```
from google.colab import files
import pandas as pd

# Prompt the file upload dialog
uploaded = files.upload()

# List the uploaded files and read the CSV file
for filename in uploaded.keys():
    print(f'User uploaded file "{filename}" with length {len(uploaded[filename])} bytes')
    # Read the uploaded file
    df = pd.read_csv(filename)
```

Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Loading the data
df = pd.read_csv(r"youtubers_df.csv")

# Data exploration
print(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   int64
 1   Username    1000 non-null   object
 2   Categories  694 non-null    object
 3   Suscribers  1000 non-null   float64
 4   Country     1000 non-null   object
 5   Visits      1000 non-null   int64
 6   Likes       1000 non-null   int64
 7   Comments    1000 non-null   int64
 8   Links       1000 non-null   object
dtypes: float64(1), int64(4), object(4)
memory usage: 70.4+ KB
None
```

```
print(df.describe())
```

```
               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
```

```
df.shape
```

```
(1000, 9)
```

```
print(df.isnull().sum())
```

```
Rank            0
Username        0
Categories    306
Suscribers      0
Country         0
Visits          0
Likes           0
Comments        0
Links           0
dtype: int64
```

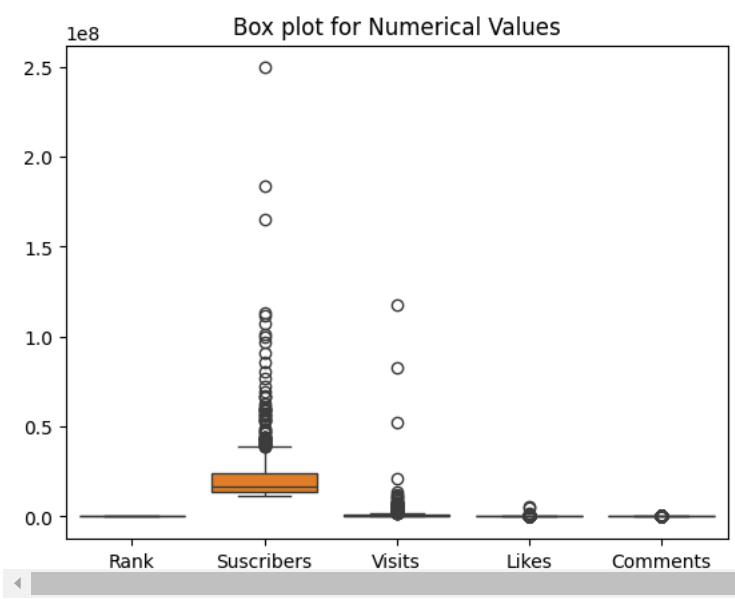
```
#Drop Rows with missing values
df = df.dropna()
```

```
df.isnull().sum()
```

	0
Rank	0
Username	0
Categories	0
Suscribers	0
Country	0
Visits	0
Likes	0
Comments	0
Links	0

dtype: int64

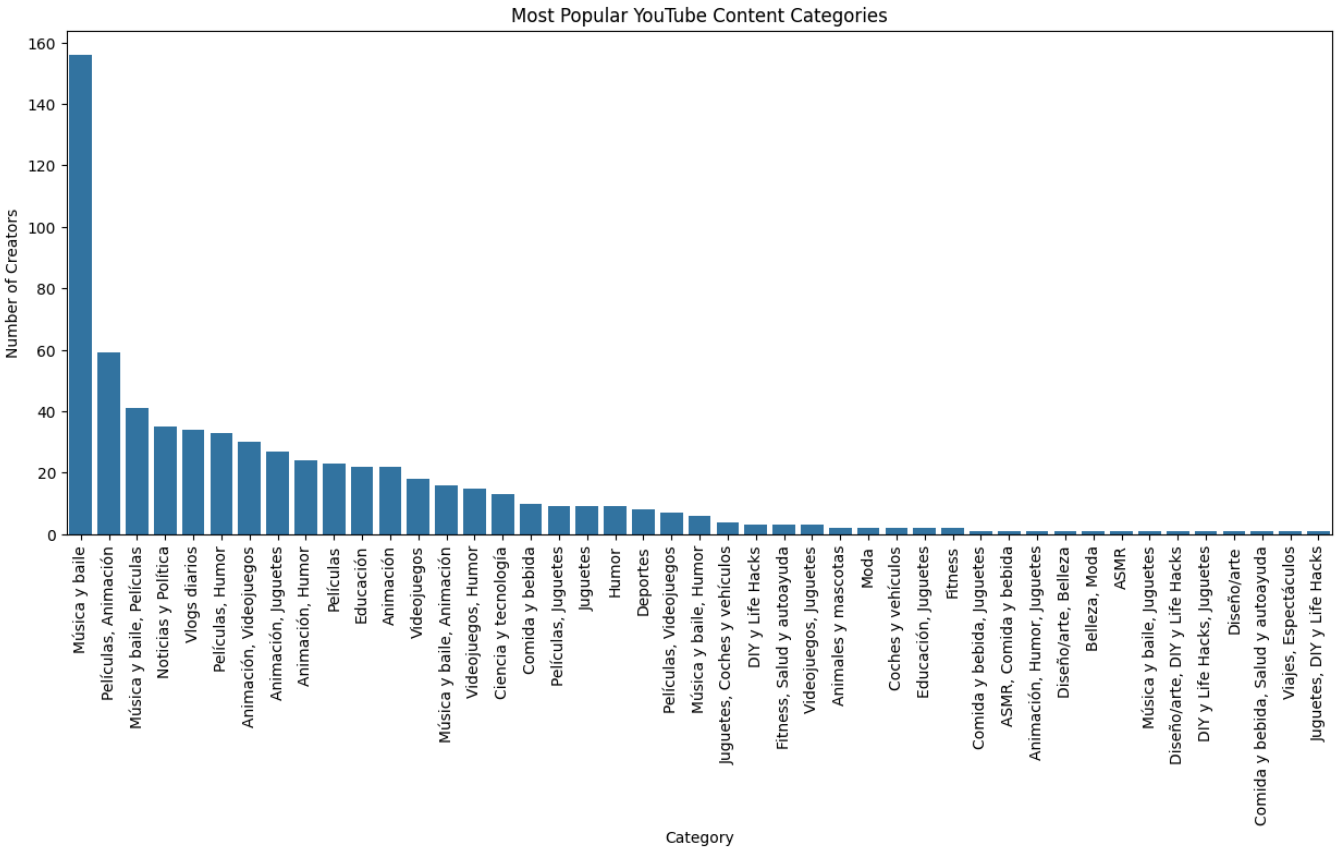
```
# Check for outliers using boxplots (For numerical values)
sns.boxplot(data=df)
plt.title("Box plot for Numerical Values")
plt.show()
```




```
# Function to remove outliers using z-score method
from scipy.stats import zscore
def remove_outliers_zscore(dataframe, columns, threshold=3):
    df_no_outliers = dataframe.copy()
    for column in columns:
        z_scores = zscore(df_no_outliers[column])
        df_no_outliers = df_no_outliers[(z_scores < threshold) & (z_scores > -threshold)]
    return df_no_outliers

