In [7]:
```
pip install pandas
```

```
Requirement already satisfied: pandas in c:\users\rahul\anaconda3\lib\site
-packages (1.4.2)
Requirement already satisfied: pytz>=2020.1 in c:\users\rahul\anaconda3\li
b\site-packages (from pandas) (2021.3)
Requirement already satisfied: python-dateutil>=2.8.1 in c:\users\rahul\an
aconda3\lib\site-packages (from pandas) (2.8.2)
Requirement already satisfied: numpy>=1.18.5 in c:\users\rahul\anaconda3\l
ib\site-packages (from pandas) (1.21.5)
Requirement already satisfied: six>=1.5 in c:\users\rahul\anaconda3\lib\si
te-packages (from python-dateutil>=2.8.1->pandas) (1.16.0)
Note: you may need to restart the kernel to use updated packages.
```

In [2]:
```
import pandas as pd
```

In [10]:
```
import pandas as pd

# Read the CSV file into a DataFrame, specifying the encoding
file_path = r'C:\Users\Rahul\OneDrive\Desktop\DsResearch\Media and Technolog
df = pd.read_csv(file_path, encoding='latin1')  # or encoding='utf-16'

# Explore the data
print(df.head())  # Display the first few rows
print(df.columns)  # Check column names
print(df.dtypes)  # Check data types
print(df.describe())  # Summary statistics
print(df.isnull().sum())  # Check for missing values
```

```
   rank                    Youtuber  subscribers   video views  \
0     1                    T-Series  245000000.0  2.280000e+11
1     2              YouTube Movies  170000000.0  0.000000e+00
2     3                     MrBeast  166000000.0  2.836884e+10
3     4  Cocomelon - Nursery Rhymes  162000000.0  1.640000e+11
4     5                   SET India  159000000.0  1.480000e+11

           category                       Title  uploads Country of origin
\
0             Music                    T-Series    20082             India
1  Film & Animation               youtubemovies        1     United States
2     Entertainment                     MrBeast      741     United States
3         Education  Cocomelon - Nursery Rhymes      966     United States
4             Shows                   SET India   116536             India

         Country Abbreviation  ... subscribers_for_last_30_days  created_y
ear  \
0          india           IN  ...                    2000000.0        200
6.0
1  United States           US  ...                          NaN        200
6.0
2  United States           US  ...                    8000000.0        201
2.0
3  United States           US  ...                    1000000.0        200
6.0
4          India           IN  ...                    1000000.0        200
6.0

   created_month  created_date  Gross tertiary education enrollment (%)  \
0            Mar          13.0                                     28.1
1            NaN           5.0                                     88.2
2            Feb          20.0                                     88.2
3            Sep           1.0                                     88.2
4            Sep          20.0                                     28.1

     Population  Unemployment rate  Urban_population   Latitude  Longitude
0  1.366418e+09               5.36       471031528.0  20.593684  78.962880
1  3.282395e+08              14.70       270663028.0  37.090240 -95.712891
2  3.282395e+08              14.70       270663028.0  37.090240 -95.712891
3  3.282395e+08              14.70       270663028.0  37.090240 -95.712891
4  1.366418e+09               5.36       471031528.0  20.593684  78.962880

[5 rows x 29 columns]
Index(['rank', 'Youtuber', 'subscribers', 'video views', 'category', 'Titl
e',
       'uploads', 'Country of origin', 'Country', 'Abbreviation',
       'channel_type', 'video_views_rank', 'country_rank', 'channel_type_r
ank',
       'video_views_for_the_last_30_days', 'lowest_monthly_earnings',
       'highest_monthly_earnings', 'lowest_yearly_earnings',
       'highest_yearly_earnings', 'subscribers_for_last_30_days',
       'created_year', 'created_month', 'created_date',
       'Gross tertiary education enrollment (%)', 'Population',
       'Unemployment rate', 'Urban_population', 'Latitude', 'Longitude'],
      dtype='object')
rank                                         int64
Youtuber                                    object
subscribers                                float64
video views                                float64
category                                    object
Title                                       object
```

```
uploads                                      int64
Country of origin                           object
Country                                     object
Abbreviation                                object
channel_type                                object
video_views_rank                           float64
country_rank                               float64
channel_type_rank                          float64
video_views_for_the_last_30_days           float64
lowest_monthly_earnings                    float64
highest_monthly_earnings                   float64
lowest_yearly_earnings                     float64
highest_yearly_earnings                    float64
subscribers_for_last_30_days               float64
created_year                               float64
created_month                               object
created_date                               float64
Gross tertiary education enrollment (%)    float64
Population                                 float64
Unemployment rate                          float64
Urban_population                           float64
Latitude                                   float64
Longitude                                  float64
dtype: object
              rank   subscribers   video views        uploads  \
count  1006.000000  1.003000e+03  1.006000e+03    1006.000000   
mean    497.472167  2.319501e+07  1.112411e+10    9168.335984   
std     288.738758  1.783047e+07  1.424148e+10   34028.189437   
min       1.000000  1.230000e+07  0.000000e+00       0.000000   
25%     247.250000  1.450000e+07  4.281427e+09     194.000000   
50%     498.500000  1.770000e+07  7.751292e+09     726.500000   
75%     748.750000  2.475000e+07  1.357357e+10    2606.500000   
max     995.000000  2.450000e+08  2.280000e+11  301308.000000   

       video_views_rank  country_rank  channel_type_rank  \
count      1.005000e+03    887.000000         971.000000   
mean       5.607670e+05    384.289741         742.311020   
std        1.368886e+06   1227.359768        1938.126477   
min        1.000000e+00      1.000000           1.000000   
25%        3.220000e+02     11.000000          26.000000   
50%        9.190000e+02     50.000000          65.000000   
75%        3.645000e+03    123.000000         139.000000   
max        4.057944e+06   7741.000000        7741.000000   

       video_views_for_the_last_30_days  lowest_monthly_earnings  \
count                      9.490000e+02              1006.000000   
mean                       1.760978e+08             37034.348489   
std                        4.152933e+08             71869.653679   
min                        1.000000e+00                 0.000000   
25%                        1.974000e+07              2700.000000   
50%                        6.408500e+07             13250.000000   
75%                        1.692420e+08             38125.000000   
max                        6.589000e+09            850900.000000   

       highest_monthly_earnings  ...  highest_yearly_earnings  \
count              1.006000e+03  ...             1.006000e+03   
mean               5.922453e+05  ...             7.110327e+06   
std                1.148967e+06  ...             1.379921e+07   
min                0.000000e+00  ...             0.000000e+00   
25%                4.340000e+04  ...             5.207500e+05   
50%                2.117000e+05  ...             2.550000e+06   
```

```
75%                6.101750e+05  ...            7.300000e+06
max                1.360000e+07  ...            1.634000e+08

       subscribers_for_last_30_days  created_year  created_date  \
count                  6.660000e+02   1001.000000   1001.000000
mean                   3.495419e+05   2012.613387     15.712288
std                    6.131554e+05      4.514131      8.765109
min                    1.000000e+00   1970.000000      1.000000
25%                    1.000000e+05   2009.000000      8.000000
50%                    2.000000e+05   2013.000000     16.000000
75%                    4.000000e+05   2016.000000     23.000000
max                    8.000000e+06   2022.000000     31.000000

       Gross tertiary education enrollment (%)    Population  \
count                               880.000000  8.800000e+02
mean                                 63.597273  4.304586e+08
std                                  26.095537  4.735536e+08
min                                   7.600000  2.025060e+05
25%                                  36.300000  8.313280e+07
50%                                  68.000000  3.282395e+08
75%                                  88.200000  3.282395e+08
max                                 113.100000  1.397715e+09

       Unemployment rate  Urban_population    Latitude   Longitude
count         880.000000      8.800000e+02  880.000000  880.000000
mean            9.258966      2.239747e+08   26.639994  -13.811287
std             4.889912      1.550381e+08   20.516025   84.728810
min             0.750000      3.558800e+04  -38.416097 -172.104629
25%             5.270000      5.590832e+07   20.593684  -95.712891
50%             8.880000      2.706630e+08   37.090240  -51.925280
75%            14.700000      2.706630e+08   37.090240   78.962880
max            14.720000      8.429340e+08   61.924110  138.252924

[8 rows x 21 columns]
rank                                         0
Youtuber                                     0
subscribers                                  3
video views                                  0
category                                    55
Title                                        0
uploads                                      0
Country of origin                          125
Country                                    125
Abbreviation                               125
channel_type                                32
video_views_rank                             1
country_rank                               119
channel_type_rank                           35
video_views_for_the_last_30_days            57
lowest_monthly_earnings                      0
highest_monthly_earnings                     0
lowest_yearly_earnings                       0
highest_yearly_earnings                      0
subscribers_for_last_30_days               340
created_year                                 5
created_month                               12
created_date                                 5
Gross tertiary education enrollment (%)    126
Population                                 126
Unemployment rate                          126
Urban_population                           126
```

```
Latitude                                    126
Longitude                                   126
dtype: int64
```

In [13]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Step 1: Data Loading
df = pd.read_csv(r'C:\Users\Rahul\OneDrive\Desktop\DsResearch\Media and Tec

# Step 2: Data Preprocessing

# Check for missing values
print("Missing values:")
print(df.isnull().sum())

# Handle missing values if necessary
# For example, if there are missing values in the 'category' column, you ca
df['category'].fillna('Unknown', inplace=True)

# Convert data types if necessary
# For example, convert 'created_date' column to datetime format
df['created_date'] = pd.to_datetime(df['created_date'])

# Drop irrelevant columns if needed
# For example, if 'Abbreviation' is not relevant for analysis, you can drop
df.drop(columns=['Abbreviation'], inplace=True)
```

```
Missing values:
rank                                         0
Youtuber                                     0
subscribers                                  3
video views                                  0
category                                    55
Title                                        0
uploads                                      0
Country of origin                          125
Country                                    125
Abbreviation                               125
channel_type                                32
video_views_rank                             1
country_rank                               119
channel_type_rank                           35
video_views_for_the_last_30_days            57
lowest_monthly_earnings                      0
highest_monthly_earnings                     0
lowest_yearly_earnings                       0
highest_yearly_earnings                      0
subscribers_for_last_30_days               340
created_year                                 5
created_month                               12
created_date                                 5
Gross tertiary education enrollment (%)    126
Population                                 126
Unemployment rate                          126
Urban_population                           126
Latitude                                   126
Longitude                                  126
dtype: int64
```

In [17]:
```
# Step 3: Analysis and Visualization

#1. What are the top 10 YouTube channels based on the number of subscribers
top_10_subscribers = df.nlargest(10, 'subscribers')[['Youtuber', 'subscribe
print("Top 10 YouTube channels based on subscribers:")
print(top_10_subscribers)
```

```
Top 10 YouTube channels based on subscribers:
                     Youtuber  subscribers
0                    T-Series  245000000.0
1              YouTube Movies  170000000.0
2                     MrBeast  166000000.0
3   Cocomelon - Nursery Rhymes  162000000.0
4                   SET India  159000000.0
5                       Music  119000000.0
6          ýýý Kids Diana Show  112000000.0
7                    PewDiePie  111000000.0
8                 Like Nastya  106000000.0
9               Vlad and Niki   98900000.0
```

In [19]:
```
# Question 2: Category with the highest average number of subscribers

avg_subscribers_by_category = df.groupby('category')['subscribers'].mean().
print("Category with the highest average number of subscribers:", avg_subsc
```

```
Category with the highest average number of subscribers: Shows
```

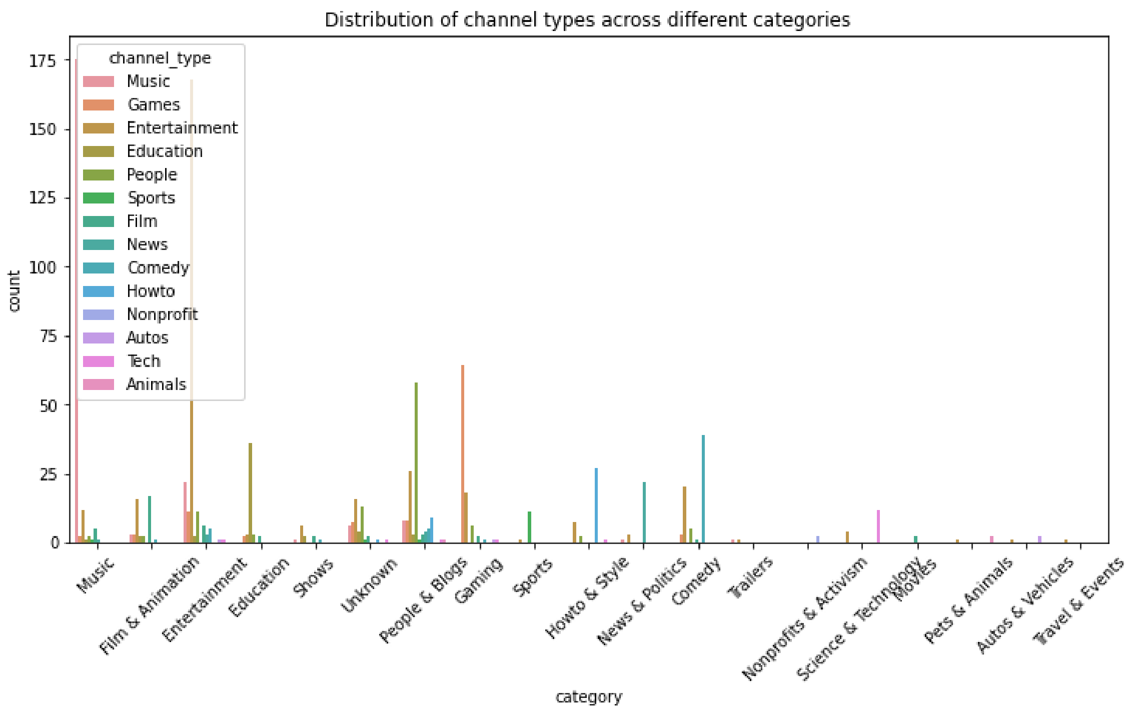
In [20]:
```
# Question 3: Average number of videos uploaded by YouTube channels in each

avg_videos_by_category = df.groupby('category')['uploads'].mean()
print("Average number of videos uploaded by YouTube channels in each catego
print(avg_videos_by_category)
```

```
Average number of videos uploaded by YouTube channels in each category:
category
Autos & Vehicles           1550.666667
Comedy                     1202.557143
Education                  3087.086957
Entertainment             12052.445378
Film & Animation           2861.844444
Gaming                     4285.273684
Howto & Style              1695.500000
Movies                     3553.000000
Music                      2325.945813
News & Politics          112484.384615
Nonprofits & Activism    102912.000000
People & Blogs             9256.793893
Pets & Animals             3562.800000
Science & Technology       2114.058824
Shows                     27443.692308
Sports                    19129.833333
Trailers                   6839.000000
Travel & Events             766.000000
Unknown                     790.345455
Name: uploads, dtype: float64
```

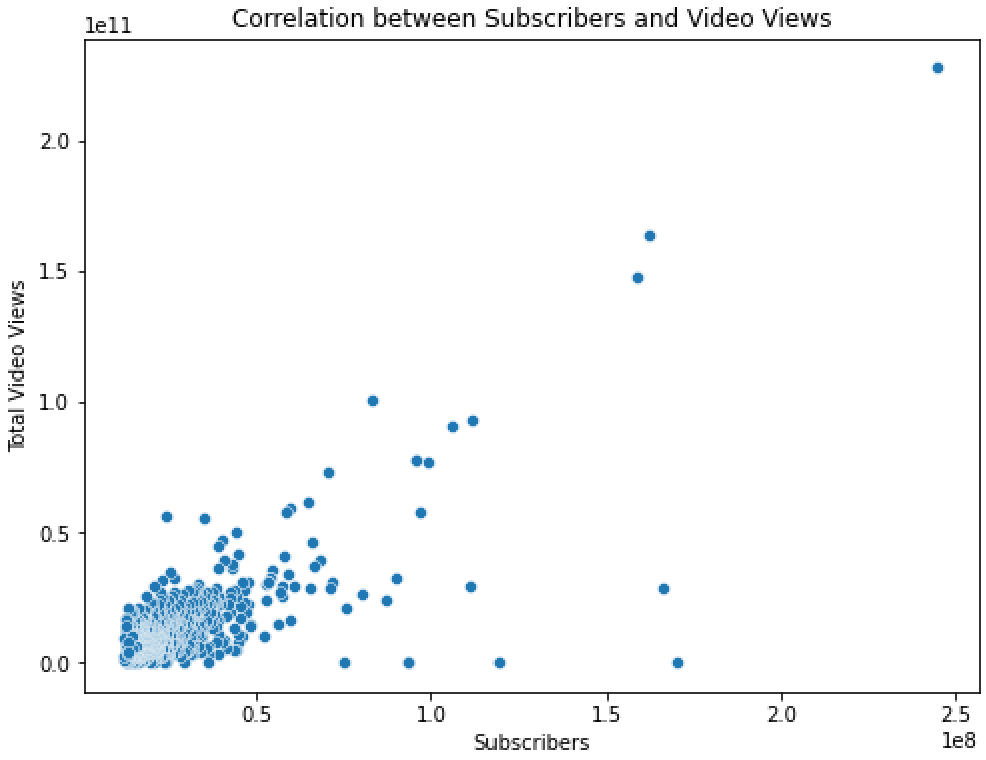
In [21]:
```
# Question 4: Top 5 countries with the highest number of YouTube channels

top_5_countries = df['Country'].value_counts().head(5)
print("Top 5 countries with the highest number of YouTube channels:")
print(top_5_countries)
```

```
Top 5 countries with the highest number of YouTube channels:
United States     315
India             169
Brazil             62
United Kingdom     44
Mexico             33
Name: Country, dtype: int64
```

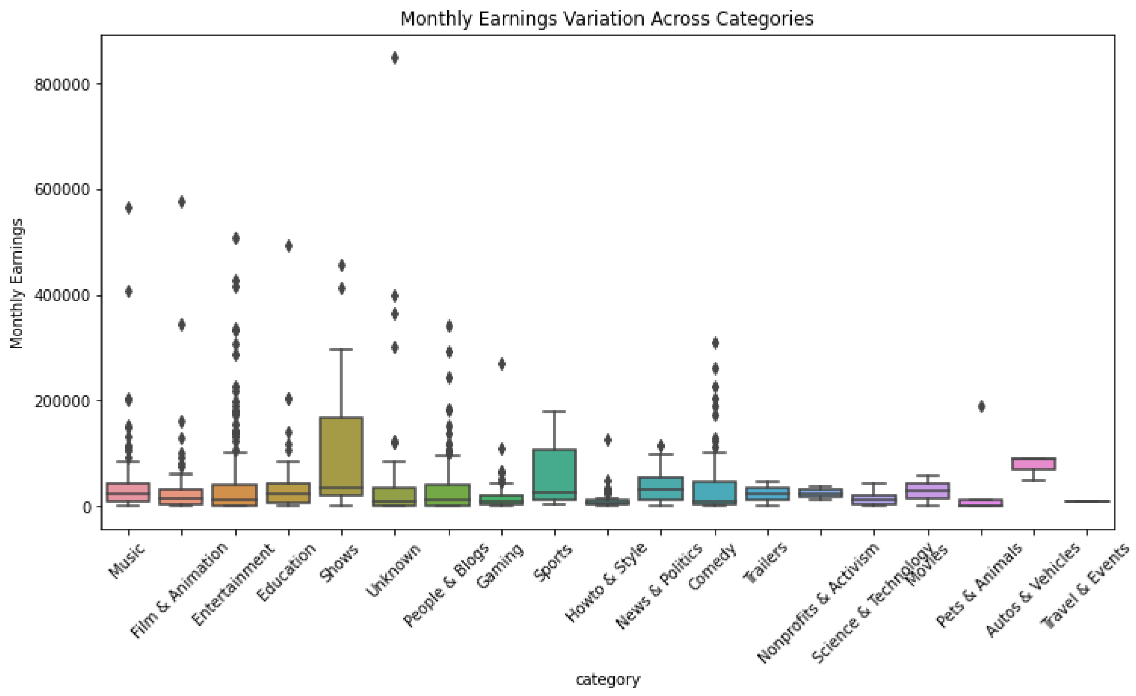
In [22]:
```
# Question 5: Distribution of channel types across different categories

plt.figure(figsize=(12, 6))
sns.countplot(data=df, x='category', hue='channel_type')
plt.title("Distribution of channel types across different categories")
plt.xticks(rotation=45)
plt.show()
```



In [23]:
```
# Question 6: Is there a correlation between the number of subscribers and

plt.figure(figsize=(8, 6))
sns.scatterplot(data=df, x='subscribers', y='video views')
plt.title("Correlation between Subscribers and Video Views")
plt.xlabel("Subscribers")
plt.ylabel("Total Video Views")
plt.show()
```

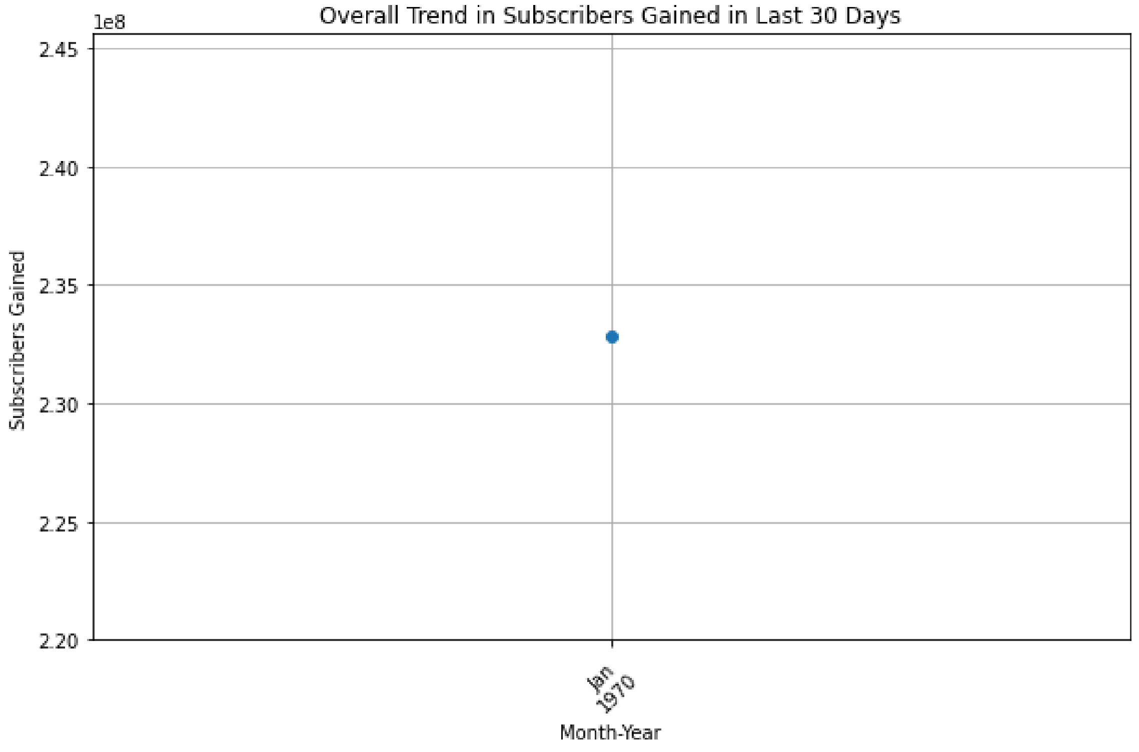


```
In [24]: # Question 7: How do the monthly earnings vary throughout different categor

         plt.figure(figsize=(12, 6))
         sns.boxplot(data=df, x='category', y='lowest_monthly_earnings')
         plt.title("Monthly Earnings Variation Across Categories")
         plt.xticks(rotation=45)
         plt.ylabel("Monthly Earnings")
         plt.show()
```



In [28]:
```
# Question 8: What is the overall trend in subscribers gained in the last 3
plt.figure(figsize=(10, 6))
df['created_date'] = pd.to_datetime(df['created_date'])
df['month_year'] = df['created_date'].dt.to_period('M')
monthly_subscribers = df.groupby('month_year')['subscribers_for_last_30_day
monthly_subscribers.plot(kind='line', marker='o')
plt.title("Overall Trend in Subscribers Gained in Last 30 Days")
plt.xlabel("Month-Year")
plt.ylabel("Subscribers Gained")
plt.xticks(rotation=45)
plt.grid(True)

# Explicitly set the limits for the x-axis
min_date = monthly_subscribers.index.min().to_timestamp()
max_date = monthly_subscribers.index.max().to_timestamp()
plt.xlim(min_date, max_date)

plt.show()
```

```
C:\Users\Rahul\anaconda3\lib\site-packages\pandas\plotting\_matplotlib\cor
e.py:1244: UserWarning: Attempting to set identical left == right == 0.0 r
esults in singular transformations; automatically expanding.
  ax.set_xlim(left, right)
C:\Users\Rahul\AppData\Local\Temp\ipykernel_32988\1477017500.py:16: UserWa
rning: Attempting to set identical left == right == 0 results in singular
transformations; automatically expanding.
  plt.xlim(min_date, max_date)
```



In [29]:
```
# Print the monthly_subscribers variable for debugging
print(monthly_subscribers)
```

```
month_year
1970-01    232794874.0
Freq: M, Name: subscribers_for_last_30_days, dtype: float64
```

In [30]:
```
# Check the data type of the 'created_date' column
print(df['created_date'].dtype)

# Inspect the first few rows of the DataFrame
print(df.head())
```

```
datetime64[ns]
   rank                    Youtuber  subscribers   video views  \
0     1                    T-Series  245000000.0  2.280000e+11
1     2              YouTube Movies  170000000.0  0.000000e+00
2     3                     MrBeast  166000000.0  2.836884e+10
3     4  Cocomelon - Nursery Rhymes  162000000.0  1.640000e+11
4     5                   SET India  159000000.0  1.480000e+11

           category                       Title  uploads Country of origin
\
0             Music                    T-Series    20082             India
1  Film & Animation               youtubemovies        1     United States
2     Entertainment                     MrBeast      741     United States
3         Education  Cocomelon - Nursery Rhymes      966     United States
4             Shows                   SET India   116536             India

         Country   channel_type  ...  created_year  created_month  \
0          india          Music  ...        2006.0            Mar
1  United States          Games  ...        2006.0            NaN
2  United States  Entertainment  ...        2012.0            Feb
3  United States      Education  ...        2006.0            Sep
4          India  Entertainment  ...        2006.0            Sep

                   created_date  Gross tertiary education enrollment (%)
\
0 1970-01-01 00:00:00.000000013                                     28.1
1 1970-01-01 00:00:00.000000005                                     88.2
2 1970-01-01 00:00:00.000000020                                     88.2
3 1970-01-01 00:00:00.000000001                                     88.2
4 1970-01-01 00:00:00.000000020                                     28.1

     Population  Unemployment rate  Urban_population   Latitude  Longitude
\
0  1.366418e+09               5.36       471031528.0  20.593684  78.962880
1  3.282395e+08              14.70       270663028.0  37.090240 -95.712891
2  3.282395e+08              14.70       270663028.0  37.090240 -95.712891
3  3.282395e+08              14.70       270663028.0  37.090240 -95.712891
4  1.366418e+09               5.36       471031528.0  20.593684  78.962880

   month_year
0     1970-01
1     1970-01
2     1970-01
3     1970-01
4     1970-01

[5 rows x 29 columns]
```

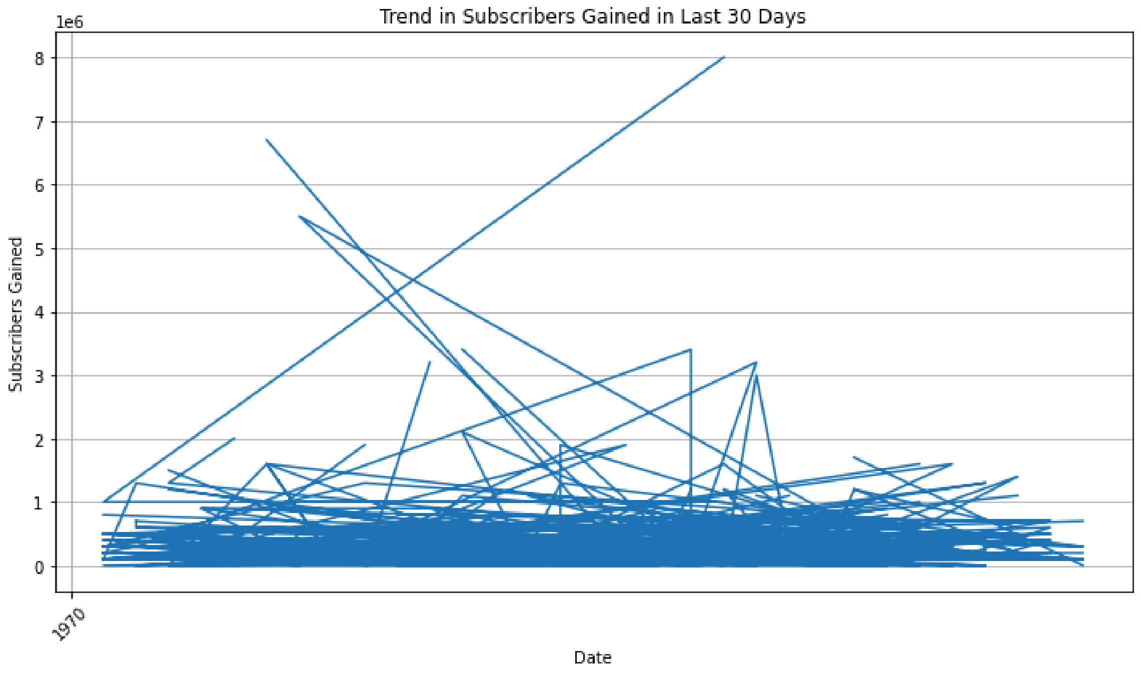
In [31]:
```
# Convert the 'created_date' column to datetime format
df['created_date'] = pd.to_datetime(df['created_date'])

# Extract the month and year from the 'created_date' column
df['month_year'] = df['created_date'].dt.to_period('M')

# Check the unique values in the 'month_year' column
print(df['month_year'].unique())
```

```
<PeriodArray>
['1970-01', 'NaT']
Length: 2, dtype: period[M]
```

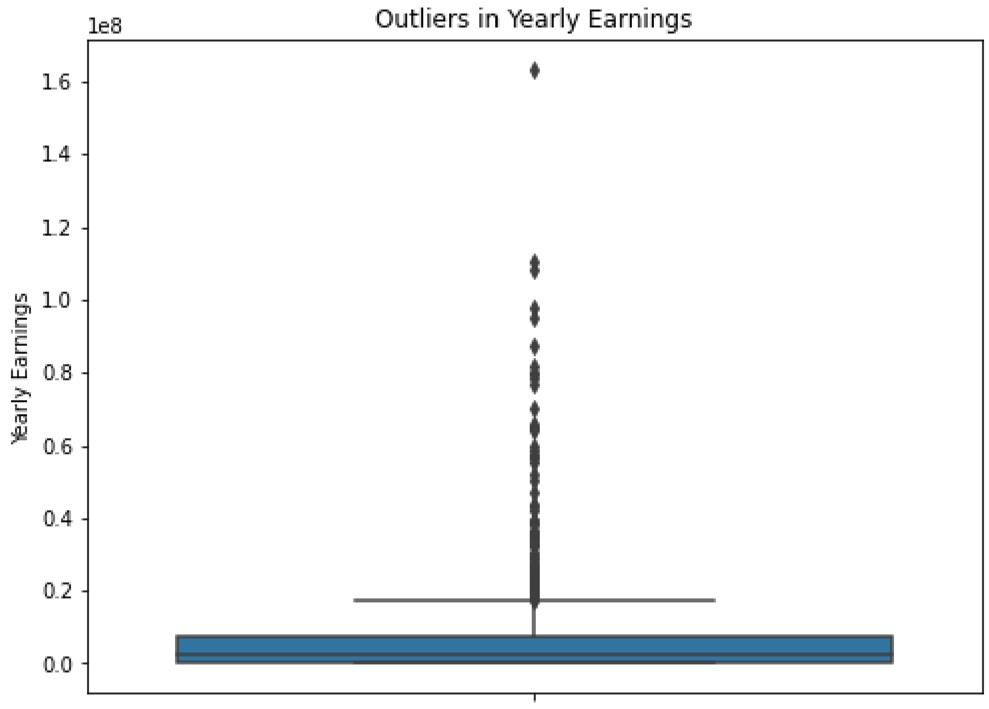
In [32]:
```
# Check for missing or invalid date values in the 'created_date' column
missing_dates = df[df['created_date'].isnull()]
invalid_dates = df[~df['created_date'].notnull()]

# Print the missing and invalid date values
print("Missing dates:")
print(missing_dates)
print("\nInvalid dates:")
print(invalid_dates)
```

```
Missing dates:
     rank               Youtuber  subscribers   video views         cate
gory  \
236   237            Chris Brown   25200000.0  1.552057e+10            M
usic
468   469  Good Mythical Morning   18300000.0  8.798045e+09   Entertain
ment
508   509     The Game Theorists   17600000.0  3.752347e+09           Ga
ming
735   736           LEGENDA FUNK   14500000.0  2.440718e+09          Unk
nown
762   763           Harry Styles   14400000.0  5.689224e+09  People & B
logs

                    Title  uploads Country of origin        Country  \
236            ChrisBrown        0               NaN            NaN
468  Goodmythicalmorning        0               NaN            NaN
508     TheGameTheorists        0         Australia      Australia
735          LegendaFUNK        0            Brazil         Brazil
```

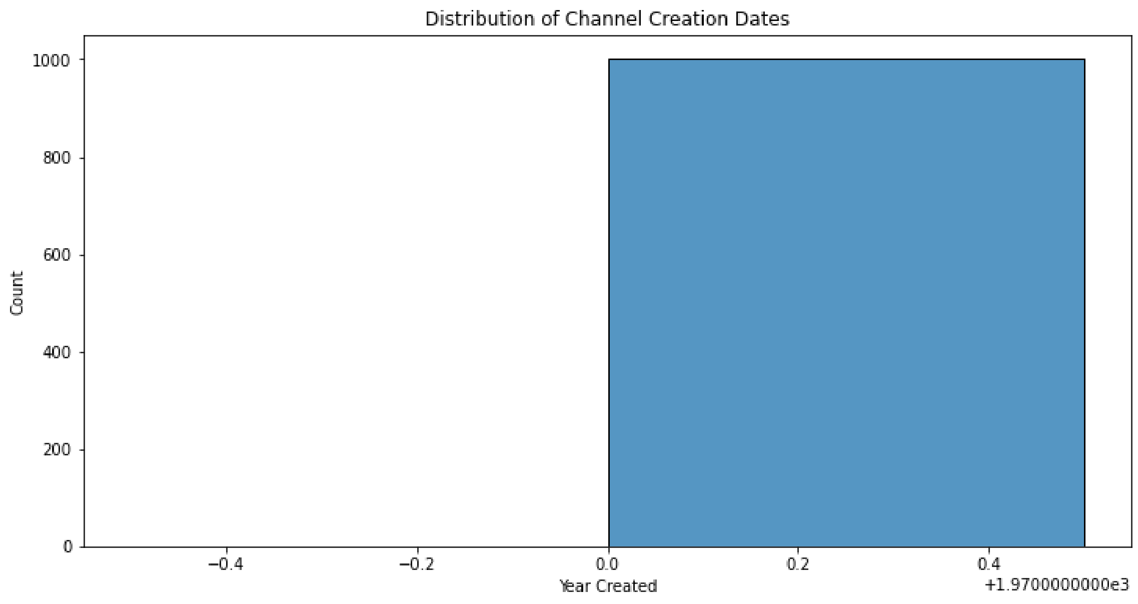
In [33]:
```
import matplotlib.pyplot as plt

# Filter out rows with missing or invalid dates
valid_dates_df = df[df['created_date'].notnull()]

# Plotting the trend in subscribers gained in the last 30 days across all c
plt.figure(figsize=(10, 6))
plt.plot(valid_dates_df['created_date'], valid_dates_df['subscribers_for_la
plt.title('Trend in Subscribers Gained in Last 30 Days')
plt.xlabel('Date')
plt.ylabel('Subscribers Gained')
plt.xticks(rotation=45)
plt.grid(True)
plt.tight_layout()
plt.show()
```



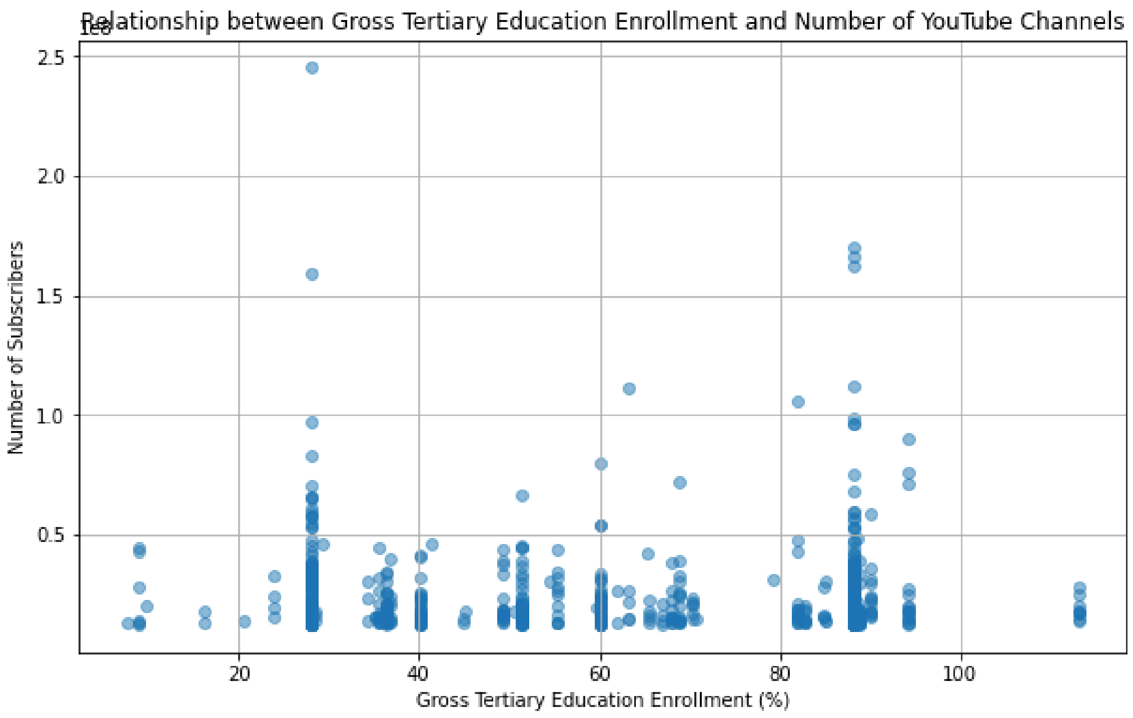
In [34]:
```python
# Question 9: Are there any outliers in terms of yearly earnings from YouTu
plt.figure(figsize=(8, 6))
sns.boxplot(data=df, y='highest_yearly_earnings')
plt.title("Outliers in Yearly Earnings")
plt.ylabel("Yearly Earnings")
plt.show()
```



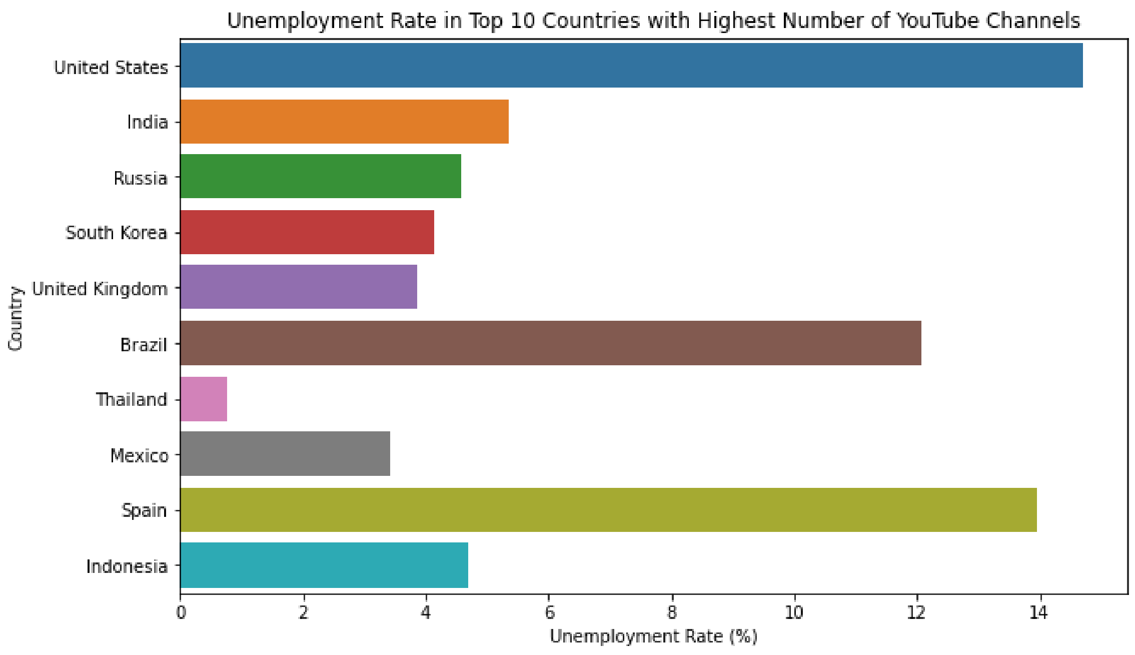
In [35]:
```python
# Question 10: What is the distribution of channel creation dates? Is there
plt.figure(figsize=(12, 6))
df['created_date'] = pd.to_datetime(df['created_date'])
df['year_created'] = df['created_date'].dt.year
sns.histplot(data=df, x='year_created', bins=len(df['year_created'].unique(
plt.title("Distribution of Channel Creation Dates")
plt.xlabel("Year Created")
plt.ylabel("Count")
plt.show()
```



In [36]:
```
#11. Is there a relationship between gross tertiary education enrollment an
plt.figure(figsize=(10, 6))
plt.scatter(df['Gross tertiary education enrollment (%)'], df['subscribers'
plt.title('Relationship between Gross Tertiary Education Enrollment and Num
plt.xlabel('Gross Tertiary Education Enrollment (%)')
plt.ylabel('Number of Subscribers')
plt.grid(True)
plt.show()
```



In [37]:
```
#12. How does the unemployment rate vary among the top 10 countries with th
# Filter the top 10 countries with the highest number of YouTube channels
top_10_countries = df['Country'].value_counts().head(10).index

# Create a bar plot for the unemployment rate in the top 10 countries
plt.figure(figsize=(10, 6))
sns.barplot(x='Unemployment rate', y='Country', data=df[df['Country'].isin(
plt.title('Unemployment Rate in Top 10 Countries with Highest Number of You
plt.xlabel('Unemployment Rate (%)')
plt.ylabel('Country')
plt.show()
```



In [39]:
```
#13. What is the average urban population percentage in countries with YouT
# Calculate the average urban population percentage
avg_urban_population = df['Urban_population'].mean()
print("Average Urban Population Percentage:", avg_urban_population)
```

```
Average Urban Population Percentage: 223974718.82045454
```

In [ ]:

In [ ]: