Business Case : Problem Statement

To identify trends in Netflix content and viewership patterns to recommend strategies for improving customer engagement and business growth.

In [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

1. Data Loading and Analysing basic metrics

In [2]:
```
!gdown 1H0352lf0fxtGxWY5MhN2VmCb8jIpOOdp
```

```
Downloading...
From: https://drive.google.com/uc?id=1H0352lf0fxtGxWY5MhN2VmCb8jIpOOdp
To: /content/netflix.csv

  0% 0.00/3.40M [00:00<?, ?B/s]
100% 3.40M/3.40M [00:00<00:00, 67.1MB/s]
```

In [3]:
```
netflix = pd.read_csv("netflix.csv")
```

In [4]:
```
netflix.rename({"listed_in":"genre"}, axis=1, inplace=True)
```

In [5]:
```
# Inspecting random 7 rows from the dataset.

netflix.sample(7)
```

Out[5]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	genre	description
7945	s7946	Movie	Saturday Church	Damon Cardasis	Luka Kain, Regina Taylor, Margot Bingham, Marq...	United States	October 6, 2020	2017	TV-MA	83 min	Dramas, LGBTQ Movies	A teen struggling with gender identity and fam...
4215	s4216	TV Show	See You in Time	NaN	Hans Chung, Mini Tsai, Albee Huang, David Chiu...	NaN	January 4, 2019	2017	TV-14	1 Season	International TV Shows, Romantic TV Shows, TV ...	A series of mysterious text messages from the ...
3897	s3898	TV Show	KO One	NaN	Aaron Yan, Jiro Wang, Calvin Ka-Sing Chen, Dan...	Taiwan	April 19, 2019	2005	TV-MA	1 Season	International TV Shows, TV Action & Adventure,...	Gifted with special powers, fighting skills an...
4449	s4450	Movie	Life in a ... Metro	Anurag Basu	Dharmendra, Irrfan Khan, Konkona Sen Sharma, K...	India	November 1, 2018	2007	TV-14	126 min	Dramas, International Movies, Music & Musicals	A group of Mumbai up-and-comers search for lov...
6415	s6416	Movie	Candyman	Bernard Rose	Virginia Madsen, Tony Todd, Xander Berkeley, K...	United States, United Kingdom	October 1, 2019	1992	R	99 min	Cult Movies, Horror Movies	Grad student Helen Lyle unintentionally summon...
7343	s7344	TV Show	Love Cuisine	NaN	Lego Lee, Allison Lin, Duncan Chow, Nita Lei, ...	Taiwan	August 1, 2016	2015	TV-MA	1 Season	International TV Shows, Romantic TV Shows, TV ...	A successful Taiwanese chef moves home from Eu...
7004	s7005	Movie	Hot Property	Max McGill	MyAnna Buring, Tom Rhys Harries, Ella Smith, S...	United Kingdom	August 13, 2017	2016	TV-MA	82 min	Comedies, Independent Movies	When a resourceful but flighty Londoner is thr...

In [6]:
```
# Getting the number of rows & columns.

netflix.shape
```

Out[6]:
```
(8807, 12)
```

- The dataset contains 8807 rows and 12 columns.

In [7]:
```
# Getting the name of all the columns.

netflix.columns
```

Out[7]:
```
Index(['show_id', 'type', 'title', 'director', 'cast', 'country', 'date_added',
       'release_year', 'rating', 'duration', 'genre', 'description'],
      dtype='object')
```

In [8]:
```
# Getting the datatypes of columns and non-missing value count.

netflix.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  genre         8807 non-null   object
 11  description   8807 non-null   object
dtypes: int64(1), object(11)
memory usage: 825.8+ KB
```

The dataset contains:

- 11 object (string) columns & 1 integer column (release_year).
- Missing values in columns like director, cast, country, date_added, rating, duration.

In [9]:
```
# Checking if the dataset contains duplicate rows.

netflix.duplicated().any()
```

Out[9]:
```
False
```

- The dataset has no duplicate redundant rows.

In [10]:
```
# Getting the statistical summary of integer column : release_year.

netflix.describe()
```

Out[10]:

	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

- Most releases are recent, **with 50% of them from 2017 or later**.

In [11]:
```
# Getting the statistical summary of 11 object column's.

netflix.describe(include="object")
```

Out[11]:

	show_id	type	title	director	cast	country	date_added	rating	duration	genre	description
count	8807	8807	8807	6173	7982	7976	8797	8803	8804	8807	8807
unique	8807	2	8807	4528	7692	748	1767	17	220	514	8775
top	s1	Movie	Dick Johnson Is Dead	Rajiv Chilaka	David Attenborough	United States	January 1, 2020	TV-MA	1 Season	Dramas, International Movies	Paranormal activity at a lush, abandoned prope...
freq	1	6131	1	19	19	2818	109	3207	1793	362	4

- The dataset is diverse, with **8,807 unique content titles.**

2. Data Cleaning

In [12]:
```
# Null value analysis:

(netflix.isna().sum()/len(netflix)*100).rename("Percentage")
```

Out[12]:

	Percentage
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
genre	0.000000
description	0.000000

dtype: float64

- There is a **substantial percentage of missing values in director, cast, country.**

In [13]:
```
# Finding which columns have less than 5% missing values.

dropped_columns = netflix.columns[(netflix.isna().sum() < (0.05 * netflix.shape[0])) & (netflix.isna().sum() > 0)]
dropped_columns
```

Out[13]:
```
Index(['date_added', 'rating', 'duration'], dtype='object')
```

In [14]:
```
# Dropping rows with columns having less than 5% missing values and for greater than 5% ,filling it as Unknown, for now.

netflix.dropna(subset=dropped_columns, inplace=True, axis=0)
netflix["director"] = netflix["director"].fillna("Unknown")
netflix["cast"] = netflix["cast"].fillna("Unknown")
netflix["country"] = netflix["country"].fillna("Unknown")

netflix.isna().sum().rename("count")
```

Out[14]:

	count
show_id	0
type	0
title	0
director	0
cast	0
country	0
date_added	0
release_year	0
rating	0
duration	0
genre	0
description	0

dtype: int64

In [15]:
```
# Fixing the structure of columns: genre, date_added, duration and few others. Also adding new columns : month_added, year_added, date_added, day_added.

netflix["genre"] = netflix["genre"].str.strip()
netflix["genre"] = netflix["genre"].str.split(", ")

netflix["date_added"] = pd.to_datetime(netflix["date_added"].str.strip())
netflix["month_added"] = netflix["date_added"].dt.strftime("%B")
netflix["year_added"] = netflix["date_added"].dt.year
netflix["day_added"] = netflix["date_added"].dt.day

netflix["duration"] = netflix["duration"].str.replace(" min", "", regex=False)
netflix["duration"] = netflix["duration"].str.replace(r"\D", "", regex=True)
netflix["duration"] = netflix["duration"].astype(int)

netflix["country"] = netflix["country"].str.strip()

netflix["rating"] = netflix["rating"].str.strip()

netflix["type"] = netflix["type"].str.strip()
```

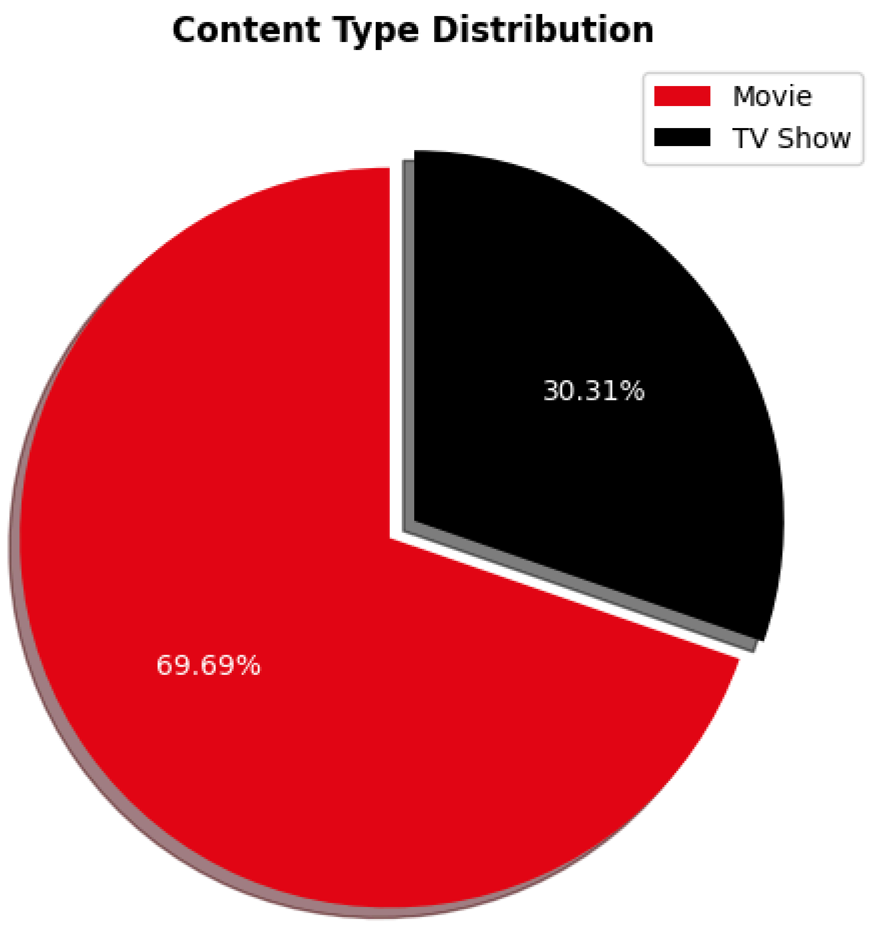
3. General Analysis

In [16]:
```
content_type = netflix["type"].value_counts()

plt.figure(figsize=(6,6))
plt.pie(content_type, labels=content_type.index, startangle=90, explode=(0.08,0), autopct="%.2f%%", colors=["#E50914", "black"], shadow=True, textprops={"color":"white"})
plt.title("Content Type Distribution", fontweight="bold")
plt.legend()

netflix["type"].value_counts()
```

Out[16]:

type	count
Movie	6126
TV Show	2664

dtype: int64



- The dataset contains **6126 movies & 2664 TV Shows.**

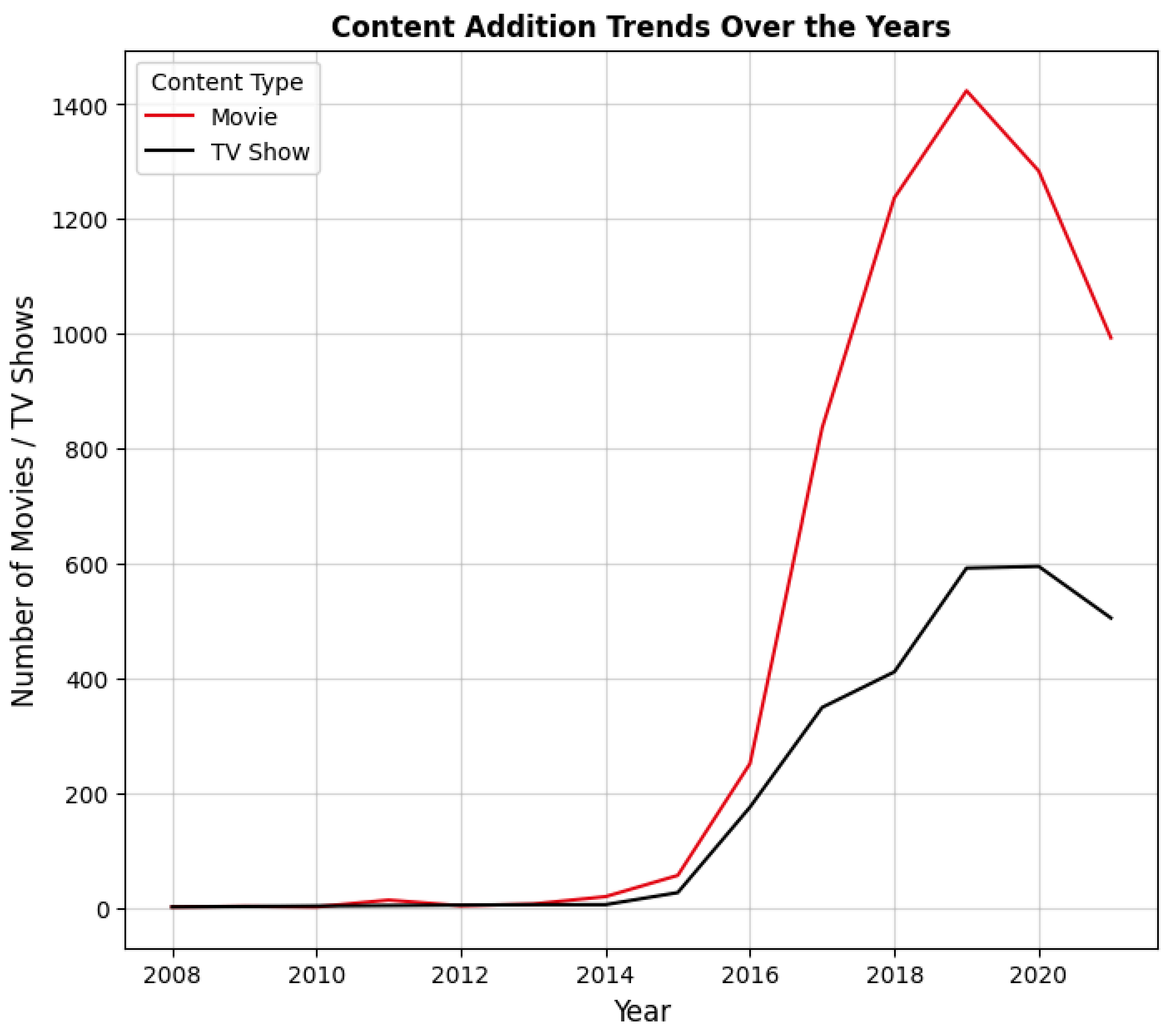
In [17]:
```
netflix.groupby("type", observed=False).agg(min_year = ("release_year", "min"), max_year = ("release_year", "max"))
```

Out[17]:

type	min_year	max_year
Movie	1942	2021
TV Show	1925	2021

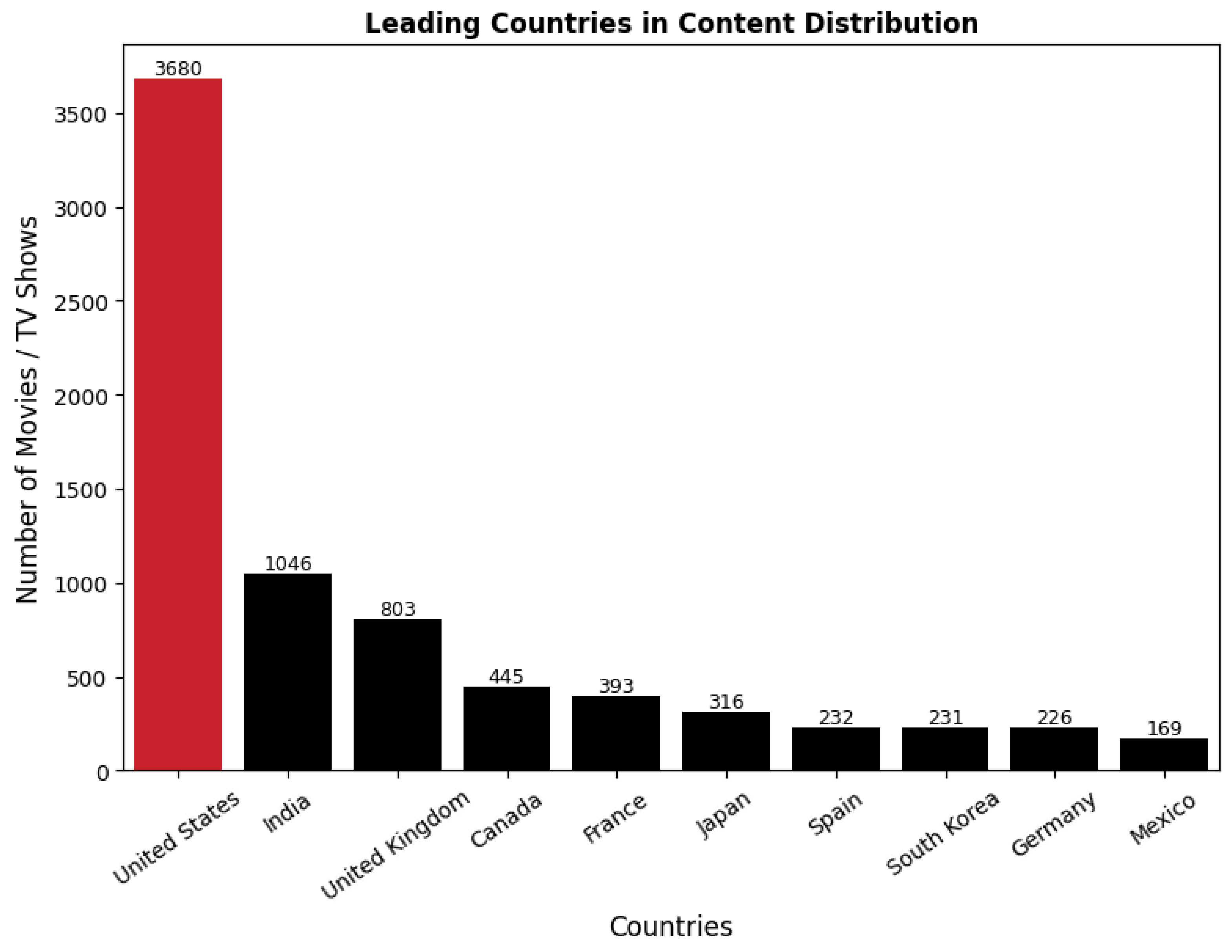
- The dataset contains **Movies from 1942 to 2021 and TV Shows from 1925 to 2021.**

In [18]:
```
content_per_year = netflix[["year_added", "type"]].value_counts().sort_index().reset_index()

plt.figure(figsize=(8,7))
sns.lineplot(data=content_per_year, x="year_added", y="count", hue="type",palette=["#E50914", "black"])
plt.legend().set_title("Content Type")
plt.xlabel("Year", fontsize=12)
plt.ylabel("Number of Movies / TV Shows", fontsize=12)
plt.title("Content Addition Trends Over the Years", fontweight="bold")
plt.grid(alpha=0.5)
plt.show()
```



