In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

In [2]:

```
df=pd.read_csv(r"C:\Users\dutta\Downloads\Task 1 YouTube Streamer Analysis-20240705T0606
df
```

Out[2]:

	Rank	Username	Categories	Suscribers	Country	Visits	Likes	Comments
0	1	tseries	Música y baile	249500000.0	India	86200.0	2700.0	78.0
1	2	MrBeast	Videojuegos, Humor	183500000.0	Estados Unidos	117400000.0	5300000.0	18500.0
2	3	CoComelon	Educación	165500000.0	Unknown	7000000.0	24700.0	0.0
3	4	SETIndia	NaN	162600000.0	India	15600.0	166.0	9.0
4	5	KidsDianaShow	Animación, Juguetes	113500000.0	Unknown	3900000.0	12400.0	0.0
...	...	...	...	...	...	...	...	...
995	996	hamzymukbang	NaN	11700000.0	Estados Unidos	397400.0	14000.0	124.0
996	997	Adaahqueen	NaN	11700000.0	India	1100000.0	92500.0	164.0
997	998	LittleAngelIndonesia	Música y baile	11700000.0	Unknown	211400.0	745.0	0.0
998	999	PenMultiplex	NaN	11700000.0	India	14000.0	81.0	1.0
999	1000	OneindiaHindi	Noticias y Política	11700000.0	India	2200.0	31.0	1.0

1000 rows × 9 columns

In [3]:

```
df.head()
```

Out[3]:

	Rank	Username	Categories	Suscribers	Country	Visits	Likes	Comments	
0	1	tseries	Música y baile	249500000.0	India	86200.0	2700.0	78.0	h
1	2	MrBeast	Videojuegos, Humor	183500000.0	Estados Unidos	117400000.0	5300000.0	18500.0	http:/
2	3	CoComelon	Educación	165500000.0	Unknown	7000000.0	24700.0	0.0	http
3	4	SETIndia	NaN	162600000.0	India	15600.0	166.0	9.0	http:/
4	5	KidsDianaShow	Animación, Juguetes	113500000.0	Unknown	3900000.0	12400.0	0.0	ht

In [4]:

```
df.shape
```

Out[4]:

```
(1000, 9)
```

In [5]:

```
len(df)
```

Out[5]:

```
1000
```

In [6]:

```
len(df.columns)
```

Out[6]:

```
9
```

In [7]:

```
df[['Rank','Categories','Suscribers','Country','Visits','Likes','Comments']]
```

Out[7]:

	Rank	Categories	Suscribers	Country	Visits	Likes	Comments
0	1	Música y baile	249500000.0	India	86200.0	2700.0	78.0
1	2	Videojuegos, Humor	183500000.0	Estados Unidos	117400000.0	5300000.0	18500.0
2	3	Educación	165500000.0	Unknown	7000000.0	24700.0	0.0
3	4	NaN	162600000.0	India	15600.0	166.0	9.0
4	5	Animación, Juguetes	113500000.0	Unknown	3900000.0	12400.0	0.0
...	...	...	...	...	...	...	...
995	996	NaN	11700000.0	Estados Unidos	397400.0	14000.0	124.0
996	997	NaN	11700000.0	India	1100000.0	92500.0	164.0
997	998	Música y baile	11700000.0	Unknown	211400.0	745.0	0.0
998	999	NaN	11700000.0	India	14000.0	81.0	1.0
999	1000	Noticias y Política	11700000.0	India	2200.0	31.0	1.0

1000 rows × 7 columns

In [8]:

```
df['Username'] = df['Username'].str.replace(r'\W','')
df['Categories'] = df['Categories'].str.replace(r'\W','')
df['Country'] = df['Country'].str.replace(r'\W','')
df['Links'] = df['Links'].str.replace(r'\W','')
```

In [9]:

```
df[['Username','Categories','Country','Links']]
```

Out[9]:

	Username	Categories	Country	Links
0	tseries	Música y baile	India	http://youtube.com/channel/UCq-Fj5jknLsUf-MWSy...

	Username	Categories	Country	Links
1	MrBeast	Videojuegos, Humor	Estados Unidos	http://youtube.com/channel/UCX6OQ3DkcsbYNE6H8u...
2	CoComelon	Educación	Unknown	http://youtube.com/channel/UCbCmjCuTUZos6Inko4...
3	SETIndia	NaN	India	http://youtube.com/channel/UCpEhnqL0y41EpW2TvW...
4	KidsDianaShow	Animación, Juguetes	Unknown	http://youtube.com/channel/UCk8GzjMOrta8yxDcKf...
...	...	...	...	...
995	hamzymukbang	NaN	Estados Unidos	http://youtube.com/channel/UCPKNKIdggioffXPkSm...
996	Adaahqueen	NaN	India	http://youtube.com/channel/UCk3fFpqI5kDMf__mUP...
997	LittleAngelIndonesia	Música y baile	Unknown	http://youtube.com/channel/UCdrHrQf0o0TO8YDntX...
998	PenMultiplex	NaN	India	http://youtube.com/channel/UCObyBrdrtQ20BU9PxH...
999	OneindiaHindi	Noticias y Política	India	http://youtube.com/channel/UCOjgc1p2hJ4GZi6pQQ...

1000 rows × 4 columns

In [10]:

```
#for charactical value
df['Username'] = df['Username'].fillna(df['Username'].mode()[0])
df['Categories'] = df['Categories'].fillna(df['Categories'].mode()[0])
df['Country'] = df['Country'].fillna(df['Country'].mode()[0])
df['Links'] = df['Links'].fillna(df['Links'].mode()[0])
#for Numerical Value
df['Rank'] = df['Rank'].fillna(np.mean(pd.to_numeric(df['Rank'])))
df['Suscribers'] = df['Suscribers'].fillna(np.mean(pd.to_numeric(df['Suscribers'])))
df['Visits'] = df['Visits'].fillna(np.mean(pd.to_numeric(df['Visits'])))
df['Likes'] = df['Likes'].fillna(np.mean(pd.to_numeric(df['Likes'])))
df['Comments'] = df['Comments'].fillna(np.mean(pd.to_numeric(df['Comments'])))
```

In [11]:

```
df[['Rank','Username','Categories','Suscribers','Country','Visits','Likes','Comments']]
```

Out[11]:

	Rank	Username	Categories	Suscribers	Country	Visits	Likes	Comments
0	1	tseries	Música y baile	249500000.0	India	86200.0	2700.0	78.0
1	2	MrBeast	Videojuegos, Humor	183500000.0	Estados Unidos	117400000.0	5300000.0	18500.0
2	3	CoComelon	Educación	165500000.0	Unknown	7000000.0	24700.0	0.0
3	4	SETIndia	Música y baile	162600000.0	India	15600.0	166.0	9.0
4	5	KidsDianaShow	Animación, Juguetes	113500000.0	Unknown	3900000.0	12400.0	0.0
...	...	...	...	...	...	...	...	...
995	996	hamzymukbang	Música y	11700000.0	Estados	397400.0	14000.0	124.0

	Rank	Username	Categories	Suscribers	Country	Visits	Likes	Comments
			baile		Unidos			
996	997	Adaahqueen	Música y baile	11700000.0	India	1100000.0	92500.0	164.0
997	998	LittleAngelIndonesia	Música y baile	11700000.0	Unknown	211400.0	745.0	0.0
998	999	PenMultiplex	Música y baile	11700000.0	India	14000.0	81.0	1.0
999	1000	OneindiaHindi	Noticias y Política	11700000.0	India	2200.0	31.0	1.0

1000 rows × 8 columns

```
In [12]:
```

```
df['Rank'] = df['Rank'].astype(int)
df['Suscribers'] = df['Suscribers'].astype(int)
df['Visits'] = df['Visits'].astype(int)
df['Likes'] = df['Likes'].astype(int)
df['Comments'] = df['Comments'].astype(int)
```

```
In [13]:
```

```
df[['Rank','Suscribers','Visits','Likes','Comments']]
```

```
Out[13]:
```

	Rank	Suscribers	Visits	Likes	Comments
0	1	249500000	86200	2700	78
1	2	183500000	117400000	5300000	18500
2	3	165500000	7000000	24700	0
3	4	162600000	15600	166	9
4	5	113500000	3900000	12400	0
...	...	...	...	...	...
995	996	11700000	397400	14000	124
996	997	11700000	1100000	92500	164
997	998	11700000	211400	745	0
998	999	11700000	14000	81	1
999	1000	11700000	2200	31	1

1000 rows × 5 columns

```
In [14]:
```

```
# Our New Data Frame looking like this
df[['Rank','Username','Categories','Suscribers','Country','Visits','Likes','Comments']]
```

```
Out[14]:
```

	Rank	Username	Categories	Suscribers	Country	Visits	Likes	Comments
0	1	tseries	Música y baile	249500000	India	86200	2700	78

	Rank	Username	Categories	Suscribers	Country	Visits	Likes	Comments
1	2	MrBeast	Videojuegos, Humor	183500000	Estados Unidos	117400000	5300000	18500
2	3	CoComelon	Educación	165500000	Unknown	7000000	24700	0
3	4	SETIndia	Música y baile	162600000	India	15600	166	9
4	5	KidsDianaShow	Animación, Juguetes	113500000	Unknown	3900000	12400	0
...	...	...	...	...	...	...	...	...
995	996	hamzymukbang	Música y baile	11700000	Estados Unidos	397400	14000	124
996	997	Adaahqueen	Música y baile	11700000	India	1100000	92500	164
997	998	LittleAngelIndonesia	Música y baile	11700000	Unknown	211400	745	0
998	999	PenMultiplex	Música y baile	11700000	India	14000	81	1
999	1000	OneindiaHindi	Noticias y Política	11700000	India	2200	31	1

1000 rows × 8 columns

In [15]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 9 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   Rank        1000 non-null   int32 
 1   Username    1000 non-null   object
 2   Categories  1000 non-null   object
 3   Suscribers  1000 non-null   int32 
 4   Country     1000 non-null   object
 5   Visits      1000 non-null   int32 
 6   Likes       1000 non-null   int32 
 7   Comments    1000 non-null   int32 
 8   Links       1000 non-null   object
dtypes: int32(5), object(4)
memory usage: 50.9+ KB
```

In [16]:

```
#check for null values or missing values
pd.isnull(df).sum()
```

Out[16]:
```
Rank          0
Username      0
Categories    0
Suscribers    0
Country       0
Visits        0
Likes         0
Comments      0
Links         0
dtype: int64
```

In [17]:
```python
#drop null values
df.dropna(inplace=True)
```

In [18]:
```python
#duplicate values
df.duplicated().sum()
```

Out[18]:
```
0
```

In [19]:
```python
df.shape
```

Out[19]:
```
(1000, 9)
```

In [20]:
```python
df.head()
```

Out[20]:

	Rank	Username	Categories	Suscribers	Country	Visits	Likes	Comments	
0	1	tseries	Música y baile	249500000	India	86200	2700	78	http://y
1	2	MrBeast	Videojuegos, Humor	183500000	Estados Unidos	117400000	5300000	18500	http://youtu
2	3	CoComelon	Educación	165500000	Unknown	7000000	24700	0	http://yo
3	4	SETIndia	Música y baile	162600000	India	15600	166	9	http://yout
4	5	KidsDianaShow	Animación, Juguetes	113500000	Unknown	3900000	12400	0	http://y

In [21]:
```python
df.tail()
```

Out[21]:

	Rank	Username	Categories	Suscribers	Country	Visits	Likes	Comments	
995	996	hamzymukbang	Música y baile	11700000	Estados Unidos	397400	14000	124	http://you
996	997	Adaahqueen	Música y baile	11700000	India	1100000	92500	164	http://you
997	998	LittleAngelIndonesia	Música y	11700000	Unknown	211400	745	0	http://you

	Rank	Username	Categories	Suscribers	Country	Visits	Likes	Comments	
			baile						
998	999	PenMultiplex	Música y baile	11700000	India	14000	81	1	http://yout
999	1000	OneindiaHindi	Noticias y Política	11700000	India	2200	31	1	http://yout

In [22]:

```
#describe() method returns description of the datain the DataFrame(i.e. count,mean,std,m
df.describe()
```

Out[22]:

	Rank	Suscribers	Visits	Likes	Comments
count	1000.000000	1.000000e+03	1.000000e+03	1.000000e+03	1000.000000
mean	500.500000	2.189440e+07	1.209446e+06	5.363259e+04	1288.768000
std	288.819436	1.682775e+07	5.229942e+06	2.580457e+05	6778.188308
min	1.000000	1.170000e+07	0.000000e+00	0.000000e+00	0.000000
25%	250.750000	1.380000e+07	3.197500e+04	4.717500e+02	2.000000
50%	500.500000	1.675000e+07	1.744500e+05	3.500000e+03	67.000000
75%	750.250000	2.370000e+07	8.654750e+05	2.865000e+04	472.000000
max	1000.000000	2.495000e+08	1.174000e+08	5.300000e+06	154000.000000

In [23]:

```
# use describe() for specific columns
df[['Suscribers', 'Visits', 'Likes', 'Comments']].describe()
```

Out[23]:

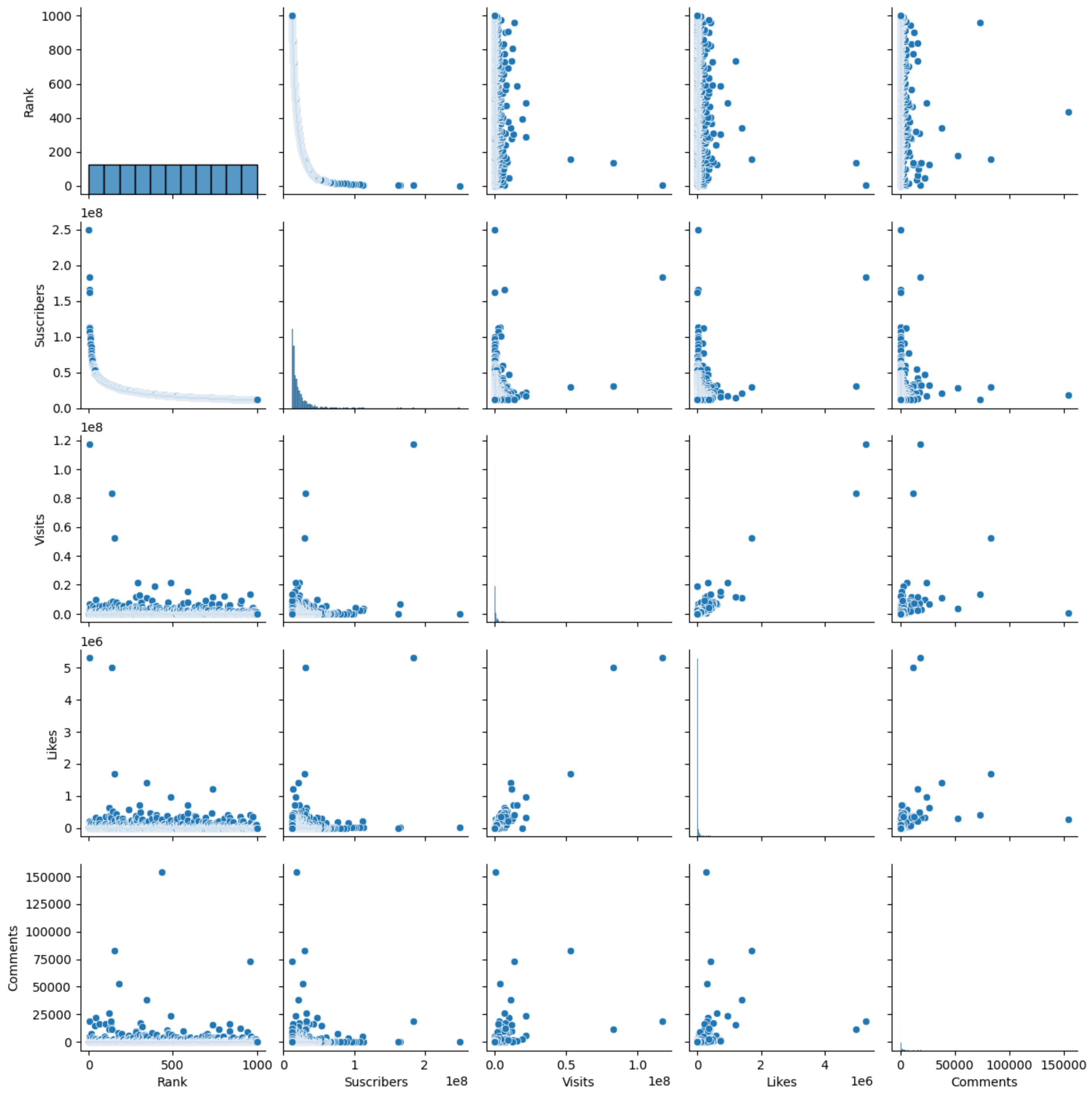
	Suscribers	Visits	Likes	Comments
count	1.000000e+03	1.000000e+03	1.000000e+03	1000.000000
mean	2.189440e+07	1.209446e+06	5.363259e+04	1288.768000
std	1.682775e+07	5.229942e+06	2.580457e+05	6778.188308
min	1.170000e+07	0.000000e+00	0.000000e+00	0.000000
25%	1.380000e+07	3.197500e+04	4.717500e+02	2.000000
50%	1.675000e+07	1.744500e+05	3.500000e+03	67.000000
75%	2.370000e+07	8.654750e+05	2.865000e+04	472.000000
max	2.495000e+08	1.174000e+08	5.300000e+06	154000.000000

In [24]:

```
# To check Relationship
sns.pairplot(data=df, kind='scatter')
```

Out[24]:

```
<seaborn.axisgrid.PairGrid at 0x2ac0ee453d0>
```

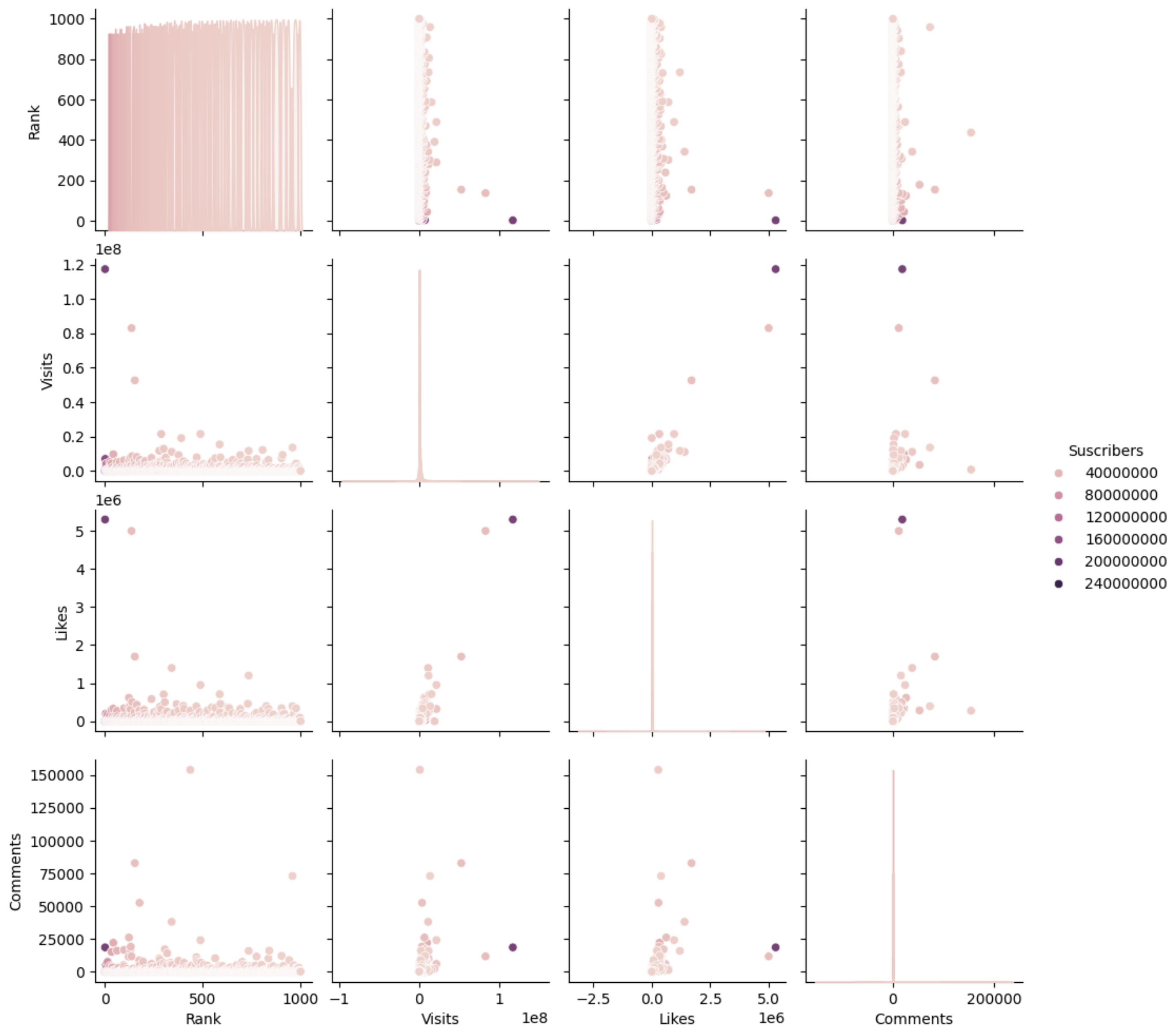


In [25]:

```
sns.pairplot(data=df,hue='Suscribers')
```

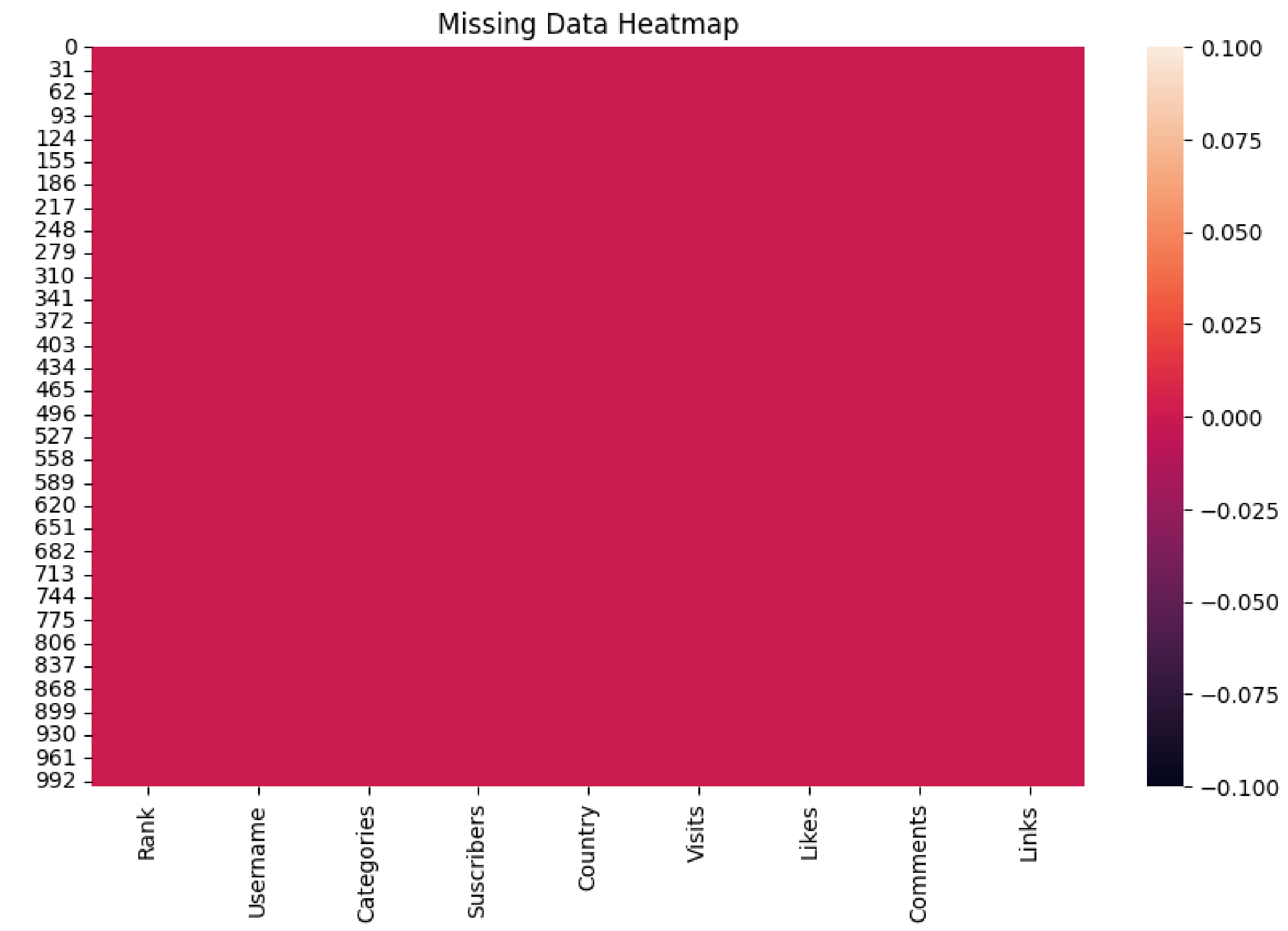
Out[25]:

```
<seaborn.axisgrid.PairGrid at 0x2ac132f2ea0>
```



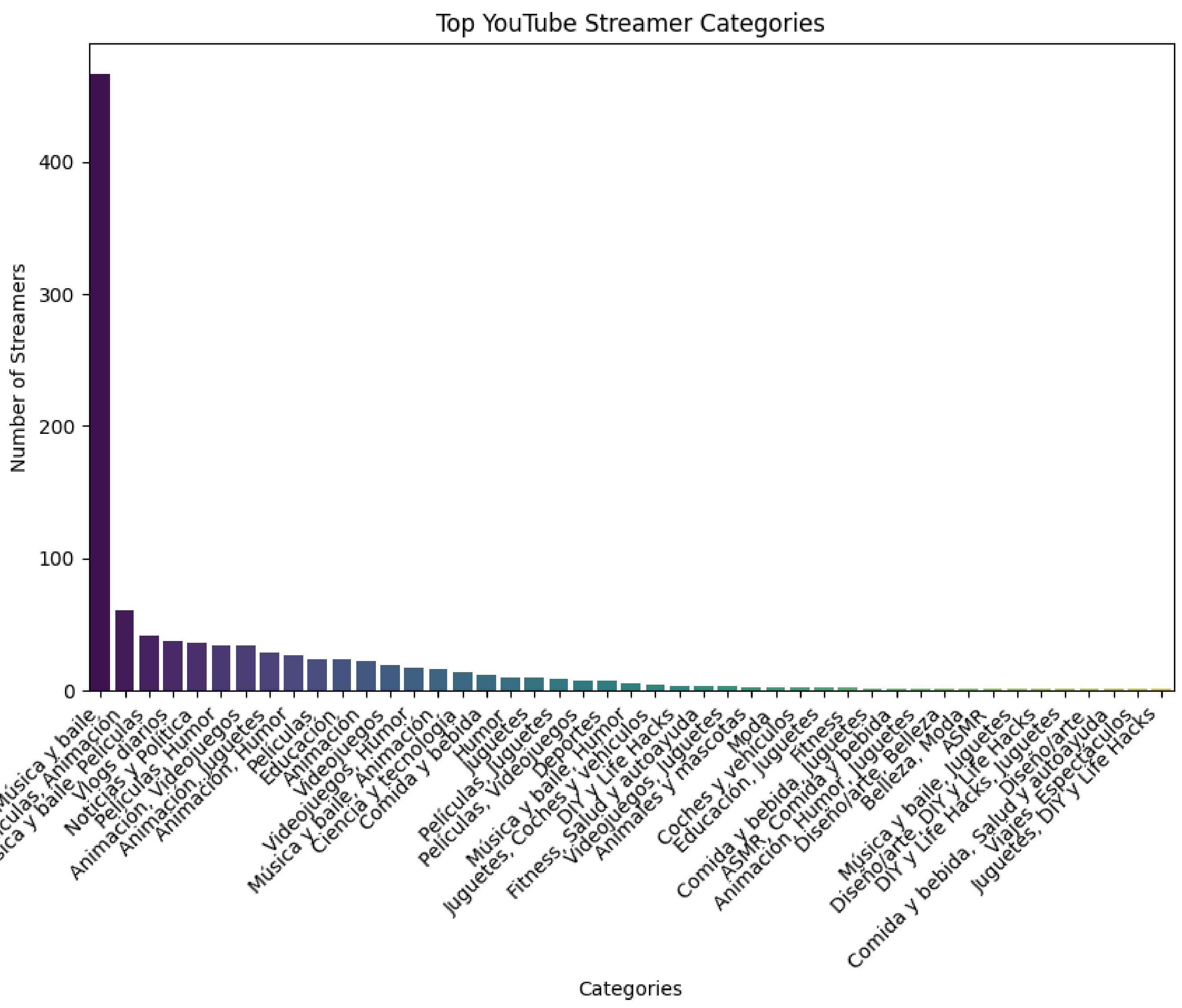
In [26]:

```python
# Visualize missing data using a heatmap
plt.figure(figsize=(10, 6))
sns.heatmap(df.isnull())
plt.title("Missing Data Heatmap")
plt.show()
```



In [27]:

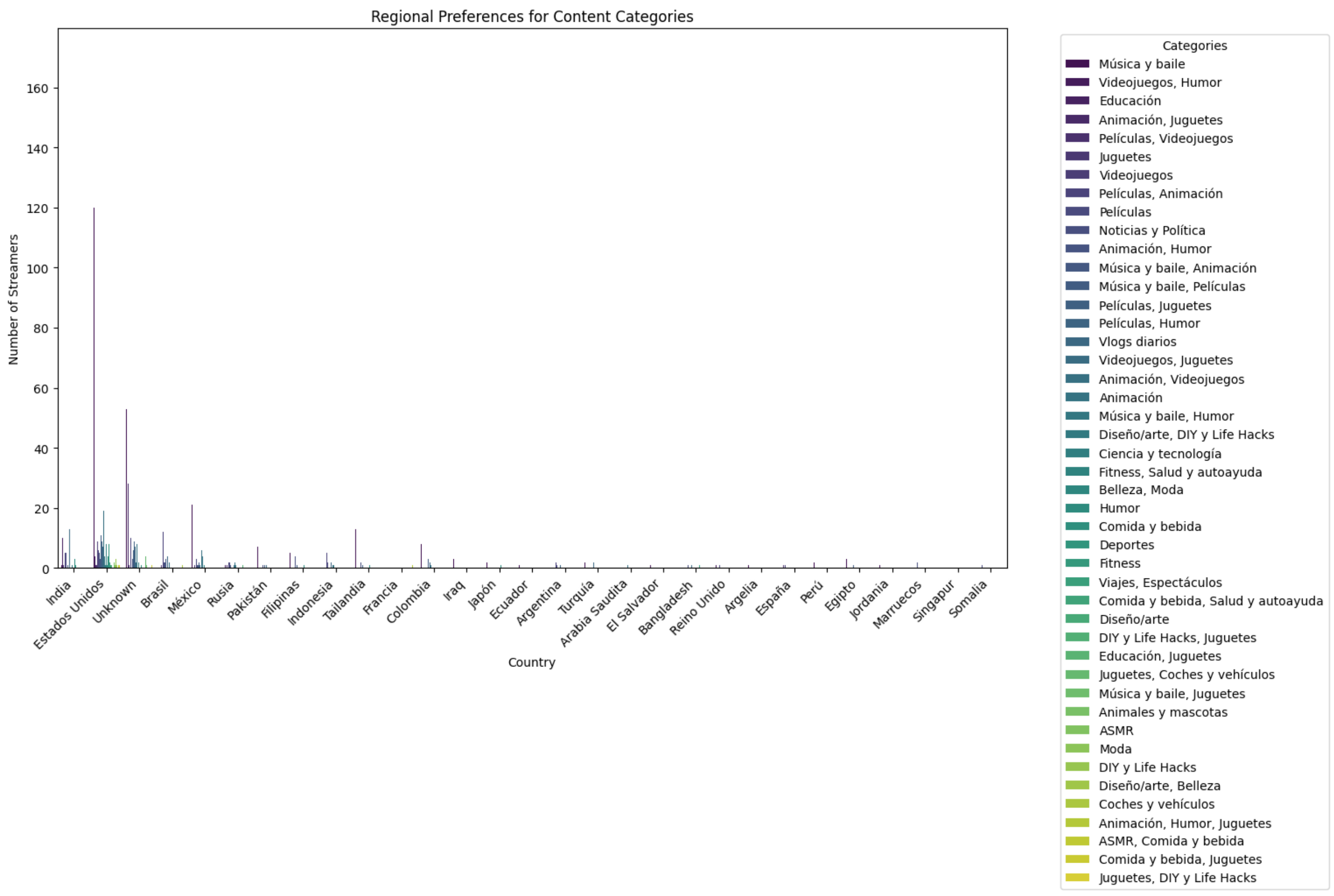
```
def analyze_category_trends(data):
    # Identify trends among the top YouTube streamers' categories
    category_counts = data['Categories'].value_counts()

    # Plot the most popular categories
    plt.figure(figsize=(10, 6))
    sns.barplot(x=category_counts.index, y=category_counts.values, palette='viridis')
    plt.title('Top YouTube Streamer Categories')
    plt.xlabel('Categories')
    plt.ylabel('Number of Streamers')
    plt.xticks(rotation=45, ha='right')
    plt.show()

def analyze_correlation(data):
    # Is there a correlation between subscribers and likes or comments?
    correlation_visits_subscribers = data['Suscribers'].corr(data['Visits'])
    correlation_likes_subscribers = data['Suscribers'].corr(data['Likes'])
    correlation_comments_subscribers = data['Suscribers'].corr(data['Comments'])
    print(f'Correlation between Subscribers and Visits: {correlation_visits_subscribers}
    print(f'Correlation between Subscribers and Likes: {correlation_likes_subscribers}')
    print(f'Correlation between Subscribers and Comments: {correlation_comments_subscrib

if __name__ == "__main__":
    analyze_category_trends(df)
    analyze_correlation(df)
```



```
Correlation between Subscribers and Visits: 0.24520315826666694
Correlation between Subscribers and Likes: 0.21163868364873312
Correlation between Subscribers and Comments: 0.03634982618984938
```

In [28]:

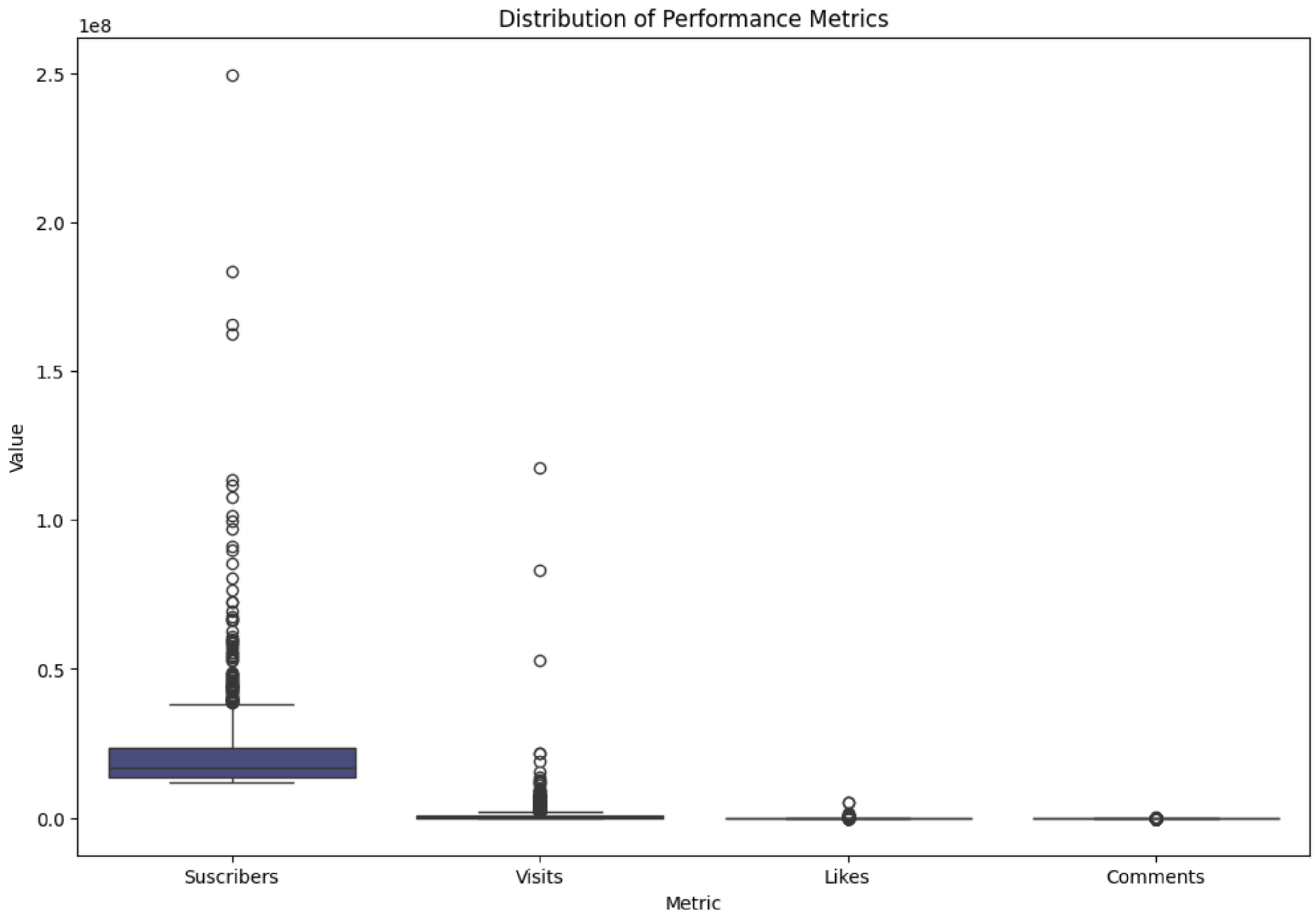
```python
def analyze_regional_preferences(data):
    # Analyze regional preferences for specific content categories
    plt.figure(figsize=(14, 8))
    sns.countplot(x='Country', hue='Categories', data=data, palette='viridis')
    plt.title('Regional Preferences for Content Categories')
    plt.xlabel('Country')
    plt.ylabel('Number of Streamers')
    plt.xticks(rotation=45, ha='right')
    plt.legend(title='Categories', bbox_to_anchor=(1.05, 1), loc='upper left')
    plt.show()
if __name__ == "__main__":
   analyze_regional_preferences(df)
```



In [29]:

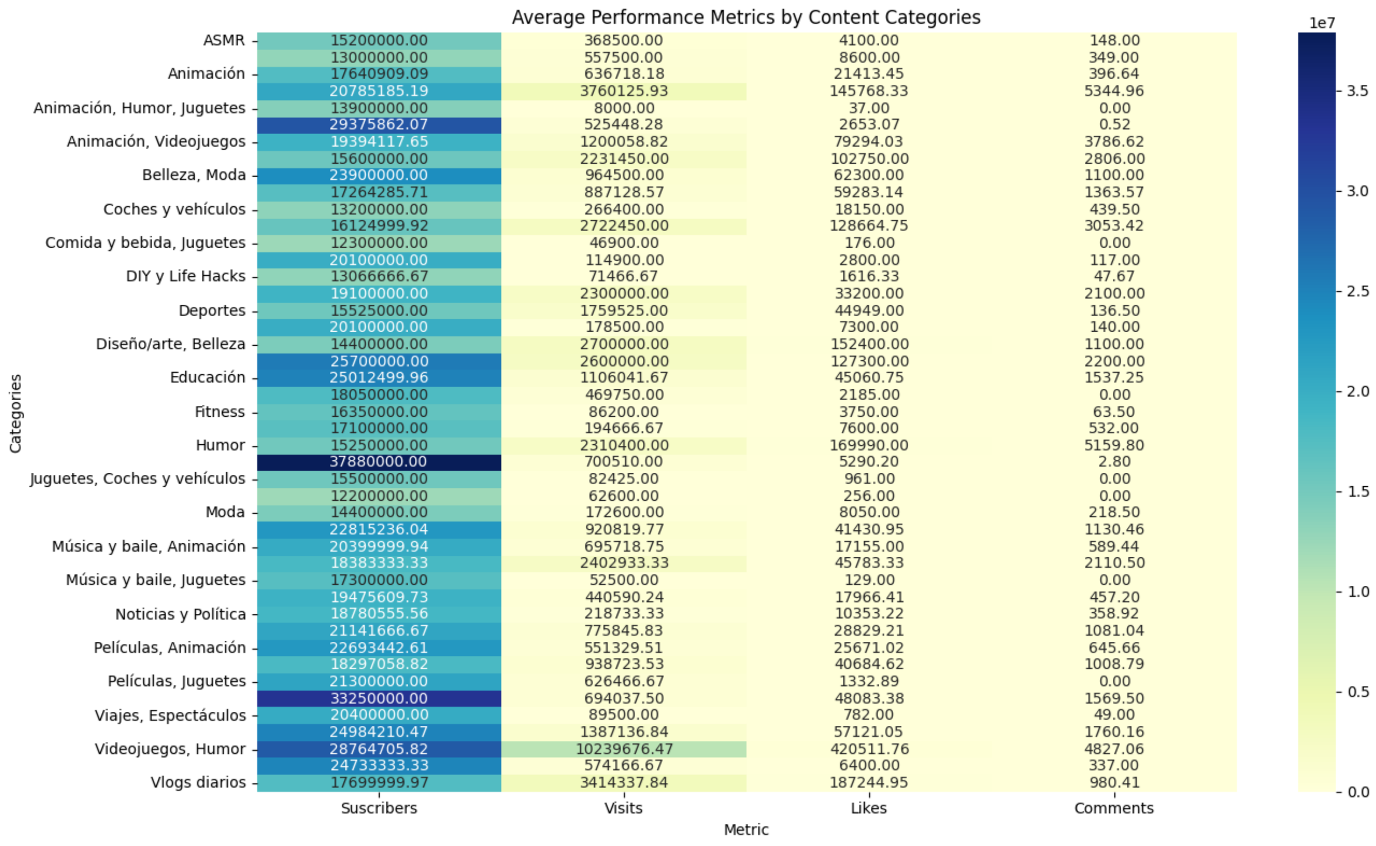
```python
