BUSINESS QUESTIONS

- What is highest and lowest budgets used and for which movies
- What is the target audience(target genre)
- Which movies have the highest icome and the genre as well
- The top performing studios

MICROSOFT MOVIE PROJECT

The aim of this projct is to give high reccommendations of movies Microsoft can use to create. Analysis will be performed moreso correlation to give us the reccoommendations Microsoft can use. EDA and data analysis will be performed. Also answered are the business questions helped to formulate the recommendation ad solution that will be provided.

1)In depth understanding of the Data

In [3]:
```
#Import the libraries needed: pandas, numpy and matplotlib
import pandas as pd
import numpy as np
import csv
import seaborn as sns
import matplotlib.pyplot as plt
import sqlite3
import zipfile
%matplotlib inline
```

In [4]:
```
#load the datasets that will be used
bom_movie_df = pd.read_csv('zippedData/bom.movie_gross.csv', index_col=0)

#load the dataset #tn.movie_budgets.csv
tn_df = pd.read_csv('zippedData/tn.movie_budgets.csv', index_col=0)


#load the dataset # rt.movie_info.tsv
movieinfo_df = pd.read_csv('zippedData/rt.movie_info.tsv', sep = '\t')


#load the dataset #tmdb.movies.csz
tmdb_df = pd.read_csv('zippedData/tmdb.movies.csv', index_col=0)

#load the dataset im.db
```

In [5]:
```
bom_movie_df = pd.read_csv('zippedData/bom.movie_gross.csv', index_col=0)
bom_movie_df
```

Out[5]:

title	studio	domestic_gross	foreign_gross	year
Toy Story 3	BV	415000000.0	652000000	2010
Alice in Wonderland (2010)	BV	334200000.0	691300000	2010
Harry Potter and the Deathly Hallows Part 1	WB	296000000.0	664300000	2010
Inception	WB	292600000.0	535700000	2010
Shrek Forever After	P/DW	238700000.0	513900000	2010
...	...	...	...	...
The Quake	Magn.	6200.0	NaN	2018
Edward II (2018 re-release)	FM	4800.0	NaN	2018
El Pacto	Sony	2500.0	NaN	2018
The Swan	Synergetic	2400.0	NaN	2018
An Actor Prepares	Grav.	1700.0	NaN	2018

3387 rows × 4 columns

In [6]:
```
bom_movie_df.describe()
```

Out[6]:

	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 [7]:
```
tn_df = pd.read_csv('zippedData/tn.movie_budgets.csv', index_col=0)
tn_df
```

Out[7]:

id	release_date	movie	production_budget	domestic_gross	worldwide_gross
1	Dec 18, 2009	Avatar	$425,000,000	$760,507,625	$2,776,345,279
2	May 20, 2011	Pirates of the Caribbean: On Stranger Tides	$410,600,000	$241,063,875	$1,045,663,875
3	Jun 7, 2019	Dark Phoenix	$350,000,000	$42,762,350	$149,762,350
4	May 1, 2015	Avengers: Age of Ultron	$330,600,000	$459,005,868	$1,403,013,963
5	Dec 15, 2017	Star Wars Ep. VIII: The Last Jedi	$317,000,000	$620,181,382	$1,316,721,747
...	...	...	...	...	...
78	Dec 31, 2018	Red 11	$7,000	$0	$0
79	Apr 2, 1999	Following	$6,000	$48,482	$240,495
80	Jul 13, 2005	Return to the Land of Wonders	$5,000	$1,338	$1,338
81	Sep 29, 2015	A Plague So Pleasant	$1,400	$0	$0
82	Aug 5, 2005	My Date With Drew	$1,100	$181,041	$181,041

5782 rows × 5 columns

In [8]:
```
tn_df.describe()
```

Out[8]:

	release_date	movie	production_budget	domestic_gross	worldwide_gross
count	5782	5782	5782	5782	5782
unique	2418	5698	509	5164	5356
top	Dec 31, 2014	Home	$20,000,000	$0	$0
freq	24	3	231	548	367

Merge the tables that will be required

bom_movie_df ' and ' tn_df '. In this case the recommended method of joining the tables is by use of concact method. using the join method in this case overlaps the columns and also some of these columns have a differrent data type, hence recommend concact. Shape of the merged data With the merged dataset we have to know the shape of the dataset. Columns, rows, statistics in this case the mean, median and standard deviation.

In [9]:
```
#merged data
new_movie_df = pd.merge(bom_movie_df,tn_df, how = "inner", left_on="title",right_on
new_movie_df
```

Out[9]:

	studio	domestic_gross_x	foreign_gross	year	release_date	movie	production_budget
0	BV	415000000.0	652000000	2010	Jun 18, 2010	Toy Story 3	$200,000,000
1	WB	292600000.0	535700000	2010	Jul 16, 2010	Inception	$160,000,000
2	P/DW	238700000.0	513900000	2010	May 21, 2010	Shrek Forever After	$165,000,000
3	Sum.	300500000.0	398000000	2010	Jun 30, 2010	The Twilight Saga: Eclipse	$68,000,000
4	Par.	312400000.0	311500000	2010	May 7, 2010	Iron Man 2	$170,000,000
...	...	...	...	...	...	...	...
1242	VE	4300000.0	NaN	2018	Jun 15, 2018	Gotti	$10,000,000
1243	RAtt.	3700000.0	NaN	2018	Dec 7, 2018	Ben is Back	$13,000,000
1244	VE	491000.0	1700000	2018	Feb 2, 2018	Bilal: A New Breed of Hero	$30,000,000
1245	RLJ	1200000.0	NaN	2018	Sep 14, 2018	Mandy	$6,000,000
1246	A24	1200000.0	NaN	2018	Apr 6, 2018	Lean on Pete	$8,000,000

1247 rows × 9 columns

In [10]:
```
#In this case we have 9169 rows and 8 columns
new_movie_df.shape
```

Out[10]:
```
(1247, 9)
```

In [11]:
```
#The .columns() function is used to identify the column names of the dataframe
new_movie_df.columns
```

Out[11]:
```
Index(['studio', 'domestic_gross_x', 'foreign_gross', 'year', 'release_date',
       'movie', 'production_budget', 'domestic_gross_y', 'worldwide_gross'],
      dtype='object')
```

In [12]:
```
#Used the .info() function to give the column names and the datatypes. In this case
new_movie_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1247 entries, 0 to 1246
Data columns (total 9 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   studio             1246 non-null   object 
 1   domestic_gross_x   1245 non-null   float64
 2   foreign_gross      1086 non-null   object 
 3   year               1247 non-null   int64  
 4   release_date       1247 non-null   object 
 5   movie              1247 non-null   object 
 6   production_budget  1247 non-null   object 
 7   domestic_gross_y   1247 non-null   object 
 8   worldwide_gross    1247 non-null   object 
dtypes: float64(1), int64(1), object(7)
memory usage: 97.4+ KB
```

In [13]:

```
#Change the 'Year' to the Dtype 'Object'
new_movie_df = new_movie_df.astype({'year':'object'})
new_movie_df.dtypes
```

Out[13]:

```
studio                object
domestic_gross_x     float64
foreign_gross         object
year                  object
release_date          object
movie                 object
production_budget     object
domestic_gross_y      object
worldwide_gross       object
dtype: object
```

In [14]:

```
#To know the statistics of the data: Mean, Median and std, count etc
new_movie_df.describe()
```

Out[14]:

	domestic_gross_x
count	1.245000e+03
mean	6.062353e+07
std	8.477607e+07
min	8.000000e+02
25%	7.500000e+06
50%	3.340000e+07
75%	7.420000e+07
max	7.001000e+08

2)Data Preparation

With the merged dataset, we have to perform EDA. In this case, we identify duplicated values, missing values and the percentage of the missing values. Using the Percentage in this case will give us a method or rather solution to the missing values: drop the missing values or use diffrerent methods to fill the missing values. Henceforth, we perform data cleaning to reduce inconsistency and improve accuracy

2.1) Check for duplicate values

In [15]:
```
#Identify the duplicated values and check how many we have.
new_movie_df.duplicated().value_counts()
```

Out[15]:
```
False    1247
dtype: int64
```

In [16]:
```
#We reomove the duplicated values and this makes a diffrence from the previous.
new_movie_df = new_movie_df.drop_duplicates()
new_movie_df.shape
```

Out[16]:
```
(1247, 9)
```

2.1) Missing values

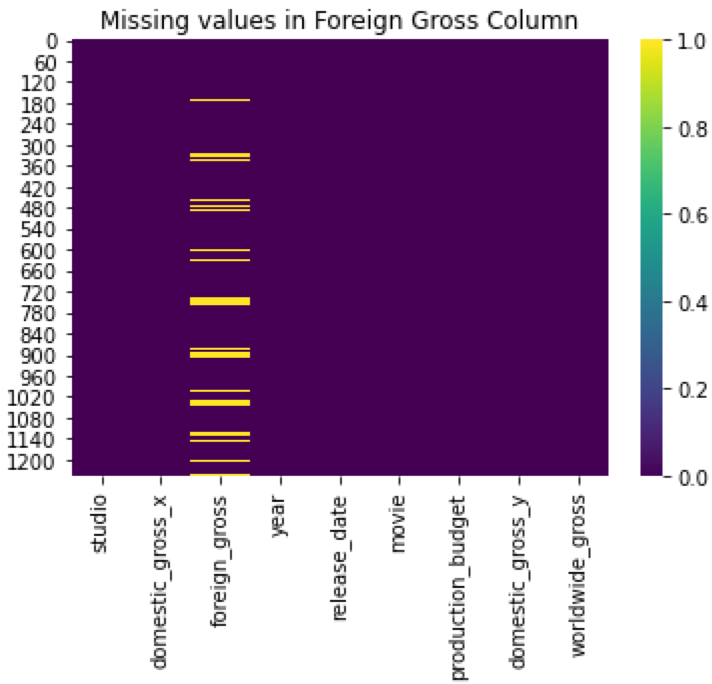
In [17]:
```
#We check for missing values in the merged dataframe, represented by Nan(null value
new_movie_df.isna().sum()
```

Out[17]:
```
studio                1
domestic_gross_x      2
foreign_gross       161
year                  0
release_date          0
movie                 0
production_budget     0
domestic_gross_y      0
worldwide_gross       0
dtype: int64
```

In [18]:
```
#Check the percentage of missing values for 'foreign_gross' column since it has the
print('Percentage of Null foreign_gross Values:', len(new_movie_df[new_movie_df.for
```

```
Percentage of Null foreign_gross Values: 0.12910986367281477
```

In [19]:
```
#Visually represent the missing values with a heatmap
sns.heatmap(new_movie_df.isnull(), cmap = 'viridis')
plt.title('Missing values in Foreign Gross Column')
plt.show()
```



In [20]:
```
#Convert 'foreign_gross' column to a numeric data type
new_movie_df['foreign_gross'] = pd.to_numeric(new_movie_df['foreign_gross'], errors

#Calculate the mean and median of the foreign_gross column in 'new_movie_budget_df'

foreign_gross_mean = new_movie_df['foreign_gross'].mean()
foreign_gross_median = new_movie_df['foreign_gross'].median()


print("Mean Value for foreign_gross column: {}".format(foreign_gross_mean))
print("Median Value for foreign_gross column: {}".format(foreign_gross_median))
```

```
Mean Value for foreign_gross column: 100945897.5896488
Median Value for foreign_gross column: 38050000.0
```

In [21]:
```
new_movie_df['foreign_gross'].isna().sum
```

Out[21]:
```
<bound method Series.sum of 0       False
1       False
2       False
3       False
4       False
        ...
1242     True
1243     True
1244    False
1245     True
1246     True
Name: foreign_gross, Length: 1247, dtype: bool>
```

In [22]:
```
new_movie_df['production_budget'] =new_movie_df['production_budget'].replace('[\$,]
new_movie_df['worldwide_gross'] =new_movie_df['worldwide_gross'].replace('[\$,]', '
new_movie_df['domestic_gross_x'] =new_movie_df['domestic_gross_x'].replace('[\$,]',
new_movie_df['domestic_gross_y'] =new_movie_df['domestic_gross_y'].replace('[\$,]',
```

Clean the merged data 'reviews_df'

- This is a dataframe that consist of 'tn_df' and 'movieinfo_df')

In [23]:
```
reviews_df = pd.merge(tn_df, movieinfo_df, how = "inner", left_on="id", right_on ="
```

In [24]:
```
#Drop the columns with missing values
reviews_df = reviews_df.dropna()
reviews_df.head()
```

Out[24]:

	id	release_date	movie	production_budget	domestic_gross	worldwide_gross	synopsis	rati
58	3	Jun 7, 2019	Dark Phoenix	$350,000,000	$42,762,350	$149,762,350	New York City, not-too-distant-future: Eric Pa...	
59	3	Nov 21, 2018	Ralph Breaks The Internet	$175,000,000	$201,091,711	$524,283,695	New York City, not-too-distant-future: Eric Pa...	
60	3	Apr 8, 2005	Sahara	$145,000,000	$68,671,925	$121,671,925	New York City, not-too-distant-future: Eric Pa...	
61	3	Oct 5, 2018	Venom	$116,000,000	$213,511,408	$853,628,605	New York City, not-too-distant-future: Eric Pa...	
62	3	Feb 18, 2005	Son of the Mask	$100,000,000	$17,018,422	$59,918,422	New York City, not-too-distant-future: Eric Pa...	

Data Analysis

i) What was the Highest profit?

- The data for analysis used in this case is the new merged data
- The output in this case will be based on the worldwide and domestic gross to give the profits made
- Also we need to find the movie that made the highest profit
- Plot showing the profit made

In [25]:
```
#Created a column called profit_needed
new_movie_df["profit_made"] = (new_movie_df["worldwide_gross"] - new_movie_df["dome


#Find the highest profit
highest_profit = new_movie_df["profit_made"].max()

print("The highest profit made is:" , highest_profit)
```

```
The highest profit made is: 1369318718.0
```

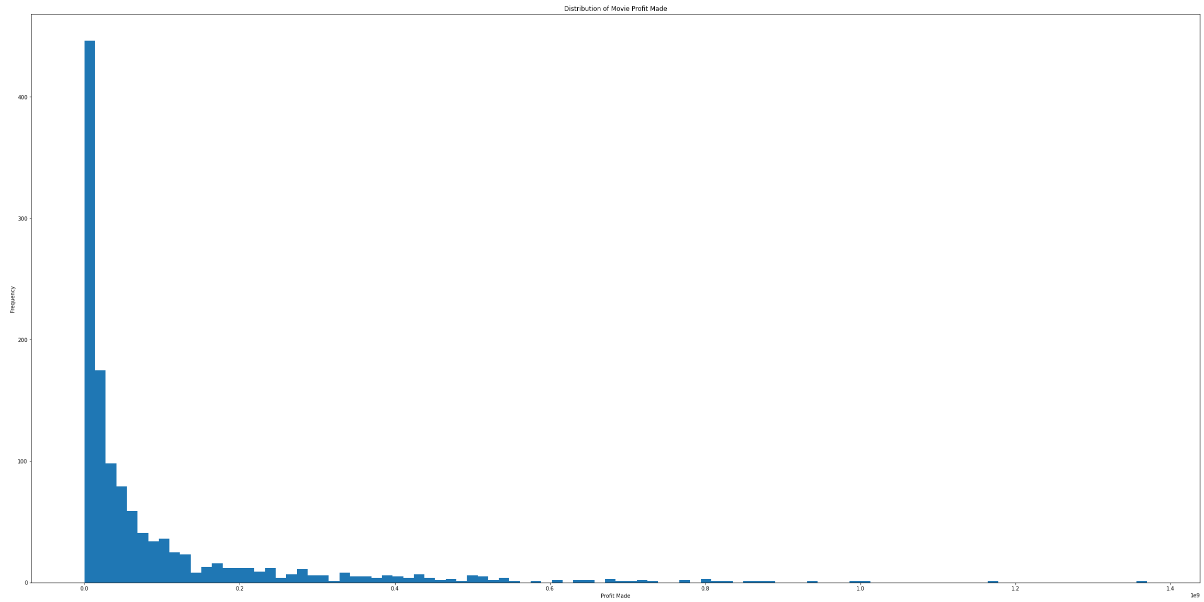
In [26]:
```
#Calculating the percentage profit
new_movie_df["percentage_profit"] = (new_movie_df["profit_made"] / new_movie_df["pr

# Finding the highest percentage
highest_percentage_profit = new_movie_df["percentage_profit"].max()

print("The highest percentage profit made is:", highest_percentage_profit)
```

```
The highest percentage profit made is: 18892.064
```

In [27]:
```
#Find the movies that made the highest profit
profitable_movies = new_movie_df[new_movie_df["profit_made"] >0]["movie"].tolist()

#To find the movie that made the highest profit
highest_profit = new_movie_df['profit_made'].idxmax()
highest_profit_movie = new_movie_df.loc[highest_profit]['movie']

print("The movie with the highest profit made is", highest_profit_movie)
```

```
The movie with the highest profit made is Avengers: Infinity War
```

In [28]:
```
#Plot a histogram of the profits made
fg, ax = plt.subplots(figsize=(40, 20))
x = new_movie_df["profit_made"]
y = new_movie_df["movie"]

plt.hist(new_movie_df['profit_made'], bins=100)
plt.xlabel('Profit Made')
plt.ylabel('Frequency')
plt.title('Distribution of Movie Profit Made')
plt.show()
```



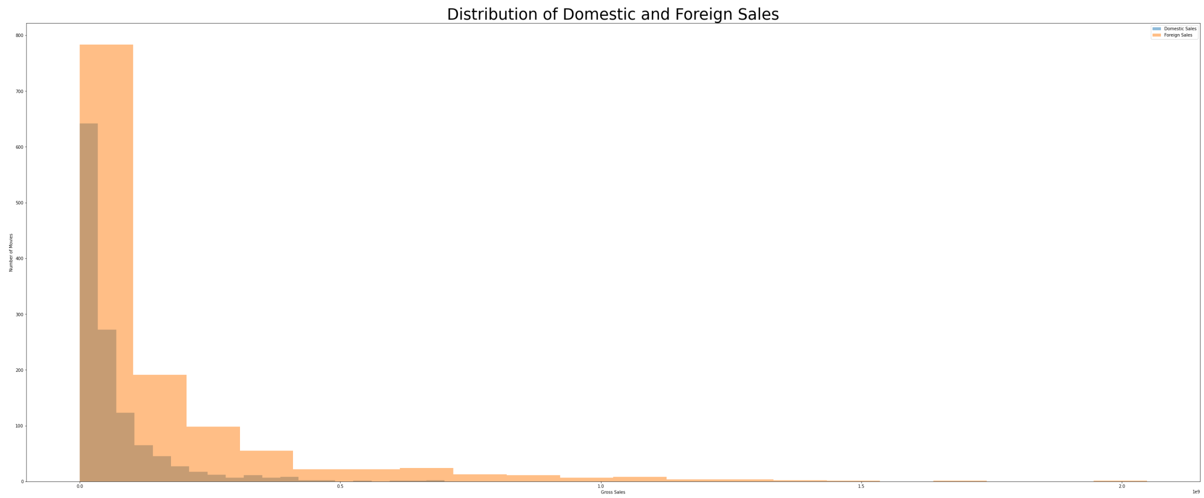
ii) What's the company's focal point

- This will depend on the foreign_gross and domestic_gross_y(the sales made)
- Plot the domestic sales and foreign sales.

In [44]:
```
print('The average worldwide sales: {} \nThe average foreign sales: {}'.format(
                                                                        new_movie_df
                                                                        new_movie_df
print('The Difference: {}'.format(
                                (new_movie_df["domestic_gross_x"].mean()-new_movie_df["
```

```
The average worldwide sales: 60623530.9124498
The average foreign sales: 152125895.94947875
The Difference: -91502365.03702894
```

In [32]:
```
#Set to plot
fg, ax = plt.subplots(figsize=(50, 20))

# Define the domestic and foreign sales
plt.hist(new_movie_df['domestic_gross_x'], bins=20, alpha=0.5, label='Domestic Sale
plt.hist(new_movie_df['worldwide_gross'], bins=20, alpha=0.5, label='Foreign Sales'

#Label the axes
plt.xlabel('Gross Sales')
plt.ylabel('Number of Movies')
plt.title('Distribution of Domestic and Foreign Sales', fontsize=37)
plt.legend()
plt.show()
```



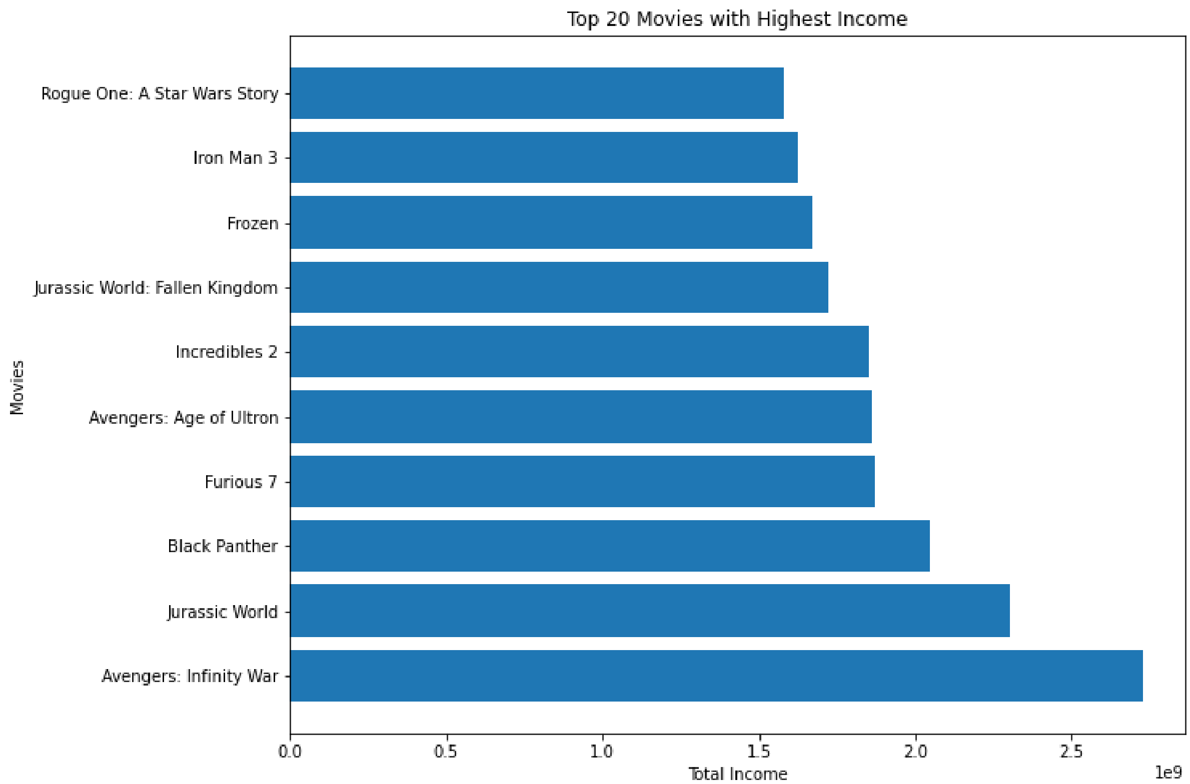
Which movies brought in the highest income.

- In this case we want to find the first 20 movies that brought the highest income
- The output is based on 'domestic_gross and worldwide_gross
- Plot showing the income and movies

In [34]:
```
# Select the necessary columns
movie_income = new_movie_df[["movie", "domestic_gross_y", "worldwide_gross"]]

# Calculate the total income for each movie
movie_income.loc[:,"total_income"] = movie_income["domestic_gross_y"] + movie_incom

# Sort the movies by total income in descending order
movie_income = movie_income.sort_values(by="total_income", ascending=False)

# Get the top 10 highest income movies
top20_movies = movie_income.head(10)

#Print the list of High Income movies
print("Top 20 Movies with Highest Income")

print(top20_movies["movie"])
```

```
Top 20 Movies with Highest Income
1154             Avengers: Infinity War
764                      Jurassic World
1155                      Black Panther
765                           Furious 7
766             Avengers: Age of Ultron
1157                      Incredibles 2
1156    Jurassic World: Fallen Kingdom
496                              Frozen
497                          Iron Man 3
911         Rogue One: A Star Wars Story
Name: movie, dtype: object
```

In [35]:
```
# Plot the bar chart
fig, ax = plt.subplots(figsize=(10, 8))
ax.barh(top20_movies["movie"], top20_movies["total_income"])
ax.set_xlabel("Total Income")
ax.set_ylabel("Movies")
ax.set_title("Top 20 Movies with Highest Income")

plt.show()
```



Finding the top studios and how much income they generate

- We use 'new movie_df' which is a merged dataframe of 'bom.movies and 'tn.movies

*Plot showing the information

In [36]:
```
# Group the data by genre and calculates the sum of the domestic and worldwide gros
studio_income =new_movie_df.groupby('studio')[['domestic_gross_y', 'worldwide_gross

#Creating a new column 'total_gross' summing up the two columns
studio_income['total_gross'] = studio_income['domestic_gross_y'] + studio_income['w

#Sort the data frame by the total gross in descending order and have the genres wit
studio_income = studio_income.sort_values('total_gross', ascending=False)

studio_income.head(10)
```

Out[36]:

	studio	domestic_gross_y	worldwide_gross	total_gross
15	BV	1.292614e+10	3.328602e+10	4.621216e+10
90	Uni.	1.070684e+10	2.732929e+10	3.803613e+10
32	Fox	9.410234e+09	2.679581e+10	3.620605e+10
94	WB	9.130528e+09	2.219381e+10	3.132434e+10
82	Sony	7.059959e+09	1.760181e+10	2.466177e+10
69	Par.	6.007203e+09	1.443821e+10	2.044541e+10
95	WB (NL)	3.417630e+09	8.540864e+09	1.195849e+10
48	LGF	3.332465e+09	6.983389e+09	1.031585e+10
64	P/DW	1.682915e+09	5.078028e+09	6.760942e+09
47	LG/S	1.499805e+09	3.815925e+09	5.315730e+09

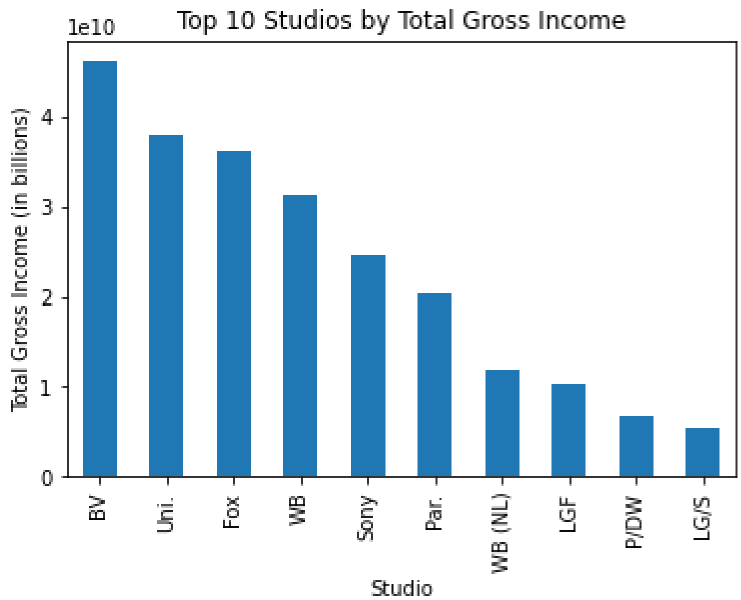
The top studios include:

- BV
- Uni.
- Fox
- WB
- Sony
- Par
- LGF
- P/DW
- LG/S

The studios enlisted above makes the highest total gross and Microsoft can indeed invest in these studios from movies produced or owned by these studios

In [37]:

```
# select the top 10 studios by total gross income
top_10_studios = studio_income.head(10)

# plot a bar graph of the total gross income for each studio
top_10_studios.plot(x='studio', y='total_gross', kind='bar', legend=None)

# set the plot title and axis labels
plt.title('Top 10 Studios by Total Gross Income')
plt.xlabel('Studio')
plt.ylabel('Total Gross Income (in billions)')

# show the plot
plt.show()
```



The Genres that brought the highest income

- To find this we merge the 'tn_df' and movieinfo_df
- The output will be the income based on the domestic and worldwide gross
- Plot showing the profitable genres

In [38]:
```
reviews_df = pd.merge(tn_df, movieinfo_df, how = "inner", left_on="id", right_on ="
```

In [40]:
```
# Group the data by genre and calculates the sum of the domestic and worldwide gros
genre_income = reviews_df.groupby('genre')[['domestic_gross', 'worldwide_gross']].s

#Creating a new column 'total_gross' summing up the two columns
genre_income['total_gross'] = genre_income['domestic_gross'] + genre_income['worldw

#Sort the data frame by the total gross in descending order and have the genres wit
genre_income = genre_income.sort_values('total_gross', ascending=False)

genre_income.head(15)
```

Out[40]:

	genre	domestic_gross	worldwide_gross	
41	Drama|Mystery and Suspense	936, 662, 225150,201,498 305, 411, 224310,676,7...	2, 053, 311, 220 302,469,017634, 954, 103 656,695...	936, 662, 22 305, 411, 2
1	Action and Adventure|Art House and Internation...	92, 054, 159133,501,348 54, 647, 948 89,296,573$...	259, 357, 408397,501,348 240, 759, 682169,296,5...	92, 054, 15
37	Drama	89, 302, 115417,719,760 122, 523, 060 42,779,261...	260, 002, 1151,305,772,799 375, 740, 705232,017...	89, 302, 11 1
31	Comedy|Kids and Family|Romance	80, 101, 125206,445,654 217, 536, 138 203,464,10...	409, 953, 905626,549,695 767, 820, 459515,419,8...	80, 101, 12 2
2	Action and Adventure|Classics|Drama	760, 507, 625293,004,164 111, 506, 43030,212,62...	2, 776, 345, 279 731,463,377269, 806, 430 49,628,...	760, 507, 62 111, 506, 4
3	Action and Adventure|Classics|Drama|Mystery an...	700, 059, 566146,408,305 215, 434, 5913,100,000...	1, 348, 258, 224 355,408,305631, 910, 531 3,100,0...	700, 059, 56 215, 434, 5
42	Drama|Romance	678, 815, 482113,805,681 257, 730, 019289,423,4...	2, 048, 134, 200 221,229,335386, 839, 614 559,757...	678, 815, 48 257, 730, 0
22	Classics|Comedy|Musical and Performing Arts	659, 363, 944127,509,326 180, 202, 163315,544,7...	2, 208, 208, 395 329,631,958518, 858, 449 887,210...	659, 363, 94 180, 202, 1
11	Action and Adventure|Mystery and Suspense	652, 270, 625100,368,560 63, 478, 83866,184,051...	1, 648, 854, 864 176,038,324163, 818, 556 172,022...	652, 270, 62 63, 478, 83
46	Musical and Performing Arts	623, 279, 547238,736,787 183, 637, 894102,608,8...	1, 517, 935, 897 756,244,673532, 938, 302 208,370...	623, 279, 54 183, 637, 8
39	Drama|Musical and Performing Arts	620, 181, 382152,901,115 160, 942, 139380,270,5...	1, 316, 721, 747 383,541,369431, 942, 139 848,998...	620, 181, 38 160, 942, 1
30	Comedy|Kids and Family	532, 177, 32460,674,817 171, 958, 438 58,183,966...	1, 049, 102, 856 181,674,817341, 528, 518 87,683,...	532, 177, 3 1
32	Comedy|Musical and Performing Arts	486, 295, 56147,225,655 149, 260, 504 228,433,66...	1, 021, 215, 193 425,522,281554, 987, 477 655,271...	486, 295, 5 1
23	Classics|Comedy|Musical and Performing Arts|Ro...	423, 315, 812232,641,920 126, 930, 660100,014,6...	1, 066, 215, 812 676,404,566361, 730, 660 302,239...	423, 315, 81 126, 930, 6
43	Drama|Science Fiction and Fantasy	42, 762, 350201,091,711 68, 671, 925 213,511,408...	149, 762, 350524,283,695 121, 671, 925853,628,6...	42, 762, 35

In [43]:

```
genre_income.tail(100)
```

Out[43]:

	genre	domestic_gross	worldwide_gross	
41	Drama|Mystery and Suspense	936, 662, 225150,201,498 305, 411, 224310,676,7...	2, 053, 311, 220302,469,017 634, 954, 103656,695...	936, 662, 305, 411
1	Action and Adventure|Art House and Internation...	92, 054, 159133,501,348 54, 647, 948 89,296,573$...	259, 357, 408397,501,348 240, 759, 682169,296,5...	92, 054,
37	Drama	89, 302, 115417,719,760 122, 523, 060 42,779,261...	260, 002, 1151,305,772,799 375, 740, 705232,017...	89, 302,
31	Comedy|Kids and Family|Romance	80, 101, 125206,445,654 217, 536, 138 203,464,10...	409, 953, 905626,549,695 767, 820, 459515,419,8...	80, 101,
2	Action and Adventure|Classics|Drama	760, 507, 625293,004,164 111, 506, 43030,212,62...	2, 776, 345, 279731,463,377 269, 806, 43049,628,...	760, 507, 111, 506
3	Action and Adventure|Classics|Drama|Mystery an...	700, 059, 566146,408,305 215, 434, 5913,100,000...	1, 348, 258, 224355,408,305 631, 910, 5313,100,0...	700, 059, 215, 434
42	Drama|Romance	678, 815, 482113,805,681 257, 730, 019289,423,4...	2, 048, 134, 200221,229,335 386, 839, 614559,757...	678, 815, 257, 730
22	Classics|Comedy|Musical and Performing Arts	659, 363, 944127,509,326 180, 202, 163315,544,7...	2, 208, 208, 395329,631,958 518, 858, 449887,210...	659, 363, 180, 202
11	Action and Adventure|Mystery and Suspense	652, 270, 625100,368,560 63, 478, 83866,184,051...	1, 648, 854, 864176,038,324 163, 818, 556172,022...	652, 270, 63, 478,
46	Musical and Performing Arts	623, 279, 547238,736,787 183, 637, 894102,608,8...	1, 517, 935, 897756,244,673 532, 938, 302208,370...	623, 279, 183, 637
39	Drama|Musical and Performing Arts	620, 181, 382152,901,115 160, 942, 139380,270,5...	1, 316, 721, 747383,541,369 431, 942, 139848,998...	620, 181, 160, 942
30	Comedy|Kids and Family	532, 177, 32460,674,817 171, 958, 438 58,183,966...	1, 049, 102, 856181,674,817 341, 528, 51887,683,...	532, 17
32	Comedy|Musical and Performing Arts	486, 295, 56147,225,655 149, 260, 504 228,433,66...	1, 021, 215, 193425,522,281 554, 987, 477655,271...	486, 29
23	Classics|Comedy|Musical and Performing Arts|Ro...	423, 315, 812232,641,920 126, 930, 660100,014,6...	1, 066, 215, 812676,404,566 361, 730, 660302,239...	423, 315, 126, 930
43	Drama|Science Fiction and Fantasy	42, 762, 350201,091,711 68, 671, 925 213,511,408...	149, 762, 350524,283,695 121, 671, 925853,628,6...	42, 762,
10	Action and Adventure|Drama|Western	408, 992, 272104,386,950 138, 540, 870144,801,0...	1, 215, 392, 272418,186,950 273, 271, 982348,629...	408, 992, 138, 540
40	Drama|Musical and Performing Arts|Romance	40, 479, 37073,103,784 101, 028, 233 55,675,313$...	215, 098, 356205,440,387 428, 056, 280139,716,7...	40, 47

	genre	domestic_gross	worldwide_gross	
5	Action and Adventure|Comedy|Mystery and Suspense	389, 813, 10189,760,956 131, 538, 435 140,015,22...	862, 316, 233432,150,894 492, 846, 291298,815,2...	389, 81
6	Action and Adventure|Drama	373, 524, 48559,874,525 118, 311, 368 104,400,89...	795, 110, 670290,930,148 290, 650, 494270,997,3...	373, 52
47	Mystery and Suspense	334, 191, 11034,297,191 101, 200, 044 94,784,201...	1, 025, 491, 110167,297,191 352, 831, 065269,065...	334, 19
35	Documentary	330, 360, 194187,224,605 31, 153, 464186,336,27...	867, 500, 281311,365,187 138, 836, 756486,124,0...	330, 360, 31, 153,
33	Comedy|Mystery and Suspense|Science Fiction an...	315, 058, 289133,298,577 32, 698, 89977,233,467...	846, 980, 024484,161,265 61, 698, 89999,123,656...	315, 058, 32, 698,
38	Drama|Kids and Family	309, 420, 425102,491,776 201, 578, 182132,556,8...	963, 420, 425405,760,225 554, 600, 000416,456,8...	309, 420, 201, 57
29	Comedy|Drama|Romance	304, 360, 277146,880,162 77, 591, 83164,063,008...	1, 110, 526, 981489,592,267 373, 993, 951402,156...	304, 360, 77, 591,
36	Documentary|Special Interest	303, 003, 568155,136,755 403, 706, 375155,190,8...	1, 017, 003, 568401,021,746 821, 706, 375456,258...	303, 003, 403, 70
20	Classics|Comedy|Drama	302, 089, 278150,358,296 107, 509, 799142,614,1...	935, 213, 767433,058,296 186, 976, 250563,749,3...	302, 089, 107, 50
27	Comedy|Drama|Kids and Family|Romance	291, 710, 957131,772,187 260, 044, 82575,030,16...	720, 539, 572319,713,881 345, 141, 403249,488,1...	291, 710, 260, 04
28	Comedy|Drama|Mystery and Suspense	255, 119, 788113,883,318 134, 806, 913130,179,0...	945, 577, 621345,004,422 265, 573, 859295,075,8...	255, 119, 134, 80
26	Comedy|Drama	234, 770, 996319,246,193 49, 554, 002100,234,83...	490, 359, 051708,272,592 215, 080, 810408,803,6...	234, 770, 49, 554,
15	Art House and International|Classics|Horror|My...	234, 362, 462292,576,195 424, 668, 04748,097,08...	459, 260, 946835,524,642 864, 868, 047150,468,0...	234, 362, 424, 66
13	Action and Adventure|Science Fiction and Fantasy	233, 921, 534412,563,408 33, 618, 855101,643,00...	747, 862, 775821,133,378 79, 076, 678232,643,00...	233, 921, 33, 618,
24	Classics|Drama	228, 778, 661195,042,377 191, 204, 754173,005,0...	467, 381, 584688,858,992 485, 004, 754331,323,4...	228, 778, 191, 20
14	Art House and International|Classics|Horror	220, 159, 10445,157,105 93, 050, 117 49,438,370$...	787, 456, 552334,486,852 286, 192, 09169,548,64...	220, 15
12	Action and Adventure|Mystery and Suspense|Scie...	200, 821, 936312,433,331 110, 550, 000324,591,7...	586, 477, 240621,156,389 203, 018, 919786,680,5...	200, 821, 110, 55
4	Action and Adventure|Comedy|Drama	200, 120, 00085,710,210 32, 131, 830 125,304,276...	374, 085, 065402,976,036 85, 131, 830339,504,27...	200, 12

	genre	domestic_gross	worldwide_gross	
25	Comedy	200, 074, 17588,246,220 176, 654, 505 48,003,015...	879, 620, 923264,246,220 370, 569, 776165,149,3...	200, 07
9	Action and Adventure|Drama|Science Fiction and...	172, 558, 87685,576,941 292, 324, 737 113,330,34...	788, 241, 137299,276,941 829, 724, 737215,300,0...	172, 55
34	Comedy|Romance	172, 062, 763292,137,260 137, 748, 063100,478,6...	400, 062, 763943,076,457 356, 148, 063133,401,8...	172, 062, 137, 748
16	Art House and International|Comedy|Drama|Music...	169, 368, 42777,042,381 187, 168, 425 106,580,05...	591, 692, 078276,928,112 573, 068, 425330,780,0...	169, 36
17	Art House and International|Drama	166, 112, 167290,201,752 226, 277, 06897,104,62...	757, 677, 748897,099,794 615, 461, 394212,282,7...	166, 112, 226, 27
21	Classics|Comedy|Drama|Romance	155, 442, 48935,088,320 89, 107, 235 48,417,850$...	542, 537, 546151,525,973 287, 916, 63395,255,48...	155, 44
19	Art House and International|Drama|Mystery and ...	145, 443, 74221,392,758 85, 364, 450 45,216,793$...	529, 530, 71539,549,758 142, 531, 552115,149,42...	145, 44
0	Action and Adventure	144, 840, 419181,030,624 186, 740, 799148,438,6...	351, 040, 419449,326,618 556, 319, 450310,650,5...	144, 840, 186, 740
18	Art House and International|Drama|Musical and ...	140, 125, 96883,670,083 131, 921, 738 59,475,623...	256, 585, 882305,270,083 289, 480, 69196,221,97...	140, 12
8	Action and Adventure|Drama|Mystery and Suspense	130, 168, 683103,144,286 64, 006, 46671,017,784...	602, 893, 340384,169,424 157, 956, 466348,547,5...	130, 168, 64, 006,
7	Action and Adventure|Drama|Horror|Mystery and ...	125, 322, 469256,393,010 381, 193, 157132,024,7...	365, 491, 792585,532,684 1, 341, 693, 157358,824...	125, 322, 381, 19
45	Horror	105, 487, 148281,723,902 249, 757, 72647,398,41...	322, 459, 006648,986,787 796, 907, 323218,853,3...	105, 487, 249, 75
44	Drama|Sports and Fitness	100, 206, 2560 64, 956, 806 40,202,379$150,947,8...	370, 541, 2560 166, 000, 000 82,145,379$276,014,...	40,202

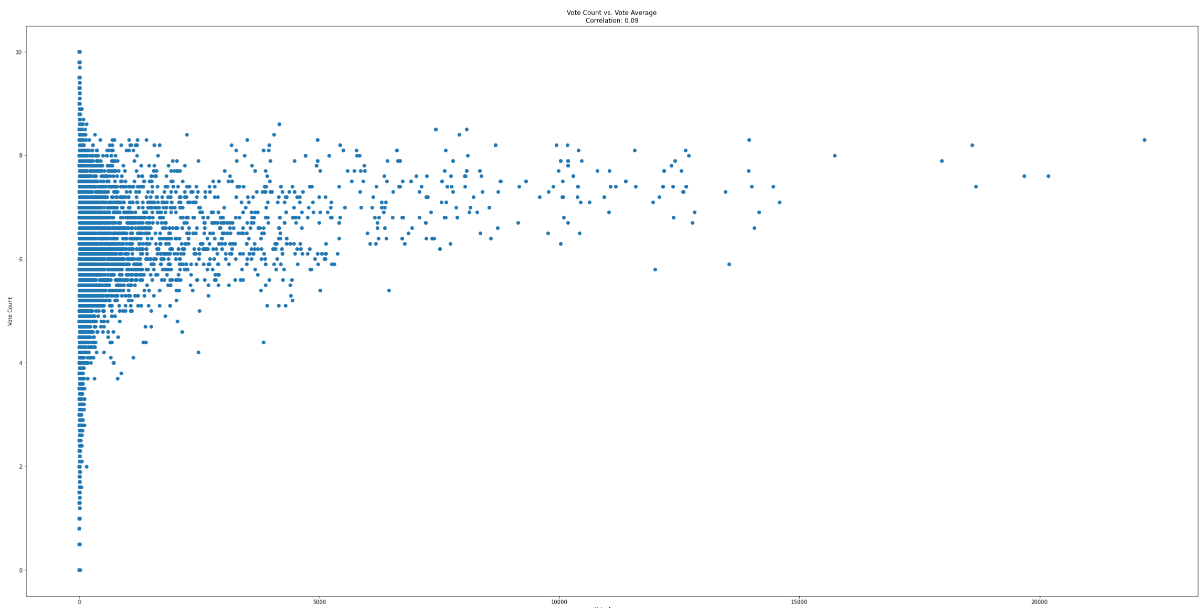
Find the correlation between the vote count and vote average

- We use the 'tmdbmovies. df
- Plot a scatter plot to show the correlation

In [42]:
```python
# calculate correlation between vote_count and vote_average
corr = tmdb_df['vote_count'].corr(tmdb_df['vote_average'])

print('Correlation between vote_count and vote_average:', corr)
```

```
Correlation between vote_count and vote_average: 0.08636988831138752
```

In [43]:
```python
# plot scatter plot
#Plot the scatter plot
fig, ax = plt.subplots(figsize=(40, 20))

#Labeliing the axes
ax.scatter(tmdb_df['vote_count'], tmdb_df['vote_average'])
ax.set_xlabel('Vote Average')
ax.set_ylabel('Vote Count')
ax.set_title('Vote Count vs. Vote Average\nCorrelation: {:.2f}'.format(corr))
plt.show()
```



Skewness and Kurtosis of the 'runtime'

- By examining the skewness and kurtosis of runtime in the movieinfo_df dataset, we can gain insights into how movie runtimes are distributed and how they compare to a normal distribution.
- This can be useful in understanding patterns in movie runtime and identifying any outliers or anomalies in the data. Box plots can also be helpful in visualizing these patterns and identifying any potential outliers.

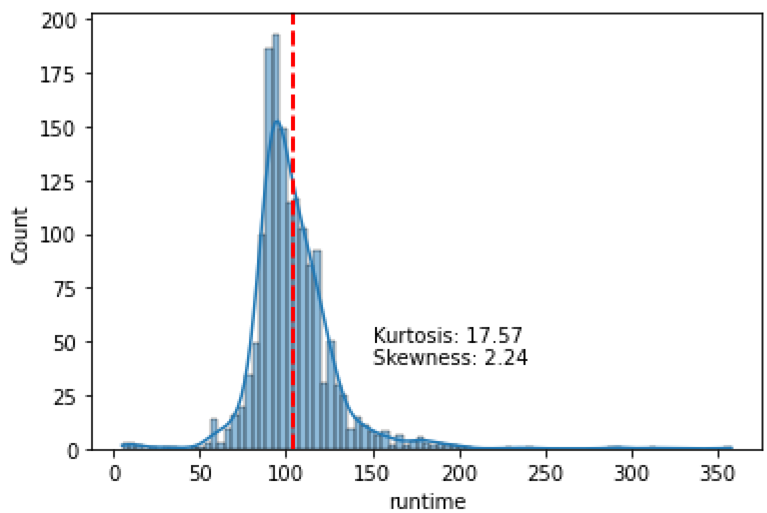
In [44]:
```python
# Read in the movieinfo_df DataFrame
movieinfo_df = pd.read_csv('zippedData/rt.movie_info.tsv', sep='\t')

# Change the 'runtime' column to numeric
movieinfo_df['runtime'] = movieinfo_df['runtime'].str.replace(' minutes', '').astyp
```

```python
# Calculate the skewness and kurtosis of the "runtime" column
runtime_skew = movieinfo_df['runtime'].skew()
runtime_kurtosis = movieinfo_df['runtime'].kurtosis()

print('Runtime skewness:', runtime_skew)
print('Runtime kurtosis:', runtime_kurtosis)
```

```
Runtime skewness: 2.242613327185838
Runtime kurtosis: 17.572857716979282
```

In [45]:

```python
# Create a histogram of the "runtime" column
sns.histplot(movieinfo_df['runtime'], kde=True)

# Add a vertical line at the mean
plt.axvline(movieinfo_df['runtime'].mean(), color='r', linestyle='dashed', linewidt

# Add text labels for the skewness and kurtosis
plt.text(150, 40, "Skewness: {:.2f}".format(movieinfo_df['runtime'].skew()))
plt.text(150, 50, "Kurtosis: {:.2f}".format(movieinfo_df['runtime'].kurtosis()))

# Show the plot
plt.show()
```



Recommendation

Microsoft recommendation for the new movie studio would be:

1) Genre Recommendation

The recommendation were based on the total gross made.

For the single Genre the best recommended are as follow:

- Family
- Adventure
- Fantasy Musical

Multicategory Highly Recommend:

- Action, Adventure, Scifi
- Adventure, Animation, Comedy
- Action, Adventure , Classics and Drama

Genres that also made a huge loss:

- Action, Adventure, Comedy
- Action , Adventure ,Scifi
- Action, Adventure, Drama

2) Target Audience

- The top recommended would be worldwide whcih would be profitable to the company.
- Difference between foreign and worldwide sales was 91502365.03702894. Having movies being released to the domestc audience would incur a significant los..

3) Top Studios to collaborate/ work with:

These are the top studios

- Studio BV
- Universal Pictures(Uni)
- Fox
- Warner Bros(WB)
- Sony
- Studio Paramount(Par)

4)Being a starting film/movie industry

these would be the to releases that can generate a high income:

- Avengers: Infinity War
- Jurassic World
- Black Panther
- Furious 7
- Avengers: Age of Ultron
- Incredibles 2
- Jurassic World: Fallen Kingdom
- Frozen
- Iron Man 3
- Rogue One: A Star Wars Story