RECOMMENDING MICROSOFT MOVIES TO MAKE

Authors:DANIEL WAHOME

Overview

A one-paragraph overview of the project, including the business problem, data, methods, results and recommendations.

Microsoft is planning to create a movie production company and would like to get recommendations on the type of films that the type of films to produce according to the gross trending movies in the box office. Using Python and the provided box office data I will separate the works per studio see what they have dona and what has worked for them the methods i have used are data exploration, dealt with missing values, explored categorical data, identified outliers, performed correlation,and visualized the data

the results I found are that films that people tend to love accordingto gross sales

Boyhood- Coming of Age story

Heartbreaker- comedy drama

Jurassic World Universe -Sci-fi Adventure Supported by a sequel to the first jurassic park boosting sales in the same year

Dark Knight Rises- Action

Harry Potter and the Deathly Hallows Part 2 - Fantasy Adventure also boosted by a prequel the part one of it

Business Problem

Microsoft is planning to create a movie production studio and would like to get recommendations on the type of films that the type of films to produce. They have no clue of where to begin and would like to utilize data from the gross trending movies in the box office to influence the kind of movies they should make

i chose to ask the following question; according to the top studio houses : which film has the highest international grossing

```
                                                    which film has the highest
    domestic grossing

                                                    which genre do these films
    belong to

                                                    what year were these films
    released
```

this will help them come up wih genres that they can start producing on. as they are new to the business and want a large ammount of people to view their films for commercial success, following the mass produced films and genres will put them in the billboards

Data Understanding

Describe the data being used for this project.

the data being used is the box_office data as it will help us to see what the domestic and international viewers like to see, then from there we can emulate

the data consists of the title, studio name, domestic gross ammount, foreign gross ammount and the year the film premiered

i intend to categorise the data accoring to the studios and find out their properties

In [1]:
```
# Import standard packages
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import os
import sqlite3
%matplotlib inline
```

In [2]:
```
#loading all csv files to use for analysis
box_office = pd.read_csv(r'C:\Users\DANIEL\Desktop\PROJECT 1 ASSETS\DATA\bom.movie_gross.csv')
tmdb = pd.read_csv(r'C:\Users\DANIEL\Desktop\PROJECT 1 ASSETS\DATA\tmdb.movies.csv')
the_numbers = pd.read_csv(r'C:\Users\DANIEL\Desktop\PROJECT 1 ASSETS\DATA\tn.movie_budgets.csv'
```

Begin with the Box Office data to understand it clean it and get it ready for manipulation and visualization

In [3]:
```
box_office
```

Out[3]:

	title	studio	domestic_gross	foreign_gross	year
0	Toy Story 3	BV	415000000.0	652000000.0	2010
1	Alice in Wonderland (2010)	BV	334200000.0	691300000.0	2010
2	Harry Potter and the Deathly Hallows Part 1	WB	296000000.0	664300000.0	2010
3	Inception	WB	292600000.0	535700000.0	2010
4	Shrek Forever After	P/DW	238700000.0	513900000.0	2010
...	...	...	...	...	...
3382	The Quake	Magn.	6200.0	NaN	2018
3383	Edward II (2018 re-release)	FM	4800.0	NaN	2018
3384	El Pacto	Sony	2500.0	NaN	2018
3385	The Swan	Synergetic	2400.0	NaN	2018
3386	An Actor Prepares	Grav.	1700.0	NaN	2018

3387 rows × 5 columns

In [4]:
```
#checking out the different data types in Box Office csv file
box_office.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   float64
 4   year            3387 non-null   int64
dtypes: float64(2), int64(1), object(2)
memory usage: 132.4+ KB
```

In [5]:
```
#a brief statistical description of the data
box_office.describe()
```

Out[5]:

	domestic_gross	foreign_gross	year
count	3.359000e+03	2.037000e+03	3387.000000
mean	2.874585e+07	7.503550e+07	2013.958075
std	6.698250e+07	1.373874e+08	2.478141
min	1.000000e+02	6.000000e+02	2010.000000
25%	1.200000e+05	3.700000e+06	2012.000000
50%	1.400000e+06	1.900000e+07	2014.000000
75%	2.790000e+07	7.550000e+07	2016.000000
max	9.367000e+08	9.605000e+08	2018.000000

In [6]:
```
#getting info on any missing data. this way i can identify how to make it uniform and usable
#then i can use it to identify outliers and do the necessary changes according to the findings
box_office.isnull().sum()
```

Out[6]:
```
title                0
studio               5
domestic_gross      28
foreign_gross     1350
year                 0
dtype: int64
```

DROPPING AND FILLING IN MISSING VALUES

In [7]:
```
box_office = box_office.dropna(subset=['studio'])
box_office
#i have chosen to drop the rows in the studio column that were missing
#this is because the data in there cannot be extroplated using mean or mode as it is not an int
#if i physically added new data(random) it wouldnt affect the analysis greatly
#so i chose to frop them
```

Out[7]:

	title	studio	domestic_gross	foreign_gross	year
0	Toy Story 3	BV	415000000.0	652000000.0	2010
1	Alice in Wonderland (2010)	BV	334200000.0	691300000.0	2010
2	Harry Potter and the Deathly Hallows Part 1	WB	296000000.0	664300000.0	2010
3	Inception	WB	292600000.0	535700000.0	2010
4	Shrek Forever After	P/DW	238700000.0	513900000.0	2010
...	...	...	...	...	...
3382	The Quake	Magn.	6200.0	NaN	2018
3383	Edward II (2018 re-release)	FM	4800.0	NaN	2018
3384	El Pacto	Sony	2500.0	NaN	2018
3385	The Swan	Synergetic	2400.0	NaN	2018
3386	An Actor Prepares	Grav.	1700.0	NaN	2018

3382 rows × 5 columns

TREATING MISSING VALUES USING MEDIAN AS THE DATA COULD CONTAIN OUTLIERS MAKING IT SKEWED I ALSO USED MEDIAN AS THE DATA IS NOT DISTRIBUTED

In [8]:
```
box_office['domestic_gross'] = box_office['domestic_gross'].fillna(box_office['domestic_gross']
box_office
box_office['foreign_gross'] = box_office['foreign_gross'].fillna(box_office['foreign_gross'].me
box_office
```

```
<ipython-input-8-e1a4cc905a53>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/i
ndexing.html#returning-a-view-versus-a-copy
  box_office['domestic_gross'] = box_office['domestic_gross'].fillna(box_office['domestic_gros
s'].median())
<ipython-input-8-e1a4cc905a53>:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/i
ndexing.html#returning-a-view-versus-a-copy
  box_office['foreign_gross'] = box_office['foreign_gross'].fillna(box_office['foreign_gros
s'].median())
```

Out[8]:

	title	studio	domestic_gross	foreign_gross	year
0	Toy Story 3	BV	415000000.0	652000000.0	2010
1	Alice in Wonderland (2010)	BV	334200000.0	691300000.0	2010
2	Harry Potter and the Deathly Hallows Part 1	WB	296000000.0	664300000.0	2010
3	Inception	WB	292600000.0	535700000.0	2010
4	Shrek Forever After	P/DW	238700000.0	513900000.0	2010
...	...	...	...	...	...
3382	The Quake	Magn.	6200.0	19000000.0	2018
3383	Edward II (2018 re-release)	FM	4800.0	19000000.0	2018
3384	El Pacto	Sony	2500.0	19000000.0	2018
3385	The Swan	Synergetic	2400.0	19000000.0	2018
3386	An Actor Prepares	Grav.	1700.0	19000000.0	2018

3382 rows × 5 columns

In [9]:
```
box_office.isnull().sum()
```

Out[9]:
```
title             0
studio            0
domestic_gross    0
foreign_gross     0
year              0
dtype: int64
```

In [10]:
```
box_office.duplicated().sum()
```

Out[10]:
```
0
```

In [11]:
```
box_office['studio'].value_counts().head(15)
```

Out[11]:
```
IFC       166
Uni.      147
WB        140
Fox       136
Magn.     136
```

```
SPC        123
Sony       110
BV         106
LGF        103
Par.       101
Eros        89
Wein.       77
CL          74
Strand      68
FoxS        67
Name: studio, dtype: int64
```

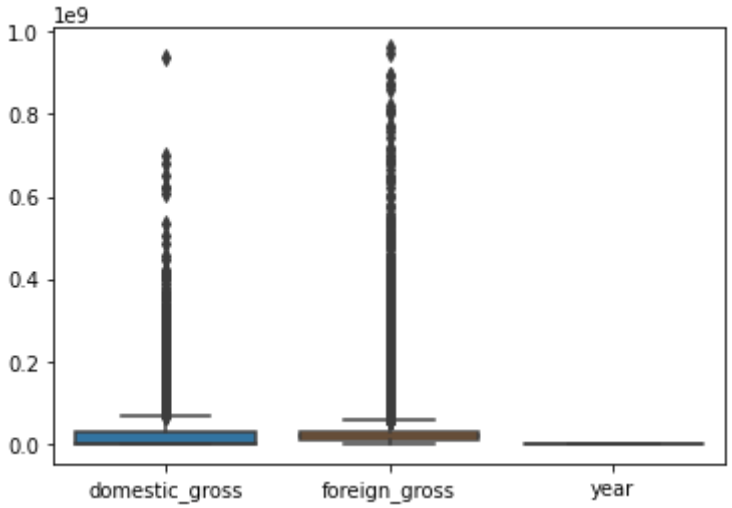
In [12]:
```
box_office['studio'].value_counts().tail(10)
```

Out[12]:
```
CP            1
Gaatri        1
SHO           1
Libre         1
Proud         1
ELS           1
Abk.          1
CineGalaxy    1
Trafalgar     1
Truly         1
Name: studio, dtype: int64
```

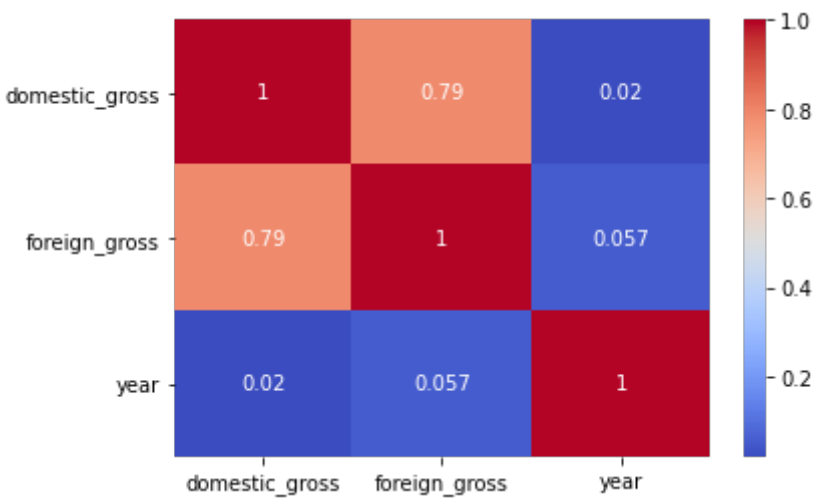
USING SCATTER PLOT TO VISUALIZE OUTLIERS

In [13]:
```
sns.boxplot(data=box_office)
#i decided not to drop the outliers as nothing is out proportional range
```

Out[13]:
```
<AxesSubplot:>
```

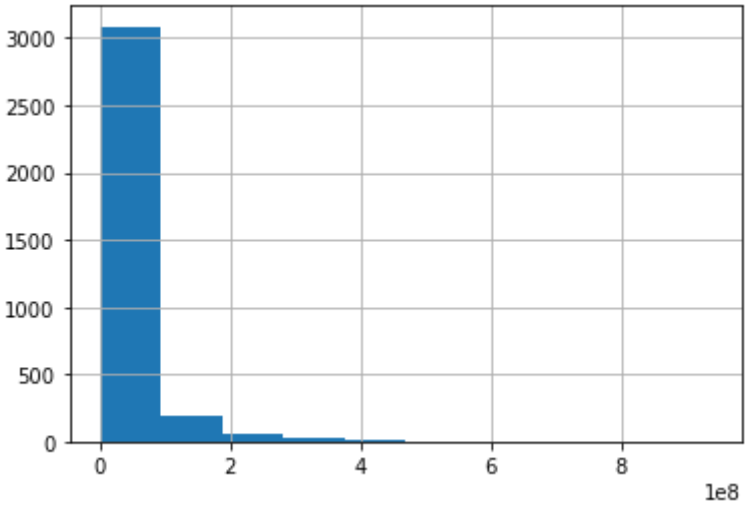


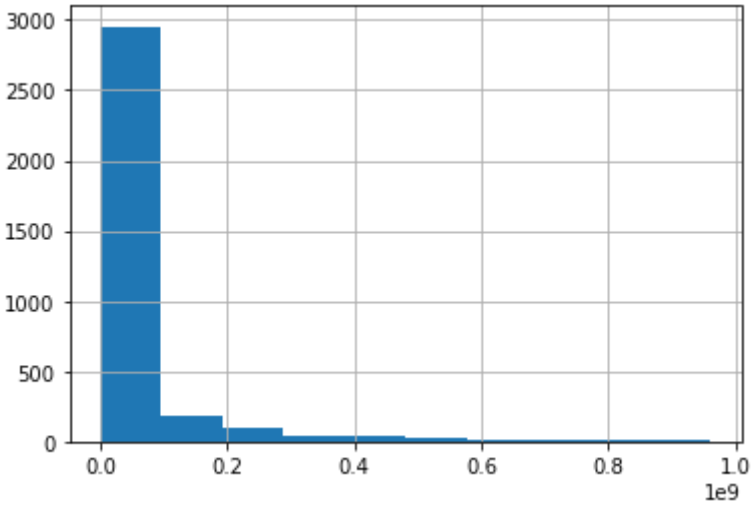
In [14]:
```
correlation_matrix = box_office.corr()

# Visualize correlation matrix
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
plt.show()
```

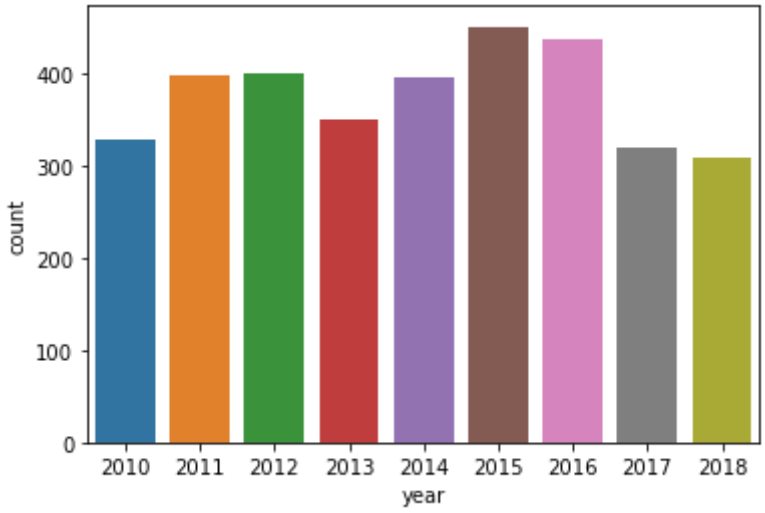


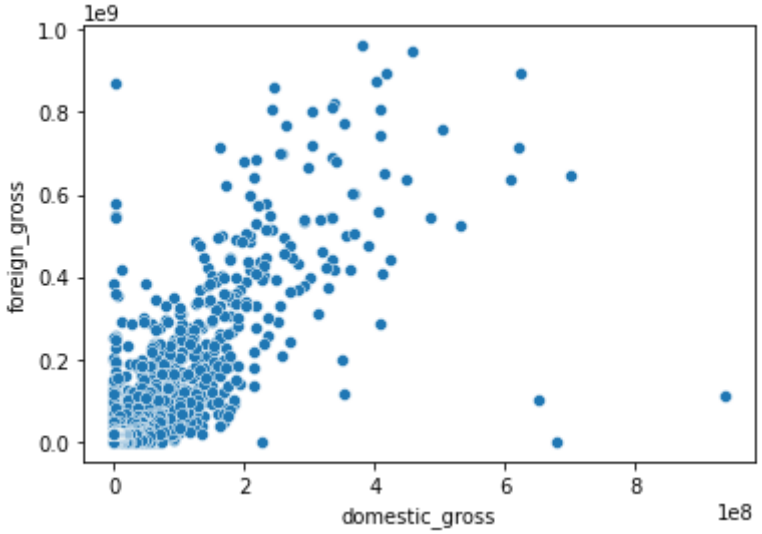
THERE IS A POSITIVE CORRELATION BETWEEN DOMESTIC AND FOREIGN GROSS AMOUNTS OF 0.79 THERE IS LITTLE TO NO CORRELATION BETWEEN YEAR AND

In [15]:
```
box_office['domestic_gross'].hist()
plt.show()
box_office['foreign_gross'].hist()
plt.show()
sns.countplot(x='year', data=box_office)
plt.show()
sns.scatterplot(x='domestic_gross', y= 'foreign_gross', data=box_office)
plt.show()
```









FOR THIS DATA SET: CATEGORIZATION BY STUDIO

1. IFC STUDIO

TITLES BY STUDIO IFC

THERE ARE 166 TITLES BY IFC STUDIO

THE HIGHEST DOMESTIC GROSSING FILM BY THEM IS "BOYHOOD"

GROSSING AT: 25400000.0

PREMIERED IN 2014

ACCORDING TO THE BAR GRAPH 2014 WAS THE YEAR WITHT THE MOST DOMESTIC GROSS INCOME

THE HIGHEST FOREIGN GROSSING FILM IS "HEARTBREAKER"

GROSSING AT: 46900000.0

PREMIERED IN 2010

ACCORDING TO THE BAR GRAPH 2010 WAS THE YEAR WITH THE MOST FOREIGN GROSS INCOME

In [16]:
```
studio1 = box_office.loc[box_office['studio']=='IFC']
studio1
```

Out[16]:

	title	studio	domestic_gross	foreign_gross	year
110	Heartbreaker	IFC	504000.0	46900000.0	2010
112	The Good, the Bad, the Weird	IFC	128000.0	44100000.0	2010
151	Soul Kitchen	IFC	277000.0	17600000.0	2010
166	Looking for Eric	IFC	55800.0	11500000.0	2010
190	Vincere	IFC	619000.0	5100000.0	2010
...	...	...	...	...	...
3324	Ghost Stories	IFC	149000.0	19000000.0	2018
3335	Mary Shelley	IFC	109000.0	19000000.0	2018
3344	The House That Jack Built	IFC	88000.0	19000000.0	2018
3361	A Ciambra	IFC	41900.0	19000000.0	2018
3374	The Escape	IFC	14000.0	19000000.0	2018

166 rows × 5 columns

In [17]:
```
highest_gross = studio1['domestic_gross'].max()
print('the highest domestic grossing film by IFC studio is:', highest_gross)
```

```
the highest domestic grossing film by IFC studio is: 25400000.0
```

In [18]:
```
target = 25400000.0
row = studio1[studio1['domestic_gross']== target]
if not row.empty:
    film = row['title'].values[0]
    print(target, film)
else:
    print('no',target)
```

```
25400000.0 Boyhood
```

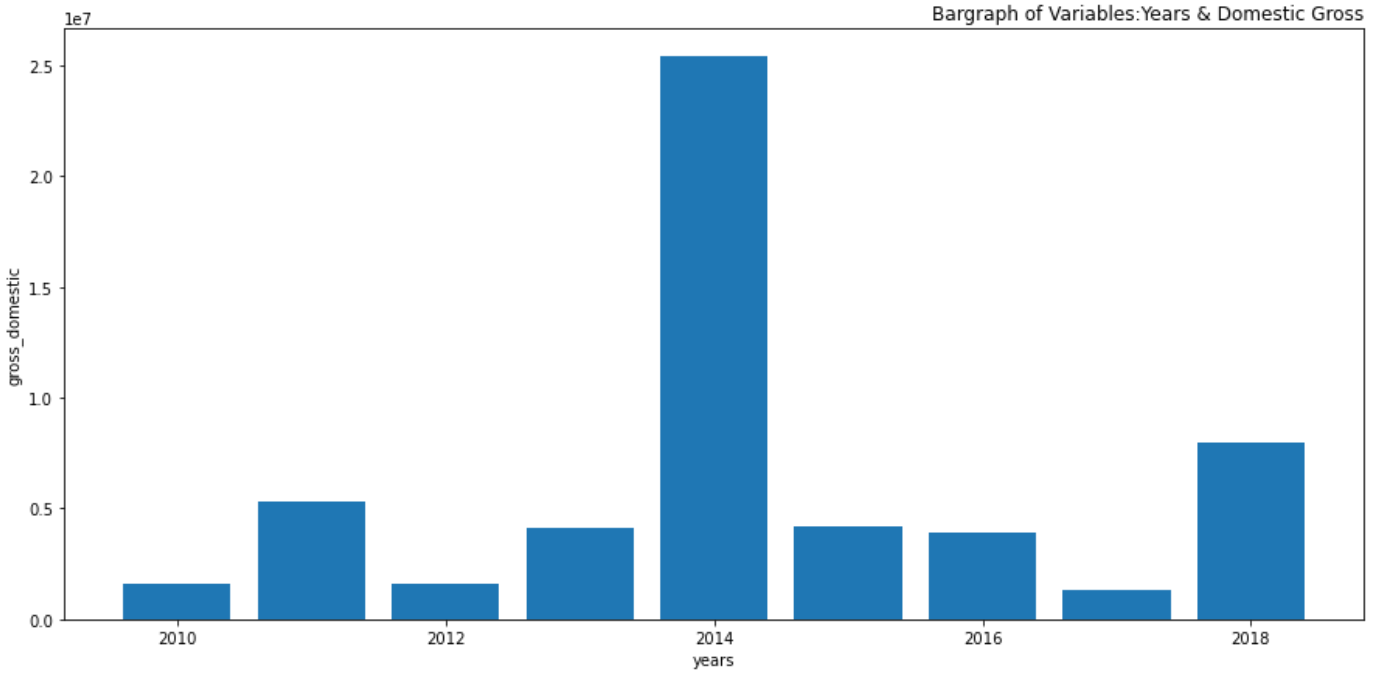
In [19]:
```
target = 'Boyhood'
row = studio1.loc[studio1['title']==target]
if not row.empty:
    year = row['year'].values[0]
    print(target, year)
else:
    print('no')
```

```
Boyhood 2014
```

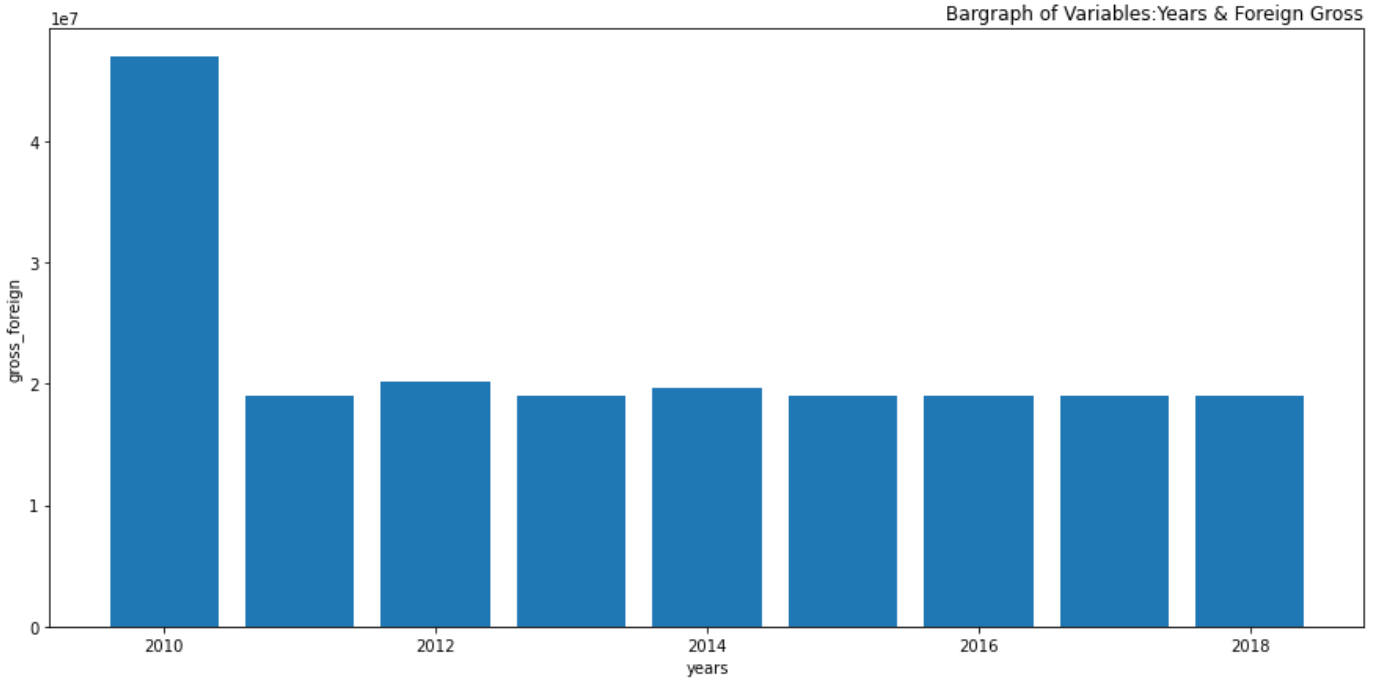
In [20]:
```
type(studio1)
```

Out[20]:
```
pandas.core.frame.DataFrame
```

In [21]:
```
studio1 = pd.DataFrame(studio1)

years = studio1['year']
grossd = studio1['domestic_gross']
grossf = studio1['foreign_gross']

fig = plt.figure(figsize= (15,7))

plt.bar(years, grossd)
plt.xlabel ('years')
plt.ylabel('gross_domestic')
plt.title('Bargraph of Variables:Years & Domestic Gross', loc = "right")

plt.show()
```



In [22]:
```
studio1 = pd.DataFrame(studio1)

years = studio1['year']
grossd = studio1['domestic_gross']
grossf = studio1['foreign_gross']

fig = plt.figure(figsize= (15,7))

plt.bar(years, grossf)
plt.xlabel ('years')
plt.ylabel('gross_foreign')
plt.title('Bargraph of Variables:Years & Foreign Gross', loc = "right")

plt.show()
```



In [23]:
```
highest_gross = studio1['foreign_gross'].max()
print('the highest domestic grossing film by IFC studio is:', highest_gross)
```

```
the highest domestic grossing film by IFC studio is: 46900000.0
```

In [24]:
```
target = 46900000.0
row = studio1[studio1['foreign_gross']== target]
if not row.empty:
    film = row['title'].values[0]
    print(target, film)
else:
    print('no',target)
```

```
46900000.0 Heartbreaker
```

In [25]:
```
target = 'Heartbreaker'
row = studio1.loc[studio1['title']==target]
if not row.empty:
    year = row['year'].values[0]
    print(target, year)
else:
    print('no')
```

```
Heartbreaker 2010
```

2. UNIVERSAL STUDIOS

STUDIO UNIVERSAL

THERE ARE TITLES BY UNIVERSAL STUDIOS

THE HIGHEST DOMESTIC GROSSING FILM BY THEM IS "JURASSIC WORLD"

GROSSING AT: 652300000.0

PREMIERED IN 2015

ACCORDING TO THE BAR GRAPH 2015 WAS THE YEAR WITH THE MOST DOMESTIC GROSS INCOME

THE HIGHEST FOREIGN GROSSING FILM IS "JURASSIC WORLD: FALLEN KINGDOM"

GROSSING AT: 891800000.0

PREMIERED IN 2018

ACCORDING TO THE BAR GRAPH 2018 WAS THE YEAR WITH THE MOST FOREIGN GROSS INCOME

In [26]:
```
studio2 = box_office.loc[box_office['studio']=='Uni.']
studio2
```

Out[26]:

	title	studio	domestic_gross	foreign_gross	year
8	Despicable Me	Uni.	251500000.0	291600000.0	2010
18	Robin Hood	Uni.	105300000.0	216400000.0	2010
20	Little Fockers	Uni.	148400000.0	162200000.0	2010
49	The Wolfman	Uni.	62000000.0	77800000.0	2010
66	Green Zone	Uni.	35100000.0	59800000.0	2010
...	...	...	...	...	...
3148	Mortal Engines	Uni.	16000000.0	67700000.0	2018
3172	Breaking In (2018)	Uni.	46800000.0	4600000.0	2018
3219	Welcome to Marwen	Uni.	10800000.0	2100000.0	2018

	title	studio	domestic_gross	foreign_gross	year
3289	Schindler's List (2018 re-release)	Uni.	833000.0	19000000.0	2018
3369	Loving Pablo	Uni.	22000.0	19000000.0	2018

147 rows × 5 columns

In [27]:
```
highest_gross = studio2['domestic_gross'].max()
print('the highest domestic grossing film by Universal studio is:', highest_gross)
```

```
the highest domestic grossing film by Universal studio is: 652300000.0
```

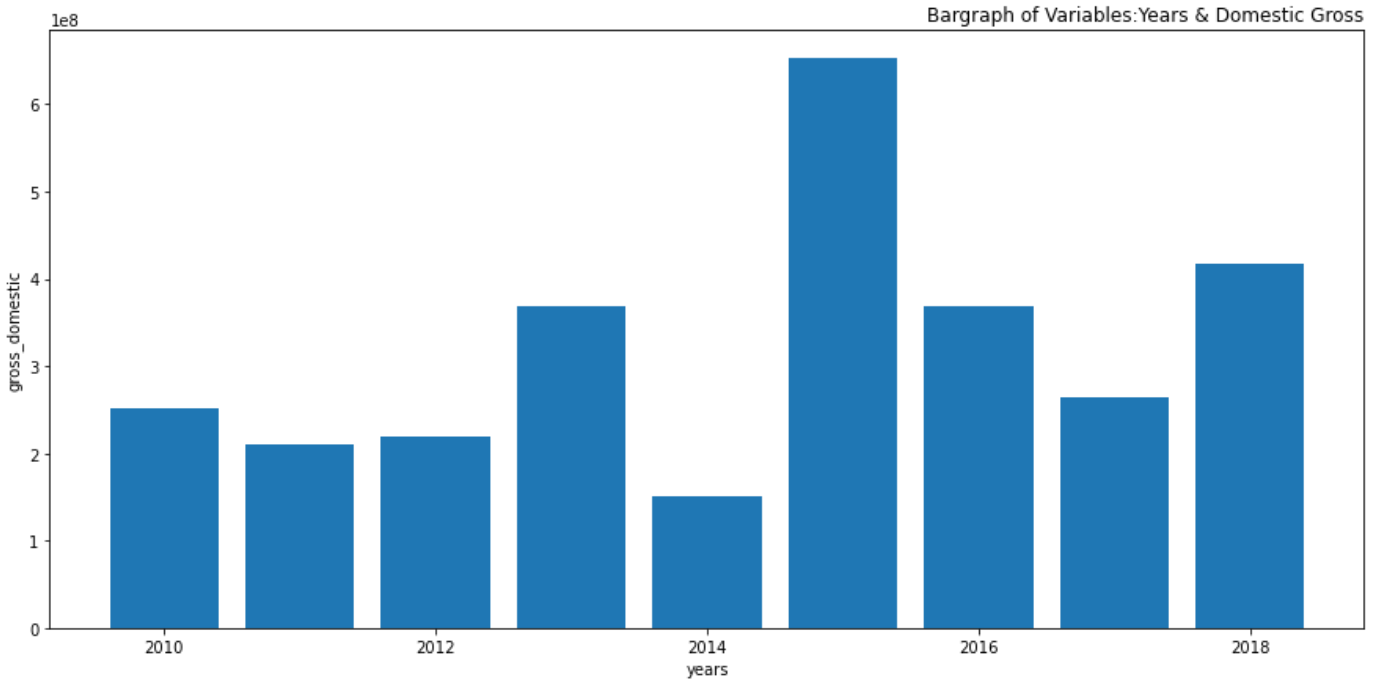
In [28]:
```
target = 652300000.0
row = studio2[studio2['domestic_gross']== target]
if not row.empty:
    film = row['title'].values[0]
    print(target, film)
else:
    print('no',target)
```

```
652300000.0 Jurassic World
```

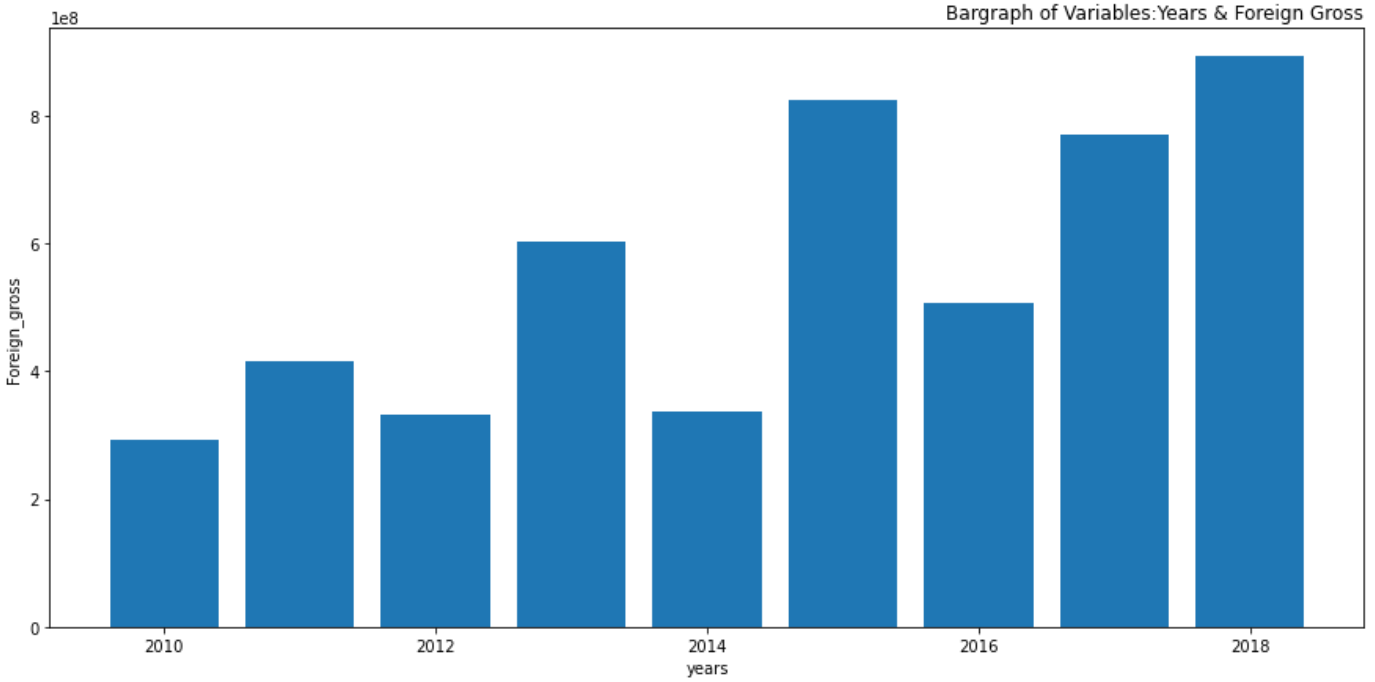
In [29]:
```
target = 'Jurassic World'
row = studio2.loc[studio2['title']==target]
if not row.empty:
    year = row['year'].values[0]
    print(target, year)
else:
    print('no')
```

```
Jurassic World 2015
```

In [30]:
```
studio2 = pd.DataFrame(studio2)

years = studio2['year']
grossd = studio2['domestic_gross']
grossf = studio2['foreign_gross']

fig = plt.figure(figsize= (15,7))

plt.bar(years, grossd)
plt.xlabel ('years')
plt.ylabel('gross_domestic')
plt.title('Bargraph of Variables:Years & Domestic Gross', loc = "right")

plt.show()
```



In [31]:
```
studio2 = pd.DataFrame(studio2)

years = studio2['year']
grossd = studio2['domestic_gross']
grossf = studio2['foreign_gross']

fig = plt.figure(figsize= (15,7))

plt.bar(years, grossf)
plt.xlabel ('years')
plt.ylabel('Foreign_gross')
plt.title('Bargraph of Variables:Years & Foreign Gross', loc = "right")

plt.show()
```



In [32]:
```
highest_gross = studio2['foreign_gross'].max()
print('the highest domestic grossing film by Universal studio is:', highest_gross)
```

```
the highest domestic grossing film by Universal studio is: 891800000.0
```

In [33]:
```
target = 891800000
row = studio2[studio2['foreign_gross']== target]
if not row.empty:
    film = row['title'].values[0]
    print(target, film)
else:
    print('no',target)
```

```
891800000 Jurassic World: Fallen Kingdom
```

In [34]:
```
target = 'Jurassic World: Fallen Kingdom'
row = studio2.loc[studio2['title']==target]
if not row.empty:
    year = row['year'].values[0]
    print(target, year)
else:
    print('no')
```

```
Jurassic World: Fallen Kingdom 2018
```

3. WARNER BROS

WARNER BROTHERS

THE HIGHEST DOMESTIC GROSSING FILM BY THEM IS "DARK NIGHT RISES"

GROSSING AT: 448100000.0

PREMIERED IN 2012

ACCORDING TO THE BAR GRAPH 2012 WAS THE YEAR WITH THE MOST DOMESTIC GROSS INCOME

THE HIGHEST FOREIGN GROSSING FILM IS " Harry Potter and the Deathly Hallows Part 2"

GROSSING AT: 960500000.0

PREMIERED IN 2011

ACCORDING TO THE BAR GRAPH 2011 WAS THE YEAR WITH THE MOST FOREIGN GROSS INCOME

In [35]:
```
studio3 = box_office.loc[box_office['studio']=='WB']
studio3
```

Out[35]:

	title	studio	domestic_gross	foreign_gross	year
2	Harry Potter and the Deathly Hallows Part 1	WB	296000000.0	664300000.0	2010
3	Inception	WB	292600000.0	535700000.0	2010
10	Clash of the Titans (2010)	WB	163200000.0	330000000.0	2010
35	Due Date	WB	100500000.0	111200000.0	2010
37	Yogi Bear	WB	100200000.0	101300000.0	2010
...	...	...	...	...	...
3161	12 Strong	WB	45800000.0	21600000.0	2018
3167	The 15:17 to Paris	WB	36300000.0	20800000.0	2018
3170	Teen Titans Go! To The Movies	WB	29800000.0	22300000.0	2018
3209	They Shall Not Grow Old	WB	18000000.0	19000000.0	2018
3264	2001: A Space Odyssey (2018 re-release)	WB	3200000.0	19000000.0	2018

140 rows × 5 columns

In [36]:
```
highest_gross = studio3['domestic_gross'].max()
print('the highest domestic grossing film by WarnerBros is:', highest_gross)
```

```
the highest domestic grossing film by WarnerBros is: 448100000.0
```

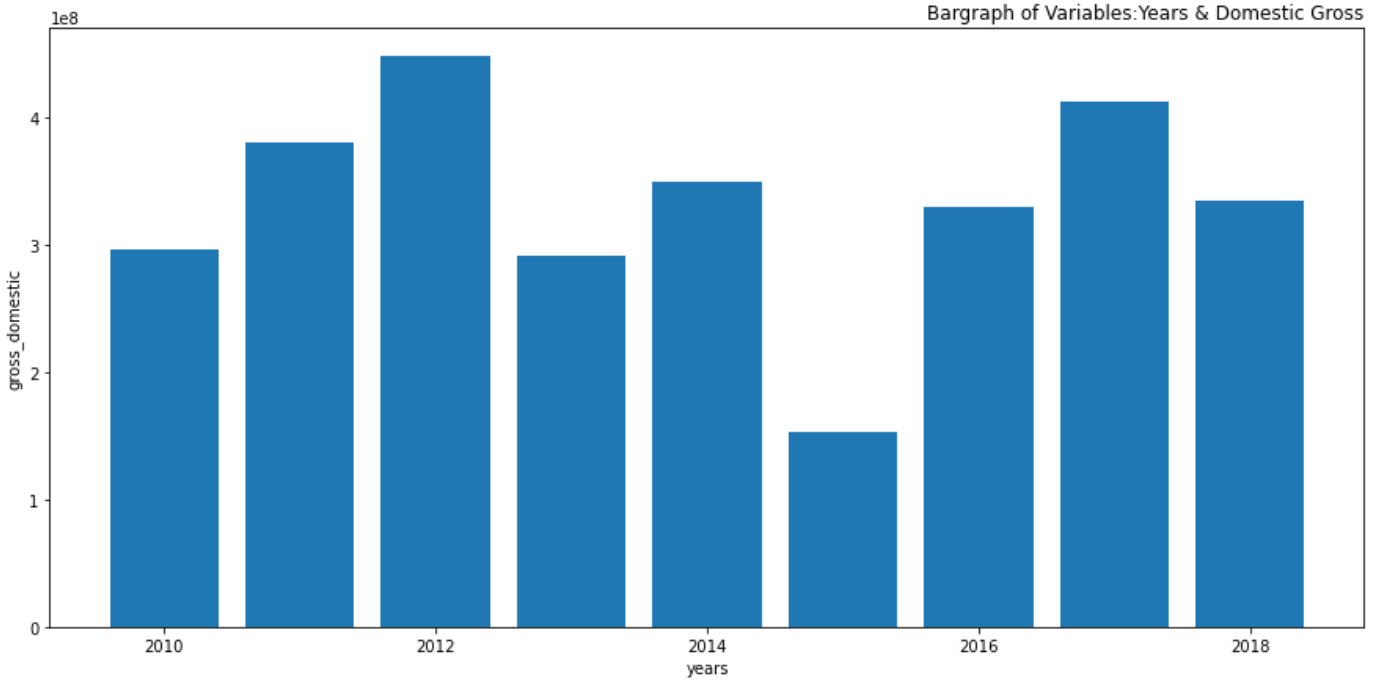
In [37]:
```
target = 448100000.0
row = studio3[studio3['domestic_gross']== target]
if not row.empty:
    film = row['title'].values[0]
    print(target, film)
else:
    print('no',target)
```

```
448100000.0 The Dark Knight Rises
```

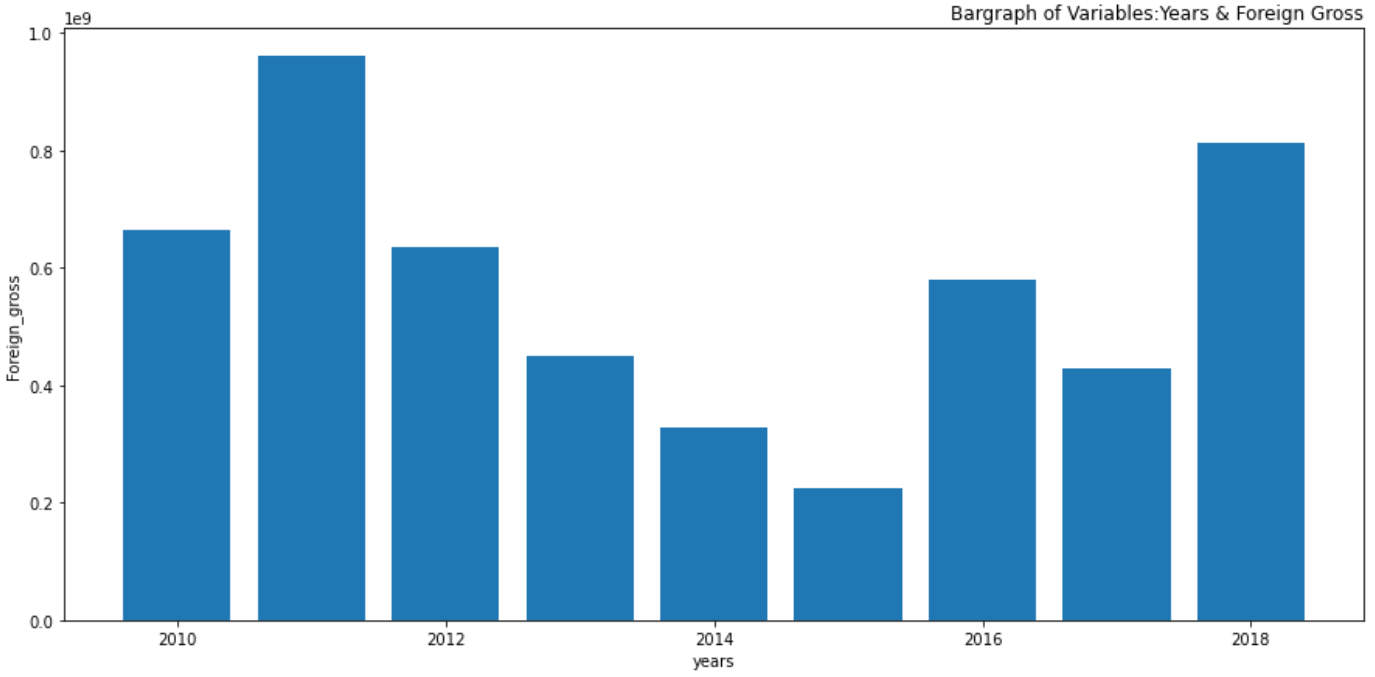
In [38]:
```
target = 'The Dark Knight Rises'
row = studio3.loc[studio3['title']==target]
if not row.empty:
    year = row['year'].values[0]
    print(target, year)
else:
    print('no')
```

```
The Dark Knight Rises 2012
```

In [39]:
```
studio3 = pd.DataFrame(studio3)

years = studio3['year']
grossd = studio3['domestic_gross']
grossf = studio3['foreign_gross']

fig = plt.figure(figsize= (15,7))

plt.bar(years, grossd)
plt.xlabel ('years')
plt.ylabel('gross_domestic')
plt.title('Bargraph of Variables:Years & Domestic Gross', loc = "right")

plt.show()
```



In [40]:
```
studio3 = pd.DataFrame(studio3)

years = studio3['year']
grossd = studio3['domestic_gross']
grossf = studio3['foreign_gross']

fig = plt.figure(figsize= (15,7))

plt.bar(years, grossf)
plt.xlabel ('years')
plt.ylabel('Foreign_gross')
plt.title('Bargraph of Variables:Years & Foreign Gross', loc = "right")

plt.show()
```



In [41]:
```
highest_gross = studio3['foreign_gross'].max()
print('the highest domestic grossing film by WarnerBros is:', highest_gross)
```

```
the highest domestic grossing film by WarnerBros is: 960500000.0
```

In [42]:
```
target = 960500000.0
row = studio3[studio3['foreign_gross']== target]
if not row.empty:
    film = row['title'].values[0]
    print(target, film)
else:
    print('no',target)
```

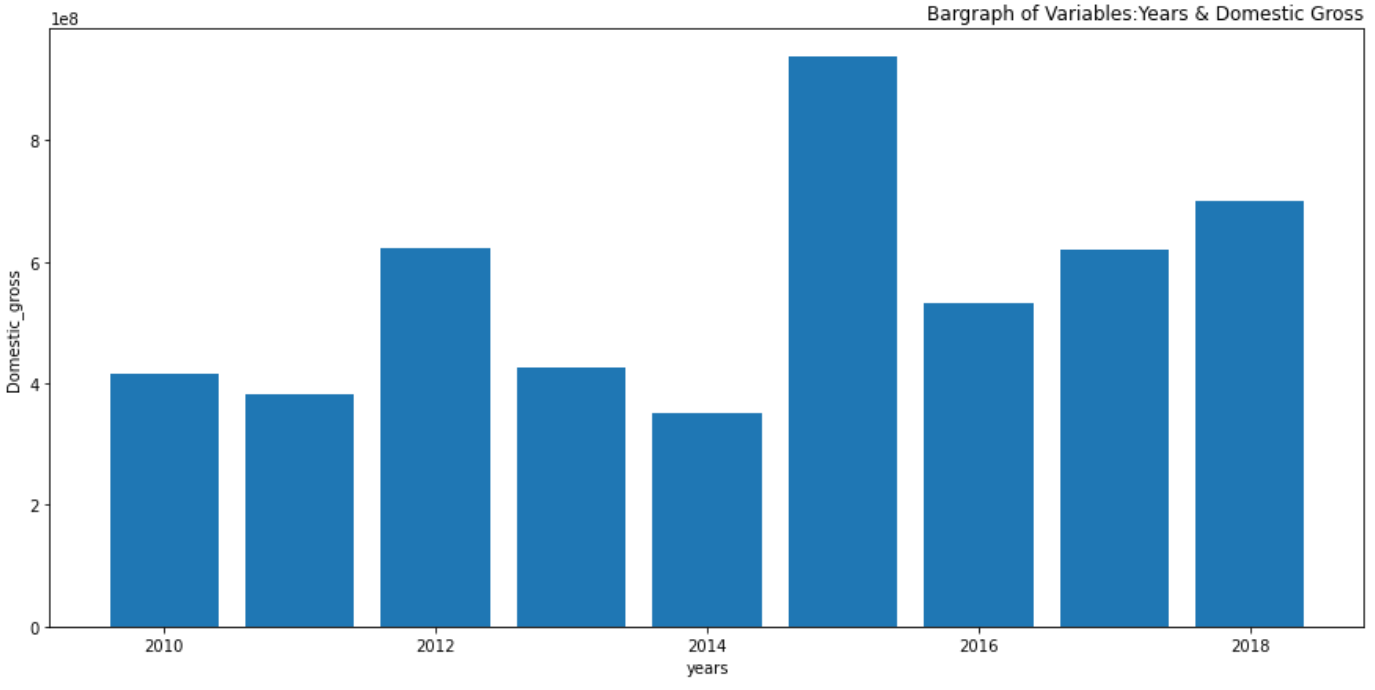
```
960500000.0 Harry Potter and the Deathly Hallows Part 2
```

In [43]:
```
target = 'Harry Potter and the Deathly Hallows Part 2'
row = studio3.loc[studio3['title']==target]
if not row.empty:
    year = row['year'].values[0]
    print(target, year)
else:
    print('no')
```

```
Harry Potter and the Deathly Hallows Part 2 2011
```

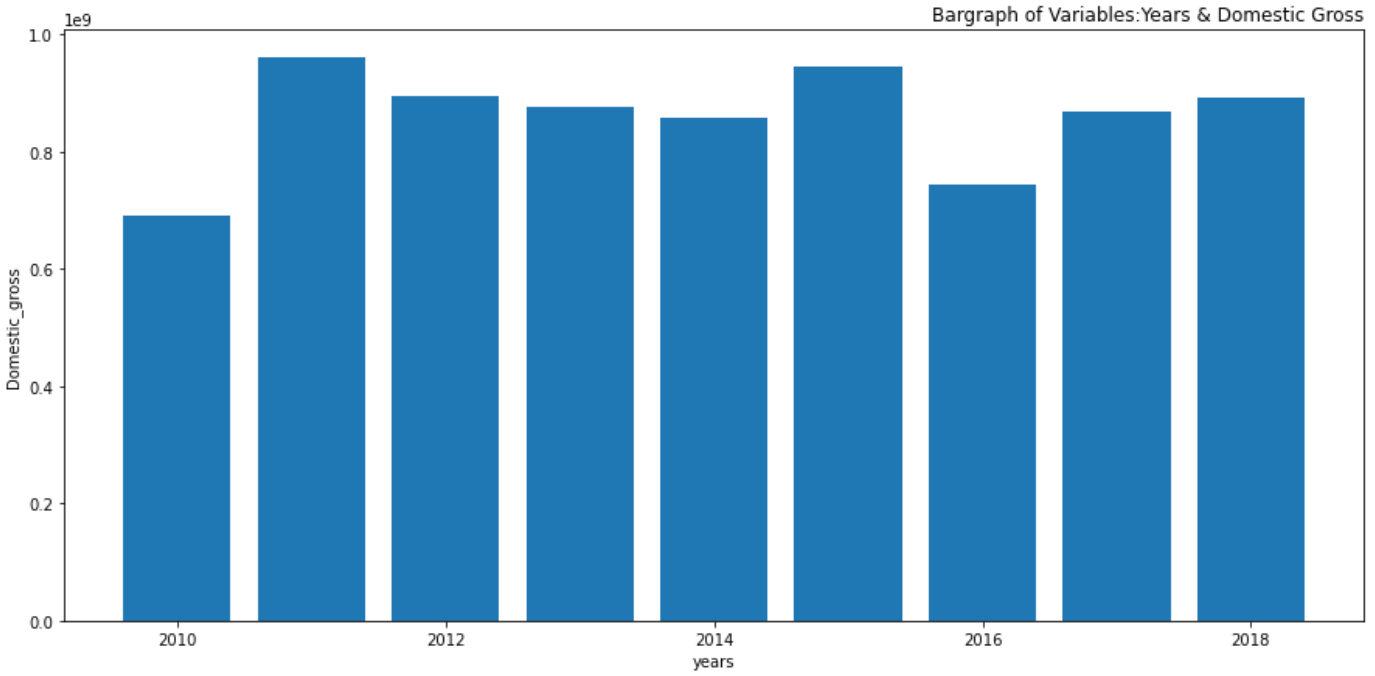
OVERVIEW PLOT OF BOX OFFICE DATA FRAME

In [44]:
```
#plotting the domestic gross over years using plt.bar

b = pd.DataFrame(box_office)

year= box_office['year']
dgross = box_office['domestic_gross']
fgross = box_office['foreign_gross']

fig = plt.figure(figsize=(15,7))

plt.bar(year, dgross)
plt.xlabel ('years')
plt.ylabel('Domestic_gross')
plt.title('Bargraph of Variables:Years & Domestic Gross', loc = "right")

plt.show()
```



overall in 2015 experienced the highest domestic grossing

In [45]:
```
c = pd.DataFrame(box_office)

year= box_office['year']
dgross = box_office['domestic_gross']
fgross = box_office['foreign_gross']

fig = plt.figure(figsize=(15,7))

plt.bar(year, fgross)
plt.xlabel ('years')
plt.ylabel('Domestic_gross')
plt.title('Bargraph of Variables:Years & Domestic Gross', loc = "right")

plt.show()
```



overall in 2011 and 2015 experienced the highest domestic grossing

CONCLUSION

domestic gross and foreign gross are postively correlated at 0.79

i analysed the top 3 studios: IFC, WARNER BROS AND UNIVERSAL STUDIOS

WARNER BROS: "The Dark Knight Rises" was the highest domestic grossing film in 2012, contributing to the highest domestic gross income that year. On the other hand, "Harry Potter and the Deathly Hallows Part 2" was the highest foreign grossing film in 2011, contributing to the highest foreign gross income in that year.

UNIVERSAL STUDIOS "Jurassic World" and "Jurassic World: Fallen Kingdom" were the highest domestic and foreign grossing films, respectively, for Universal Studios. These films contributed to the years 2015 and 2018 being the years with the highest domestic and foreign gross income, according to the provided bar graph.

IFC STUDIOS "Boyhood" and "Heartbreaker" were the highest domestic and foreign grossing films, respectively, for IFC Studio. These films contributed to 2014 being the year with the highest domestic gross income and 2010 being the year with the highest foreign gross income, according to the provided bar graph

if time permitted i would comment better and use the budgets data to come up with profit margins that these films have and use it to analyse our expected commercial success

In [46]:
```
the_numbers.head()
#simple data vis
```

Out[46]:

	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 [47]:
```
the_numbers.info()
#brief overview of data
```

```
<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 [48]:
```
#transforming the object type to integer type
the_numbers['production_budget'] = pd.to_numeric(the_numbers['production_budget'].str.replace('
the_numbers['domestic_gross'] = pd.to_numeric(the_numbers['domestic_gross'].str.replace('[\$,]'
the_numbers['worldwide_gross'] = pd.to_numeric(the_numbers['worldwide_gross'].str.replace('[\$,
```

In [49]:
```
the_numbers.info()
#confirming changes
```

```
<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   int64 
 4   domestic_gross     5782 non-null   int64 
 5   worldwide_gross    5782 non-null   int64 
dtypes: int64(4), object(2)
memory usage: 271.2+ KB
```

In [50]:
```
the_numbers['production_budget'] = the_numbers['production_budget'].astype(float)
the_numbers['domestic_gross'] = the_numbers['domestic_gross'].astype(float)
the_numbers['worldwide_gross'] = the_numbers['worldwide_gross'].astype(float)
#changing numeric to float type
```

In [51]:
```
the_numbers.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   float64
 4   domestic_gross     5782 non-null   float64
 5   worldwide_gross    5782 non-null   float64
dtypes: float64(3), int64(1), object(2)
```

```
memory usage: 271.2+ KB
```

In [52]:
```
the_numbers.describe()
```

Out[52]:

	id	production_budget	domestic_gross	worldwide_gross
count	5782.000000	5.782000e+03	5.782000e+03	5.782000e+03
mean	50.372363	3.158776e+07	4.187333e+07	9.148746e+07
std	28.821076	4.181208e+07	6.824060e+07	1.747200e+08
min	1.000000	1.100000e+03	0.000000e+00	0.000000e+00
25%	25.000000	5.000000e+06	1.429534e+06	4.125415e+06
50%	50.000000	1.700000e+07	1.722594e+07	2.798445e+07
75%	75.000000	4.000000e+07	5.234866e+07	9.764584e+07
max	100.000000	4.250000e+08	9.366622e+08	2.776345e+09

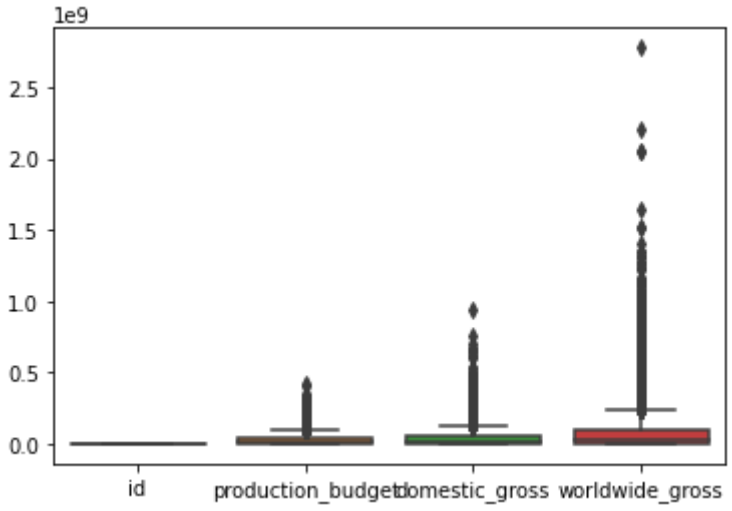
In [53]:
```
the_numbers.isnull().sum()

#the_numbers.duplicated().sum()
```

Out[53]:
```
id                 0
release_date       0
movie              0
production_budget  0
domestic_gross     0
worldwide_gross    0
dtype: int64
```

In [54]:
```
pd.set_option('display.float_format', lambda x: '%.3f' % x)
#this removes the scientific notation of putting numbers in powers of ten for readability purpo
```

In [55]:
```
sns.boxplot(data=the_numbers)
#the worldwide gross is definetly higher than domestic gross
```

Out[55]:
```
<AxesSubplot:>
```



In [56]:
```
the_numbers['domestic_gross'].isnull().mean()
#there are no missing values to affect non null mean
```

Out[56]:
```
0.0
```

DEALING WITH OUTLIERS

In [57]:
```
# Calculate the mean of non-null values in the 'domestic_gross' column
mean_domestic_gross = the_numbers['domestic_gross'][the_numbers['domestic_gross'] != 0].mean()

# Replace zeros with the mean
the_numbers['domestic_gross'] = the_numbers['domestic_gross'].replace(0, mean_domestic_gross)
#why for some reason i was geting negative infinity in my data, so i replaced zeros with their
```

In [58]:
```
# Calculate the mean of non-null values in the 'worldwide_gross' column
mean_worldwide_gross = the_numbers['worldwide_gross'][the_numbers['worldwide_gross'] != 0].mean

# Replace zeros with the mean
the_numbers['worldwide_gross'] = the_numbers['worldwide_gross'].replace(0, mean_worldwide_gross
```

In [59]:
```
the_numbers
```

Out[59]:

	id	release_date	movie	production_budget	domestic_gross	worldwide_gross
0	1	Dec 18, 2009	Avatar	425000000.000	760507625.000	2776345279.000
1	2	May 20, 2011	Pirates of the Caribbean: On Stranger Tides	410600000.000	241063875.000	1045663875.000
2	3	Jun 7, 2019	Dark Phoenix	350000000.000	42762350.000	149762350.000
3	4	May 1, 2015	Avengers: Age of Ultron	330600000.000	459005868.000	1403013963.000
4	5	Dec 15, 2017	Star Wars Ep. VIII: The Last Jedi	317000000.000	620181382.000	1316721747.000
...	...	...	...	...	...	...
5777	78	Dec 31, 2018	Red 11	7000.000	46257465.790	97687996.115
5778	79	Apr 2, 1999	Following	6000.000	48482.000	240495.000
5779	80	Jul 13, 2005	Return to the Land of Wonders	5000.000	1338.000	1338.000
5780	81	Sep 29, 2015	A Plague So Pleasant	1400.000	46257465.790	97687996.115
5781	82	Aug 5, 2005	My Date With Drew	1100.000	181041.000	181041.000

5782 rows × 6 columns

there are no null values so thers's nothing to be removed or filled

PROFIT MARGINS

CREATING TABLE DOMESTIC PROFIT MARGIN (D_PROFIT_MARGIN_PERCENTAGE)

CREATING TABLE WORLDWIDE PROFIT MARGIN (W_PROFIT_MARGIN_PERCENTAGE)

THESE COLUMNS WILL SHOW US WHICH FILM HAS THE HIGHEST PROFIT MARGIN PERCENTAGE: DOMESTIC AND WORLDWIDE

OPTIMAL = 40

In [60]:
```
profit_d = the_numbers['domestic_gross'] - the_numbers['production_budget']
d_profit_margin_percentage = (profit_d / the_numbers['domestic_gross']) * 100
the_numbers = the_numbers.assign(d_profit_margin_percentage=d_profit_margin_percentage)

profit_w = the_numbers['worldwide_gross'] - the_numbers['production_budget']
w_profit_margin_percentage = (profit_w / the_numbers['worldwide_gross']) * 100
the_numbers = the_numbers.assign(w_profit_margin_percentage=w_profit_margin_percentage)
```

In [61]:
```
the_numbers
```

Out[61]:

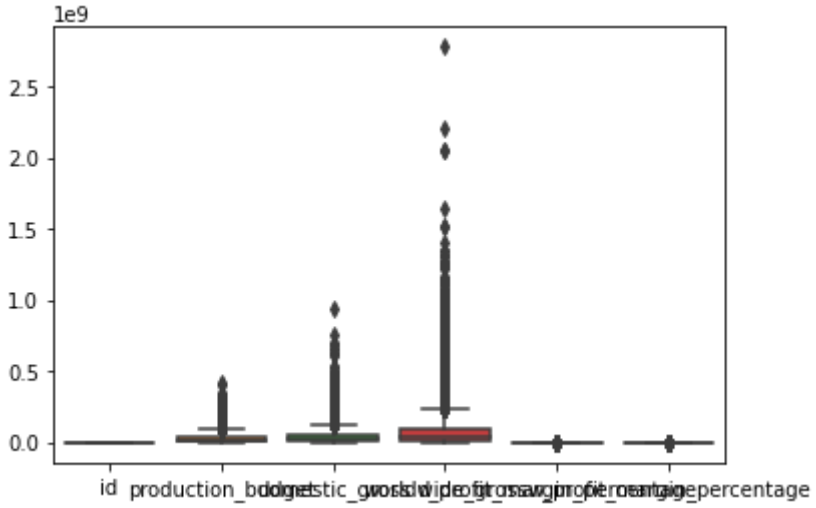
	id	release_date	movie	production_budget	domestic_gross	worldwide_gross	d_profit_margin_percenta

	id	release_date	movie	production_budget	domestic_gross	worldwide_gross	d_profit_margin_percenta
0	1	Dec 18, 2009	Avatar	425000000.000	760507625.000	2776345279.000	44.1
1	2	May 20, 2011	Pirates of the Caribbean: On Stranger Tides	410600000.000	241063875.000	1045663875.000	-70.3
2	3	Jun 7, 2019	Dark Phoenix	350000000.000	42762350.000	149762350.000	-718.4
3	4	May 1, 2015	Avengers: Age of Ultron	330600000.000	459005868.000	1403013963.000	27.9
4	5	Dec 15, 2017	Star Wars Ep. VIII: The Last Jedi	317000000.000	620181382.000	1316721747.000	48.8
...	...	...	...	...	...	...	
5777	78	Dec 31, 2018	Red 11	7000.000	46257465.790	97687996.115	99.9
5778	79	Apr 2, 1999	Following	6000.000	48482.000	240495.000	87.6
5779	80	Jul 13, 2005	Return to the Land of Wonders	5000.000	1338.000	1338.000	-273.6
5780	81	Sep 29, 2015	A Plague So Pleasant	1400.000	46257465.790	97687996.115	99.9
5781	82	Aug 5, 2005	My Date With Drew	1100.000	181041.000	181041.000	99.3

5782 rows × 8 columns

In [62]:
```
sns.boxplot(data=the_numbers)
```

Out[62]:
```
<AxesSubplot:>
```



In [63]:
```
the_numbers.describe()
```

Out[63]:

	id	production_budget	domestic_gross	worldwide_gross	d_profit_margin_percentage	w_profit_margi
count	5782.000	5782.000	5782.000	5782.000	5782.000	
mean	50.372	31587757.097	46257465.790	97687996.115	-3733.481	

	id	production_budget	domestic_gross	worldwide_gross	d_profit_margin_percentage	w_profit_margi
std	28.821	41812076.827	66881753.042	173088700.211	64079.646	
min	1.000	1100.000	388.000	26.000	-3267873.856	
25%	25.000	5000000.000	5609102.250	8210838.250	-120.723	
50%	50.000	17000000.000	27688002.000	38637682.000	16.375	
75%	75.000	40000000.000	52348661.500	97687996.115	65.470	
max	100.000	425000000.000	936662225.000	2776345279.000	99.997	

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

Out[64]:
```
id                          0
release_date                0
movie                       0
production_budget           0
domestic_gross              0
worldwide_gross             0
d_profit_margin_percentage  0
w_profit_margin_percentage  0
dtype: int64
```

In [65]:
```
the_numbers['d_profit_margin_percentage'].describe()
```

Out[65]:
```
count       5782.000
mean       -3733.481
std        64079.646
min     -3267873.856
25%         -120.723
50%           16.375
75%           65.470
max           99.997
Name: d_profit_margin_percentage, dtype: float64
```

THIS SHOWS THAT THERE IS A NEGATIVE MEAN OF THE DOMESTIC PROFIT MARGIN (-3733)

THIS MEANS THAT THE STUDIOS HAVE A HIGH PROBABILITY OF MAKING LOSES DOMESTICALLY LETS SEE HOW THAT FAIRS WORLDWIDE

In [66]:
```
the_numbers['w_profit_margin_percentage'].describe()
```

Out[66]:
```
count       5782.000
mean       -2317.354
std        54679.452
min     -3846053.846
25%          -29.252
50%           52.118
75%           78.911
max           99.999
Name: w_profit_margin_percentage, dtype: float64
```

THIS SHOWS THAT THERE IS A NEGATIVE MEAN OF THE FOREIGN PROFIT MARGIN (-2317)

THIS MEANS THAT THE STUDIOS HAVE A HIGH PROBABILITY OF MAKING LOSES INTERNATIONALLY MY CODE IS WRONG ON THIS MEAN THING