def calculate_and_visualize_metrics(data):
    # Calculate average metrics
    avg_suscribers = data['Suscribers'].mean()
    avg_visits = data['Visits'].mean()
    avg_likes = data['Likes'].mean()
    avg_comments = data['Comments'].mean()
    print(f'Average Suscribers: {avg_suscribers:0.2f}')
    print(f'Average Visits: {avg_visits:0.2f}')
    print(f'Average Likes: {avg_likes:0.2f}')
    print(f'Average Comments: {avg_comments:0.2f}')

    # Visualize metrics
    plt.figure(figsize=(12, 8))
    metrics_data = data[['Suscribers', 'Visits', 'Likes', 'Comments']]
    sns.boxplot(x="variable", y="value", data=pd.melt(metrics_data), palette='viridis')
    plt.title('Distribution of Performance Metrics')
    plt.xlabel('Metric')
    plt.ylabel('Value')
    plt.show()
if __name__ == "__main__":
 calculate_and_visualize_metrics(df)
```

```
Average Suscribers: 21894399.99
Average Visits: 1209446.31
Average Likes: 53632.59
Average Comments: 1288.77
```



In [41]:

```
def identify_high_performing_categories(data):
    # Convert numeric columns to numeric type, coercing errors to NaN
    numeric_cols = ['Suscribers', 'Visits', 'Likes', 'Comments']
    data[numeric_cols] = data[numeric_cols].apply(pd.to_numeric, errors='coerce')

    # Drop rows with NaN values (if needed)
    data.dropna(subset=numeric_cols, inplace=True)

    # Identify categories with the highest average performance metrics
    avg_metrics_by_category = data.groupby('Categories')[numeric_cols].mean()

    # Visualize the highest performing categories
    plt.figure(figsize=(14, 8))
    sns.heatmap(avg_metrics_by_category, annot=True, fmt=".2f", cmap="YlGnBu")
    plt.title('Average Performance Metrics by Content Categories')
    plt.xlabel('Metric')
    plt.ylabel('Categories')
    plt.tight_layout()
    plt.show()

if __name__ == "__main__":
    # Assuming df is your DataFrame
    identify_high_performing_categories(df)
```



In [42]:

```python
def benchmark_top_performers(data):
    # Calculate average values for each metric
    avg_subscribers = data['Suscribers'].mean()
    avg_visits = data['Visits'].mean()
    avg_likes = data['Likes'].mean()
    avg_comments = data['Comments'].mean()

    # Create a dictionary with average values
    avg_metrics = {
        'Average Subscribers': avg_subscribers,
        'Average Visits': avg_visits,
        'Average Likes': avg_likes,
        'Average Comments': avg_comments
    }

    return avg_metrics

if __name__ == "__main__":
    # Assuming df is your DataFrame
    top_performers = benchmark_top_performers(df)

    # Display the top-performing content creators
    print("Top-Performing Content Creators:")
    for metric, value in top_performers.items():
        print(f"{metric}: {value}")
```

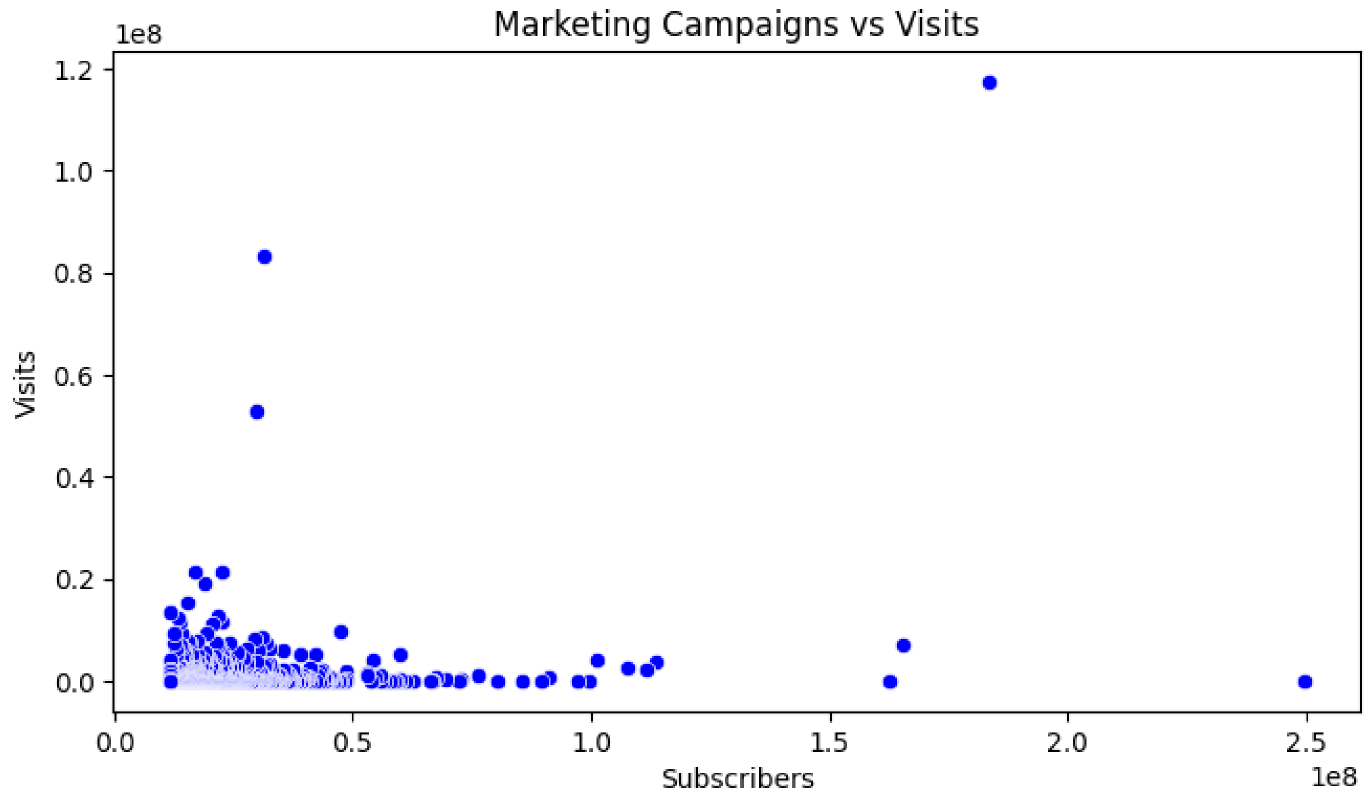
```
Top-Performing Content Creators:
Average Subscribers: 21894399.987
Average Visits: 1209446.315
Average Likes: 53632.592
Average Comments: 1288.768
```

In [43]:

```
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics.pairwise import cosine_similarity
```

In [44]:

```
import matplotlib.pyplot as plt
import seaborn as sns

def analyze_marketing_campaigns(data):
    plt.figure(figsize=(14, 8))

    # Scatter plot of Subscribers vs Visits
    plt.subplot(2, 2, 1)
    sns.scatterplot(x='Suscribers', y='Visits', data=data, color='blue')
    plt.title('Marketing Campaigns vs Visits')
    plt.xlabel('Subscribers')
    plt.ylabel('Visits')

    # Additional plots can be added similarly

    plt.tight_layout()
    plt.show()

if __name__ == "__main__":
    # Assuming df is your DataFrame
    analyze_marketing_campaigns(df)
```



In [45]:

```
def normalize_metrics(data):
    # Normalize performance metrics using Min-Max scaling
    scaler = MinMaxScaler()
    metrics_scaled = scaler.fit_transform(data[['Suscribers', 'Visits', 'Likes', 'Commen
    data[['Suscribers', 'Visits', 'Likes', 'Comments']] = metrics_scaled
    return data
```

```
def content_recommendations(username, data):
    # Normalize metrics
    normalized_data = normalize_metrics(data)

    # Check if the user exists in the dataset
    if username not in data['Username'].values:
        print(f"User {username} not found in the dataset.")
        return None

    # Extract the categories subscribed by the user
    user_categories = data.loc[data['Username'] == username, 'Categories'].values[0]

    # Create a user profile based on the average metrics of the user's subscribed catego
    user_profile = normalized_data[normalized_data['Categories'].str.split(',').apply(la
    user_profile = user_profile[['Suscribers', 'Visits', 'Likes', 'Comments']].mean()

    # Calculate cosine similarity between user profile and streamers' metrics
    similarity_scores = cosine_similarity([user_profile], normalized_data[['Suscribers',

    # Add similarity scores to the DataFrame
    data['SimilarityScore'] = similarity_scores[0]

    # Recommend top content creators based on similarity scores
    top_recommendations = data[data['Username'] != username].sort_values(by='SimilarityS

    return top_recommendations
```

In [ ]: