EDA with Netflix

Data Overview

THE NETFLIX EDA PROJECT REVOLVES AROUND EXPLORING AND ANALYZING A DATASET RELATED TO NETFLIX CONTENT. THE DATASET LIKELY INCLUDES INFORMATION ABOUT MOVIES AND TV SHOWS AVAILABLE ON THE PLATFORM. THE PURPOSE OF THE PROJECT IS TO PERFORM EXPLORATORY DATA ANALYSIS TO EXTRACT MEANINGFUL INSIGHTS AND DRAW CONCLUSIONS FROM THE DATA.

Importing Libraries

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.graph_objects as py
import warnings
warnings.filterwarnings("ignore")
```

Load Data

In [2]:
```
data = pd.read_csv("netflix_titles.csv")
```

In [3]:
```
data
```

Out[3]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	descri
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	September 25, 2021	2020	PG-13	90 min	Documentaries	A father the e hi fil
1	s2	TV Show	Blood & Water	NaN	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	September 24, 2021	2021	TV-MA	2 Seasons	International TV Shows, TV Dramas, TV Mysteries	cro path pa Cape
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	NaN	September 24, 2021	2021	TV-MA	1 Season	Crime TV Shows, International TV Shows, TV Act...	To p his f fi pov drug
3	s4	TV Show	Jailbirds New Orleans	NaN	NaN	NaN	September 24, 2021	2021	TV-MA	1 Season	Docuseries, Reality TV	F flirta and ta a
4	s5	TV Show	Kota Factory	NaN	Mayur More, Jitendra Kumar, Ranjan Raj, Alam K...	India	September 24, 2021	2021	TV-MA	2 Seasons	International TV Shows, Romantic TV Shows, TV ...	In a coa ce kno tra
...	...	...	...	...	...	...	...	...	...	...	...	
8802	s8803	Movie	Zodiac	David Fincher	Mark Ruffalo, Jake Gyllenhaal, Robert Downey J...	United States	November 20, 2019	2007	R	158 min	Cult Movies, Dramas, Thrillers	A po carto a rep ar
8803	s8804	TV Show	Zombie Dumb	NaN	NaN	NaN	July 1, 2019	2018	TV-Y7	2 Seasons	Kids' TV, Korean TV Shows, TV Comedies	While alon sp to your
8804	s8805	Movie	Zombieland	Ruben Fleischer	Jesse Eisenberg, Woody Harrelson, Emma Stone, ...	United States	November 1, 2019	2009	R	88 min	Comedies, Horror Movies	Look surviv world ov
8805	s8806	Movie	Zoom	Peter Hewitt	Tim Allen, Courteney Cox, Chevy Chase, Kate Ma...	United States	January 11, 2020	2006	PG	88 min	Children & Family Movies, Comedies	Dra from c f superh
8806	s8807	Movie	Zubaan	Mozez Singh	Vicky Kaushal, Sarah-Jane Dias, Raaghav Chanan...	India	March 2, 2019	2015	TV-14	111 min	Dramas, International Movies, Music & Musicals	A sc but boy w his wa

8807 rows × 12 columns

Top 10 Rows

In [4]:
```
data.head(10)
```

Out[4]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	descrip
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	September 25, 2021	2020	PG-13	90 min	Documentaries	A father n the e his filn
1	s2	TV Show	Blood & Water	NaN	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	September 24, 2021	2021	TV-MA	2 Seasons	International TV Shows, TV Dramas, TV Mysteries	cros paths par Cape T
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	NaN	September 24, 2021	2021	TV-MA	1 Season	Crime TV Shows, International TV Shows, TV Act...	To pr his fa fr pow drug
3	s4	TV Show	Jailbirds New Orleans	NaN	NaN	NaN	September 24, 2021	2021	TV-MA	1 Season	Docuseries, Reality TV	Fe flirta and ta a
4	s5	TV Show	Kota Factory	NaN	Mayur More, Jitendra Kumar, Ranjan Raj, Alam K...	India	September 24, 2021	2021	TV-MA	2 Seasons	International TV Shows, Romantic TV Shows, TV ...	In a c coac cer know tra
5	s6	TV Show	Midnight Mass	Mike Flanagan	Kate Siegel, Zach Gilford, Hamish Linklater, H...	NaN	September 24, 2021	2021	TV-MA	1 Season	TV Dramas, TV Horror, TV Mysteries	The a charisr y priest b
6	s7	Movie	My Little Pony: A New Generation	Robert Cullen, José Luis Ucha	Vanessa Hudgens, Kimiko Glenn, James Marsden, ...	NaN	September 24, 2021	2021	PG	91 min	Children & Family Movies	Eques divided a br eyed
7	s8	Movie	Sankofa	Haile Gerima	Kofi Ghanaba, Oyafunmike Ogunlano, Alexandra D...	United States, Ghana, Burkina Faso, United Kin...	September 24, 2021	1993	TV-MA	125 min	Dramas, Independent Movies, International Movies	On a p sho Ghana Ame mode
8	s9	TV Show	The Great British Baking Show	Andy Devonshire	Mel Giedroyc, Sue Perkins, Mary Berry, Paul Ho...	United Kingdom	September 24, 2021	2021	TV-14	9 Seasons	British TV Shows, Reality TV	A tale bat ama bakers of
9	s10	Movie	The Starling	Theodore Melfi	Melissa McCarthy, Chris O'Dowd, Kevin Kline, T...	United States	September 24, 2021	2021	PG-13	104 min	Comedies, Dramas	A wo adjusti life at conte

Description of data

In [5]:
```
data.describe()
```

Out[5]:

	release_year
count	8807.000000
mean	2014.180198
std	8.819312
min	1925.000000
25%	2013.000000
50%	2017.000000
75%	2019.000000
max	2021.000000

Information of data

In [6]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8807 entries, 0 to 8806
Data columns (total 12 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   director      6173 non-null   object
 4   cast          7982 non-null   object
 5   country       7976 non-null   object
 6   date_added    8797 non-null   object
 7   release_year  8807 non-null   int64 
 8   rating        8803 non-null   object
 9   duration      8804 non-null   object
 10  listed_in     8807 non-null   object
 11  description   8807 non-null   object
dtypes: int64(1), object(11)
memory usage: 825.8+ KB
```

checking missing values

In [7]:
```
data.isnull().sum()/len(data)*100
```

Out[7]:
```
show_id          0.000000
type             0.000000
title            0.000000
director        29.908028
cast             9.367549
country          9.435676
date_added       0.113546
release_year     0.000000
rating           0.045418
duration         0.034064
listed_in        0.000000
description      0.000000
dtype: float64
```

Handle Missing Values

In [8]:
```
data["director"]=data["director"].fillna("unkown")
```

In [9]:
```
data["cast"]=data["cast"].fillna("unkown")
```

In [10]:
```
data["country"]=data["country"].fillna("country unavailable")
```

Drops rows to handle missing values

In [11]:
```
data = data.dropna(subset=['rating', 'date_added','duration'])
```

In [12]:
```
data['date_added'] = data['date_added'].str.strip()
# Convert to datetime format
data['date_added'] = pd.to_datetime(data['date_added'], format='mixed')

# Extract year, month, and day
data['year'] = data['date_added'].dt.year
data['month'] = data['date_added'].dt.month
data['day'] = data['date_added'].dt.day
```

In [13]:
```
movie_data = data[data['type'] == 'Movie']
tv_show_data = data[data['type'] == 'TV Show']
```

In [14]:
```
# Movies have durations like '90 min', extract the numeric part
movie_data['duration_minutes'] = movie_data['duration'].str.extract('(\d+)').astype(float)

# TV Shows have durations like '1 Season' or '2 Seasons', extract the numeric part
tv_show_data['duration_seasons'] = tv_show_data['duration'].str.extract('(\d+)').astype(float)
```

Drop Columns

In [15]:
```
data.drop(["show_id","date_added","description","release_year"],axis=1,inplace=True)
```

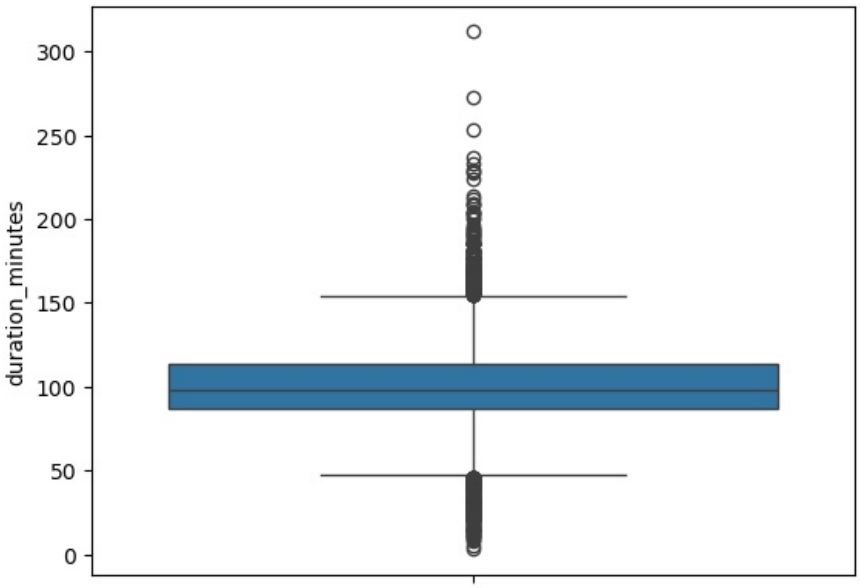
Clean Values

In [16]:
```
data.isnull().sum()/len(data)*100
```

Out[16]:
```
type          0.0
title         0.0
director      0.0
cast          0.0
country       0.0
rating        0.0
duration      0.0
listed_in     0.0
year          0.0
month         0.0
day           0.0
dtype: float64
```

Checking Outliers

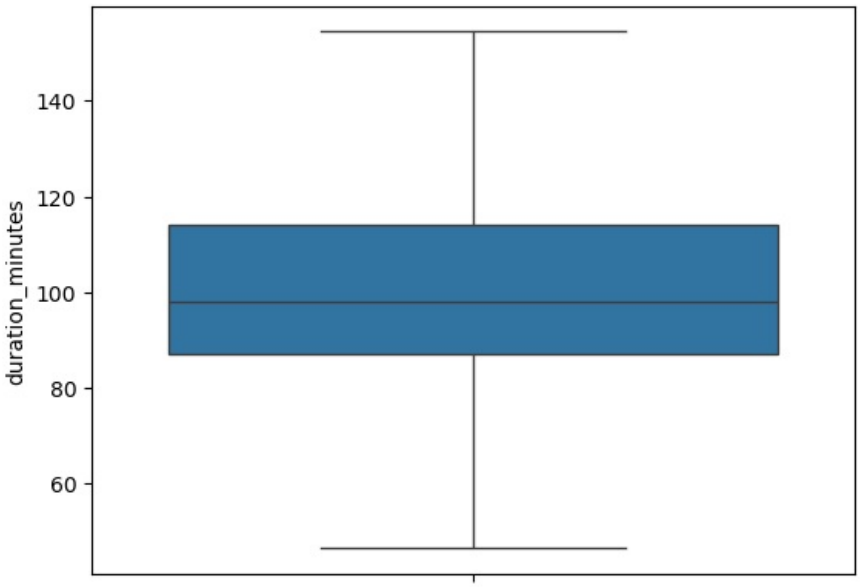
In [17]:
```
sns.boxplot(movie_data['duration_minutes'])
plt.show()
```



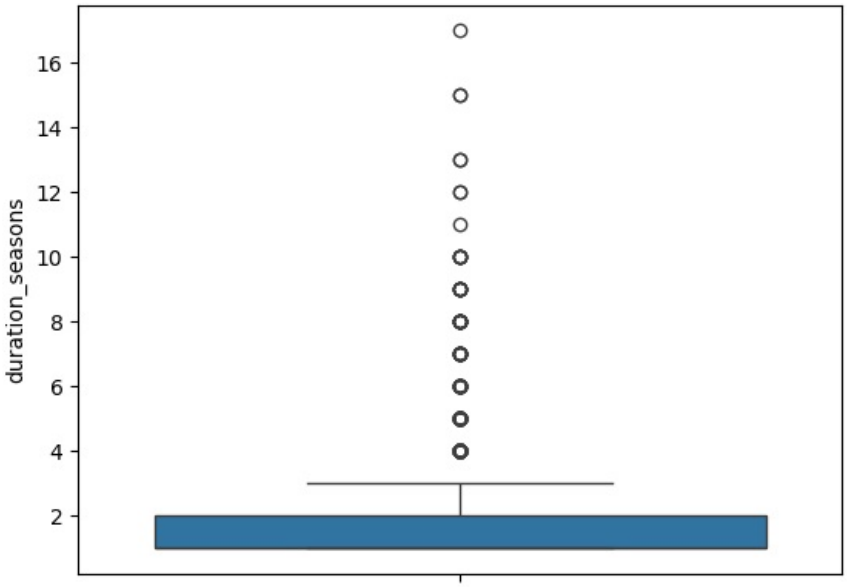
In [18]:
```
Q1 = movie_data['duration_minutes'].quantile(0.25)
Q3 = movie_data['duration_minutes'].quantile(0.75)
IQR = Q3 - Q1

lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

outliers = movie_data[(movie_data['duration_minutes'] < lower_bound) | (movie_data['duration_minutes'] > upper_
print(f"Number of outliers in duration_minutes: {len(outliers)}")

movie_data['duration_minutes'] = np.where(movie_data['duration_minutes'] < lower_bound, lower_bound, movie_data
movie_data['duration_minutes'] = np.where(movie_data['duration_minutes'] > upper_bound, upper_bound, movie_data
```

```
Number of outliers in duration_minutes: 449
```

In [19]:
```
sns.boxplot(movie_data['duration_minutes'])
plt.show()
```



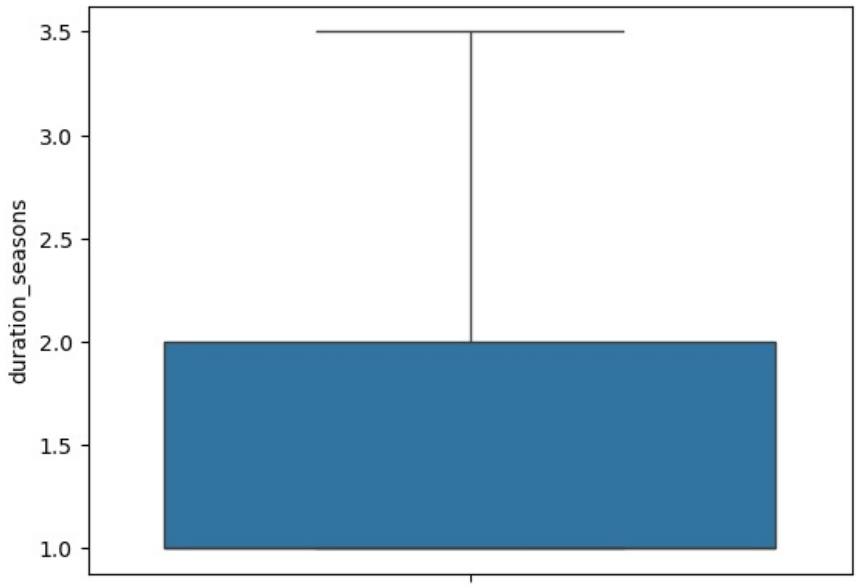
In [20]:
```
sns.boxplot(tv_show_data['duration_seasons'])
plt.show()
```



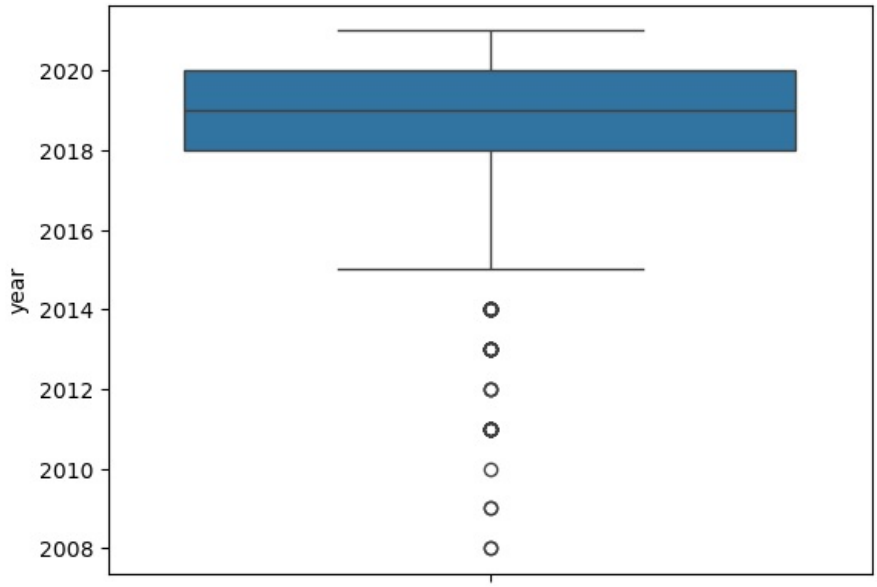
In [21]:
```
Q1 = tv_show_data['duration_seasons'].quantile(0.25)
Q3 = tv_show_data['duration_seasons'].quantile(0.75)
IQR = Q3 - Q1

lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

outliers = tv_show_data[(tv_show_data['duration_seasons'] < lower_bound) | (tv_show_data['duration_seasons'] > 
print(f"Number of outliers in duration_seasons: {len(outliers)}")

tv_show_data['duration_seasons'] = np.where(tv_show_data['duration_seasons'] < lower_bound, lower_bound, tv_sho
tv_show_data['duration_seasons'] = np.where(tv_show_data['duration_seasons'] > upper_bound, upper_bound, tv_sho
```

```
Number of outliers in duration_seasons: 254
```

In [22]:
```
sns.boxplot(tv_show_data['duration_seasons'])
plt.show()
```



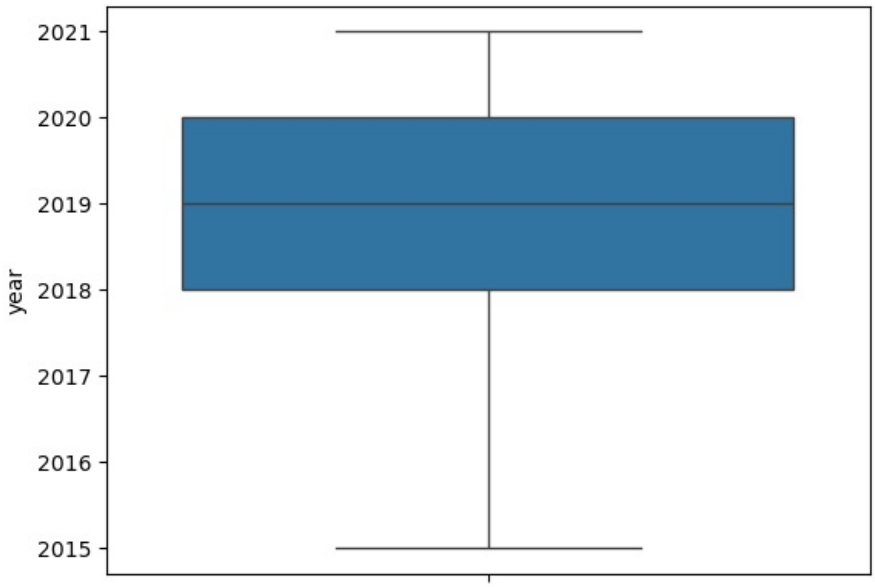
In [23]:
```
sns.boxplot(data["year"])
plt.show()
```



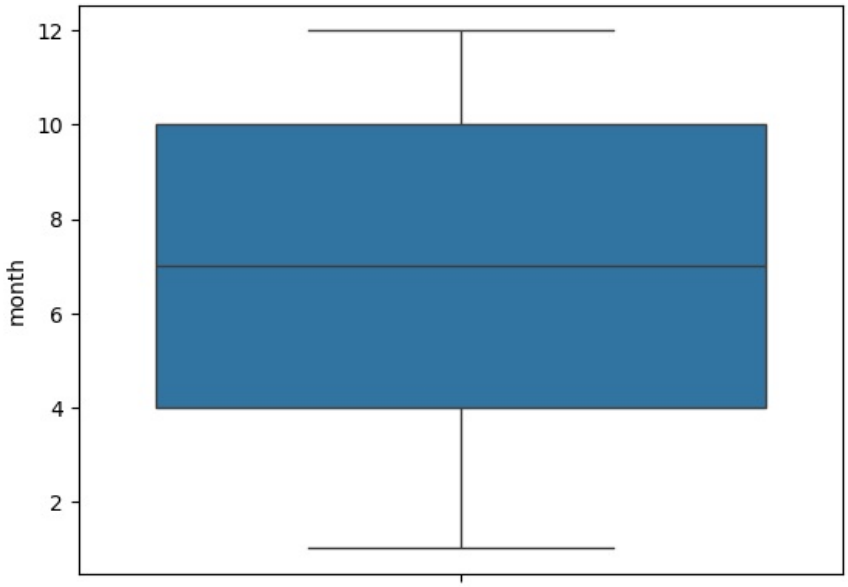
In [24]:
```
Q1 = data["year"].quantile(0.25)
Q3 = data["year"].quantile(0.75)
IQR = Q3 - Q1

lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

outliers = data[(data["year"] < lower_bound) | (data["year"] > upper_bound)]
print(f"Number of outliers in year: {len(outliers)}")

data["year"] = np.where(data["year"] < lower_bound, lower_bound, data["year"])
data["year"] = np.where(data["year"] > upper_bound, upper_bound, data["year"])
```

```
Number of outliers in year: 56
```

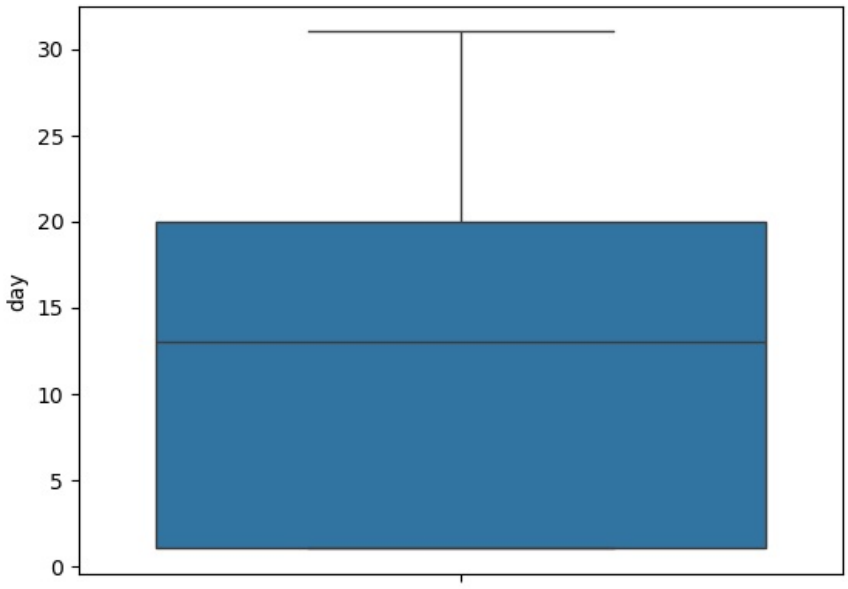
In [25]:
```
sns.boxplot(data["year"])
plt.show()
```



In [26]:
```
sns.boxplot(data["month"])
plt.show()
```



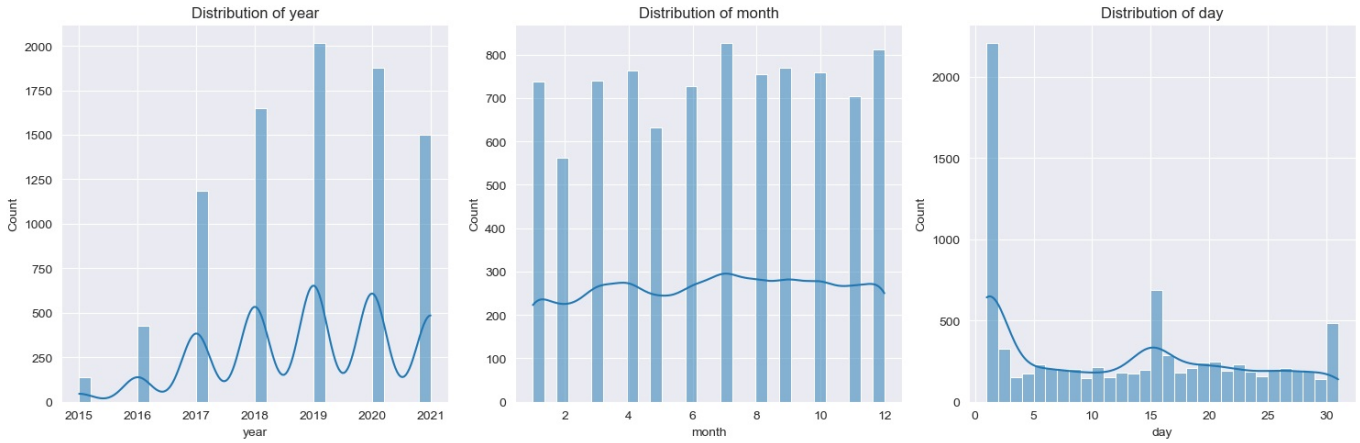
In [27]:
```
sns.boxplot(data["day"])
plt.show()
```



Plotting Distribution plots

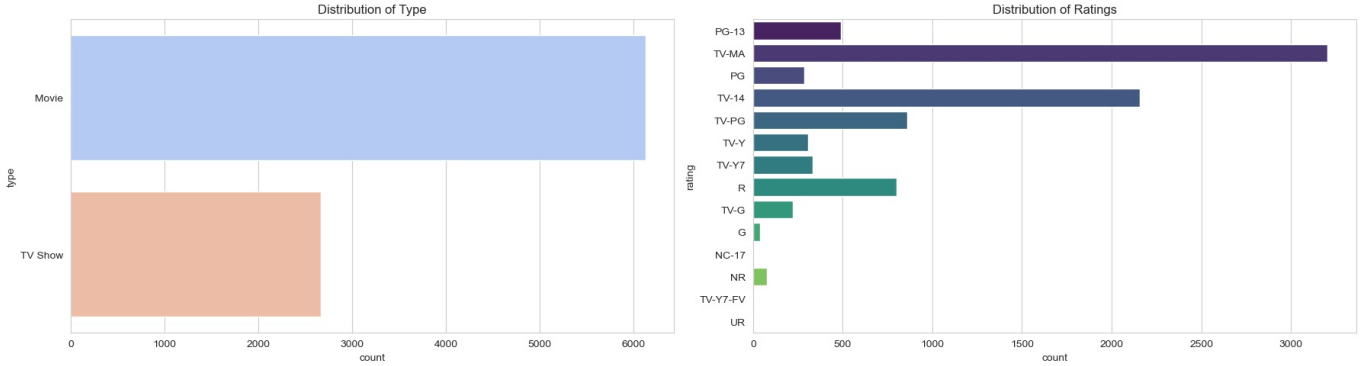
In [28]:
```
# Select numeric columns
numeric_cols = ['year', 'month', 'day']
```

```
sns.set_style("darkgrid")
plt.figure(figsize=(15, 5))
for i, col in enumerate(numeric_cols, 1):
    plt.subplot(1, 3, i)
    sns.histplot(data[col], bins=30, kde=True)
    plt.title(f"Distribution of {col}")
plt.tight_layout()
plt.show()
```



