Video Streaming Services EDA

February 29, 2024

1 Video Streaming Service Platforms - EDA

1.0.1 SUMMARY:

We were tasked with coming up with 4 interesting questions to answer using the data sets provided. Our group chose to examing the 4 largest video streaming platforms: Netflix, Hulu, Disney+, and Amazon Prime Video.

The data provided was simply a list of each platform's title catalog, including information on rating, duration, director, and more. That being said, many of these data fields needed significant cleaning, as you will see below, thus limiting our analysis to type (TV Show/Movie), title, date_added, release_year, rating, duration, and main genre. The data sets contain mostly qualitative data, as performance metrics such as total plays, total positive ratings, new user registration data were not included. This fact made us take significant time in determining who our stakeholder for this analysis could be.

Overall, we settled on the following stakeholder profile: Producer who is trying figure out which streaming platform to sell their new R-rated movie to, as well as, what content-type and genre should future content focus on.

We attempt to answer the following questions for our defined stakeholder: 1) Which streaming service has the highest volume of "new" content? 2) Which streaming service is the most likely to buy an R-rated movie? 3) Which content type is more likely to be bought? 4) Which genres should future content offerings focus on in order to increase the likelihood of being purchased by a streaming platform?

1.0.2 CONCLUSIONS:

Based on our extensive EDA below, we can attempt to answer the four questions posed: 1) Which streaming service has the highest volume of "new" content? - Based on our analysis, we can look at the distribution of title counts by release year. The platform with the highest proportion of titles from recent years compared to their full library should give us an idea which platform focuses their efforts on new content the most. In this case, Amazon Prime Video has the highest proportion of titles with 2021 release year in their library, so we can conclude that they put high value on having the newest content. 2) Which streaming service is the most likely to buy an R-rated movie? - Based on our analysis, we can look at the distribution of title counts by rating. Similiar to the above question, it's a matter of proportion. Whichever platform has the highest proportion of R-rated movies in their library should signify a willingness to purchase R-rated movies. In this case, Amazon Prime Video has the highest proportion of titles with an R rating, thus showing a high willingness to purchase R-rated movies. An important caveat here is that Amazon Prime Video has the largest library of content, showing a willingness to buy everything. 3) Which content type is more likely to be bought? - This question requires us to drill down a bit further by looking at the proportion of movies to tv shows across all streaming platforms. We can say that movies are more likely to be bought by Netflix, Disney+, and Amazon Prime Video than by Hulu. However, Hulu is the only platform that seems to balance it's offerings of movies and tv shows, showing a greater willingness to purchase tv shows than the other platforms. 4) Which genres should future content offerings focus on in order to increase the likelihood of being purchased by a streaming platform? - When looking at the distribution of genres by title count, we can get an idea of which genres are the most popular. Drama is the genre most popular on Netflix and Amazon Prime Video, while Action/Adventure take the top spot on Hulu and Disney+. To answer this question completely, we would have to answer it on a case-by-case basis for each title.

2 Import Statements

[2]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sqlalchemy import create_engine
from sqlalchemy.sql import text
%matplotlib inline
```

3 Database Read In

First, we must read in the 4 databases we will be working with. 3 are in Excel format and 1 is in PostgreSQL.

3.0.1 Netflix DataFrame

[15]:

```
netflix_df = pd.read_csv('./data/netflix_titles.csv')
```

3.0.2 Hulu DataFrame

[3]:

```
hulu_df = pd.read_csv('./data/hulu_titles.csv')
```

3.0.3 Disney+ DataFrame

[4]:

```
disneyplus_df = pd.read_csv('./data/disney_plus_titles.csv')
```

3.0.4 Amazon Prime Video DataFrame (FROM SQL)

[4]:

```
engine = create_engine('postgresql+psycopg2://postgres:hellosql@localhost/
  ↪amazonprimestreaming')
```

[5]:

```
amazon_sql = "SELECT * FROM amazon_prime_titles"
amazon_df = pd.read_sql(amazon_sql, engine)
```

4 Data Cleaning

We follow the same basic steps for data cleaning each individual data set. The steps we took are as follows: 1) Drop columns that we are uninterested in, such as director, cast, description - Because we cannot guarantee that specific values are correct, such as the spelling of a director's or cast member's name being consistently correct, analysis on those columns is futile.

2) Drop NA values responsibly
   - We must pay attention to the number of records lost to a .dropna() operation. Dropping too many will ruin the validity of the data set.
3) Data Engineering and calculated column creation
   - Due to multiple genres being listed per title in the listed_in column, we had to split them out into their own columns based on a delimiter. We ended up taking the first genre listed as the main genre and removed any secondary genres.

4.0.1 Netflix Data Cleaning

```
[16]: clean_netflix_df = netflix_df.copy()
```

```
[17]: clean_netflix_df = clean_netflix_df.drop(['director', 'cast', 'description'],␣
        ↪axis=1)
```

```
[18]: clean_netflix_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8807 entries, 0 to 8806
Data columns (total 9 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   show_id       8807 non-null   object
 1   type          8807 non-null   object
 2   title         8807 non-null   object
 3   country       7976 non-null   object
 4   date_added    8797 non-null   object
 5   release_year  8807 non-null   int64
 6   rating        8803 non-null   object
 7   duration      8804 non-null   object
 8   listed_in     8807 non-null   object
dtypes: int64(1), object(8)
memory usage: 619.4+ KB
```

```
[19]: new_netflix_df = clean_netflix_df['listed_in'].str.split(',', expand=True)
```

```
[20]: frames = [clean_netflix_df, new_netflix_df]
      final_netflix_df = pd.concat(frames, axis=1, join="inner")
```

```
[21]: new_df = final_netflix_df.drop('listed_in', axis=1)
```

[22]:
```
fin_netflix_df = new_df.rename(columns = {0: 'genre_1', 1: 'genre_2', 2:␣
 ↪'genre_3'})
```

[23]:
```
finished_netflix_df = fin_netflix_df.drop(['genre_2', 'genre_3'], axis=1)
```

[15]:
```
finished_netflix_df.to_csv('./GitHub/GC_DA_Capstone/final_data/netflix_df.csv',␣
 ↪index = False)
```

4.0.2 Hulu Data Cleaning

[16]:
```
clean_hulu_df = hulu_df.copy()
```

[17]:
```
cleaner_hulu_df = clean_hulu_df.drop(['director', 'cast', 'description'],␣
 ↪axis=1)
```

[18]:
```
cleaner_hulu_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3073 entries, 0 to 3072
Data columns (total 9 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   show_id       3073 non-null   object
 1   type          3073 non-null   object
 2   title         3073 non-null   object
 3   country       1620 non-null   object
 4   date_added    3045 non-null   object
 5   release_year  3073 non-null   int64 
 6   rating        2553 non-null   object
 7   duration      2594 non-null   object
 8   listed_in     3073 non-null   object
dtypes: int64(1), object(8)
memory usage: 216.2+ KB
```

[19]:
```
new_hulu_df = cleaner_hulu_df['listed_in'].str.split(',', expand=True)
```

[20]:
```
frames = [cleaner_hulu_df, new_hulu_df]
final_hulu_df = pd.concat(frames, axis=1, join="inner")
```

[21]:
```
fin_hulu_df = final_hulu_df.rename(columns = {0: 'genre_1', 1: 'genre_2', 2:␣
 ↪'genre_3'})
```

[22]:
```
finished_hulu_df = fin_hulu_df.drop(['listed_in','genre_2', 'genre_3'], axis=1)
```

[23]:
```
finished_hulu_df.to_csv('./GitHub/GC_DA_Capstone/final_data/hulu_df.csv', index␣
 ↪= False)
```

4.0.3 Disney+ Data Cleaning

[24]:

```
clean_disney_df = disneyplus_df.copy()
```

[25]:

```
cleaner_disney_df = clean_disney_df.drop(['director', 'cast', 'description'],␣
 ↪axis=1)
```

[26]:

```
cleaner_disney_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1450 entries, 0 to 1449
Data columns (total 9 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   show_id       1450 non-null   object
 1   type          1450 non-null   object
 2   title         1450 non-null   object
 3   country       1231 non-null   object
 4   date_added    1447 non-null   object
 5   release_year  1450 non-null   int64
 6   rating        1447 non-null   object
 7   duration      1450 non-null   object
 8   listed_in     1450 non-null   object
dtypes: int64(1), object(8)
memory usage: 102.1+ KB
```

[27]:

```
cleaned_disney_df = cleaner_disney_df.dropna()
```

[28]:

```
cleaned_disney_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 1228 entries, 2 to 1449
Data columns (total 9 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   show_id       1228 non-null   object
 1   type          1228 non-null   object
 2   title         1228 non-null   object
 3   country       1228 non-null   object
 4   date_added    1228 non-null   object
 5   release_year  1228 non-null   int64
 6   rating        1228 non-null   object
 7   duration      1228 non-null   object
 8   listed_in     1228 non-null   object
dtypes: int64(1), object(8)
memory usage: 95.9+ KB
```

[29]:

```
new_disney_df = cleaned_disney_df['listed_in'].str.split(',', expand=True)
```

```
[30]: frames = [cleaned_disney_df, new_disney_df]
      final_disney_df = pd.concat(frames, axis=1, join="inner")
```

```
[31]: fin_disney_df = final_disney_df.rename(columns = {0: 'genre_1', 1: 'genre_2', 2:
       ↪ 'genre_3'})
```

```
[32]: finished_disney_df = fin_disney_df.drop(['listed_in','genre_2', 'genre_3'],␣
       ↪axis=1)
```

```
[33]: finished_disney_df.to_csv('./GitHub/GC_DA_Capstone/final_data/disney_df.csv',␣
       ↪index = False)
```

4.0.4 Amazon Prime Video Data Cleaning

```
[6]: clean_amazon_df = amazon_df.copy()
```

```
[7]: cleaner_amazon_df = clean_amazon_df.drop(['director', 'cast', 'description'],␣
      ↪axis=1)
```

```
[8]: cleaner_amazon_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9668 entries, 0 to 9667
Data columns (total 9 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   show_id       9668 non-null   object
 1   type          9668 non-null   object
 2   title         9668 non-null   object
 3   country       672 non-null    object
 4   date_added    155 non-null    object
 5   release_year  9668 non-null   int64
 6   rating        9331 non-null   object
 7   duration      9668 non-null   object
 8   listed_in     9668 non-null   object
dtypes: int64(1), object(8)
memory usage: 679.9+ KB
```

```
[9]: new_amazon_df = cleaner_amazon_df['listed_in'].str.split(',', expand=True)
```

```
[10]: frames = [cleaner_amazon_df, new_amazon_df]
      final_amazon_df = pd.concat(frames, axis=1, join="inner")
```

```
[11]: fin_amazon_df = final_amazon_df.rename(columns = {0: 'genre_1', 1: 'genre_2', 2:
       ↪ 'genre_3', 3: 'genre_4', 4: 'genre_5'})
```

```
[12]: finished_amazon_df = fin_amazon_df.drop(['listed_in','genre_2', 'genre_3',␣
       ↪'genre_4', 'genre_5'], axis=1)
```

```
[41]: finished_amazon_df.to_csv('./GitHub/GC_DA_Capstone/final_data/amazon_df.csv',␣
       ↪index = False)
```

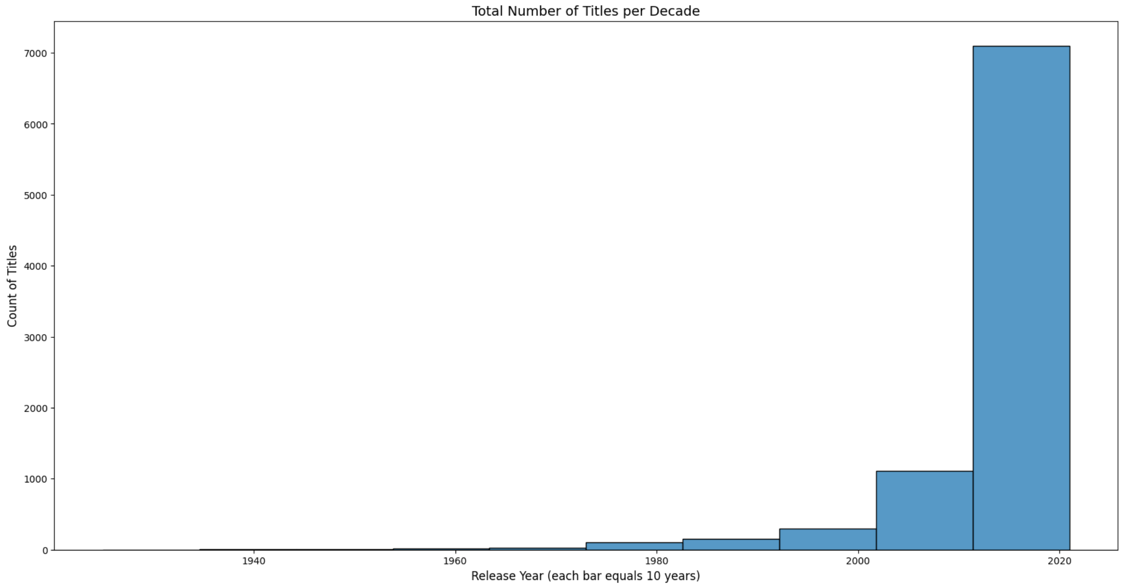
5 Netflix EDA

The figure below shows the distribution of titles by release year. It attempts to answer the question: Which streaming service has the highest volume of "new" content?

```
[42]: plt.figure(figsize = (20.0, 10.0))
      ax = sns.histplot(finished_netflix_df, x = 'release_year',  bins= 10)

      plt.xlabel("Release Year (each bar equals 10 years)", fontsize = 12)
      plt.ylabel("Count of Titles", fontsize = 12)
      plt.title("Total Number of Titles per Decade", fontsize = 14)
```

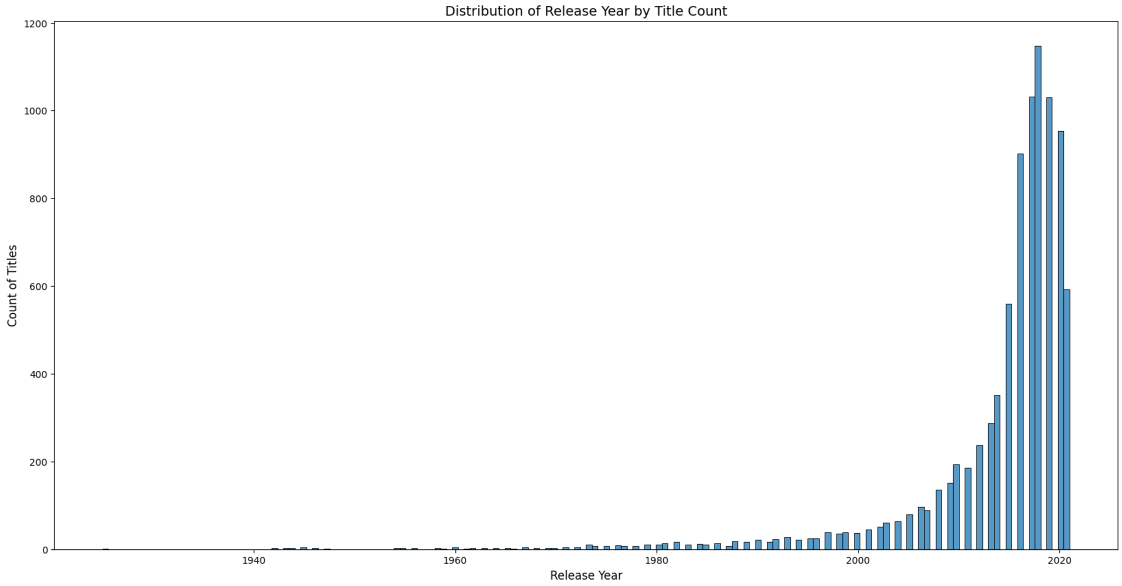
```
[42]: Text(0.5, 1.0, 'Total Number of Titles per Decade')
```



```
[43]: plt.figure(figsize = (20.0, 10.0))
      sns.histplot(finished_netflix_df, x = 'release_year')

      plt.xlabel("Release Year", fontsize = 12)
      plt.ylabel("Count of Titles", fontsize = 12)
      plt.title("Distribution of Release Year by Title Count", fontsize = 14)
```

```
[43]: Text(0.5, 1.0, 'Distribution of Release Year by Title Count')
```

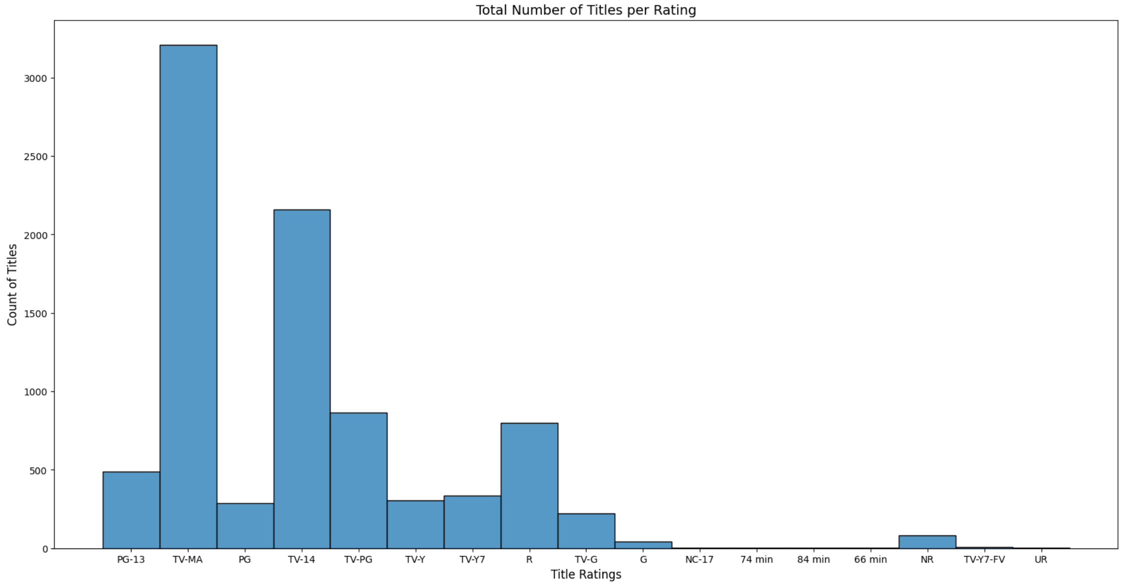


The figure below shows the distribution of titles by rating. It attempts to answer the question: Which streaming service is most likely to purchase an R-rated movie?

```
[44]: plt.figure(figsize=(20.0,10.0))
      sns.histplot(finished_netflix_df, x = 'rating')

      plt.xlabel("Title Ratings", fontsize = 12)
      plt.ylabel("Count of Titles", fontsize = 12)
      plt.title("Total Number of Titles per Rating", fontsize = 14)
```

```
[44]: Text(0.5, 1.0, 'Total Number of Titles per Rating')
```



[24]:

```
finished_netflix_df['rating'].value_counts()
```

[24]:

```
rating
TV-MA       3207
TV-14       2160
TV-PG        863
R            799
PG-13        490
TV-Y7        334
TV-Y         307
PG           287
TV-G         220
NR            80
G             41
TV-Y7-FV       6
NC-17          3
UR             3
74 min         1
84 min         1
66 min         1
Name: count, dtype: int64
```

[45]:

```
genre_date_breakdown = finished_netflix_df.groupby('release_year')['genre_1'].
  ↪value_counts().reset_index()
gd_breakdown_df = pd.DataFrame(genre_date_breakdown)
```

[46]:

```
netflix_only_df = gd_breakdown_df[gd_breakdown_df['release_year'] >= 2007]
```

The figure below shows the genre breakdown per year for all titles in the library. It attempts to give us information that will help answer the question: Which genres should future content offerings focus on in order to increase the likelihood of being purchased by a streaming platform?

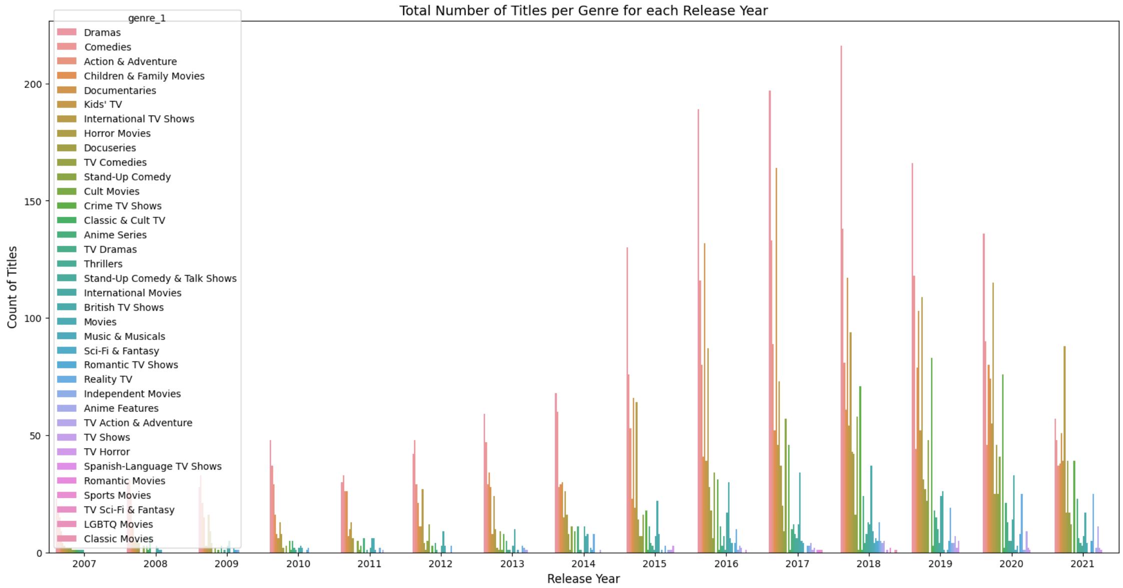
[47]:

```
plt.figure(figsize = (20.0, 10.0))
sns.barplot(netflix_only_df, x = 'release_year', y = 'count', hue = 'genre_1')

plt.xlabel("Release Year", fontsize = 12)
plt.ylabel("Count of Titles", fontsize = 12)
plt.title("Total Number of Titles per Genre for each Release Year", fontsize =␣
  ↪14)
```

[47]:

```
Text(0.5, 1.0, 'Total Number of Titles per Genre for each Release Year')
```



[48]:
```
genre_counts = finished_netflix_df['genre_1'].value_counts()
genre_df = pd.DataFrame(genre_counts)
display(genre_df)
```

```
                              count
genre_1
Dramas                         1600
Comedies                       1210
Action & Adventure              859
Documentaries                   829
International TV Shows          774
Children & Family Movies        605
Crime TV Shows                  399
Kids' TV                        388
Stand-Up Comedy                 334
Horror Movies                   275
British TV Shows                253
Docuseries                      221
Anime Series                    176
International Movies            128
TV Comedies                     120
Reality TV                      120
Classic Movies                   80
TV Dramas                        67
Thrillers                        65
Movies                           57
TV Action & Adventure            40
Stand-Up Comedy & Talk Shows     34
```

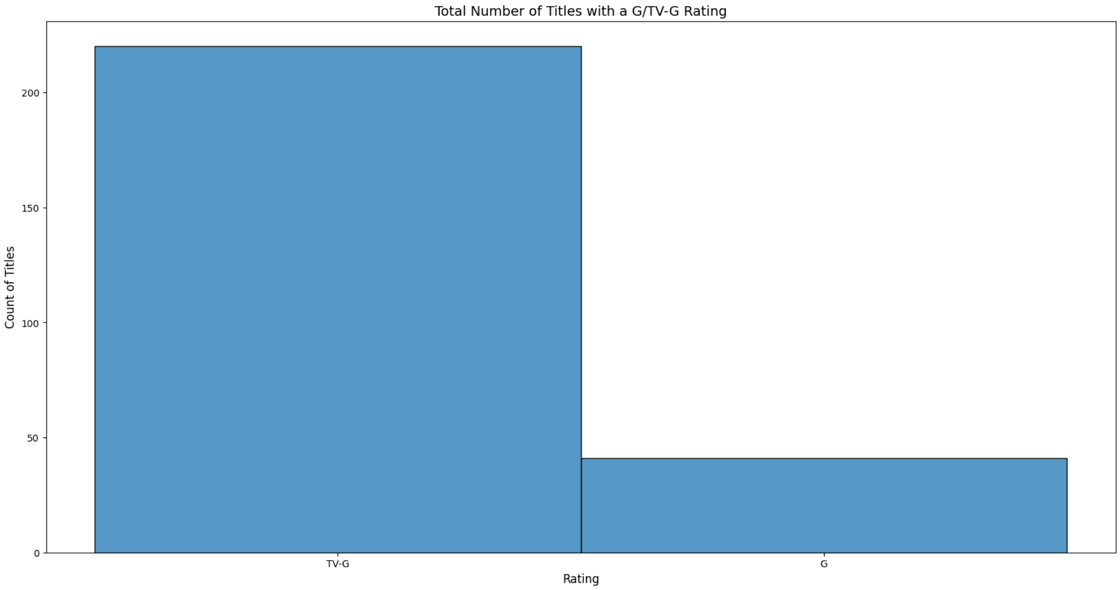
```
Romantic TV Shows               32
Classic & Cult TV               22
Anime Features                  21
Independent Movies              20
Music & Musicals                18
TV Shows                        16
Sci-Fi & Fantasy                13
Cult Movies                     12
TV Horror                       11
Romantic Movies                  3
Spanish-Language TV Shows        2
LGBTQ Movies                     1
TV Sci-Fi & Fantasy              1
Sports Movies                    1
```

[49]:
```
g_tv_g_netflix_only_df = finished_netflix_df[(finished_netflix_df['rating'] ==␣
  ↪'G') | (finished_netflix_df['rating'] == 'TV-G')]
```

[50]:
```
plt.figure(figsize = (20.0, 10.0))
sns.histplot(g_tv_g_netflix_only_df, x = 'rating')

plt.xlabel("Rating", fontsize = 12)
plt.ylabel("Count of Titles", fontsize = 12)
plt.title("Total Number of Titles with a G/TV-G Rating", fontsize = 14)
```

[50]:
```
Text(0.5, 1.0, 'Total Number of Titles with a G/TV-G Rating')
```



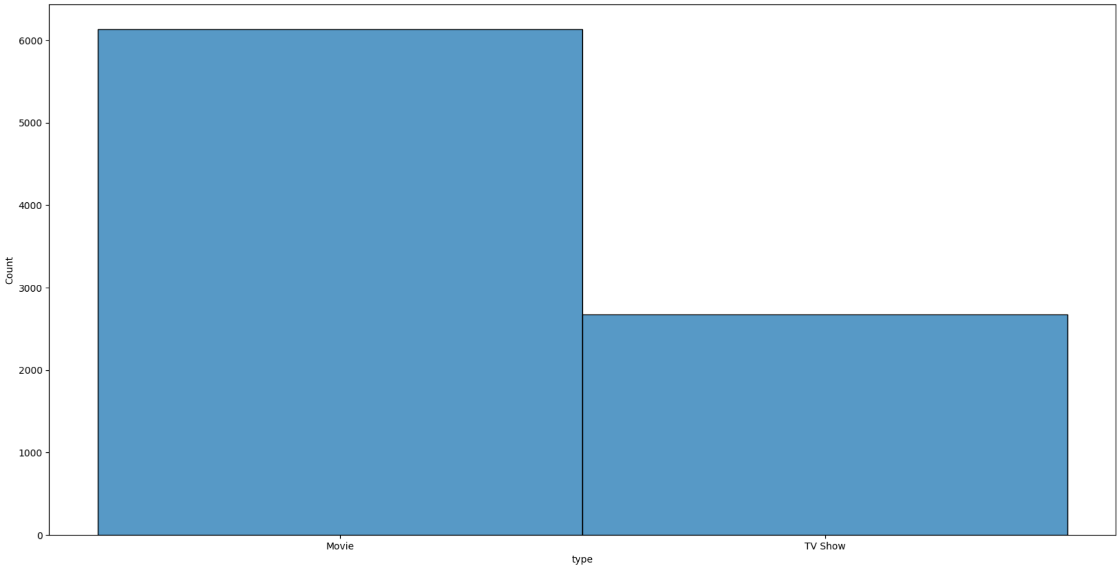
[51]:
```
g_tv_g_netflix_only_df['rating'].value_counts()
```

```
[51]: rating
      TV-G    220
      G        41
      Name: count, dtype: int64
```

The figure below shows the proportion of movies and tv shows currently on the platform. It gives us some information we can use to answer the question: Which content type is more likely to be bought?

```
[52]: plt.figure(figsize = (20.0, 10.0))
      sns.histplot(finished_netflix_df, x='type')
```

```
[52]: <Axes: xlabel='type', ylabel='Count'>
```



```
[53]: finished_netflix_df['type'].value_counts()
```

```
[53]: type
      Movie      6131
      TV Show    2676
      Name: count, dtype: int64
```

6 Hulu EDA

The EDA for each data set follows the steps taken in the Netflix EDA section. All tables and charts show the same concepts, just with different data. We can then compare the results shown in the figure to one another to get an idea if there are key differences between streaming platforms in regards to the questions asked.

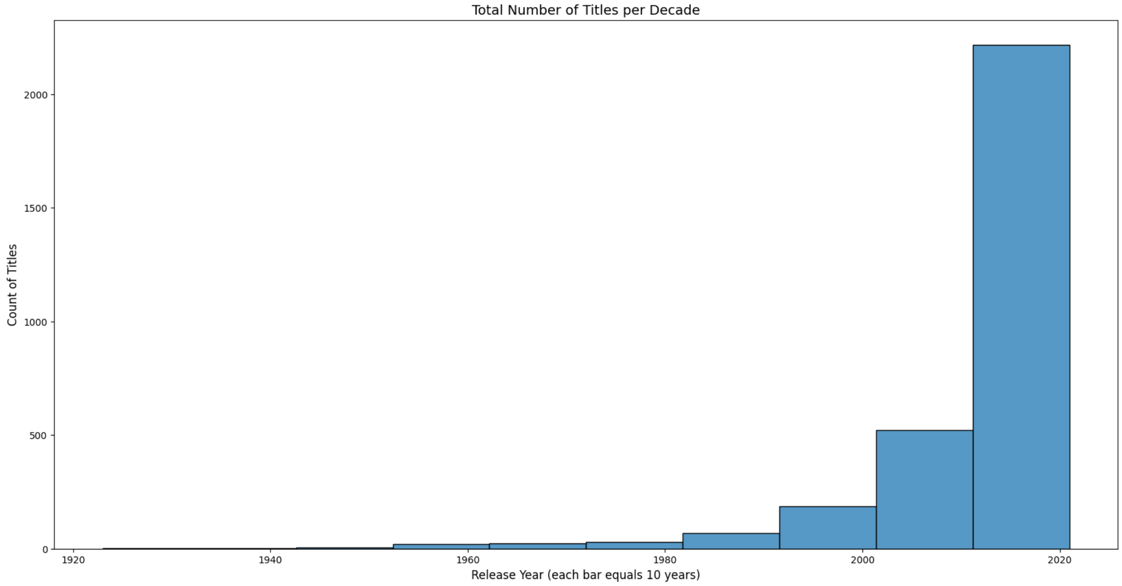
[54]:

```
plt.figure(figsize = (20.0, 10.0))
ax = sns.histplot(finished_hulu_df, x = 'release_year',  bins= 10)

plt.xlabel("Release Year (each bar equals 10 years)", fontsize = 12)
plt.ylabel("Count of Titles", fontsize = 12)
plt.title("Total Number of Titles per Decade", fontsize = 14)
```

[54]:

```
Text(0.5, 1.0, 'Total Number of Titles per Decade')
```



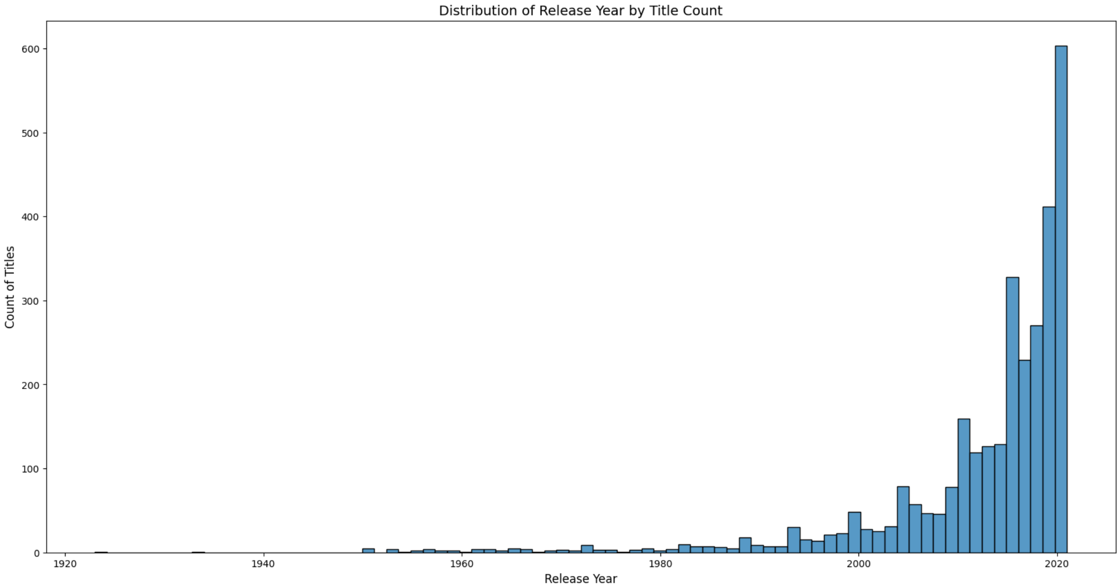
[55]:

```
plt.figure(figsize = (20.0, 10.0))
sns.histplot(finished_hulu_df, x = 'release_year')

plt.xlabel("Release Year", fontsize = 12)
plt.ylabel("Count of Titles", fontsize = 12)
plt.title("Distribution of Release Year by Title Count", fontsize = 14)
```

[55]:

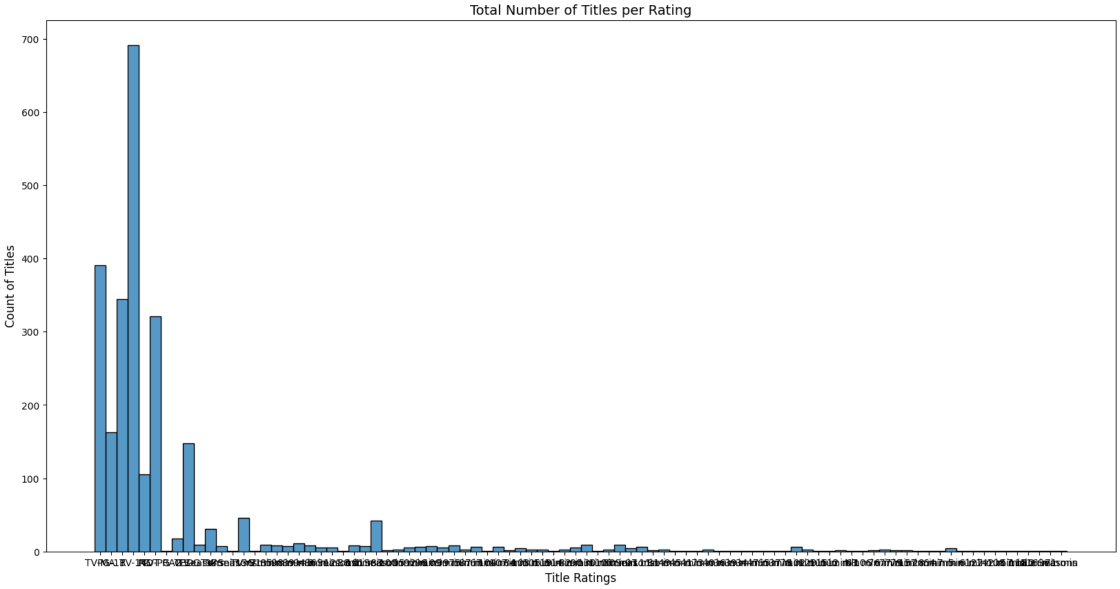
```
Text(0.5, 1.0, 'Distribution of Release Year by Title Count')
```



[56]:
```
plt.figure(figsize=(20.0,10.0))
sns.histplot(finished_hulu_df, x = 'rating')

plt.xlabel("Title Ratings", fontsize = 12)
plt.ylabel("Count of Titles", fontsize = 12)
plt.title("Total Number of Titles per Rating", fontsize = 14)
```

[56]:
```
Text(0.5, 1.0, 'Total Number of Titles per Rating')
```



[57]:

```
finished_hulu_df['rating'].value_counts()
```

[57]:

```
rating
TV-14      691
TV-MA      391
R          345
TV-PG      321
PG-13      163
         …
34 min       1
47 min       1
65 min       1
37 min       1
71 min       1
Name: count, Length: 88, dtype: int64
```

[58]:

```
genre_date_breakdown = finished_hulu_df.groupby('release_year')['genre_1'].
  ↪value_counts().reset_index()
gd_breakdown_df = pd.DataFrame(genre_date_breakdown)
```

[59]:

```
hulu_only_df = gd_breakdown_df[gd_breakdown_df['release_year'] >= 2007]
```

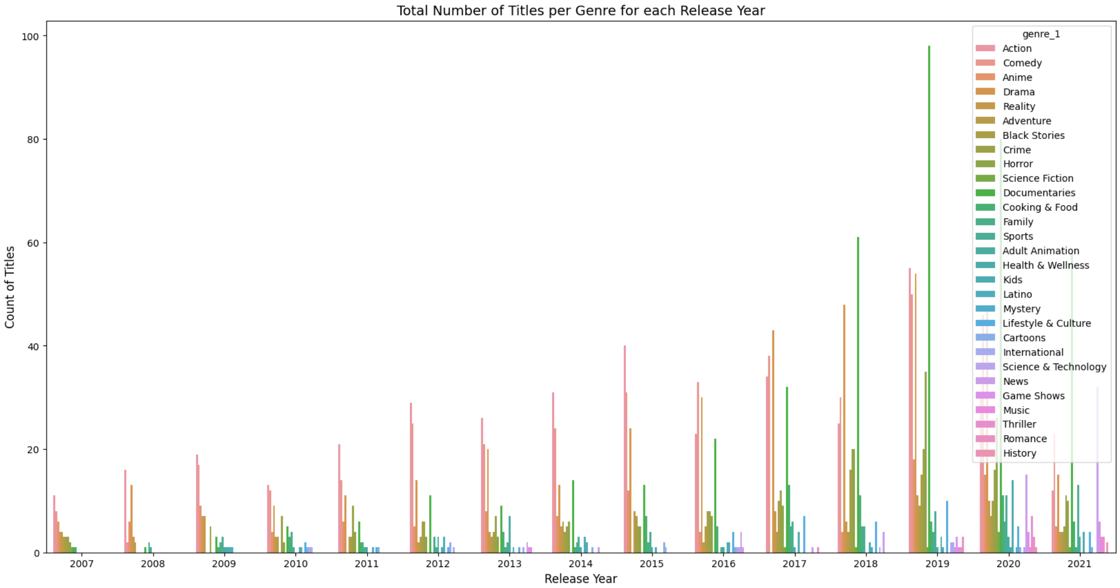
[60]:

```
plt.figure(figsize = (20.0, 10.0))
sns.barplot(hulu_only_df, x = 'release_year', y = 'count', hue = 'genre_1')

plt.xlabel("Release Year", fontsize = 12)
plt.ylabel("Count of Titles", fontsize = 12)
plt.title("Total Number of Titles per Genre for each Release Year", fontsize =␣
  ↪14)
```

[60]:

```
Text(0.5, 1.0, 'Total Number of Titles per Genre for each Release Year')
```



```
[61]: genre_counts = finished_hulu_df['genre_1'].value_counts()
      genre_df = pd.DataFrame(genre_counts)
      display(genre_df)
```

```
                      count
genre_1
Action                  555
Comedy                  468
Documentaries           433
Drama                   415
Crime                   162
Horror                  149
Anime                   131
Black Stories           105
Adventure                81
Cooking & Food           80
Reality                  74
Sports                   74
News                     65
Family                   49
Lifestyle & Culture      43
Kids                     41
Adult Animation          29
Classics                 23
Game Shows               15
Cartoons                 14
Thriller                 13
Latino                   12
```

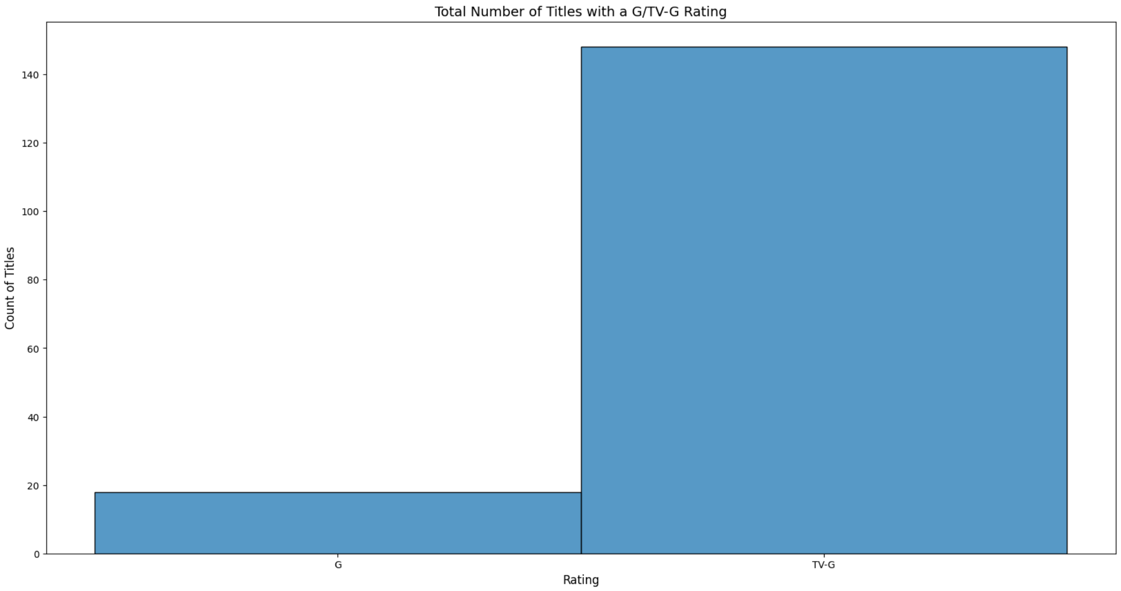
```
Science Fiction          10
International             7
Romance                   6
Science & Technology      6
Music                     6
History                   3
Mystery                   2
Health & Wellness         2
```

[62]:
```
g_tv_g_hulu_only_df = finished_hulu_df[(finished_hulu_df['rating'] == 'G') |␣
  ↪(finished_hulu_df['rating'] == 'TV-G')]
```

[63]:
```
plt.figure(figsize = (20.0, 10.0))
sns.histplot(g_tv_g_hulu_only_df, x = 'rating')

plt.xlabel("Rating", fontsize = 12)
plt.ylabel("Count of Titles", fontsize = 12)
plt.title("Total Number of Titles with a G/TV-G Rating", fontsize = 14)
```

[63]:
```
Text(0.5, 1.0, 'Total Number of Titles with a G/TV-G Rating')
```

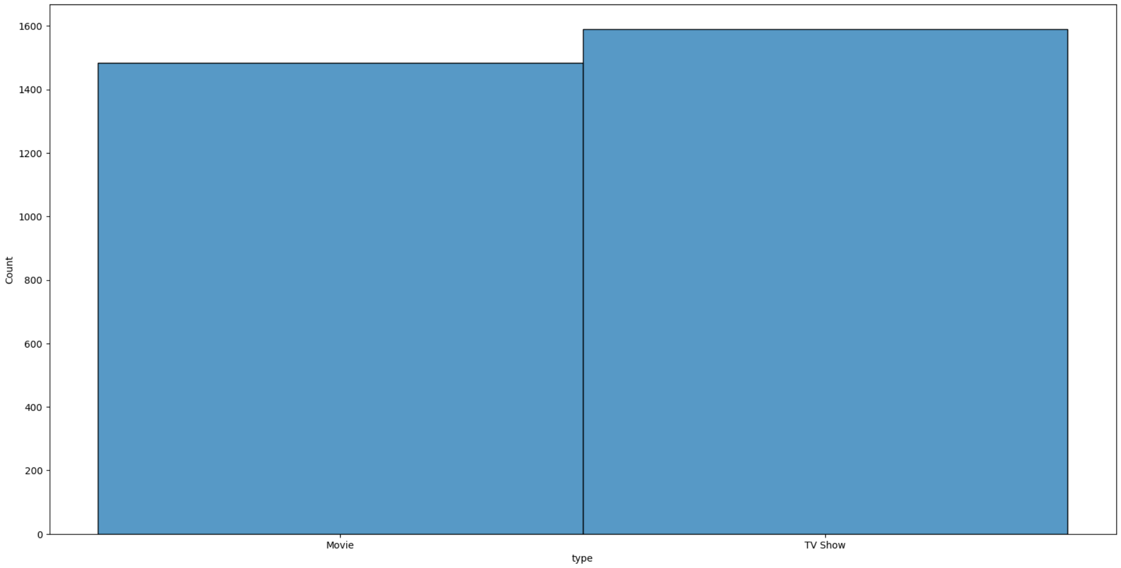


[64]:
```
g_tv_g_hulu_only_df['rating'].value_counts()
```

[64]:
```
rating
TV-G    148
G        18
Name: count, dtype: int64
```

[65]:
```
plt.figure(figsize = (20.0, 10.0))
sns.histplot(finished_hulu_df, x='type')
```

[65]:
```
<Axes: xlabel='type', ylabel='Count'>
```

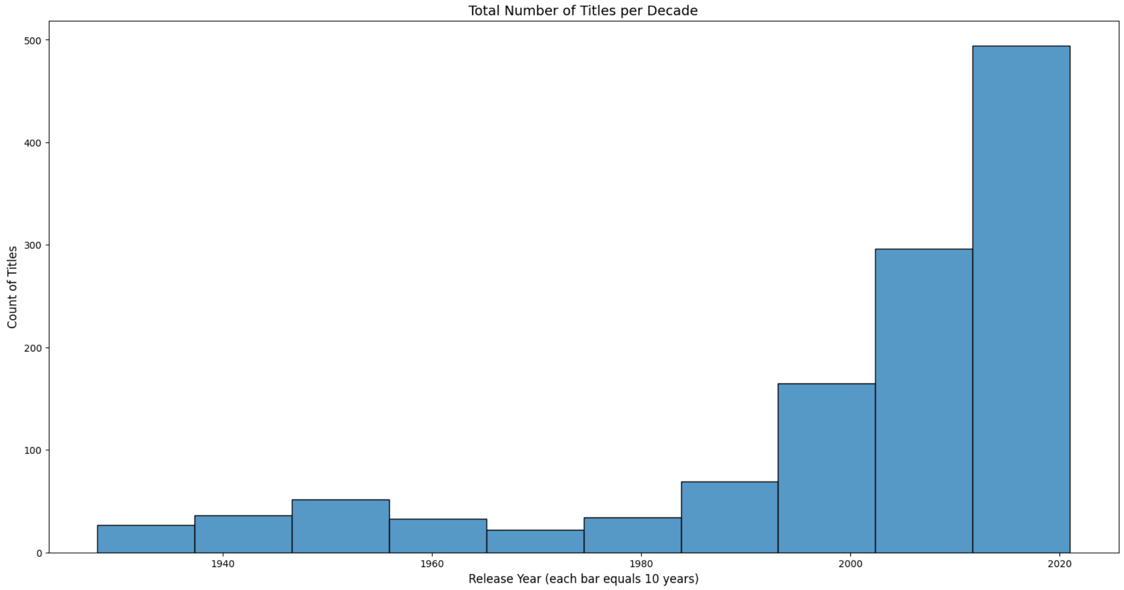


7 Disney+ EDA

[66]:
```
plt.figure(figsize = (20.0, 10.0))
ax = sns.histplot(finished_disney_df, x = 'release_year',  bins= 10)

plt.xlabel("Release Year (each bar equals 10 years)", fontsize = 12)
plt.ylabel("Count of Titles", fontsize = 12)
plt.title("Total Number of Titles per Decade", fontsize = 14)
```

[66]:
```
Text(0.5, 1.0, 'Total Number of Titles per Decade')
```



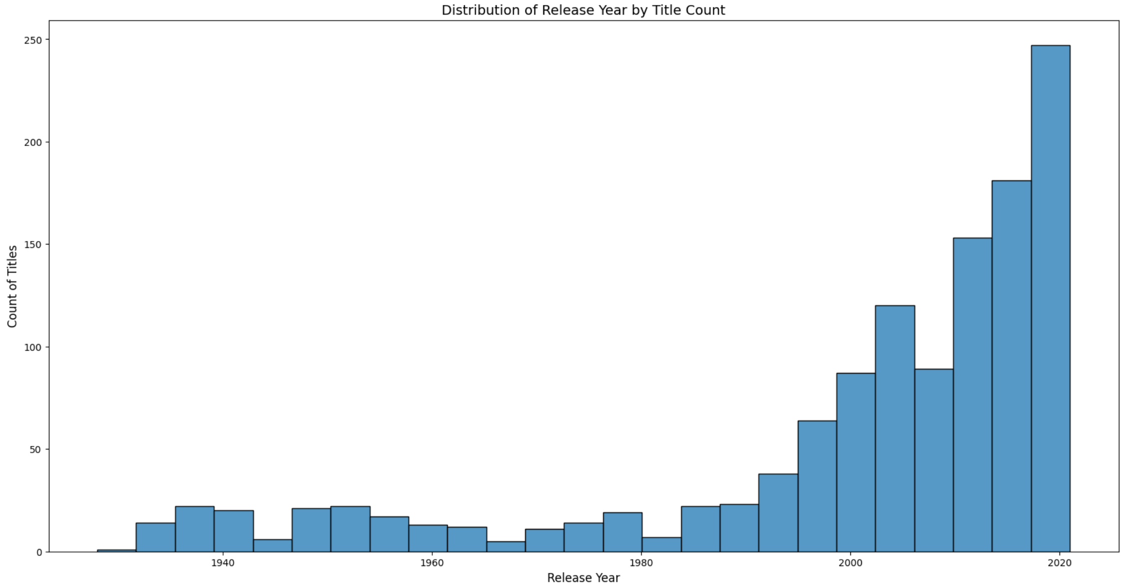
[67]:

```
plt.figure(figsize = (20.0, 10.0))
sns.histplot(finished_disney_df, x = 'release_year')

plt.xlabel("Release Year", fontsize = 12)
plt.ylabel("Count of Titles", fontsize = 12)
plt.title("Distribution of Release Year by Title Count", fontsize = 14)
```

[67]:

```
Text(0.5, 1.0, 'Distribution of Release Year by Title Count')
```



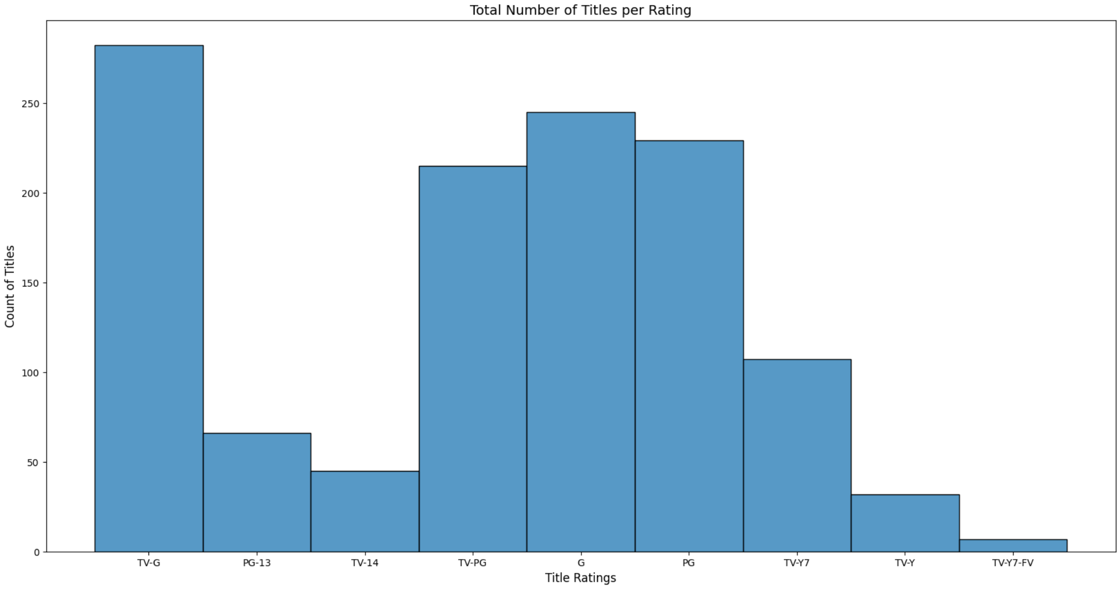
[68]:

```
plt.figure(figsize=(20.0,10.0))
sns.histplot(finished_disney_df, x = 'rating')

plt.xlabel("Title Ratings", fontsize = 12)
plt.ylabel("Count of Titles", fontsize = 12)
plt.title("Total Number of Titles per Rating", fontsize = 14)
```

[68]:

```
Text(0.5, 1.0, 'Total Number of Titles per Rating')
```



[69]:

```
finished_disney_df['rating'].value_counts()
```

[69]:

```
rating
TV-G        282
G           245
PG          229
TV-PG       215
TV-Y7       107
PG-13        66
TV-14        45
TV-Y         32
TV-Y7-FV      7
Name: count, dtype: int64
```

[70]:

```
genre_date_breakdown = finished_disney_df.groupby('release_year')['genre_1'].
  ↪value_counts().reset_index()
gd_breakdown_df = pd.DataFrame(genre_date_breakdown)
```

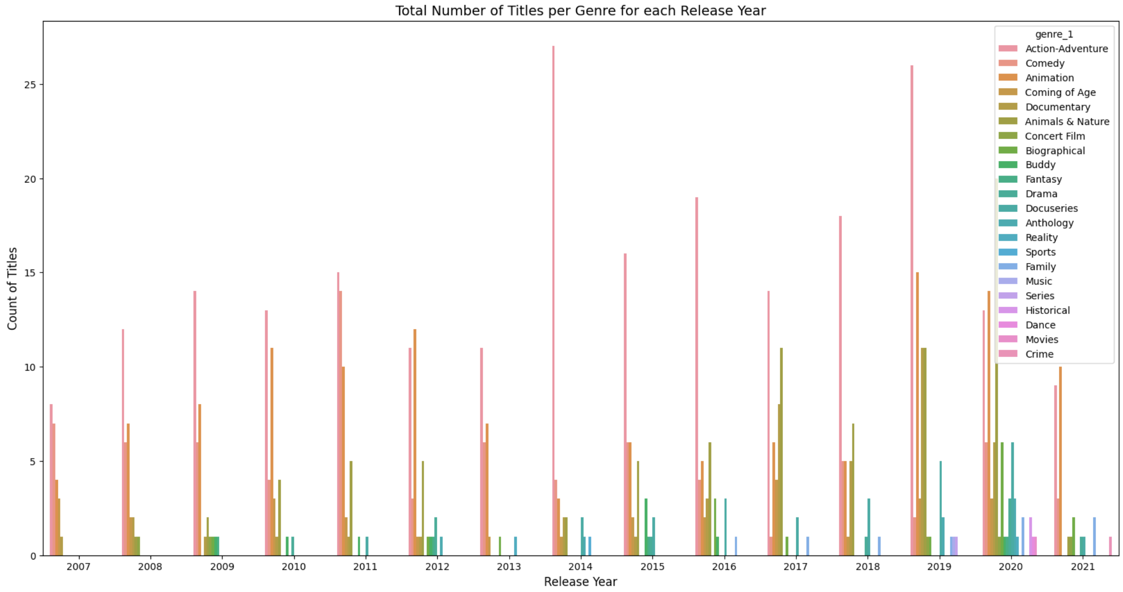
[71]:

```
disney_only_df = gd_breakdown_df[gd_breakdown_df['release_year'] >= 2007]
```

```
[72]: plt.figure(figsize = (20.0, 10.0))
      sns.barplot(disney_only_df, x = 'release_year', y = 'count', hue = 'genre_1')

      plt.xlabel("Release Year", fontsize = 12)
      plt.ylabel("Count of Titles", fontsize = 12)
      plt.title("Total Number of Titles per Genre for each Release Year", fontsize =␣
       ↪14)
```

```
[72]: Text(0.5, 1.0, 'Total Number of Titles per Genre for each Release Year')
```



```
[73]: genre_counts = finished_disney_df['genre_1'].value_counts()
      genre_df = pd.DataFrame(genre_counts)
      display(genre_df)
```

```
                    count
genre_1
Action-Adventure      408
Animation             280
Comedy                180
Animals & Nature      113
Coming of Age          53
Documentary            51
Biographical           31
Docuseries             25
Drama                  24
Buddy                  19
Family                 13
Anthology               7
```

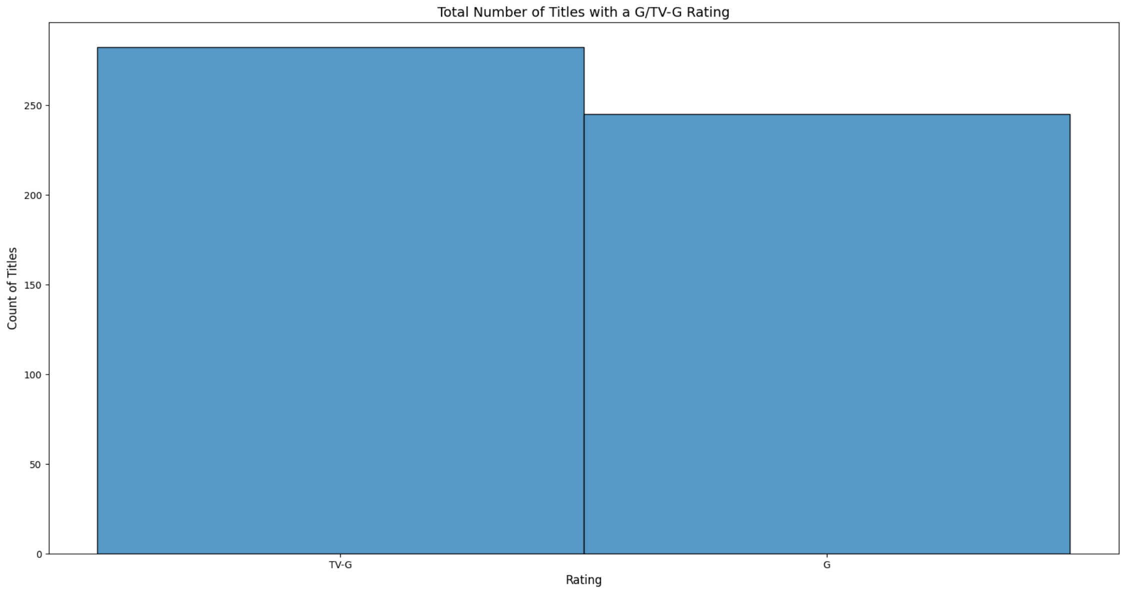
```
Concert Film          5
Fantasy               5
Crime                 3
Historical            2
Reality               2
Movies                1
Dance                 1
Musical               1
Sports                1
Music                 1
Kids                  1
Series                1
```

[74]:
```
g_tv_g_disney_only_df = finished_disney_df[(finished_disney_df['rating'] ==␣
 ↪'G') | (finished_disney_df['rating'] == 'TV-G')]
```

[75]:
```
plt.figure(figsize = (20.0, 10.0))
sns.histplot(g_tv_g_disney_only_df, x = 'rating')

plt.xlabel("Rating", fontsize = 12)
plt.ylabel("Count of Titles", fontsize = 12)
plt.title("Total Number of Titles with a G/TV-G Rating", fontsize = 14)
```

[75]:
```
Text(0.5, 1.0, 'Total Number of Titles with a G/TV-G Rating')
```



[76]:
```
g_tv_g_disney_only_df['rating'].value_counts()
```

[76]:

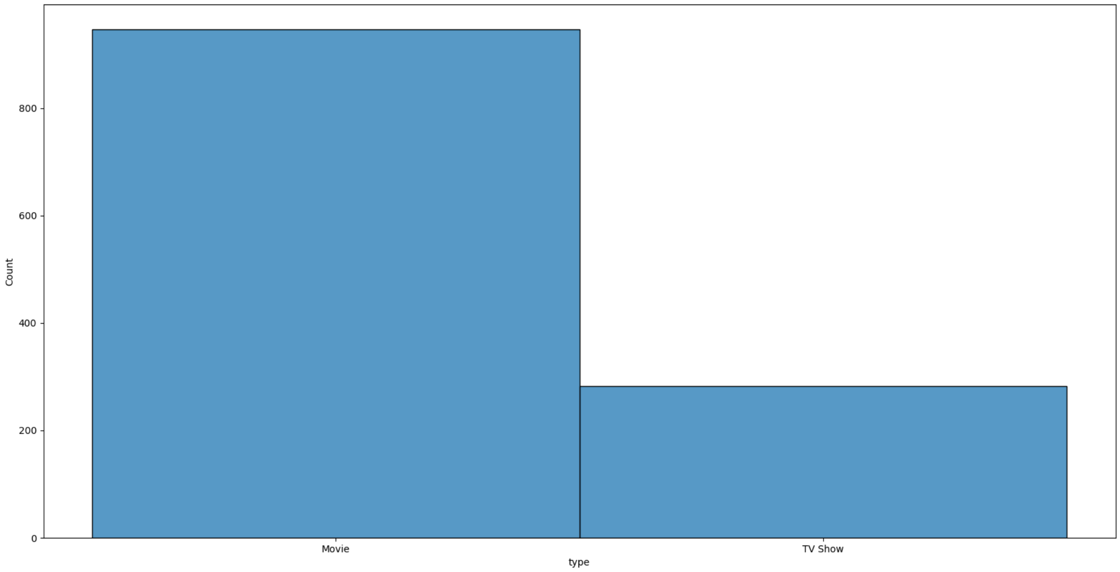
```
rating
TV-G    282
G       245
Name: count, dtype: int64
```

[77]:

```
plt.figure(figsize = (20.0, 10.0))
sns.histplot(finished_disney_df, x='type')
```

[77]:

```
<Axes: xlabel='type', ylabel='Count'>
```



8 Amazon Prime Video EDA

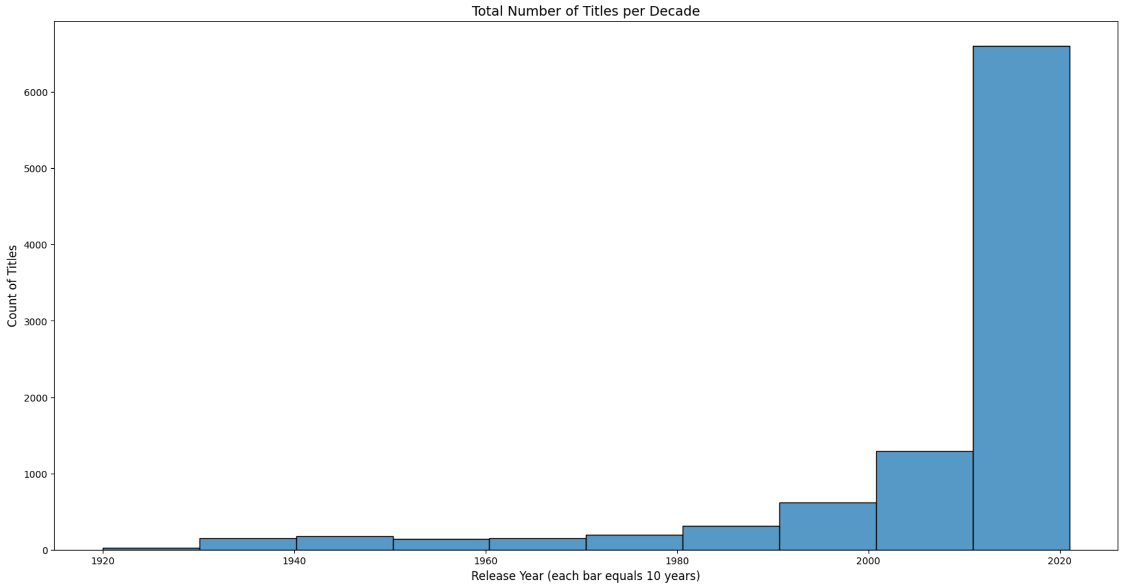
[78]:

```
plt.figure(figsize = (20.0, 10.0))
ax = sns.histplot(finished_amazon_df, x = 'release_year',  bins= 10)

plt.xlabel("Release Year (each bar equals 10 years)", fontsize = 12)
plt.ylabel("Count of Titles", fontsize = 12)
plt.title("Total Number of Titles per Decade", fontsize = 14)
```

[78]:

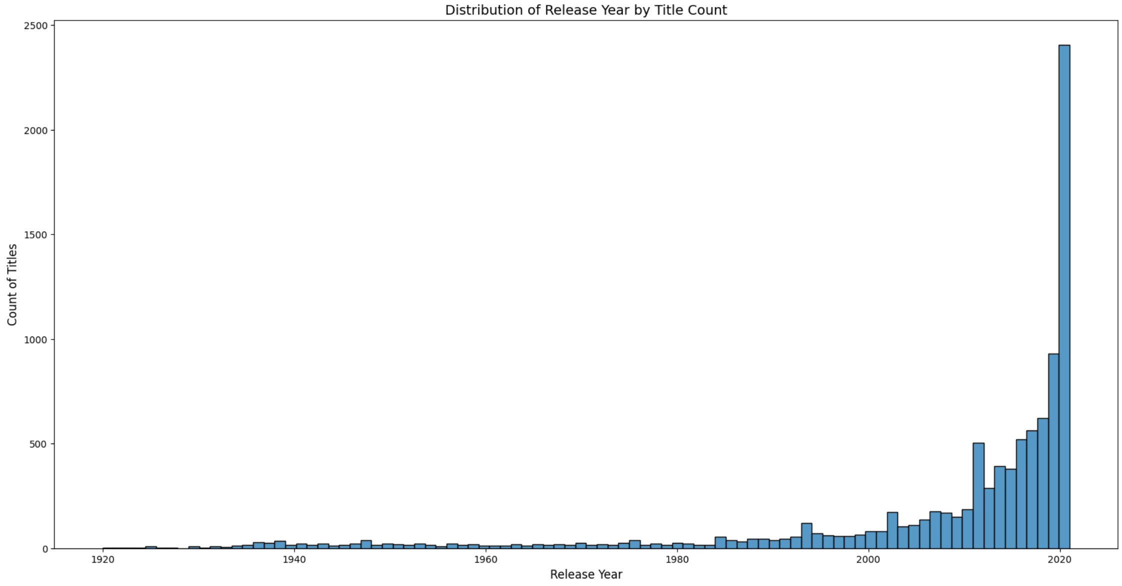
```
Text(0.5, 1.0, 'Total Number of Titles per Decade')
```



```
[79]: plt.figure(figsize = (20.0, 10.0))
      sns.histplot(finished_amazon_df, x = 'release_year')

      plt.xlabel("Release Year", fontsize = 12)
      plt.ylabel("Count of Titles", fontsize = 12)
      plt.title("Distribution of Release Year by Title Count", fontsize = 14)
```

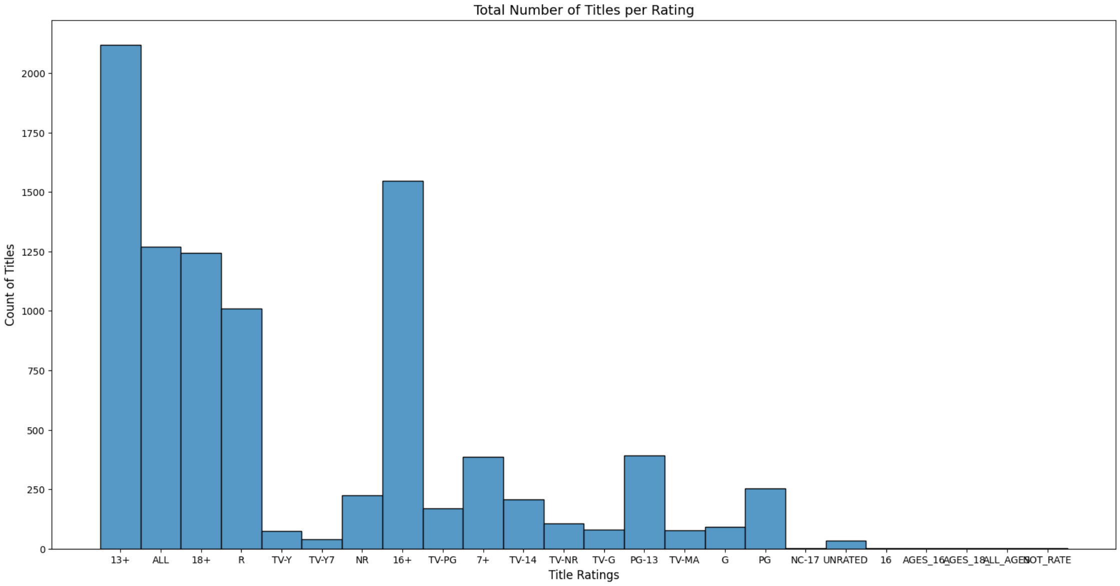
```
[79]: Text(0.5, 1.0, 'Distribution of Release Year by Title Count')
```



[80]:
```
plt.figure(figsize=(20.0,10.0))
sns.histplot(finished_amazon_df, x = 'rating')

plt.xlabel("Title Ratings", fontsize = 12)
plt.ylabel("Count of Titles", fontsize = 12)
plt.title("Total Number of Titles per Rating", fontsize = 14)
```

[80]:
```
Text(0.5, 1.0, 'Total Number of Titles per Rating')
```



[81]:
```
finished_amazon_df['rating'].value_counts()
```

[81]:
```
rating
13+         2117
16+         1547
ALL         1268
18+         1243
R           1010
PG-13        393
7+           385
PG           253
NR           223
TV-14        208
TV-PG        169
TV-NR        105
G             93
TV-G          81
TV-MA         77
TV-Y          74
```

```
TV-Y7          39
UNRATED        33
NC-17           3
AGES_18_        3
NOT_RATE        3
AGES_16_        2
16              1
ALL_AGES        1
Name: count, dtype: int64
```

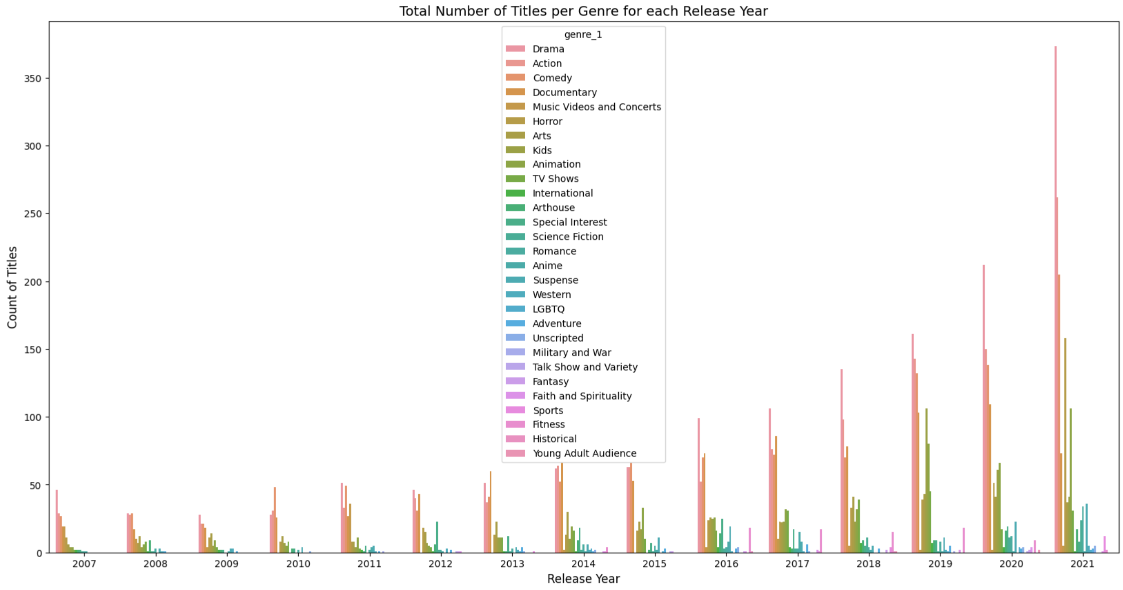
[82]:
```
genre_date_breakdown = finished_amazon_df.groupby('release_year')['genre_1'].
  ↪value_counts().reset_index()
gd_breakdown_df = pd.DataFrame(genre_date_breakdown)
```

[83]:
```
amazon_only_df = gd_breakdown_df[gd_breakdown_df['release_year'] >= 2007]
```

[84]:
```
plt.figure(figsize = (20.0, 10.0))
sns.barplot(amazon_only_df, x = 'release_year', y = 'count', hue = 'genre_1')

plt.xlabel("Release Year", fontsize = 12)
plt.ylabel("Count of Titles", fontsize = 12)
plt.title("Total Number of Titles per Genre for each Release Year", fontsize =␣
  ↪14)
```

[84]:
```
Text(0.5, 1.0, 'Total Number of Titles per Genre for each Release Year')
```



[85]:
```
genre_counts = finished_amazon_df['genre_1'].value_counts()
genre_df = pd.DataFrame(genre_counts)
display(genre_df)
```

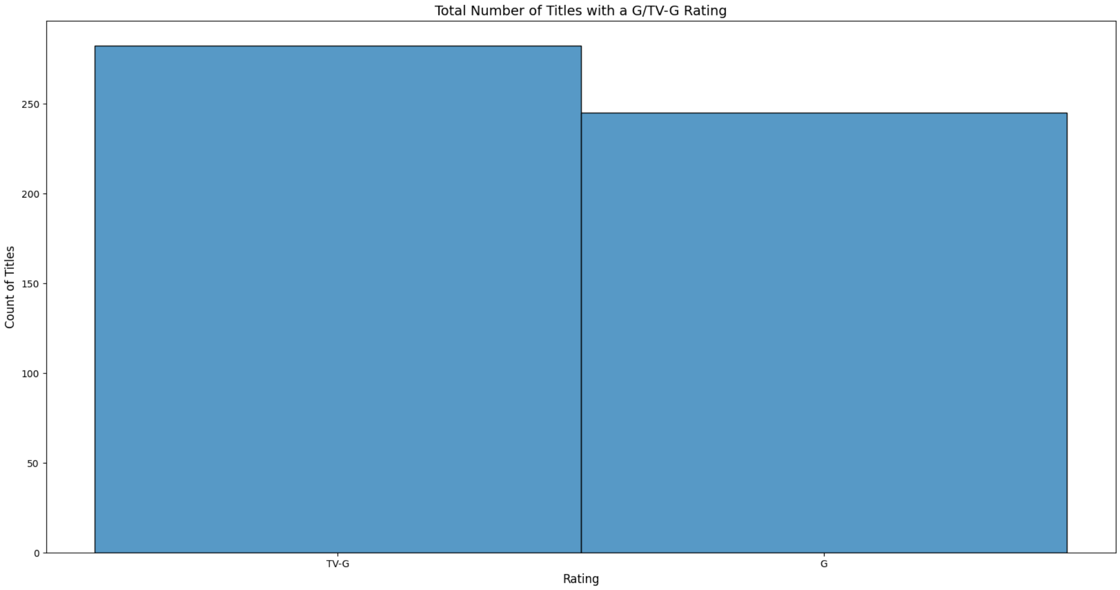
```
                           count
genre_1
Drama                       2216
Action                      1657
Comedy                      1475
Documentary                  913
Horror                       535
Animation                    498
Arts                         457
Kids                         373
TV Shows                     263
Suspense                     194
Special Interest             188
Arthouse                     132
Romance                      126
Music Videos and Concerts    103
Western                      102
Science Fiction               85
Fitness                       83
Adventure                     71
International                 47
Anime                         44
Unscripted                    29
Sports                        19
Fantasy                       18
Faith and Spirituality        13
LGBTQ                         13
Military and War               5
Young Adult Audience           3
Talk Show and Variety          3
Historical                     3
```

[86]:
```
g_tv_g_amazon_only_df = finished_disney_df[(finished_disney_df['rating'] ==␣
 ↪'G') | (finished_disney_df['rating'] == 'TV-G')]
```

[87]:
```
plt.figure(figsize = (20.0, 10.0))
sns.histplot(g_tv_g_amazon_only_df, x = 'rating')

plt.xlabel("Rating", fontsize = 12)
plt.ylabel("Count of Titles", fontsize = 12)
plt.title("Total Number of Titles with a G/TV-G Rating", fontsize = 14)
```

[87]:
```
Text(0.5, 1.0, 'Total Number of Titles with a G/TV-G Rating')
```

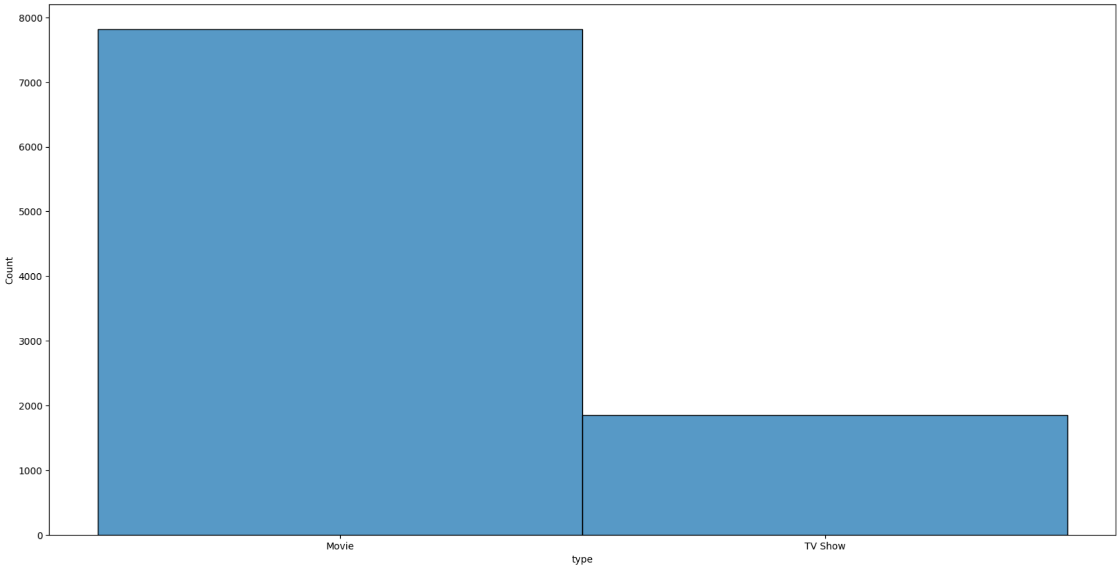


```
[88]: g_tv_g_amazon_only_df['rating'].value_counts()
```

```
[88]: rating
      TV-G    282
      G       245
      Name: count, dtype: int64
```

```
[13]: plt.figure(figsize = (20.0, 10.0))
      sns.histplot(finished_amazon_df, x='type')
```

```
[13]: <Axes: xlabel='type', ylabel='Count'>
```


Count
Movie
TV Show
type