Final Project Submission

Please fill out:

- Student name: Ryan Sung Hwa Chung
- Student pace: Part Time
- Scheduled project review date/time: 6/5/2021
- Instructor name: Yish
- Blog post URL:

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

Importing Data Sets 1-4

In [2]:
```
movies_df = pd.read_csv('C://Users//rychu//Documents//Flatiron//Phase-1-Project-Docs//zippedData//b
title_df = pd.read_csv('C://Users//rychu//Documents//Flatiron//Phase-1-Project-Docs//zippedData//im
ratings_df = pd.read_csv('C://Users//rychu//Documents//Flatiron//Phase-1-Project-Docs//zippedData//
crew_df = pd.read_csv('C://Users//rychu//Documents//Flatiron//Phase-1-Project-Docs//zippedData//imd
```

In [3]:
```
movies_df.info()
title_df.info()
ratings_df.info()
crew_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3387 entries, 0 to 3386
Data columns (total 5 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   title           3387 non-null   object 
 1   studio          3382 non-null   object 
 2   domestic_gross  3359 non-null   float64
 3   foreign_gross   2037 non-null   object 
 4   year            3387 non-null   int64  
dtypes: float64(1), int64(1), object(3)
memory usage: 132.4+ KB
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 146144 entries, 0 to 146143
Data columns (total 6 columns):
 #   Column           Non-Null Count   Dtype  
---  ------           --------------   -----  
 0   tconst           146144 non-null  object 
 1   primary_title    146144 non-null  object 
 2   original_title   146123 non-null  object 
 3   start_year       146144 non-null  int64  
 4   runtime_minutes  114405 non-null  float64
 5   genres           140736 non-null  object 
dtypes: float64(1), int64(1), object(4)
memory usage: 6.7+ MB
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 73856 entries, 0 to 73855
Data columns (total 3 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   tconst         73856 non-null  object 
 1   averagerating  73856 non-null  float64
 2   numvotes       73856 non-null  int64  
dtypes: float64(1), int64(1), object(1)
```

```
memory usage: 1.7+ MB
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 146144 entries, 0 to 146143
Data columns (total 3 columns):
 #   Column     Non-Null Count   Dtype 
---  ------     --------------   ----- 
 0   tconst     146144 non-null  object
 1   directors  140417 non-null  object
 2   writers    110261 non-null  object
dtypes: object(3)
memory usage: 3.3+ MB
```

In [4]:
```
display(movies_df.head())
movies_df.head()
```

	title	studio	domestic_gross	foreign_gross	year
0	Toy Story 3	BV	415000000.0	652000000	2010
1	Alice in Wonderland (2010)	BV	334200000.0	691300000	2010
2	Harry Potter and the Deathly Hallows Part 1	WB	296000000.0	664300000	2010
3	Inception	WB	292600000.0	535700000	2010
4	Shrek Forever After	P/DW	238700000.0	513900000	2010

Out[4]:

	title	studio	domestic_gross	foreign_gross	year
0	Toy Story 3	BV	415000000.0	652000000	2010
1	Alice in Wonderland (2010)	BV	334200000.0	691300000	2010
2	Harry Potter and the Deathly Hallows Part 1	WB	296000000.0	664300000	2010
3	Inception	WB	292600000.0	535700000	2010
4	Shrek Forever After	P/DW	238700000.0	513900000	2010

In [5]:
```
display(title_df.head())
title_df.head()
```

	tconst	primary_title	original_title	start_year	runtime_minutes	genres
0	tt0063540	Sunghursh	Sunghursh	2013	175.0	Action,Crime,Drama
1	tt0066787	One Day Before the Rainy Season	Ashad Ka Ek Din	2019	114.0	Biography,Drama
2	tt0069049	The Other Side of the Wind	The Other Side of the Wind	2018	122.0	Drama
3	tt0069204	Sabse Bada Sukh	Sabse Bada Sukh	2018	NaN	Comedy,Drama
4	tt0100275	The Wandering Soap Opera	La Telenovela Errante	2017	80.0	Comedy,Drama,Fantasy

Out[5]:

	tconst	primary_title	original_title	start_year	runtime_minutes	genres
0	tt0063540	Sunghursh	Sunghursh	2013	175.0	Action,Crime,Drama
1	tt0066787	One Day Before the Rainy Season	Ashad Ka Ek Din	2019	114.0	Biography,Drama
2	tt0069049	The Other Side of the Wind	The Other Side of the Wind	2018	122.0	Drama
3	tt0069204	Sabse Bada Sukh	Sabse Bada Sukh	2018	NaN	Comedy,Drama
4	tt0100275	The Wandering Soap Opera	La Telenovela Errante	2017	80.0	Comedy,Drama,Fantasy

In [6]:
```
display(ratings_df.head())
ratings_df.head()
```

	tconst	averagerating	numvotes
0	tt10356526	8.3	31
1	tt10384606	8.9	559
2	tt1042974	6.4	20
3	tt1043726	4.2	50352
4	tt1060240	6.5	21

Out[6]:

	tconst	averagerating	numvotes
0	tt10356526	8.3	31
1	tt10384606	8.9	559
2	tt1042974	6.4	20
3	tt1043726	4.2	50352
4	tt1060240	6.5	21

In [7]:
```
display(crew_df.head())
crew_df.head()
```

	tconst	directors	writers
0	tt0285252	nm0899854	nm0899854
1	tt0438973	NaN	nm0175726,nm1802864
2	tt0462036	nm1940585	nm1940585
3	tt0835418	nm0151540	nm0310087,nm0841532
4	tt0878654	nm0089502,nm2291498,nm2292011	nm0284943

Out[7]:

	tconst	directors	writers
0	tt0285252	nm0899854	nm0899854
1	tt0438973	NaN	nm0175726,nm1802864
2	tt0462036	nm1940585	nm1940585
3	tt0835418	nm0151540	nm0310087,nm0841532
4	tt0878654	nm0089502,nm2291498,nm2292011	nm0284943

In [8]:
```
display(movies_df.describe())
movies_df.describe()
```

	domestic_gross	year
count	3.359000e+03	3387.000000
mean	2.874585e+07	2013.958075
std	6.698250e+07	2.478141
min	1.000000e+02	2010.000000
25%	1.200000e+05	2012.000000
50%	1.400000e+06	2014.000000

	domestic_gross	year
75%	2.790000e+07	2016.000000
max	9.367000e+08	2018.000000

Out[8]:

	domestic_gross	year
count	3.359000e+03	3387.000000
mean	2.874585e+07	2013.958075
std	6.698250e+07	2.478141
min	1.000000e+02	2010.000000
25%	1.200000e+05	2012.000000
50%	1.400000e+06	2014.000000
75%	2.790000e+07	2016.000000
max	9.367000e+08	2018.000000

In [9]:
```
display(title_df.describe())
title_df.describe()
```

	start_year	runtime_minutes
count	146144.000000	114405.000000
mean	2014.621798	86.187247
std	2.733583	166.360590
min	2010.000000	1.000000
25%	2012.000000	70.000000
50%	2015.000000	87.000000
75%	2017.000000	99.000000
max	2115.000000	51420.000000

Out[9]:

	start_year	runtime_minutes
count	146144.000000	114405.000000
mean	2014.621798	86.187247
std	2.733583	166.360590
min	2010.000000	1.000000
25%	2012.000000	70.000000
50%	2015.000000	87.000000
75%	2017.000000	99.000000
max	2115.000000	51420.000000

In [10]:
```
display(ratings_df.describe())
ratings_df.describe()
```

	averagerating	numvotes
count	73856.000000	7.385600e+04

	averagerating	numvotes
mean	6.332729	3.523662e+03
std	1.474978	3.029402e+04
min	1.000000	5.000000e+00
25%	5.500000	1.400000e+01
50%	6.500000	4.900000e+01
75%	7.400000	2.820000e+02
max	10.000000	1.841066e+06

Out[10]:

	averagerating	numvotes
count	73856.000000	7.385600e+04
mean	6.332729	3.523662e+03
std	1.474978	3.029402e+04
min	1.000000	5.000000e+00
25%	5.500000	1.400000e+01
50%	6.500000	4.900000e+01
75%	7.400000	2.820000e+02
max	10.000000	1.841066e+06

In [11]:
```
display(crew_df.describe())
crew_df.describe()
```

	tconst	directors	writers
count	146144	140417	110261
unique	146144	98525	91920
top	tt2992674	nm3266654	nm0000636
freq	1	62	80

Out[11]:

	tconst	directors	writers
count	146144	140417	110261
unique	146144	98525	91920
top	tt2992674	nm3266654	nm0000636
freq	1	62	80

Cleaning 1st Dataframe

In [12]:
```
movies_df.isna().any()
```

Out[12]:
```
title             False
studio             True
domestic_gross     True
foreign_gross      True
year              False
dtype: bool
```

In [13]:
```
movies_df.isna().sum()
```

Out[13]:
```
title               0
studio              5
domestic_gross     28
foreign_gross    1350
year                0
dtype: int64
```

In [14]:
```
movies_df = movies_df.drop('foreign_gross', axis = 1)
```

In [15]:
```
movies_df.isna().sum()
```

Out[15]:
```
title             0
studio            5
domestic_gross   28
year              0
dtype: int64
```

In [16]:
```
mean_domestic_gross = movies_df['domestic_gross'].mean()
movies_df['domestic_gross'].fillna(mean_domestic_gross, inplace=True)
movies_df.isna().sum()
```

Out[16]:
```
title            0
studio           5
domestic_gross   0
year             0
dtype: int64
```

In [17]:
```
movies_df['studio'].fillna(value="N/A", inplace=True)
movies_df.isna().sum()
```

Out[17]:
```
title            0
studio           0
domestic_gross   0
year             0
dtype: int64
```

In [18]:
```
movies_df.isna().any().any()
```

Out[18]:
```
False
```

In [19]:
```
movies_df.drop_duplicates(subset='title', inplace=True)
movies_df.reset_index(drop=True, inplace=True)
```

In [20]:
```
movies_df.info()
movies_df.head()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3386 entries, 0 to 3385
Data columns (total 4 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   title           3386 non-null   object
 1   studio          3386 non-null   object
 2   domestic_gross  3386 non-null   float64
 3   year            3386 non-null   int64
dtypes: float64(1), int64(1), object(2)
memory usage: 105.9+ KB
```

Out[20]:

	title	studio	domestic_gross	year
0	Toy Story 3	BV	415000000.0	2010
1	Alice in Wonderland (2010)	BV	334200000.0	2010
2	Harry Potter and the Deathly Hallows Part 1	WB	296000000.0	2010
3	Inception	WB	292600000.0	2010

	title	studio	domestic_gross	year
4	Shrek Forever After	P/DW	238700000.0	2010

In [21]:
```
movies_df = movies_df.astype({'title': object, 'studio': object, 'domestic_gross': int, 'year': int
```

In [22]:
```
movies_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3386 entries, 0 to 3385
Data columns (total 4 columns):
 #   Column          Non-Null Count  Dtype 
---  ------          --------------  ----- 
 0   title           3386 non-null   object
 1   studio          3386 non-null   object
 2   domestic_gross  3386 non-null   int32 
 3   year            3386 non-null   int32 
dtypes: int32(2), object(2)
memory usage: 79.5+ KB
```

Cleaning 2nd Dataframe

In [23]:
```
title_df.isna().any()
```

Out[23]:
```
tconst             False
primary_title      False
original_title      True
start_year         False
runtime_minutes     True
genres              True
dtype: bool
```

In [24]:
```
title_df.isna().sum()
```

Out[24]:
```
tconst                0
primary_title         0
original_title       21
start_year            0
runtime_minutes    31739
genres              5408
dtype: int64
```

In [25]:
```
title_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 146144 entries, 0 to 146143
Data columns (total 6 columns):
 #   Column           Non-Null Count   Dtype  
---  ------           --------------   -----  
 0   tconst           146144 non-null  object 
 1   primary_title    146144 non-null  object 
 2   original_title   146123 non-null  object 
 3   start_year       146144 non-null  int64  
 4   runtime_minutes  114405 non-null  float64
 5   genres           140736 non-null  object 
dtypes: float64(1), int64(1), object(4)
memory usage: 6.7+ MB
```

In [26]:
```
title_df = title_df.drop('genres', axis = 1)
```

In [27]:
```
title_df['original_title'].fillna(value="N/A", inplace=True)
```

In [28]:
```
mean_runtime_minutes = title_df['runtime_minutes'].mean()
title_df['runtime_minutes'].fillna(mean_runtime_minutes, inplace=True)
title_df.isna().sum()
```

Out[28]:
```
tconst             0
primary_title      0
original_title     0
start_year         0
runtime_minutes    0
dtype: int64
```

In [29]:
```
title_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 146144 entries, 0 to 146143
Data columns (total 5 columns):
 #   Column           Non-Null Count   Dtype
---  ------           --------------   -----
 0   tconst           146144 non-null  object
 1   primary_title    146144 non-null  object
 2   original_title   146144 non-null  object
 3   start_year       146144 non-null  int64
 4   runtime_minutes  146144 non-null  float64
dtypes: float64(1), int64(1), object(3)
memory usage: 5.6+ MB
```

In [30]:
```
title_df = title_df.astype({'tconst': object, 'primary_title': object, 'original_title': object, 's
```

In [31]:
```
title_df.drop_duplicates(subset='tconst', inplace=True)
title_df.reset_index(drop=True, inplace=True)
```

In [32]:
```
title_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 146144 entries, 0 to 146143
Data columns (total 5 columns):
 #   Column           Non-Null Count   Dtype
---  ------           --------------   -----
 0   tconst           146144 non-null  object
 1   primary_title    146144 non-null  object
 2   original_title   146144 non-null  object
 3   start_year       146144 non-null  object
 4   runtime_minutes  146144 non-null  int32
dtypes: int32(1), object(4)
memory usage: 5.0+ MB
```

Cleaning 3rd Dataframe

In [33]:
```
ratings_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 73856 entries, 0 to 73855
Data columns (total 3 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   tconst         73856 non-null  object
 1   averagerating  73856 non-null  float64
 2   numvotes       73856 non-null  int64
dtypes: float64(1), int64(1), object(1)
memory usage: 1.7+ MB
```

In [34]:
```
ratings_df.isna().any()
```

Out[34]:
```
tconst           False
averagerating    False
numvotes         False
dtype: bool
```

In [35]:
```
ratings_df.isna().sum()
```

Out[35]:
```
tconst           0
averagerating    0
```

```
numvotes         0
dtype: int64
```

In [36]:
```
ratings_df.drop_duplicates(subset='tconst', inplace=True)
ratings_df.reset_index(drop=True, inplace=True)
```

In [37]:
```
ratings_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 73856 entries, 0 to 73855
Data columns (total 3 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   tconst         73856 non-null  object 
 1   averagerating  73856 non-null  float64
 2   numvotes       73856 non-null  int64  
dtypes: float64(1), int64(1), object(1)
memory usage: 1.7+ MB
```

Cleaning 4th Data Set

In [38]:
```
crew_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 146144 entries, 0 to 146143
Data columns (total 3 columns):
 #   Column     Non-Null Count   Dtype  
---  ------     --------------   -----  
 0   tconst     146144 non-null  object 
 1   directors  140417 non-null  object 
 2   writers    110261 non-null  object 
dtypes: object(3)
memory usage: 3.3+ MB
```

In [39]:
```
crew_df.isna().any()
```

Out[39]:
```
tconst       False
directors     True
writers       True
dtype: bool
```

In [40]:
```
crew_df.isna().sum()
```

Out[40]:
```
tconst           0
directors     5727
writers      35883
dtype: int64
```

In [41]:
```
crew_df['directors'].fillna(value="N/A", inplace=True)
crew_df['writers'].fillna(value="N/A", inplace=True)
```

In [42]:
```
crew_df.isna().sum()
```

Out[42]:
```
tconst       0
directors    0
writers      0
dtype: int64
```

In [43]:
```
crew_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 146144 entries, 0 to 146143
Data columns (total 3 columns):
 #   Column     Non-Null Count   Dtype  
---  ------     --------------   -----  
 0   tconst     146144 non-null  object 
 1   directors  146144 non-null  object 
 2   writers    146144 non-null  object 
```

```
dtypes: object(3)
memory usage: 3.3+ MB
```

In [44]:
```
crew_df.drop_duplicates(subset='tconst', inplace=True)
crew_df.reset_index(drop=True, inplace=True)
crew_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 146144 entries, 0 to 146143
Data columns (total 3 columns):
 #   Column     Non-Null Count   Dtype 
---  ------     --------------   ----- 
 0   tconst     146144 non-null  object
 1   directors  146144 non-null  object
 2   writers    146144 non-null  object
dtypes: object(3)
memory usage: 3.3+ MB
```

Merging DFs 1-4

In [45]:
```
movies_df.info()
title_df.info()
ratings_df.info()
crew_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3386 entries, 0 to 3385
Data columns (total 4 columns):
 #   Column          Non-Null Count  Dtype 
---  ------          --------------  ----- 
 0   title           3386 non-null   object
 1   studio          3386 non-null   object
 2   domestic_gross  3386 non-null   int32 
 3   year            3386 non-null   int32 
dtypes: int32(2), object(2)
memory usage: 79.5+ KB
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 146144 entries, 0 to 146143
Data columns (total 5 columns):
 #   Column           Non-Null Count   Dtype 
---  ------           --------------   ----- 
 0   tconst           146144 non-null  object
 1   primary_title    146144 non-null  object
 2   original_title   146144 non-null  object
 3   start_year       146144 non-null  object
 4   runtime_minutes  146144 non-null  int32 
dtypes: int32(1), object(4)
memory usage: 5.0+ MB
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 73856 entries, 0 to 73855
Data columns (total 3 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   tconst         73856 non-null  object 
 1   averagerating  73856 non-null  float64
 2   numvotes       73856 non-null  int64  
dtypes: float64(1), int64(1), object(1)
memory usage: 1.7+ MB
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 146144 entries, 0 to 146143
Data columns (total 3 columns):
 #   Column     Non-Null Count   Dtype 
---  ------     --------------   ----- 
 0   tconst     146144 non-null  object
 1   directors  146144 non-null  object
 2   writers    146144 non-null  object
dtypes: object(3)
memory usage: 3.3+ MB
```

In [46]:
```
# Merge 4 to 2, then 4+2 to 3, then to 1
```

In [47]:
```
df_4_2 = title_df.merge(crew_df, how = "inner", on = "tconst")
df_4_2.head()
```

Out[47]:

	tconst	primary_title	original_title	start_year	runtime_minutes	directors	
0	tt0063540	Sunghursh	Sunghursh	2013	175	nm0712540	nm0023551,nm1194313,nı
1	tt0066787	One Day Before the Rainy Season	Ashad Ka Ek Din	2019	114	nm0002411	
2	tt0069049	The Other Side of the Wind	The Other Side of the Wind	2018	122	nm0000080	nı
3	tt0069204	Sabse Bada Sukh	Sabse Bada Sukh	2018	86	nm0611531	
4	tt0100275	The Wandering Soap Opera	La Telenovela Errante	2017	80	nm0765384,nm0749914	nı

In [48]:
```
df_4_2.isna().any()
df_4_2.isna().sum()
```

Out[48]:
```
tconst             0
primary_title      0
original_title     0
start_year         0
runtime_minutes    0
directors          0
writers            0
dtype: int64
```

In [49]:
```
df_4_2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 146144 entries, 0 to 146143
Data columns (total 7 columns):
 #   Column           Non-Null Count   Dtype 
---  ------           --------------   ----- 
 0   tconst           146144 non-null  object
 1   primary_title    146144 non-null  object
 2   original_title   146144 non-null  object
 3   start_year       146144 non-null  object
 4   runtime_minutes  146144 non-null  int32 
 5   directors        146144 non-null  object
 6   writers          146144 non-null  object
dtypes: int32(1), object(6)
memory usage: 8.4+ MB
```

In [50]:
```
df_432 = ratings_df.merge(df_4_2, how = "inner", on = "tconst")
```

In [51]:
```
df_432.info()
df_432.head(15)
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 73856 entries, 0 to 73855
Data columns (total 9 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   tconst           73856 non-null  object 
 1   averagerating    73856 non-null  float64
 2   numvotes         73856 non-null  int64  
 3   primary_title    73856 non-null  object 
 4   original_title   73856 non-null  object 
 5   start_year       73856 non-null  object 
 6   runtime_minutes  73856 non-null  int32  
```

```
 7   directors        73856 non-null  object
 8   writers          73856 non-null  object
dtypes: float64(1), int32(1), int64(1), object(6)
memory usage: 5.4+ MB
```

Out[51]:

	tconst	averagerating	numvotes	primary_title	original_title	start_year	runtime_minutes	direct
0	tt10356526	8.3	31	Laiye Je Yaarian	Laiye Je Yaarian	2019	117	nm8353
1	tt10384606	8.9	559	Borderless	Borderless	2019	87	nm9932562,nm9250
2	tt1042974	6.4	20	Just Inès	Just Inès	2010	90	nm1915
3	tt1043726	4.2	50352	The Legend of Hercules	The Legend of Hercules	2014	99	nm0001
4	tt1060240	6.5	21	Até Onde?	Até Onde?	2011	73	nm1926
5	tt1069246	6.2	326	Habana Eva	Habana Eva	2010	106	nm0868
6	tt1094666	7.0	1613	The Hammer	Hamill	2010	108	nm1464
7	tt1130982	6.4	571	The Night Clerk	Avant l'aube	2011	104	nm1292
8	tt1156528	7.2	265	Silent Sonata	Circus Fantasticus	2011	77	nm0121
9	tt1161457	4.2	148	Vanquisher	The Vanquisher	2016	90	nm2874
10	tt1171222	5.1	8296	Baggage Claim	Baggage Claim	2013	96	nm0847
11	tt1174693	5.8	2381	The Four-Faced Liar	The Four-Faced Liar	2010	87	nm0153
12	tt1181840	7.0	5494	Jack and the Cuckoo-Clock Heart	Jack et la mécanique du coeur	2013	94	nm0540962,nm1071
13	tt1193623	8.0	5	Buried Prayers	Buried Prayers	2010	86	nm1746
14	tt1199588	5.5	74	Black Widow	Black Widow	2010	75	nm0736

In [52]:
```
df_432_edited = df_432.rename(columns = {'primary_title': 'title'})
```

In [53]:
```
df_432_edited.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 73856 entries, 0 to 73855
Data columns (total 9 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   tconst           73856 non-null  object
 1   averagerating    73856 non-null  float64
 2   numvotes         73856 non-null  int64
 3   title            73856 non-null  object
 4   original_title   73856 non-null  object
 5   start_year       73856 non-null  object
 6   runtime_minutes  73856 non-null  int32
 7   directors        73856 non-null  object
 8   writers          73856 non-null  object
dtypes: float64(1), int32(1), int64(1), object(6)
memory usage: 5.4+ MB
```

In [54]:
```
df_432_edited.head()
```

Out[54]:

	tconst	averagerating	numvotes	title	original_title	start_year	runtime_minutes	directors
0	tt10356526	8.3	31	Laiye Je Yaarian	Laiye Je Yaarian	2019	117	nm8353804
1	tt10384606	8.9	559	Borderless	Borderless	2019	87	nm9932562,nm9250842
2	tt1042974	6.4	20	Just Inès	Just Inès	2010	90	nm1915232
3	tt1043726	4.2	50352	The Legend of Hercules	The Legend of Hercules	2014	99	nm0001317
4	tt1060240	6.5	21	Até Onde?	Até Onde?	2011	73	nm1926349

In [55]:
```
movies_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3386 entries, 0 to 3385
Data columns (total 4 columns):
 #   Column          Non-Null Count  Dtype 
---  ------          --------------  ----- 
 0   title           3386 non-null   object
 1   studio          3386 non-null   object
 2   domestic_gross  3386 non-null   int32 
 3   year            3386 non-null   int32 
dtypes: int32(2), object(2)
memory usage: 79.5+ KB
```

Creating Final Dataframe

In [56]:
```
df_1234 = movies_df.merge(df_432_edited, how = "inner", on = "title")
```

In [57]:
```
df_1234.head()
```

Out[57]:

	title	studio	domestic_gross	year	tconst	averagerating	numvotes	original_title	start_year	runtime_mi
0	Toy Story 3	BV	415000000	2010	tt0435761	8.3	682218	Toy Story 3	2010	
1	Inception	WB	292600000	2010	tt1375666	8.8	1841066	Inception	2010	
2	Shrek Forever After	P/DW	238700000	2010	tt0892791	6.3	167532	Shrek Forever After	2010	
3	The Twilight Saga: Eclipse	Sum.	300500000	2010	tt1325004	5.0	211733	The Twilight Saga: Eclipse	2010	
4	Iron Man 2	Par.	312400000	2010	tt1228705	7.0	657690	Iron Man 2	2010	

In [58]:
```
df_1234.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 3025 entries, 0 to 3024
Data columns (total 12 columns):
 #   Column          Non-Null Count  Dtype 
---  ------          --------------  ----- 
 0   title           3025 non-null   object
 1   studio          3025 non-null   object
 2   domestic_gross  3025 non-null   int32 
```

```
 3   year             3025 non-null   int32
 4   tconst           3025 non-null   object
 5   averagerating    3025 non-null   float64
 6   numvotes         3025 non-null   int64
 7   original_title   3025 non-null   object
 8   start_year       3025 non-null   object
 9   runtime_minutes  3025 non-null   int32
 10  directors        3025 non-null   object
 11  writers          3025 non-null   object
dtypes: float64(1), int32(3), int64(1), object(7)
memory usage: 271.8+ KB
```

In [59]:
```
cleaned_df = df_1234
```

In [60]:
```
cleaned_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 3025 entries, 0 to 3024
Data columns (total 12 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   title            3025 non-null   object
 1   studio           3025 non-null   object
 2   domestic_gross   3025 non-null   int32
 3   year             3025 non-null   int32
 4   tconst           3025 non-null   object
 5   averagerating    3025 non-null   float64
 6   numvotes         3025 non-null   int64
 7   original_title   3025 non-null   object
 8   start_year       3025 non-null   object
 9   runtime_minutes  3025 non-null   int32
 10  directors        3025 non-null   object
 11  writers          3025 non-null   object
dtypes: float64(1), int32(3), int64(1), object(7)
memory usage: 271.8+ KB
```

In [61]:
```
cleaned_df = cleaned_df.set_index('tconst')
```

In [62]:
```
cleaned_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 3025 entries, tt0435761 to tt5718046
Data columns (total 11 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   title            3025 non-null   object
 1   studio           3025 non-null   object
 2   domestic_gross   3025 non-null   int32
 3   year             3025 non-null   int32
 4   averagerating    3025 non-null   float64
 5   numvotes         3025 non-null   int64
 6   original_title   3025 non-null   object
 7   start_year       3025 non-null   object
 8   runtime_minutes  3025 non-null   int32
 9   directors        3025 non-null   object
 10  writers          3025 non-null   object
dtypes: float64(1), int32(3), int64(1), object(6)
memory usage: 248.1+ KB
```

In [63]:
```
cleaned_df.isna().any()
```

Out[63]:
```
title             False
studio            False
domestic_gross    False
year              False
averagerating     False
numvotes          False
original_title    False
start_year        False
runtime_minutes   False
directors         False
```

```
writers            False
dtype: bool
```

In [64]:
```
cleaned_df.isna().sum()
```

Out[64]:
```
title              0
studio             0
domestic_gross     0
year               0
averagerating      0
numvotes           0
original_title     0
start_year         0
runtime_minutes    0
directors          0
writers            0
dtype: int64
```

In [65]:
```
cleaned_df.head(20)
```

Out[65]:

tconst	title	studio	domestic_gross	year	averagerating	numvotes	original_title	start_year	runtime_min
tt0435761	Toy Story 3	BV	415000000	2010	8.3	682218	Toy Story 3	2010	
tt1375666	Inception	WB	292600000	2010	8.8	1841066	Inception	2010	
tt0892791	Shrek Forever After	P/DW	238700000	2010	6.3	167532	Shrek Forever After	2010	
tt1325004	The Twilight Saga: Eclipse	Sum.	300500000	2010	5.0	211733	The Twilight Saga: Eclipse	2010	
tt1228705	Iron Man 2	Par.	312400000	2010	7.0	657690	Iron Man 2	2010	
tt0398286	Tangled	BV	200800000	2010	7.8	366366	Tangled	2010	
tt1323594	Despicable Me	Uni.	251500000	2010	7.7	464511	Despicable Me	2010	
tt0892769	How to Train Your Dragon	P/DW	217600000	2010	8.1	611299	How to Train Your Dragon	2010	
tt0980970	The Chronicles of Narnia: The Voyage of the Da...	Fox	104400000	2010	6.3	129663	The Chronicles of Narnia: The Voyage of the Da...	2010	
tt1504320	The King's Speech	Wein.	135500000	2010	8.0	593629	The King's Speech	2010	
tt1155076	The Karate Kid	Sony	176600000	2010	6.2	146401	The Karate Kid	2010	
tt0473075	Prince of Persia: The Sands of Time	BV	90800000	2010	6.6	254975	Prince of Persia: The Sands of Time	2010	
tt0947798	Black Swan	FoxS	107000000	2010	8.0	648854	Black Swan	2010	
tt1001526	Megamind	P/DW	148400000	2010	7.3	207488	Megamind	2010	

tconst	title	studio	domestic_gross	year	averagerating	numvotes	original_title	start_year	runtime_min
tt6858500	Robin Hood	Uni.	105300000	2010	7.6	5	Robin Hood	2018	
tt2363363	Robin Hood	Uni.	105300000	2010	6.3	78	Robin Hood	2013	
tt4532826	Robin Hood	Uni.	105300000	2010	5.3	41588	Robin Hood	2018	
tt0955308	Robin Hood	Uni.	105300000	2010	6.6	239480	Robin Hood	2010	
tt0938283	The Last Airbender	Par.	131800000	2010	4.1	137734	The Last Airbender	2010	
tt0970866	Little Fockers	Uni.	148400000	2010	5.5	99222	Little Fockers	2010	

Cleaning of 5th Data Set for Additional Info

In [66]:
```
budgets_df = pd.read_csv('C://Users//rychu//Documents//Flatiron//dsc-phase-1-project//zippedData//t
```

In [67]:
```
budgets_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5782 entries, 0 to 5781
Data columns (total 6 columns):
 #   Column             Non-Null Count  Dtype 
---  ------             --------------  ----- 
 0   id                 5782 non-null   int64 
 1   release_date       5782 non-null   object
 2   movie              5782 non-null   object
 3   production_budget  5782 non-null   object
 4   domestic_gross     5782 non-null   object
 5   worldwide_gross    5782 non-null   object
dtypes: int64(1), object(5)
memory usage: 271.2+ KB
```

In [68]:
```
budgets_df.head()
```

Out[68]:

	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 [69]:
```
budgets_df.isna().any()
```

Out[69]:
```
id                 False
release_date       False
movie              False
production_budget  False
domestic_gross     False
worldwide_gross    False
dtype: bool
```

In [70]:
```
budgets_df = budgets_df.drop('id', axis = 1)
```

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

Out[71]:

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

In [72]:
```
budgets_df = budgets_df.rename(columns = {'movie': 'title'})
```

In [73]:
```
budgets_df.head()
```

Out[73]:

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

In [74]:
```
budgets_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5782 entries, 0 to 5781
Data columns (total 5 columns):
 #   Column             Non-Null Count  Dtype 
---  ------             --------------  ----- 
 0   release_date       5782 non-null   object
 1   title              5782 non-null   object
 2   production_budget  5782 non-null   object
 3   domestic_gross     5782 non-null   object
 4   worldwide_gross    5782 non-null   object
dtypes: object(5)
memory usage: 226.0+ KB
```

Merging 5th to Final Cleaned DF

In [75]:
```
finalcleaned_df = cleaned_df.merge(budgets_df, how = "inner", on = "title")
```

In [76]:
```
finalcleaned_df.head(10)
```

Out[76]:

	title	studio	domestic_gross_x	year	averagerating	numvotes	original_title	start_year	runtime_minutes
0	Toy Story 3	BV	415000000	2010	8.3	682218	Toy Story 3	2010	103
1	Inception	WB	292600000	2010	8.8	1841066	Inception	2010	148
2	Shrek Forever After	P/DW	238700000	2010	6.3	167532	Shrek Forever After	2010	93

	title	studio	domestic_gross_x	year	averagerating	numvotes	original_title	start_year	runtime_minutes	
3	The Twilight Saga: Eclipse	Sum.	300500000	2010	5.0	211733	The Twilight Saga: Eclipse	2010	124	
4	Iron Man 2	Par.	312400000	2010	7.0	657690	Iron Man 2	2010	124	
5	Tangled	BV	200800000	2010	7.8	366366	Tangled	2010	100	r
6	Despicable Me	Uni.	251500000	2010	7.7	464511	Despicable Me	2010	95	r
7	How to Train Your Dragon	P/DW	217600000	2010	8.1	611299	How to Train Your Dragon	2010	98	r
8	The Chronicles of Narnia: The Voyage of the Da...	Fox	104400000	2010	6.3	129663	The Chronicles of Narnia: The Voyage of the Da...	2010	113	
9	The Karate Kid	Sony	176600000	2010	6.2	146401	The Karate Kid	2010	140	

In [77]:
```
finalcleaned_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1413 entries, 0 to 1412
Data columns (total 15 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   title              1413 non-null   object 
 1   studio             1413 non-null   object 
 2   domestic_gross_x   1413 non-null   int32  
 3   year               1413 non-null   int32  
 4   averagerating      1413 non-null   float64
 5   numvotes           1413 non-null   int64  
 6   original_title     1413 non-null   object 
 7   start_year         1413 non-null   object 
 8   runtime_minutes    1413 non-null   int32  
 9   directors          1413 non-null   object 
 10  writers            1413 non-null   object 
 11  release_date       1413 non-null   object 
 12  production_budget  1413 non-null   object 
 13  domestic_gross_y   1413 non-null   object 
 14  worldwide_gross    1413 non-null   object 
dtypes: float64(1), int32(3), int64(1), object(10)
memory usage: 160.1+ KB
```

In [78]:
```
finalcleaned_df = finalcleaned_df.drop('domestic_gross_x', axis = 1)
```

In [79]:
```
finalcleaned_df = finalcleaned_df.drop('year', axis = 1)
```

In [80]:
```
finalcleaned_df = finalcleaned_df.drop('start_year', axis = 1)
```

In [81]:
```
finalcleaned_df = finalcleaned_df.drop('original_title', axis = 1)
```

In [82]:
```
finalcleaned_df = finalcleaned_df.rename(columns = {'domestic_gross_y': 'domestic_gross'})
```

In [83]:
```
finalcleaned_df.head(10)
```

Out[83]:

	title	studio	averagerating	numvotes	runtime_minutes	directors

	title	studio	averagerating	numvotes	runtime_minutes	directors	
0	Toy Story 3	BV	8.3	682218	103	nm0881279	nm0005124,nm000
1	Inception	WB	8.8	1841066	148	nm0634240	
2	Shrek Forever After	P/DW	6.3	167532	93	nm0593610	nm0825308,nm0458441,nm0
3	The Twilight Saga: Eclipse	Sum.	5.0	211733	124	nm1720541	
4	Iron Man 2	Par.	7.0	657690	124	nm0269463	nm0857620,nm0498278,nm141
5	Tangled	BV	7.8	366366	100	nm1977355,nm0397174	nm155
6	Despicable Me	Uni.	7.7	464511	95	nm1853544,nm0719208	nm066
7	How to Train Your Dragon	P/DW	8.1	611299	98	nm0761498,nm0213450	nm0204030,nm0213450,nm0
8	The Chronicles of Narnia: The Voyage of the Da...	Fox	6.3	129663	113	nm0000776	nm1321655,nm132
9	The Karate Kid	Sony	6.2	146401	140	nm0958969	

In [84]:
```
finalcleaned_df.isna().any()
```

Out[84]:
```
title               False
studio              False
averagerating       False
numvotes            False
runtime_minutes     False
directors           False
writers             False
release_date        False
production_budget   False
domestic_gross      False
worldwide_gross     False
dtype: bool
```

In [85]:
```
finalcleaned_df.isna().sum()
```

Out[85]:
```
title               0
studio              0
averagerating       0
numvotes            0
runtime_minutes     0
directors           0
writers             0
release_date        0
production_budget   0
domestic_gross      0
worldwide_gross     0
dtype: int64
```

In [86]:
```
finalcleaned_df.head(20)
```

Out[86]:

	title	studio	averagerating	numvotes	runtime_minutes	directors	
0	Toy Story 3	BV	8.3	682218	103	nm0881279	nm0005124,nm00
1	Inception	WB	8.8	1841066	148	nm0634240	
2	Shrek Forever After	P/DW	6.3	167532	93	nm0593610	nm0825308,nm0458441,nm
3	The Twilight Saga: Eclipse	Sum.	5.0	211733	124	nm1720541	
4	Iron Man 2	Par.	7.0	657690	124	nm0269463	nm0857620,nm0498278,nm14
5	Tangled	BV	7.8	366366	100	nm1977355,nm0397174	nm15
6	Despicable Me	Uni.	7.7	464511	95	nm1853544,nm0719208	nm06
7	How to Train Your Dragon	P/DW	8.1	611299	98	nm0761498,nm0213450	nm0204030,nm0213450,nm
8	The Chronicles of Narnia: The Voyage of the Da...	Fox	6.3	129663	113	nm0000776	nm1321655,nm13
9	The Karate Kid	Sony	6.2	146401	140	nm0958969	
10	The Karate Kid	Sony	6.2	146401	140	nm0958969	
11	Black Swan	FoxS	8.0	648854	108	nm0004716	nm21
12	Megamind	P/DW	7.3	207488	95	nm0569891	
13	Robin Hood	Uni.	7.6	5	86	nm7789892	
14	Robin Hood	Uni.	7.6	5	86	nm7789892	
15	Robin Hood	Uni.	6.3	78	92	nm2347483	
16	Robin Hood	Uni.	6.3	78	92	nm2347483	
17	Robin Hood	Uni.	5.3	41588	116	nm1163264	
18	Robin Hood	Uni.	5.3	41588	116	nm1163264	
19	Robin Hood	Uni.	6.6	239480	140	nm0000631	nm00

In [87]:
```
finalcleaned_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1413 entries, 0 to 1412
```

```
Data columns (total 11 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   title              1413 non-null   object
 1   studio             1413 non-null   object
 2   averagerating      1413 non-null   float64
 3   numvotes           1413 non-null   int64
 4   runtime_minutes    1413 non-null   int32
 5   directors          1413 non-null   object
 6   writers            1413 non-null   object
 7   release_date       1413 non-null   object
 8   production_budget  1413 non-null   object
 9   domestic_gross     1413 non-null   object
 10  worldwide_gross    1413 non-null   object
dtypes: float64(1), int32(1), int64(1), object(8)
memory usage: 126.9+ KB
```

In [88]:
```
finalcleaned_df['production_budget'] = finalcleaned_df['production_budget'].map(lambda x: x.lstrip(
finalcleaned_df['domestic_gross'] = finalcleaned_df['domestic_gross'].map(lambda x: x.lstrip('$').r
finalcleaned_df['worldwide_gross'] = finalcleaned_df['worldwide_gross'].map(lambda x: x.lstrip('$')
```

In [89]:
```
finalcleaned_df['production_budget'].replace(',', '', regex=True, inplace=True)
finalcleaned_df['domestic_gross'].replace(',', '', regex=True, inplace=True)
finalcleaned_df['worldwide_gross'].replace(',', '', regex=True, inplace=True)
```

In [90]:
```
finalcleaned_df.head()
```

Out[90]:

	title	studio	averagerating	numvotes	runtime_minutes	directors	
0	Toy Story 3	BV	8.3	682218	103	nm0881279	nm0005124,nm0004056,nm0881
1	Inception	WB	8.8	1841066	148	nm0634240	
2	Shrek Forever After	P/DW	6.3	167532	93	nm0593610	nm0825308,nm0458441,nm0501359,nm00
3	The Twilight Saga: Eclipse	Sum.	5.0	211733	124	nm1720541	nm0742
4	Iron Man 2	Par.	7.0	657690	124	nm0269463	nm0857620,nm0498278,nm1411347,nm1293

In [91]:
```
finalcleaned_df = finalcleaned_df.astype({"title": 'object', "studio": 'object', "averagerating": '
```

In [92]:
```
finalcleaned_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1413 entries, 0 to 1412
Data columns (total 11 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   title              1413 non-null   object
 1   studio             1413 non-null   object
 2   averagerating      1413 non-null   float64
 3   numvotes           1413 non-null   int64
 4   runtime_minutes    1413 non-null   int64
 5   directors          1413 non-null   object
 6   writers            1413 non-null   object
 7   release_date       1413 non-null   string
 8   production_budget  1413 non-null   int64
 9   domestic_gross     1413 non-null   int64
 10  worldwide_gross    1413 non-null   int64
dtypes: float64(1), int64(5), object(4), string(1)
memory usage: 132.5+ KB
```

In [93]:
```
finalcleaned_df.head(15)
```

Out[93]:

	title	studio	averagerating	numvotes	runtime_minutes	directors	
0	Toy Story 3	BV	8.3	682218	103	nm0881279	nm0005124,nm00
1	Inception	WB	8.8	1841066	148	nm0634240	
2	Shrek Forever After	P/DW	6.3	167532	93	nm0593610	nm0825308,nm0458441,nm
3	The Twilight Saga: Eclipse	Sum.	5.0	211733	124	nm1720541	
4	Iron Man 2	Par.	7.0	657690	124	nm0269463	nm0857620,nm0498278,nm14
5	Tangled	BV	7.8	366366	100	nm1977355,nm0397174	nm15
6	Despicable Me	Uni.	7.7	464511	95	nm1853544,nm0719208	nm06
7	How to Train Your Dragon	P/DW	8.1	611299	98	nm0761498,nm0213450	nm0204030,nm0213450,nm
8	The Chronicles of Narnia: The Voyage of the Da...	Fox	6.3	129663	113	nm0000776	nm1321655,nm13
9	The Karate Kid	Sony	6.2	146401	140	nm0958969	
10	The Karate Kid	Sony	6.2	146401	140	nm0958969	
11	Black Swan	FoxS	8.0	648854	108	nm0004716	nm21
12	Megamind	P/DW	7.3	207488	95	nm0569891	
13	Robin Hood	Uni.	7.6	5	86	nm7789892	
14	Robin Hood	Uni.	7.6	5	86	nm7789892	

Data Analysis and Visualizations

In [94]:
```
# Question 1: What does the relationship look like between movie budgets and their income? (For tho
# Question 2: Does the alloted budget have a significant impact on rating score? (For those that ha
# Question 3: What seasons averaged the highest gross? (For those that had a profit > 0)
```

Let's Analyze the data and further develop the information with are looking for in the DF

In [95]:
```
finalcleaned_df['domestic_difference'] = finalcleaned_df['domestic_gross'] - finalcleaned_df['produ
finalcleaned_df['worldwide_difference'] = finalcleaned_df['worldwide_gross'] - finalcleaned_df['pro
```

In [96]:
```
finalcleaned_df.head(10)
```

Out[96]:

	title	studio	averagerating	numvotes	runtime_minutes	directors	
0	Toy Story 3	BV	8.3	682218	103	nm0881279	nm0005124,nm000
1	Inception	WB	8.8	1841066	148	nm0634240	
2	Shrek Forever After	P/DW	6.3	167532	93	nm0593610	nm0825308,nm0458441,nm0
3	The Twilight Saga: Eclipse	Sum.	5.0	211733	124	nm1720541	
4	Iron Man 2	Par.	7.0	657690	124	nm0269463	nm0857620,nm0498278,nm141
5	Tangled	BV	7.8	366366	100	nm1977355,nm0397174	nm155
6	Despicable Me	Uni.	7.7	464511	95	nm1853544,nm0719208	nm066
7	How to Train Your Dragon	P/DW	8.1	611299	98	nm0761498,nm0213450	nm0204030,nm0213450,nm0
8	The Chronicles of Narnia: The Voyage of the Da...	Fox	6.3	129663	113	nm0000776	nm1321655,nm132
9	The Karate Kid	Sony	6.2	146401	140	nm0958969	

In [97]:
```
finalcleaned_df['release_date'] = pd.to_datetime(finalcleaned_df['release_date'], format='%b %d, %Y
```

In [98]:
```
finalcleaned_df.head(20)
```

Out[98]:

	title	studio	averagerating	numvotes	runtime_minutes	directors	
0	Toy Story 3	BV	8.3	682218	103	nm0881279	nm0005124,nm00
1	Inception	WB	8.8	1841066	148	nm0634240	
2	Shrek Forever After	P/DW	6.3	167532	93	nm0593610	nm0825308,nm0458441,nm
3	The Twilight Saga: Eclipse	Sum.	5.0	211733	124	nm1720541	
4	Iron Man 2	Par.	7.0	657690	124	nm0269463	nm0857620,nm0498278,nm14
5	Tangled	BV	7.8	366366	100	nm1977355,nm0397174	nm15
6	Despicable Me	Uni.	7.7	464511	95	nm1853544,nm0719208	nm06
7	How to Train Your Dragon	P/DW	8.1	611299	98	nm0761498,nm0213450	nm0204030,nm0213450,nm

	title	studio	averagerating	numvotes	runtime_minutes	directors	
8	The Chronicles of Narnia: The Voyage of the Da...	Fox	6.3	129663	113	nm0000776	nm1321655,nm13
9	The Karate Kid	Sony	6.2	146401	140	nm0958969	
10	The Karate Kid	Sony	6.2	146401	140	nm0958969	
11	Black Swan	FoxS	8.0	648854	108	nm0004716	nm21
12	Megamind	P/DW	7.3	207488	95	nm0569891	
13	Robin Hood	Uni.	7.6	5	86	nm7789892	
14	Robin Hood	Uni.	7.6	5	86	nm7789892	
15	Robin Hood	Uni.	6.3	78	92	nm2347483	
16	Robin Hood	Uni.	6.3	78	92	nm2347483	
17	Robin Hood	Uni.	5.3	41588	116	nm1163264	
18	Robin Hood	Uni.	5.3	41588	116	nm1163264	
19	Robin Hood	Uni.	6.6	239480	140	nm0000631	nm00

In [99]:
```
finalcleaned_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1413 entries, 0 to 1412
Data columns (total 13 columns):
 #   Column                Non-Null Count  Dtype         
---  ------                --------------  -----         
 0   title                 1413 non-null   object        
 1   studio                1413 non-null   object        
 2   averagerating         1413 non-null   float64       
 3   numvotes              1413 non-null   int64         
 4   runtime_minutes       1413 non-null   int64         
 5   directors             1413 non-null   object        
 6   writers               1413 non-null   object        
 7   release_date          1413 non-null   datetime64[ns]
 8   production_budget     1413 non-null   int64         
 9   domestic_gross        1413 non-null   int64         
 10  worldwide_gross       1413 non-null   int64         
 11  domestic_difference   1413 non-null   int64         
 12  worldwide_difference  1413 non-null   int64         
dtypes: datetime64[ns](1), float64(1), int64(7), object(4)
memory usage: 154.5+ KB
```

Ok With All Data Needed Properly Formatted + Organized... Time to Graph

In [100…
```
# Initial Graphs
```

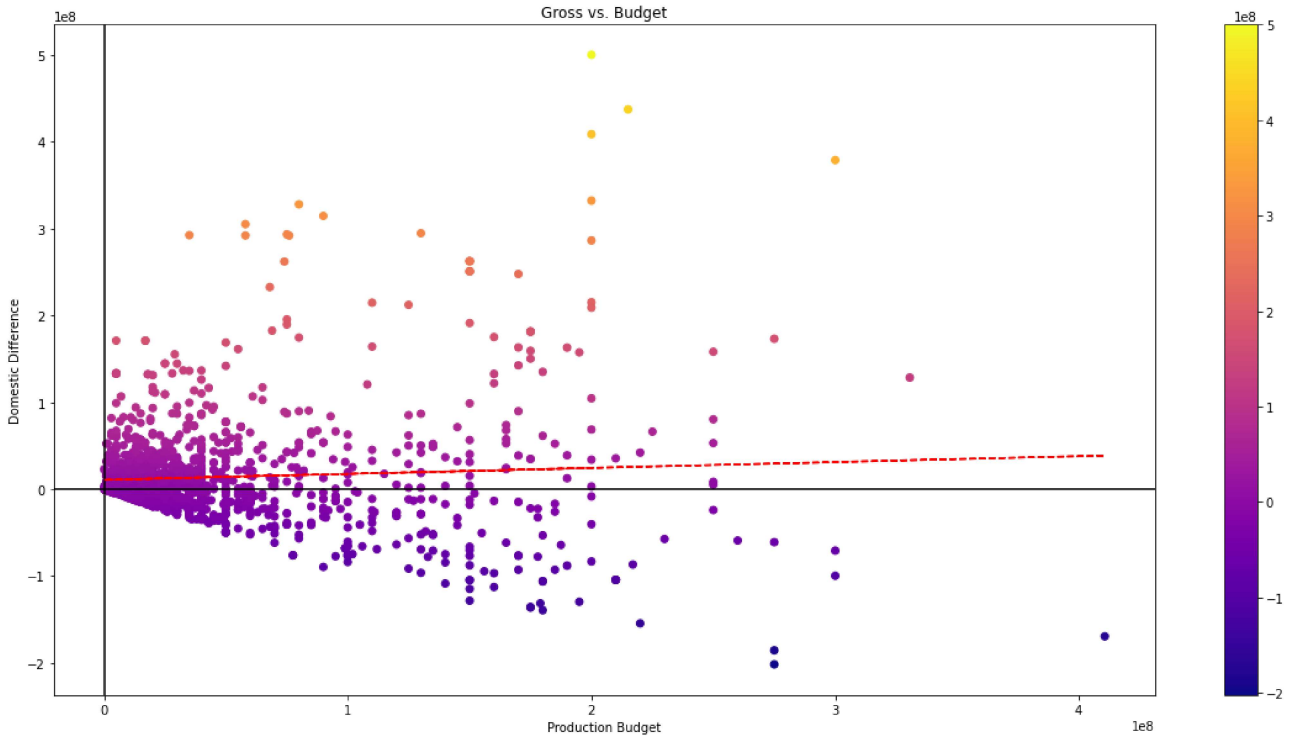
In [101…
```
finalcleaned_df['domestic_difference'].describe()
```

Out[101…
```
count    1.413000e+03
mean     1.383999e+07
std      6.060979e+07
min     -2.019413e+08
25%     -1.134288e+07
50%     -5.445900e+04
75%      2.681666e+07
max      5.000596e+08
Name: domestic_difference, dtype: float64
```

In [102…
```
finalcleaned_df['worldwide_difference'].describe()
```

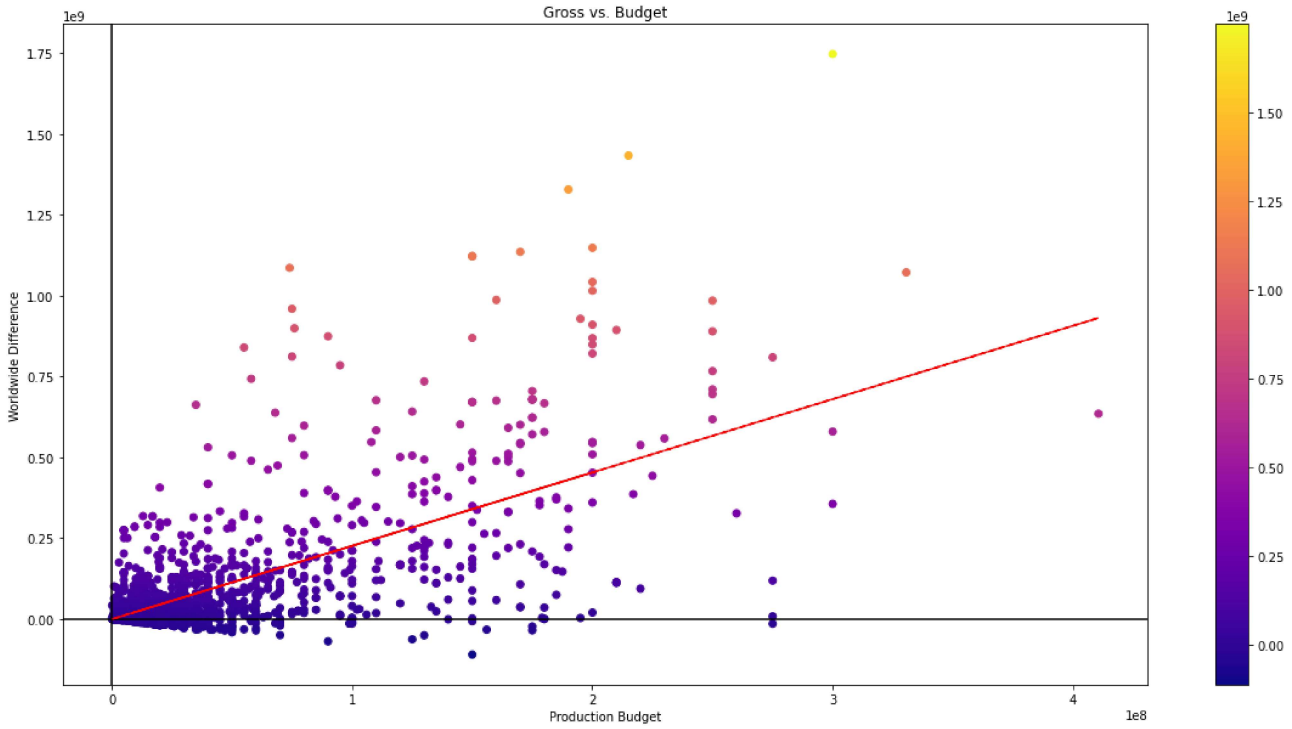
Out[102…
```
count    1.413000e+03
mean     1.040634e+08
std      1.931573e+08
min     -1.104502e+08
25%      1.525973e+06
50%      3.115860e+07
75%      1.191555e+08
max      1.748134e+09
Name: worldwide_difference, dtype: float64
```

In [103…
```
x = finalcleaned_df['production_budget']
y = finalcleaned_df['domestic_difference']

plt.figure(figsize = (20, 10))
plt.scatter(x, y, c=y, cmap = 'plasma')
plt.axvline(0, c = 'black')
plt.axhline(0, c = 'black')

plt.xlabel("Production Budget")
plt.ylabel("Domestic Difference")
plt.title("Gross vs. Budget")

z = np.polyfit(x, y, 1)
p = np.poly1d(z)
plt.plot(x,p(x),"r--")

plt.colorbar()
plt.show()
```



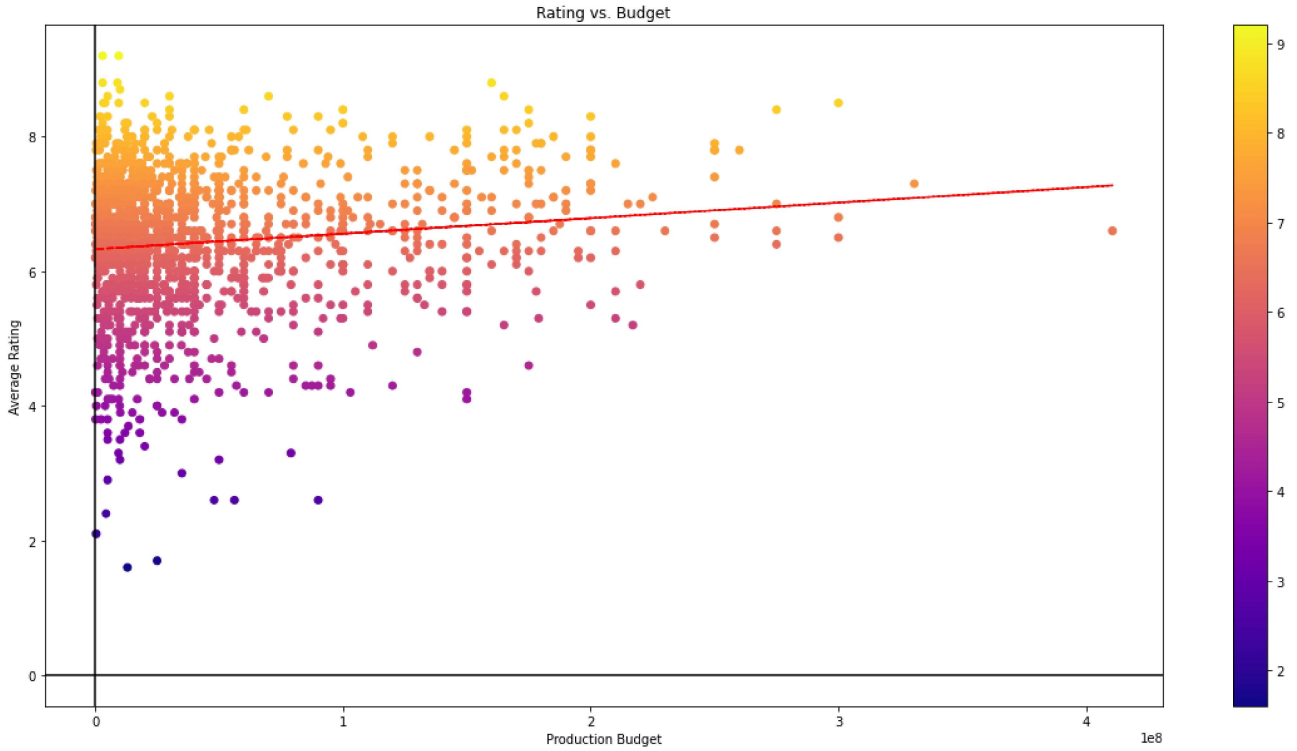
In [104…

```
x = finalcleaned_df['production_budget']
y = finalcleaned_df['worldwide_difference']

plt.figure(figsize = (20, 10))
plt.scatter(x, y, c=y, cmap = 'plasma')
plt.axvline(0, c = 'black')
plt.axhline(0, c = 'black')

plt.xlabel("Production Budget")
plt.ylabel("Worldwide Difference")
plt.title("Gross vs. Budget")

z = np.polyfit(x, y, 1)
p = np.poly1d(z)
plt.plot(x,p(x),"r--")

plt.colorbar()
plt.show()
```



In [105…

```
x = finalcleaned_df['production_budget']
y = finalcleaned_df['averagerating']

plt.figure(figsize = (20, 10))
plt.scatter(x, y, c=y, cmap = 'plasma')
plt.axvline(0, c = 'black')
plt.axhline(0, c = 'black')

plt.xlabel("Production Budget")
plt.ylabel("Average Rating")
plt.title("Rating vs. Budget")

z = np.polyfit(x, y, 1)
p = np.poly1d(z)
plt.plot(x,p(x),"r--")

plt.colorbar()
plt.show()
```



In [106…
```
finalcleaned_df.head(10)
```

Out[106…

	title	studio	averagerating	numvotes	runtime_minutes	directors	
0	Toy Story 3	BV	8.3	682218	103	nm0881279	nm0005124,nm000
1	Inception	WB	8.8	1841066	148	nm0634240	
2	Shrek Forever After	P/DW	6.3	167532	93	nm0593610	nm0825308,nm0458441,nm0
3	The Twilight Saga: Eclipse	Sum.	5.0	211733	124	nm1720541	
4	Iron Man 2	Par.	7.0	657690	124	nm0269463	nm0857620,nm0498278,nm141
5	Tangled	BV	7.8	366366	100	nm1977355,nm0397174	nm155
6	Despicable Me	Uni.	7.7	464511	95	nm1853544,nm0719208	nm066
7	How to Train Your Dragon	P/DW	8.1	611299	98	nm0761498,nm0213450	nm0204030,nm0213450,nm0
8	The Chronicles of Narnia: The Voyage of the Da...	Fox	6.3	129663	113	nm0000776	nm1321655,nm132
9	The Karate Kid	Sony	6.2	146401	140	nm0958969	

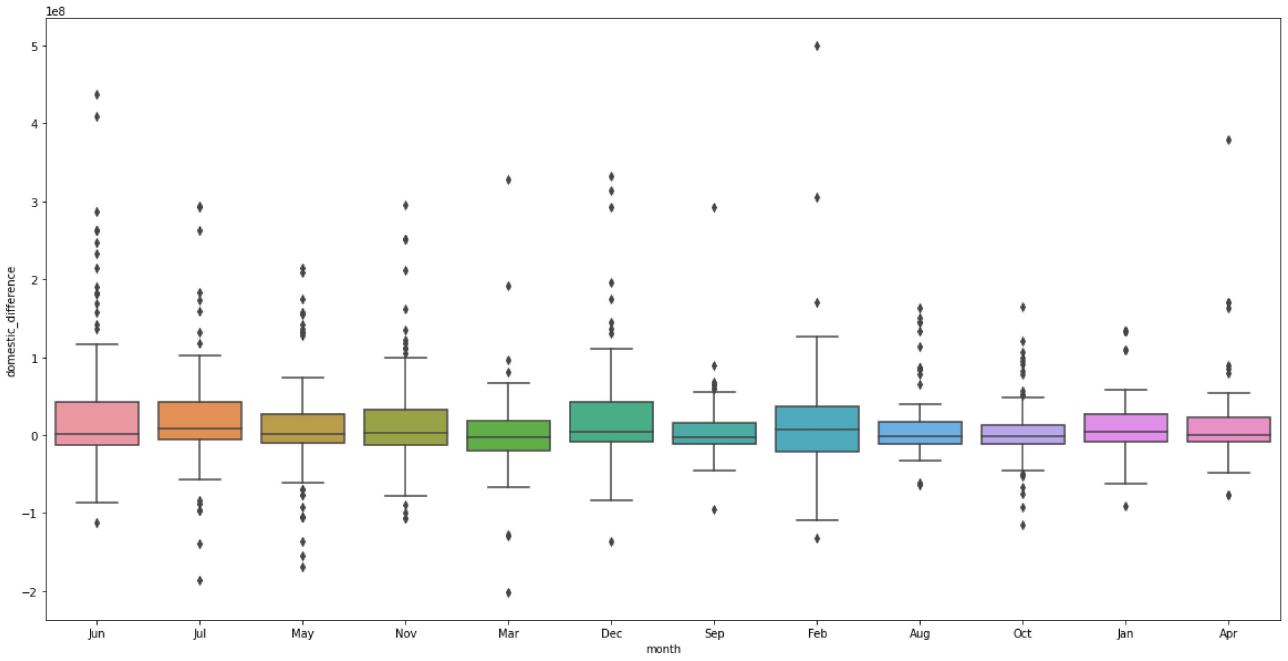
In [107…
```
finalcleaned_df['month'] = finalcleaned_df['release_date'].dt.strftime('%b')
```

In [108…
```
finalcleaned_df.head()
```

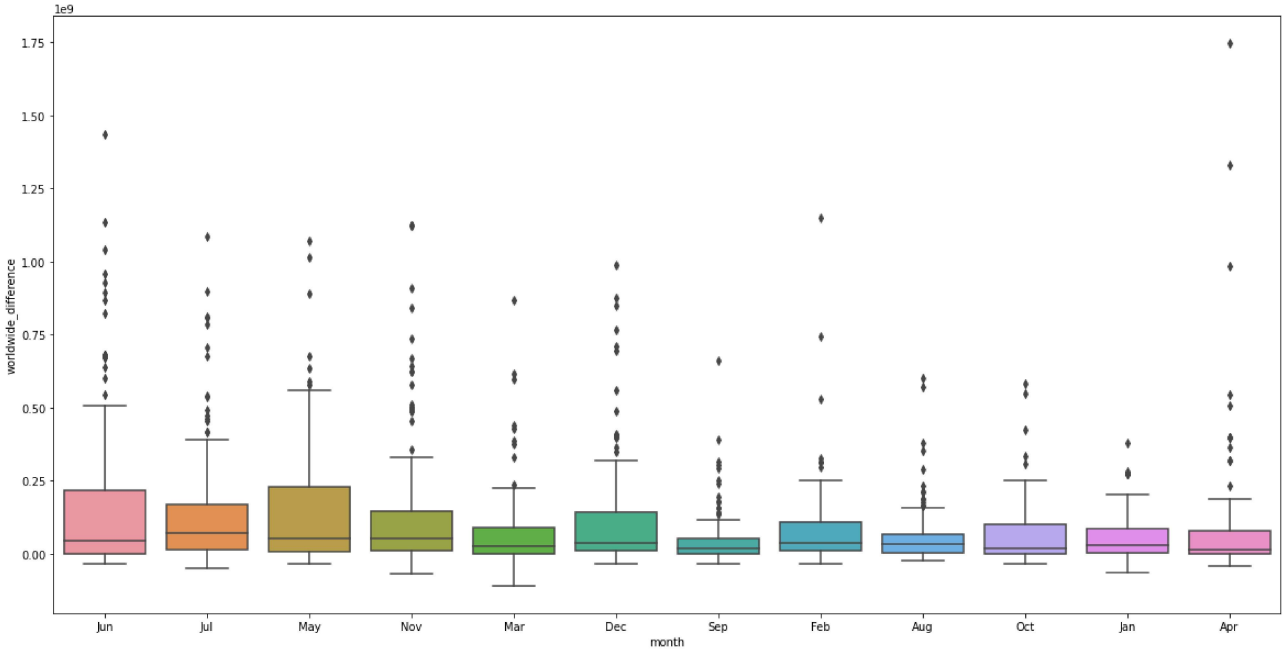
Out[108…

	title	studio	averagerating	numvotes	runtime_minutes	directors	
0	Toy Story 3	BV	8.3	682218	103	nm0881279	nm0005124,nm0004056,nm0881
1	Inception	WB	8.8	1841066	148	nm0634240	
2	Shrek Forever After	P/DW	6.3	167532	93	nm0593610	nm0825308,nm0458441,nm0501359,nm00
3	The Twilight Saga: Eclipse	Sum.	5.0	211733	124	nm1720541	nm0742
4	Iron Man 2	Par.	7.0	657690	124	nm0269463	nm0857620,nm0498278,nm1411347,nm1293

In [109…
```
fig, ax = plt.subplots()
fig.set_size_inches((20,10))
sns.boxplot(x='month',y='domestic_difference',data=finalcleaned_df,ax=ax)
plt.show()
```



In [110…
```
fig, ax = plt.subplots()
fig.set_size_inches((20,10))
sns.boxplot(x='month',y='worldwide_difference',data=finalcleaned_df,ax=ax)
plt.show()
```



With Visualizations Completed, Time to Confirm Findings

In [111…
```
finalcleaned_df.groupby('month', as_index=True)['domestic_gross'].mean()
```

Out[111…
```
month
Apr    4.579376e+07
Aug    4.501349e+07
Dec    6.737212e+07
Feb    6.048584e+07
Jan    4.145290e+07
Jul    7.826111e+07
Jun    1.002354e+08
Mar    5.200844e+07
May    8.554047e+07
Nov    7.470179e+07
Oct    3.395967e+07
Sep    3.290262e+07
Name: domestic_gross, dtype: float64
```

In [112…
```
# May has the highest average domestic gross income
```

In [113…
```
finalcleaned_df.groupby('month', as_index=True)['worldwide_gross'].mean()
```

Out[113…
```
month
Apr    1.263916e+08
Aug    9.609669e+07
Dec    1.622217e+08
Feb    1.417541e+08
Jan    8.849601e+07
Jul    2.098260e+08
Jun    2.476531e+08
Mar    1.242663e+08
May    2.317882e+08
Nov    1.995040e+08
Oct    8.869919e+07
Sep    7.435284e+07
Name: worldwide_gross, dtype: float64
```

In [114…
```
# August has the highest average worldwide gross income
```

In [115…
```
finalcleaned_df.groupby('month', as_index=True)['domestic_difference'].mean()
```

Out[115…
```
month
Apr    1.313363e+07
Aug    1.018117e+07
Dec    1.905489e+07
Feb    1.375352e+07
Jan    1.402617e+07
Jul    2.192543e+07
Jun    3.530755e+07
Mar    3.047722e+06
May    8.152970e+06
Nov    1.585212e+07
Oct    3.047366e+06
Sep    5.176227e+06
Name: domestic_difference, dtype: float64
```

In [116…
```
# May has the highest difference
```

In [117…
```
finalcleaned_df.groupby('month', as_index=True)['worldwide_difference'].mean()
```

Out[117…
```
month
Apr    9.373143e+07
Aug    6.126437e+07
Dec    1.139045e+08
Feb    9.502177e+07
Jan    6.106928e+07
Jul    1.534903e+08
Jun    1.827252e+08
Mar    7.530562e+07
May    1.544007e+08
Nov    1.406543e+08
Oct    5.778688e+07
Sep    4.662645e+07
Name: worldwide_difference, dtype: float64
```

In [118…
```
# Feb has the highest difference
```

In [119…
```
finalcleaned_df['int_ar'] = finalcleaned_df['averagerating'].astype(int)
```

In [120…
```
finalcleaned_df.head()
```

Out[120…

	title	studio	averagerating	numvotes	runtime_minutes	directors	
0	Toy Story 3	BV	8.3	682218	103	nm0881279	nm0005124,nm0004056,nm0881
1	Inception	WB	8.8	1841066	148	nm0634240	
2	Shrek Forever After	P/DW	6.3	167532	93	nm0593610	nm0825308,nm0458441,nm0501359,nm00
3	The Twilight Saga: Eclipse	Sum.	5.0	211733	124	nm1720541	nm0742
4	Iron Man 2	Par.	7.0	657690	124	nm0269463	nm0857620,nm0498278,nm1411347,nm1293

In [121…
```
finalcleaned_df.groupby('int_ar', as_index=True)['domestic_difference'].mean()
```

Out[121…
```
int_ar
1    2.930267e+07
2    4.431218e+07
3    1.111837e+07
4    9.821580e+06
```

```
5   -2.751099e+06
6    6.689148e+06
7    3.061533e+07
8    4.524509e+07
9   -3.561632e+06
Name: domestic_difference, dtype: float64
```

In [122…
```
finalcleaned_df.groupby('int_ar', as_index=True)['worldwide_difference'].mean()
```

Out[122…
```
int_ar
1    4.544898e+07
2    1.482497e+08
3    4.871673e+07
4    7.150578e+07
5    5.604755e+07
6    8.617372e+07
7    1.491513e+08
8    2.398932e+08
9   -1.363053e+06
Name: worldwide_difference, dtype: float64
```

In [123…
```
# It seems movies with an average rating of 4 had the highest average of domestic profit
# It seems movies with an average rating of 6 had the highest average of worldwide profit

# Keep in mind the difference of number of movies in for each rating
```

Creation of New Data Frames and Graphs for a Different Look / Understanding

In [124…
```
dom_pos_diff_df = finalcleaned_df[finalcleaned_df['domestic_difference'] > 0]
```

In [125…
```
ww_pos_diff_df = finalcleaned_df[finalcleaned_df['worldwide_difference'] > 0]
```

dom_pos_diff_df.head()

In [126…
```
dom_pos_diff_df.head()
```

Out[126…

	title	studio	averagerating	numvotes	runtime_minutes	directors	
0	Toy Story 3	BV	8.3	682218	103	nm0881279	nm0005124,nm0004056,nm0881
1	Inception	WB	8.8	1841066	148	nm0634240	
2	Shrek Forever After	P/DW	6.3	167532	93	nm0593610	nm0825308,nm0458441,nm0501359,nm00
3	The Twilight Saga: Eclipse	Sum.	5.0	211733	124	nm1720541	nm0742
4	Iron Man 2	Par.	7.0	657690	124	nm0269463	nm0857620,nm0498278,nm1411347,nm1293

In [127…
```
dom_pos_diff_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 706 entries, 0 to 1398
Data columns (total 15 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   title                706 non-null    object
 1   studio               706 non-null    object
 2   averagerating        706 non-null    float64
```

```
 3   numvotes              706 non-null    int64
 4   runtime_minutes       706 non-null    int64
 5   directors             706 non-null    object
 6   writers               706 non-null    object
 7   release_date          706 non-null    datetime64[ns]
 8   production_budget     706 non-null    int64
 9   domestic_gross        706 non-null    int64
 10  worldwide_gross       706 non-null    int64
 11  domestic_difference   706 non-null    int64
 12  worldwide_difference  706 non-null    int64
 13  month                 706 non-null    object
 14  int_ar                706 non-null    int32
dtypes: datetime64[ns](1), float64(1), int32(1), int64(7), object(5)
memory usage: 85.5+ KB
```

In [128…
```
ww_pos_diff_df.head()
```

Out[128…

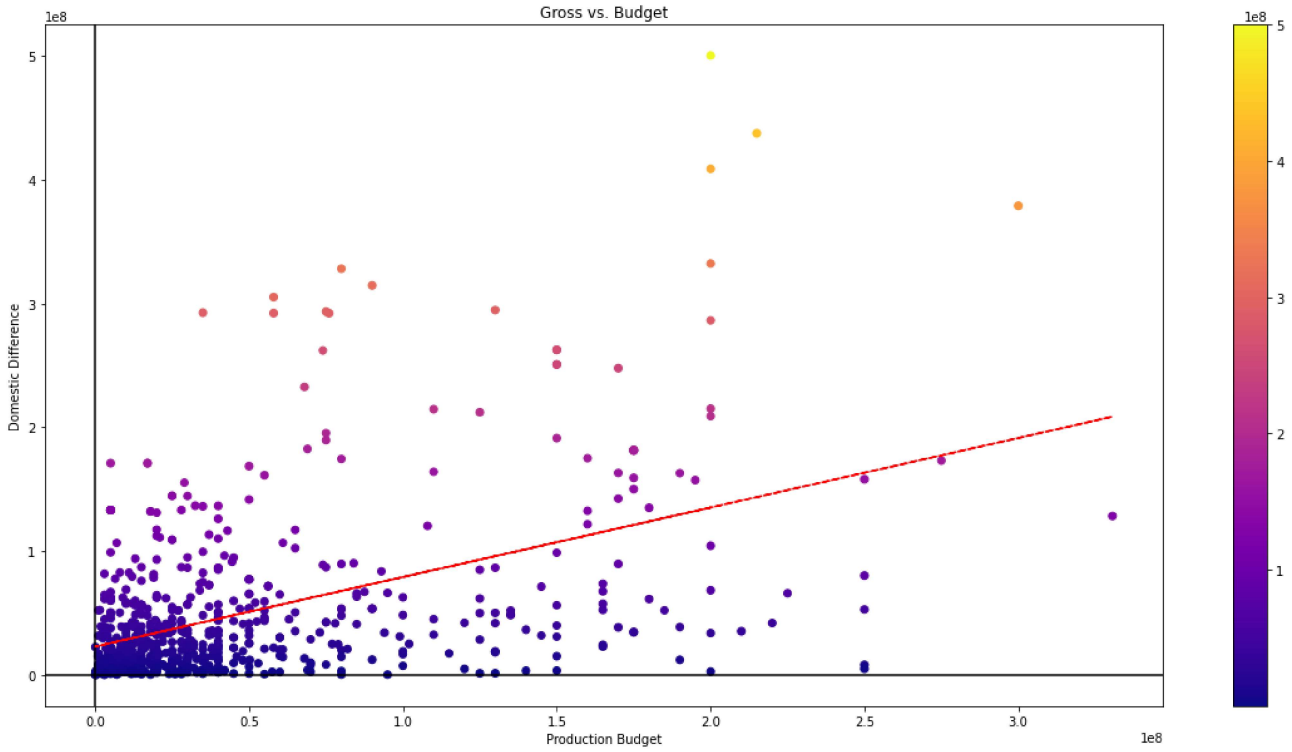
	title	studio	averagerating	numvotes	runtime_minutes	directors	
0	Toy Story 3	BV	8.3	682218	103	nm0881279	nm0005124,nm0004056,nm0881
1	Inception	WB	8.8	1841066	148	nm0634240	
2	Shrek Forever After	P/DW	6.3	167532	93	nm0593610	nm0825308,nm0458441,nm0501359,nm00
3	The Twilight Saga: Eclipse	Sum.	5.0	211733	124	nm1720541	nm0742
4	Iron Man 2	Par.	7.0	657690	124	nm0269463	nm0857620,nm0498278,nm1411347,nm1293

In [129…
```
ww_pos_diff_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1096 entries, 0 to 1404
Data columns (total 15 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   title                 1096 non-null   object
 1   studio                1096 non-null   object
 2   averagerating         1096 non-null   float64
 3   numvotes              1096 non-null   int64
 4   runtime_minutes       1096 non-null   int64
 5   directors             1096 non-null   object
 6   writers               1096 non-null   object
 7   release_date          1096 non-null   datetime64[ns]
 8   production_budget     1096 non-null   int64
 9   domestic_gross        1096 non-null   int64
 10  worldwide_gross       1096 non-null   int64
 11  domestic_difference   1096 non-null   int64
 12  worldwide_difference  1096 non-null   int64
 13  month                 1096 non-null   object
 14  int_ar                1096 non-null   int32
dtypes: datetime64[ns](1), float64(1), int32(1), int64(7), object(5)
memory usage: 132.7+ KB
```

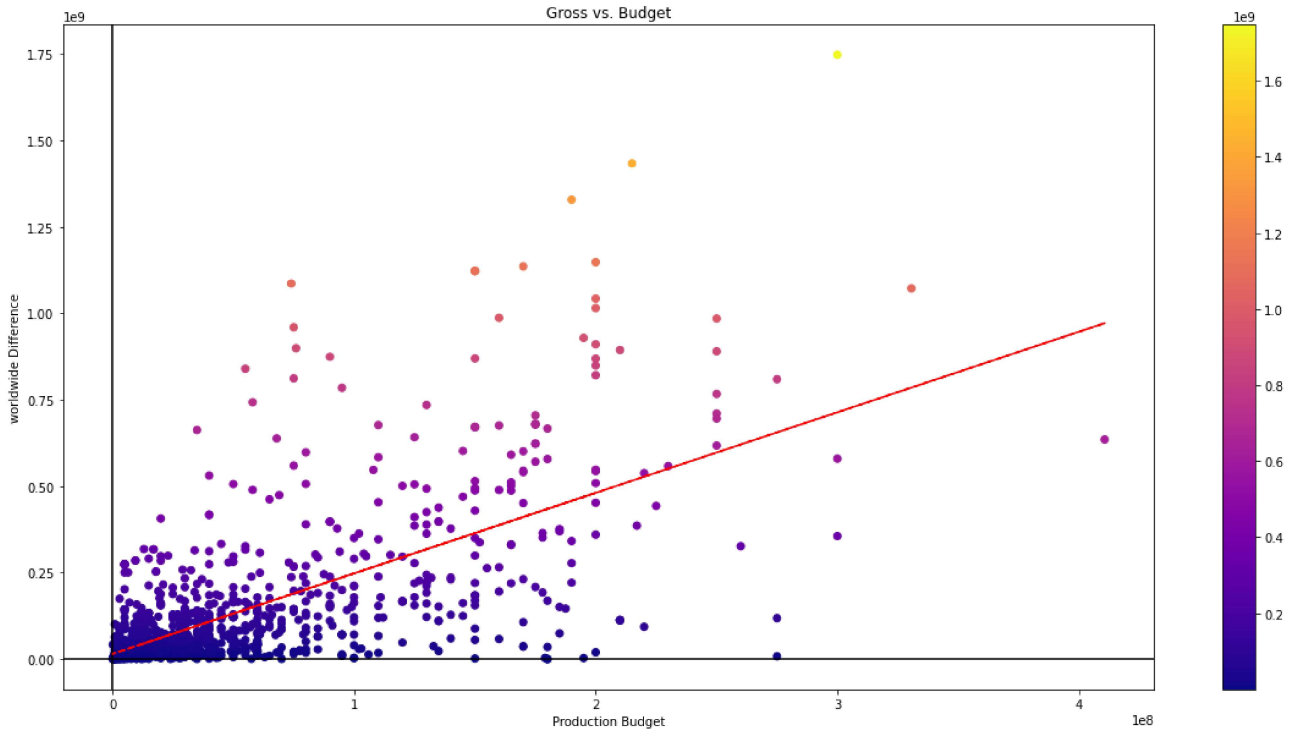
In [130…
```
x = dom_pos_diff_df['production_budget']
y = dom_pos_diff_df['domestic_difference']

plt.figure(figsize = (20, 10))
plt.scatter(x, y, c=y, cmap = 'plasma')
plt.axvline(0, c = 'black')
plt.axhline(0, c = 'black')
```

```
plt.xlabel("Production Budget")
plt.ylabel("Domestic Difference")
plt.title("Gross vs. Budget")

z = np.polyfit(x, y, 1)
p = np.poly1d(z)
plt.plot(x,p(x),"r--")

plt.colorbar()
plt.show()
```



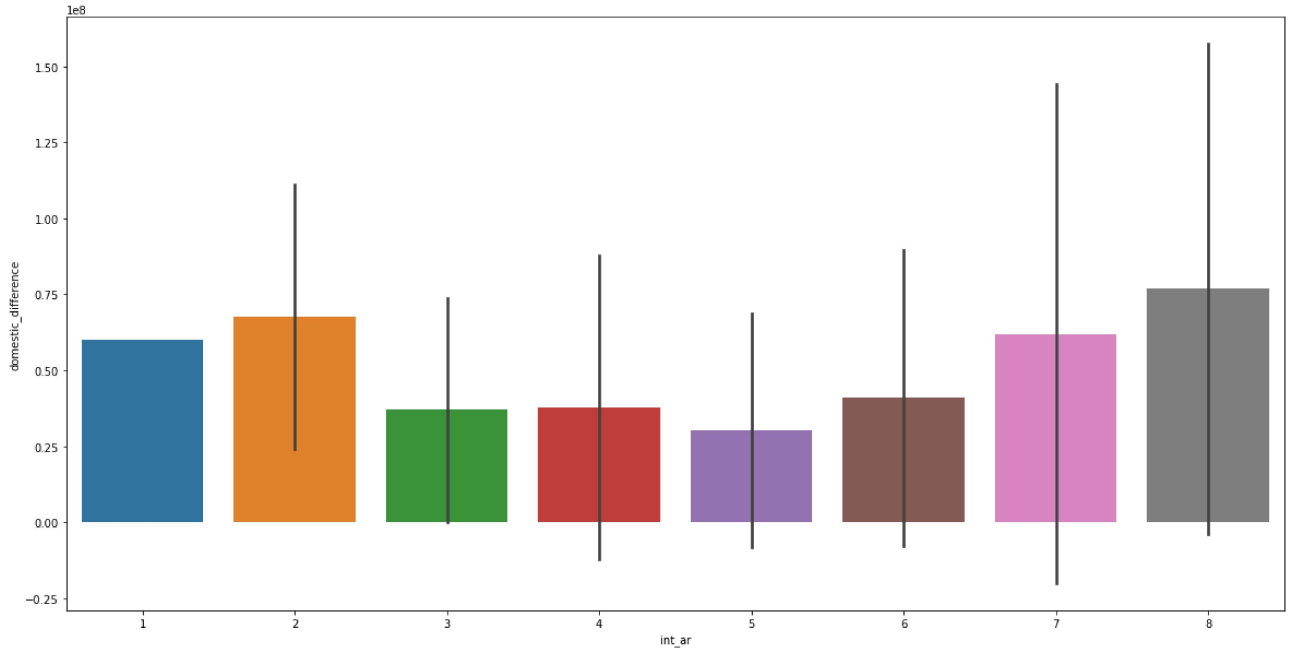
In [131…

```
x = ww_pos_diff_df['production_budget']
y = ww_pos_diff_df['worldwide_difference']

plt.figure(figsize = (20, 10))
plt.scatter(x, y, c=y, cmap = 'plasma')
plt.axvline(0, c = 'black')
plt.axhline(0, c = 'black')

plt.xlabel("Production Budget")
plt.ylabel("worldwide Difference")
plt.title("Gross vs. Budget")

z = np.polyfit(x, y, 1)
p = np.poly1d(z)
plt.plot(x,p(x),"r--")

plt.colorbar()
plt.show()
```



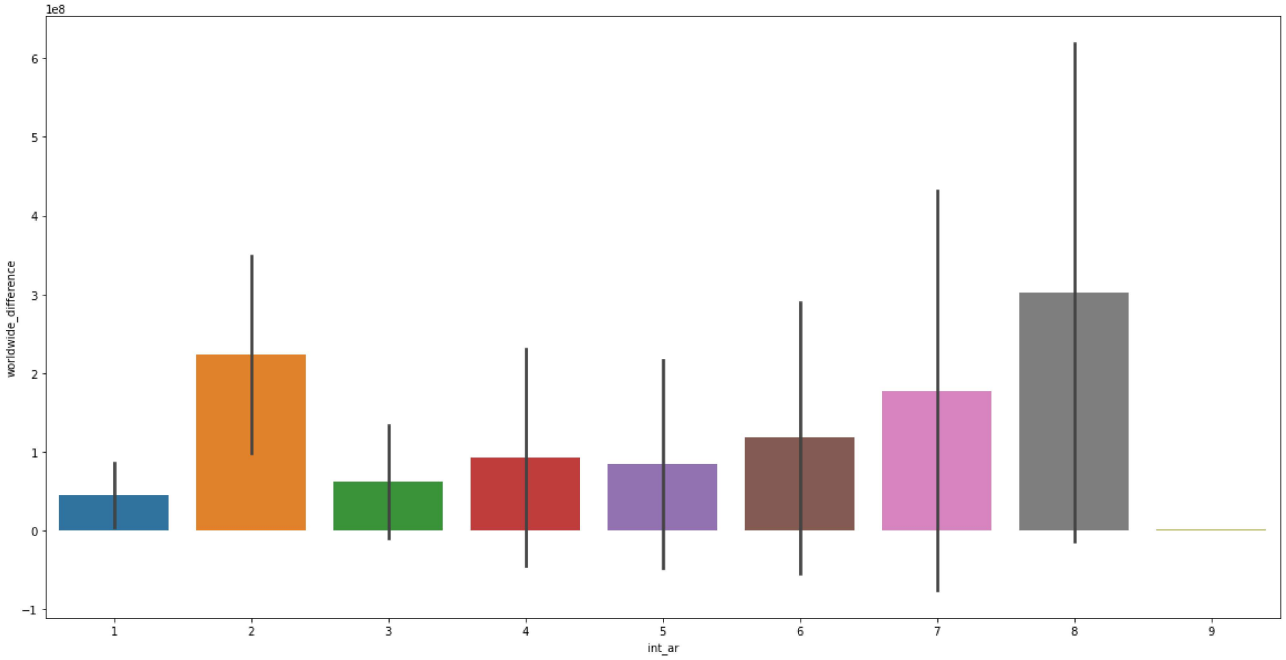
In [132…
```
fig, ax = plt.subplots()
fig.set_size_inches((20,10))
sns.barplot(x="int_ar", y="domestic_difference", data=dom_pos_diff_df, ci="sd")
```

Out[132… <AxesSubplot:xlabel='int_ar', ylabel='domestic_difference'>



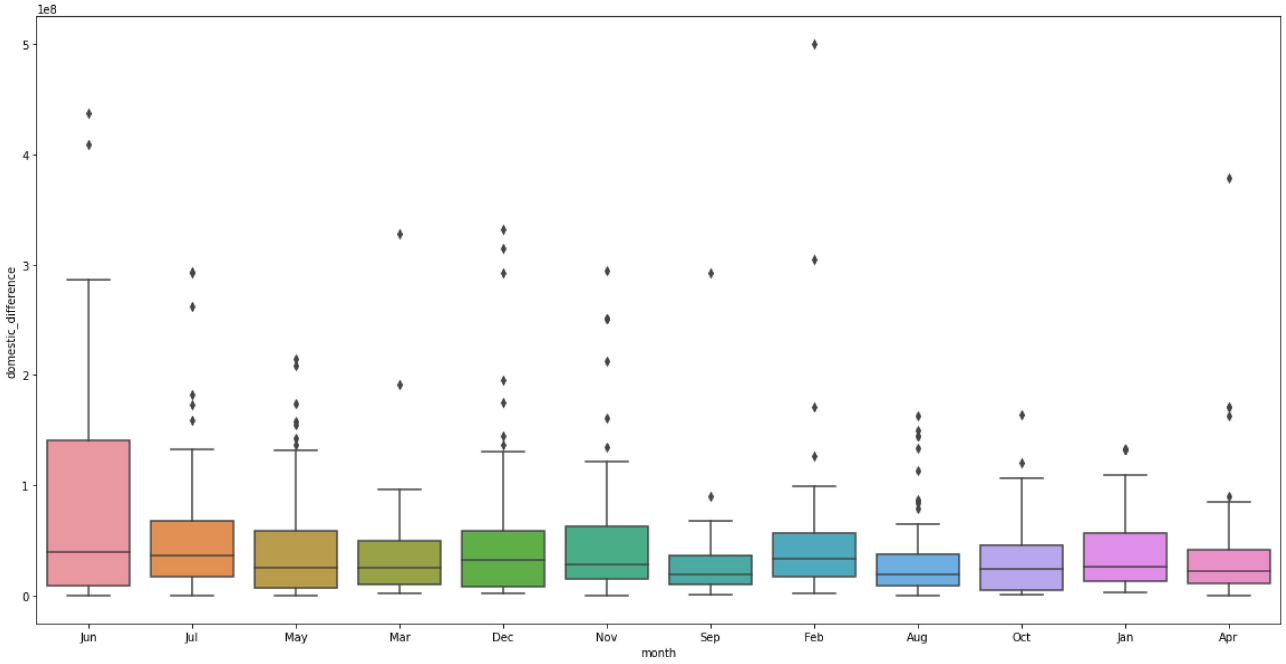
In [133…
```
fig, ax = plt.subplots()
fig.set_size_inches((20,10))
sns.barplot(x="int_ar", y="worldwide_difference", data=ww_pos_diff_df, ci="sd")
```

Out[133… <AxesSubplot:xlabel='int_ar', ylabel='worldwide_difference'>



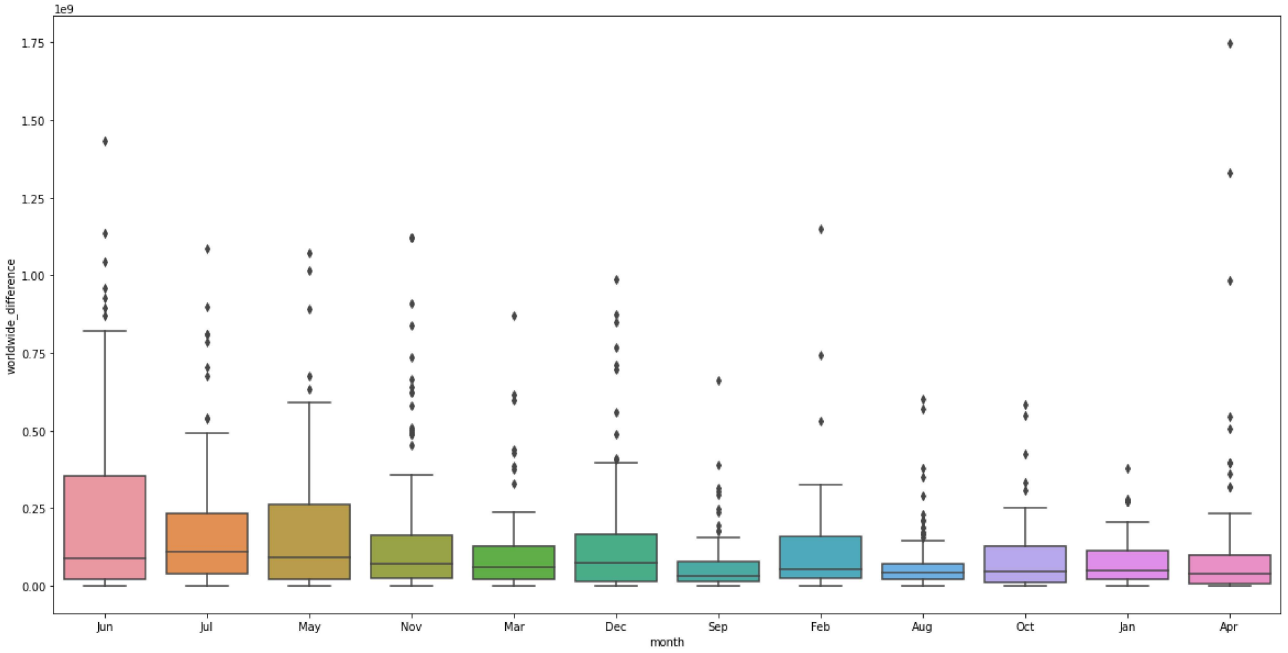
In [134…

```
fig, ax = plt.subplots()
fig.set_size_inches((20,10))
sns.boxplot(x='month',y='domestic_difference',data=dom_pos_diff_df,ax=ax)
plt.show()
```



In [135…

```
fig, ax = plt.subplots()
fig.set_size_inches((20,10))
sns.boxplot(x='month',y='worldwide_difference',data=ww_pos_diff_df,ax=ax)
plt.show()
```



In [136…
```
dom_pos_diff_df.groupby('month', as_index=True)['domestic_difference'].mean()
```

Out[136…
```
month
Apr    4.037548e+07
Aug    3.690016e+07
Dec    5.178366e+07
Feb    5.459813e+07
Jan    4.115104e+07
Jul    5.629612e+07
Jun    8.645974e+07
Mar    3.836554e+07
May    4.922720e+07
Nov    5.144501e+07
Oct    3.320405e+07
Sep    3.035262e+07
Name: domestic_difference, dtype: float64
```

In [137…
```
ww_pos_diff_df.groupby('month', as_index=True)['worldwide_difference'].mean()
```

Out[137…
```
month
Apr    1.293251e+08
Aug    7.745949e+07
Dec    1.394131e+08
Feb    1.213736e+08
Jan    8.268804e+07
Jul    1.886464e+08
Jun    2.475511e+08
Mar    1.112896e+08
May    1.872412e+08
Nov    1.722788e+08
Oct    8.469473e+07
Sep    7.134185e+07
Name: worldwide_difference, dtype: float64
```

All Graphs and Information Needed Completed!

In [138…
```
# Time to wrap things up!!!

# Realized I was working on the dsc repo and should have started with the creation of the repo for
# I believe I have properly converted / transferred all files to my repo for this project
```