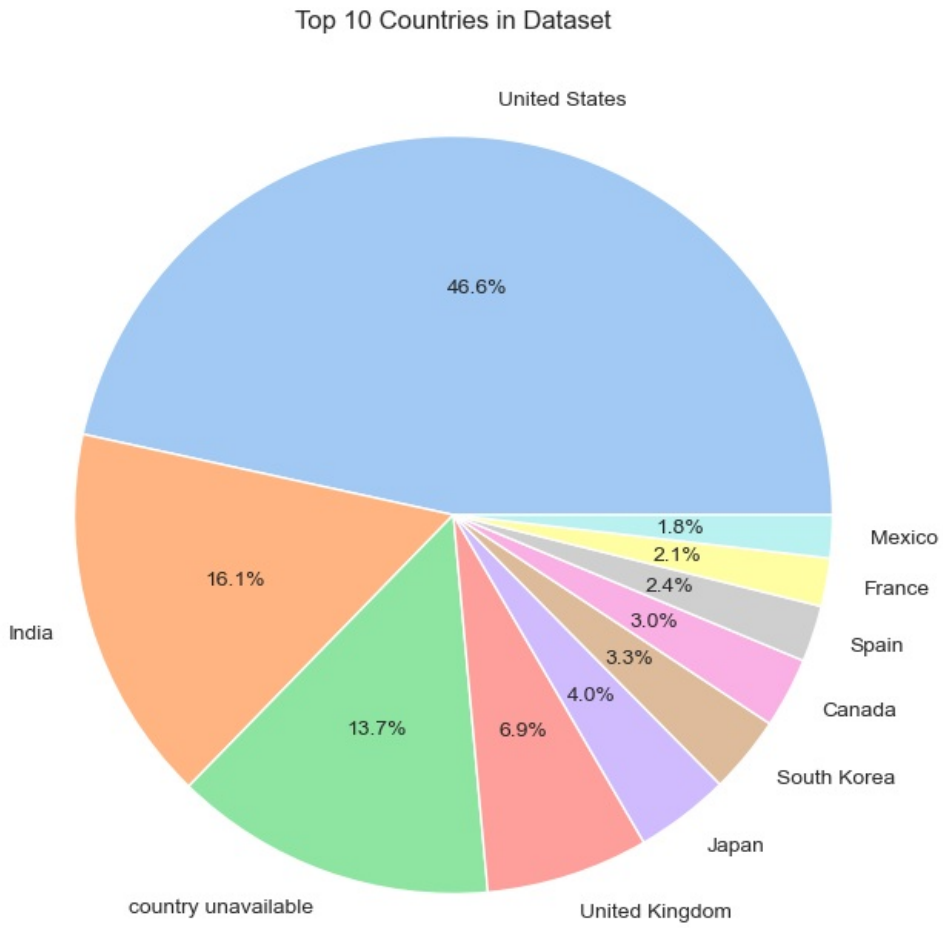
In [29]:

```
# Select categorical columns for visualization
categorical_columns = ['type', 'director', 'cast', 'country', 'rating', 'listed_in']
sns.set_style("whitegrid")
fig, axes = plt.subplots(1, 2, figsize=(18, 5))
sns.countplot(y=data['type'], ax=axes[0], palette="coolwarm").set_title("Distribution of Type")
sns.countplot(y=data['rating'], ax=axes[1], palette="viridis").set_title("Distribution of Ratings")
plt.tight_layout()
plt.show()
```



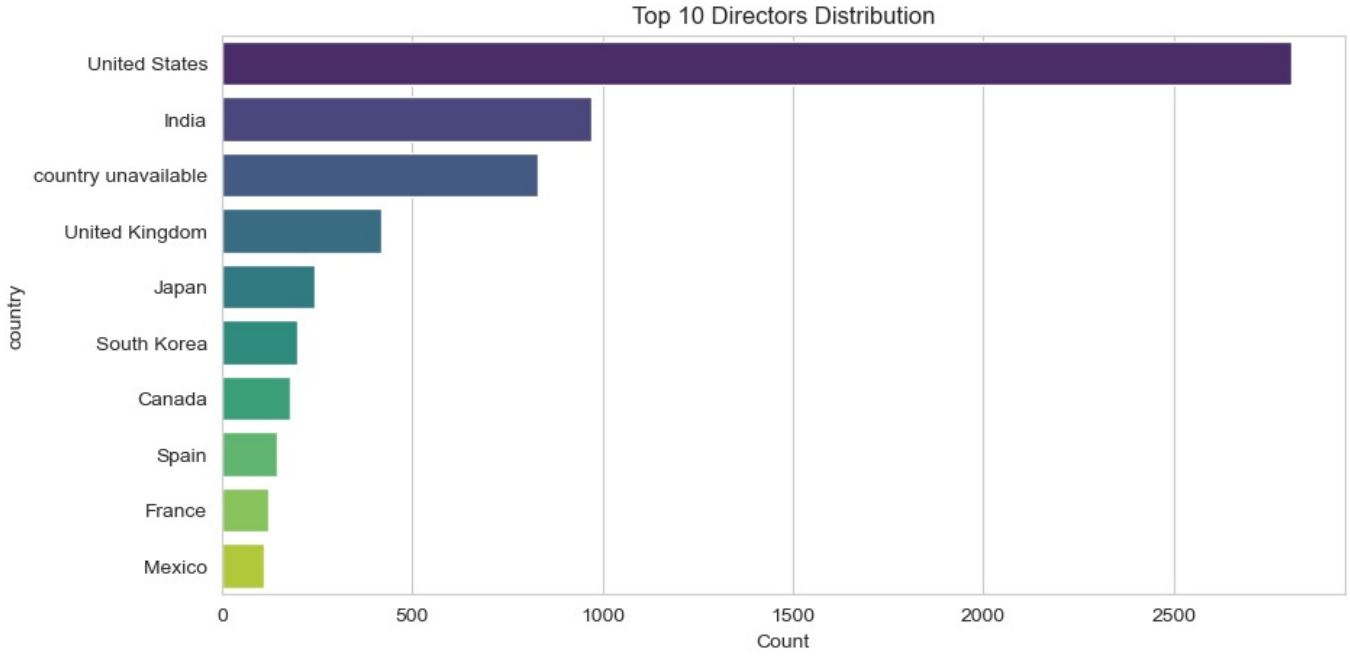
In [30]:

```
# Pie chart for 'country' distribution (top 10 countries for clarity)
top_countries = data['country'].value_counts().head(10)
plt.figure(figsize=(8, 8))
plt.pie(top_countries, labels=top_countries.index, autopct='%1.1f%%', colors=sns.color_palette("pastel"))
plt.title("Top 10 Countries in Dataset")
plt.show()
```



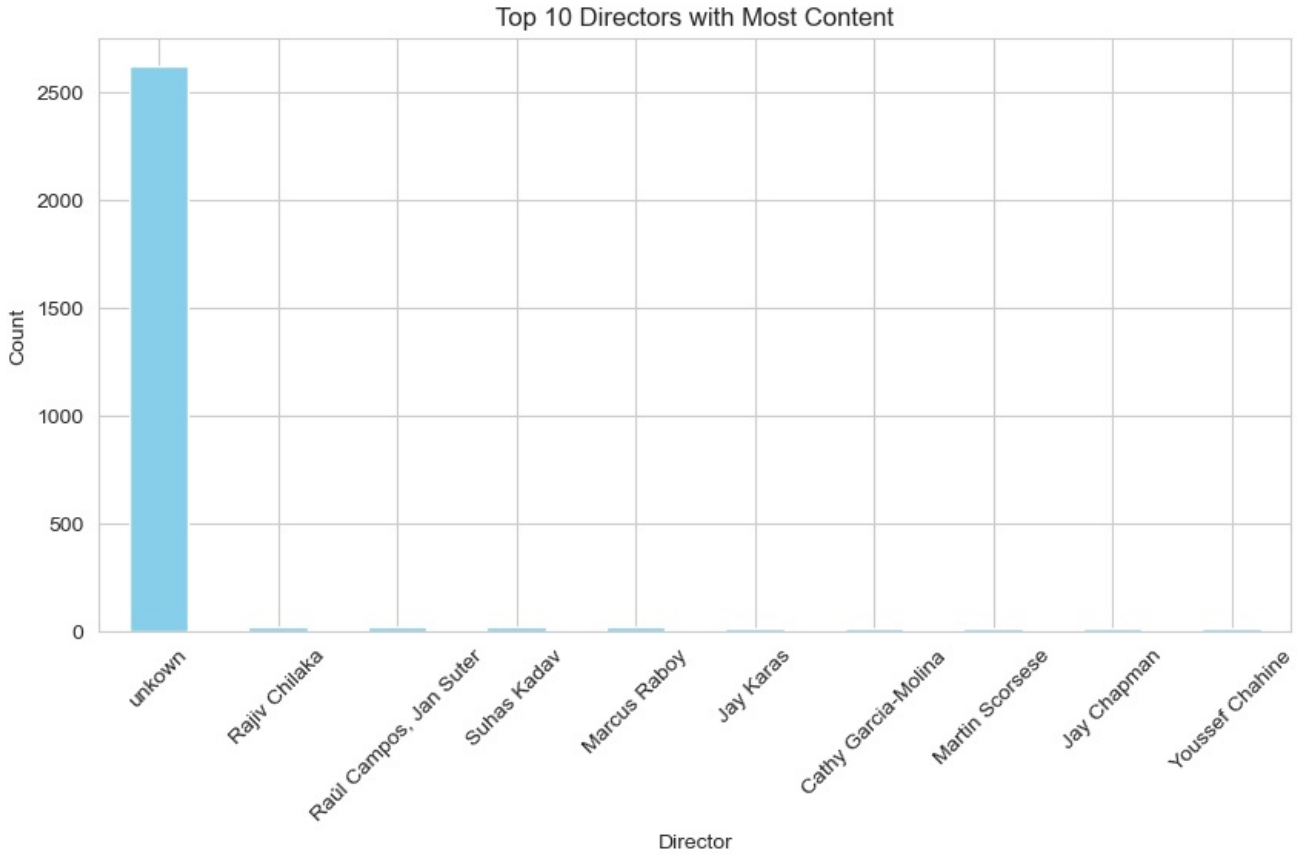
Distribution of Director per Country

In [31]:
```
# Specify the categorical column
categorical_column = "country" # Column name for directors
# Get the top directors
top_n = 10 # Number of top directors to display
top_directors = data[categorical_column].value_counts().nlargest(top_n)
# Check distribution
plt.figure(figsize=(10, 5))
sns.barplot(y=top_directors.index, x=top_directors.values, palette="viridis")
plt.xlabel("Count")
plt.ylabel(categorical_column)
plt.title(f"Top {top_n} Directors Distribution")
plt.show()
```



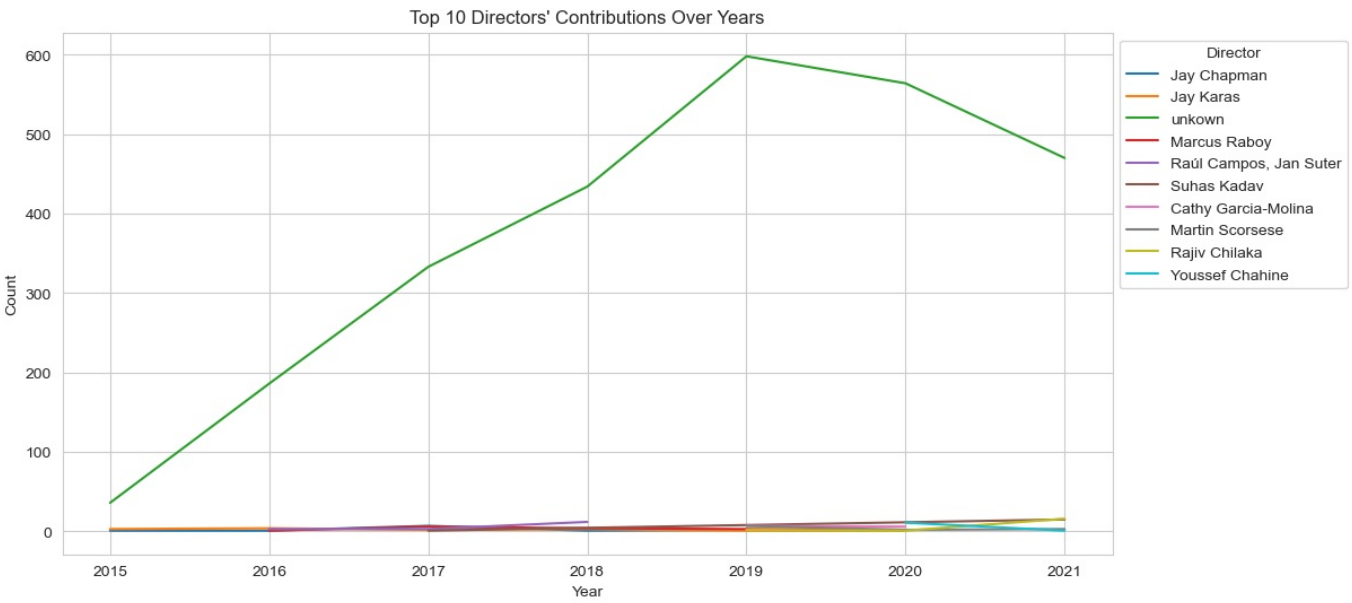
Top 10 Directors with Most Content

In [32]:
```
top_directors = data['director'].value_counts().head(10)
plt.figure(figsize=(10,5))
top_directors.plot(kind='bar', color='skyblue')
plt.title("Top 10 Directors with Most Content")
plt.xlabel("Director")
plt.ylabel("Count")
plt.xticks(rotation=45)
plt.show()
```



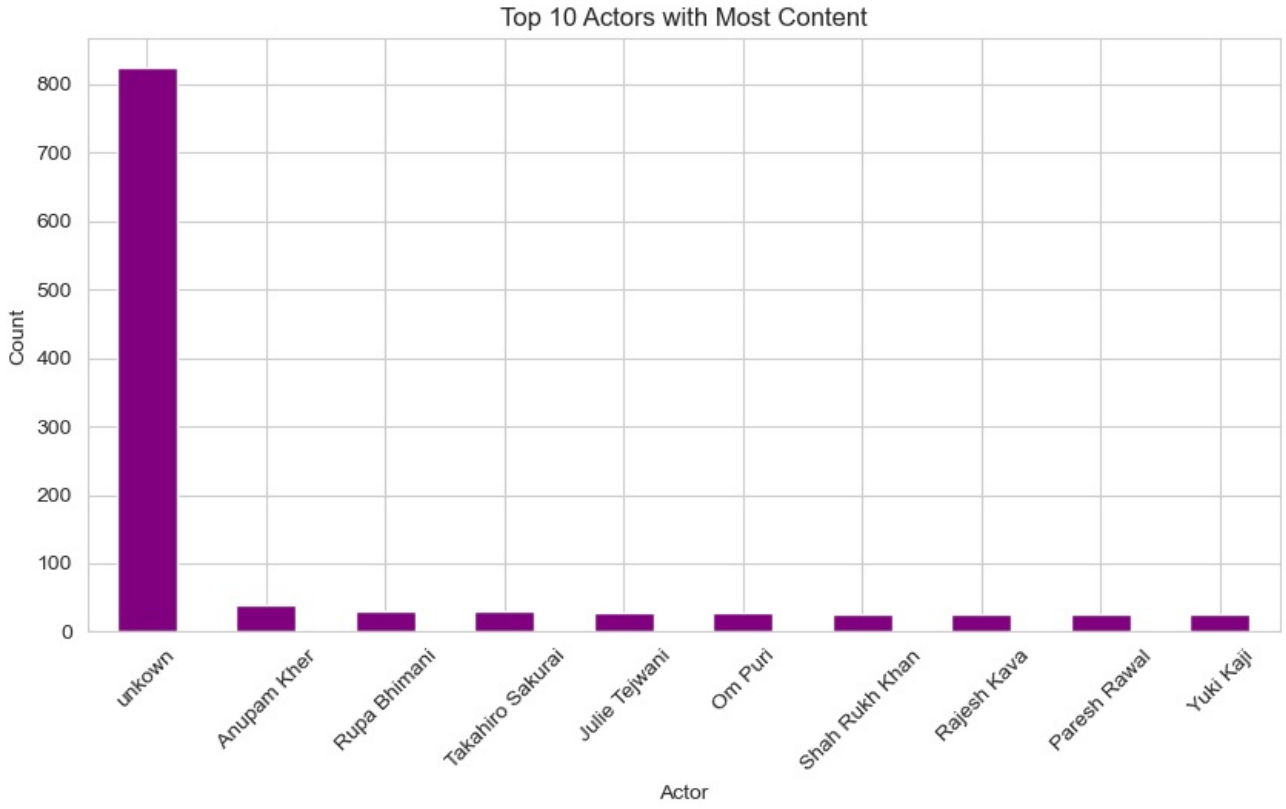
Director Contributions Over Time

In [56]:
```
top_director_list = data['director'].value_counts().head(10).index
df_director_filtered = data[data['director'].isin(top_director_list)]
plt.figure(figsize=(12,6))
sns.lineplot(data=df_director_filtered.groupby(['year', 'director']).size().reset_index(name='count'), x='year'
plt.title("Top 10 Directors' Contributions Over Years")
plt.xlabel("Year")
plt.ylabel("Count")
plt.legend(title="Director", bbox_to_anchor=(1,1))
plt.show()
```



