```
#!/usr/bin/env python
# coding: utf-8

# In[1]:


# imports all my libraries
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import re
from datetime import datetime
import seaborn as sns



pd.set_option("display.max_rows", None, "display.max_columns", None)


# In[2]:


#load the datasets

movies = pd.read_csv("movies.csv")
ratings = pd.read_csv("ratings.csv")
links = pd.read_csv("links.csv")
tags = pd.read_csv("tags.csv")




# In[3]:


# Inspecting the dataframe
print(f"Movieshape: {movies.shape}")
movies.head()




# In[4]:


print(f"RatingsShape: {ratings.shape}")
movies.head(6)


# In[5]:


#show structure and data type
movies.info()
ratings.info()


# In[6]:


#check for missing values
movies.isna().sum()
ratings.isna().sum()


# In[7]:


ratings_movies = pd.merge(ratings, movies, on='movieId', how='left')


# In[8]:


duplicates = ratings_movies[ratings_movies.duplicated()]


# In[9]:


print (duplicates)


# In[10]:


if np.issubdtype(ratings_movies['timestamp'].dtype, np.number):
    ratings_movies['rating_ts'] = pd.to_datetime(ratings_movies['timestamp'], unit='s')
else:
    ratings_movies['rating_ts'] = pd.to_datetime(ratings_movies['timestamp'], errors='coerce')

ratings_movies[['timestamp', 'rating_ts']].head()


# In[11]:


df = ratings.merge(movies, on='movieId', how='left')
df.head()
ratings_movies.head(20)


# In[12]:


#extract movie year

def extract_year(title):
    if isinstance(title, str):
        m = re.search(r'\((\d{4})\)', title)
        return int(m.group(1)) if m else np.nan
    return np.nan


# In[34]:


#duplicate the merged dataset and create a new column called release year

ratings_movies_feat = ratings_movies.copy()

# extract the release year from the movie title
```

```
ratings_movies_feat['release_year'] = ratings_movies_feat['title'].str.extract(r'\((\d{4})\)').astype('Int64')


# In[35]:


#fills empty cells with NO genre listed

ratings_movies_feat['genres'] = ratings_movies_feat['genres'].fillna('(no genres listed)')


# In[36]:


#creates a new column called genres list

ratings_movies_feat['genre_list'] = ratings_movies_feat['genres'].apply(lambda s: s.split('|') if s!='(no genres listed)' else [])


# In[37]:


# counts the number of items in the genres list

ratings_movies_feat['genre_count'] = ratings_movies_feat['genre_list'].apply(len)


# In[38]:


# 3. Primary genre (first listed)

ratings_movies_feat['primary_genre'] = ratings_movies_feat['genre_list'].apply(lambda L: L[0] if len(L)>0 else 'Unknown')


# In[39]:


# 4. Movie title length (helps detect complexity or naming pattern)

ratings_movies_feat['title_length'] = ratings_movies_feat['title'].apply(lambda x: len(str(x)))


# In[40]:


#5. Release decade (group movies by era)

ratings_movies_feat['release_decade'] = (ratings_movies_feat['release_year'] //10*10).astype('Int64').astype(str)+'s'


# In[43]:


#6. Has sequel (detects if movie title contains a sequel number)
ratings_movies_feat['has_sequel'] = ratings_movies_feat['title'].apply(lambda x: 1 if re.search(r'\b[2-9]\b', str(x)) else 0)


# In[44]:


# Exploratory Data Analysis.

sns.set(style="whitegrid",context="talk")
plt.rcParams["figure.figsize"] =(10,6)


# In[45]:


# Rating Distribution

print(ratings_movies["rating"].describe())

sns.histplot(ratings_movies["rating"], bins=10, kde=True)
plt.title("Distribution of Movie Ratings")
plt.xlabel("Rating")
plt.ylabel("Count")
plt.show()


# In[46]:


# Explode genres so each movie-genre combination gets a separate row
ratings_exploded = ratings_movies.assign(
    genre=ratings_movies["genres"].str.split('|')
).explode("genre")

# Average rating and number of ratings per genre
genre_stats = ratings_exploded.groupby("genre")["rating"].agg(["mean", "count"]).sort_values("count", ascending=False)

# Select only top 10 genres
top10_genres = genre_stats.head(10)

# Plot average rating per genre (only top 10)
sns.barplot(data=top10_genres, x="mean", y=top10_genres.index, palette="viridis")
plt.title("Average Rating by Genre (Top 10)")
plt.xlabel("Average Rating")
plt.ylabel("Genre")
plt.show()




# In[47]:


# Convert timestamp to datetime and extract year

ratings_movies["rating_ts"] = pd.to_datetime(ratings_movies["timestamp"], unit="s", errors="coerce")
ratings_movies["rating_year"] = ratings_movies["rating_ts"].dt.year


# In[48]:


# Average rating per year

yearly_trend = ratings_movies.groupby("rating_year")["rating"].mean().dropna()

sns.lineplot(x=yearly_trend.index, y=yearly_trend.values, marker="o")
```

```
plt.title("Average Movie Rating Over Time")
plt.xlabel("Year")
plt.ylabel("Average Rating")
plt.show()


# In[49]:


# textual Summary

print("Average rating overall:", round(ratings_movies['rating'].mean(), 2))
print("Most common rating:", ratings_movies['rating'].mode()[0])
print("Top-rated genres:\n", genre_stats.head(5))


# In[50]:


# Split genres and expand rows
ratings_exploded = ratings_movies.assign(
    genre=ratings_movies["genres"].str.split('|')
).explode("genre")

# Count number of ratings per genre
genre_popularity = ratings_exploded["genre"].value_counts().head(10)

# Plot
sns.barplot(x=genre_popularity.values, y=genre_popularity.index, palette="Set2")
plt.title("Most Watched (Most Rated) Genres - Top 10")
plt.xlabel("Number of Ratings")
plt.ylabel("Genre")
plt.show()


# In[51]:


# Count how many ratings were made each year
year_popularity = ratings_movies["rating_year"].value_counts().sort_index().tail(10)

# Plot
sns.barplot(x=year_popularity.index, y=year_popularity.values, palette="magma")
plt.title("Top 10 Years with the Most Movie Ratings")
plt.xlabel("Year")
plt.ylabel("Number of Ratings")
plt.show()


# In[52]:


# Count number of ratings per genre
genre_popularity = ratings_exploded["genre"].value_counts()

# Get bottom 10 (least watched)
least_watched_genres = genre_popularity.tail(10)

# Display
least_watched_genres


# In[53]:


# Sorted barplot (from least to more)
least_sorted = least_watched_genres.sort_values(ascending=True)
plt.figure(figsize=(8,5))
sns.barplot(
    x=least_sorted.values,
    y=least_sorted.index,
    palette="Set3"   # pµñË Mixed colors
)
plt.title("Least Watched (Least Rated) Genres - Bottom 10 (Mixed Colors)")
plt.xlabel("Number of Ratings")
plt.ylabel("Genre")
plt.tight_layout()
plt.show()


# In[57]:


# Merged ratings_movies to get my new sheet
ratings_movies_feat.to_csv("merged_ratings_movies.csv", index=False)


# In[ ]:
```