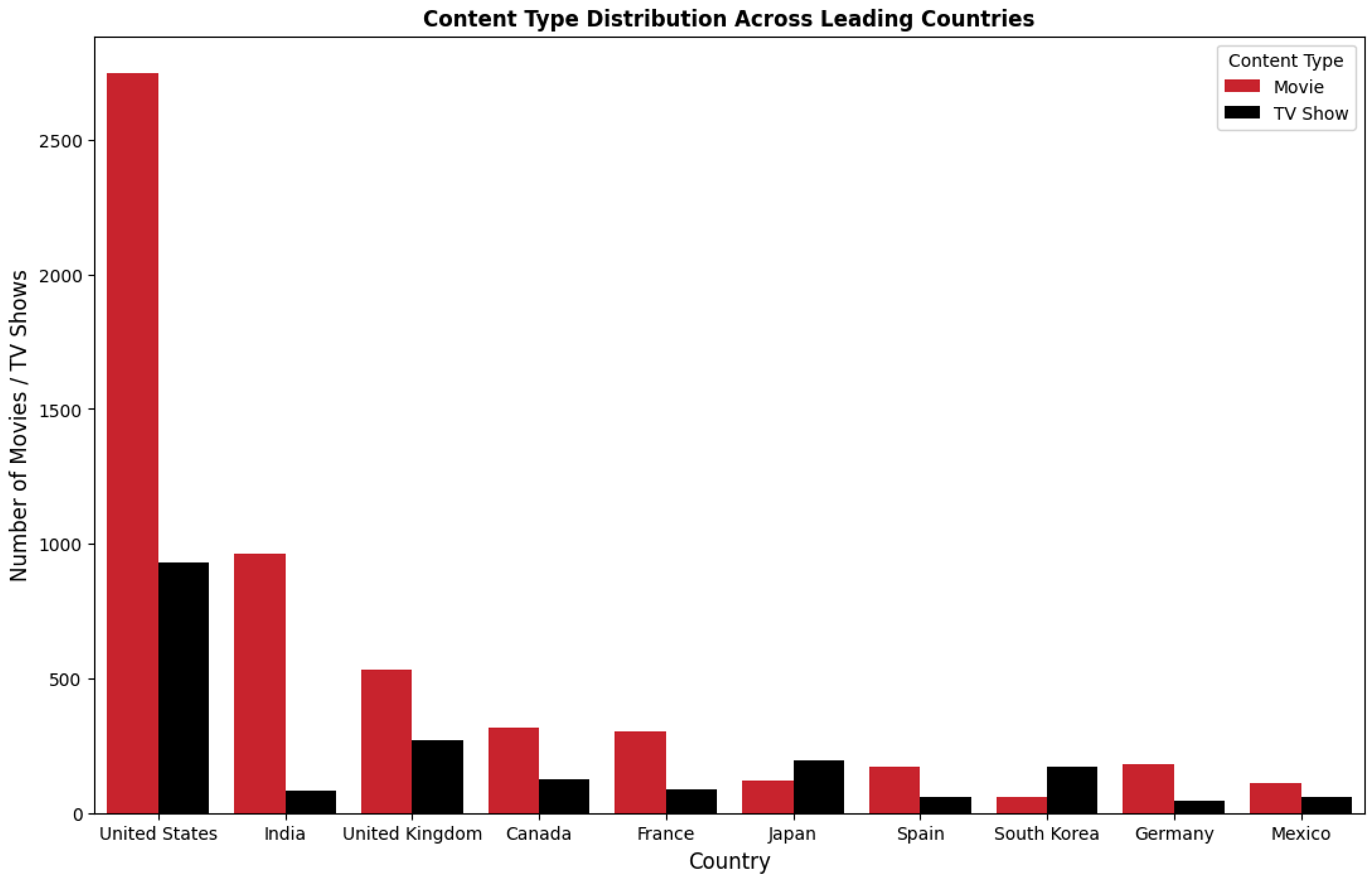
- **From 2014**, there was a sharp increase in the addition of both Movies and TV Shows, followed by a decline around **2019-2020.**

In [19]:
```
top_countries = netflix["country"].str.split(", ").explode().value_counts().drop("Unknown").iloc[0:10].reset_index()

plt.figure(figsize=(9,6))
colors = ["#E50914"] + ["black"] * (len(top_countries) - 1)
bar_plot = sns.barplot(data=top_countries, x="country", y="count", hue="country", palette=colors)
plt.xlabel("Countries", fontsize=12)
plt.ylabel("Number of Movies / TV Shows", fontsize=12)
plt.title("Leading Countries in Content Distribution", fontweight="bold")
plt.xticks(rotation=35)

for index, value in enumerate(top_countries["count"]):
    plt.text(index, value, str(value), ha='center', va='bottom', fontsize=9)


plt.show()
```



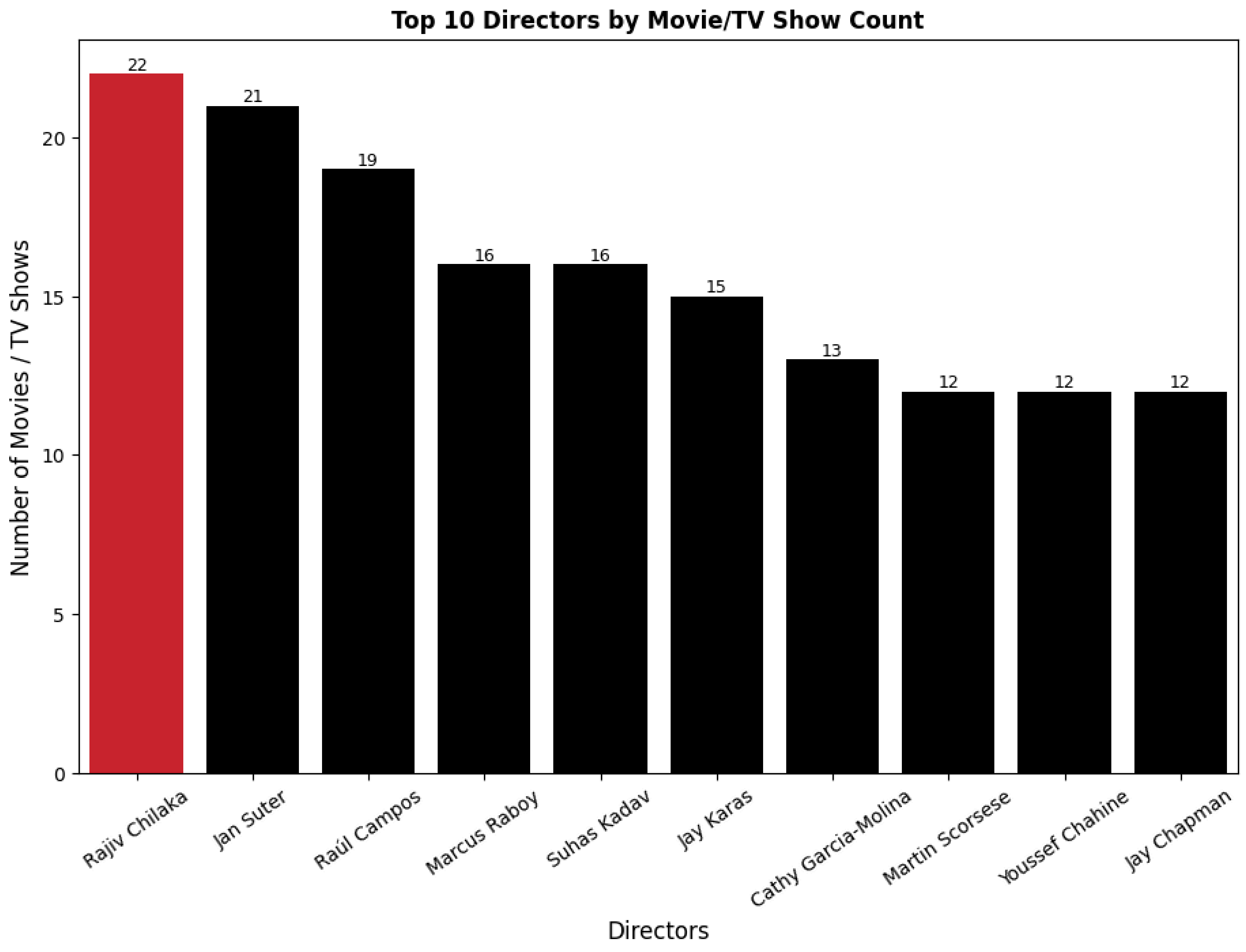
- **USA** has the highest number of content, followed by **India, UK, Canada & France Korea.**
- USA produces **3.5 times more** content than India.

In [20]:
```
top_countries = netflix["country"].str.split(", ").explode().value_counts().drop("Unknown").iloc[0:10].index

exploded_netflix = netflix.assign(country=netflix["country"].str.split(", ")).explode("country")
top_countries_type = exploded_netflix.loc[exploded_netflix["country"].isin(top_countries),["type","country"]]


plt.figure(figsize=(13,8))
sns.countplot(data=top_countries_type, x="country", hue="type", palette=["#E50914", "black"], order=top_countries)
plt.title("Content Type Distribution Across Leading Countries", fontweight="bold")
plt.legend().set_title("Content Type")
plt.xlabel("Country", fontsize=12)
plt.ylabel("Number of Movies / TV Shows", fontsize=12)
plt.show()
```



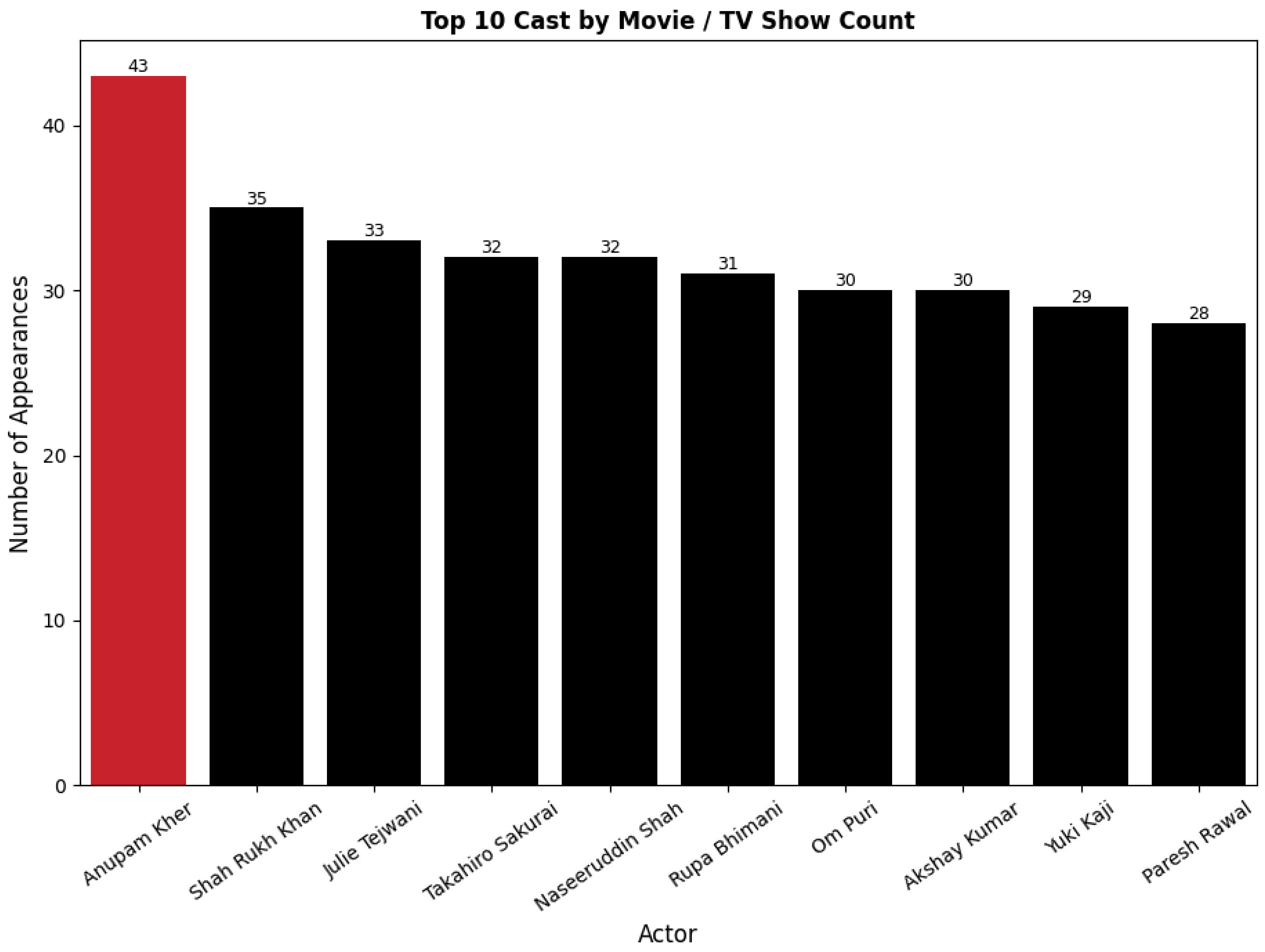
- The **top 5 leading countries** generally have a **higher number of movies compared to TV shows.** However, **South Korea and Japan** stand out, as they show a **reverse trend with a greater number of TV shows than movies.**

In [21]:
```
top10_director = netflix["director"].str.split(", ").explode().value_counts().iloc[1:11].reset_index()

plt.figure(figsize=(11,7))
colors = ["#E50914"] + ["black"] * (len(top10_director) - 1)
bar_plot = sns.barplot(data=top10_director, x="director", y="count", hue="director", palette=colors)
plt.xlabel("Directors", fontsize=12)
plt.ylabel("Number of Movies / TV Shows", fontsize=12)
plt.title("Top 10 Directors by Movie/TV Show Count", fontweight="bold")
plt.xticks(rotation=35)

for index, value in enumerate(top10_director["count"]):
    plt.text(index, value, str(value), ha='center', va='bottom', fontsize=9)

plt.show()
```



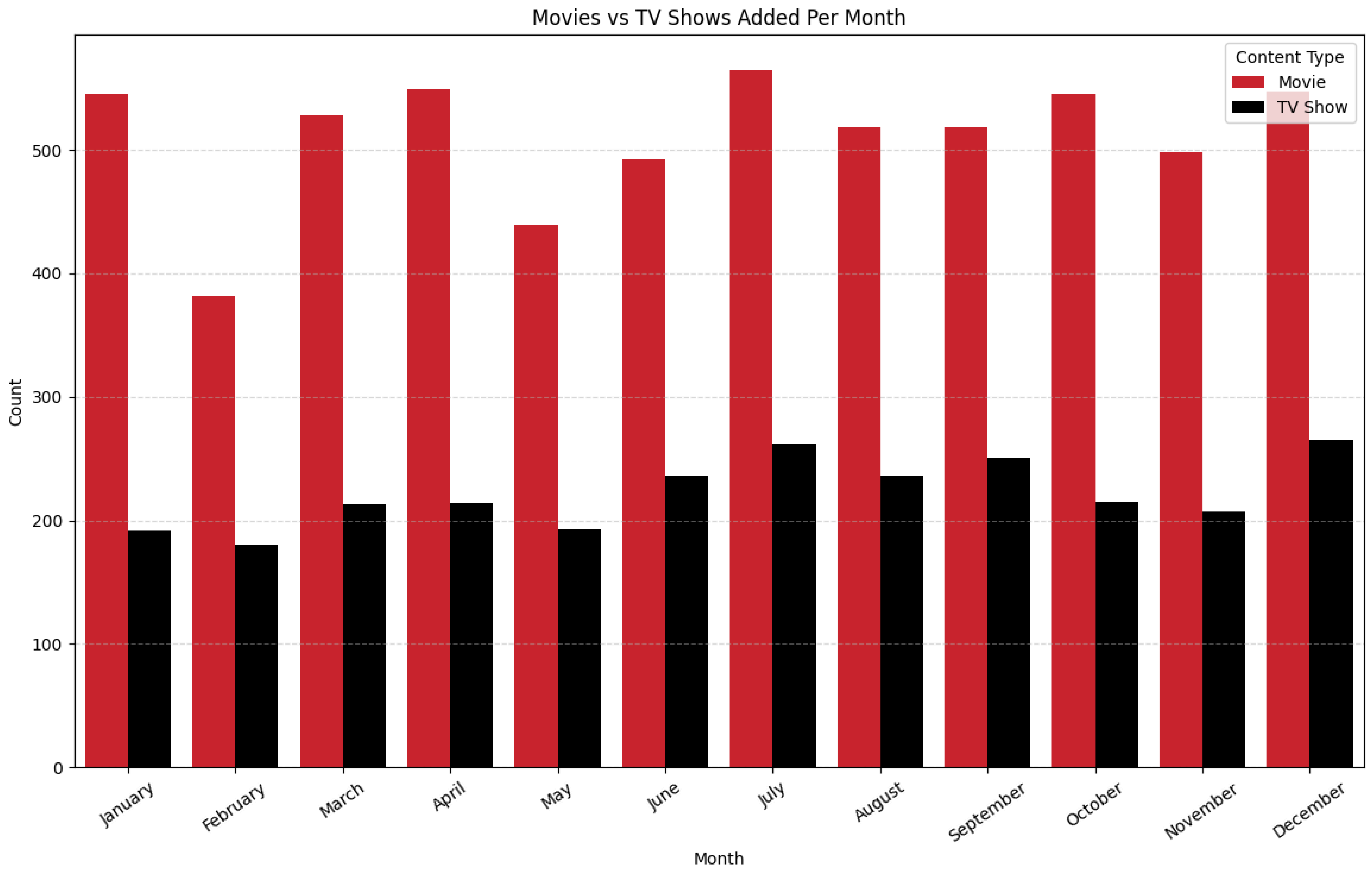
- **Rajiv Chilaka, Jan Suter, Raúl Campos** are the top 3 leading directors with most number of content.

In [22]:
```python
top10_cast = netflix["cast"].str.split(", ").explode().value_counts().iloc[1:11].reset_index()

plt.figure(figsize=(11,7))
colors = ["#E50914"] + ["black"] * (len(top10_cast) - 1)
bar_plot = sns.barplot(data=top10_cast, x="cast", y="count", hue="cast", palette=colors)
plt.xlabel("Actor", fontsize=12)
plt.ylabel("Number of Appearances", fontsize=12)
plt.title("Top 10 Cast by Movie / TV Show Count", fontweight="bold")
plt.xticks(rotation=35)

for index, value in enumerate(top10_cast["count"]):
    plt.text(index, value, str(value), ha='center', va='bottom', fontsize=9)

plt.show()
```



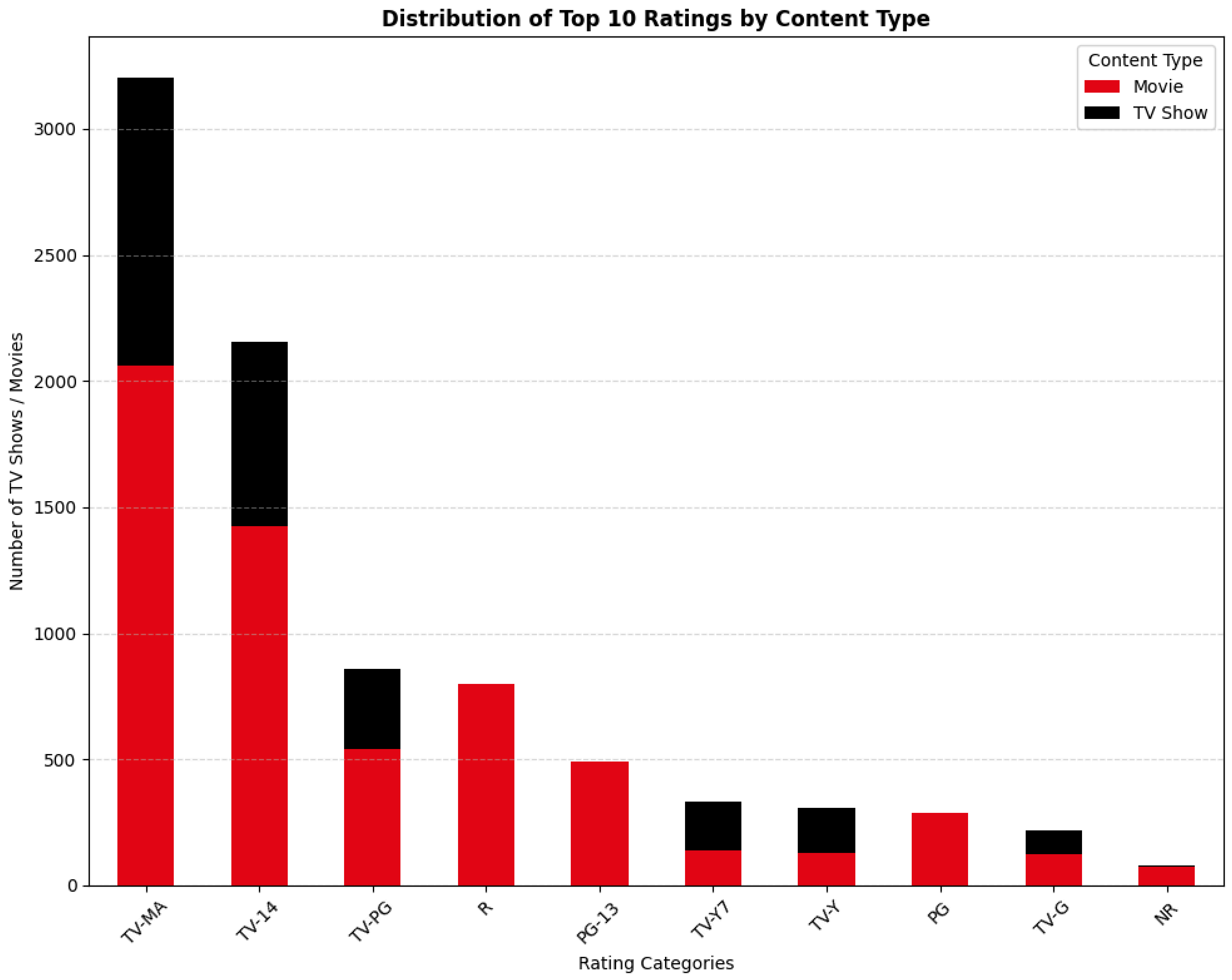
- **Anupam Kher, Shah Rukh Khan, Julie Tejwani** are the top 3 leading cast with most number of content.

In [23]:
```
month_order = ["January","February","March","April","May","June","July","August","September","October","November","December"]
movie_per_month = netflix[netflix["type"] == "Movie"]['month_added'].value_counts().reindex(month_order)
tv_shows_per_month = netflix[netflix["type"] == "TV Show"]['month_added'].value_counts().reindex(month_order)

plt.figure(figsize=(14, 8))
sns.countplot(data=netflix, x="month_added", hue="type", order=month_order, palette={"Movie": "#E50914", "TV Show": "black"})
plt.xlabel("Month")
plt.ylabel("Count")
plt.title("Movies vs TV Shows Added Per Month")
plt.xticks(rotation=35)
plt.legend(title="Content Type")
plt.grid(axis="y", linestyle="--", alpha=0.5, zorder=0)
plt.show()
```



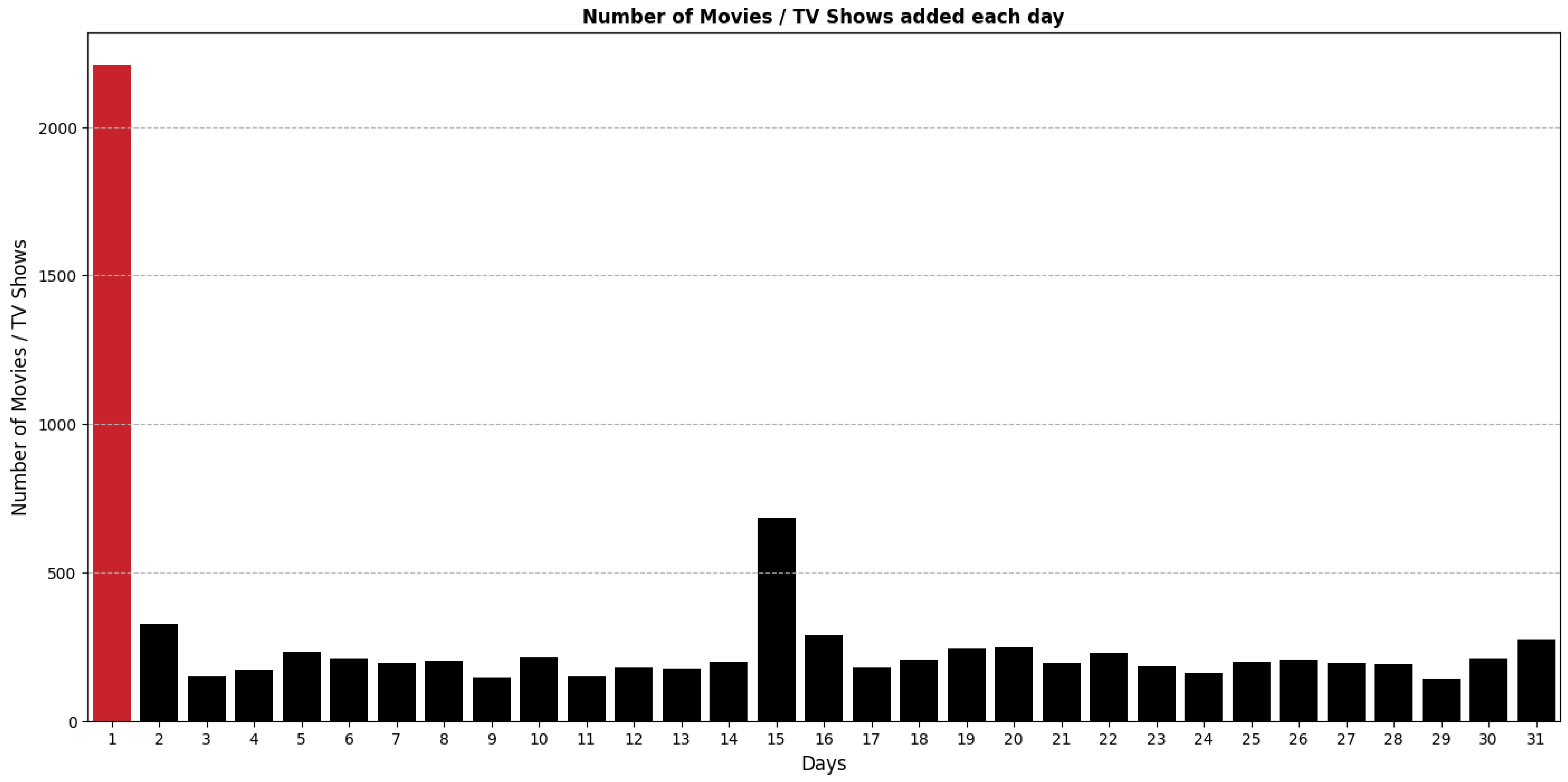
- Movies are consistently added in **higher numbers** than TV shows each month, with **July** having the **highest spike for both categories.**

```
In [24]: top10_ratings = netflix["rating"].value_counts().reset_index().iloc[:10]
top10_data = netflix[netflix["rating"].isin(top10_ratings["rating"])]
ratings_crosstab = pd.crosstab(index=top10_data["rating"], columns=top10_data["type"])
ratings_crosstab = ratings_crosstab.loc[ratings_crosstab.sum(axis=1).sort_values(ascending=False).index]

ratings_crosstab.plot(kind='bar', stacked=True, figsize=(10, 8), color=["#E50914", "black"])
plt.xlabel('Rating Categories')
plt.ylabel('Number of TV Shows / Movies')
plt.title("Distribution of Top 10 Ratings by Content Type", fontweight="bold")
plt.legend(title="Content Type", labels=["Movie", "TV Show"])
plt.xticks(rotation=45)
plt.grid(axis="y", linestyle="--", alpha=0.5, zorder=0)
plt.tight_layout()
plt.show()
```



- The **top three rating categories** are TV-MA (Mature Audience), TV-14 (14 and Older) and TV-PG (Parental Guidance).

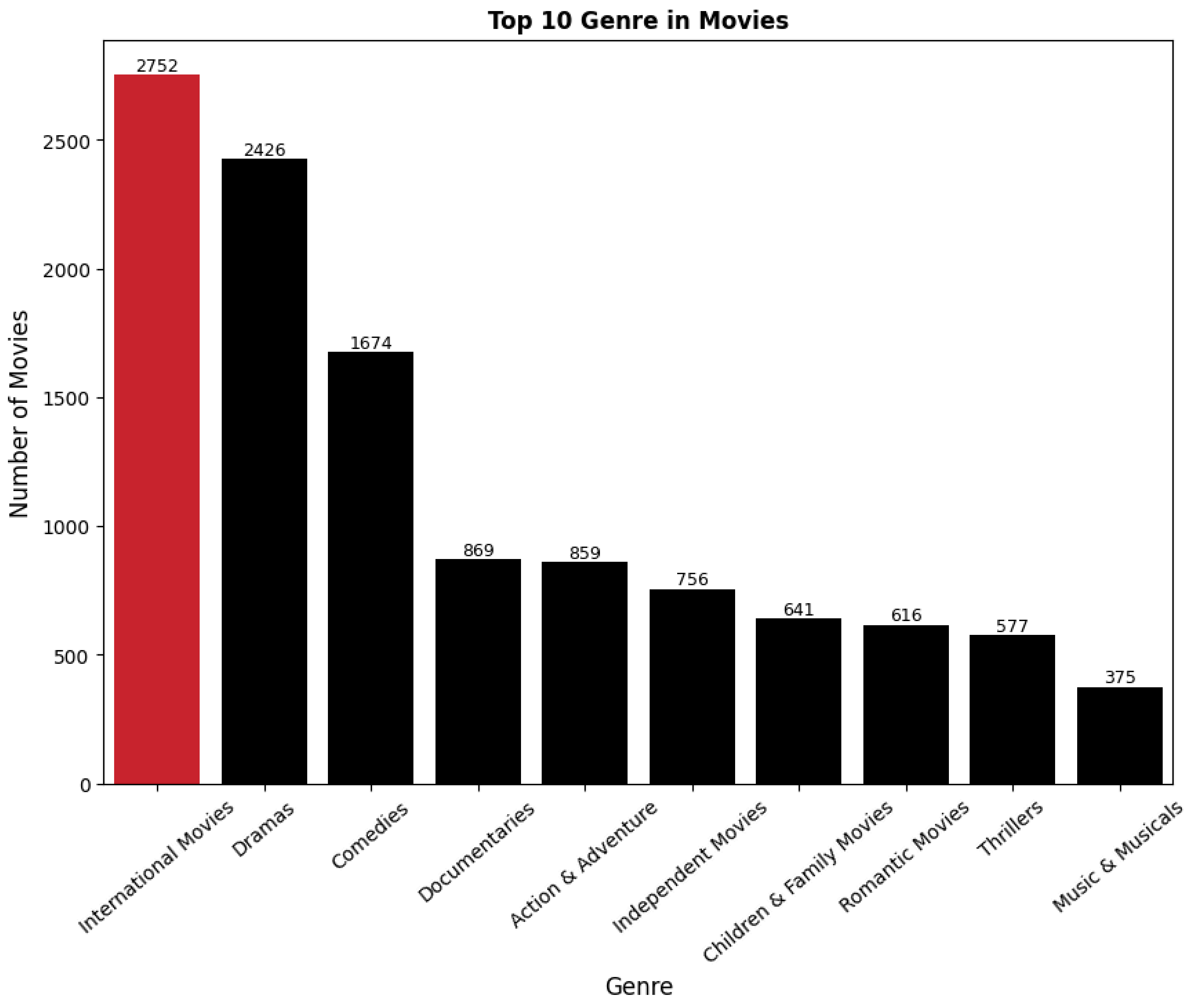
In [25]:
```
netflix_day = netflix["day_added"].value_counts().reset_index()

plt.figure(figsize=(17,8))
sns.barplot(data=netflix_day, x="day_added", y="count", hue="day_added", palette=["#E50914"] + ["black"] * 30, legend=False)
plt.title("Number of Movies / TV Shows added each day", fontweight="bold")
plt.xlabel("Days", fontsize=12)
plt.ylabel("Number of Movies / TV Shows", fontsize=12)
plt.grid(axis="y", linestyle="--", )
plt.show()
```



- A **significantly higher** number of content additions occur on the **1st day of each month.**

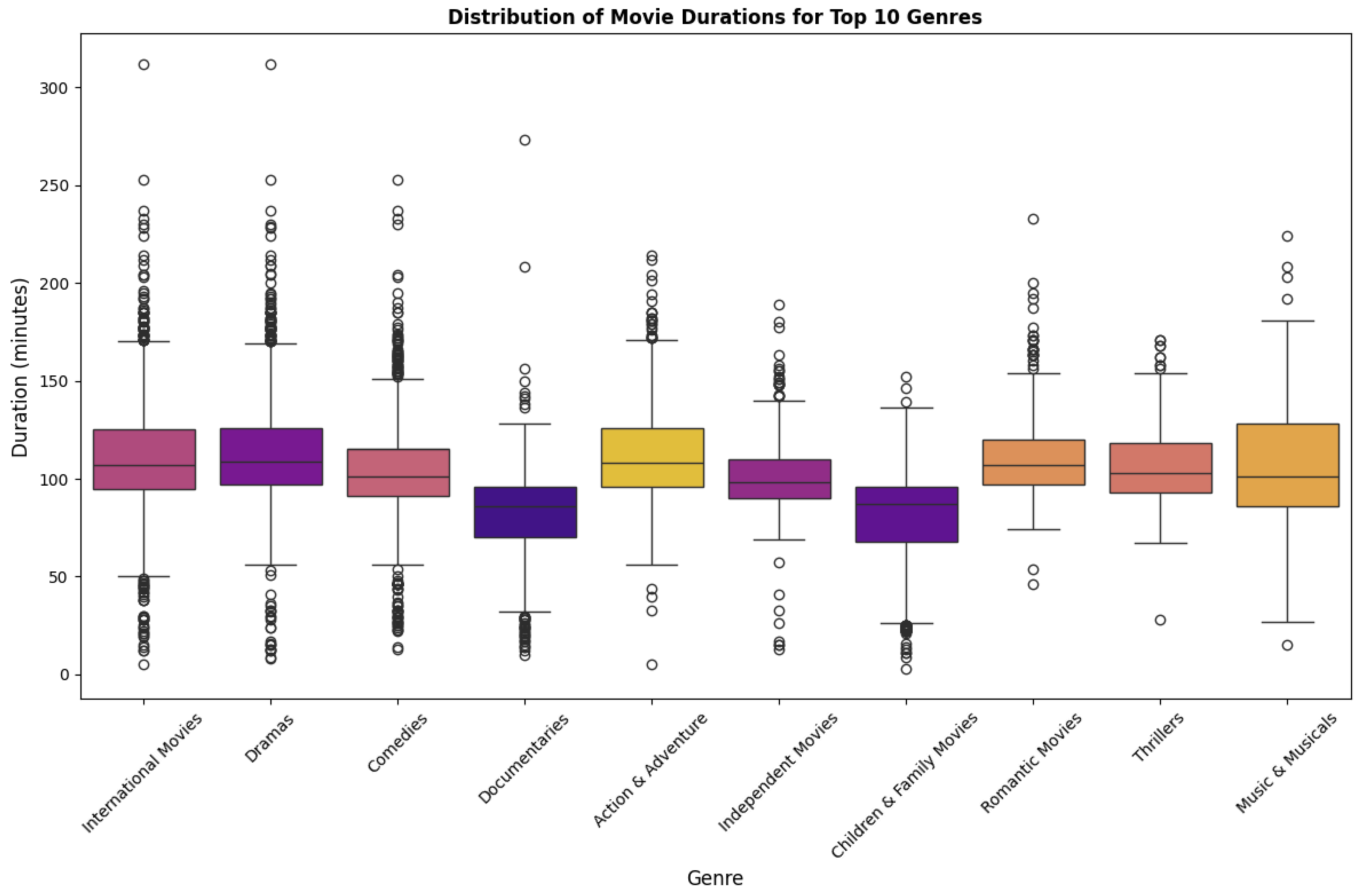
4. Movies Analysis

```
In [26]: movies_exploded = netflix[netflix["type"]=="Movie"].explode("genre")

top10_genre = movies_exploded["genre"].value_counts().iloc[0:10].reset_index()

plt.figure(figsize=(10,7))
colors = ["#E50914"] + ["black"] * (len(top10_genre) - 1)
bar_plot = sns.barplot(data=top10_genre, x="genre", y="count", hue="genre", palette=colors)
plt.xlabel("Genre", fontsize=12)
plt.ylabel("Number of Movies", fontsize=12)
plt.title("Top 10 Genre in Movies", fontweight="bold")
plt.xticks(rotation=40)

for index, value in enumerate(top10_genre["count"]):
    plt.text(index, value, str(value), ha='center', va='bottom', fontsize=9)
plt.show()
```



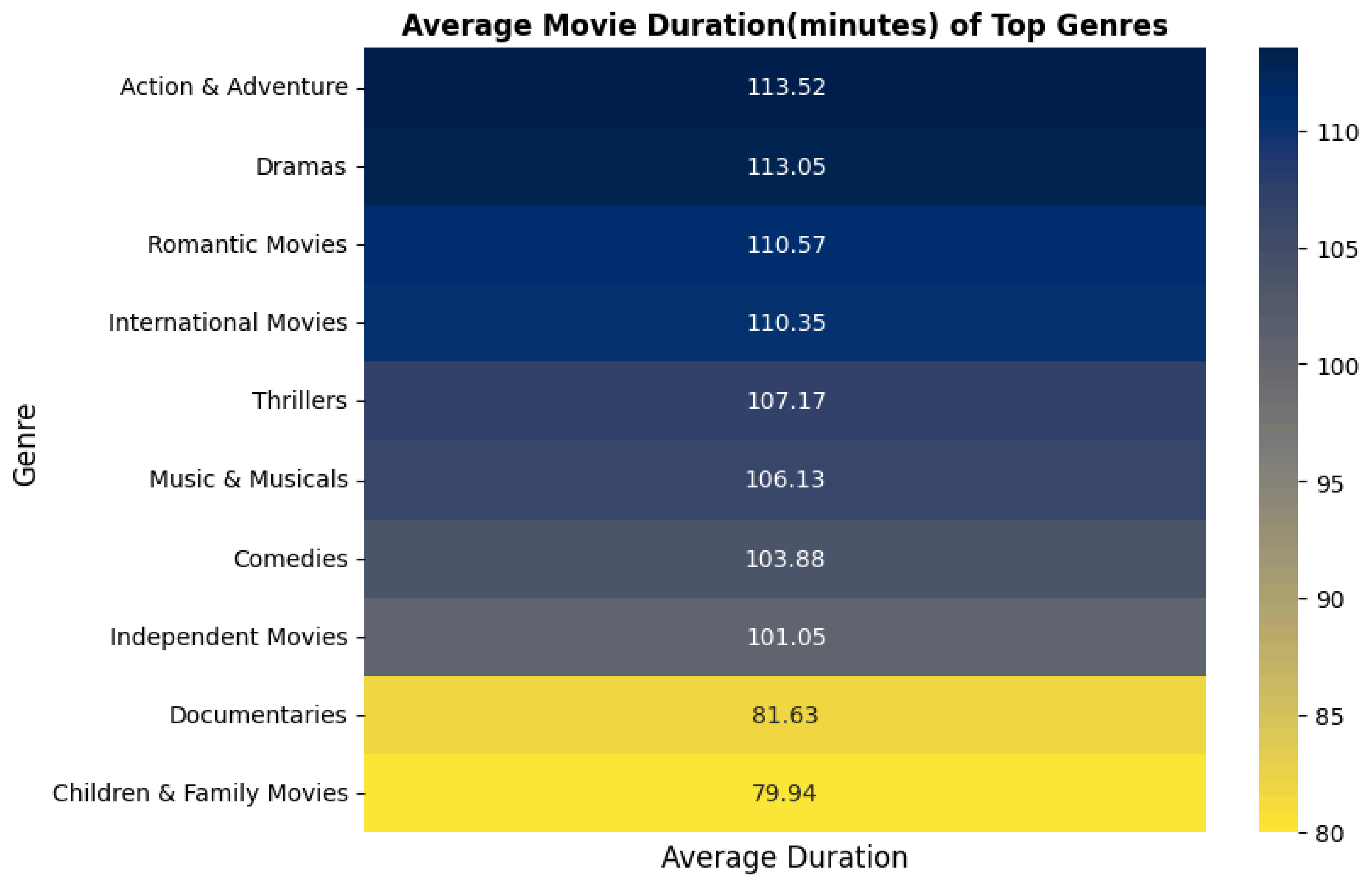
- International, Dramas, Comedies, Documentaries, Action & Advenrture are the **top 5 popular genre** movies.

In [27]:
```
top10_movies_genre = movies_exploded["genre"].value_counts().reset_index().iloc[:10]
top_movies_data = movies_exploded[movies_exploded["genre"].isin(top10_movies_genre["genre"])]


plt.figure(figsize=(12,8))
sns.boxplot(data=top_movies_data, x="genre", y="duration", hue="genre", palette="plasma", order=top10_movies_genre["genre"])
plt.title("Distribution of Movie Durations for Top 10 Genres", fontweight="bold")
plt.xlabel("Genre", fontsize=12)
plt.ylabel("Duration (minutes)", fontsize=12)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



- Movie durations **vary widely** across genres, with documentaries and children's movies tending to have **shorter durations**, while music & musicals and action & adventure films show a **broader range**.

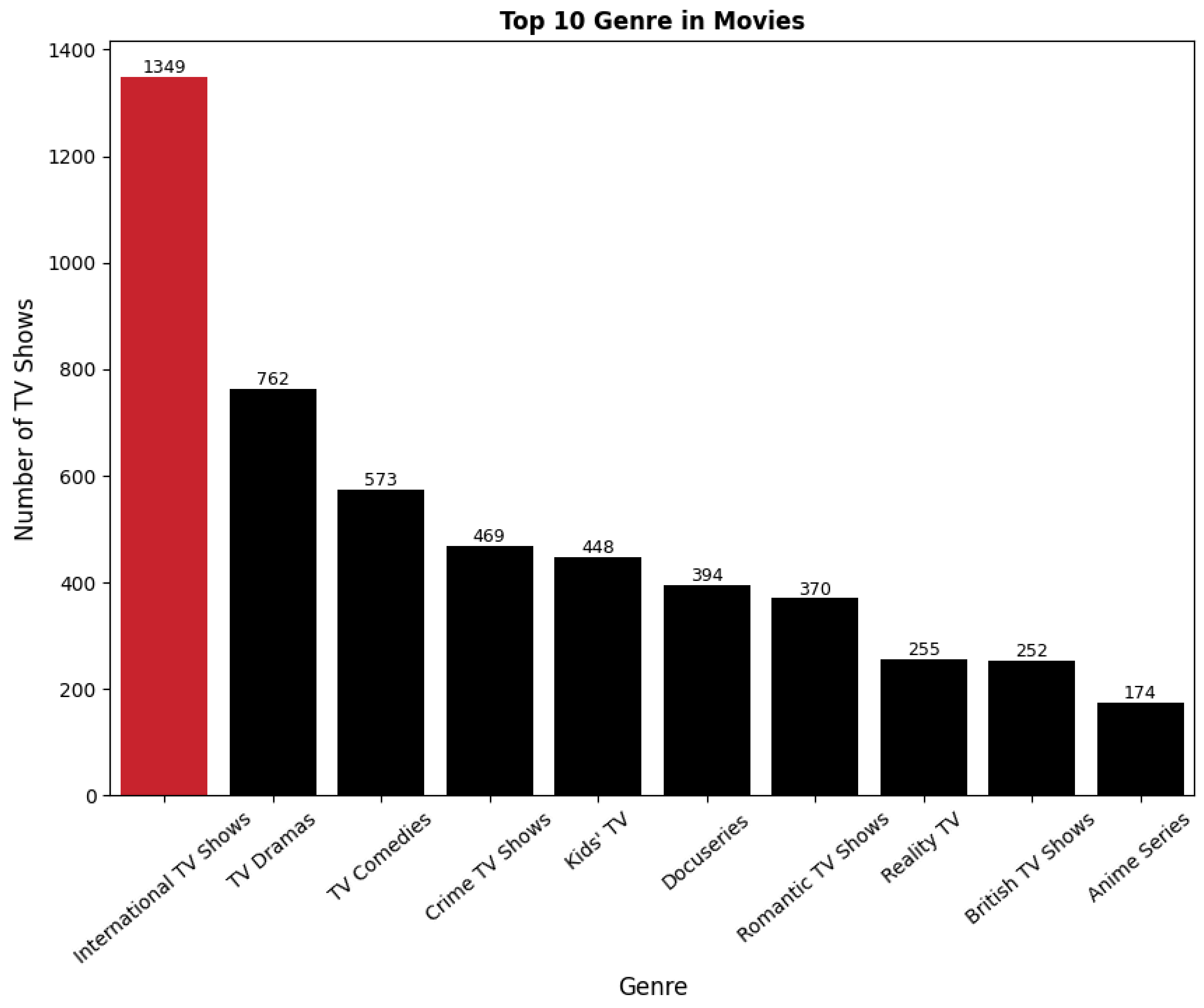
In [28]:
```
duration_mean_by_genre = top_movies_data.pivot_table(index="genre", values="duration", aggfunc="mean").sort_values(["duration"], ascending=False)

plt.figure(figsize=(8,6))
sns.heatmap(data=duration_mean_by_genre, annot=True, fmt=".2f", xticklabels="", cmap="cividis_r")
plt.title("Average Movie Duration(minutes) of Top Genres", fontsize=12, fontweight="bold")
plt.xlabel("Average Duration", fontsize=12)
plt.ylabel("Genre", fontsize=12)
plt.show()
```



- Action & Adventure and Dramas have the **highest average** movie durations (in minutes).

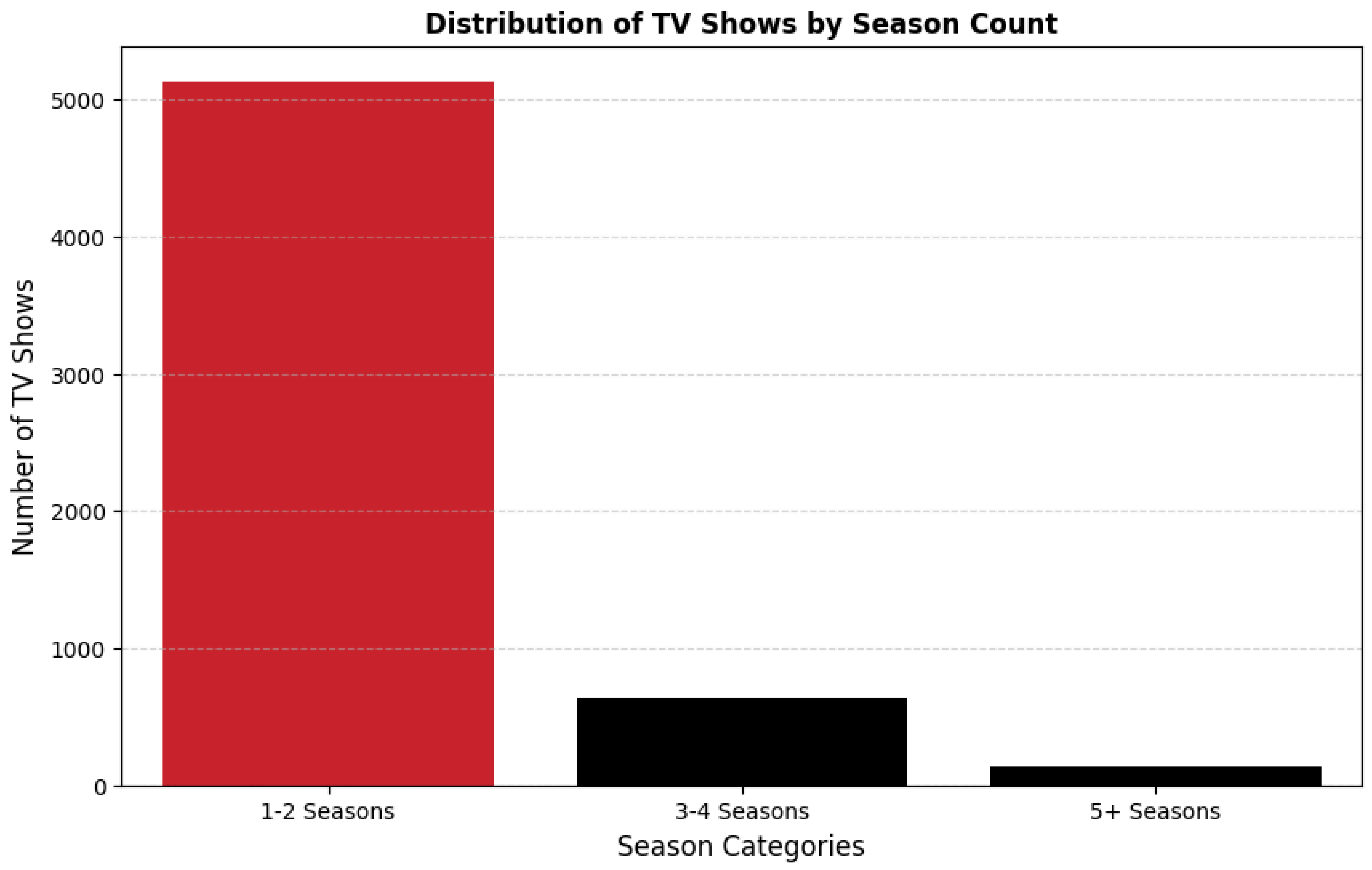
5. TV Shows Analysis

In [29]:
```
tv_shows_exploded = netflix[netflix["type"]=="TV Show"].explode("genre")
top10_genre = tv_shows_exploded["genre"].value_counts().reset_index().iloc[:10]

plt.figure(figsize=(10,7))
colors = ["#E50914"] + ["black"] * (len(top10_genre) - 1)
bar_plot = sns.barplot(data=top10_genre, x="genre", y="count", hue="genre", palette=colors)
plt.xlabel("Genre", fontsize=12)
plt.ylabel("Number of TV Shows", fontsize=12)
plt.title("Top 10 Genre in Movies", fontweight="bold")
plt.xticks(rotation=40)

for index, value in enumerate(top10_genre["count"]):
    plt.text(index, value, str(value), ha='center', va='bottom', fontsize=9)

plt.show()
```



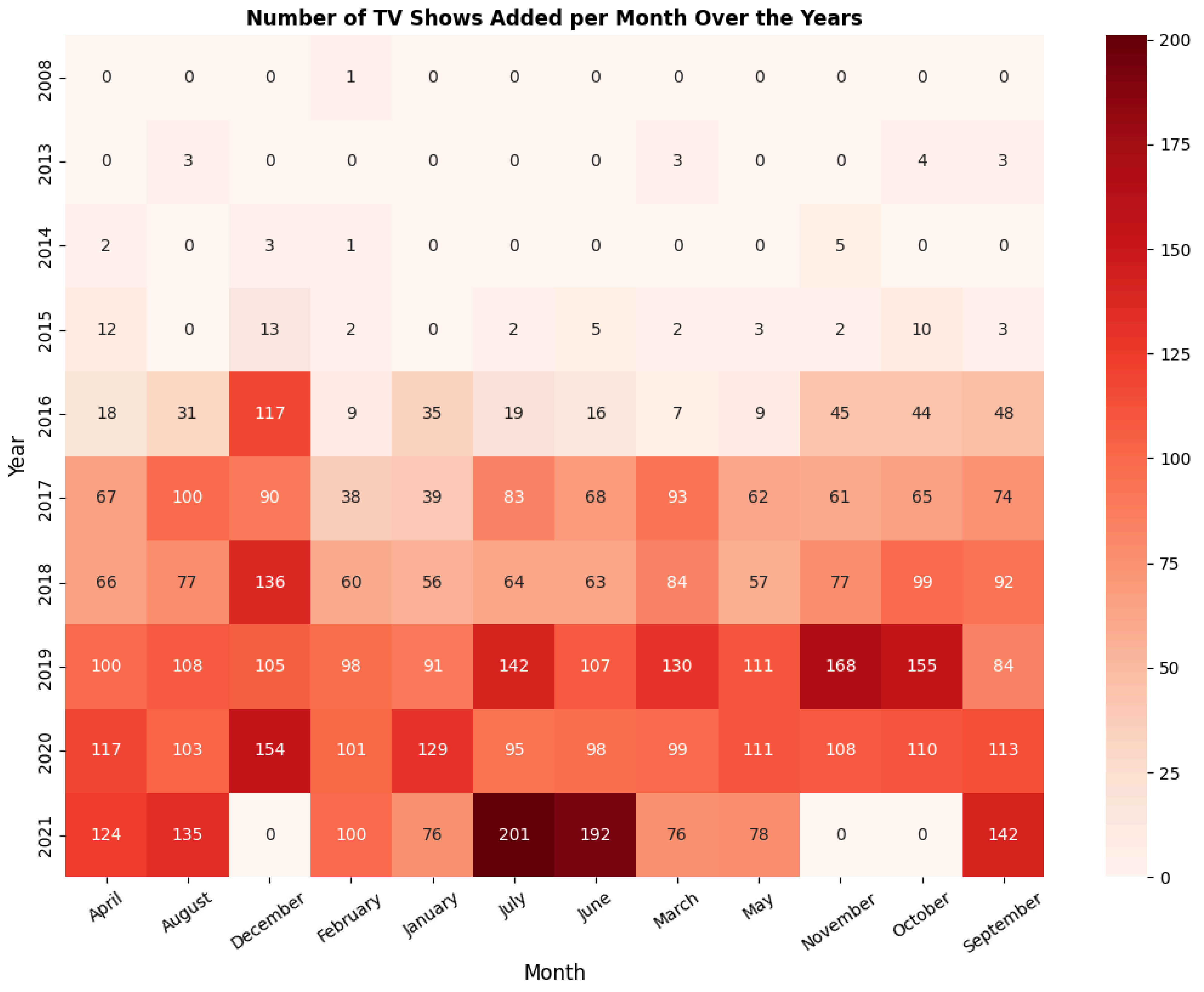
- International, Dramas, Comedies, Crime, Kids are **the top 5 popular genres** in TV Shows.

In [30]:
```
max_two_seasons, three_to_four_seasons, five_or_more_seasons  = 2,4,5
labels = ["1-2 Seasons", "3-4 Seasons", "5+ Seasons"]
bins = [0, max_two_seasons, three_to_four_seasons, five_or_more_seasons]

tv_shows_exploded["duration_category"] = pd.cut(tv_shows_exploded["duration"], labels=labels, bins=bins, right=True)
tv_show_duration_category = tv_shows_exploded["duration_category"].value_counts().reset_index()

plt.figure(figsize=(10,6))
sns.barplot(data=tv_show_duration_category, x="duration_category", y="count", hue="duration_category", palette=["#E50914"]+["black"]*2)
plt.title("Distribution of TV Shows by Season Count", fontweight="bold")
plt.xlabel("Season Categories", fontsize=12)
plt.ylabel("Number of TV Shows", fontsize=12)
plt.grid(axis="y", linestyle="--", alpha=0.5)
plt.show()
```



- The **majority** of TV Shows have **1-2 seasons,** while TV Shows with **5 or more seasons are few.**

In [31]:
```
tv_shows_pivot = tv_shows_exploded.pivot_table(index="year_added", columns="month_added", aggfunc="size", fill_value=0)

plt.figure(figsize=(13,9))
sns.heatmap(tv_shows_pivot, cmap="Reds", annot=True, fmt="d")
plt.title("Number of TV Shows Added per Month Over the Years", fontweight="bold")
plt.xlabel("Month", fontsize=12)
plt.ylabel("Year", fontsize=12)
plt.xticks(rotation=35)
plt.show()
```



- **July and June** exhibit the highest number of TV Shows additions compared to the other months.

6. Business Insights:

- **Content production peaked until 2020,** but after that, it declined, potentially due to COVID-19.
- **Movies are consistently added in higher numbers than TV Shows,** with a sharp increase in content additions from **2014 onwards** and a **decline post-2019.**
- **Netflix is focusing more on movies** compared to TV Shows, with a **70:30 movie-to-TV Show ratio.**

- **United States leads content production,** followed by **India, the UK, Canada, and France.**
- **South Korea and Japan stand out** as countries where TV Shows are more dominant than movies.
- **Rajiv Chilaka, Jan Suter, and Raúl Campos** are the most prolific directors on Netflix.
- **Anupam Kher Shah Rukh Khan, and Julie Tejwani** are among the most frequently appearing cast.
- **Mature audience content (TV-MA) dominates Netflix,** followed by **TV-14 and TV-PG categories.**
- **July** has the highest spike in content additions.
- **Action & Adventure and Dramas have the longest movie durations**, while Documentaries and Kid's movies are typically shorter.
- **TV Shows mostly have 1-2 seasons,** while those with 5+ seasons are rare.

7. Recommendations:

- Focus on producing more **International Movies, Dramas, and Comedies,** as these genres are the most popluar across Netflix.
- **Prioritize movie releases over TV Shows,** as Netflix has a **70:30 ratio** favoring movies.
- **Increase content additions in July and December,** as these months see the highest number of new additions.
- **Leverage popular actors and directors** (e.g., **Anupam Kher Shah Rukh Khan, Julie Tejwani, Rajiv Chilaka, Jan Suter, Raúl Campos**) to enhance engagement.
- **Target Indian audiences** by increasing the production of Indian movies, as India is a key contributor to Netflix content.
- **Maintain an optimal movie length of 80-120 minutes**, as this aligns with the most-watched content duration.
- **Release content on the 1st of each month,** as it has historically been the most frequent release data.
- **Diversify TV Show offerings** with **1-2 season formats**, as most TV Shows on Netflix fall within this range.