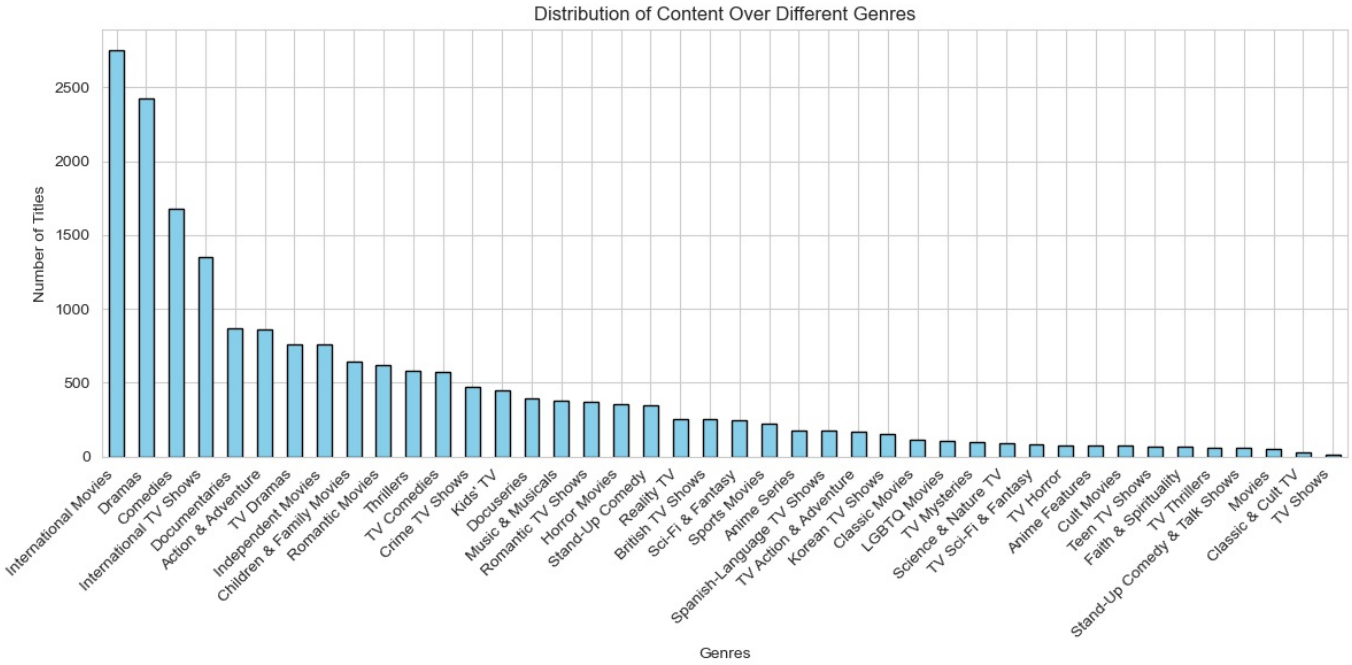
Top 10 Actors with Most Content

In [53]:
```
top_actors = data['cast'].str.split(',').explode().value_counts().head(10)
plt.figure(figsize=(10,5))
top_actors.plot(kind='bar', color='purple')
plt.title("Top 10 Actors with Most Content")
plt.xlabel("Actor")
plt.ylabel("Count")
plt.xticks(rotation=45)
plt.show()
```



visualizations to represent the distribution of content over different genres.

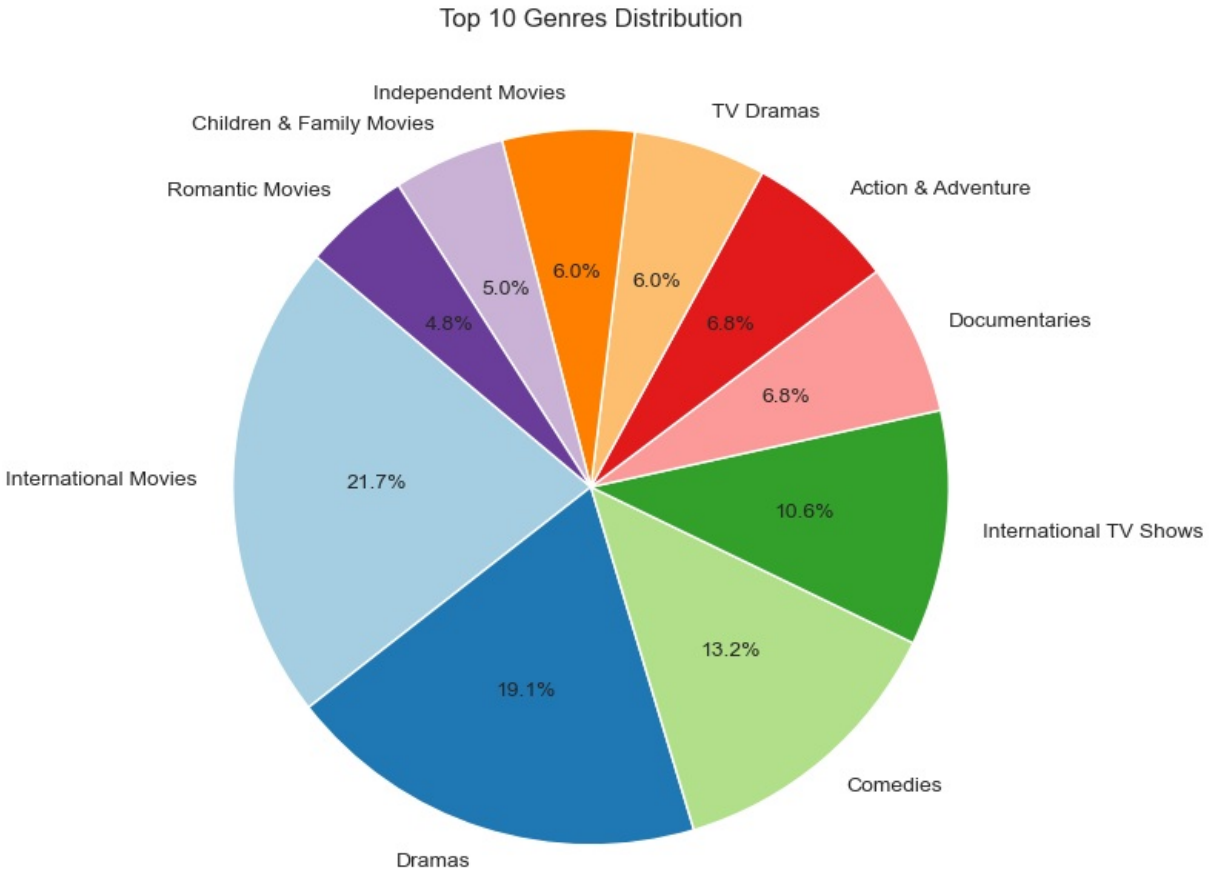
In [33]:
```
 # Split the 'listed_in' column into multiple genres
data['genres'] = data['listed_in'].str.split(', ')

# Flatten the list of genres into a single series for counting
all_genres = data['genres'].explode()

# Count the occurrences of each genre
genre_counts = all_genres.value_counts()

# Plotting the bar chart
plt.figure(figsize=(12, 6))
genre_counts.plot(kind='bar', color='skyblue', edgecolor='black')
plt.title('Distribution of Content Over Different Genres')
plt.xlabel('Genres')
plt.ylabel('Number of Titles')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```



This bar graph shows that international Movies has largest content than the other geners in second position dramas and so on

Top 10 Geners distribution over content

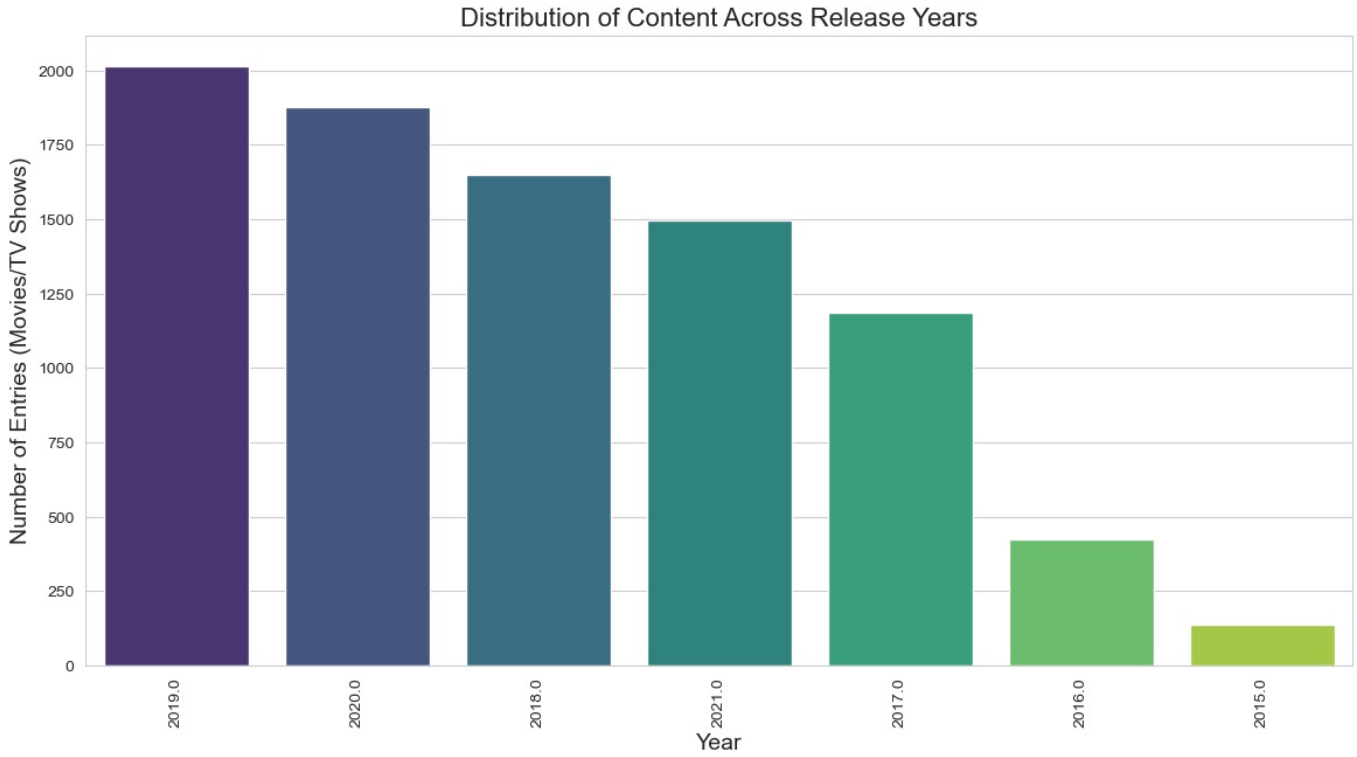
In [34]:
```
top_genres = genre_counts.head(10)
plt.figure(figsize=(8, 8))
top_genres.plot(kind='pie', autopct='%1.1f%%', startangle=140, colors=plt.cm.Paired.colors)
plt.title('Top 10 Genres Distribution')
plt.ylabel('')
plt.tight_layout()
plt.show()
```



Visualize the distribution of content across release years.

In [35]:
```
plt.figure(figsize=(14, 7))
sns.countplot(data=data, x='year', palette='viridis', order=data['year'].value_counts().index)
```

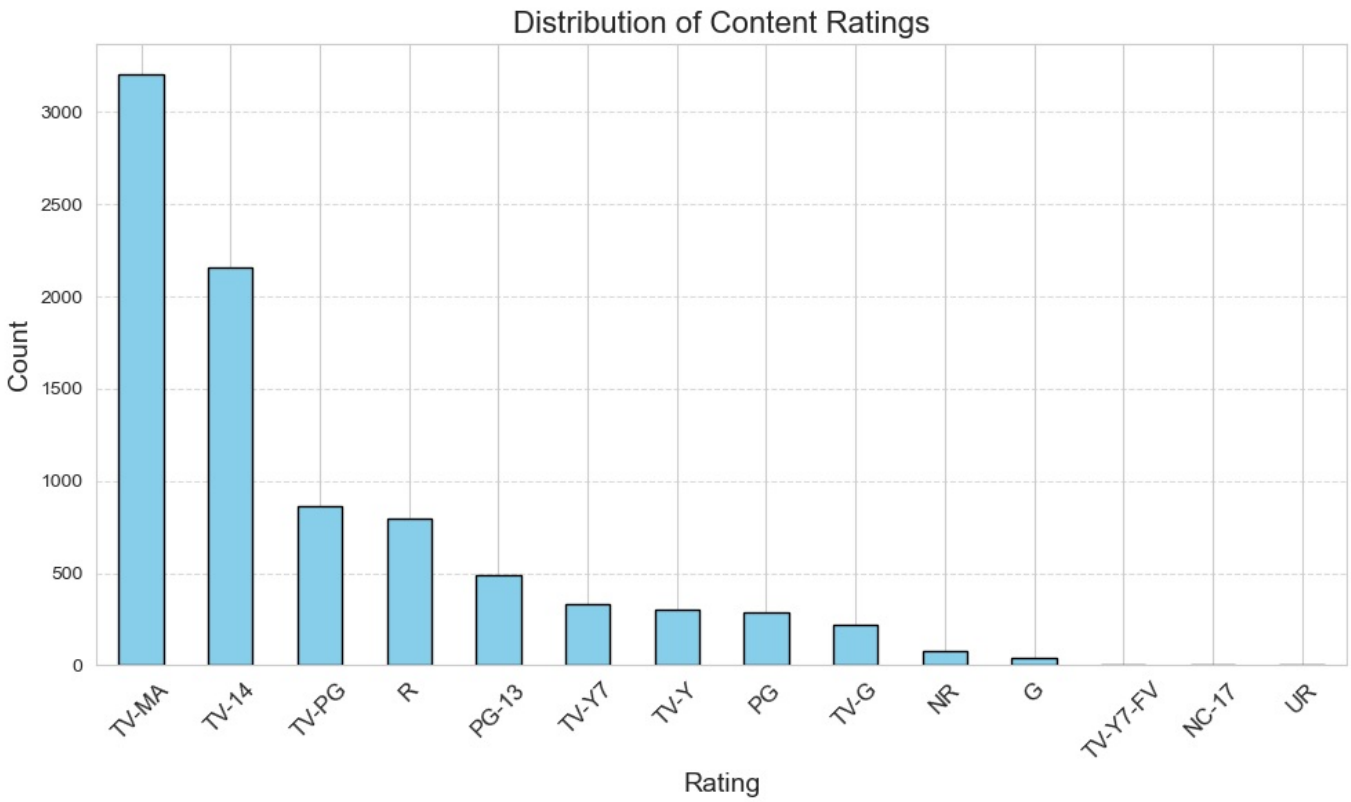
```
plt.title('Distribution of Content Across Release Years', fontsize=16)
plt.xlabel('Year', fontsize=14)
plt.ylabel('Number of Entries (Movies/TV Shows)', fontsize=14)
plt.xticks(rotation=90)
plt.show()
```



This shows that content Across release years are increases upto 2019 and then it start decresing

Analyze the distribution of content ratings.

In [36]:
```
# Analyze the distribution of content ratings
rating_distribution = data['rating'].value_counts()

