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

In [ ]:
```
movies=pd.read_csv("C:\\Users\\Manikanta\\Drive_Documents\\DataSets\\Movies MetaDat
movies.head()
```

```
C:\Users\Manikanta\AppData\Local\Temp\ipykernel_4012\1885499080.py:1: DtypeWarnin
g: Columns (10) have mixed types. Specify dtype option on import or set low_memory
=False.
  movies=pd.read_csv("C:\\Users\\Manikanta\\Drive_Documents\\DataSets\\Movies Meta
Data\\movies_metadata.csv")
```

Out[ ]:

	adult	belongs_to_collection	budget	genres	homepage	id	imdb
0	False	{'id': 10194, 'name': 'Toy Story Collection', ...	30000000	[{'id': 16, 'name': 'Animation'}, {'id': 35, '...	http://toystory.disney.com/toy-story	862	tt0114
1	False	NaN	65000000	[{'id': 12, 'name': 'Adventure'}, {'id': 14, '...	NaN	8844	tt0113
2	False	{'id': 119050, 'name': 'Grumpy Old Men Collect...	0	[{'id': 10749, 'name': 'Romance'}, {'id': 35, ...	NaN	15602	tt0113
3	False	NaN	16000000	[{'id': 35, 'name': 'Comedy'}, {'id': 18, 'nam...	NaN	31357	tt0114
4	False	{'id': 96871, 'name': 'Father of the Bride Col...	0	[{'id': 35, 'name': 'Comedy'}]	NaN	11862	tt0113

5 rows × 24 columns

In [ ]:
```
df=movies[['id','imdb_id','title','genres','budget','revenue','adult','belongs_to_c
df.head()
```

Out[ ]:

	id	imdb_id	title	genres	budget	revenue	adult	belongs_to_collection	c
0	862	tt0114709	Toy Story	[{'id': 16, 'name': 'Animation'}, {'id': 35, '...	30000000	373554033.0	False	{'id': 10194, 'name': 'Toy Story Collection', ...	
1	8844	tt0113497	Jumanji	[{'id': 12, 'name': 'Adventure'}, {'id': 14, '...	65000000	262797249.0	False	NaN	
2	15602	tt0113228	Grumpier Old Men	[{'id': 10749, 'name': 'Romance'}, {'id': 35, ...	0	0.0	False	{'id': 119050, 'name': 'Grumpy Old Men Collect...	
3	31357	tt0114885	Waiting to Exhale	[{'id': 35, 'name': 'Comedy'}, {'id': 18, 'nam...	16000000	81452156.0	False	NaN	
4	11862	tt0113041	Father of the Bride Part II	[{'id': 35, 'name': 'Comedy'}]	0	76578911.0	False	{'id': 96871, 'name': 'Father of the Bride Col...	

In [ ]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 45466 entries, 0 to 45465
Data columns (total 17 columns):
 #   Column                 Non-Null Count  Dtype  
---  ------                 --------------  -----  
 0   id                     45466 non-null  object 
 1   imdb_id                45449 non-null  object 
 2   title                  45460 non-null  object 
 3   genres                 45466 non-null  object 
 4   budget                 45466 non-null  object 
 5   revenue                45460 non-null  float64
 6   adult                  45466 non-null  object 
 7   belongs_to_collection  4494 non-null   object 
 8   original_language      45455 non-null  object 
 9   popularity             45461 non-null  object 
 10  production_companies   45463 non-null  object 
 11  production_countries   45463 non-null  object 
 12  release_date           45379 non-null  object 
 13  runtime                45203 non-null  float64
 14  status                 45379 non-null  object 
 15  vote_average           45460 non-null  float64
 16  vote_count             45460 non-null  float64
dtypes: float64(4), object(13)
memory usage: 5.9+ MB
```

In [ ]:
```
import pandas as pd
import json


# Replace single quotes with double quotes in the 'genres' column
df['genres'] = df['genres'].apply(lambda x: x.replace("'", "\""))

# Convert the 'genres' column to a list of dictionaries
df['genres'] = df['genres'].apply(lambda x: json.loads(x))
```

```
# Create a list to store the genre data
genre_data = []

# Iterate through each row and extract genre data
for _, row in df.iterrows():
    movie_id = row['id']
    genres = row['genres']

    for genre in genres:
        genre_data.append({'movie_id': movie_id, 'genre_id': genre['id'], 'genre_na

# Create a new DataFrame from the genre data
genres_df = pd.DataFrame(genre_data)
```

```
C:\Users\Manikanta\AppData\Local\Temp\ipykernel_4012\4196781843.py:6: SettingWithC
opyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  df['genres'] = df['genres'].apply(lambda x: x.replace("'", "\""))
C:\Users\Manikanta\AppData\Local\Temp\ipykernel_4012\4196781843.py:9: SettingWithC
opyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  df['genres'] = df['genres'].apply(lambda x: json.loads(x))
```

In [ ]:
```
genres_df
```

Out[ ]:

	movie_id	genre_id	genre_name
0	862	16	Animation
1	862	35	Comedy
2	862	10751	Family
3	8844	12	Adventure
4	8844	14	Fantasy
...	...	...	...
91101	439050	10751	Family
91102	111109	18	Drama
91103	67758	28	Action
91104	67758	18	Drama
91105	67758	53	Thriller

91106 rows × 3 columns

In [ ]:
```
#save file

genres_df.to_csv("path/file.csv")
```

In [ ]:
```
# Create a deep copy of the DataFrame to avoid SettingWithCopyWarning
df = df.copy()

# Replace single quotes with double quotes in the 'production_companies' column
df['production_companies'] = df['production_companies'].apply(lambda x: str(x).repl

# Convert the 'production_companies' column to a list of dictionaries
def parse_production_companies(x):
    try:
        return json.loads(x) if pd.notna(x) else []
    except json.JSONDecodeError:
        return []

df['production_companies'] = df['production_companies'].apply(parse_production_comp

# Create a list to store the production company data
company_data = []

# Iterate through each row and extract production company data
for _, row in df.iterrows():
    movie_id = row['id']
    production_companies = row['production_companies']

    for company in production_companies:
        company_data.append({'movie_id': movie_id, 'company_id': company['id'], 'co

# Create a new DataFrame from the production company data
companies_df = pd.DataFrame(company_data)
```

In [ ]:
```
companies_df
```

Out[ ]:

	movie_id	company_id	company_name
0	862	3	Pixar Animation Studios
1	8844	559	TriStar Pictures
2	8844	2550	Teitler Film
3	8844	10201	Interscope Communications
4	15602	6194	Warner Bros.
...	...	...	...
68504	30840	16323	20th Century Fox Television
68505	30840	38978	CanWest Global Communications
68506	111109	19653	Sine Olivia
68507	67758	6165	American World Pictures
68508	227506	88753	Yermoliev

68509 rows × 3 columns

In [ ]:
```
#save file
companies_df.to_csv("path/file.csv")
```

In [ ]:
```
df = df.copy()

# Replace single quotes with double quotes in the 'production_countries' column
df['production_countries'] = df['production_countries'].apply(lambda x: str(x).repl
```

```
# Convert the 'production_countries' column to a list of dictionaries
def parse_production_countries(x):
    try:
        return json.loads(x) if pd.notna(x) else []
    except json.JSONDecodeError:
        return []

df['production_countries'] = df['production_countries'].apply(parse_production_cour

# Create a list to store the production country data
country_data = []

# Iterate through each row and extract production country data
for _, row in df.iterrows():
    movie_id = row['id']
    production_countries = row['production_countries']

    if isinstance(production_countries, list):  # Check if the value is a list
        for country in production_countries:
            country_data.append({'movie_id': movie_id, 'iso_3166_1': country['iso_3

# Create a new DataFrame from the production country data
countries_df = pd.DataFrame(country_data)
```

In [ ]:
```
countries_df
```

Out[ ]:

	movie_id	iso_3166_1	country_name
0	862	US	United States of America
1	8844	US	United States of America
2	15602	US	United States of America
3	31357	US	United States of America
4	11862	US	United States of America
...	...	...	...
49403	439050	IR	Iran
49404	111109	PH	Philippines
49405	67758	US	United States of America
49406	227506	RU	Russia
49407	461257	GB	United Kingdom

49408 rows × 3 columns

In [ ]:
```
#save file
countries_df.to_csv("path/file.csv")
```

In [ ]:
```
movies=movies[['id','imdb_id','title','budget','revenue','adult','original_language
movies
```

Out[ ]:

	id	imdb_id	title	budget	revenue	adult	original_language	popularity
0	862	tt0114709	Toy Story	30000000	373554033.0	False	en	21.946943
1	8844	tt0113497	Jumanji	65000000	262797249.0	False	en	17.015539
2	15602	tt0113228	Grumpier Old Men	0	0.0	False	en	11.7129
3	31357	tt0114885	Waiting to Exhale	16000000	81452156.0	False	en	3.859495
4	11862	tt0113041	Father of the Bride Part II	0	76578911.0	False	en	8.387519
...	...	...	...	...	...	...	...	...
45461	439050	tt6209470	Subdue	0	0.0	False	fa	0.072051
45462	111109	tt2028550	Century of Birthing	0	0.0	False	tl	0.178241
45463	67758	tt0303758	Betrayal	0	0.0	False	en	0.903007
45464	227506	tt0008536	Satan Triumphant	0	0.0	False	en	0.003503
45465	461257	tt6980792	Queerama	0	0.0	False	en	0.163015

45466 rows × 13 columns

In [ ]:
```
#save file
movies.to_csv("path/file.csv")
```