This Notebook Explores the IMDB Database and The Movie Database Data

Goal of this notebook is to collect the data we need and merge the two data sets together

In [1]:
```
# import necessary
import pandas as pd
import sqlite3
import pandasql
import numpy as np
import zipfile

# changes the display for float values
pd.options.display.float_format = '{:.0f}'.format

conn = sqlite3.connect("../data/zippedData/im.db")
cur = conn.cursor()
```

In [2]:
```
# open and close the zip file containing imdb database
with zipfile.ZipFile("../data/zippedData/im.db.zip", 'r') as zip_ref:
    zip_ref.extractall("../data/zippedData/")
```

In [3]:
```
# take a look at all the tables in the database
pd.read_sql(
'''
SELECT *
FROM sqlite_master
''',
conn)
```

Out[3]:

	type	name	tbl_name	rootpage	sql
0	table	movie_basics	movie_basics	2	CREATE TABLE "movie_basics" (\n"movie_id" TEXT...
1	table	directors	directors	3	CREATE TABLE "directors" (\n"movie_id" TEXT,\n...
2	table	known_for	known_for	4	CREATE TABLE "known_for" (\n"person_id" TEXT,\...
3	table	movie_akas	movie_akas	5	CREATE TABLE "movie_akas" (\n"movie_id" TEXT,\...
4	table	movie_ratings	movie_ratings	6	CREATE TABLE "movie_ratings" (\n"movie_id" TEX...
5	table	persons	persons	7	CREATE TABLE "persons" (\n"person_id" TEXT,\n ...
6	table	principals	principals	8	CREATE TABLE "principals" (\n"movie_id" TEXT,\...
7	table	writers	writers	9	CREATE TABLE "writers" (\n"movie_id" TEXT,\n ...

In [4]:
```
# first look at movie_basics
q1 = pd.read_sql(
'''
SELECT
    *
FROM
```

```
    movie_basics
ORDER BY
    start_year ASC
''',
conn)
```

In [5]:
```
q1.head(15) # movie basics is a table showing all movies from 2010- and gives runtime a
```

Out[5]:

	movie_id	primary_title	original_title	start_year	runtime_minutes	genres
0	tt0146592	Pál Adrienn	Pál Adrienn	2010	136	Drama
1	tt0154039	So Much for Justice!	Oda az igazság	2010	100	History
2	tt0162942	Children of the Green Dragon	A zöld sárkány gyermekei	2010	89	Drama
3	tt0230212	The Final Journey	The Final Journey	2010	120	Drama
4	tt0312305	Quantum Quest: A Cassini Space Odyssey	Quantum Quest: A Cassini Space Odyssey	2010	45	Adventure,Animation,Sci-Fi
5	tt0326592	The Overnight	The Overnight	2010	88	None
6	tt0326965	In My Sleep	In My Sleep	2010	104	Drama,Mystery,Thriller
7	tt0331312	This Wretched Life	This Wretched Life	2010	99	Comedy,Drama
8	tt0337882	Blind Sided	Blind Sided	2010	nan	Comedy,Crime,Drama
9	tt0378546	The Payback Man	The Payback Man	2010	nan	Crime,Drama,Thriller
10	tt0393049	Anderson's Cross	Anderson's Cross	2010	98	Comedy,Drama,Romance
11	tt0396123	Den milde smerte	Den milde smerte	2010	280	Drama
12	tt0398286	Tangled	Tangled	2010	100	Adventure,Animation,Comedy
13	tt0402887	Capture the Flag	Capture the Flag	2010	nan	Drama
14	tt0403645	Burnt by the Sun 2	Utomlennye solntsem 2	2010	181	Drama,History,War

In [6]:
```
# first look at directors table
q1 = pd.read_sql(
'''
SELECT
    *
FROM
    directors
''',
conn)
```

```
q1.head()
```

Out[6]:

	movie_id	person_id
0	tt0285252	nm0899854
1	tt0462036	nm1940585
2	tt0835418	nm0151540
3	tt0835418	nm0151540
4	tt0878654	nm0089502

Two columns for the person_id and the movie_id

In [7]:
```
# first look at the known_for table
q1 = pd.read_sql(
'''
SELECT
    *
FROM
    known_for
''',
conn)

q1.head(10)
```

Out[7]:

	person_id	movie_id
0	nm0061671	tt0837562
1	nm0061671	tt2398241
2	nm0061671	tt0844471
3	nm0061671	tt0118553
4	nm0061865	tt0896534
5	nm0061865	tt6791238
6	nm0061865	tt0287072
7	nm0061865	tt1682940
8	nm0062070	tt1470654
9	nm0062070	tt0363631

In [8]:
```
# first look at the movie_akas table
q1 = pd.read_sql(
'''
SELECT
    *
FROM
    movie_akas

''',
conn)
```

```
q1
```

Out[8]:

	movie_id	ordering	title	region	language	types	attributes	is_original_title
0	tt0369610	10	Джурасик свят	BG	bg	None	None	0
1	tt0369610	11	Jurashikku warudo	JP	None	imdbDisplay	None	0
2	tt0369610	12	Jurassic World: O Mundo dos Dinossauros	BR	None	imdbDisplay	None	0
3	tt0369610	13	O Mundo dos Dinossauros	BR	None	None	short title	0
4	tt0369610	14	Jurassic World	FR	None	imdbDisplay	None	0
...	...	...	...	...	...	...	...	...
331698	tt9827784	2	Sayonara kuchibiru	None	None	original	None	1
331699	tt9827784	3	Farewell Song	XWW	en	imdbDisplay	None	0
331700	tt9880178	1	La atención	None	None	original	None	1
331701	tt9880178	2	La atención	ES	None	None	None	0
331702	tt9880178	3	The Attention	XWW	en	imdbDisplay	None	0

331703 rows × 8 columns

In [9]:
```
# filter the movie_akas table to only show US movies
q1 = pd.read_sql(
'''
SELECT
    DISTINCT region
FROM
    movie_akas
WHERE
    region LIKE '%US%'
ORDER BY
    region DESC
''',
conn)
```

The next query we wanted the correct name for the main composer of the movie because there are many different workers under the main composer with similar titles. We used Hans Zimmer as an example to find what the main composer's job title is

In [10]:
```
# This query is
q1 = pd.read_sql(
'''
```

```
SELECT
    principals.*,
    per.primary_name
FROM
    principals
    INNER JOIN persons AS per ON principals.person_id = per.person_id
WHERE
    per.primary_name == "Hans Zimmer"

''',
conn)

q1
```

Out[10]:

	movie_id	ordering	person_id	category	job	characters	primary_name
0	tt0816692	9	nm0001877	composer	None	None	Hans Zimmer
1	tt1210819	10	nm0001877	composer	None	None	Hans Zimmer
2	tt1270766	10	nm0001877	composer	None	None	Hans Zimmer
3	tt1341188	9	nm0001877	composer	None	None	Hans Zimmer
4	tt1375666	7	nm0001877	composer	None	None	Hans Zimmer
5	tt1531683	3	nm0001877	composer	None	None	Hans Zimmer
6	tt1578275	9	nm0001877	composer	None	None	Hans Zimmer
7	tt1823672	8	nm0001877	composer	None	None	Hans Zimmer
8	tt2514298	10	nm0001877	composer	None	None	Hans Zimmer
9	tt3062096	9	nm0001877	composer	None	None	Hans Zimmer
10	tt3286560	10	nm0001877	composer	None	None	Hans Zimmer
11	tt3874544	10	nm0001877	composer	None	None	Hans Zimmer
12	tt4218572	10	nm0001877	composer	None	None	Hans Zimmer
13	tt5013056	7	nm0001877	composer	None	None	Hans Zimmer
14	tt5732482	1	nm0001877	self	None	["Himself - Piano","Guitar","Banjo"]	Hans Zimmer
15	tt7380834	7	nm0001877	composer	None	None	Hans Zimmer
16	tt7689424	10	nm0001877	composer	None	None	Hans Zimmer
17	tt8632844	8	nm0001877	composer	None	None	Hans Zimmer

The next query is the query we are going to run that will have most of the data we need for our analysis. We joined several tables together to get all the information.

In [11]:

```
# Decided to also filter 'composer' due to it 'composer' being the main composer of the
# opposed to 'head composer', 'main composer', 'lead composer', etc.
movie_info = pd.read_sql(
'''
SELECT
    mb.movie_id,
    mb.primary_title,
```

```
    mb.original_title,
    mb.start_year,
    mb.genres,
    mr.averagerating AS average_rating,
    mr.numvotes AS num_votes,
    per.primary_name AS persons_name,
    princ.category AS persons_job
FROM
    movie_basics AS mb
    INNER JOIN movie_ratings AS mr ON mb.movie_id = mr.movie_id
    INNER JOIN principals AS princ ON mb.movie_id = princ.movie_id
    INNER JOIN persons AS per ON princ.person_id = per.person_id
    INNER JOIN movie_akas AS ma ON mb.movie_id = ma.movie_id
WHERE
    princ.category IN ('actor', 'actress', 'director', 'writer', 'producer', 'composer'
    AND ma.region == 'US'
ORDER BY
    mb.start_year ASC
''',
conn)

movie_info
```

Out[11]:

	movie_id	primary_title	original_title	start_year	genres	average_rating	num_
0	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	3
1	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	3
2	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	3
3	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	3
4	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	3
...	...	...	...	...	...	...	
214270	tt9562694	Alien Warfare	Alien Warfare	2019	Action,Sci-Fi	3	
214271	tt9562694	Alien Warfare	Alien Warfare	2019	Action,Sci-Fi	3	
214272	tt9562694	Alien Warfare	Alien Warfare	2019	Action,Sci-Fi	3	
214273	tt9562694	Alien Warfare	Alien Warfare	2019	Action,Sci-Fi	3	
214274	tt9562694	Alien Warfare	Alien Warfare	2019	Action,Sci-Fi	3	

214275 rows × 9 columns

In [12]:
```
# how many movies are there in the dataset
len(movie_info['movie_id'].unique())
```

Out[12]: 26526

Now that we have our movie info dataframe we can look to merge it with the tn.movie_budgets csv

In [13]:
```
# import our tn.movie_budgets.csv
tn_movie_budgets = pd.read_csv('../data/zippedData/tn.movie_budgets.csv.gz')
```

In [14]:
```
# now let's take a look at it
print(tn_movie_budgets.shape)
tn_movie_budgets.head()
```

```
(5782, 6)
```

Out[14]:

	id	release_date	movie	production_budget	domestic_gross	worldwide_gross
0	1	Dec 18, 2009	Avatar	$425,000,000	$760,507,625	$2,776,345,279
1	2	May 20, 2011	Pirates of the Caribbean: On Stranger Tides	$410,600,000	$241,063,875	$1,045,663,875
2	3	Jun 7, 2019	Dark Phoenix	$350,000,000	$42,762,350	$149,762,350
3	4	May 1, 2015	Avengers: Age of Ultron	$330,600,000	$459,005,868	$1,403,013,963
4	5	Dec 15, 2017	Star Wars Ep. VIII: The Last Jedi	$317,000,000	$620,181,382	$1,316,721,747

In [15]:
```
# how many movies are in the dataset
len(tn_movie_budgets['movie'].unique())
```

Out[15]:
```
5698
```

In [16]:
```
# now let's look at our movie_info data frame
print(movie_info.shape)
movie_info.head()
```

```
(214275, 9)
```

Out[16]:

	movie_id	primary_title	original_title	start_year	genres	average_rating	num_votes
0	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
1	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
2	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
3	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
4	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813

They have a common movie title column. I'll create a new column in each that is a normalization of the movie titles. They will be lower case and stripped of any trailing white spaces.

In [17]:
```
# test the strip and lower methods on a string
x = '   HeLo ThiS is A movIE   '
```

```
x.strip().lower()
```

Out[17]: 'helo this is a movie'

In [18]:
```
# first the tn_movie_budgets
tn_movie_budgets['title_norm'] = tn_movie_budgets['movie'].str.strip().str.lower()

tn_movie_budgets
```

Out[18]:

	id	release_date	movie	production_budget	domestic_gross	worldwide_gross	title_norm
0	1	Dec 18, 2009	Avatar	$425,000,000	$760,507,625	$2,776,345,279	avatar
1	2	May 20, 2011	Pirates of the Caribbean: On Stranger Tides	$410,600,000	$241,063,875	$1,045,663,875	pirates of the caribbean: on stranger tides
2	3	Jun 7, 2019	Dark Phoenix	$350,000,000	$42,762,350	$149,762,350	dark phoenix
3	4	May 1, 2015	Avengers: Age of Ultron	$330,600,000	$459,005,868	$1,403,013,963	avengers: age of ultron
4	5	Dec 15, 2017	Star Wars Ep. VIII: The Last Jedi	$317,000,000	$620,181,382	$1,316,721,747	star wars ep. viii: the last jedi
...	...	...	...	...	...	...	...
5777	78	Dec 31, 2018	Red 11	$7,000	$0	$0	red 11
5778	79	Apr 2, 1999	Following	$6,000	$48,482	$240,495	following
5779	80	Jul 13, 2005	Return to the Land of Wonders	$5,000	$1,338	$1,338	return to the land of wonders
5780	81	Sep 29, 2015	A Plague So Pleasant	$1,400	$0	$0	a plague so pleasant
5781	82	Aug 5, 2005	My Date With Drew	$1,100	$181,041	$181,041	my date with drew

5782 rows × 7 columns

In [19]:
```
tn_movie_budgets.loc[tn_movie_budgets['movie'] == 'Alice in Wonderland']
```

Out[19]:

	id	release_date	movie	production_budget	domestic_gross	worldwide_gross	title_norm
50	51	Mar 5, 2010	Alice in Wonderland	$200,000,000	$334,191,110	$1,025,491,110	alice in wonderland
4759	60	Jul 28, 1951	Alice in Wonderland	$3,000,000	$0	$0	alice in wonderland

In [20]:
```
# now the movie_info dataframe
movie_info['title_norm'] = movie_info['primary_title'].str.strip().str.lower()

movie_info.head()
```

Out[20]:

	movie_id	primary_title	original_title	start_year	genres	average_rating	num_votes
0	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
1	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
2	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
3	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
4	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813

In [21]:
```
movie_info.groupby(by=['movie_id', 'primary_title']).count().sort_values(by='primary_ti
```

Out[21]:

		original_title	start_year	genres	average_rating	num_votes	persons_na
movie_id	**primary_title**						
tt2346170	**#1 Serial Killer**	8	8	8	8	8	
tt3120962	**#5**	2	2	2	2	2	
tt5074174	**#BeRobin the Movie**	1	1	1	1	1	
tt6856592	**#Captured**	9	9	9	9	9	
tt5803530	**#DigitalLivesMatter**	10	10	10	10	10	
...	**...**	...	...	...	...	...	
tt6194704	**À 2 heures de Paris**	6	6	6	6	6	
tt3550444	**Ækte vare**	9	9	9	9	9	
tt1822381	**Évocateur: The Morton Downey Jr. Movie**	4	4	4	4	4	
tt1754950	**Última sesión**	8	8	8	8	8	
tt5065762	**Últimos días en La Habana**	9	9	9	9	9	

26526 rows × 8 columns

In [22]:
```
movie_info.loc[movie_info['movie_id'] == 'tt2049386']
```

Out[22]:

	movie_id	primary_title	original_title	start_year	genres	average_rating	num_votes	per
8978	tt2049386	Alice in Wonderland	Alice in Wonderland	2010	Fantasy,Musical	6	6	
8979	tt2049386	Alice in Wonderland	Alice in Wonderland	2010	Fantasy,Musical	6	6	La
8980	tt2049386	Alice in Wonderland	Alice in Wonderland	2010	Fantasy,Musical	6	6	
8981	tt2049386	Alice in Wonderland	Alice in Wonderland	2010	Fantasy,Musical	6	6	
8982	tt2049386	Alice in Wonderland	Alice in Wonderland	2010	Fantasy,Musical	6	6	
8983	tt2049386	Alice in Wonderland	Alice in Wonderland	2010	Fantasy,Musical	6	6	
8984	tt2049386	Alice in Wonderland	Alice in Wonderland	2010	Fantasy,Musical	6	6	

Now I'll merge them based on the title_norm column

In [23]:
```
movie_info.merge(tn_movie_budgets, how='inner', on='title_norm',suffixes=('_movie_info'
```

Out[23]:

	movie_id	primary_title	original_title	start_year	genres	average_rating	num_
0	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35
1	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35
2	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35
3	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35
4	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35
...	...	...	...	...	...	...	
26613	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
26614	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
26615	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
26616	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
26617	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5

26618 rows × 16 columns

The problem here is that movies that share the same name but different years are causing problems. To fix this we can add the year to the title_norm column for each to differentiate movies with same names and different years.

In [24]:
```
# adding release year to title_norm for tn_movie_budgets
tn_movie_budgets['title_norm'] = tn_movie_budgets['movie'].str.strip().str.lower() + ' '
                                 + tn_movie_budgets['release_date'].str[-4:]
```

In [25]:
```
tn_movie_budgets.head()
```

Out[25]:

	id	release_date	movie	production_budget	domestic_gross	worldwide_gross	title_norm
0	1	Dec 18, 2009	Avatar	$425,000,000	$760,507,625	$2,776,345,279	avatar 2009
1	2	May 20, 2011	Pirates of the Caribbean: On Stranger Tides	$410,600,000	$241,063,875	$1,045,663,875	pirates of the caribbean: on stranger tides 2011
2	3	Jun 7, 2019	Dark Phoenix	$350,000,000	$42,762,350	$149,762,350	dark phoenix 2019
3	4	May 1, 2015	Avengers: Age of Ultron	$330,600,000	$459,005,868	$1,403,013,963	avengers: age of ultron 2015
4	5	Dec 15, 2017	Star Wars Ep. VIII: The Last Jedi	$317,000,000	$620,181,382	$1,316,721,747	star wars ep. viii: the last jedi 2017

In [26]:
```
movie_info.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 214275 entries, 0 to 214274
Data columns (total 10 columns):
 #   Column          Non-Null Count   Dtype  
---  ------          --------------   -----  
 0   movie_id        214275 non-null  object 
 1   primary_title   214275 non-null  object 
 2   original_title  214275 non-null  object 
 3   start_year      214275 non-null  int64  
 4   genres          213612 non-null  object 
 5   average_rating  214275 non-null  float64
 6   num_votes       214275 non-null  int64  
 7   persons_name    214275 non-null  object 
 8   persons_job     214275 non-null  object 
 9   title_norm      214275 non-null  object 
dtypes: float64(1), int64(2), object(7)
memory usage: 16.3+ MB
```

In [27]:
```
# adding release year to title_norm for movie_info
movie_info['title_norm'] = movie_info['primary_title'].str.strip().str.lower() + ' ' \
                           + movie_info['start_year'].astype(str)
```

In [28]:
```
movie_info.head()
```

Out[28]:

	movie_id	primary_title	original_title	start_year	genres	average_rating	num_votes
0	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
1	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
2	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
3	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
4	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813

title_norm worked for both now we'll try to merge the tables again

In [29]:
```
df_movie_info_budget = movie_info.merge(tn_movie_budgets,
                 how='inner',
                 on='title_norm',
                 suffixes=('_movie_info', '_movie_budgets'))
```

In [30]:
```
# The merged looked like it worked now it's time to take a look at the new dataframe
df_movie_info_budget.head(10)
```

Out[30]:

	movie_id	primary_title	original_title	start_year	genres	average_rating	num_votes
0	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
1	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
2	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
3	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
4	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
5	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813

	movie_id	primary_title	original_title	start_year	genres	average_rating	num_votes
6	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
7	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
8	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
9	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813

In [31]:
```
df_movie_info_budget.tail(10)
```

Out[31]:

	movie_id	primary_title	original_title	start_year	genres	average_rating	num_
20414	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
20415	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
20416	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
20417	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
20418	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
20419	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
20420	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
20421	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
20422	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
20423	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5

In [32]:
```
# how many movies are in the dataset
len(df_movie_info_budget['movie_id'].unique())
```

Out[32]:
```
1357
```

In [33]:
```
df_movie_info_budget.duplicated().sum()
```

Out[33]: 7735

In [34]:
```
df_movie_info_budget[df_movie_info_budget.duplicated()].head(20)
```

Out[34]:

	movie_id	primary_title	original_title	start_year	genres	average_rating	num_votes
10	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
11	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
12	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
13	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
14	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
15	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
16	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
17	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
18	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
19	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
20	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
21	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
22	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
23	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881

	movie_id	primary_title	original_title	start_year	genres	average_rating	num_vote
24	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
25	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
26	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
27	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
28	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
29	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881

In [35]:
```
df_movie_info_budget.loc[(df_movie_info_budget['primary_title'] == 'Alice in Wonderland
                        & (df_movie_info_budget['persons_name'] == 'Johnny Depp')]
```

Out[35]:

	movie_id	primary_title	original_title	start_year	genres	average_rating	num_vote
0	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
10	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
20	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881

There are duplicate rows in the new data frame. I'll modify the data frame to drop the duplicate rows

In [36]:
```
# drop the duplicate rows
df_movie_info_budget.drop_duplicates(inplace=True)
```

In [37]:
```
# check if the duplicates were dropped correctly
df_movie_info_budget.loc[(df_movie_info_budget['primary_title'] == 'Alice in Wonderland
                        & (df_movie_info_budget['persons_name'] == 'Johnny Depp')]
```

Out[37]:

	movie_id	primary_title	original_title	start_year	genres	average_rating	num_votes

	movie_id	primary_title	original_title	start_year	genres	average_rating	num_votes
0	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813

Removing the duplicates worked

In [38]:
```
# taking a look at the new data frame
df_movie_info_budget.head()
```

Out[38]:

	movie_id	primary_title	original_title	start_year	genres	average_rating	num_votes
0	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
1	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
2	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
3	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
4	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813

In [39]:
```
# look at the info of the dataframe
df_movie_info_budget.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 12689 entries, 0 to 20414
Data columns (total 16 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   movie_id           12689 non-null  object 
 1   primary_title      12689 non-null  object 
 2   original_title     12689 non-null  object 
 3   start_year         12689 non-null  int64  
 4   genres             12689 non-null  object 
 5   average_rating     12689 non-null  float64
 6   num_votes          12689 non-null  int64  
 7   persons_name       12689 non-null  object 
 8   persons_job        12689 non-null  object 
 9   title_norm         12689 non-null  object 
 10  id                 12689 non-null  int64  
 11  release_date       12689 non-null  object 
 12  movie              12689 non-null  object 
 13  production_budget  12689 non-null  object 
 14  domestic_gross     12689 non-null  object 
 15  worldwide_gross    12689 non-null  object 
dtypes: float64(1), int64(3), object(12)
memory usage: 1.6+ MB
```

In [40]:
```
len(df_movie_info_budget['title_norm'].unique())
```

Out[40]: 1348

Now I am going to normalize the production_budget, domestic_gross, and worldwide_gross to convert them to integers

In [41]:
```
# test the replace and strip method on the string
x = '$900,000,000'
x.replace(',', '').strip('$')
```

Out[41]: '900000000'

In [42]:
```
# list for the columns to be changed
columns = ['production_budget', 'domestic_gross', 'worldwide_gross']

# for loop that will go into each column I want to change and
# replace, strip, and convert it to a int
for column in columns:
    df_movie_info_budget[column] = df_movie_info_budget[column].str.replace(',','').str
```

In [43]:
```
df_movie_info_budget.head(20)
```

Out[43]:

	movie_id	primary_title	original_title	start_year	genres	average_rating	num_vote
0	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
1	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
2	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
3	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
4	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
5	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
6	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
7	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881

	movie_id	primary_title	original_title	start_year	genres	average_rating	num_vote
8	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
9	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
30	tt2049386	Alice in Wonderland	Alice in Wonderland	2010	Fantasy,Musical	6	
31	tt2049386	Alice in Wonderland	Alice in Wonderland	2010	Fantasy,Musical	6	
32	tt2049386	Alice in Wonderland	Alice in Wonderland	2010	Fantasy,Musical	6	
33	tt2049386	Alice in Wonderland	Alice in Wonderland	2010	Fantasy,Musical	6	
34	tt2049386	Alice in Wonderland	Alice in Wonderland	2010	Fantasy,Musical	6	
35	tt2049386	Alice in Wonderland	Alice in Wonderland	2010	Fantasy,Musical	6	
36	tt2049386	Alice in Wonderland	Alice in Wonderland	2010	Fantasy,Musical	6	
37	tt1294226	The Last Song	The Last Song	2010	Drama,Music,Romance	6	7491
38	tt1294226	The Last Song	The Last Song	2010	Drama,Music,Romance	6	7491
39	tt1294226	The Last Song	The Last Song	2010	Drama,Music,Romance	6	7491

The columns were correctly changed to integer values

Now I want to get a list of all the unique genres in the genre column

This is going to be necessary so I can analyze the specific genre of a movie as a opposed to the many genres it shares.

I have to change Musical to Singing because for the cleaning process to work correctly I need to test if the genre (ex. 'Music') appears in the genres column of the dataframe. But the problem is that Music will be true if the genre is a Musical. So to avoid this error I will change Musical to Singing in the genres column.

In [44]:
```
# replace Musical with Singing
df_movie_info_budget['genres'] = df_movie_info_budget['genres'].str.replace('Musical','
```

In [45]:
```
# turn the 'genres' column into a list
genres = list(df_movie_info_budget['genres'].unique())
```

In [46]:
```
# turn it into a list of a set of the genres list
genres = list(set(genres))

genres
```

Out[46]:
```
['Comedy,Crime,Drama',
 'Adventure,Family,Sci-Fi',
 'Animation,Family',
 'Comedy,Sport',
 'Comedy,Drama,Fantasy',
 'Comedy,Family,Romance',
 'Fantasy,Horror,Thriller',
 'Family',
 'Adventure,Comedy,Crime',
 'Horror,Mystery,Sci-Fi',
 'Romance,Sci-Fi,Thriller',
 'Action,Adventure,Animation',
 'Drama,Western',
 'Drama,Singing',
 'Drama,Music,Thriller',
 'Animation,Comedy,Family',
 'Action,Fantasy,Western',
 'Action,Adventure,Mystery',
 'Comedy,Horror',
 'Adventure,Drama,Thriller',
 'Action,Comedy,Fantasy',
 'Drama,Family',
 'Comedy,Fantasy',
 'Action',
 'Drama,Music,Singing',
 'Action,Crime,Sci-Fi',
 'Comedy,Horror,Romance',
 'Comedy,Romance',
 'Crime,Documentary',
 'Drama,Fantasy,Romance',
 'Comedy,Horror,Sci-Fi',
 'Action,Drama,Mystery',
 'Horror,Sci-Fi,Thriller',
 'Action,Comedy',
 'Biography,Drama,Sport',
 'Action,Drama,Family',
 'Adventure,Drama,Fantasy',
 'Comedy,Family',
 'Adventure,Drama,Western',
 'Biography,Documentary',
 'Adventure,Comedy,Family',
 'Adventure,Animation,Comedy',
 'Adventure,Mystery,Sci-Fi',
 'Comedy,Drama,Romance',
 'Action,Comedy,Family',
 'Biography,Comedy,Crime',
 'Comedy,Western',
 'Adventure,Drama,History',
 'Drama,War',
 'Action,Comedy,Sci-Fi',
 'Drama,History,Mystery',
 'Biography,Family,Sport',
```

```
'Biography,Drama,War',
'Comedy,Drama,History',
'Biography,Drama,Singing',
'Comedy,Crime,Romance',
'Drama,Romance,Thriller',
'Biography,Crime,Drama',
'Comedy,Fantasy,Horror',
'Action,Family,Fantasy',
'Comedy,Drama,Horror',
'Action,Adventure,Biography',
'Action,Sci-Fi,Thriller',
'Action,Crime,Thriller',
'Action,Comedy,Sport',
'Action,Adventure,Crime',
'Comedy,Mystery',
'Action,Drama,Sport',
'Drama,Music,Romance',
'Drama,Fantasy,Horror',
'Drama,Romance',
'Crime,Drama,Thriller',
'Crime,Horror,Mystery',
'Horror,Mystery,Thriller',
'Action,Drama',
'Adventure,Comedy',
'Action,Adventure,Drama',
'Music',
'Drama,Horror,Mystery',
'Action,Sci-Fi',
'Adventure,Drama,Horror',
'Comedy,Music',
'Drama,Mystery,War',
'Action,Comedy,Documentary',
'Adventure,Horror,Mystery',
'Action,Fantasy,Thriller',
'Comedy,Drama,Sport',
'Action,Drama,Thriller',
'Family,Sci-Fi',
'Horror',
'Action,Adventure,Family',
'Biography,Documentary,Music',
'Family,Fantasy,Singing',
'Crime,Mystery,Thriller',
'Biography,Drama,Thriller',
'Action,Mystery,Sci-Fi',
'Action,Drama,War',
'Action,Biography,Crime',
'Action,Horror',
'Biography,Comedy,Drama',
'Animation,Comedy,Drama',
'Comedy,Mystery,Sci-Fi',
'Action,Adventure',
'Drama,Thriller',
'Biography,Drama,Romance',
'Comedy,Fantasy,Romance',
'Action,Comedy,Romance',
'Crime,Drama,Romance',
'Adventure,Comedy,Drama',
'Action,Drama,History',
'Documentary,Sport',
'Action,Adventure,Western',
'Comedy,Romance,Sport',
'Comedy,Drama,Mystery',
'Adventure,Comedy,Music',
'Drama,Horror,Thriller',
'Drama,History,Thriller',
```

```
'Drama',
'Comedy,Drama,Family',
'Action,Horror,Thriller',
'Adventure,Fantasy',
'Comedy,Music,War',
'Romance',
'Drama,Romance,Sci-Fi',
'Documentary',
'Comedy,Drama,Music',
'Horror,Thriller',
'Horror,Sci-Fi',
'Drama,Fantasy',
'Adventure,Drama,Sci-Fi',
'Adventure,Comedy,Romance',
'Crime,Drama,Mystery',
'Horror,Mystery',
'Biography,Drama',
'Action,Adventure,Sci-Fi',
'Adventure,Comedy,Sci-Fi',
'Action,Adventure,Comedy',
'Action,Animation,Comedy',
'Action,Crime',
'Comedy,Fantasy,Singing',
'Action,Crime,Horror',
'Comedy',
'Adventure,Biography,Drama',
'Action,Biography,Drama',
'Comedy,Crime,Thriller',
'Comedy,Mystery,Thriller',
'Drama,Family,Fantasy',
'Action,Comedy,Drama',
'Western',
'Action,Comedy,Horror',
'Action,Adventure,Horror',
'Singing',
'Biography,Drama,Family',
'Comedy,Drama',
'Drama,Sport',
'Comedy,Sci-Fi',
'Action,Fantasy,Horror',
'Crime,Drama,Horror',
'Comedy,Fantasy,Sci-Fi',
'Drama,Family,Music',
'Action,Fantasy,War',
'Adventure,Family,Fantasy',
'Fantasy,Singing',
'Action,Thriller',
'Documentary,Music',
'Drama,Fantasy,Music',
'Drama,Horror',
'Adventure,Drama,Family',
'Documentary,Drama',
'Action,Comedy,Crime',
'Biography,Drama,Mystery',
'Action,Adventure,Fantasy',
'Action,Horror,Sci-Fi',
'Action,Drama,Sci-Fi',
'Action,Adventure,Thriller',
'Adventure,Animation,Family',
'Action,Biography,Comedy',
'Fantasy',
'Comedy,Music,Romance',
'Fantasy,Horror,Mystery',
'Action,Crime,Fantasy',
'Drama,History,War',
```

```
 'Action,Crime,Mystery',
 'Comedy,Drama,Singing',
 'Action,Biography,Documentary',
 'Action,Mystery,Thriller',
 'Drama,Mystery,Thriller',
 'Action,Drama,Fantasy',
 'Comedy,Family,Fantasy',
 'Thriller',
 'Adventure,Comedy,Fantasy',
 'Drama,Sci-Fi',
 'Crime,Thriller',
 'Mystery,Sci-Fi,Thriller',
 'Crime,Drama',
 'Adventure,Biography,Comedy',
 'Adventure,Drama,Romance',
 'Comedy,Crime',
 'Drama,Fantasy,Mystery',
 'Biography,Drama,Music',
 'Adventure,Drama',
 'Drama,Mystery,Sci-Fi',
 'Drama,Horror,Sci-Fi',
 'Biography,Drama,History',
 'Sci-Fi,Thriller',
 'Biography,Drama,Fantasy',
 'Drama,Music',
 'Comedy,Documentary',
 'Drama,Romance,War',
 'Mystery,Thriller',
 'Drama,Mystery,Romance',
 'Drama,Mystery',
 'Drama,Family,Sport',
 'Drama,Thriller,Western',
 'Drama,Sci-Fi,Thriller',
 'Action,Crime,Drama',
 'Animation']
```

In [47]:
```
# test the split method on a string
'Action,Biography,Drama'.split(',')
```

Out[47]:
```
['Action', 'Biography', 'Drama']
```

In [48]:
```
# for loop to take each string item in genre and split it by ','
# and put the values into a list of lists
genres_split = []

for string in genres:
    string = string.split(',')
    genres_split.append(string)
genres_split
```

Out[48]:
```
[['Comedy', 'Crime', 'Drama'],
 ['Adventure', 'Family', 'Sci-Fi'],
 ['Animation', 'Family'],
 ['Comedy', 'Sport'],
 ['Comedy', 'Drama', 'Fantasy'],
 ['Comedy', 'Family', 'Romance'],
 ['Fantasy', 'Horror', 'Thriller'],
 ['Family'],
 ['Adventure', 'Comedy', 'Crime'],
 ['Horror', 'Mystery', 'Sci-Fi'],
 ['Romance', 'Sci-Fi', 'Thriller'],
 ['Action', 'Adventure', 'Animation'],
 ['Drama', 'Western'],
```

```
['Drama', 'Singing'],
['Drama', 'Music', 'Thriller'],
['Animation', 'Comedy', 'Family'],
['Action', 'Fantasy', 'Western'],
['Action', 'Adventure', 'Mystery'],
['Comedy', 'Horror'],
['Adventure', 'Drama', 'Thriller'],
['Action', 'Comedy', 'Fantasy'],
['Drama', 'Family'],
['Comedy', 'Fantasy'],
['Action'],
['Drama', 'Music', 'Singing'],
['Action', 'Crime', 'Sci-Fi'],
['Comedy', 'Horror', 'Romance'],
['Comedy', 'Romance'],
['Crime', 'Documentary'],
['Drama', 'Fantasy', 'Romance'],
['Comedy', 'Horror', 'Sci-Fi'],
['Action', 'Drama', 'Mystery'],
['Horror', 'Sci-Fi', 'Thriller'],
['Action', 'Comedy'],
['Biography', 'Drama', 'Sport'],
['Action', 'Drama', 'Family'],
['Adventure', 'Drama', 'Fantasy'],
['Comedy', 'Family'],
['Adventure', 'Drama', 'Western'],
['Biography', 'Documentary'],
['Adventure', 'Comedy', 'Family'],
['Adventure', 'Animation', 'Comedy'],
['Adventure', 'Mystery', 'Sci-Fi'],
['Comedy', 'Drama', 'Romance'],
['Action', 'Comedy', 'Family'],
['Biography', 'Comedy', 'Crime'],
['Comedy', 'Western'],
['Adventure', 'Drama', 'History'],
['Drama', 'War'],
['Action', 'Comedy', 'Sci-Fi'],
['Drama', 'History', 'Mystery'],
['Biography', 'Family', 'Sport'],
['Biography', 'Drama', 'War'],
['Comedy', 'Drama', 'History'],
['Biography', 'Drama', 'Singing'],
['Comedy', 'Crime', 'Romance'],
['Drama', 'Romance', 'Thriller'],
['Biography', 'Crime', 'Drama'],
['Comedy', 'Fantasy', 'Horror'],
['Action', 'Family', 'Fantasy'],
['Comedy', 'Drama', 'Horror'],
['Action', 'Adventure', 'Biography'],
['Action', 'Sci-Fi', 'Thriller'],
['Action', 'Crime', 'Thriller'],
['Action', 'Comedy', 'Sport'],
['Action', 'Adventure', 'Crime'],
['Comedy', 'Mystery'],
['Action', 'Drama', 'Sport'],
['Drama', 'Music', 'Romance'],
['Drama', 'Fantasy', 'Horror'],
['Drama', 'Romance'],
['Crime', 'Drama', 'Thriller'],
['Crime', 'Horror', 'Mystery'],
['Horror', 'Mystery', 'Thriller'],
['Action', 'Drama'],
['Adventure', 'Comedy'],
['Action', 'Adventure', 'Drama'],
['Music'],
```

```
['Drama', 'Horror', 'Mystery'],
['Action', 'Sci-Fi'],
['Adventure', 'Drama', 'Horror'],
['Comedy', 'Music'],
['Drama', 'Mystery', 'War'],
['Action', 'Comedy', 'Documentary'],
['Adventure', 'Horror', 'Mystery'],
['Action', 'Fantasy', 'Thriller'],
['Comedy', 'Drama', 'Sport'],
['Action', 'Drama', 'Thriller'],
['Family', 'Sci-Fi'],
['Horror'],
['Action', 'Adventure', 'Family'],
['Biography', 'Documentary', 'Music'],
['Family', 'Fantasy', 'Singing'],
['Crime', 'Mystery', 'Thriller'],
['Biography', 'Drama', 'Thriller'],
['Action', 'Mystery', 'Sci-Fi'],
['Action', 'Drama', 'War'],
['Action', 'Biography', 'Crime'],
['Action', 'Horror'],
['Biography', 'Comedy', 'Drama'],
['Animation', 'Comedy', 'Drama'],
['Comedy', 'Mystery', 'Sci-Fi'],
['Action', 'Adventure'],
['Drama', 'Thriller'],
['Biography', 'Drama', 'Romance'],
['Comedy', 'Fantasy', 'Romance'],
['Action', 'Comedy', 'Romance'],
['Crime', 'Drama', 'Romance'],
['Adventure', 'Comedy', 'Drama'],
['Action', 'Drama', 'History'],
['Documentary', 'Sport'],
['Action', 'Adventure', 'Western'],
['Comedy', 'Romance', 'Sport'],
['Comedy', 'Drama', 'Mystery'],
['Adventure', 'Comedy', 'Music'],
['Drama', 'Horror', 'Thriller'],
['Drama', 'History', 'Thriller'],
['Drama'],
['Comedy', 'Drama', 'Family'],
['Action', 'Horror', 'Thriller'],
['Adventure', 'Fantasy'],
['Comedy', 'Music', 'War'],
['Romance'],
['Drama', 'Romance', 'Sci-Fi'],
['Documentary'],
['Comedy', 'Drama', 'Music'],
['Horror', 'Thriller'],
['Horror', 'Sci-Fi'],
['Drama', 'Fantasy'],
['Adventure', 'Drama', 'Sci-Fi'],
['Adventure', 'Comedy', 'Romance'],
['Crime', 'Drama', 'Mystery'],
['Horror', 'Mystery'],
['Biography', 'Drama'],
['Action', 'Adventure', 'Sci-Fi'],
['Adventure', 'Comedy', 'Sci-Fi'],
['Action', 'Adventure', 'Comedy'],
['Action', 'Animation', 'Comedy'],
['Action', 'Crime'],
['Comedy', 'Fantasy', 'Singing'],
['Action', 'Crime', 'Horror'],
['Comedy'],
['Adventure', 'Biography', 'Drama'],
```

```
['Action', 'Biography', 'Drama'],
['Comedy', 'Crime', 'Thriller'],
['Comedy', 'Mystery', 'Thriller'],
['Drama', 'Family', 'Fantasy'],
['Action', 'Comedy', 'Drama'],
['Western'],
['Action', 'Comedy', 'Horror'],
['Action', 'Adventure', 'Horror'],
['Singing'],
['Biography', 'Drama', 'Family'],
['Comedy', 'Drama'],
['Drama', 'Sport'],
['Comedy', 'Sci-Fi'],
['Action', 'Fantasy', 'Horror'],
['Crime', 'Drama', 'Horror'],
['Comedy', 'Fantasy', 'Sci-Fi'],
['Drama', 'Family', 'Music'],
['Action', 'Fantasy', 'War'],
['Adventure', 'Family', 'Fantasy'],
['Fantasy', 'Singing'],
['Action', 'Thriller'],
['Documentary', 'Music'],
['Drama', 'Fantasy', 'Music'],
['Drama', 'Horror'],
['Adventure', 'Drama', 'Family'],
['Documentary', 'Drama'],
['Action', 'Comedy', 'Crime'],
['Biography', 'Drama', 'Mystery'],
['Action', 'Adventure', 'Fantasy'],
['Action', 'Horror', 'Sci-Fi'],
['Action', 'Drama', 'Sci-Fi'],
['Action', 'Adventure', 'Thriller'],
['Adventure', 'Animation', 'Family'],
['Action', 'Biography', 'Comedy'],
['Fantasy'],
['Comedy', 'Music', 'Romance'],
['Fantasy', 'Horror', 'Mystery'],
['Action', 'Crime', 'Fantasy'],
['Drama', 'History', 'War'],
['Action', 'Crime', 'Mystery'],
['Comedy', 'Drama', 'Singing'],
['Action', 'Biography', 'Documentary'],
['Action', 'Mystery', 'Thriller'],
['Drama', 'Mystery', 'Thriller'],
['Action', 'Drama', 'Fantasy'],
['Comedy', 'Family', 'Fantasy'],
['Thriller'],
['Adventure', 'Comedy', 'Fantasy'],
['Drama', 'Sci-Fi'],
['Crime', 'Thriller'],
['Mystery', 'Sci-Fi', 'Thriller'],
['Crime', 'Drama'],
['Adventure', 'Biography', 'Comedy'],
['Adventure', 'Drama', 'Romance'],
['Comedy', 'Crime'],
['Drama', 'Fantasy', 'Mystery'],
['Biography', 'Drama', 'Music'],
['Adventure', 'Drama'],
['Drama', 'Mystery', 'Sci-Fi'],
['Drama', 'Horror', 'Sci-Fi'],
['Biography', 'Drama', 'History'],
['Sci-Fi', 'Thriller'],
['Biography', 'Drama', 'Fantasy'],
['Drama', 'Music'],
['Comedy', 'Documentary'],
```

```
['Drama', 'Romance', 'War'],
['Mystery', 'Thriller'],
['Drama', 'Mystery', 'Romance'],
['Drama', 'Mystery'],
['Drama', 'Family', 'Sport'],
['Drama', 'Thriller', 'Western'],
['Drama', 'Sci-Fi', 'Thriller'],
['Action', 'Crime', 'Drama'],
['Animation']]
```

In [49]:
```
# create a set to add all the unique genres too
unique_genres = set()

# for loop to take each item in the genres_split and
# add it to the unique_genres list. End result should be
# list of all unique genres that appear in the 'genres' column
for item in genres_split:
    for genre in item:
        unique_genres.add(genre)

unique_genres = list(unique_genres)
unique_genres
```

Out[49]:
```
['War',
 'Sci-Fi',
 'Family',
 'Sport',
 'Adventure',
 'Fantasy',
 'Drama',
 'Romance',
 'Action',
 'Mystery',
 'Documentary',
 'Crime',
 'Music',
 'Thriller',
 'Biography',
 'History',
 'Comedy',
 'Western',
 'Singing',
 'Horror',
 'Animation']
```

In [50]:
```
len(unique_genres)
```

Out[50]: 21

In [51]:
```
# Test conditional
'Horror' in 'Action,Crime,Horror'
```

Out[51]: True

Now I have a list of all the data frames I'll create a new df_genres to contain the movie_id, title_norm, and genres columns

In [52]:
```
df_movie_info_budget.head()
```

Out[52]:

	movie_id	primary_title	original_title	start_year	genres	average_rating	num_votes

	movie_id	primary_title	original_title	start_year	genres	average_rating	num_votes
0	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
1	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
2	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
3	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
4	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813

In [53]:
```
relevant_columns = ['movie_id', 'title_norm', 'genres']
df_genres = df_movie_info_budget[relevant_columns]
```

In [54]:
```
df_genres.head(20)
```

Out[54]:

	movie_id	title_norm	genres
0	tt1014759	alice in wonderland 2010	Adventure,Family,Fantasy
1	tt1014759	alice in wonderland 2010	Adventure,Family,Fantasy
2	tt1014759	alice in wonderland 2010	Adventure,Family,Fantasy
3	tt1014759	alice in wonderland 2010	Adventure,Family,Fantasy
4	tt1014759	alice in wonderland 2010	Adventure,Family,Fantasy
5	tt1014759	alice in wonderland 2010	Adventure,Family,Fantasy
6	tt1014759	alice in wonderland 2010	Adventure,Family,Fantasy
7	tt1014759	alice in wonderland 2010	Adventure,Family,Fantasy
8	tt1014759	alice in wonderland 2010	Adventure,Family,Fantasy
9	tt1014759	alice in wonderland 2010	Adventure,Family,Fantasy
30	tt2049386	alice in wonderland 2010	Fantasy,Singing
31	tt2049386	alice in wonderland 2010	Fantasy,Singing
32	tt2049386	alice in wonderland 2010	Fantasy,Singing
33	tt2049386	alice in wonderland 2010	Fantasy,Singing
34	tt2049386	alice in wonderland 2010	Fantasy,Singing
35	tt2049386	alice in wonderland 2010	Fantasy,Singing
36	tt2049386	alice in wonderland 2010	Fantasy,Singing

	movie_id	title_norm	genres
37	tt1294226	the last song 2010	Drama,Music,Romance
38	tt1294226	the last song 2010	Drama,Music,Romance
39	tt1294226	the last song 2010	Drama,Music,Romance

Now I want to create new columns with the list of genres and add a 1 if that movie contains that genre and 0 if it doesn't

In [55]:
```
# for loop to create new columns in df_genre
for genre in unique_genres:
    df_genres[genre] = np.where(df_genres['genres'].str.contains(genre), 1, 0)
#     if genre in df_genres[genres]: 1 else: 0
```

```
<ipython-input-55-4338c4b62e2d>:3: SettingWithCopyWarning:
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/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy
  df_genres[genre] = np.where(df_genres['genres'].str.contains(genre), 1, 0)
```

In [56]:
```
df_genres
```

Out[56]:

	movie_id	title_norm	genres	War	Sci-Fi	Family	Sport	Adventure	Fantasy
0	tt1014759	alice in wonderland 2010	Adventure,Family,Fantasy	0	0	1	0	1	1
1	tt1014759	alice in wonderland 2010	Adventure,Family,Fantasy	0	0	1	0	1	1
2	tt1014759	alice in wonderland 2010	Adventure,Family,Fantasy	0	0	1	0	1	1
3	tt1014759	alice in wonderland 2010	Adventure,Family,Fantasy	0	0	1	0	1	1
4	tt1014759	alice in wonderland 2010	Adventure,Family,Fantasy	0	0	1	0	1	1
...	...	...	...	...	...	...	...	...	...
20410	tt6139732	aladdin 2019	Adventure,Comedy,Family	0	0	1	0	1	0
20411	tt6139732	aladdin 2019	Adventure,Comedy,Family	0	0	1	0	1	0
20412	tt6139732	aladdin 2019	Adventure,Comedy,Family	0	0	1	0	1	0
20413	tt6139732	aladdin 2019	Adventure,Comedy,Family	0	0	1	0	1	0

	movie_id	title_norm	genres	War	Sci-Fi	Family	Sport	Adventure	Fantasy
20414	tt6139732	aladdin 2019	Adventure,Comedy,Family	0	0	1	0	1	0

12689 rows × 24 columns

In [57]:
```
# sanity check. I'll add a new column that adds all the rows accross
df_genres['total_genre'] = df_genres.iloc[:,2:].sum(axis=1)
```

In [58]:
```
df_genres
```

Out[58]:

	movie_id	title_norm	genres	War	Sci-Fi	Family	Sport	Adventure	Fantasy
0	tt1014759	alice in wonderland 2010	Adventure,Family,Fantasy	0	0	1	0	1	1
1	tt1014759	alice in wonderland 2010	Adventure,Family,Fantasy	0	0	1	0	1	1
2	tt1014759	alice in wonderland 2010	Adventure,Family,Fantasy	0	0	1	0	1	1
3	tt1014759	alice in wonderland 2010	Adventure,Family,Fantasy	0	0	1	0	1	1
4	tt1014759	alice in wonderland 2010	Adventure,Family,Fantasy	0	0	1	0	1	1
...	...	...	...	...	...	...	...	...	...
20410	tt6139732	aladdin 2019	Adventure,Comedy,Family	0	0	1	0	1	0
20411	tt6139732	aladdin 2019	Adventure,Comedy,Family	0	0	1	0	1	0
20412	tt6139732	aladdin 2019	Adventure,Comedy,Family	0	0	1	0	1	0
20413	tt6139732	aladdin 2019	Adventure,Comedy,Family	0	0	1	0	1	0
20414	tt6139732	aladdin 2019	Adventure,Comedy,Family	0	0	1	0	1	0

12689 rows × 25 columns

In [59]:
```
# finds the counts of the totals of the genres per movie
df_genres['total_genre'].value_counts()
```

Out[59]:
```
3    8561
2    2967
1    1161
Name: total_genre, dtype: int64
```

In [60]:
```
# check to see if a movie has 4 genres (it shouldn't)
df_genres.loc[df_genres['total_genre'] == 4]
```

Out[60]:

movie_id	title_norm	genres	War	Sci-Fi	Family	Sport	Adventure	Fantasy	Drama	...	Music	Thril

0 rows × 25 columns

In [61]:
```
# Take a look at the movies that Musical was
# replaced by Singing
df_genres.loc[df_genres['Singing'] == 1]
```

Out[61]:

	movie_id	title_norm	genres	War	Sci-Fi	Family	Sport	Adventure	Fantasy	I
30	tt2049386	alice in wonderland 2010	Fantasy,Singing	0	0	0	0	0	1	
31	tt2049386	alice in wonderland 2010	Fantasy,Singing	0	0	0	0	0	1	
32	tt2049386	alice in wonderland 2010	Fantasy,Singing	0	0	0	0	0	1	
33	tt2049386	alice in wonderland 2010	Fantasy,Singing	0	0	0	0	0	1	
34	tt2049386	alice in wonderland 2010	Fantasy,Singing	0	0	0	0	0	1	
...	...	...	...	...	...	...	...	...	...	...
17074	tt1485796	the greatest showman 2017	Biography,Drama,Singing	0	0	0	0	0	0	
17075	tt1485796	the greatest showman 2017	Biography,Drama,Singing	0	0	0	0	0	0	
17076	tt1485796	the greatest showman 2017	Biography,Drama,Singing	0	0	0	0	0	0	

	movie_id	title_norm	genres	War	Sci-Fi	Family	Sport	Adventure	Fantasy	I
17077	tt1485796	the greatest showman 2017	Biography,Drama,Singing	0	0	0	0	0	0	
17078	tt1485796	the greatest showman 2017	Biography,Drama,Singing	0	0	0	0	0	0	

66 rows × 25 columns

Everything looks great now. next step is to drop total_genre column and transform this data from wide to long format

Tansforming the data to long format will allow me analyze the data based on the genre later.

In [62]:
```
# drop the total_genre column
df_genres.drop(columns='total_genre', inplace=True)
```

In [63]:
```
# get the columns we're looking to transform to long format
list(df_genres.iloc[:,3:].columns)
```

Out[63]:
```
['War',
 'Sci-Fi',
 'Family',
 'Sport',
 'Adventure',
 'Fantasy',
 'Drama',
 'Romance',
 'Action',
 'Mystery',
 'Documentary',
 'Crime',
 'Music',
 'Thriller',
 'Biography',
 'History',
 'Comedy',
 'Western',
 'Singing',
 'Horror',
 'Animation']
```

In [64]:
```
# Now to change to long format from wide
cols_to_keep = ['movie_id', 'title_norm', 'genres']
cols_to_move = list(df_genres.iloc[:,3:].columns)

df_genre_long = pd.melt(df_genres,
                        id_vars=cols_to_keep,
                        value_vars=cols_to_move,
                        var_name='genre',
                        value_name='genre_true_or_false')
```

In [65]:
```
# take a look at the new format
df_genre_long.head(15)
```

Out[65]:

	movie_id	title_norm	genres	genre	genre_true_or_false
0	tt1014759	alice in wonderland 2010	Adventure,Family,Fantasy	War	0
1	tt1014759	alice in wonderland 2010	Adventure,Family,Fantasy	War	0
2	tt1014759	alice in wonderland 2010	Adventure,Family,Fantasy	War	0
3	tt1014759	alice in wonderland 2010	Adventure,Family,Fantasy	War	0
4	tt1014759	alice in wonderland 2010	Adventure,Family,Fantasy	War	0
5	tt1014759	alice in wonderland 2010	Adventure,Family,Fantasy	War	0
6	tt1014759	alice in wonderland 2010	Adventure,Family,Fantasy	War	0
7	tt1014759	alice in wonderland 2010	Adventure,Family,Fantasy	War	0
8	tt1014759	alice in wonderland 2010	Adventure,Family,Fantasy	War	0
9	tt1014759	alice in wonderland 2010	Adventure,Family,Fantasy	War	0
10	tt2049386	alice in wonderland 2010	Fantasy,Singing	War	0
11	tt2049386	alice in wonderland 2010	Fantasy,Singing	War	0
12	tt2049386	alice in wonderland 2010	Fantasy,Singing	War	0
13	tt2049386	alice in wonderland 2010	Fantasy,Singing	War	0
14	tt2049386	alice in wonderland 2010	Fantasy,Singing	War	0

Now the next step is to drop all the rows with 0 in the genre_true_or_false column

This is because we want to keep all the rows with 1s because that means the movie is of that genre

In [66]:
```
# create new dataframe with only 1s
df_genre_long = df_genre_long.loc[df_genre_long['genre_true_or_false'] == 1]
```

In [67]:
```
# now I'll drop duplicate values
df_genre_long.drop_duplicates(inplace=True)
```

In [68]:
```
# sort values by title_norm column
df_genre_long.sort_values(by='title_norm', inplace=True)
```

In [69]:
```
# drop genres and genre_true_or_false columns because they are unnecessary
df_genre_long.drop(columns=['genres', 'genre_true_or_false'], inplace=True)

df_genre_long
```

Out[69]:

	movie_id	title_norm	genre
249762	tt3526286	#horror 2015	Horror
84805	tt3526286	#horror 2015	Drama
148250	tt3526286	#horror 2015	Crime

	movie_id	title_norm	genre
249996	tt1179933	10 cloverfield lane 2016	Horror
85039	tt1179933	10 cloverfield lane 2016	Drama
...	...	...	...
205279	tt1222817	zookeeper 2011	Comedy
212016	tt1608290	zoolander 2 2016	Comedy
212740	tt2948356	zootopia 2016	Comedy
263496	tt2948356	zootopia 2016	Animation
60472	tt2948356	zootopia 2016	Adventure

3472 rows × 3 columns

Now that we have a long format genre table we can merge it with the movie info dataframe

In [70]:
```
df_movie_info_budget.head()
```

Out[70]:

	movie_id	primary_title	original_title	start_year	genres	average_rating	num_votes
0	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
1	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
2	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
3	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
4	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813

In [71]:
```
df_genre_long.head()
```

Out[71]:

	movie_id	title_norm	genre
249762	tt3526286	#horror 2015	Horror
84805	tt3526286	#horror 2015	Drama
148250	tt3526286	#horror 2015	Crime
249996	tt1179933	10 cloverfield lane 2016	Horror

	movie_id	title_norm	genre
85039	tt1179933	10 cloverfield lane 2016	Drama

In [72]:
```
# merging the two dataframes together
df_movie_genre_long =  df_movie_info_budget.merge(df_genre_long,
                                                  how='inner',
                                                  suffixes=('_movie_info', '_genre_lo
                                                  on='movie_id')

df_movie_genre_long.head()
```

Out[72]:

	movie_id	primary_title	original_title	start_year	genres	average_rating	num_votes
0	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
1	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
2	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
3	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
4	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813

In [73]:
```
# check for duplicates
df_movie_genre_long.loc[df_movie_genre_long.duplicated()]
```

Out[73]:

	movie_id	primary_title	original_title	start_year	genres	average_rating	num_votes	persons_name	

In [74]:
```
# check out the duplicates more
df_movie_genre_long.loc[df_movie_genre_long['primary_title'] == 'Aladdin']
```

Out[74]:

	movie_id	primary_title	original_title	start_year	genres	average_rating	num_
32751	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
32752	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
32753	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
32754	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
32755	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
32756	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5

	movie_id	primary_title	original_title	start_year	genres	average_rating	num_
32757	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
32758	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
32759	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
32760	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
32761	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
32762	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
32763	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
32764	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
32765	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
32766	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
32767	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
32768	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
32769	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
32770	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
32771	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
32772	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
32773	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
32774	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
32775	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
32776	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
32777	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5

In [75]:
```
# remove duplicates
df_movie_genre_long.drop_duplicates(inplace=True)
```

In [76]:
```
df_movie_genre_long.head()
```

Out[76]:

	movie_id	primary_title	original_title	start_year	genres	average_rating	num_votes
0	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
1	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
2	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
3	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
4	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813

In [77]:
```
# check out final data frame
print(df_movie_genre_long.shape)
df_movie_genre_long.head(10)
```

```
(32778, 18)
```

Out[77]:

	movie_id	primary_title	original_title	start_year	genres	average_rating	num_votes
0	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
1	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
2	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
3	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
4	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
5	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
6	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
7	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
8	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
9	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813

In [78]:
```
# check for any inconsistency
df_movie_genre_long.loc[df_movie_genre_long['primary_title'] == 'Alice in Wonderland']
```

Out[78]:

	movie_id	primary_title	original_title	start_year	genres	average_rating	num_vote
0	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
1	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
2	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
3	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
4	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
5	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
6	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
7	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
8	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
9	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
10	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
11	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
12	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
13	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
14	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
15	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
16	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
17	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
18	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
19	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881

	movie_id	primary_title	original_title	start_year	genres	average_rating	num_vote
20	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
21	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
22	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
23	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
24	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
25	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
26	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
27	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
28	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
29	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35881
30	tt2049386	Alice in Wonderland	Alice in Wonderland	2010	Fantasy,Singing	6	6
31	tt2049386	Alice in Wonderland	Alice in Wonderland	2010	Fantasy,Singing	6	6
32	tt2049386	Alice in Wonderland	Alice in Wonderland	2010	Fantasy,Singing	6	6
33	tt2049386	Alice in Wonderland	Alice in Wonderland	2010	Fantasy,Singing	6	6
34	tt2049386	Alice in Wonderland	Alice in Wonderland	2010	Fantasy,Singing	6	6
35	tt2049386	Alice in Wonderland	Alice in Wonderland	2010	Fantasy,Singing	6	6
36	tt2049386	Alice in Wonderland	Alice in Wonderland	2010	Fantasy,Singing	6	6
37	tt2049386	Alice in Wonderland	Alice in Wonderland	2010	Fantasy,Singing	6	6
38	tt2049386	Alice in Wonderland	Alice in Wonderland	2010	Fantasy,Singing	6	6
39	tt2049386	Alice in Wonderland	Alice in Wonderland	2010	Fantasy,Singing	6	6
40	tt2049386	Alice in Wonderland	Alice in Wonderland	2010	Fantasy,Singing	6	6

	movie_id	primary_title	original_title	start_year	genres	average_rating	num_vote
41	tt2049386	Alice in Wonderland	Alice in Wonderland	2010	Fantasy,Singing	6	
42	tt2049386	Alice in Wonderland	Alice in Wonderland	2010	Fantasy,Singing	6	
43	tt2049386	Alice in Wonderland	Alice in Wonderland	2010	Fantasy,Singing	6	

Data Preparation and Cleaning is done for the most part

Here are the two data frames we are left with

In [79]:
```
df_movie_genre_long
```

Out[79]:

	movie_id	primary_title	original_title	start_year	genres	average_rating	num_
0	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	3
1	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	3
2	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	3
3	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	3
4	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	3
...	...	...	...	...	...	...	
32773	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	
32774	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	
32775	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	
32776	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	
32777	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	

32778 rows × 18 columns

In [80]:
```
df_movie_info_budget
```

Out[80]:

	movie_id	primary_title	original_title	start_year	genres	average_rating	num_
0	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35
1	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35
2	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35
3	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35
4	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35
...	...	...	...	...	...	...	
20410	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
20411	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
20412	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
20413	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5
20414	tt6139732	Aladdin	Aladdin	2019	Adventure,Comedy,Family	7	5

12689 rows × 16 columns

In [81]:
```
!ls
```

```
RT.ipynb
all_data_copy.ipynb
all_data_preparation.ipynb
budget_v_profit.ipynb
data_preparation.ipynb
genre_v_boxoffice.ipynb
genre_v_budget.ipynb
imdb_first_look.ipynb
talent_v_revenue.ipynb
the_movies_db_api.ipynb
```

In [82]:
```
# save both dataframes as csv files
df_movie_genre_long.to_csv('../data/movie_genre_long.csv')
df_movie_info_budget.to_csv('../data/movie_info_budget.csv')
```

Adding New API Data From The Movie

Database

Our group fetched updated data from the The Movie Database. This includes updated budget and worldwide revenue for the movies. I am going to add this new data to our df_movie_genre_long dataframe from above. And save the final dataframe we will use for analysis as 'all_data.csv'

In [83]:
```
# grabbing the data
df_movie_genre_long = pd.read_csv('../data/movie_genre_long.csv', index_col=0)
df_tmdb = pd.read_csv('../data/tmdb_filtered.csv', index_col=0)

# sanity check
df_tmdb.head()
```

Out[83]:

	budget	genres	id	imdb_id	original_title	release_date	revenue	vote_average	vote
0	250000000	[{'id': 12, 'name': 'Adventure'}, {'id': 14, '...	12444	tt0926084	Harry Potter and the Deathly Hallows: Part 1	2010-10-17	954305868	8	
1	165000000	[{'id': 14, 'name': 'Fantasy'}, {'id': 12, 'na...	10191	tt0892769	How to Train Your Dragon	2010-03-18	494878759	8	
2	200000000	[{'id': 12, 'name': 'Adventure'}, {'id': 28, '...	10138	tt1228705	Iron Man 2	2010-04-28	623933331	7	
3	30000000	[{'id': 16, 'name': 'Animation'}, {'id': 12, '...	862	tt0114709	Toy Story	1995-10-30	373554033	8	
4	160000000	[{'id': 28, 'name': 'Action'}, {'id': 878, 'na...	27205	tt1375666	Inception	2010-07-15	825532764	8	

In [84]:
```
# filtering extraneous columns and changing column
# names so they differentiate from df_movie_genre_long columns
df_tmdb_filtered = df_tmdb[['budget',
                            'revenue',
                            'imdb_id',
                            'vote_average',
                            'vote_count']]

df_tmdb_filtered.rename(columns={'budget':'new_budget_api',
                                 'revenue':'new_ww_revenue_api'},
                                  inplace=True)

df_tmdb_filtered.head()
```

```
C:\Users\ghall\anaconda3\envs\learn-env\lib\site-packages\pandas\core\frame.py:4296: Set
tingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy
  return super().rename(
```

Out[84]:

	new_budget_api	new_ww_revenue_api	imdb_id	vote_average	vote_count
0	250000000	954305868	tt0926084	8	16170
1	165000000	494878759	tt0892769	8	11036
2	200000000	623933331	tt1228705	7	18118
3	30000000	373554033	tt0114709	8	15570
4	160000000	825532764	tt1375666	8	31837

In [85]:
```
# this shows us how many unique movies there are
# in the dataframe
len(df_movie_genre_long['movie_id'].unique())
```

Out[85]:
```
1357
```

In [86]:
```
# a look at the columns
df_movie_genre_long.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 32778 entries, 0 to 32777
Data columns (total 18 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   movie_id               32778 non-null  object
 1   primary_title          32778 non-null  object
 2   original_title         32778 non-null  object
 3   start_year             32778 non-null  int64
 4   genres                 32778 non-null  object
 5   average_rating         32778 non-null  float64
 6   num_votes              32778 non-null  int64
 7   persons_name           32778 non-null  object
 8   persons_job            32778 non-null  object
 9   title_norm_movie_info  32778 non-null  object
 10  id                     32778 non-null  int64
 11  release_date           32778 non-null  object
 12  movie                  32778 non-null  object
 13  production_budget      32778 non-null  int64
 14  domestic_gross         32778 non-null  int64
 15  worldwide_gross        32778 non-null  int64
 16  title_norm_genre_long  32778 non-null  object
 17  genre                  32778 non-null  object
dtypes: float64(1), int64(6), object(11)
memory usage: 4.8+ MB
```

In [87]:
```
# merge the two dataframes
df_all_data = df_movie_genre_long.merge(df_tmdb_filtered,
                          how='inner',
                          left_on='movie_id',
                          right_on='imdb_id')

df_all_data.head()
```

Out[87]:

	movie_id	primary_title	original_title	start_year	genres	average_rating	num_votes
0	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
1	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
2	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
3	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813
4	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	358813

5 rows × 23 columns

I'll add a new column budget_diff to see any difference between the new budget data we collected the previous budget data

In [88]:
```
# create new column budget_diff
df_all_data['budget_diff'] = df_all_data['new_budget_api'] - df_all_data['production_bu

df_all_data
```

Out[88]:

	movie_id	primary_title	original_title	start_year	genres	average_rating	num_v
0	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35
1	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35
2	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35
3	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35
4	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35
...	...	...	...	...	...	...	
34028	tt3829266	The Predator	The Predator	2018	Action,Adventure,Sci-Fi	5	9
34029	tt3829266	The Predator	The Predator	2018	Action,Adventure,Sci-Fi	5	9
34030	tt3829266	The Predator	The Predator	2018	Action,Adventure,Sci-Fi	5	9
34031	tt3829266	The Predator	The Predator	2018	Action,Adventure,Sci-Fi	5	9
34032	tt3829266	The Predator	The Predator	2018	Action,Adventure,Sci-Fi	5	9

34033 rows × 24 columns

Now I want to check the mean of the absolute difference of the budget_diff column. This will give us an idea on average how off the old data was from the new correct data. I'll create a new test df called 'x' to filter out the dataframe where budget_diff does not equal 0 AND new_budget_api does not equal 0. Then I will find the absolute value of the column values and find the mean from there.

In [89]:
```
# create new test dataframe called 'x'
x = df_all_data.loc[(df_all_data['budget_diff'] != 0)  \
                    & (df_all_data['new_budget_api'] != 0)]\
                    .groupby(['movie_id','primary_title']).mean().reset_index()

# Mean of the absolute value of the budget differnce
x['budget_diff'].abs().mean()
```

Out[89]: 9001177.188191881

In [90]:
```
# sanity check
x.head(10)
```

Out[90]:

	movie_id	primary_title	start_year	average_rating	num_votes	id	production_budget	domestic_gro
0	tt0249516	Foodfight!	2012	2	8248	26	45000000	
1	tt0359950	The Secret Life of Walter Mitty	2013	7	275300	37	91000000	582368
2	tt0369610	Jurassic World	2015	7	539338	34	215000000	6522706
3	tt0401729	John Carter	2012	7	241792	14	275000000	730586
4	tt0451279	Wonder Woman	2017	8	487527	55	150000000	4125634
5	tt0453562	42	2013	8	77703	22	31000000	950202
6	tt0455407	The Crazies	2010	6	104465	36	19000000	391235
7	tt0471042	Tower Heist	2011	6	125102	5	85000000	780465
8	tt0472399	The Mechanic	2011	7	141254	42	42500000	291214
9	tt0477080	Unstoppable	2010	7	173019	19	95000000	815629

In [91]:
```
# this will give us the names of the movies where
# the budget difference isn't 0
x.merge(df_all_data[['movie_id', 'primary_title']],
        how='left',
        on='movie_id')
```

Out[91]:

	movie_id	primary_title_x	start_year	average_rating	num_votes	id	production_budget	domest
0	tt0249516	Foodfight!	2012	2	8248	26	45000000	

	movie_id	primary_title_x	start_year	average_rating	num_votes	id	production_budget	domest
1	tt0249516	Foodfight!	2012	2	8248	26	45000000	
2	tt0249516	Foodfight!	2012	2	8248	26	45000000	
3	tt0249516	Foodfight!	2012	2	8248	26	45000000	
4	tt0249516	Foodfight!	2012	2	8248	26	45000000	
...	...	...	...	...	...	...	...	
7418	tt7334528	Uncle Drew	2018	6	9739	85	18000000	4
7419	tt7334528	Uncle Drew	2018	6	9739	85	18000000	4
7420	tt7334528	Uncle Drew	2018	6	9739	85	18000000	4
7421	tt7334528	Uncle Drew	2018	6	9739	85	18000000	4
7422	tt7334528	Uncle Drew	2018	6	9739	85	18000000	4

7423 rows × 15 columns

Conclusion: The old data has values that are not valid for the most part. Going forward with our analysis we will use the columns with the new data collected from the The Movie Database api.

In [92]:
```
df_all_data.drop(columns='budget_diff',inplace=True)
```

In [93]:
```
# final check of the data
df_all_data
```

Out[93]:

	movie_id	primary_title	original_title	start_year	genres	average_rating	num_v
0	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35
1	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35
2	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35
3	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35
4	tt1014759	Alice in Wonderland	Alice in Wonderland	2010	Adventure,Family,Fantasy	6	35
...	...	...	...	...	...	...	
34028	tt3829266	The Predator	The Predator	2018	Action,Adventure,Sci-Fi	5	9
34029	tt3829266	The Predator	The Predator	2018	Action,Adventure,Sci-Fi	5	9
34030	tt3829266	The Predator	The Predator	2018	Action,Adventure,Sci-Fi	5	9

	movie_id	primary_title	original_title	start_year	genres	average_rating	num_v
34031	tt3829266	The Predator	The Predator	2018	Action,Adventure,Sci-Fi	5	9
34032	tt3829266	The Predator	The Predator	2018	Action,Adventure,Sci-Fi	5	9

34033 rows × 23 columns

Now we have the dataframe we are going to use for our analysis

In [94]:
```
# save this dataframe as a csv
df_all_data.to_csv('../data/all_data.csv')
```