plt.figure(figsize=(10, 6))
rating_distribution.plot(kind='bar', color='skyblue', edgecolor='black')
plt.title('Distribution of Content Ratings', fontsize=16)
plt.xlabel('Rating', fontsize=14)
plt.ylabel('Count', fontsize=14)
plt.xticks(rotation=45, fontsize=12)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()
```

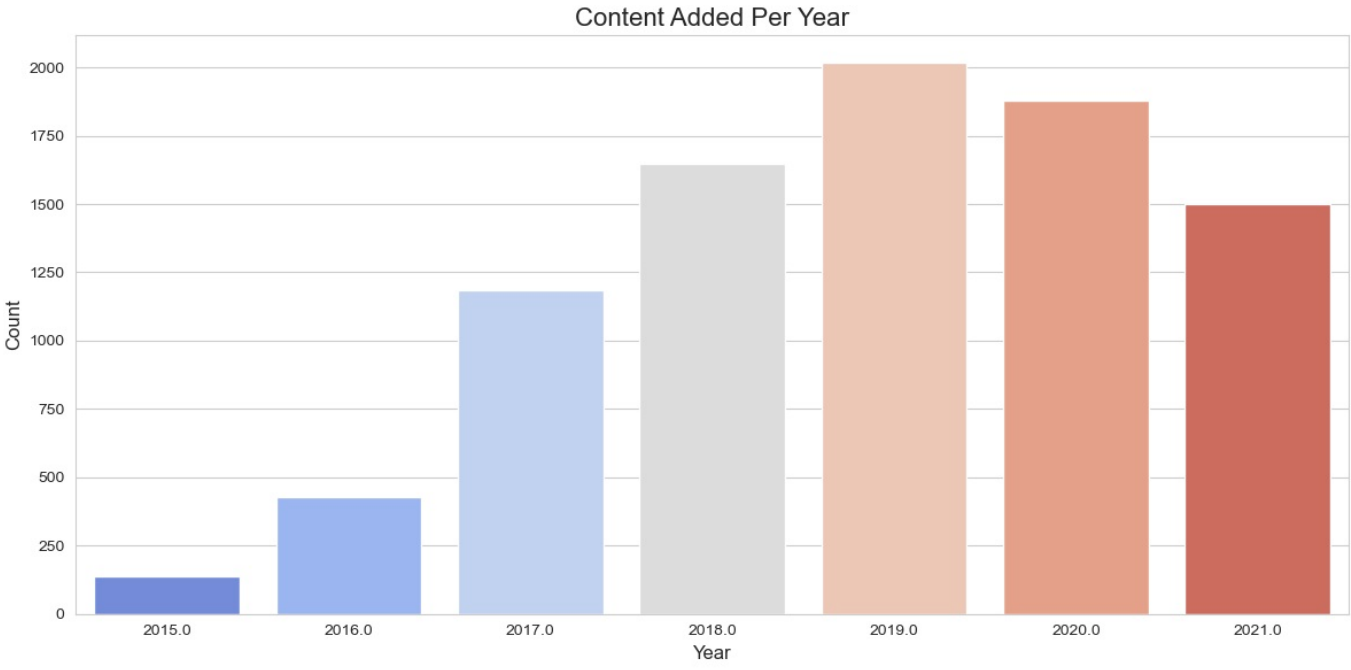


It shows that TV-MA has highest content rating

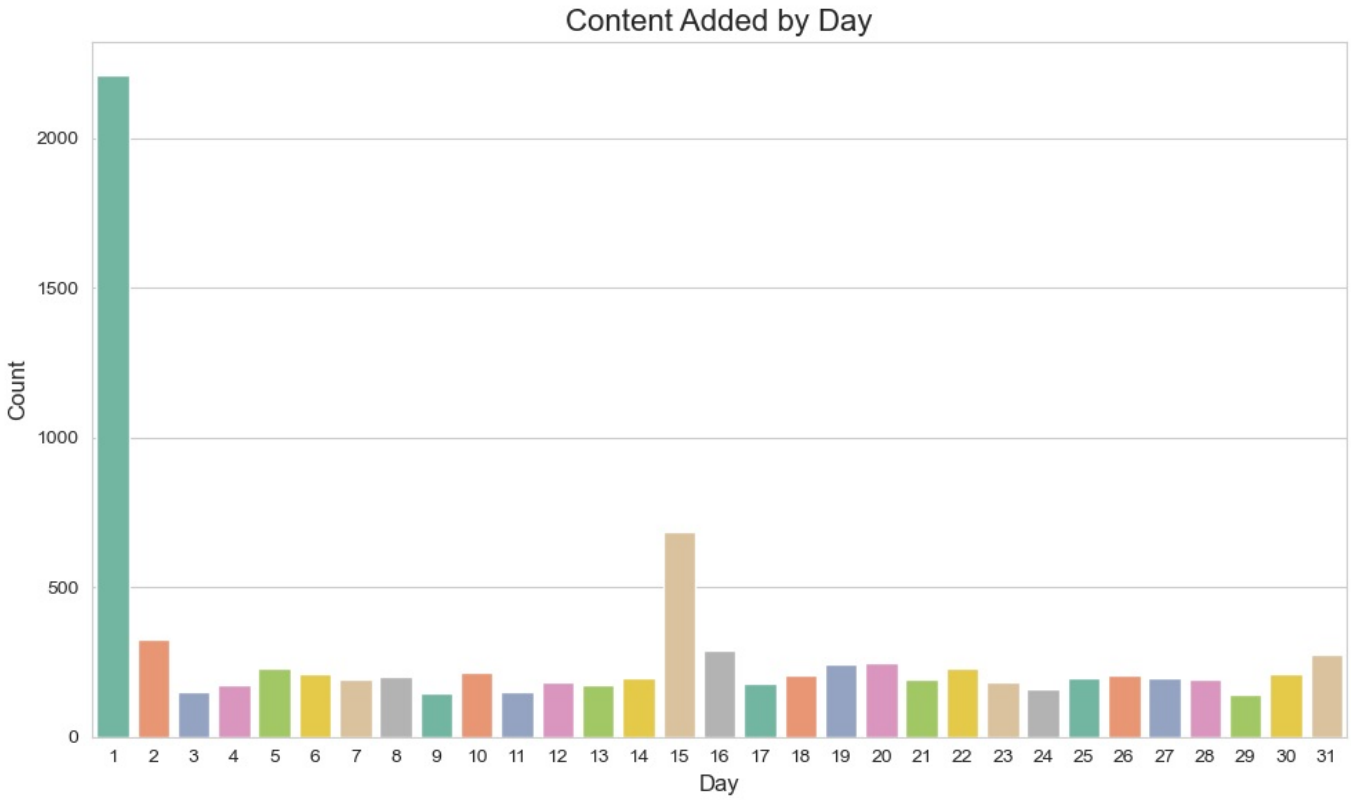
Time Series Analysis:

If there's a temporal component, perform time series analysis to identify trends and patterns over time.

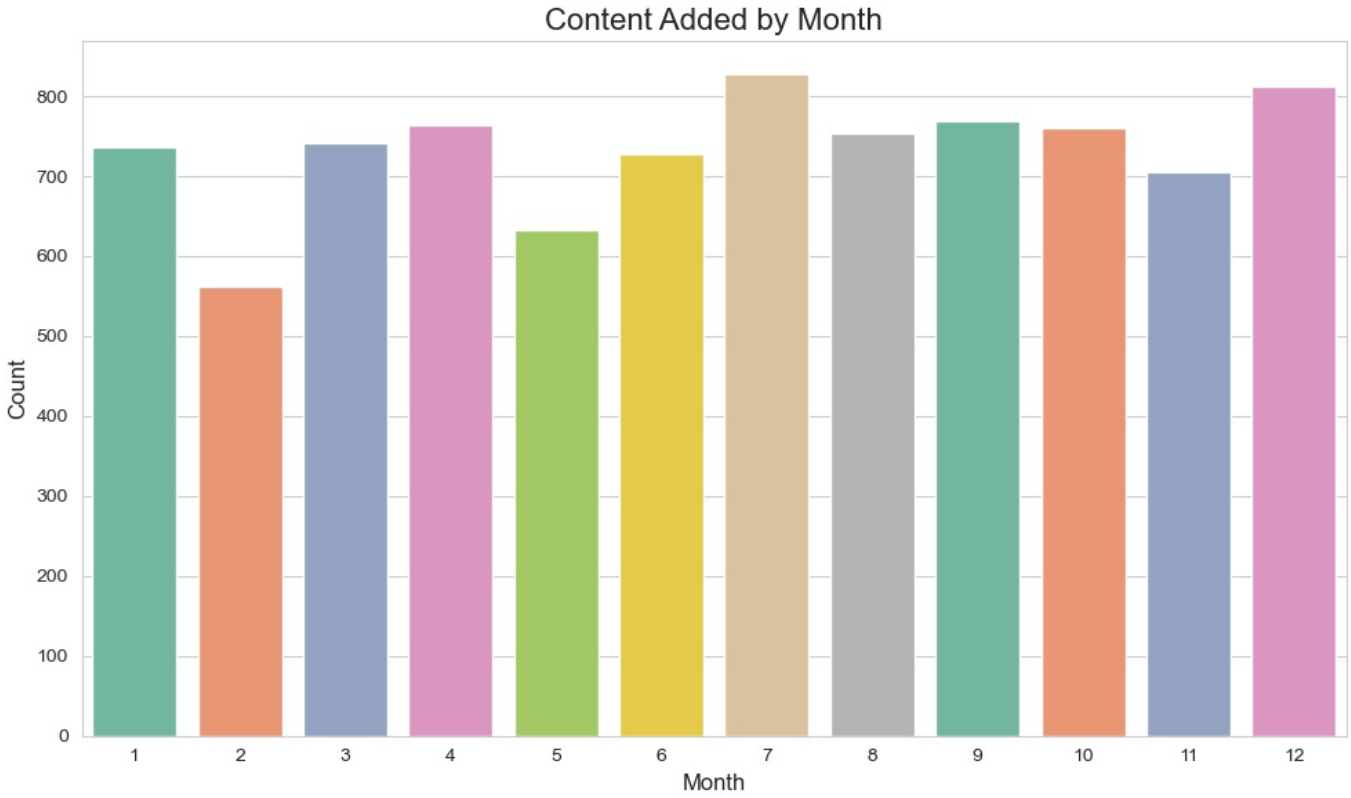
In [37]:
```
plt.figure(figsize=(12, 6))
sns.countplot(data=data, x='year', palette='coolwarm', order=sorted(data['year'].unique()))
plt.title('Content Added Per Year', fontsize=16)
plt.xlabel('Year', fontsize=12)
plt.ylabel('Count', fontsize=12)
plt.tight_layout()
plt.show()
```



In [38]:
```
plt.figure(figsize=(10, 6))
sns.countplot(data=data, x='day', palette='Set2')
plt.title('Content Added by Day', fontsize=16)
plt.xlabel('Day', fontsize=12)
plt.ylabel('Count', fontsize=12)
plt.tight_layout()
plt.show()
```



In [39]:
```
plt.figure(figsize=(10, 6))
sns.countplot(data=data, x='month', palette='Set2')
plt.title('Content Added by Month', fontsize=16)
plt.xlabel('Month', fontsize=12)
plt.ylabel('Count', fontsize=12)
plt.tight_layout()
plt.show()
```

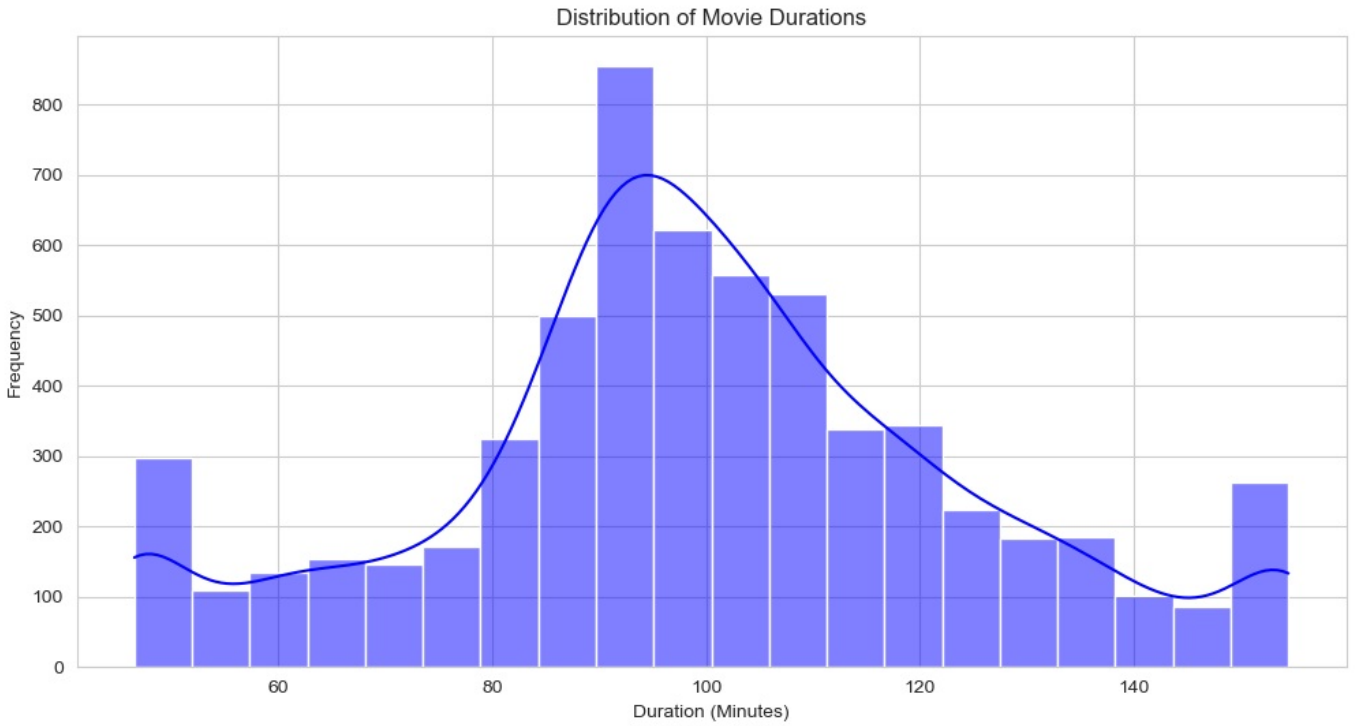


In 2019 year content added are highest and on july month and 1 day of month has highest content added

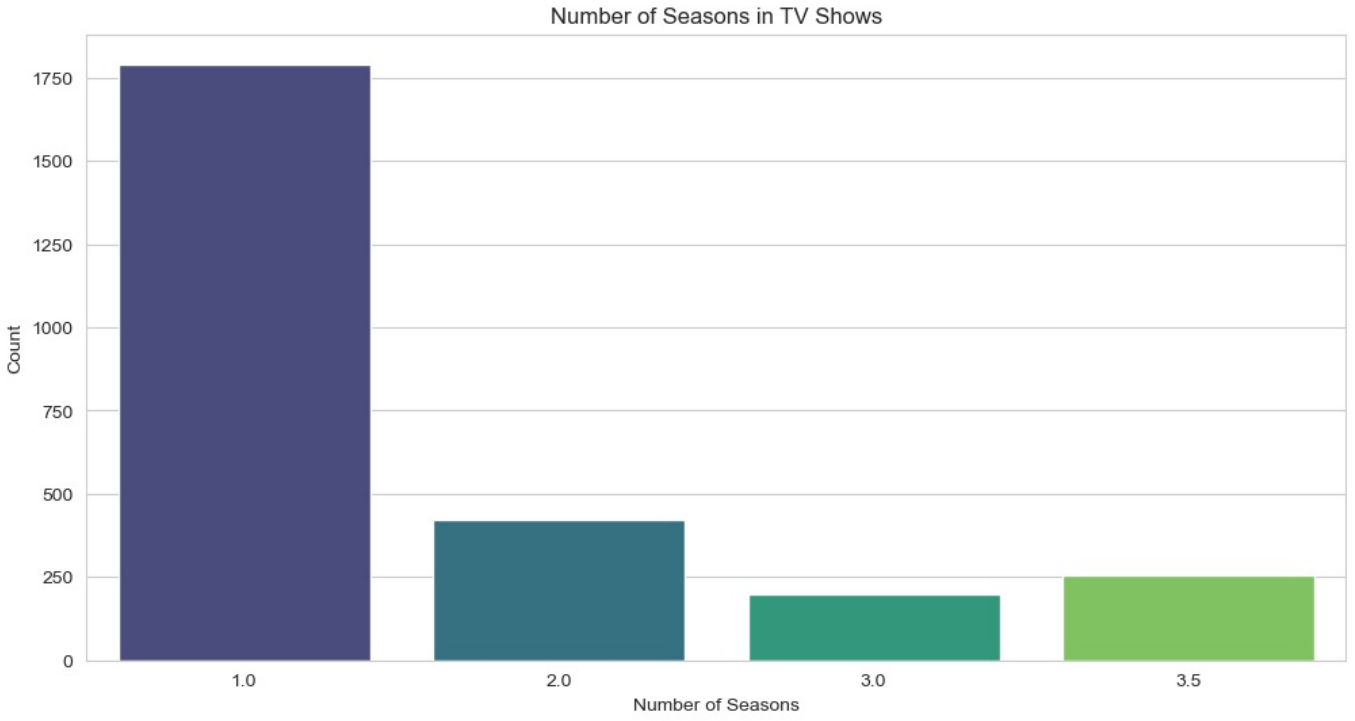
Explore the length of movies or episodes and identify any trends.

In [40]:
```
plt.figure(figsize=(12, 6))
sns.histplot(movie_data['duration_minutes'], bins=20, kde=True, color='blue')
plt.title('Distribution of Movie Durations')
plt.xlabel('Duration (Minutes)')
plt.ylabel('Frequency')
```

```
plt.show()
```



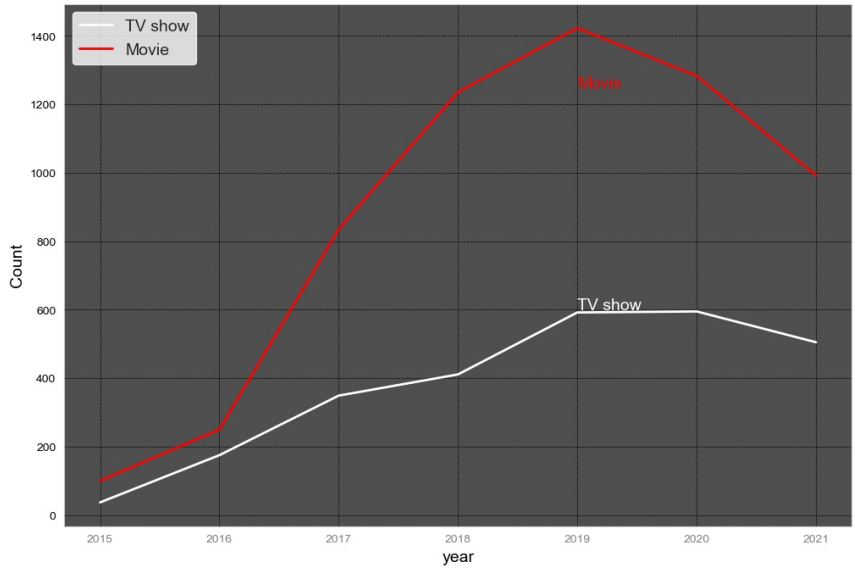
In [41]:
```
plt.figure(figsize=(12, 6))
sns.countplot(x='duration_seasons', data=tv_show_data, palette='viridis')
plt.title('Number of Seasons in TV Shows')
plt.xlabel('Number of Seasons')
plt.ylabel('Count')
plt.show()
```



There are larger duration of movies but maxiumum movies are in 90 mins long and tv shows are maximum of 1 season

In [42]:
```
type_data = data[['type', 'year']]

tv_shows = type_data[type_data['type'] == 'TV Show'].groupby('year')['type'].count()
movies = type_data[type_data['type'] == 'Movie'].groupby('year')['type'].count()

plt.figure(figsize=(12, 8))  # Set the figure size

plt.plot(tv_shows.index, tv_shows.values, label='TV show', color='white', linewidth=2, markersize=6)


plt.plot(movies.index, movies.values, label='Movie', color='red', linewidth=2, markersize=6)


plt.title('Movie vs TV Show, Trend Analysis per year_added', fontsize=20, color='white')
```

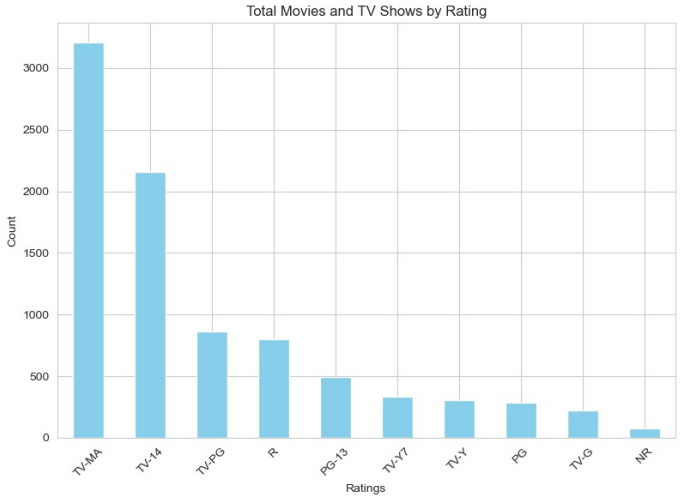
```
plt.xlabel('year', fontsize=14, color='black')
plt.ylabel('Count', fontsize=14, color='black')

plt.legend(fontsize=14,  loc='upper left',facecolor='white', edgecolor='white')
plt.grid(color='black', linestyle='--', linewidth=0.5)
plt.gca().set_facecolor('#4f4f4f')  # Set background to black
plt.xticks(fontsize=10, color='grey')
plt.yticks(fontsize=10, color='black')


plt.text(2019, 1250, 'Movie', fontsize=14, color='red')
plt.text(2019, 600, 'TV show', fontsize=14, color='white')
plt.text(2010, 1350, 'Movie VS TV show analysis trend per year added', fontsize=20, color='white')

plt.show()
```

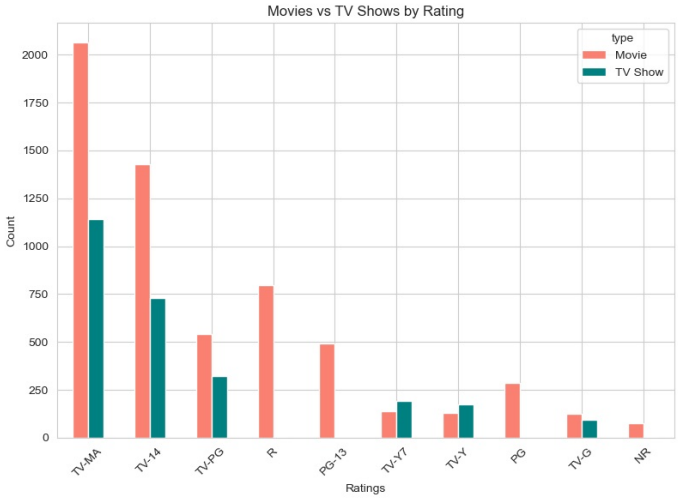


Top Lists and Recommendations

Identify and present top-rated movies or TV shows based on user ratings.

In [43]:
```
# Group data by 'type' and 'rating' for analysis
rating_type_counts = data.groupby(['rating', 'type']).size().unstack(fill_value=0)

# Total counts by rating for Movies and TV Shows
movie_tv_counts = rating_type_counts.sum(axis=1).sort_values(ascending=False).head(10)
top_ratings = movie_tv_counts.index

# Filter for top ratings to focus on the most popular ones
filtered_rating_type_counts = rating_type_counts.loc[top_ratings]

# Plotting
fig, axes = plt.subplots(1, 2, figsize=(16, 6))

# Total Movies and TV Shows by Rating
movie_tv_counts.plot(kind="bar", ax=axes[0], color="skyblue")
axes[0].set_title("Total Movies and TV Shows by Rating")
axes[0].set_xlabel("Ratings")
axes[0].set_ylabel("Count")
axes[0].tick_params(axis='x', rotation=45)

