Dataset Link:: https://www.kaggle.com/datasets/rahuldogra/top5000youtubechannels

Questions to be answered in thiss project

1. Display All Rows Except the Last 5 rows Using Head Method
2. Display All Rows Except the First 5 Rows Using Tail Method
3. Find Shape of Our Dataset (Number of Rows And Number of Columns)
4. Get Information About Our Dataset Like Total Number Rows, Total Number of Columns, Datatypes of Each Column And Memory Requirement
5. Get Overall Statistics About The Dataframe
6. Data Cleaning (Replace '--' to NaN)
7. Check Null Values In The Dataset
8. Data Cleaning [Rank Column]
9. Data Cleaning [Video Uploads & Subscribers]
10. Data Cleaning [Grade Column]
11. Find Average Views For Each Channel
12. Find Out Top Five Channels With Maximum Number of Video Uploads
13. Find Correlation Matrix

14. Which Grade Has A Maximum Number of Video Uploads? 15.Which Grade Has The Highest Average Views?
15. Which Grade Has The Highest Number of Subscribers?
16. Which Grade Has The Highest Video Views?

In [1]:
```
# Import the dataset and required libraries
```

In [2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [3]:
```
df = pd.read_csv("top-5000-youtube-channels.csv")
```

In [4]:
```
df.head()
```

Out[4]:

	Rank	Grade	Channel name	Video Uploads	Subscribers	Video views
0	1st	A++	Zee TV	82757	18752951	20869786591
1	2nd	A++	T-Series	12661	61196302	47548839843
2	3rd	A++	Cocomelon - Nursery Rhymes	373	19238251	9793305082
3	4th	A++	SET India	27323	31180559	22675948293
4	5th	A++	WWE	36756	32852346	26273668433

1. Display All Rows Except the Last 5 rows Using Head Method

In [5]:
```
df.head(-5)
```

Out[5]:

	Rank	Grade	Channel name	Video Uploads	Subscribers	Video views
0	1st	A++	Zee TV	82757	18752951	20869786591
1	2nd	A++	T-Series	12661	61196302	47548839843
2	3rd	A++	Cocomelon - Nursery Rhymes	373	19238251	9793305082
3	4th	A++	SET India	27323	31180559	22675948293
4	5th	A++	WWE	36756	32852346	26273668433
...	...	...	...	...	...	...
4990	4,991st	B+	Ho Ngoc Ha's Official Channel	208	--	127185704
4991	4,992nd	B+	Toys to Learn Colors	11	663114	141933264
4992	4,993rd	B+	KAZKA	25	131766	74304638
4993	4,994th	B+	United CUBE (CUBE Entertainment...	1055	1586835	371299166
4994	4,995th	B+	Wings Marathi	1735	1099659	346175699

4995 rows × 6 columns

2. Display All Rows Except the First 5 Rows Using Tail Method

In [6]:
```
df.tail(-5)
```

Out[6]:

	Rank	Grade	Channel name	Video Uploads	Subscribers	Video views
5	6th	A++	Movieclips	30243	17149705	16618094724
6	7th	A++	netd müzik	8500	11373567	23898730764
7	8th	A++	ABS-CBN Entertainment	100147	12149206	17202609850
8	9th	A++	Ryan ToysReview	1140	16082927	24518098041
9	10th	A++	Zee Marathi	74607	2841811	2591830307
...	...	...	...	...	...	...
4995	4,996th	B+	Uras Benlioğlu	706	2072942	441202795
4996	4,997th	B+	HI-TECH MUSIC LTD	797	1055091	377331722
4997	4,998th	B+	Mastersaint	110	3265735	311758426
4998	4,999th	B+	Bruce McIntosh	3475	32990	14563764
4999	5,000th	B+	SehatAQUA	254	21172	73312511

4995 rows × 6 columns

3. Find Shape of Our Dataset (Number of Rows And Number of Columns)

In [7]:
```
df.shape
```

Out[7]:
```
(5000, 6)
```

4. Get Information About Our Dataset Like Total Number Rows, Total Number of Columns, Datatypes of Each Column And Memory Requirement

In [8]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5000 entries, 0 to 4999
Data columns (total 6 columns):
 #   Column         Non-Null Count  Dtype 
---  ------         --------------  ----- 
 0   Rank           5000 non-null   object
 1   Grade          5000 non-null   object
 2   Channel name   5000 non-null   object
 3   Video Uploads  5000 non-null   object
 4   Subscribers    5000 non-null   object
 5   Video views    5000 non-null   int64 
dtypes: int64(1), object(5)
memory usage: 234.5+ KB
```

5. Get Overall Statistics About The Dataframe

In [9]:
```
df.describe(include ='all')
```

Out[9]:

	Rank	Grade	Channel name	Video Uploads	Subscribers	Video views
count	5000	5000	5000	5000	5000	5.000000e+03
unique	5000	6	4993	2286	4612	NaN
top	5,000th	B+	Learn Colors For Kids	26	--	NaN
freq	1	2956	2	17	387	NaN
mean	NaN	NaN	NaN	NaN	NaN	1.071449e+09
std	NaN	NaN	NaN	NaN	NaN	2.003844e+09
min	NaN	NaN	NaN	NaN	NaN	7.500000e+01
25%	NaN	NaN	NaN	NaN	NaN	1.862329e+08
50%	NaN	NaN	NaN	NaN	NaN	4.820548e+08
75%	NaN	NaN	NaN	NaN	NaN	1.124368e+09
max	NaN	NaN	NaN	NaN	NaN	4.754884e+10

In [11]:
```
# Video Views are described in exponential format, we need to convert into decim

pd.options.display.float_format = '{:.2f}'.format
```

In [12]:
```
df.describe()
```

Out[12]:

	Video views
count	5000.00
mean	1071449400.15
std	2003843972.12
min	75.00
25%	186232945.75
50%	482054780.00
75%	1124367826.75
max	47548839843.00

In [13]:
```
df.describe(include ='all')
```

Out[13]:

	Rank	Grade	Channel name	Video Uploads	Subscribers	Video views
count	5000	5000	5000	5000	5000	5000.00
unique	5000	6	4993	2286	4612	NaN
top	5,000th	B+	Learn Colors For Kids	26	--	NaN
freq	1	2956	2	17	387	NaN
mean	NaN	NaN	NaN	NaN	NaN	1071449400.15
std	NaN	NaN	NaN	NaN	NaN	2003843972.12
min	NaN	NaN	NaN	NaN	NaN	75.00
25%	NaN	NaN	NaN	NaN	NaN	186232945.75
50%	NaN	NaN	NaN	NaN	NaN	482054780.00
75%	NaN	NaN	NaN	NaN	NaN	1124367826.75
max	NaN	NaN	NaN	NaN	NaN	47548839843.00

6. Data Cleaning (Replace '--' to NaN)

In [14]:
```
df.head(20)
```

Out[14]:

	Rank	Grade	Channel name	Video Uploads	Subscribers	Video views
0	1st	A++	Zee TV	82757	18752951	20869786591
1	2nd	A++	T-Series	12661	61196302	47548839843
2	3rd	A++	Cocomelon - Nursery Rhymes	373	19238251	9793305082
3	4th	A++	SET India	27323	31180559	22675948293
4	5th	A++	WWE	36756	32852346	26273668433
5	6th	A++	Movieclips	30243	17149705	16618094724
6	7th	A++	netd müzik	8500	11373567	23898730764
7	8th	A++	ABS-CBN Entertainment	100147	12149206	17202609850
8	9th	A++	Ryan ToysReview	1140	16082927	24518098041
9	10th	A++	Zee Marathi	74607	2841811	2591830307
10	11th	A+	5-Minute Crafts	2085	33492951	8587520379
11	12th	A+	Canal KondZilla	822	39409726	19291034467
12	13th	A+	Like Nastya Vlog	150	7662886	2540099931
13	14th	A+	Ozuna	50	18824912	8727783225
14	15th	A+	Wave Music	16119	15899764	10989179147
15	16th	A+	Ch3Thailand	49239	11569723	9388600275
16	17th	A+	WORLDSTARHIPHOP	4778	15830098	11102158475
17	18th	A+	Vlad and Nikita	53	--	1428274554
18	19th	A+	Badabun	3060	23603062	5860444053
19	20th	A+	WorkpointOfficial	24287	17687229	14022189654

In [15]:
```
df.replace('--', np.nan, regex = True)
```

Out[15]:

	Rank	Grade	Channel name	Video Uploads	Subscribers	Video views
0	1st	A++	Zee TV	82757	18752951	20869786591
1	2nd	A++	T-Series	12661	61196302	47548839843
2	3rd	A++	Cocomelon - Nursery Rhymes	373	19238251	9793305082
3	4th	A++	SET India	27323	31180559	22675948293
4	5th	A++	WWE	36756	32852346	26273668433
...	...	...	...	...	...	...
4995	4,996th	B+	Uras Benlioğlu	706	2072942	441202795
4996	4,997th	B+	HI-TECH MUSIC LTD	797	1055091	377331722
4997	4,998th	B+	Mastersaint	110	3265735	311758426
4998	4,999th	B+	Bruce McIntosh	3475	32990	14563764
4999	5,000th	B+	SehatAQUA	254	21172	73312511

5000 rows × 6 columns

7. Check Null Values In The Dataset

In [16]:
```
df.isnull()
```

Out[16]:

	Rank	Grade	Channel name	Video Uploads	Subscribers	Video views
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
...	...	...	...	...	...	...
4995	False	False	False	False	False	False
4996	False	False	False	False	False	False
4997	False	False	False	False	False	False
4998	False	False	False	False	False	False
4999	False	False	False	False	False	False

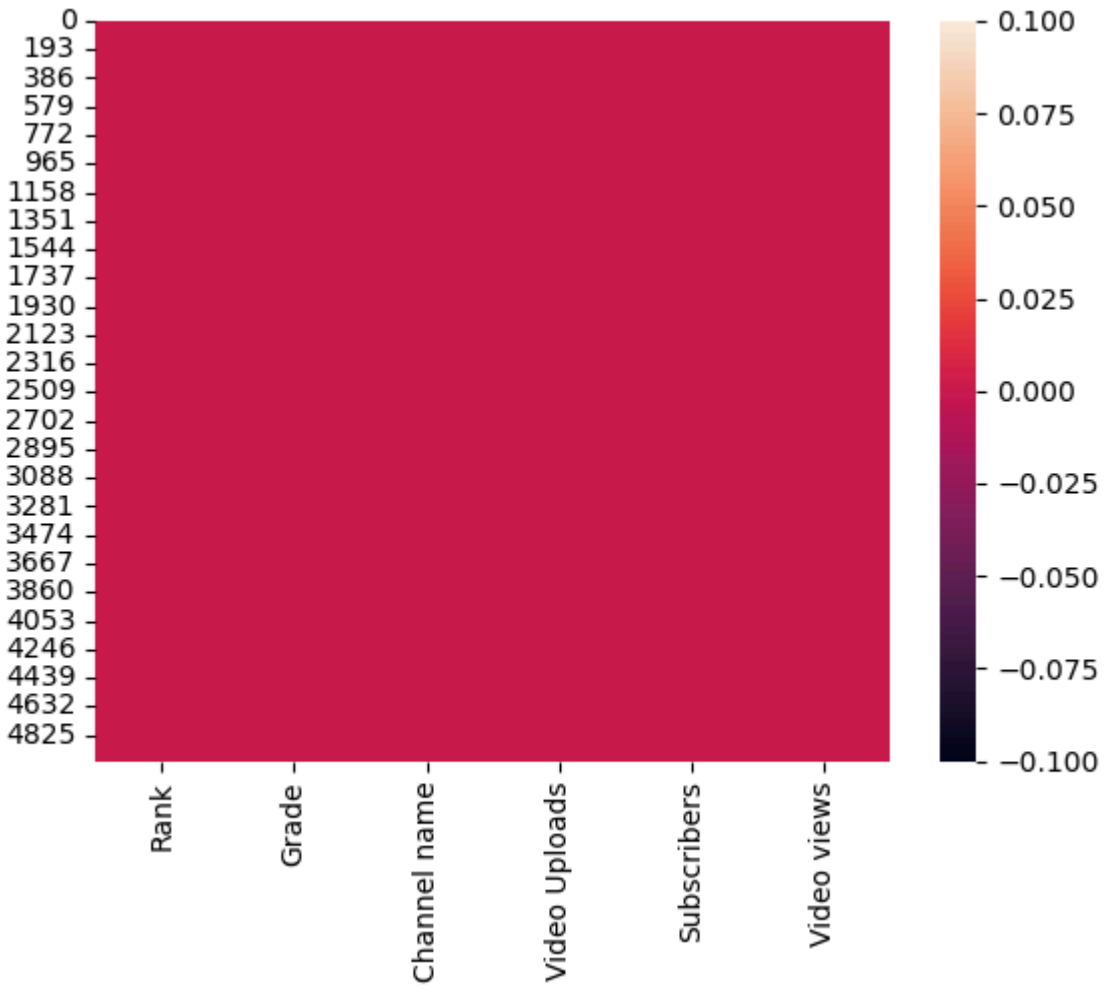
5000 rows × 6 columns

In [17]:
```
df.isnull().sum()
```

Out[17]:
```
Rank            0
Grade           0
Channel name    0
Video Uploads   0
Subscribers     0
Video views     0
dtype: int64
```

In [18]:
```
sns.heatmap(df.isnull())
```

Out[18]: <Axes: >



In [19]:
```
df.dropna(axis = 0, inplace = True)
```

In [20]:
```
per_missing = df.isnull().sum() * 100 / len(df)
```

In [21]:
```
per_missing
```

Out[21]:
```
Rank            0.00
Grade           0.00
Channel name    0.00
Video Uploads   0.00
Subscribers     0.00
Video views     0.00
dtype: float64
```

8. Data Cleaning [Rank Column]

In [22]:
```
df.head()
```

Out[22]:

	Rank	Grade	Channel name	Video Uploads	Subscribers	Video views
0	1st	A++	Zee TV	82757	18752951	20869786591
1	2nd	A++	T-Series	12661	61196302	47548839843
2	3rd	A++	Cocomelon - Nursery Rhymes	373	19238251	9793305082
3	4th	A++	SET India	27323	31180559	22675948293
4	5th	A++	WWE	36756	32852346	26273668433

At the end we have to give this data to machine learning algorithms. Most of the ML algorithms can only understand numerical values. Either int or float.

To clean this rank column, we are going to perform 3 steps.

1 - we will remove the string in "Rank column"

2 - we will remove this commas

3 - we will convert the data types of rank columns to int

In [24]:
```
df['Rank'] = df['Rank'].str[0:2]
```

In [25]:
```
df['Rank']
```

Out[25]:
```
0       1s
1       2n
2       3r
3       4t
4       5t
        ..
4995    4,
4996    4,
4997    4,
4998    4,
4999    5,
Name: Rank, Length: 5000, dtype: object
```

In [26]:
```
df.head()
```

Out[26]:

	Rank	Grade	Channel name	Video Uploads	Subscribers	Video views
0	1s	A++	Zee TV	82757	18752951	20869786591
1	2n	A++	T-Series	12661	61196302	47548839843
2	3r	A++	Cocomelon - Nursery Rhymes	373	19238251	9793305082
3	4t	A++	SET India	27323	31180559	22675948293
4	5t	A++	WWE	36756	32852346	26273668433

We have removed the alphabets from "Rank" column. Now we have to remove the commas from the column.

In [27]:
```
df['Rank'] = df['Rank'].str.replace(',' ,'')
```

In [28]:
```
df.tail()
```

Out[28]:

	Rank	Grade	Channel name	Video Uploads	Subscribers	Video views
4995	4	B+	Uras Benlioğlu	706	2072942	441202795
4996	4	B+	HI-TECH MUSIC LTD	797	1055091	377331722
4997	4	B+	Mastersaint	110	3265735	311758426
4998	4	B+	Bruce McIntosh	3475	32990	14563764
4999	5	B+	SehatAQUA	254	21172	73312511

9. Data Cleaning [Video Uploads & Subscribers]

In [33]:
```
df.dtypes
```

Out[33]:
```
Rank            object
Grade           object
Channel name    object
Video Uploads   object
Subscribers     object
Video views      int64
dtype: object
```

In [34]:
```
df["Video Uploads"]
```

Out[34]:
```
0       82757
1       12661
2         373
3       27323
4       36756
        ...
4995      706
4996      797
4997      110
4998     3475
4999      254
Name: Video Uploads, Length: 5000, dtype: object
```

In [35]:
```
# Replace '--' with NaN
df['Video Uploads'] = df['Video Uploads'].replace('--', np.nan)

# Convert the column to integers
df['Video Uploads'] = df['Video Uploads'].astype(float).astype('Int32')
```

In [36]:
```
# Replace NaN with 0
df['Video Uploads'] = df['Video Uploads'].fillna(0)

# Convert the column to integers
df['Video Uploads'] = df['Video Uploads'].astype(int)
```

In [37]:
```
df.dtypes
```

Out[37]:
```
Rank             object
Grade            object
Channel name     object
Video Uploads     int64
Subscribers      object
Video views       int64
dtype: object
```

In [38]:
```
# Replace '--' with NaN
df['Subscribers'] = df['Subscribers'].replace('--', np.nan)
```

In [39]:
```
df['Subscribers'] = df['Subscribers'].fillna(0)
```

In [40]:
```
# Replace non-numeric values with NaN
df['Subscribers'] = pd.to_numeric(df['Subscribers'], errors='coerce')

# Convert the column to integers
df['Subscribers'] = df['Subscribers'].astype('Int32')
```

In [41]:
```
df.dtypes
```

Out[41]:
```
Rank             object
Grade            object
Channel name     object
Video Uploads     int64
Subscribers       Int32
Video views       int64
dtype: object
```

10. Data Cleaning [Grade Column]

In [42]:
```
df.head()
```

Out[42]:

	Rank	Grade	Channel name	Video Uploads	Subscribers	Video views
0	1s	A++	Zee TV	82757	18752951	20869786591
1	2n	A++	T-Series	12661	61196302	47548839843
2	3r	A++	Cocomelon - Nursery Rhymes	373	19238251	9793305082
3	4t	A++	SET India	27323	31180559	22675948293
4	5t	A++	WWE	36756	32852346	26273668433

In [43]:
```
# Replace the Grade values using mapping methods

df['Grade'].unique()
```

Out[43]:
```
array(['A++ ', 'A+ ', 'A ', '\xa0 ', 'A- ', 'B+ '], dtype=object)
```

In [44]:
```
# now we are going to map this unique grade values to numeric values.

df['Grade'] = df['Grade'].map({'A++ ':5,'A+ ':4,'A ':3,'A- ':2,'B+ ':1})
```

In [45]:
```
df.head()
```

Out[45]:

	Rank	Grade	Channel name	Video Uploads	Subscribers	Video views
0	1s	5.00	Zee TV	82757	18752951	20869786591
1	2n	5.00	T-Series	12661	61196302	47548839843
2	3r	5.00	Cocomelon - Nursery Rhymes	373	19238251	9793305082
3	4t	5.00	SET India	27323	31180559	22675948293
4	5t	5.00	WWE	36756	32852346	26273668433

In [46]:
```
df.dtypes
```

Out[46]:
```
Rank             object
Grade           float64
Channel name     object
Video Uploads     int64
Subscribers       Int32
Video views       int64
dtype: object
```

11. Find Average Views For Each Channel

In [47]:
```
df.columns
```

Out[47]:
```
Index(['Rank', 'Grade', 'Channel name', 'Video Uploads', 'Subscribers',
       'Video views'],
      dtype='object')
```

In [48]:
```
df['avg_views'] = df['Video views'] / df['Video Uploads']
```

In [49]:
```
df['avg_views']
```

Out[49]:
```
0        252181.53
1       3755535.89
2      26255509.60
3        829921.62
4        714813.05
           ...
4995     624933.14
4996     473440.05
4997    2834167.51
4998       4191.01
4999     288631.93
Name: avg_views, Length: 5000, dtype: float64
```

So this mean we got 1000 views from 50 uploads

In [50]:
```
df.head()
```

Out[50]:

	Rank	Grade	Channel name	Video Uploads	Subscribers	Video views	avg_views
0	1s	5.00	Zee TV	82757	18752951	20869786591	252181.53
1	2n	5.00	T-Series	12661	61196302	47548839843	3755535.89
2	3r	5.00	Cocomelon - Nursery Rhymes	373	19238251	9793305082	26255509.60
3	4t	5.00	SET India	27323	31180559	22675948293	829921.62
4	5t	5.00	WWE	36756	32852346	26273668433	714813.05

12. Find Out Top Five Channels With Maximum Number of Video Uploads

In [51]:
```
df.columns
```

Out[51]:
```
Index(['Rank', 'Grade', 'Channel name', 'Video Uploads', 'Subscribers',
       'Video views', 'avg_views'],
      dtype='object')
```

In [52]:
```
df.sort_values(by='Video Uploads', ascending = False).head()
```

Out[52]:

	Rank	Grade	Channel name	Video Uploads	Subscribers	Video views	avg_views
3453	3	1.00	AP Archive	422326	746325	548619569	1299.04
1149	1	2.00	YTN NEWS	355996	820108	1640347646	4607.77
2223	2	1.00	SBS Drama	335521	1418619	1565758044	4666.65
323	32	3.00	GMA News	269065	2599175	2786949164	10357.90
2956	2	1.00	MLB	267649	1434206	1329206392	4966.23

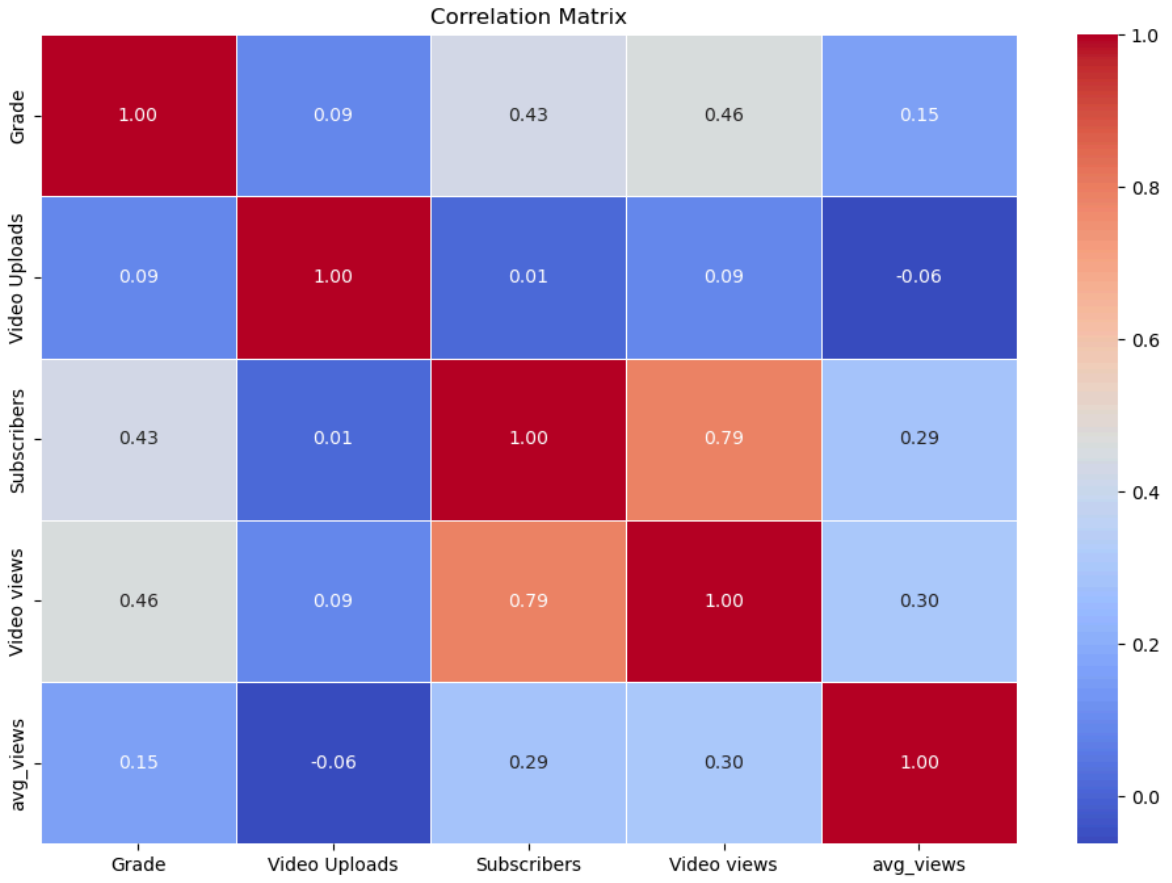
13. Find Correlation Matrix

In [53]:
```
df.dtypes
```

Out[53]:
```
Rank              object
Grade            float64
Channel name      object
Video Uploads      int64
Subscribers        Int32
Video views        int64
avg_views        float64
dtype: object
```

In [54]:
```
# Select only numeric columns
numeric_columns = df.select_dtypes(include=[np.number])

# Calculate the correlation matrix
correlation_matrix = numeric_columns.corr()
```

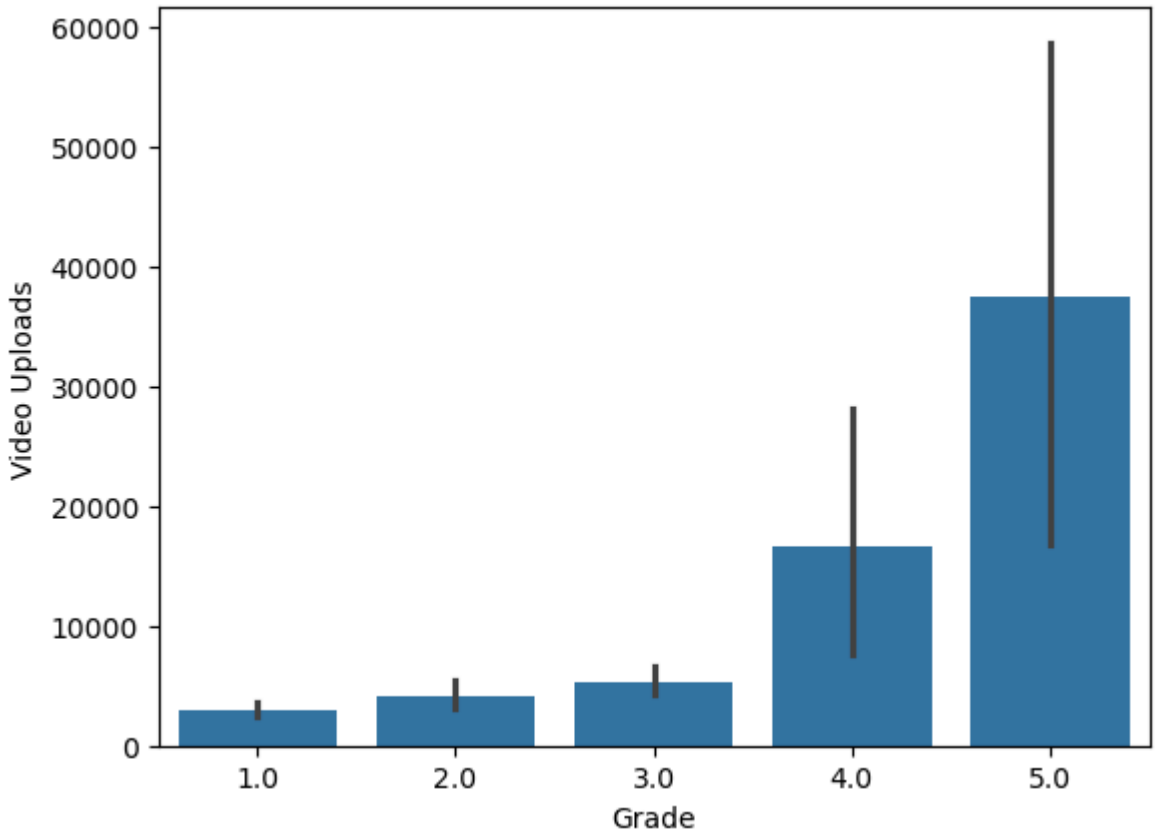
In [55]:
```
# Select only numeric columns
numeric_columns = df.select_dtypes(include=[np.number])

# Calculate the correlation matrix
correlation_matrix = numeric_columns.corr()

# Create a heatmap for the correlation matrix
plt.figure(figsize=(12, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f", linewidt
plt.title('Correlation Matrix')
plt.show()
```



14. Which Grade Has A Maximum Number of Video Uploads?

In [56]:
```
sns.barplot(x='Grade',y='Video Uploads',data=df)
```

Out[56]:
```
<Axes: xlabel='Grade', ylabel='Video Uploads'>
```

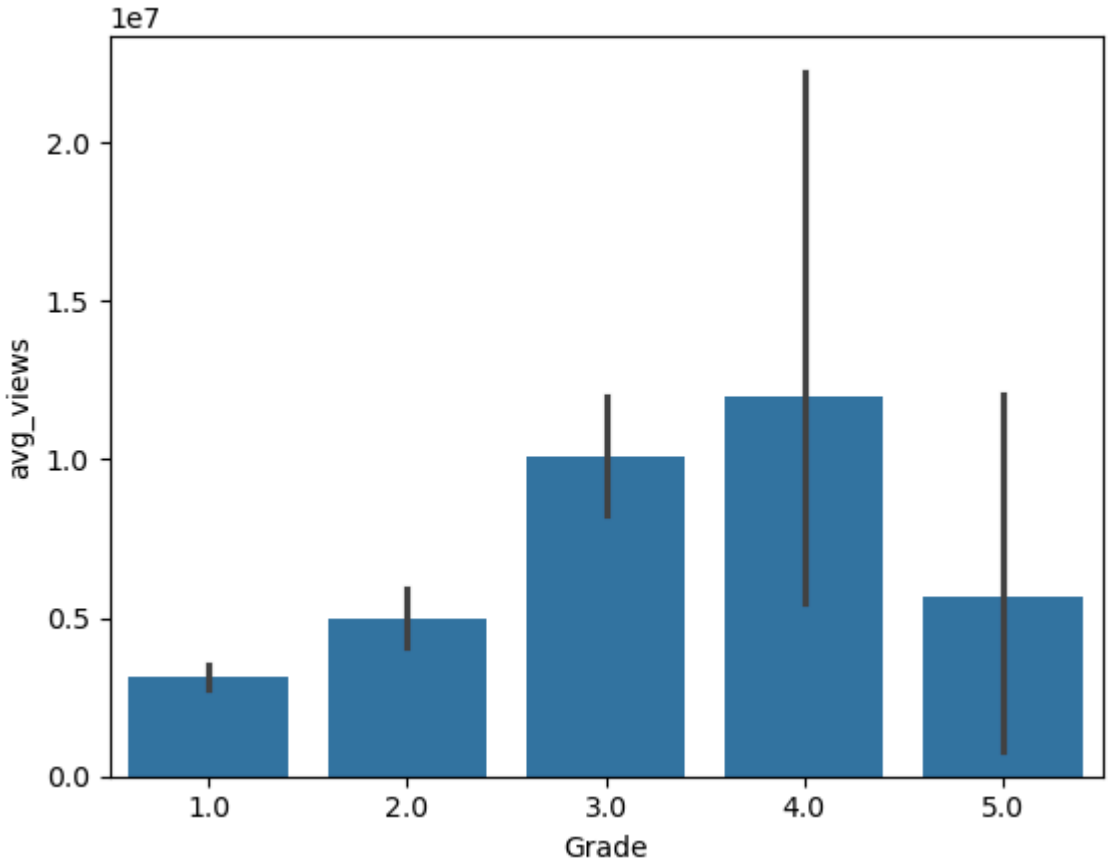


From the above graph it is clear that when the grade is high, video upload is also high. Video Uploads are high in 'A++' video channels.

15.Which Grade Has The Highest Average Views?

In [58]:
```
sns.barplot(x='Grade',y='avg_views',data=df)
```

Out[58]:
```
<Axes: xlabel='Grade', ylabel='avg_views'>
```

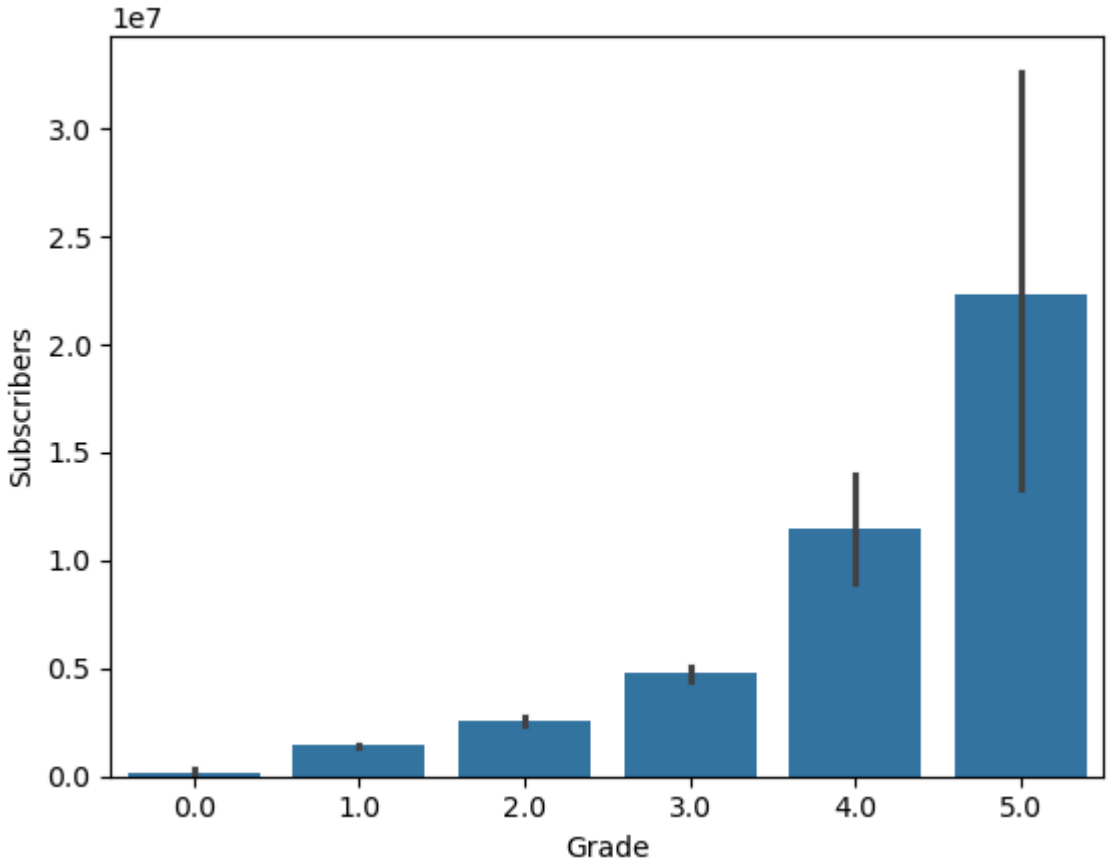


Channels with 'A+' grade has higher number of average views compared to others.

16. Which Grade Has The Highest Number of Subscribers?

In [59]:
```
# since we got " boolean value of NA is ambiguous" while executing barplot, I'm

newdata_filled = df.fillna(0)  # You can replace 0 with any value you want.
sns.barplot(x='Grade', y='Subscribers', data=newdata_filled)
```

Out[59]:
```
<Axes: xlabel='Grade', ylabel='Subscribers'>
```



From the above graph it is clear that channel with the 'A++' Grade has the highest number of subscribers

17. Which Grade Has The Highest Video Views?

In [60]:
```
df.groupby('Grade')['Video views'].sum()
```

Out[60]:
```
Grade
1.00    1556398001373
2.00    1066136831368
3.00    2273948590311
4.00     248177945463
5.00     211990911928
Name: Video views, dtype: int64
```

reference -

https://www.youtube.com/watch?v=nrc-n98pF2w&list=PL_1pt6K-CLoDMEbYy2PcZuITWEjqMfyoA&index=13

In [ ]:
```

```