# Remove outliers
columns_to_clean = ['Suscribers', 'Visits', 'Likes', 'Rank']
df = remove_outliers_zscore(df, columns_to_clean)
df.head()
```

	Rank	Username	Categories	Suscribers	Country	Visits	Likes	Comments	L
39	40	JuegaGerman	Películas, Animación	48600000.0	México	2000000	117100	3000	http://youtube.com/channel/UCYiGq8XF7YQD00x7
40	41	BillieEilish	Música y baile	48600000.0	Estados Unidos	208100	27200	476	http://youtube.com/channel/UCiGm_E4ZwYSHV3bc
42	43	sonymusicindiaVEVO	Música y baile	47500000.0	India	36600	664	28	http://youtube.com/channel/UC3MLnJtqc_phABE

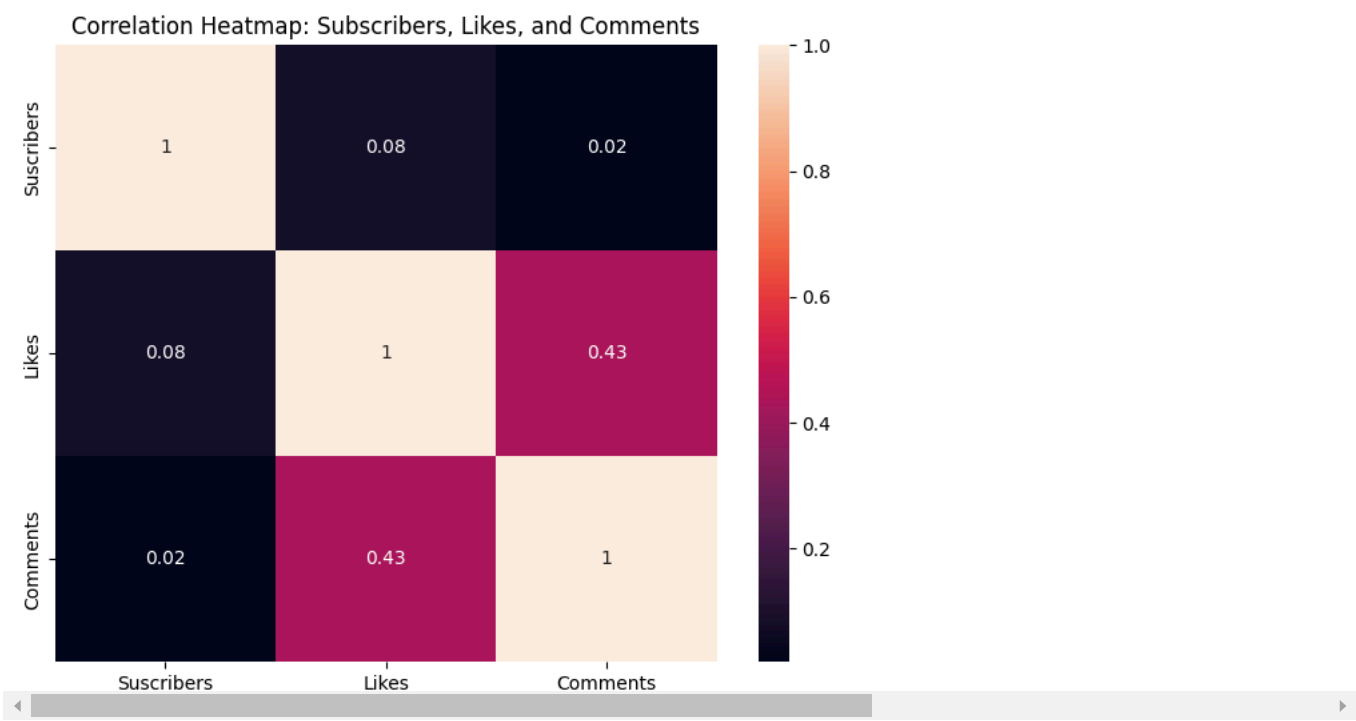
```
#To identify the most popular categories
category_counts = df['Categories'].value_counts()
plt.figure(figsize=(15,6))
sns.barplot(x=category_counts.index, y=category_counts.values)
plt.xticks(rotation=90)
plt.title("Most Popular YouTube Content Categories")
plt.xlabel("Category")
plt.ylabel("Number of Creators")
plt.show()
```




```
#The Correlation between Subscribers, Likes, and Comments
corr_likes = np.corrcoef(df['Suscribers'], df['Likes']) [0,1]
corr_comments = np.corrcoef(df['Suscribers'], df['Comments']) [0,1]
print(f'Correlation b/w Subscribers and Likes: {corr_likes}')
print(f'Correlation b/w Subscribers and Comments: {corr_comments}')
```

```
Correlation b/w Subscribers and Likes: 0.07958997301642444
Correlation b/w Subscribers and Comments: 0.01990529146597779
```

```
#The Correlation between Subscribers, Likes, and Comments
correlation = df[['Suscribers', 'Likes', 'Comments']].corr()
plt.figure(figsize=(8,6))
sns.heatmap(correlation, annot=True)
plt.title("Correlation Heatmap: Subscribers, Likes, and Comments")
plt.show()
```




```
#Audience Study
#To count the number of creators in each combination
category_country_counts = df.groupby(['Categories', 'Country'])['Categories'].count().reset_index(name='Count')
print(category_country_counts)
```

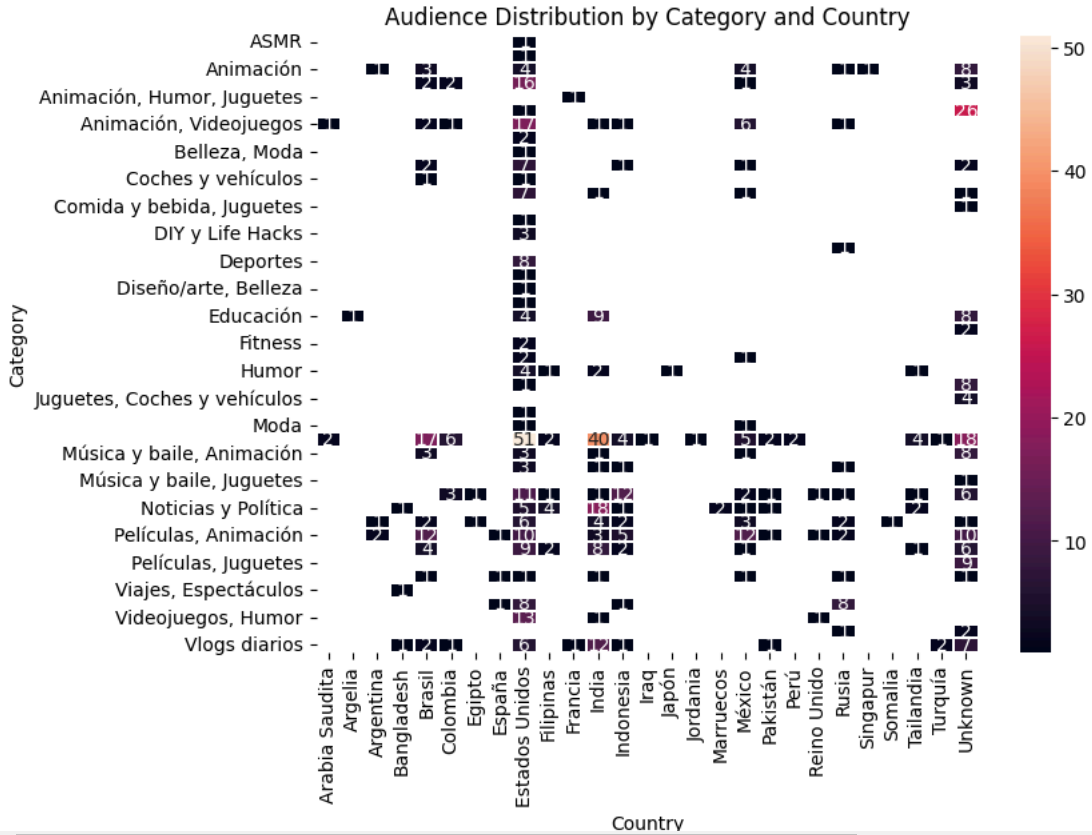
```
                Categories         Country  Count
0                     ASMR  Estados Unidos      1
1    ASMR, Comida y bebida  Estados Unidos      1
2               Animación       Argentina      1
3               Animación          Brasil      3
4               Animación  Estados Unidos      4
..                    ...             ...    ...
163          Vlogs diarios           India     12
164          Vlogs diarios       Indonesia      1
165          Vlogs diarios        Pakistán      1
166          Vlogs diarios         Turquía      2
167          Vlogs diarios         Unknown      7

[168 rows x 3 columns]
```

```
country_visit_count = df.groupby('Country')['Visits'].sum().reset_index()
country_visit_count
```

	Country	Visits
0	Arabia Saudita	3474500
1	Argelia	333500
2	Argentina	6371400
3	Bangladesh	100700
4	Brasil	18643600
5	Colombia	6256400
6	Egipto	305400
7	España	1984300
8	Estados Unidos	207221500
9	Filipinas	8914600
10	Francia	5308000
11	India	35783000
12	Indonesia	11556900
13	Iraq	103600
14	Japón	2100000
15	Jordania	267600
16	Marruecos	12000
17	México	31365000
18	Pakistán	1969600
19	Perú	2219400
20	Reino Unido	9250700
21	Rusia	23299600
22	Singapur	26400
23	Somalia	1900000
24	Tailandia	2553800
25	Turquía	1604900
26	Unknown	47211700

```
#Create a pivot table for better visualization
pivot_table = category_country_counts.pivot(index='Categories',columns='Country', values='Count')
```

```
#Heatmap for streamers audiences by country and category
plt.figure(figsize=(8,6))
sns.heatmap(pivot_table, annot=True, linewidths=0.3)
plt.title("Audience Distribution by Category and Country")
plt.xlabel("Country")
plt.ylabel("Category")
plt.show()
```




```
#Benchmarking
#Calculate average values for each metric
avg_subscribers = df['Suscribers'].mean()
avg_visits = df['Visits'].mean()
avg_likes = df['Likes'].mean()
avg_comments = df['Comments'].mean()
print(f'Average Subscribers: {avg_subscribers}')
print(f'Average Visits: {avg_visits}')
print(f'Average Likes: {avg_likes}')
print(f'Average Comments: {avg_comments}')
```

```
Average Subscribers: 20573867.069486406
Average Visits: 649755.4380664653
Average Likes: 24143.44259818731
Average Comments: 1016.6510574018126
```

```
# Identify streamers with above-average performance
above_avg_streamers = df[(df['Suscribers'] > avg_subscribers) &(df['Visits'] > avg_visits) &(df['Likes'] > avg_likes) &(df['Comments'] :
print(above_avg_streamers)
```

```
     Rank            Username                     Categories  Suscribers  \
14     15                 BTS                 Música y baile  76500000.0
26     27         dudeperfect                    Videojuegos  59700000.0
34     35         TaylorSwift                 Música y baile  54100000.0
37     38        ArianaGrande                 Música y baile  52900000.0
39     40         JuegaGerman           Películas, Animación  48600000.0
58     59           Mikecrack           Películas, Animación  43400000.0
62     63      KimberlyLoaiza                 Música y baile  42100000.0
70     71         JessNoLimit           Películas, Animación  39600000.0
96     97       TotalGaming093         Películas, Videojuegos  36300000.0
100   101           markiplier         Animación, Videojuegos  35500000.0
109   110        SSSniperWolf               Animación, Humor  34200000.0
145   146       jacksepticeye               Animación, Humor  30400000.0
171   172     SandeepSeminars                  Vlogs diarios  28000000.0
177   178              DanTDM         Animación, Videojuegos  27800000.0
179   180         brentrivera             Videojuegos, Humor  27600000.0
180   181            NichLmao                  Vlogs diarios  27500000.0
195   196          nickiminaj                 Música y baile  26100000.0
202   203        VanossGaming         Animación, Videojuegos  25900000.0
206   207           AlejoIgoa                     Animación  25700000.0
207   208               ZHCYT  Diseño/arte, DIY y Life Hacks  25700000.0
234   235                 rug             Videojuegos, Humor  24300000.0
241   242  juandediospantojaa      Música y baile, Películas  24000000.0
243   244        JamesCharles                  Belleza, Moda  23900000.0
272   273       AmiRodrigueZZ               Animación, Humor  22900000.0
278   279         StokesTwins             Videojuegos, Humor  22700000.0
281   282             SSundee         Animación, Videojuegos  22700000.0
285   286          BenAzelart             Videojuegos, Humor  22500000.0
```

```
302   303          royaltyfam                          Humor  21900000.0
304   305            infinite             Videojuegos, Humor  21700000.0
315   316      lyricalemonade      Música y baile, Animación  21100000.0
318   319          kurzgesagt                      Educación  21100000.0
319   320               romeo                  Música y baile  21100000.0

            Country    Visits   Likes  Comments  \
14            India    969700  180300      7400
26   Estados Unidos   5300000  156500      4200
34   Estados Unidos   4300000  300400     15000
37   Estados Unidos   1100000   85800      3800
39           México   2000000  117100      3000
58           México   2200000  183400      1800
62           México   5300000  271300     16000
70        Indonesia   1300000   73500      1600
96            India   1500000  129400      4900
100  Estados Unidos   2100000  126500      3800
109  Estados Unidos   1200000   34600      2100
145  Estados Unidos   1600000   83400      2300
171           India   1200000   58500      4000
177  Estados Unidos   3500000  285000     52500
179  Estados Unidos   6400000  154100      5000
180  Estados Unidos   1500000   85800      1600
195  Estados Unidos   1600000   98300      7600
202  Estados Unidos   1300000   56500      1100
206       Argentina   5700000  208400      1700
207  Estados Unidos   2600000  127300      2200
234  Estados Unidos   3200000   85300      5100
241          México   3000000  133200      3600
243  Estados Unidos    964500   62300      1100
```

```
#Display information about the top-performing streamers
top_performing_streamers = above_avg_streamers.sort_values(by=['Suscribers'],ascending=True)
print("Top Performing Streamers:")
print(top_performing_streamers[['Username', 'Suscribers', 'Visits', 'Likes','Comments']])
```

```
Top Performing Streamers:
                 Username  Suscribers    Visits   Likes  Comments
319               romeo  21100000.0   3200000   53900      1600
315      lyricalemonade  21100000.0   2800000  127300      5800
318          kurzgesagt  21100000.0   4900000  253500     14000
304            infinite  21700000.0    884800   45700      1400
302          royaltyfam  21900000.0   4700000   67000      6600
285          BenAzelart  22500000.0   3700000   44900      2700
281             SSundee  22700000.0   1700000   59800      1800
278         StokesTwins  22700000.0  11700000  235000     10000
272       AmiRodrigueZZ  22900000.0   4300000  294400      1300
243        JamesCharles  23900000.0    964500   62300      1100
241  juandediospantojaa  24000000.0   3000000  133200      3600
234                 rug  24300000.0   3200000   85300      5100
207               ZHCYT  25700000.0   2600000  127300      2200
206           AlejoIgoa  25700000.0   5700000  208400      1700
202        VanossGaming  25900000.0   1300000   56500      1100
195          nickiminaj  26100000.0   1600000   98300      7600
180            NichLmao  27500000.0   1500000   85800      1600
179         brentrivera  27600000.0   6400000  154100      5000
177              DanTDM  27800000.0   3500000  285000     52500
171     SandeepSeminars  28000000.0   1200000   58500      4000
145       jacksepticeye  30400000.0   1600000   83400      2300
109        SSSniperWolf  34200000.0   1200000   34600      2100
100          markiplier  35500000.0   2100000  126500      3800
96       TotalGaming093  36300000.0   1500000  129400      4900
70          JessNoLimit  39600000.0   1300000   73500      1600
62       KimberlyLoaiza  42100000.0   5300000  271300     16000
58            Mikecrack  43400000.0   2200000  183400      1800
39          JuegaGerman  48600000.0   2000000  117100      3000
37         ArianaGrande  52900000.0   1100000   85800      3800
34          TaylorSwift  54100000.0   4300000  300400     15000
26          dudeperfect  59700000.0   5300000  156500      4200
14                  BTS  76500000.0    969700  180300      7400
```

```
#Distribution of Categories
#To calculate category counts
category_counts = df['Categories'].value_counts()
print(category_counts)
```

```
Categories
Música y baile                        156
Películas, Animación                   59
Música y baile, Películas              41
Noticias y Política                    35
Vlogs diarios                          34
Películas, Humor                       33
Animación, Videojuegos                 30
Animación, Juguetes                    27
```

```
Animación, Humor                      24
Películas                             23
Educación                             22
Animación                             22
Videojuegos                           18
Música y baile, Animación             16
Videojuegos, Humor                    15
Ciencia y tecnología                  13
Comida y bebida                       10
Películas, Juguetes                    9
Juguetes                               9
Humor                                  9
Deportes                               8
Películas, Videojuegos                 7
Música y baile, Humor                  6
Juguetes, Coches y vehículos           4
DIY y Life Hacks                       3
Fitness, Salud y autoayuda             3
Videojuegos, Juguetes                  3
Animales y mascotas                    2
Moda                                   2
Coches y vehículos                     2
Educación, Juguetes                    2
Fitness                                2
Comida y bebida, Juguetes              1
ASMR, Comida y bebida                  1
Animación, Humor, Juguetes             1
Diseño/arte, Belleza                   1
Belleza, Moda                          1
ASMR                                   1
Música y baile, Juguetes               1
Diseño/arte, DIY y Life Hacks          1
DIY y Life Hacks, Juguetes             1
Diseño/arte                            1
Comida y bebida, Salud y autoayuda     1
Viajes, Espectáculos                   1
Juguetes, DIY y Life Hacks             1
Name: count, dtype: int64
```

```
#Create a bar plot to visualize the number of streamers in each category
plt.figure(figsize=(10,8))
sns.barplot(x=category_counts.index, y=category_counts.values, palette="flare")
plt.title("Streamers in each content category")
plt.xlabel("Category")
10
plt.ylabel("Number of streamers")
plt.xticks(rotation=90)
```

```
<ipython-input-48-441f3c4bace1>:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set

  sns.barplot(x=category_counts.index, y=category_counts.values, palette="flare")
([0,
  1,
  2,
  3,
  4,
  5,
  6,
  7,
  8,
  9,
  10,
  11,
  12,
  13,
  14,
  15,
  16,
  17,
  18,
  19,
  20,
  21,
  22,
  23,
  24,
  25,
  26,
  27,
  28,
  29,
  30,
  31,
  32,
  33,
  34,
  35,
  36,
  37,
  38,
  39,
  40,
  41,
  42,
  43,
  44],
 [Text(0, 0, 'Música y baile'),
  Text(1, 0, 'Películas, Animación'),
  Text(2, 0, 'Música y baile, Películas'),
  Text(3, 0, 'Noticias y Política'),
  Text(4, 0, 'Vlogs diarios'),
  Text(5, 0, 'Películas, Humor'),
  Text(6, 0, 'Animación, Videojuegos'),
```

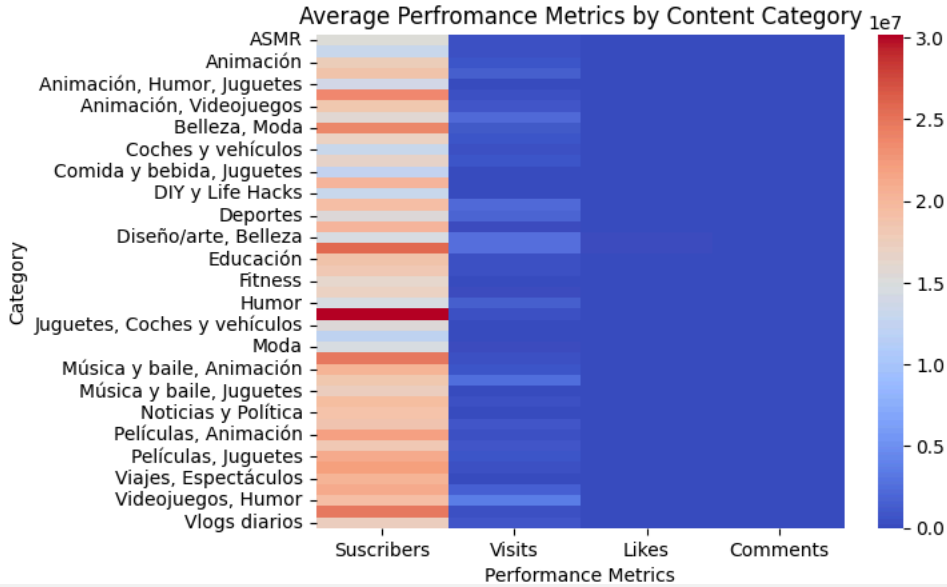
```
#Identify the category with the highest number of streamers
max_category = category_counts.idxmax()
max_count = category_counts.max()
12
print(f"The category with the highest number of streamers is '{max_category}'␣with '{max_count}' streamers")
```

```
The category with the highest number of streamers is 'Música y baile'␣with '156' streamers
```

```
 Text(17, 0, 'Películas, Juguetes'),
```

```
 #Calculate average metrics by category
ambc= df.groupby('Categories')[['Suscribers', 'Visits', 'Likes', 'Comments']].mean()
#Visualize the average performance metrics by category
sns.heatmap(ambc, cmap="coolwarm", fmt=".2f")
plt.title("Average Perfromance Metrics by Content Category")
plt.xlabel("Performance Metrics")
plt.ylabel("Category")
```

```
Text(50.2222222222222, 0.5, 'Category')
```

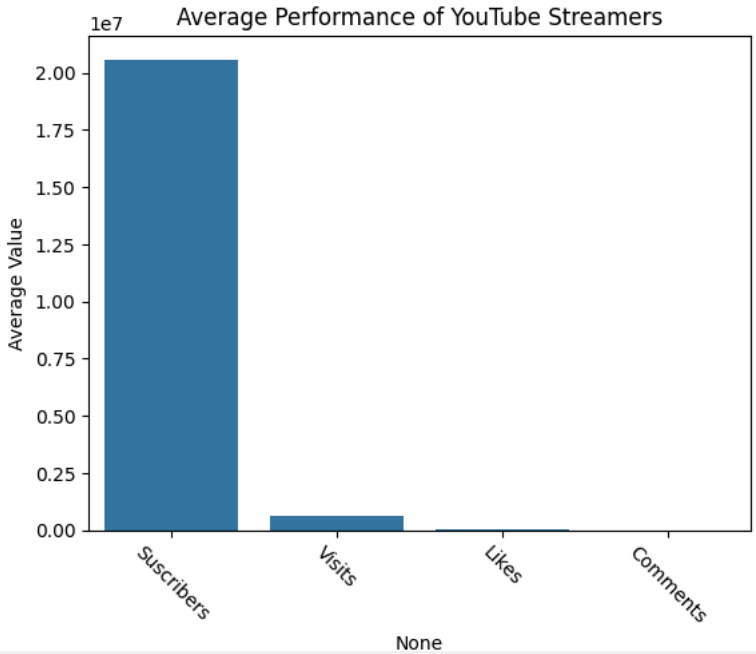



```
#Performance Metrics
# To caculate the average metrics
average_metrics= df[['Suscribers', 'Visits', 'Likes', 'Comments']].mean()
print(average_metrics)
```

```
Suscribers    2.057387e+07
Visits        6.497554e+05
Likes         2.414344e+04
Comments      1.016651e+03
dtype: float64
```

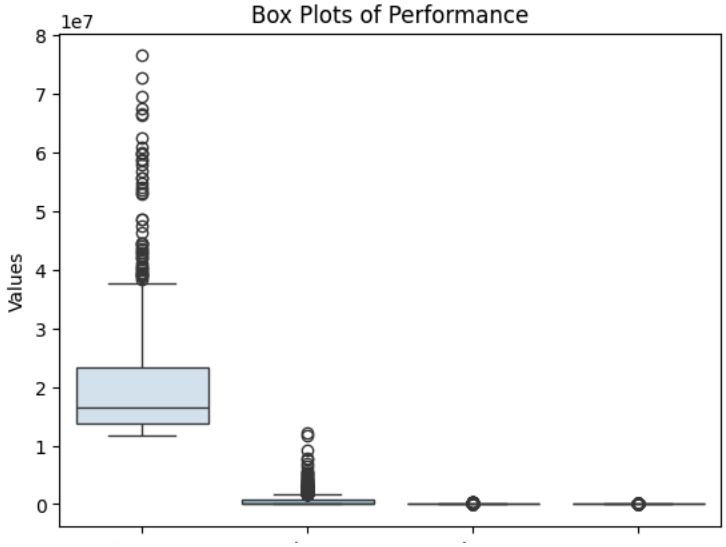
```
#Barplot of average metrics
sns.barplot(x=average_metrics.index, y=average_metrics.values)
plt.title("Average Performance of YouTube Streamers")
plt.ylabel("Average Value")
plt.xticks(rotation=-45)
```

```
([0, 1, 2, 3],
 [Text(0, 0, 'Suscribers'),
  Text(1, 0, 'Visits'),
  Text(2, 0, 'Likes'),
  Text(3, 0, 'Comments')])
```



```
#For patterns in the metrics
sns.boxplot(data=df[['Suscribers', 'Visits', 'Likes', 'Comments']],palette="Blues")
plt.title("Box Plots of Performance")
plt.ylabel("Values")
plt.xticks(rotation=-45)
```

```
([0, 1, 2, 3],
 [Text(0, 0, 'Suscribers'),
  Text(1, 0, 'Visits'),
  Text(2, 0, 'Likes'),
  Text(3, 0, 'Comments')])
```




```
#Content Recommendations
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
from sklearn.preprocessing import LabelEncoder
from sklearn.linear_model import LinearRegression
```

```
df['user_id']=range(1,len(df['Username'])+1)
```

```
x=df[['Rank', 'Visits', 'Comments']]
y=df['user_id']
```

```
x_train,x_test, y_train, y_test= train_test_split(x,y,test_size=0.3)
```

```
model = LinearRegression()
```