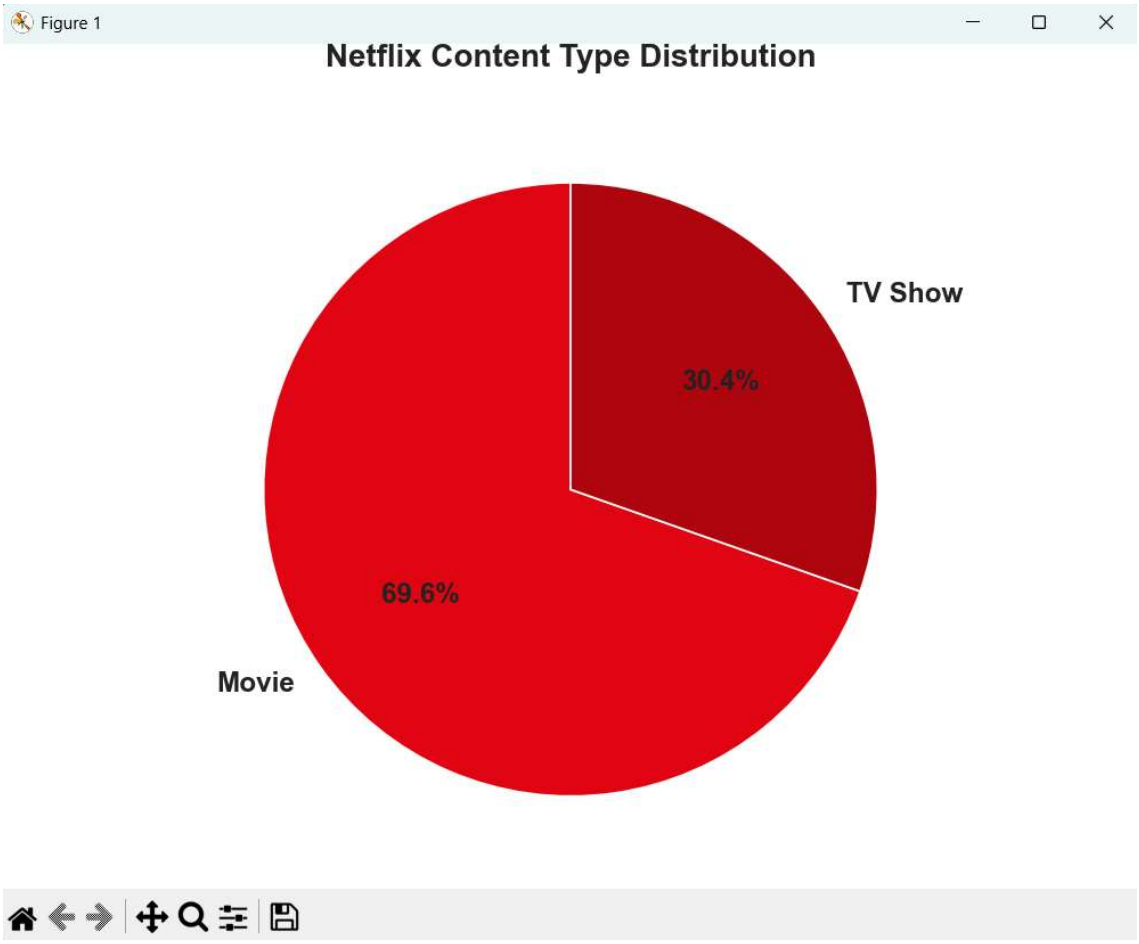
OPERATIONS ON Netflix Movies and TV Shows - CLEANED

The "Netflix Movies and TV Shows" dataset is a tabular dataset with listings for all movies and tv shows on Netflix. Fields include director, title, ratings, release year, duration, and more.

```
# QUESTION 1: Content Type Distribution (Pie Chart)
# ==========================================
# QUESTION: What is the proportion of Movies vs TV Shows in the Netflix catalog?
#
# VALIDATION/JUSTIFICATION:
# This visualization is essential because:
# 1. It provides a quick overview of Netflix's content strategy (movie-focused vs series-focused)
# 2. Pie charts are ideal for showing parts of a whole when there are few categories (2 in this case)
# 3. Percentages help stakeholders understand resource allocation and content balance
# 4. This metric influences content acquisition decisions and user interface design
# 5. Investors and analysts use this to assess Netflix's content diversification strategy

import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Set style for better-looking plots
sns.set_style("darkgrid")
plt.rcParams['figure.figsize'] = (12, 6)

# Step 1: Load the Netflix dataset (robust path handling)
def load_netflix_dataset() -> pd.DataFrame:
    candidate_paths = (
```

```
        '../netflix_titles_CLEANED.csv',
        '../netflix_titles.CLEANED.csv',
        'netflix_titles_CLEANED.csv',
        'netflix_titles.CLEANED.csv',
    )
    last_err = None
    for p in candidate_paths:
        try:
            return pd.read_csv(p)
        except FileNotFoundError as e:
            last_err = e
            continue
    raise last_err

df = load_netflix_dataset()

print("\n" + "="*50)
print("QUESTION 1: Content Type Distribution")
print("="*50)

type_counts = df['type'].value_counts()
print("\nContent Type Counts:")
print(type_counts)

plt.figure(figsize=(8, 8))
colors = ['#E50914', '#B20710']
plt.pie(type_counts.values, labels=type_counts.index,
autopct='%1.1f%%',
        colors=colors, startangle=90, textprops={'fontsize': 14,
'weight': 'bold'})
plt.title('Netflix Content Type Distribution', fontsize=16,
weight='bold', pad=20)
plt.tight_layout()
plt.show()
```



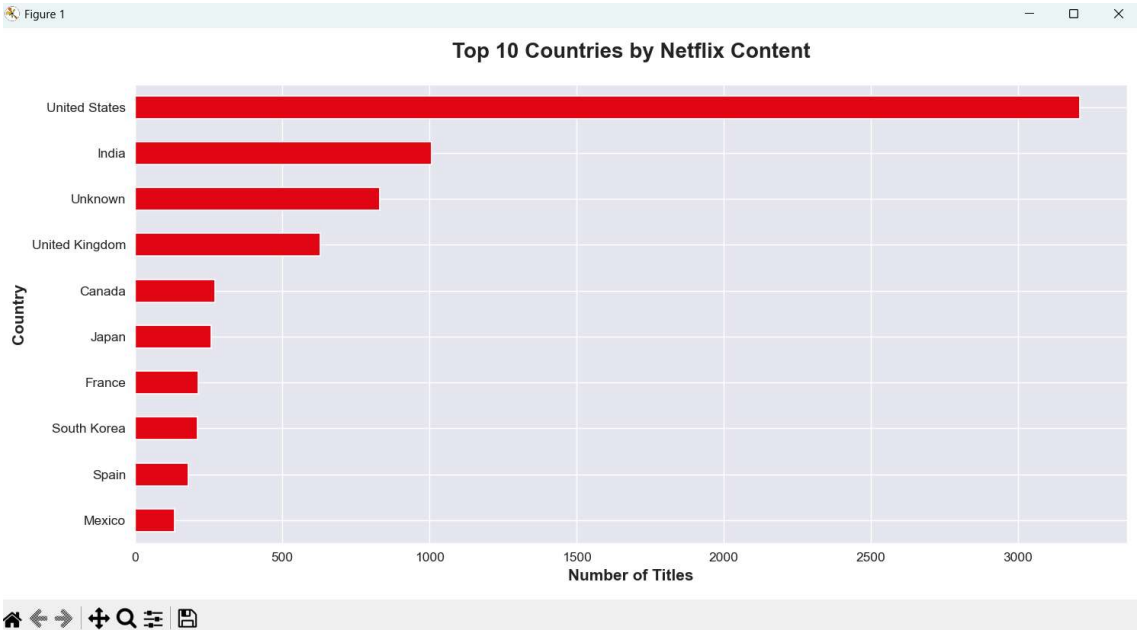
PS D:\lasyafds> python -u "d:\lasyafds\questions\q1_q01.py"

==
QUESTION 1: Content Type Distribution
==

Content Type Counts:
type
Movie   6131
TV Show   2676
Name: count, dtype: int64

```python
# ============================================
# QUESTION 2: Top 10 Countries by Content (Horizontal Bar Chart)
# ============================================
# QUESTION: Which countries produce the most content available on Netflix?
#
# VALIDATION/JUSTIFICATION:
# This analysis is critical because:
# 1. It reveals Netflix's global content sourcing strategy and market focus
# 2. Horizontal bar charts are excellent for comparing categories with long names (country names)
# 3. Top 10 filtering prevents overcrowding while showing the most significant contributors
# 4. This helps identify which regional markets Netflix is investing in most heavily
# 5. Content localization teams can use this to prioritize dubbing and subtitle efforts
# 6. It shows cultural diversity and international expansion patterns
# 7. Regional licensing and production partnerships can be evaluated based on this data

import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

sns.set_style("darkgrid")
plt.rcParams['figure.figsize'] = (12, 6)

def load_netflix_dataset() -> pd.DataFrame:
    candidate_paths = (
        '../netflix_titles_CLEANED.csv',
        '../netflix_titles.CLEANED.csv',
        'netflix_titles_CLEANED.csv',
        'netflix_titles.CLEANED.csv',
    )
    last_err = None
    for p in candidate_paths:
        try:
            return pd.read_csv(p)
        except FileNotFoundError as e:
            last_err = e
            continue
```

```
    raise last_err

df = load_netflix_dataset()

print("\n" + "="*50)
print("QUESTION 2: Top Countries by Content")
print("="*50)

# Extract first country from countries column (some have multiple
countries)
df['primary_country'] =
df['countries'].fillna('Unknown').str.split(',').str[0].str.strip()

country_counts = df['primary_country'].value_counts().head(10)
print("\nTop 10 Countries:")
print(country_counts)

plt.figure(figsize=(12, 6))
country_counts.plot(kind='barh', color='#E50914')
plt.title('Top 10 Countries by Netflix Content', fontsize=16,
weight='bold', pad=20)
plt.xlabel('Number of Titles', fontsize=12, weight='bold')
plt.ylabel('Country', fontsize=12, weight='bold')
plt.gca().invert_yaxis()
plt.tight_layout()
plt.show()
```



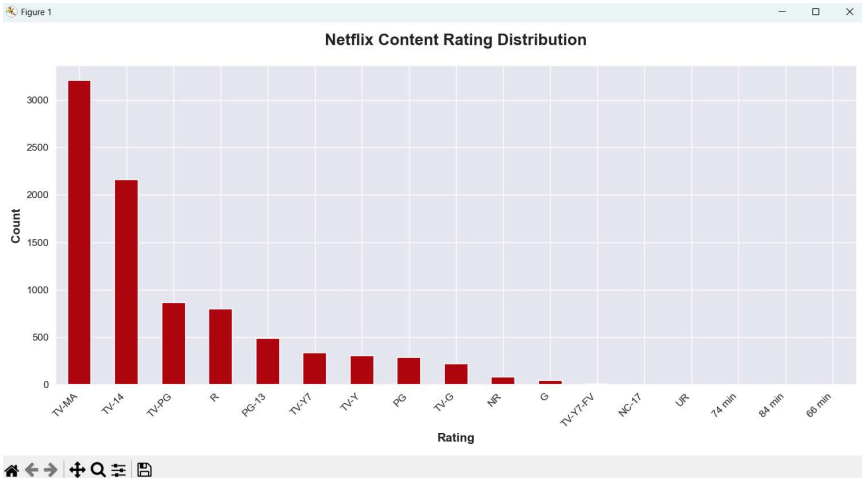
PS D:\lasyafds> python -u "d:\lasyafds\questions\tempCodeRunnerFile.py"

==
QUESTION 2: Top Countries by Content
==

Top 10 Countries:
primary_country
United States 3211
India 1008
Unknown 831
United Kingdom 628
Canada 271
Japan 259
France 213
South Korea 212
Spain 181
Mexico 134
Name: count, dtype: int64

```python
# ============================================
# QUESTION 3: Content Rating Distribution (Bar Chart)
# ============================================
# QUESTION: What is the distribution of content across different maturity ratings?
#
# VALIDATION/JUSTIFICATION:
# This visualization is important because:
# 1. It shows whether Netflix caters more to adults, families, or children
# 2. Rating distribution affects parental control features and content warnings
# 3. Advertisers need this data to determine appropriate ad placements
# 4. Content compliance teams use this to ensure regional rating requirements are met
# 5. Bar charts effectively show frequency distributions across discrete categories
# 6. This helps in understanding target audience demographics
# 7. Family-friendly vs mature content balance is a key business metric
# 8. Different markets have different rating preferences, affecting regional strategies

import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

sns.set_style("darkgrid")
plt.rcParams['figure.figsize'] = (12, 6)

def load_netflix_dataset() -> pd.DataFrame:
    candidate_paths = (
        '../netflix_titles_CLEANED.csv',
        '../netflix_titles.CLEANED.csv',
        'netflix_titles_CLEANED.csv',
        'netflix_titles.CLEANED.csv',
    )
    last_err = None
    for p in candidate_paths:
        try:
            return pd.read_csv(p)
        except FileNotFoundError as e:
            last_err = e
            continue
```

```
    raise last_err

df = load_netflix_dataset()

print("\n" + "="*50)
print("QUESTION 3: Content Rating Distribution")
print("="*50)

rating_counts = df['rating'].value_counts()
print("\nRating Distribution:")
print(rating_counts)

plt.figure(figsize=(12, 6))
rating_counts.plot(kind='bar', color='#B20710')
plt.title('Netflix Content Rating Distribution', fontsize=16,
weight='bold', pad=20)
plt.xlabel('Rating', fontsize=12, weight='bold')
plt.ylabel('Count', fontsize=12, weight='bold')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```



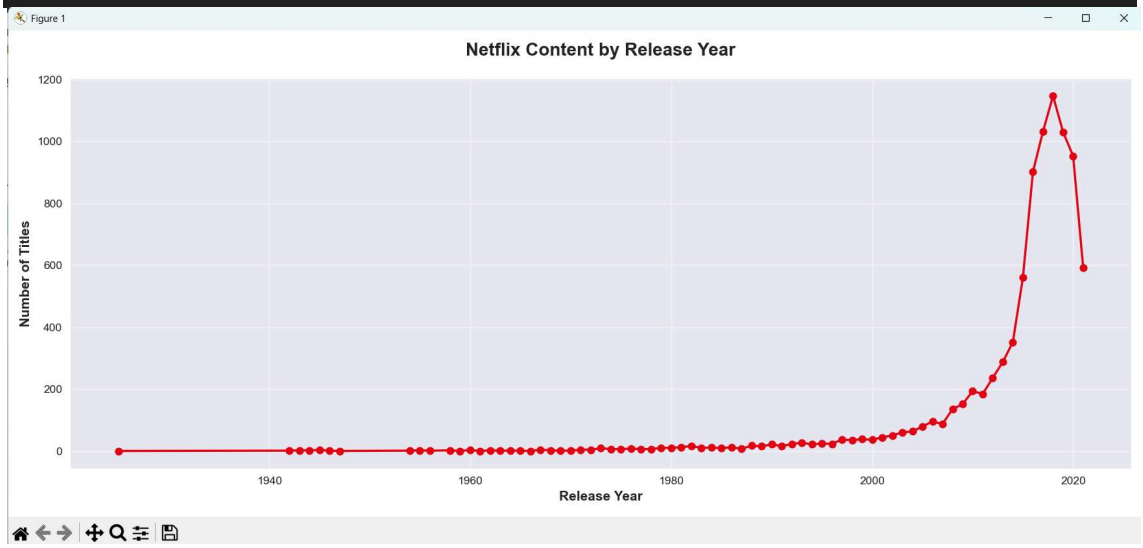
PS D:\lasyafds> python -u "d:\lasyafds\questions\tempCodeRunnerFile.py"

==
QUESTION 3: Content Rating Distribution
==

Rating Distribution:
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

```python
# ============================================
# QUESTION 4: Release Year Trend (Line Chart)
# ============================================
# QUESTION: How has the volume of content production changed over the
years?
#
# VALIDATION/JUSTIFICATION:
# This time-series analysis is valuable because:
# 1. Line charts are optimal for showing trends and patterns over
continuous time periods
# 2. It reveals whether Netflix focuses on newer content or maintains a
classic library
# 3. Spikes or drops in certain years can indicate industry events or
strategic shifts
# 4. Content freshness is a key competitive advantage in streaming
services
# 5. Historical content acquisition patterns help predict future
content needs
# 6. This helps identify the "golden age" of content production for the
platform
# 7. Investors use this to assess whether Netflix is acquiring recent
vs archival content
# 8. Marketing teams can highlight "new releases" vs "classic
collection" campaigns

import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

sns.set_style("darkgrid")
plt.rcParams['figure.figsize'] = (14, 6)

def load_netflix_dataset() -> pd.DataFrame:
    candidate_paths = (
        '../netflix_titles_CLEANED.csv',
        '../netflix_titles.CLEANED.csv',
        'netflix_titles_CLEANED.csv',
        'netflix_titles.CLEANED.csv',
    )
    last_err = None
    for p in candidate_paths:
        try:
            return pd.read_csv(p)
```

```
        except FileNotFoundError as e:
            last_err = e
            continue
    raise last_err

df = load_netflix_dataset()

print("\n" + "="*50)
print("QUESTION 4: Content Release Year Trend")
print("="*50)

year_counts = df['release_year'].value_counts().sort_index()
print("\nContent by Release Year:")
print(year_counts.tail(10))

plt.figure(figsize=(14, 6))
plt.plot(year_counts.index, year_counts.values, marker='o',
         color='#E50914', linewidth=2, markersize=6)
plt.title('Netflix Content by Release Year', fontsize=16,
weight='bold', pad=20)
plt.xlabel('Release Year', fontsize=12, weight='bold')
plt.ylabel('Number of Titles', fontsize=12, weight='bold')
plt.grid(True, alpha=0.3)
plt.tight_layout()
plt.show()
```



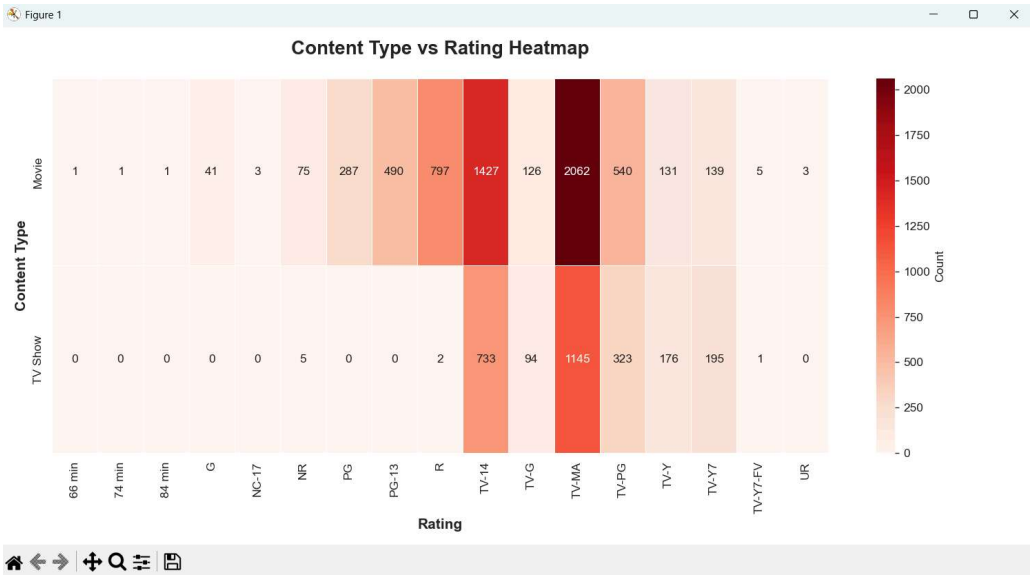
PS D:\lasyafds> python -u "d:\lasyafds\questions\q1_q04.py"

==
QUESTION 4: Content Release Year Trend
==

Content by Release Year:
release_year
2012 237
2013 288
2014 352
2015 560
2016 902
2017 1032
2018 1147
2019 1030
2020 953
2021 592
Name: count, dtype: int64

```python
# ============================================
# QUESTION 5: Heatmap - Content Type by Rating
# ============================================
# QUESTION: How do content ratings vary between Movies and TV Shows?
#
# VALIDATION/JUSTIFICATION:
# This cross-tabulation analysis is powerful because:
# 1. Heatmaps excel at showing relationships between two categorical variables
# 2. Color intensity immediately highlights which combinations are most/least common
# 3. It reveals if movies tend to have different ratings than TV shows
# 4. Content acquisition teams can identify gaps (e.g., lack of PG-rated TV shows)
# 5. This helps in strategic content planning to balance the catalog
# 6. Annotation with exact counts provides precise data alongside visual patterns
# 7. Marketing can tailor campaigns based on rating-type combinations
# 8. This validates whether content guidelines differ between movies and series

import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

sns.set_style("darkgrid")
plt.rcParams['figure.figsize'] = (12, 6)

def load_netflix_dataset() -> pd.DataFrame:
    candidate_paths = (
        '../netflix_titles_CLEANED.csv',
        '../netflix_titles.CLEANED.csv',
        'netflix_titles_CLEANED.csv',
        'netflix_titles.CLEANED.csv',
    )
    last_err = None
    for p in candidate_paths:
        try:
            return pd.read_csv(p)
        except FileNotFoundError as e:
            last_err = e
            continue
    raise last_err
```

```python
df = load_netflix_dataset()

print("\n" + "="*50)
print("QUESTION 5: Content Type vs Rating Heatmap")
print("="*50)

pivot_table = pd.crosstab(df['type'], df['rating'])
print("\nPivot Table:")
print(pivot_table)

plt.figure(figsize=(12, 6))
sns.heatmap(pivot_table, annot=True, fmt='d', cmap='Reds',
            cbar_kws={'label': 'Count'}, linewidths=0.5)
plt.title('Content Type vs Rating Heatmap', fontsize=16, weight='bold',
pad=20)
plt.xlabel('Rating', fontsize=12, weight='bold')
plt.ylabel('Content Type', fontsize=12, weight='bold')
plt.tight_layout()
plt.show()
```

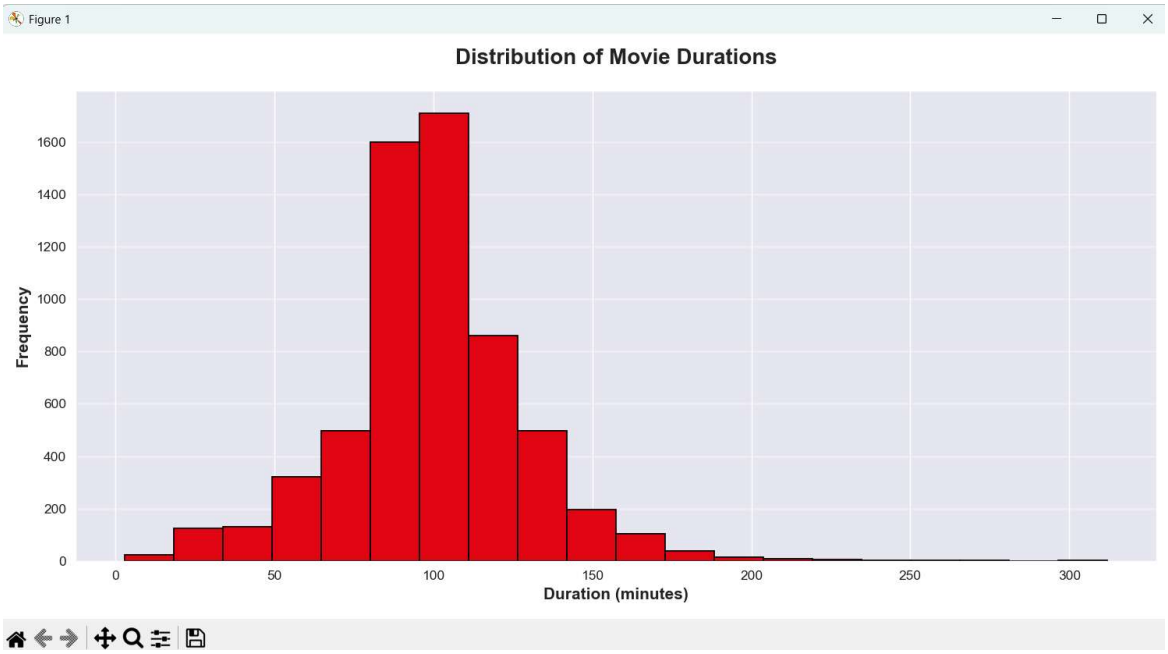



```
PS D:\lasyafds> python -u "d:\lasyafds\questions\tempCodeRunnerFile.py"

==================================================
QUESTION 5: Content Type vs Rating Heatmap
==================================================

Pivot Table:
rating   66 min  74 min  84 min   G  NC-17  NR   PG  PG-13    R  TV-14
TV-G  TV-MA  TV-PG  TV-Y  TV-Y7  TV-Y7-FV  UR
```

```
type
Movie          1       1       1   41       3   75   287    490   797    1427
126    2062    540    131    139          5    3
TV Show        0       0       0    0       0    5     0      0     2     733
94    1145    323    176    195          1    0
```

```
# ============================================
# QUESTION 6: Movie Duration Distribution (Histogram)
# ============================================
# QUESTION: What is the typical length of movies in the Netflix
catalog?
#
# VALIDATION/JUSTIFICATION:
# This distribution analysis is important because:
# 1. Histograms are ideal for showing frequency distributions of
continuous data
# 2. It reveals user preferences - are people watching short, medium,
or long films?
# 3. Content acquisition can target specific duration ranges based on
gaps
# 4. Scheduling and recommendation algorithms use duration as a key
feature
# 5. Understanding duration patterns helps in content categorization
("Quick Watch", "Feature Films")
# 6. Production teams can benchmark their content against industry
standards
# 7. This identifies outliers (very short or very long movies) that may
need special handling
# 8. User session time and engagement metrics correlate strongly with
content duration

import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

sns.set_style("darkgrid")
plt.rcParams['figure.figsize'] = (12, 6)