# Side-by-Side Comparison
filtered_rating_type_counts.plot(kind="bar", ax=axes[1], stacked=False, color=["salmon", "teal"])
axes[1].set_title("Movies vs TV Shows by Rating")
axes[1].set_xlabel("Ratings")
axes[1].set_ylabel("Count")
axes[1].tick_params(axis='x', rotation=45)
plt.tight_layout()
plt.show()
```



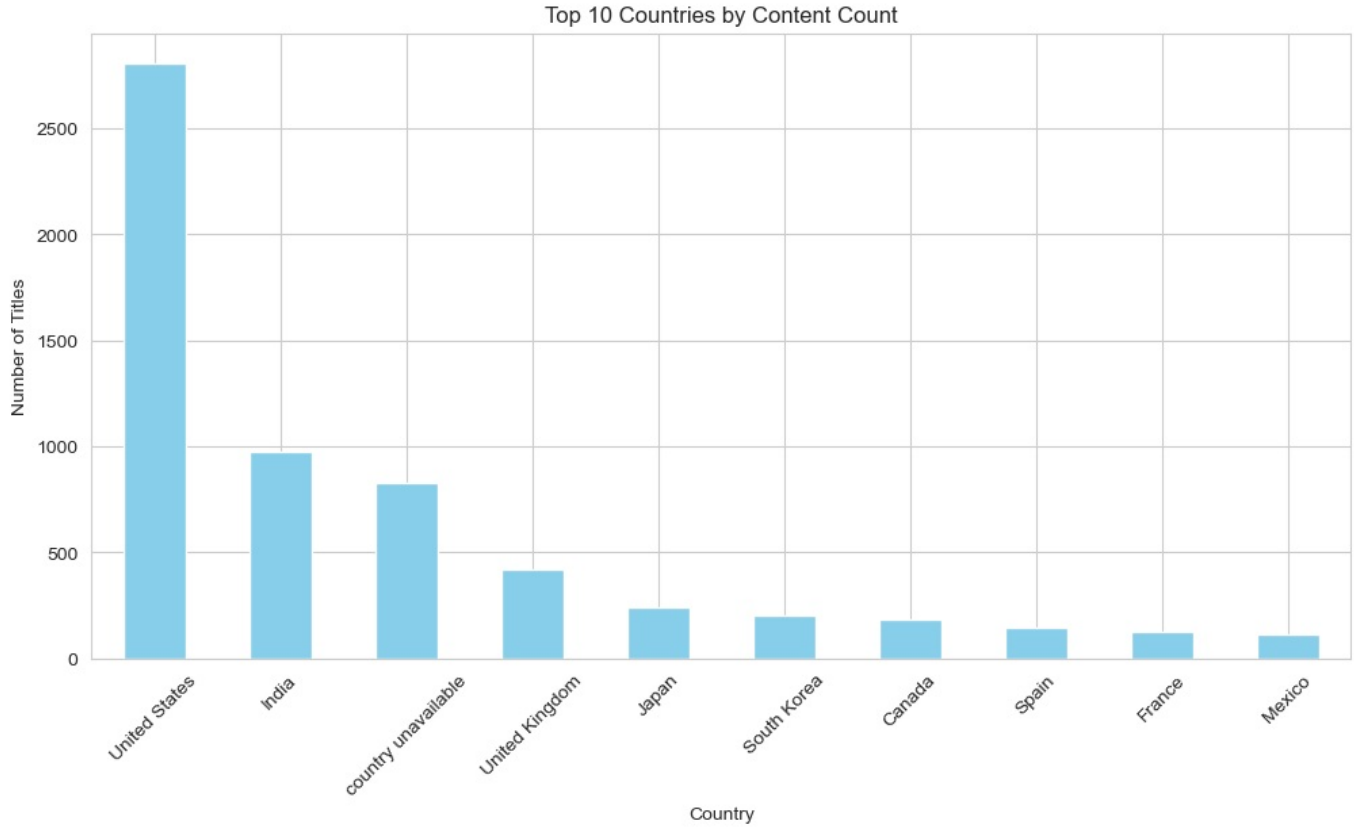


it shows that TV-MA rating are highest and movies are more rated then TV shows. It shows big difference by users rating and also shows that users prefer movies over TV shows.

Geographical Analysis

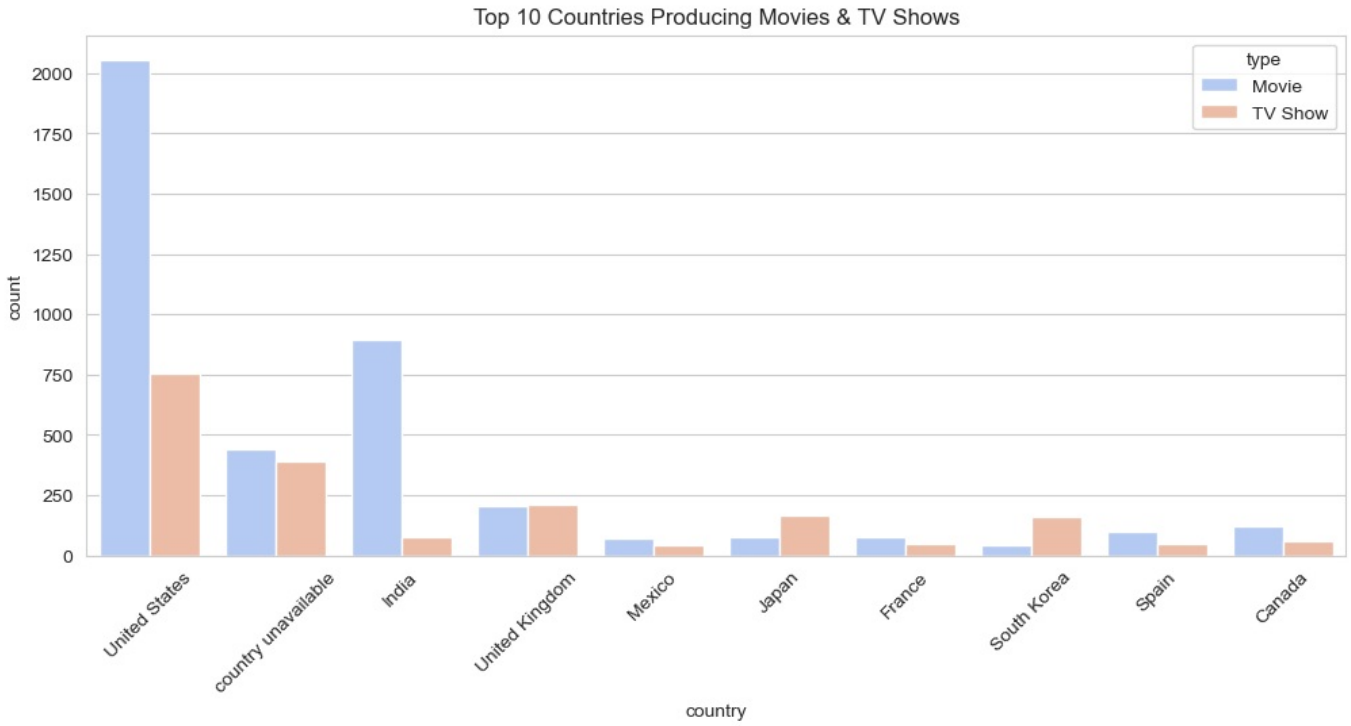
Explore the geographical distribution of content

In [44]:
```
# Count the number of entries per country
country_distribution = data['country'].value_counts()
plt.figure(figsize=(12, 6))
country_distribution.head(10).plot(kind='bar', color='skyblue')
plt.title("Top 10 Countries by Content Count")
plt.xlabel("Country")
plt.ylabel("Number of Titles")
plt.xticks(rotation=45)
plt.show()
```



It shows that united states has biggest content than other countries

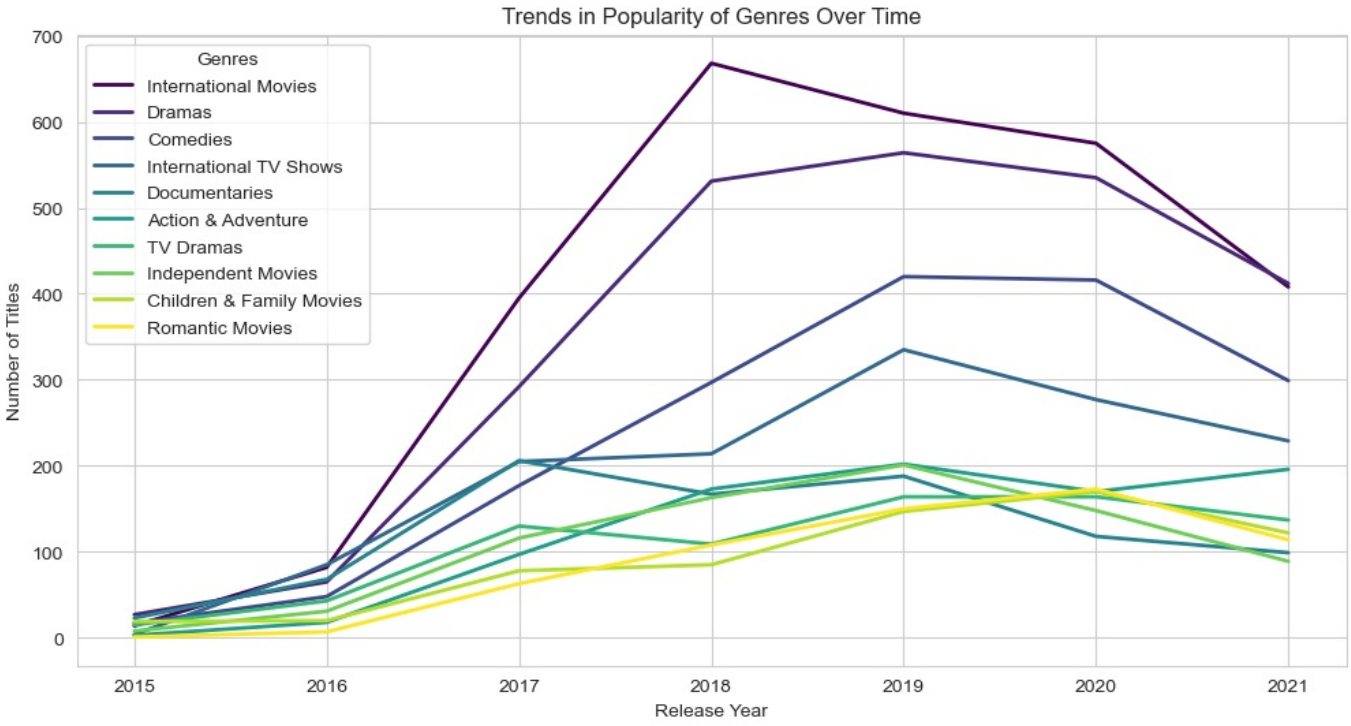
In [45]:
```
plt.figure(figsize=(12, 5))
top_countries = data['country'].value_counts().head(10).index
sns.countplot(data=data[data['country'].isin(top_countries)], x='country', hue='type', palette="coolwarm")
plt.title("Top 10 Countries Producing Movies & TV Shows")
plt.xticks(rotation=45)
plt.show()
```



countries which produces most movies and shows are unitesd states

In [46]:
```
plt.figure(figsize=(12, 6))
genre_over_time = data.explode("genres").groupby(["year", "genres"]).size().unstack().fillna(0)
top_genres = genre_over_time.sum().sort_values(ascending=False).head(10).index # Top 10 genres
genre_over_time[top_genres].plot(figsize=(12, 6), colormap="viridis", linewidth=2)
plt.xlabel("Release Year")
plt.ylabel("Number of Titles")
plt.title("Trends in Popularity of Genres Over Time")
plt.legend(title="Genres")
plt.show()
```

```
<Figure size 1200x600 with 0 Axes>
```

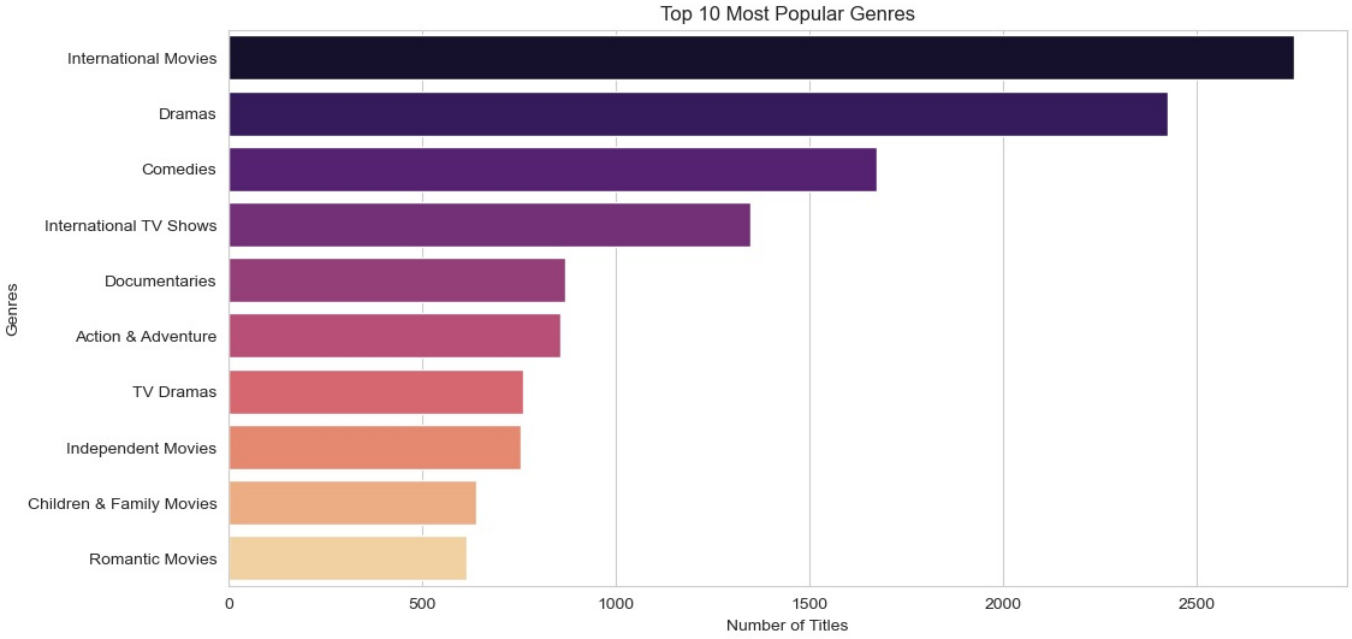


User Preferences:

Investigate whether certain genres or types of content are more popular among users.

In [47]:
```
if "listed_in" in data.columns:
  genre_list = [genre.strip() for sublist in data["listed_in"].dropna().str.split(",") for genre in sublist]
  genre_counts = pd.Series(genre_list).value_counts().head(10) # Top 10 genres
  plt.figure(figsize=(12, 6))
  sns.barplot(x=genre_counts.values, y=genre_counts.index, palette="magma")
  plt.xlabel("Number of Titles")
```

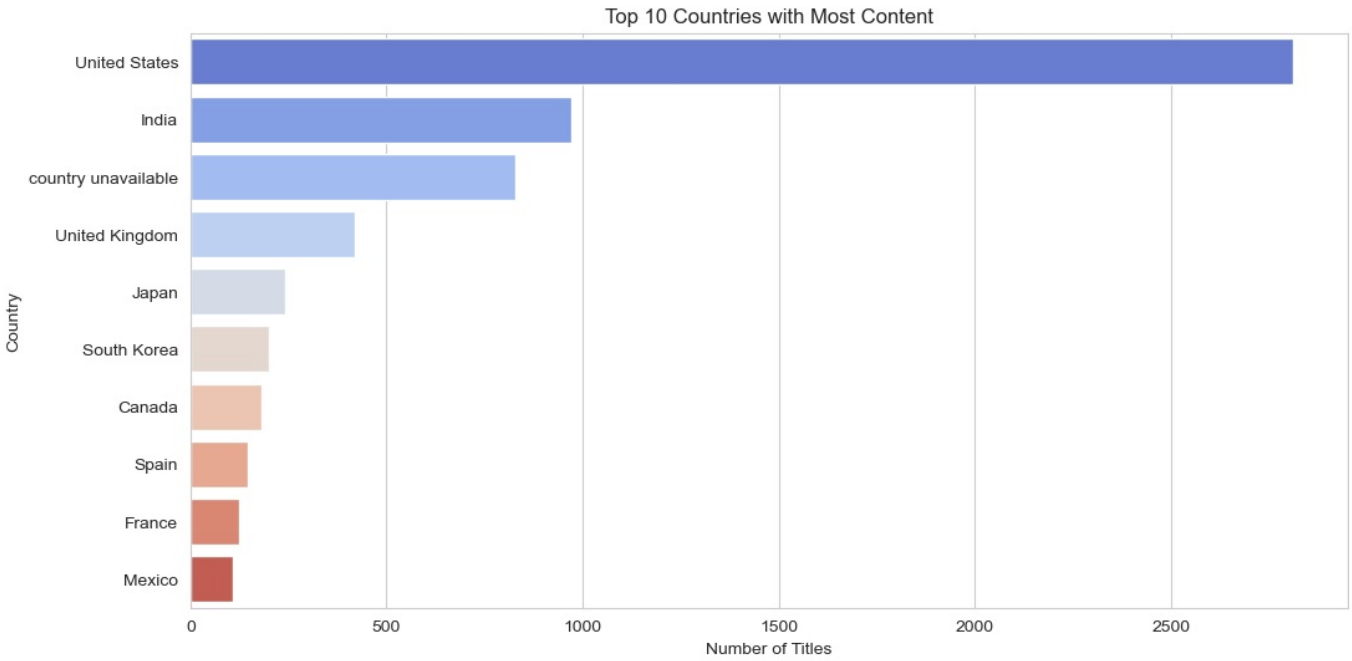
```
  plt.ylabel("Genres")
  plt.title("Top 10 Most Popular Genres")
  plt.show()
else:
  print("Column 'Listed_IN' not found in the dataset.")
```



Language Analysis:

If applicable, analyze the distribution of content in different languages.

In [48]:
```
if "country" in data.columns:
  plt.figure(figsize=(12, 6))
  country_counts = data["country"].value_counts().head(10) # Top 10 countries
  sns.barplot(x=country_counts.values, y=country_counts.index, palette="coolwarm")
  plt.xlabel("Number of Titles")
  plt.ylabel("Country")
  plt.title("Top 10 Countries with Most Content")
  plt.show()
else:
  print("Column 'Country' not found in the dataset.")
```

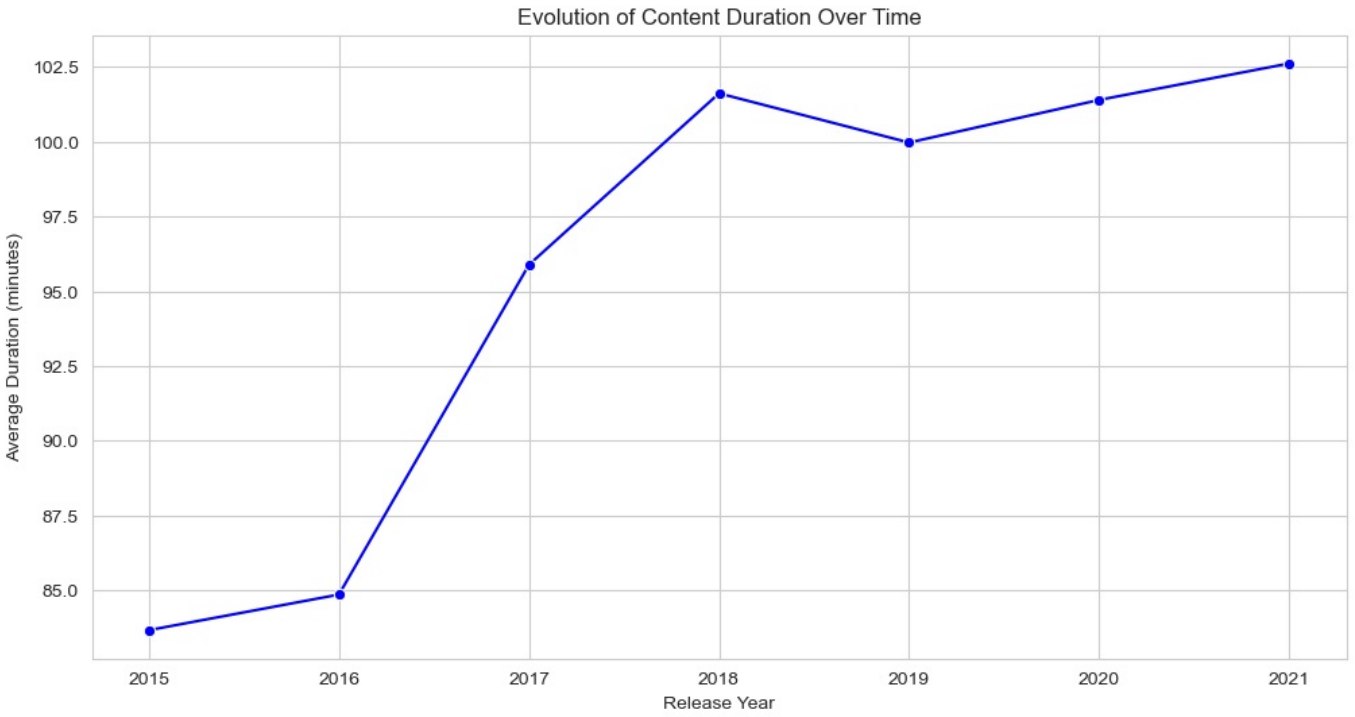


Content Evolution Over Time:

Explore how the characteristics of content (e.g., duration, ratings) have evolved over theyears.

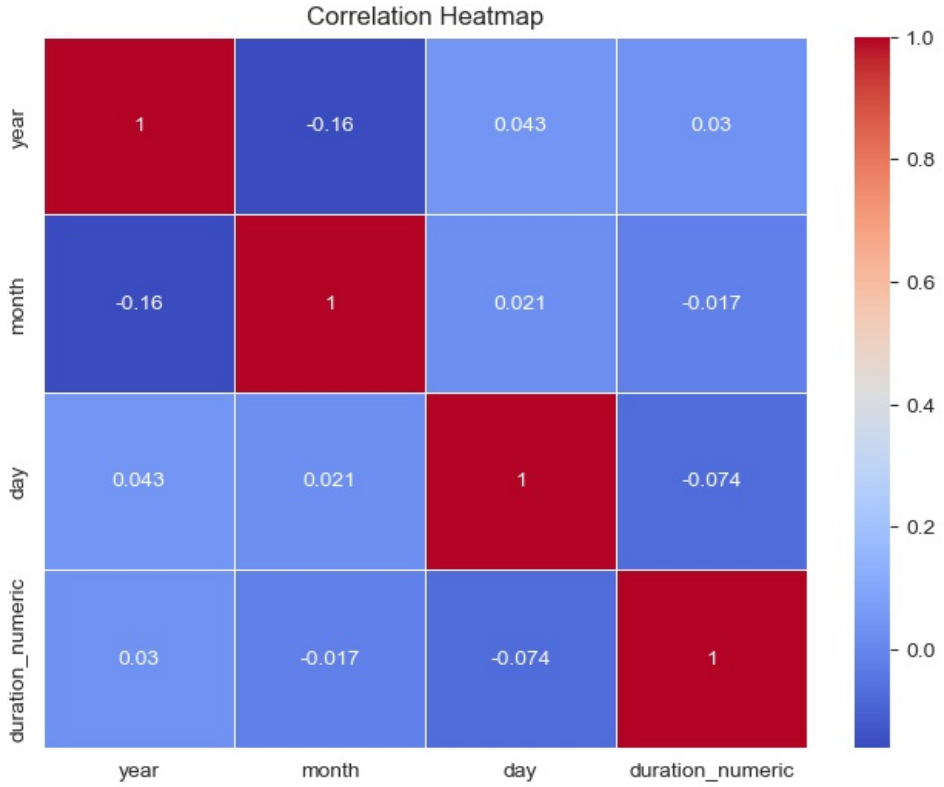
In [49]:
```
if "year" in data.columns and 'duration_minutes' in movie_data.columns:
  plt.figure(figsize=(12, 6))
  sns.lineplot(x=data["year"], y=movie_data['duration_minutes'], ci=None, marker="o", color="blue")
```

```
  plt.xlabel("Release Year")
  plt.ylabel("Average Duration (minutes)")
  plt.title("Evolution of Content Duration Over Time")
  plt.show()
else:
  print("Columns 'Release_Year' or 'Duration(minutes)' not found in the dataset.")
```

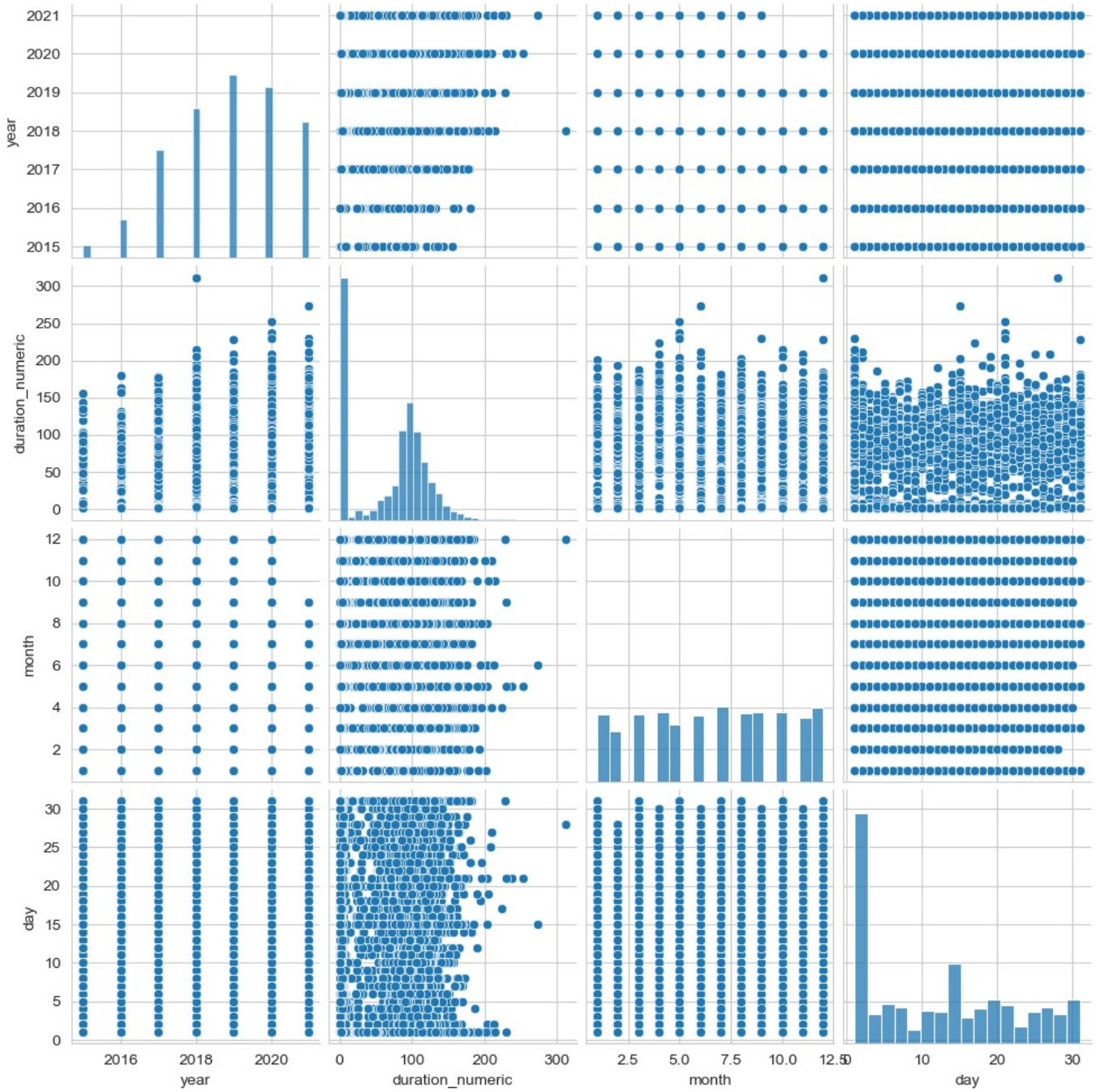


Correlations between Numeric columns

In [50]:
```
data['duration_numeric'] = data['duration'].apply(
    lambda x: int(x.split()[0]) if x.split()[1] in ['min', 'Season', 'Seasons'] else 0
)
plt.figure(figsize=(8, 6))
sns.heatmap(data[['year', 'month', 'day', 'duration_numeric']].corr(), annot=True, cmap='coolwarm', linewidths=
plt.title("Correlation Heatmap")
plt.show()
```



In [51]:
```
sns.pairplot(data[["year", "duration_numeric", "month","day"]])
plt.show()
```



1. Summary of Findings:

1. **Dataset Overview:** Contains 8,790 entries with details about

movies and TV shows, including type, title, director, cast, country, release year, rating, duration, and genres.

2. **Content Type:** Movies dominate (70% of entries), with the rest being TV shows.
3. **Release Year:** Most content was released between 2017 and 2021, showing a focus on recent productions.
4. **Country of Production:** The United States leads, followed by India and the UK. A significant portion of entries lacks specific country data.
5. **Ratings:** "TV-MA" (Mature Audience) is the most common, highlighting an audience preference for mature content.
6. **Genres:** Popular genres include "Dramas," "Comedies," "Action & Adventure," and "Documentaries."
7. **Duration:**

- *Movies:* Feature lengths typically range between 90-120 minutes.
- *TV Shows:* Most shows have 1-3 seasons, reflecting a preference for shorter series.

2. Key Trends and Patterns:

- **Steady Growth:** Content production increased significantly after 2010, peaking in the last five years.

- **Global Appeal:** Genres like "International Movies" and "TV Dramas" cater to diverse audiences.
- **Ratings Concentration:** Focus on content rated "TV-MA" or "TV-14" aligns with teenage and adult demographics.
- **Season Lengths:** TV shows tend to avoid longer runs, with a noticeable preference for shorter series.

3. Recommendations and Insights:

1. **Content Production Strategy:**

- **Prioritize Movies:** Allocate more resources to feature-length films within the 90-120 minute range, especially in high-demand genres like "Dramas," "Comedies," and "Action & Adventure."
- **Short TV Shows:** Focus on producing limited series (1-3 seasons), which resonate well with audiences seeking concise storytelling.

2. **Target Audience Insights:**

- **Age Demographics:** Develop mature content ("TV-MA") for adult audiences and teenage-friendly shows ("TV-14") to capture the largest viewership segments.
- **Global Storytelling:** Expand "International Movies" to attract viewers in regions like India and the UK.

3. **Release Timing:**

- **Seasonal Trends:** Capitalize on popular release months, particularly June through September, to maximize engagement during high-demand periods.

4. **Genre Diversification:**

- **Niche Genres:** Explore combining underrepresented genres such as "Action Documentaries" or "Romantic Thrillers" to attract niche audiences.
- **Documentaries:** Continue investing in high-quality documentaries, which have shown consistent demand.

5. **Country-Specific Strategy:**

- **Regional Content:** Invest in productions tailored to regional audiences (e.g., Bollywood-style films for India or historical dramas for the UK).
- **Global Distribution:** Enhance visibility of non-U.S. content to broaden international appeal.

6. **TV Show Optimization:**

- **Season Length:** Develop shows with 1-3 seasons to align with current viewer preferences.
- **Serialized Content:** For longer series, focus on serialized storytelling to retain engagement across seasons.

4. Conclusion:

The dataset highlights significant audience preferences for movies, mature content, and globally resonant storytelling. A strategic focus on producing concise, high-quality content in diverse genres and regions can help platforms meet evolving viewer demands while maximizing reach and engagement.

In [ ]: