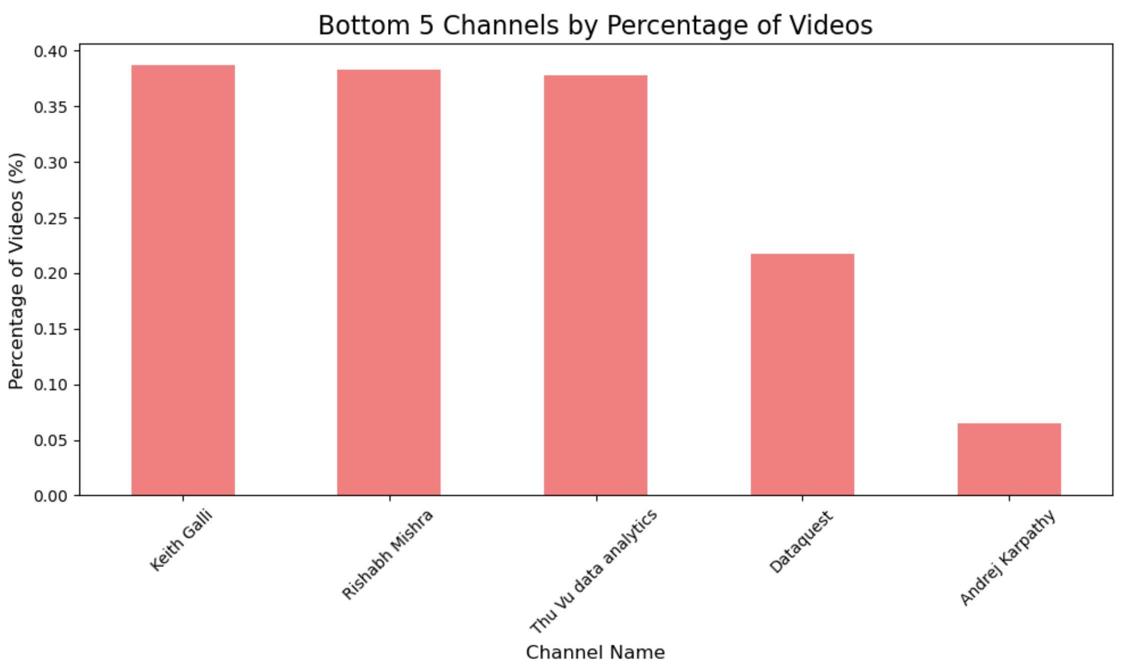
16 1. Highest 5 Channels by Number of Videos (%) ?

[65]:
```python
# Calculate the percentage of videos for each channel
channel_counts = df['Channel_Name'].value_counts()
total_videos = channel_counts.sum()
channel_percentages = (channel_counts / total_videos) * 100

# Get the top 5 channels
top_5_channels = channel_percentages.head(5)

# Plot the bar chart
plt.figure(figsize=(10, 6))
top_5_channels.plot(kind='bar', color='skyblue')
plt.title("Top 5 Channels by Percentage of Videos", fontsize=16)
plt.ylabel("Percentage of Videos (%)", fontsize=12)
plt.xlabel("Channel Name", fontsize=12)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



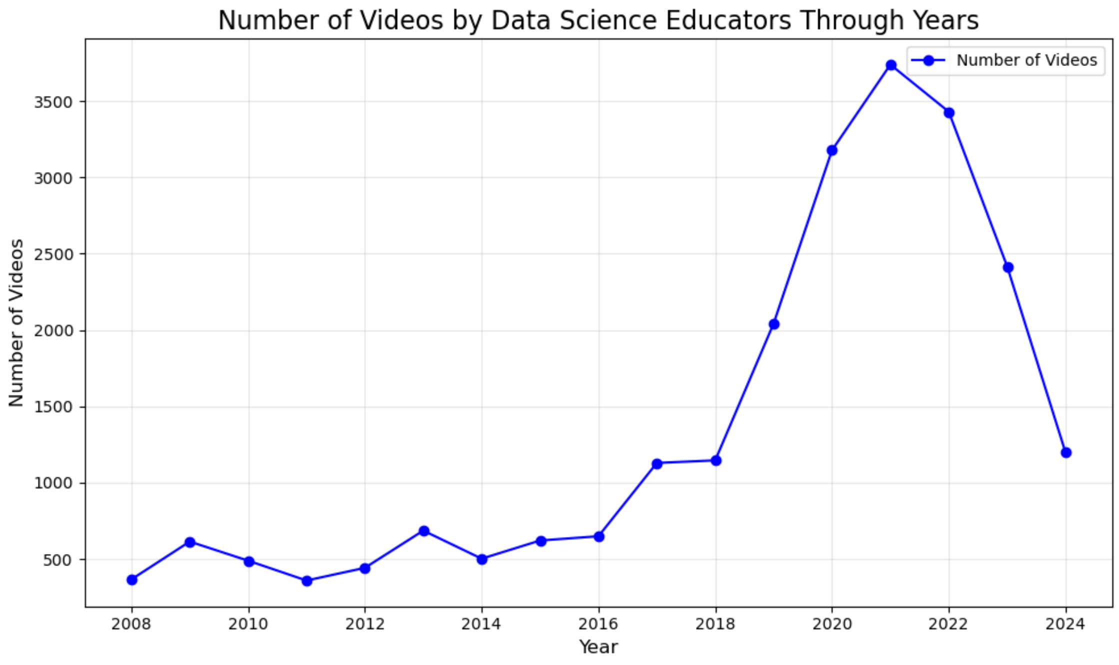
17 2. Lowest 5 Channels by Number of Videos (%)

[68]:
```python
# Get the bottom 5 channels
bottom_5_channels = channel_percentages.tail(5)

# Plot the bar chart
plt.figure(figsize=(10, 6))
bottom_5_channels.plot(kind='bar', color='lightcoral')
plt.title("Bottom 5 Channels by Percentage of Videos", fontsize=16)
plt.ylabel("Percentage of Videos (%)", fontsize=12)
plt.xlabel("Channel Name", fontsize=12)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



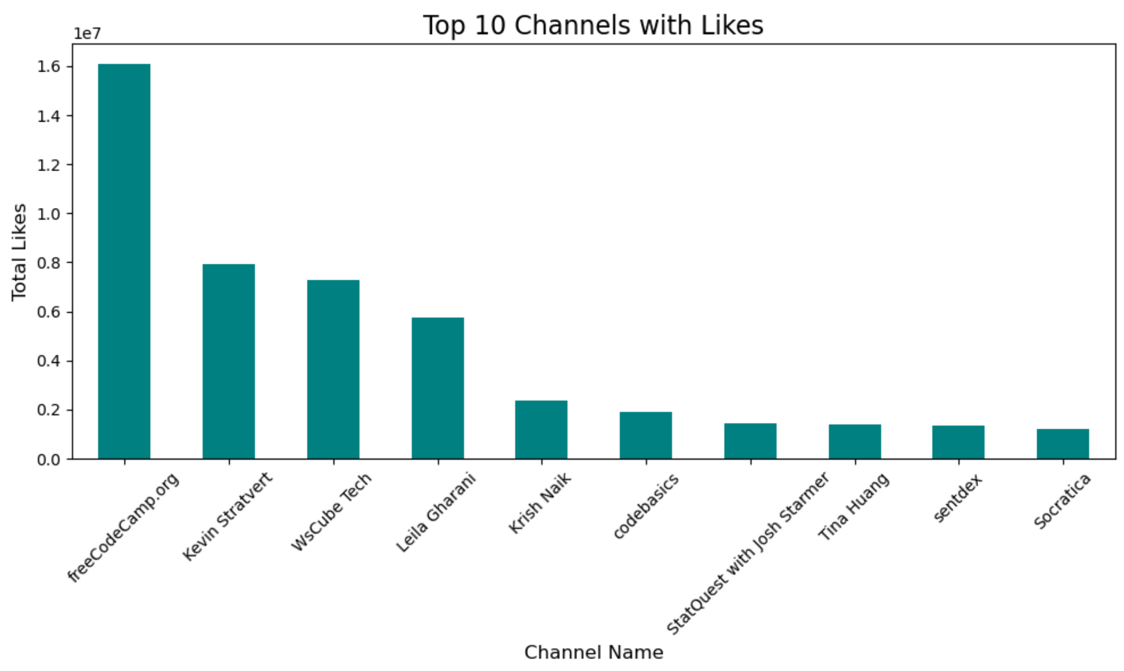
18 Number of videos of data science educators through years ?

[71]:
```python
# Convert 'Published_date' to datetime format
df['Published_date'] = pd.to_datetime(df['Published_date'])

# Extract the year from the publication date
df['Year'] = df['Published_date'].dt.year

# Group by year and count the number of videos
videos_per_year = df.groupby('Year').size()

# Plot the line chart
plt.figure(figsize=(10, 6))
plt.plot(videos_per_year.index, videos_per_year.values, marker='o',␣
 ↪color='blue', linestyle='-', label='Number of Videos')

# Customize the plot
plt.title("Number of Videos by Data Science Educators Through Years",␣
 ↪fontsize=16)
plt.xlabel("Year", fontsize=12)
plt.ylabel("Number of Videos", fontsize=12)
plt.grid(alpha=0.3)
plt.legend()
plt.tight_layout()
plt.show()
```



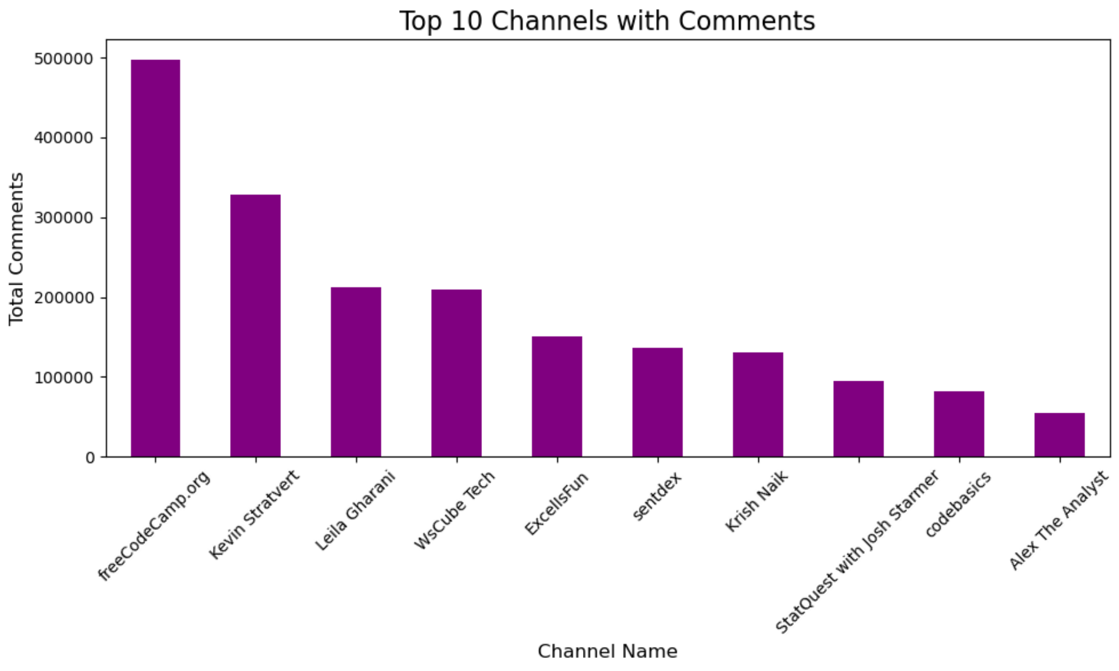
19 Top 10 Channels with Likes

[74]:
```
# Group by channel and sum up the likes
top_channels_likes = df.groupby('Channel_Name')['Like_count'].sum().
 ↪sort_values(ascending=False).head(10)

# Plot the bar chart
top_channels_likes.plot(kind='bar', figsize=(10, 6), color='teal')
plt.title("Top 10 Channels with Likes", fontsize=16)
plt.xlabel("Channel Name", fontsize=12)
plt.ylabel("Total Likes", fontsize=12)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



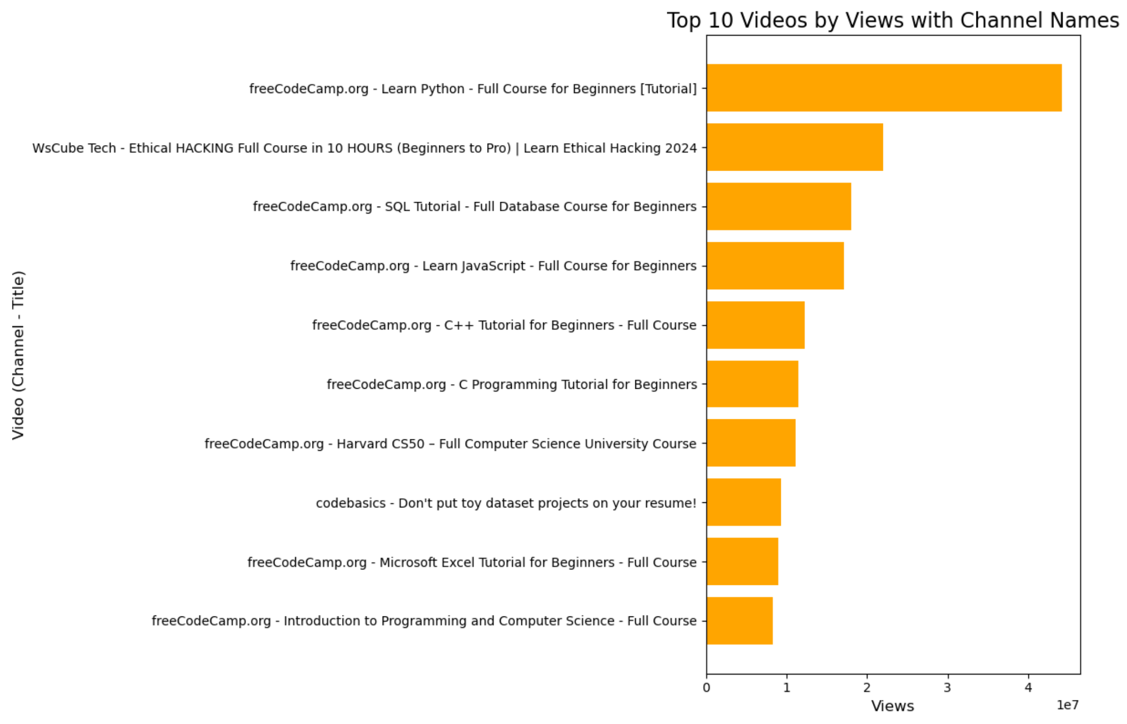
20 Top 10 Channels with Comments

[77]:
```
# Group by channel and sum up the comments
top_channels_comments = df.groupby('Channel_Name')['Comment_Count'].sum().
 ↪sort_values(ascending=False).head(10)

# Plot the bar chart
top_channels_comments.plot(kind='bar', figsize=(10, 6), color='purple')
plt.title("Top 10 Channels with Comments", fontsize=16)
plt.xlabel("Channel Name", fontsize=12)
plt.ylabel("Total Comments", fontsize=12)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



21 Top 10 videos by views

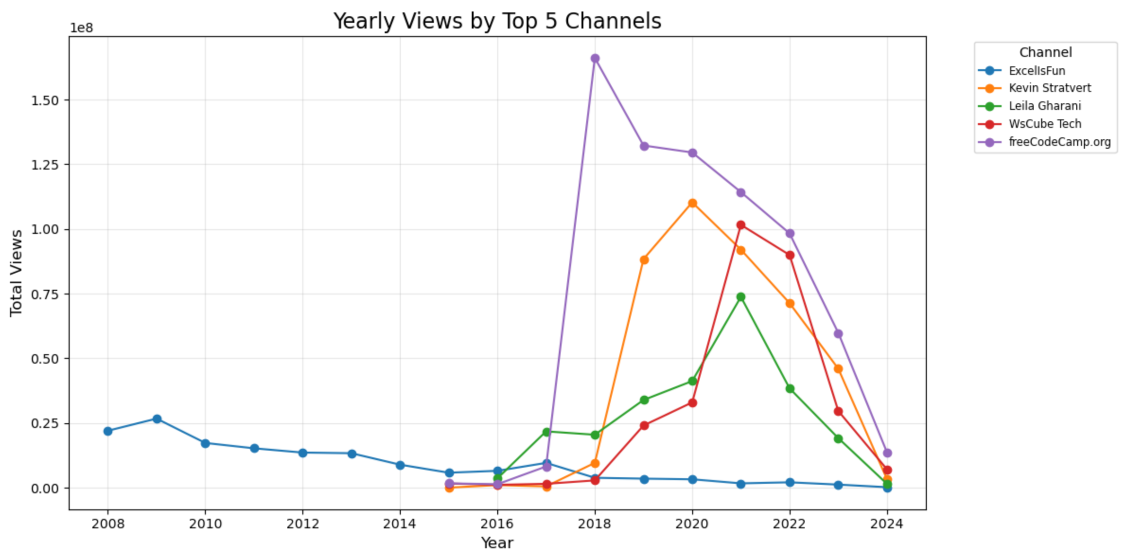
[94]:
```
# Sort videos by views and get the top 10
top_videos_views = df[['Channel_Name', 'Title', 'Views']].
 ↪sort_values(by='Views', ascending=False).head(10)

# Combine channel name and title for labeling
top_videos_views['Label'] = top_videos_views['Channel_Name'] + " - " +␣
 ↪top_videos_views['Title']

# Plot the bar chart
plt.figure(figsize=(12, 8))
plt.barh(top_videos_views['Label'], top_videos_views['Views'], color='orange')
plt.xlabel('Views', fontsize=12)
plt.ylabel('Video (Channel - Title)', fontsize=12)
plt.title('Top 10 Videos by Views with Channel Names', fontsize=16)
plt.gca().invert_yaxis()  # To display the highest view at the top
plt.tight_layout()
plt.show()
```

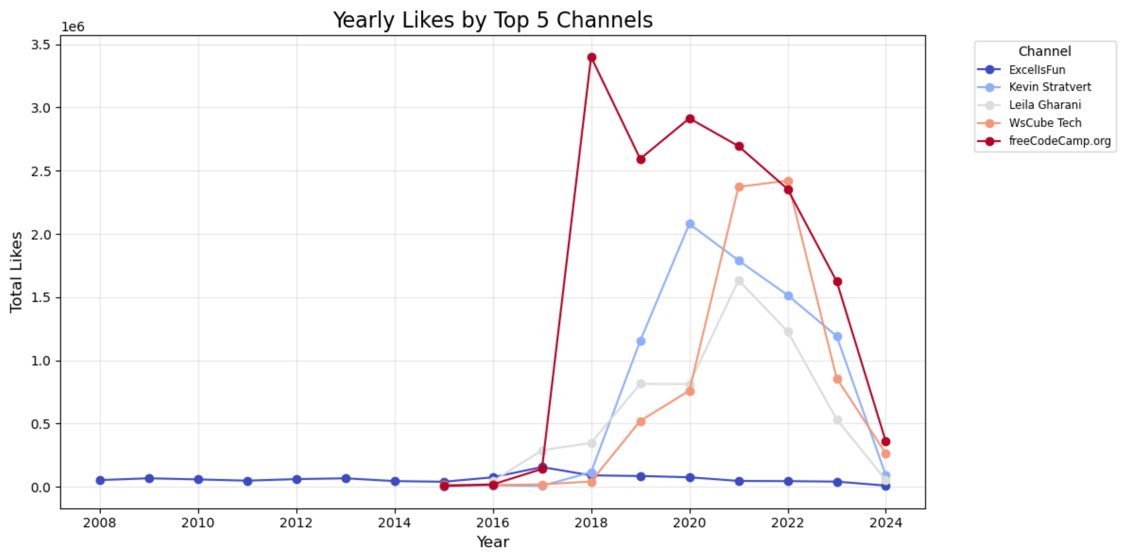


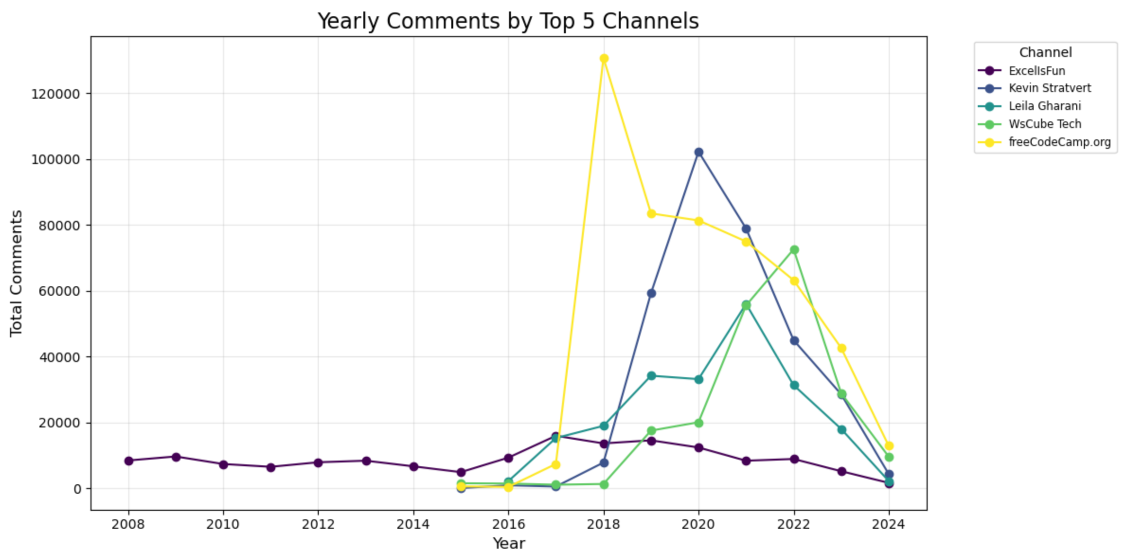
22 Top 5 channels with more than 10 million total likes || views || Comments

[96]:
```
# Ensure 'Published_date' is in datetime format
df['Published_date'] = pd.to_datetime(df['Published_date'])

# Extract year
df['Year'] = df['Published_date'].dt.year

# Group by year and channel, summing up views, likes, and comments
yearly_channel_stats = df.groupby(['Year', 'Channel_Name'])[['Views',␣
↪'Like_count', 'Comment_Count']].sum().reset_index()

# Calculate total views for each channel to identify top 5
total_views_per_channel = yearly_channel_stats.groupby('Channel_Name')['Views'].
↪sum().sort_values(ascending=False).head(5)

# Filter the original data to include only top 5 channels by total views
top_5_channels_data = yearly_channel_stats[yearly_channel_stats['Channel_Name'].
↪isin(total_views_per_channel.index)]

# Pivot the data for each metric (Views, Likes, Comments) to make plotting␣
↪easier
```

```
views_pivot = top_5_channels_data.pivot(index='Year', columns='Channel_Name',␣
 ↪values='Views')
likes_pivot = top_5_channels_data.pivot(index='Year', columns='Channel_Name',␣
 ↪values='Like_count')
comments_pivot = top_5_channels_data.pivot(index='Year',␣
 ↪columns='Channel_Name', values='Comment_Count')

# Plot Views by Channel for Top 5 Channels
plt.figure(figsize=(12, 6))
views_pivot.plot(ax=plt.gca(), marker='o')
plt.title("Yearly Views by Top 5 Channels", fontsize=16)
plt.xlabel("Year", fontsize=12)
plt.ylabel("Total Views", fontsize=12)
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left', fontsize='small',␣
 ↪title='Channel')
plt.grid(alpha=0.3)
plt.tight_layout()
plt.show()

# Plot Likes by Channel for Top 5 Channels
plt.figure(figsize=(12, 6))
likes_pivot.plot(ax=plt.gca(), marker='o', colormap='coolwarm')
plt.title("Yearly Likes by Top 5 Channels", fontsize=16)
plt.xlabel("Year", fontsize=12)
plt.ylabel("Total Likes", fontsize=12)
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left', fontsize='small',␣
 ↪title='Channel')
plt.grid(alpha=0.3)
plt.tight_layout()
plt.show()

# Plot Comments by Channel for Top 5 Channels
plt.figure(figsize=(12, 6))
comments_pivot.plot(ax=plt.gca(), marker='o', colormap='viridis')
plt.title("Yearly Comments by Top 5 Channels", fontsize=16)
plt.xlabel("Year", fontsize=12)
plt.ylabel("Total Comments", fontsize=12)
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left', fontsize='small',␣
 ↪title='Channel')
plt.grid(alpha=0.3)
plt.tight_layout()
plt.show()
```


Yearly Views by Top 5 Channels
1e8
Total Views
Year
Channel
ExcelIsFun
Kevin Stratvert
Leila Gharani
WsCube Tech
freeCodeCamp.org

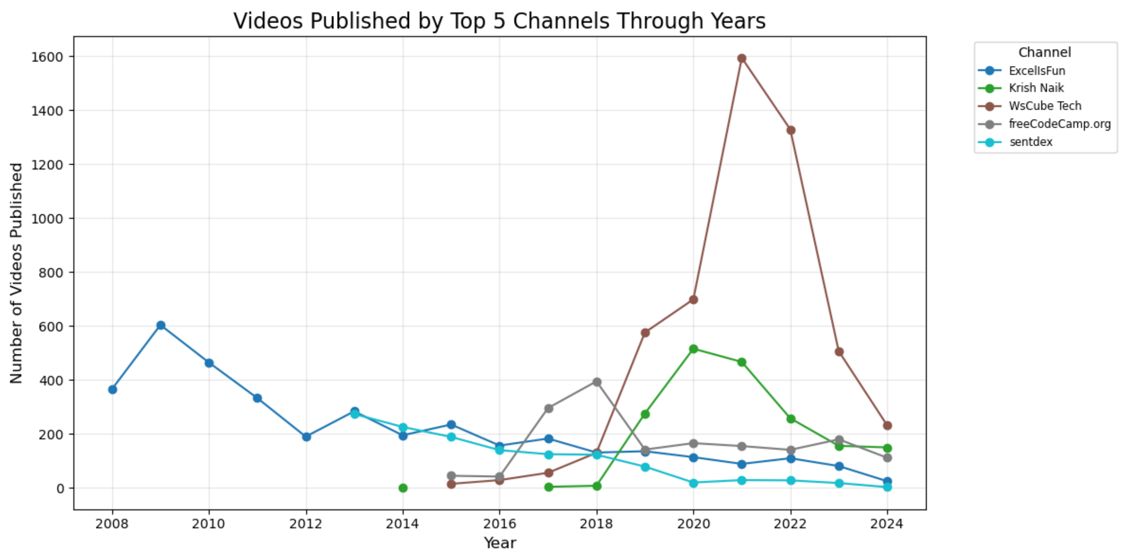

Yearly Likes by Top 5 Channels
1e6
Total Likes
Year
Channel
ExcelIsFun
Kevin Stratvert
Leila Gharani
WsCube Tech
freeCodeCamp.org



23 Videos Published by Channel per Year

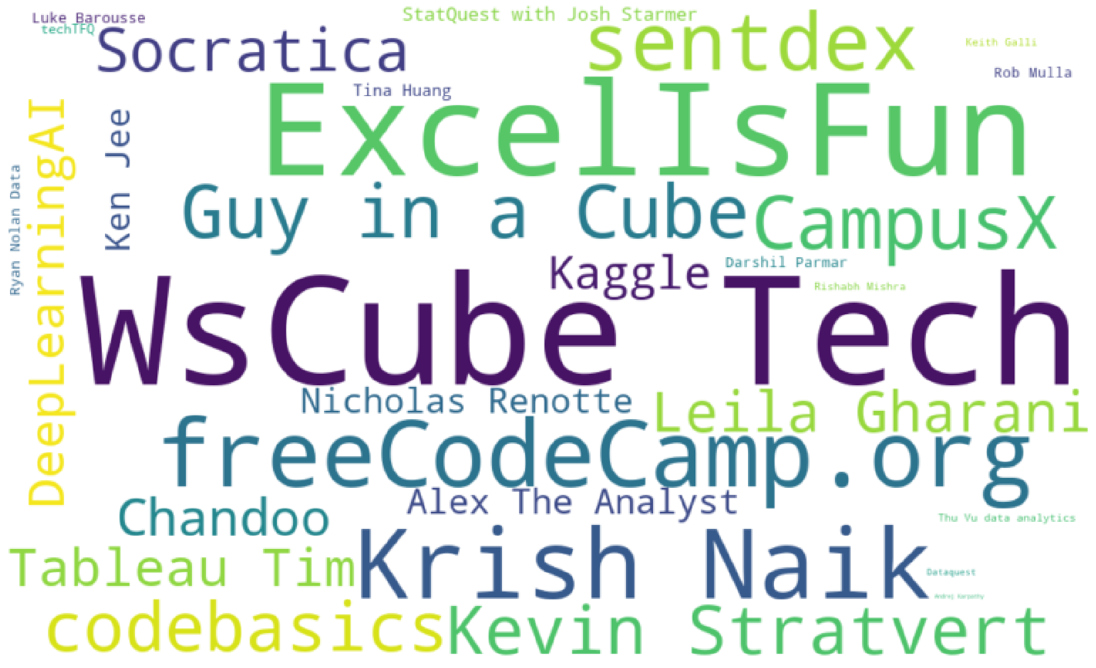
[102]:
```
# Ensure 'Published_date' is in datetime format
df['Published_date'] = pd.to_datetime(df['Published_date'])

# Extract year
df['Year'] = df['Published_date'].dt.year

# Group by year and channel, counting the number of videos published
videos_per_year = df.groupby(['Year', 'Channel_Name']).size().
  ↪reset_index(name='Video_Count')

# Calculate total videos published by each channel across all years
total_videos_per_channel = videos_per_year.
  ↪groupby('Channel_Name')['Video_Count'].sum().sort_values(ascending=False).
  ↪head(5)

# Filter the data to include only the top 5 channels
top_5_channels_data = videos_per_year[videos_per_year['Channel_Name'].
  ↪isin(total_videos_per_channel.index)]

# Pivot the data for the top 5 channels to make plotting easier
top_5_channels_pivot = top_5_channels_data.pivot(index='Year',␣
  ↪columns='Channel_Name', values='Video_Count')

# Plot the trend of videos published by the top 5 channels
plt.figure(figsize=(12, 6))
```

```
top_5_channels_pivot.plot(ax=plt.gca(), marker='o', linestyle='-',␣
 ↪colormap='tab10')  # Using a colormap for better differentiation
plt.title("Videos Published by Top 5 Channels Through Years", fontsize=16)
plt.xlabel("Year", fontsize=12)
plt.ylabel("Number of Videos Published", fontsize=12)
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left', fontsize='small',␣
 ↪title='Channel')
plt.grid(alpha=0.3)
plt.tight_layout()
plt.show()
```



24 WordCloud of Channels

[127]:
```
# Import necessary libraries
from wordcloud import WordCloud
import matplotlib.pyplot as plt

# Generate a word cloud based on the frequency of channel names
wordcloud = WordCloud(width=1000, height=600, background_color='white',␣
 ↪colormap='viridis').generate_from_frequencies(df['Channel_Name'].
 ↪value_counts())

# Plot the word cloud
plt.figure(figsize=(10, 6))
plt.imshow(wordcloud, interpolation='bilinear')
plt.title('Word Cloud of Channels', fontsize=16)
plt.axis('off')  # Disable the axis for better visualization
plt.show()
```


Word Cloud of Channels

```
[ ]:
```