def load_netflix_dataset() -> pd.DataFrame:
    candidate_paths = (
        '../netflix_titles_CLEANED.csv',
        '../netflix_titles.CLEANED.csv',
        'netflix_titles_CLEANED.csv',
```

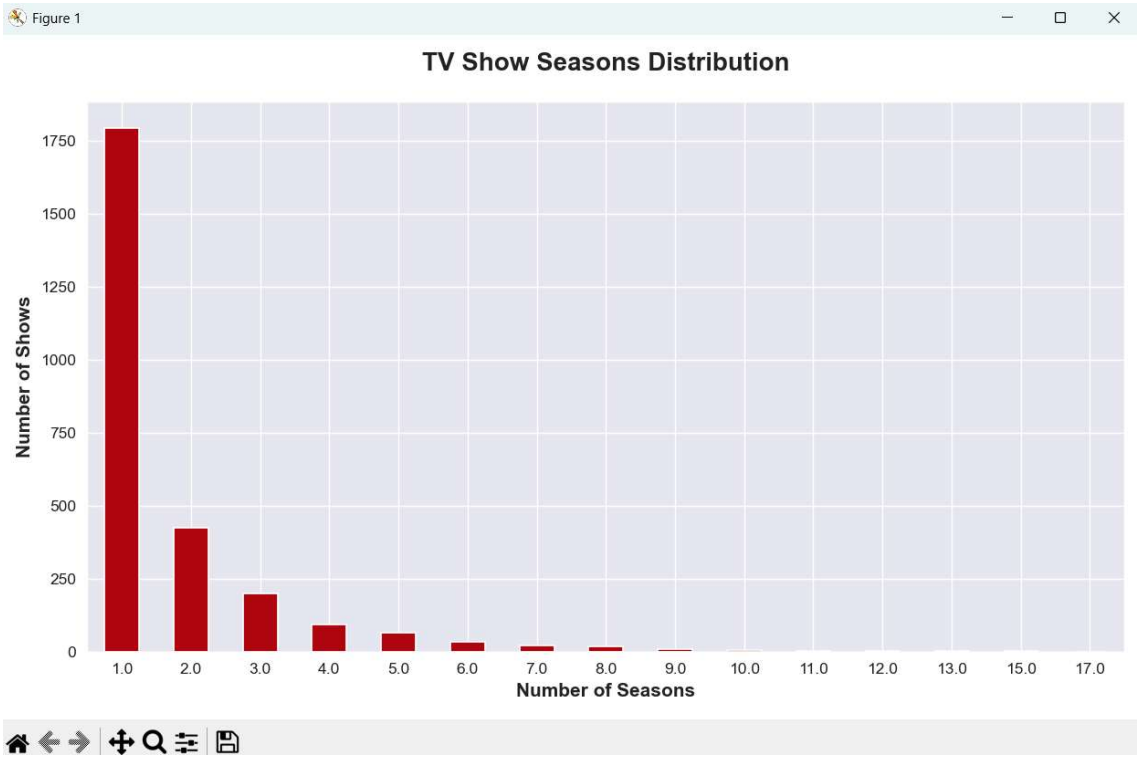
```
        'netflix_titles.CLEANED.csv',
    )
    last_err = None
    for p in candidate_paths:
        try:
            return pd.read_csv(p)
        except FileNotFoundError as e:
            last_err = e
            continue
    raise last_err

df = load_netflix_dataset()

print("\n" + "="*50)
print("QUESTION 6: Movie Duration Distribution")
print("="*50)

movies_df = df[df['type'] == 'Movie'].copy()
movies_df['duration_min'] =
movies_df['duration'].str.extract(r'(\d+)').astype(float)

print("\nMovie Duration Statistics:")
print(movies_df['duration_min'].describe())

plt.figure(figsize=(12, 6))
plt.hist(movies_df['duration_min'].dropna(), bins=20, color='#E50914',
edgecolor='black')
plt.title('Distribution of Movie Durations', fontsize=16,
weight='bold', pad=20)
plt.xlabel('Duration (minutes)', fontsize=12, weight='bold')
plt.ylabel('Frequency', fontsize=12, weight='bold')
plt.grid(True, alpha=0.3, axis='y')
plt.tight_layout()
plt.show()
```




```
PS D:\lasyafds> python -u "d:\lasyafds\questions\tempCodeRunnerFile.py"

==================================================
QUESTION 6: Movie Duration Distribution
==================================================

Movie Duration Statistics:
count    6128.000000
mean       99.577187
std        28.290593
min         3.000000
25%        87.000000
50%        98.000000
75%       114.000000
max       312.000000
Name: duration_min, dtype: float64
```

```
# ============================================
# QUESTION 7: TV Show Seasons Distribution (Bar Chart)
# ============================================
# QUESTION: How many seasons do most TV shows have on Netflix?
#
# VALIDATION/JUSTIFICATION:
# This analysis is crucial because:
# 1. It shows whether Netflix favors limited series or long-running
shows
# 2. Bar charts clearly show discrete season counts and their
frequencies
# 3. Multi-season shows indicate higher production investment and user
engagement
# 4. Single-season shows might be mini-series, cancelled shows, or
ongoing series
# 5. Content strategy differs for shows with different season counts
(binge vs episodic)
# 6. This helps predict future content pipeline and renewal decisions
# 7. Longer shows provide more "stickiness" - users invest more time in
multi-season series
# 8. Production costs scale with season count, affecting budget
allocation
# 9. Recommendation algorithms treat single vs multi-season shows
differently

import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

sns.set_style("darkgrid")
plt.rcParams['figure.figsize'] = (10, 6)

def load_netflix_dataset() -> pd.DataFrame:
    candidate_paths = (
        '../netflix_titles_CLEANED.csv',
        '../netflix_titles.CLEANED.csv',
        'netflix_titles_CLEANED.csv',
        'netflix_titles.CLEANED.csv',
    )
    last_err = None
    for p in candidate_paths:
        try:
            return pd.read_csv(p)
```

```
        except FileNotFoundError as e:
            last_err = e
            continue
    raise last_err

df = load_netflix_dataset()

print("\n" + "="*50)
print("QUESTION 7: TV Show Seasons Distribution")
print("="*50)

tv_shows_df = df[df['type'] == 'TV Show'].copy()
tv_shows_df['num_seasons'] =
tv_shows_df['duration'].str.extract(r'(\d+)').astype(float)

season_counts = tv_shows_df['num_seasons'].value_counts().sort_index()
print("\nSeasons Distribution:")
print(season_counts)

plt.figure(figsize=(10, 6))
season_counts.plot(kind='bar', color='#B20710')
plt.title('TV Show Seasons Distribution', fontsize=16, weight='bold',
pad=20)
plt.xlabel('Number of Seasons', fontsize=12, weight='bold')
plt.ylabel('Number of Shows', fontsize=12, weight='bold')
plt.xticks(rotation=0)
plt.tight_layout()
plt.show()
```

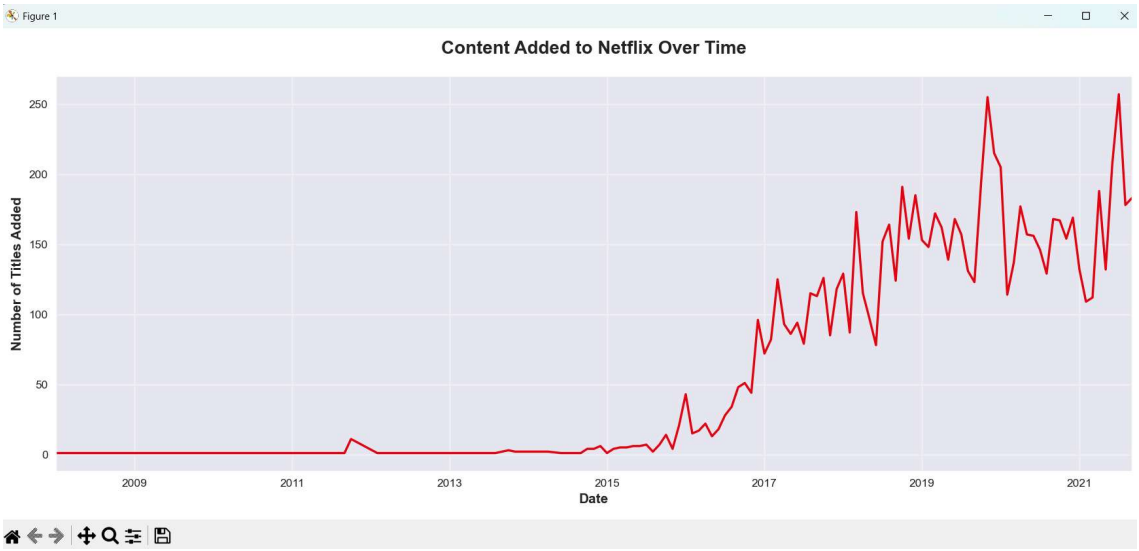



```
PS D:\lasyafds> python -u "d:\lasyafds\questions\tempCodeRunnerFile.py"

==================================================
QUESTION 7: TV Show Seasons Distribution
==================================================

Seasons Distribution:
num_seasons
1.0     1793
2.0      425
3.0      199
4.0       95
5.0       65
6.0       33
7.0       23
8.0       17
9.0        9
10.0       7
11.0       2
12.0       2
13.0       3
15.0       2
17.0       1
Name: count, dtype: int64
```

```
# ================================================
```

```python
# QUESTION 8: Content Added Over Time (Line Chart)
# ================================================
# QUESTION: At what rate is Netflix adding new content to its platform over time?
#
# VALIDATION/JUSTIFICATION:
# This time-series analysis is essential because:
# 1. Line charts excel at showing trends and growth patterns over time
# 2. It reveals Netflix's content acquisition velocity and strategic timing
# 3. Spikes might correspond to major releases, seasonal trends, or strategic launches
# 4. Investors track content addition rate as a key performance indicator
# 5. Competitive analysis: comparing growth rate with other streaming platforms
# 6. This helps predict infrastructure needs (storage, bandwidth, servers)
# 7. Marketing teams can align campaigns with content drop patterns
# 8. Monthly/yearly patterns reveal content licensing cycles and renewal schedules
# 9. Declining trends might indicate market saturation or strategic pivots
# 10. This validates whether Netflix is maintaining its promised content refresh rate

import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

sns.set_style("darkgrid")
plt.rcParams['figure.figsize'] = (14, 6)

def load_netflix_dataset() -> pd.DataFrame:
    candidate_paths = (
        '../netflix_titles_CLEANED.csv',
        '../netflix_titles.CLEANED.csv',
        'netflix_titles_CLEANED.csv',
        'netflix_titles.CLEANED.csv',
    )
    last_err = None
    for p in candidate_paths:
        try:
```

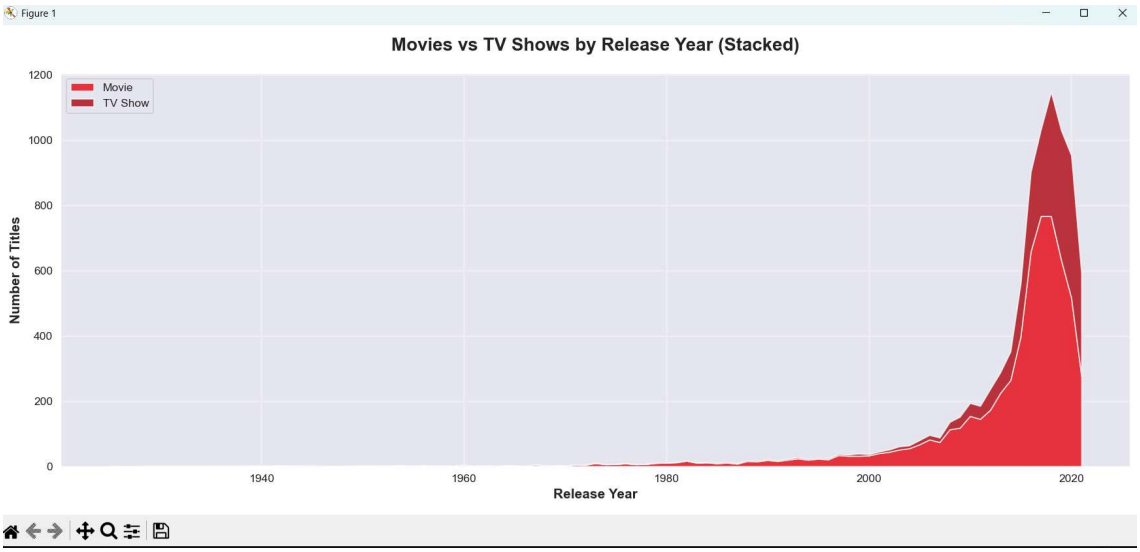
```
            return pd.read_csv(p)
        except FileNotFoundError as e:
            last_err = e
            continue
    raise last_err

df = load_netflix_dataset()

print("\n" + "="*50)
print("QUESTION 8: Content Added to Netflix Over Time")
print("="*50)

df['date_added'] = pd.to_datetime(df['date_added'], errors='coerce')
df['month_added'] = df['date_added'].dt.to_period('M')

monthly_additions = df.groupby('month_added').size().sort_index()

print("\nMonthly Content Additions:")
print(monthly_additions.tail(10))

plt.figure(figsize=(14, 6))
monthly_additions.plot(color='#E50914', linewidth=2)
plt.title('Content Added to Netflix Over Time', fontsize=16,
weight='bold', pad=20)
plt.xlabel('Date', fontsize=12, weight='bold')
plt.ylabel('Number of Titles Added', fontsize=12, weight='bold')
plt.grid(True, alpha=0.3)
plt.tight_layout()
plt.show()
```




```
PS D:\lasyafds> python -u "d:\lasyafds\questions\tempCodeRunnerFile.py"

==================================================
QUESTION 8: Content Added to Netflix Over Time
==================================================

Monthly Content Additions:
month_added
2020-12    169
2021-01    132
2021-02    109
2021-03    112
2021-04    188
2021-05    132
2021-06    207
2021-07    257
2021-08    178
2021-09    183
Freq: M, dtype: int64
```

```python
# ============================================
# QUESTION 9: Movies vs TV Shows by Year (Stacked Area Chart)
# ============================================
# QUESTION: How has the balance between Movies and TV Shows evolved
over release years?
#
# VALIDATION/JUSTIFICATION:
# This visualization is valuable because:
# 1. Stacked area charts show both total volume and composition changes
over time
# 2. It reveals strategic shifts in content type preferences across
different eras
# 3. This helps understand if Netflix is pivoting toward original
series vs films
# 4. Historical trends inform future content production roadmaps
# 5. Different decades had different production norms (more movies in
90s, more series recently)
# 6. Investment allocation between movie rights and TV show licenses
can be evaluated
# 7. This shows how streaming has influenced content production
patterns

import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

sns.set_style("darkgrid")
plt.rcParams['figure.figsize'] = (14, 6)

def load_netflix_dataset() -> pd.DataFrame:
    candidate_paths = (
        '../netflix_titles_CLEANED.csv',
        '../netflix_titles.CLEANED.csv',
        'netflix_titles_CLEANED.csv',
        'netflix_titles.CLEANED.csv',
    )
    last_err = None
    for p in candidate_paths:
        try:
            return pd.read_csv(p)
        except FileNotFoundError as e:
            last_err = e
            continue
```

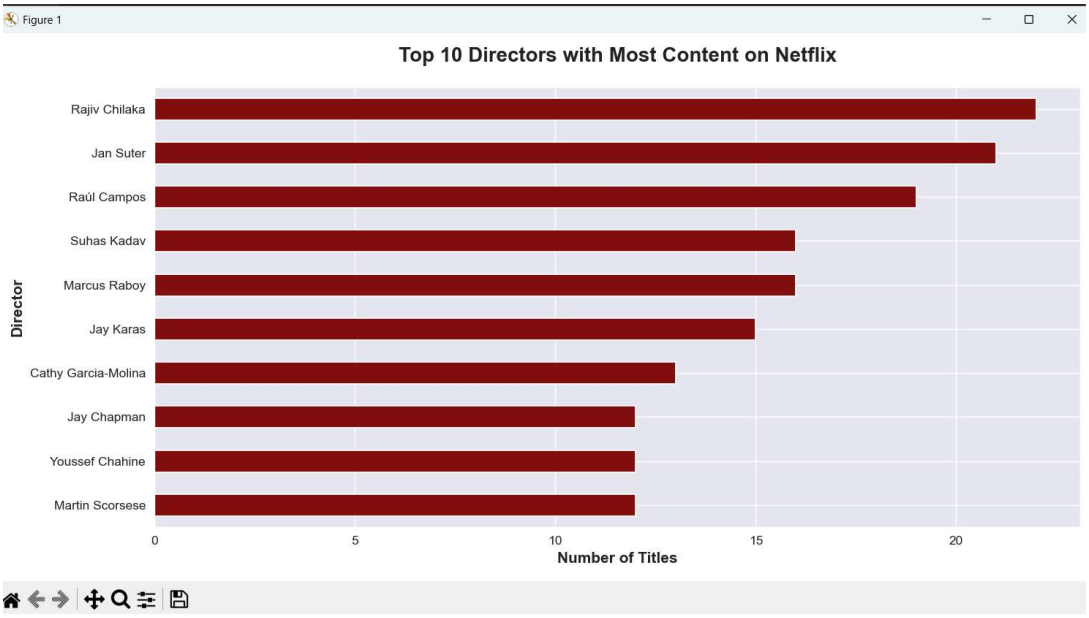
```
    raise last_err

df = load_netflix_dataset()

print("\n" + "="*50)
print("QUESTION 9: Movies vs TV Shows Over Release Years")
print("="*50)

year_type = df.groupby(['release_year',
'type']).size().unstack(fill_value=0)
# Ensure both columns exist for stackplot
year_type = year_type.reindex(columns=['Movie', 'TV Show'],
fill_value=0)
print("\nContent Type by Year:")
print(year_type.tail(10))

plt.figure(figsize=(14, 6))
plt.stackplot(year_type.index, year_type['Movie'], year_type['TV
Show'],
              labels=['Movie', 'TV Show'], colors=['#E50914',
'#B20710'], alpha=0.8)
plt.title('Movies vs TV Shows by Release Year (Stacked)', fontsize=16,
weight='bold', pad=20)
plt.xlabel('Release Year', fontsize=12, weight='bold')
plt.ylabel('Number of Titles', fontsize=12, weight='bold')
plt.legend(loc='upper left')
plt.grid(True, alpha=0.3)
plt.tight_layout()
plt.show()
```




```
PS D:\lasyafds> python -u "d:\lasyafds\questions\tempCodeRunnerFile.py"

==================================================
QUESTION 9: Movies vs TV Shows Over Release Years
==================================================

Content Type by Year:
type          Movie  TV Show
release_year
2012            173       64
2013            225       63
2014            264       88
2015            398      162
2016            658      244
2017            767      265
2018            767      380
2019            633      397
2020            517      436
2021            277      315
```

```python
# ============================================
# QUESTION 10: Top 10 Directors by Content Count (Bar Chart)
# ============================================
# QUESTION: Which directors have the most content on Netflix?
#
# VALIDATION/JUSTIFICATION:
# This analysis is important because:
# 1. Identifies key content creators and potential partnership
opportunities
# 2. Bar charts effectively rank and compare discrete entities
# 3. High-volume directors might have exclusive deals or strong
relationships with Netflix
# 4. This data supports talent acquisition and retention strategies
# 5. Marketing can leverage popular directors for promotional campaigns
# 6. Understanding director portfolios helps in content curation and
collections
# 7. Fans of specific directors can be targeted with personalized
recommendations
# 8. This reveals Netflix's investment in auteur-driven vs commercial
content

import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

sns.set_style("darkgrid")
plt.rcParams['figure.figsize'] = (12, 6)

def load_netflix_dataset() -> pd.DataFrame:
    candidate_paths = (
        '../netflix_titles_CLEANED.csv',
        '../netflix_titles.CLEANED.csv',
        'netflix_titles_CLEANED.csv',
        'netflix_titles.CLEANED.csv',
    )
    last_err = None
    for p in candidate_paths:
        try:
            return pd.read_csv(p)
        except FileNotFoundError as e:
            last_err = e
            continue
    raise last_err
```

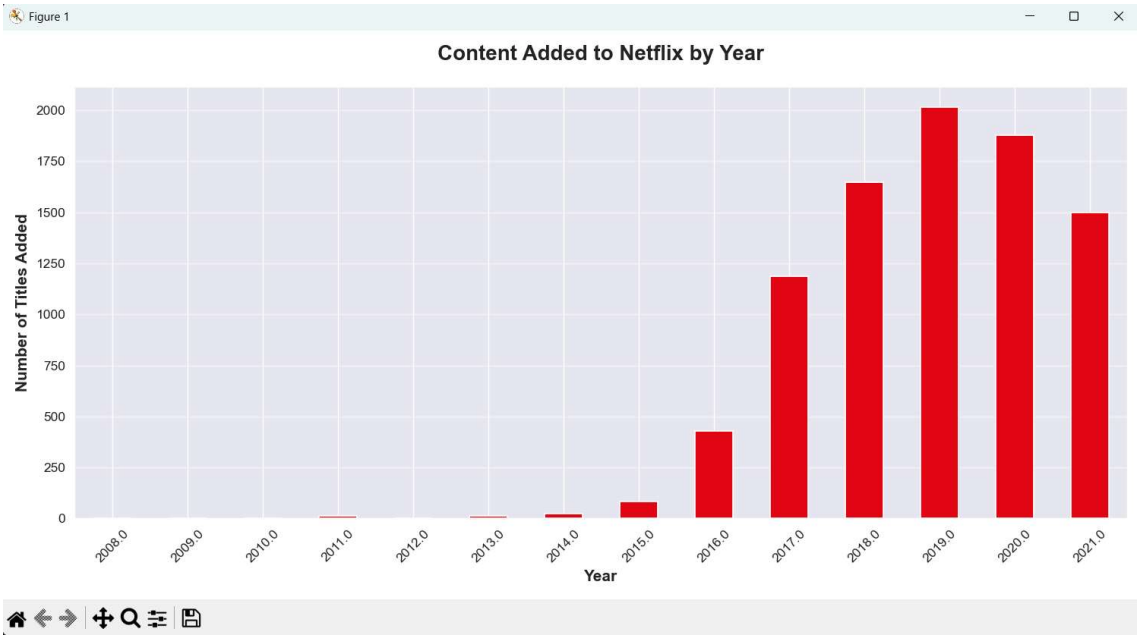
```
df = load_netflix_dataset()

print("\n" + "="*50)
print("QUESTION 10: Top 10 Directors by Content Count")
print("="*50)

directors_list =
df['directors'].dropna().str.split(',').explode().str.strip()
director_counts = directors_list.value_counts().head(10)

print("\nTop 10 Directors:")
print(director_counts)

plt.figure(figsize=(12, 6))
director_counts.plot(kind='barh', color='#831010')
plt.title('Top 10 Directors with Most Content on Netflix', fontsize=16,
weight='bold', pad=20)
plt.xlabel('Number of Titles', fontsize=12, weight='bold')
plt.ylabel('Director', fontsize=12, weight='bold')
plt.gca().invert_yaxis()
plt.tight_layout()
plt.show()
```




```
PS D:\lasyafds> python -u "d:\lasyafds\questions\q1_q10.py"

======================================================
QUESTION 10: Top 10 Directors by Content Count
======================================================

Top 10 Directors:
directors
Rajiv Chilaka          22
Jan Suter              21
Raúl Campos            19
Suhas Kadav            16
Marcus Raboy           16
Jay Karas              15
Cathy Garcia-Molina    13
Jay Chapman            12
Youssef Chahine        12
Martin Scorsese        12
Name: count, dtype: int64
```

```
# ============================================
# QUESTION 11: Content Addition by Year (Bar Chart)
# ============================================
# QUESTION: In which years did Netflix add the most content to its
platform?
#
# VALIDATION/JUSTIFICATION:
# This analysis is critical because:
# 1. Bar charts clearly show year-over-year comparisons of content
additions
# 2. Peak years indicate major content acquisition initiatives or
market expansion
# 3. Declining years might signal market maturity or strategic content
pruning
# 4. This correlates with Netflix's subscriber growth and competitive
landscape
# 5. Content budgets and licensing deals can be inferred from addition
patterns
# 6. Regulatory changes or competitive pressures show up as
year-over-year changes
# 7. This validates Netflix's "content is king" strategy execution over
time

import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

sns.set_style("darkgrid")
plt.rcParams['figure.figsize'] = (12, 6)

def load_netflix_dataset() -> pd.DataFrame:
    candidate_paths = (
        '../netflix_titles_CLEANED.csv',
        '../netflix_titles.CLEANED.csv',
        'netflix_titles_CLEANED.csv',
        'netflix_titles.CLEANED.csv',
    )
    last_err = None
    for p in candidate_paths:
        try:
            return pd.read_csv(p)
        except FileNotFoundError as e:
            last_err = e
```

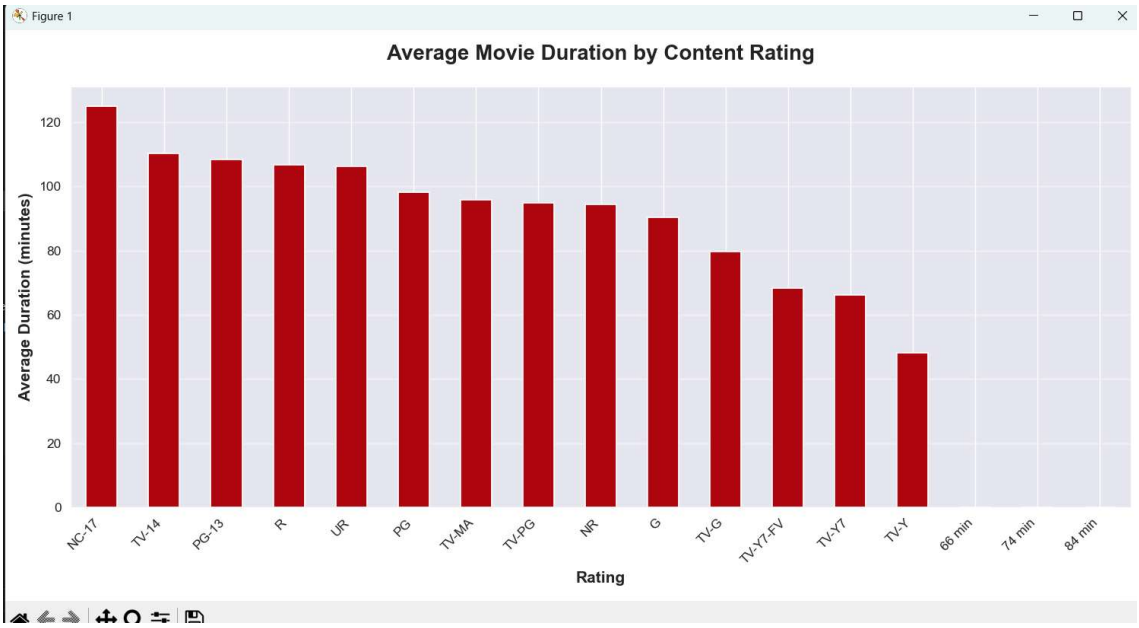
```
            continue
    raise last_err

df = load_netflix_dataset()

print("\n" + "="*50)
print("QUESTION 11: Content Added by Year")
print("="*50)

df['date_added'] = pd.to_datetime(df['date_added'], errors='coerce')
df['year_added'] = df['date_added'].dt.year

yearly_additions = df['year_added'].value_counts().sort_index()
print("\nContent Added per Year:")
print(yearly_additions)

plt.figure(figsize=(12, 6))
yearly_additions.plot(kind='bar', color='#E50914')
plt.title('Content Added to Netflix by Year', fontsize=16,
weight='bold', pad=20)
plt.xlabel('Year', fontsize=12, weight='bold')
plt.ylabel('Number of Titles Added', fontsize=12, weight='bold')
plt.xticks(rotation=45)
plt.grid(True, alpha=0.3, axis='y')
plt.tight_layout()
plt.show()
```




```
PS D:\lasyafds> python -u "d:\lasyafds\questions\q1_q11.py"

==================================================
QUESTION 11: Content Added by Year
==================================================

Content Added per Year:
year_added
2008.0       2
2009.0       2
2010.0       1
2011.0      13
2012.0       3
2013.0      11
2014.0      24
2015.0      82
2016.0     429
2017.0    1188
2018.0    1649
2019.0    2016
2020.0    1879
2021.0    1498
Name: count, dtype: int64
PS D:\lasyafds>
```

```
# ============================================
# QUESTION 12: Average Movie Duration by Rating (Bar Chart)
# ============================================
# QUESTION: Does movie runtime vary by content rating?
#
# VALIDATION/JUSTIFICATION:
# This analysis reveals important patterns:
# 1. Grouped bar charts show average values across categories
effectively
# 2. Family content (PG, PG-13) might have different runtime norms than
mature content (R, TV-MA)
# 3. This helps content creators understand industry standards for
different audiences
# 4. Programming blocks and scheduling decisions depend on content
length by rating
# 5. Children's content typically has shorter runtimes than adult
content
# 6. This validates whether Netflix's catalog aligns with typical
rating-duration relationships
# 7. User session planning differs for short family movies vs long
adult dramas

import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

sns.set_style("darkgrid")
plt.rcParams['figure.figsize'] = (12, 6)

def load_netflix_dataset() -> pd.DataFrame:
    candidate_paths = (
        '../netflix_titles_CLEANED.csv',
        '../netflix_titles.CLEANED.csv',
        'netflix_titles_CLEANED.csv',
        'netflix_titles.CLEANED.csv',
    )
    last_err = None
    for p in candidate_paths:
        try:
            return pd.read_csv(p)
        except FileNotFoundError as e:
```

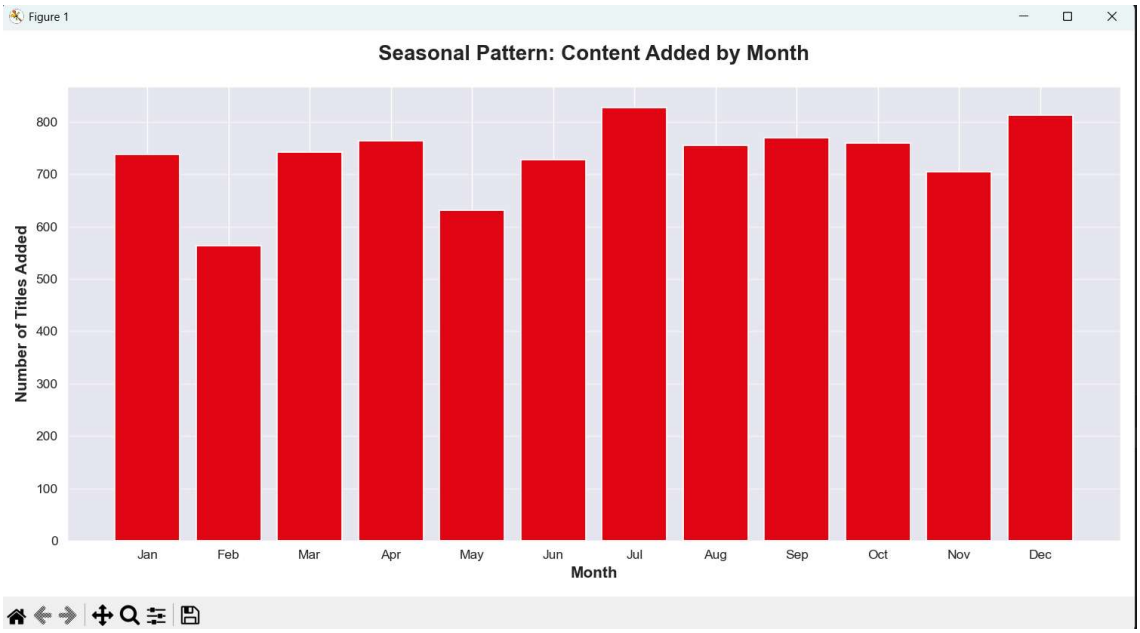
```
            last_err = e
            continue
    raise last_err

df = load_netflix_dataset()

print("\n" + "="*50)
print("QUESTION 12: Average Movie Duration by Rating")
print("="*50)

movies_df = df[df['type'] == 'Movie'].copy()
movies_df['duration_min'] =
movies_df['duration'].str.extract(r'(\d+)').astype(float)

avg_duration_by_rating =
movies_df.groupby('rating')['duration_min'].mean().sort_values(ascendin
g=False)
print("\nAverage Duration by Rating:")
print(avg_duration_by_rating)

plt.figure(figsize=(12, 6))
avg_duration_by_rating.plot(kind='bar', color='#B20710')
plt.title('Average Movie Duration by Content Rating', fontsize=16,
weight='bold', pad=20)
plt.xlabel('Rating', fontsize=12, weight='bold')
plt.ylabel('Average Duration (minutes)', fontsize=12, weight='bold')
plt.xticks(rotation=45, ha='right')
plt.grid(True, alpha=0.3, axis='y')
plt.tight_layout()
plt.show()
```




```
PS D:\lasyafds> python -u "d:\lasyafds\questions\tempCodeRunnerFile.py"

==================================================
QUESTION 12: Average Movie Duration by Rating
==================================================

Average Duration by Rating:
rating
NC-17       125.000000
TV-14       110.290820
PG-13       108.330612
R           106.720201
UR          106.333333
PG           98.282230
TV-MA        95.889913
TV-PG        94.851852
NR           94.533333
G            90.268293
TV-G         79.666667
TV-Y7-FV     68.400000
TV-Y7        66.287770
TV-Y         48.114504
66 min             NaN
74 min             NaN
84 min             NaN
Name: duration_min, dtype: float64
PS D:\lasyafds>
```

```python
# ============================================
# QUESTION 13: Content Addition by Month of Year (Bar Chart)
# ============================================
# QUESTION: Are there seasonal patterns in when Netflix adds content?
#
# VALIDATION/JUSTIFICATION:
# This seasonal analysis is valuable because:
# 1. Identifies if Netflix has strategic content drop periods (e.g., holiday seasons)
# 2. Bar charts effectively show cyclical patterns across 12 months
# 3. Content marketing campaigns can be planned around high-addition months
# 4. Competitor analysis: are there industry-wide seasonal patterns?
# 5. Licensing renewals and expirations often follow calendar patterns
# 6. Major entertainment events (awards season, summer blockbusters) influence timing
# 7. This helps predict future content pipeline and manage subscriber expectations
# 8. Resource allocation (QA, localization teams) can be planned based on seasonal peaks

import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

sns.set_style("darkgrid")
plt.rcParams['figure.figsize'] = (12, 6)

def load_netflix_dataset() -> pd.DataFrame:
    candidate_paths = (
        '../netflix_titles_CLEANED.csv',
        '../netflix_titles.CLEANED.csv',
        'netflix_titles_CLEANED.csv',
        'netflix_titles.CLEANED.csv',
    )
    last_err = None
    for p in candidate_paths:
        try:
            return pd.read_csv(p)
        except FileNotFoundError as e:
            last_err = e
            continue
    raise last_err
```

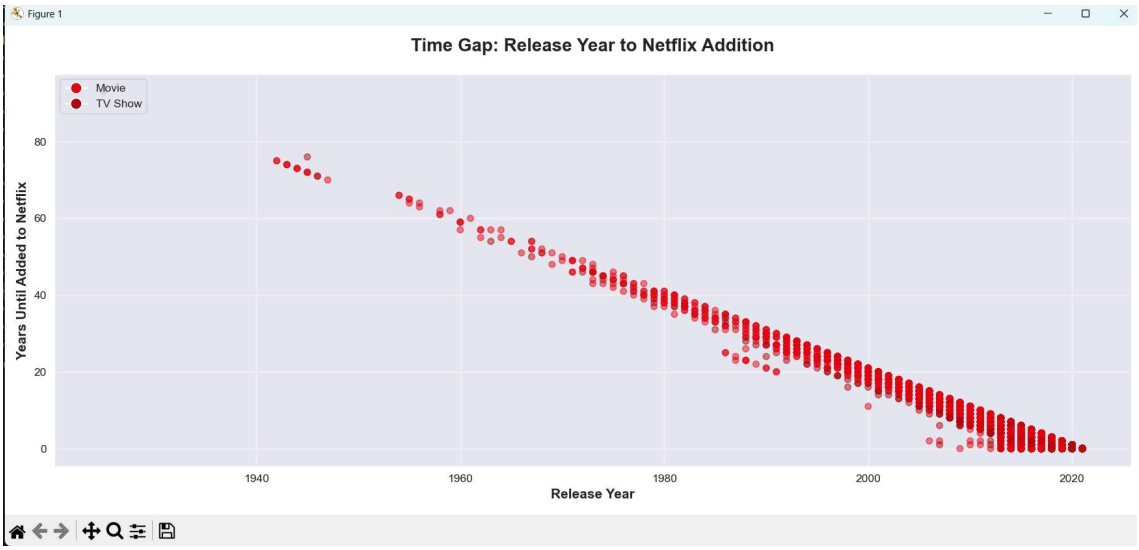
```
df = load_netflix_dataset()

print("\n" + "="*50)
print("QUESTION 13: Content Addition by Month")
print("="*50)

df['date_added'] = pd.to_datetime(df['date_added'], errors='coerce')
df['month_number'] = df['date_added'].dt.month
monthly_pattern = df['month_number'].value_counts().sort_index()
month_names = ['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun', 'Jul', 'Aug',
'Sep', 'Oct', 'Nov', 'Dec']

print("\nContent Added by Month:")
print(monthly_pattern)

plt.figure(figsize=(12, 6))
plt.bar(range(1, 13), [monthly_pattern.get(i, 0) for i in range(1,
13)], color='#E50914')
plt.title('Seasonal Pattern: Content Added by Month', fontsize=16,
weight='bold', pad=20)
plt.xlabel('Month', fontsize=12, weight='bold')
plt.ylabel('Number of Titles Added', fontsize=12, weight='bold')
plt.xticks(range(1, 13), month_names)
plt.grid(True, alpha=0.3, axis='y')
plt.tight_layout()
plt.show()
```




```
PS D:\lasyafds> python -u "d:\lasyafds\questions\tempCodeRunnerFile.py"

==================================================
QUESTION 13: Content Addition by Month
==================================================

Content Added by Month:
month_number
1.0     738
2.0     563
3.0     742
4.0     764
5.0     632
6.0     728
7.0     827
8.0     755
9.0     770
10.0    760
11.0    705
12.0    813
Name: count, dtype: int64
```

```
# ============================================
# QUESTION 14: Release Year vs Date Added Gap (Scatter Plot)
# ============================================
# QUESTION: How long after release does content typically get added to
Netflix?
#
# VALIDATION/JUSTIFICATION:
# This time-lag analysis is insightful because:
# 1. Scatter plots show relationships between two continuous variables
effectively
# 2. It reveals if Netflix acquires fresh content or relies on
catalog/archive content
# 3. Short gaps indicate aggressive recent content acquisition or
original productions
# 4. Long gaps suggest catalog content or classic movie collections
# 5. This informs licensing strategy and production vs acquisition
balance
# 6. Windowing strategies (theatrical → streaming → TV) affect
acquisition timing
# 7. Competitive advantage: shorter gaps mean fresher content for
subscribers
# 8. Different gaps for movies vs TV shows indicate different content
strategies

import pandas as pd
import matplotlib.pyplot as plt
from matplotlib.lines import Line2D
import seaborn as sns

sns.set_style("darkgrid")
plt.rcParams['figure.figsize'] = (14, 6)

def load_netflix_dataset() -> pd.DataFrame:
    candidate_paths = (
        '../netflix_titles_CLEANED.csv',
        '../netflix_titles.CLEANED.csv',
        'netflix_titles_CLEANED.csv',
        'netflix_titles.CLEANED.csv',
    )
    last_err = None
    for p in candidate_paths:
        try:
            return pd.read_csv(p)
```

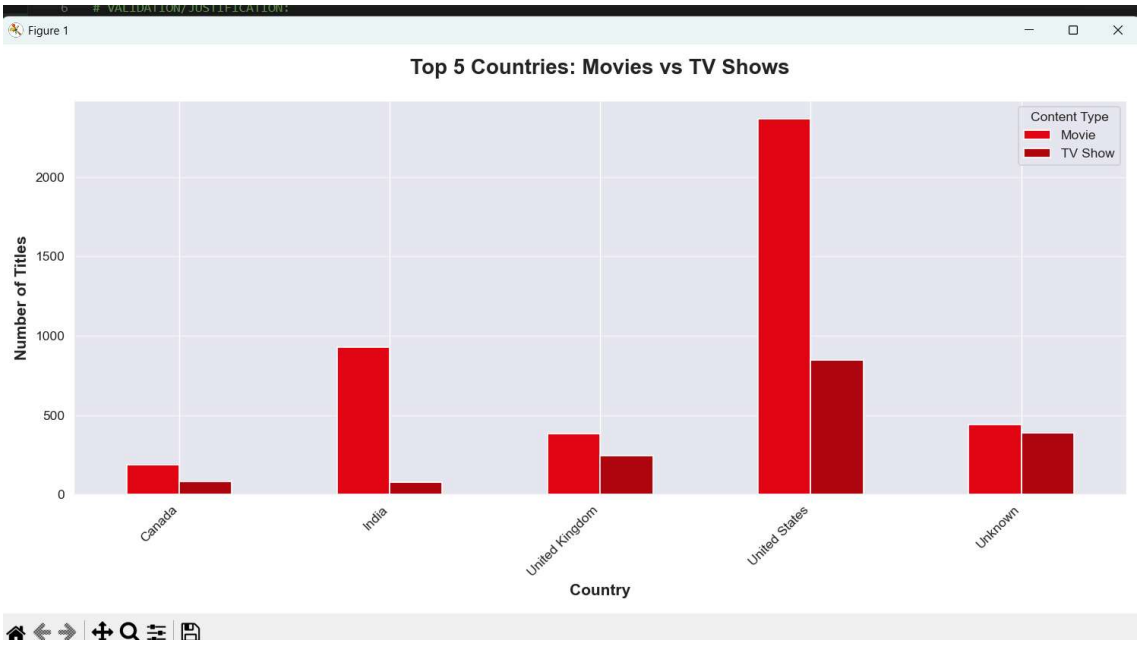
```
        except FileNotFoundError as e:
            last_err = e
            continue
    raise last_err

df = load_netflix_dataset()

print("\n" + "="*50)
print("QUESTION 14: Release Year vs Addition Date Gap")
print("="*50)

df['date_added'] = pd.to_datetime(df['date_added'], errors='coerce')
df['year_added'] = df['date_added'].dt.year
df['year_gap'] = df['year_added'] - df['release_year']
df_gap = df[df['year_gap'].notna() & (df['year_gap'] >= 0)].copy()

print("\nYear Gap Statistics:")
print(df_gap['year_gap'].describe())

plt.figure(figsize=(14, 6))
plt.scatter(df_gap['release_year'], df_gap['year_gap'],
            c=df_gap['type'].map({'Movie': '#E50914', 'TV Show':
'#B20710'}),
            alpha=0.5, s=30)
plt.title('Time Gap: Release Year to Netflix Addition', fontsize=16,
weight='bold', pad=20)
plt.xlabel('Release Year', fontsize=12, weight='bold')
plt.ylabel('Years Until Added to Netflix', fontsize=12, weight='bold')
plt.grid(True, alpha=0.3)
legend_elements = [Line2D([0], [0], marker='o', color='w',
markerfacecolor='#E50914',
                          markersize=10, label='Movie'),
                   Line2D([0], [0], marker='o', color='w',
markerfacecolor='#B20710',
                          markersize=10, label='TV Show')]
plt.legend(handles=legend_elements, loc='upper left')
plt.tight_layout()
plt.show()
```




```
PS D:\lasyafds> python -u "d:\lasyafds\questions\tempCodeRunnerFile.py"

==================================================
QUESTION 14: Release Year vs Addition Date Gap
==================================================

Year Gap Statistics:
count    8783.000000
mean        4.697825
std         8.790808
min         0.000000
25%         0.000000
50%         1.000000
75%         5.000000
max        93.000000
Name: year_gap, dtype: float64
```

```python
# ============================================
# QUESTION 15: Top 5 Countries - Content Type Breakdown (Grouped Bar Chart)
# ============================================
# QUESTION: How does the Movie/TV Show split differ across top content-producing countries?
#
# VALIDATION/JUSTIFICATION:
# This comparative analysis is powerful because:
# 1. Grouped bar charts excel at comparing subcategories across main categories
# 2. Regional content preferences and production capabilities vary significantly
# 3. Some countries specialize in films (Bollywood) while others in series (US TV industry)
# 4. This informs regional partnership strategies and content acquisition focus
# 5. Cultural differences in storytelling formats (episodic vs feature-length) are revealed
# 6. Market-specific content strategies can be developed based on regional strengths
# 7. This validates whether Netflix's regional content mix matches local production trends
# 8. Investment priorities can be adjusted per region based on content type strengths

import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

sns.set_style("darkgrid")
plt.rcParams['figure.figsize'] = (12, 6)

def load_netflix_dataset() -> pd.DataFrame:
    candidate_paths = (
        '../netflix_titles_CLEANED.csv',
        '../netflix_titles.CLEANED.csv',
        'netflix_titles_CLEANED.csv',
        'netflix_titles.CLEANED.csv',
    )
    last_err = None
    for p in candidate_paths:
        try:
```

```
            return pd.read_csv(p)
        except FileNotFoundError as e:
            last_err = e
            continue
    raise last_err

df = load_netflix_dataset()

# Precompute primary country for this analysis
df['primary_country'] =
df['countries'].fillna('Unknown').str.split(',').str[0].str.strip()

print("\n" + "="*50)
print("QUESTION 15: Top 5 Countries - Content Type Breakdown")
print("="*50)

top5_countries = df['primary_country'].value_counts().head(5).index
top5_df = df[df['primary_country'].isin(top5_countries)]
country_type_pivot = pd.crosstab(top5_df['primary_country'],
top5_df['type'])

print("\nTop 5 Countries Content Breakdown:")
print(country_type_pivot)

ax = country_type_pivot.plot(kind='bar', color=['#E50914', '#B20710'],
figsize=(12, 6))
plt.title('Top 5 Countries: Movies vs TV Shows', fontsize=16,
weight='bold', pad=20)
plt.xlabel('Country', fontsize=12, weight='bold')
plt.ylabel('Number of Titles', fontsize=12, weight='bold')
plt.legend(title='Content Type', fontsize=10)
plt.xticks(rotation=45, ha='right')
plt.grid(True, alpha=0.3, axis='y')
plt.tight_layout()
plt.show()
```




```
PS D:\lasyafds> python -u "d:\lasyafds\questions\q1_q15.py"

==================================================
QUESTION 15: Top 5 Countries - Content Type Breakdown
==================================================

Top 5 Countries Content Breakdown:
type             Movie  TV Show
primary_country
Canada             187       84
India              927       81
United Kingdom     382      246
United States     2364      847
Unknown            440      391
```