In [1]:

```
import pandas as pd
import matplotlib.pyplot as plt

import seaborn as sns
from sklearn.linear_model import LinearRegression
```

In [2]:

```
pd.options.display.float_format = '{:,.2f}'.format

from pandas.plotting import register_matplotlib_converters
register_matplotlib_converters()
```

In [3]:

```
data = pd.read_csv('cost_revenue_dirty.csv')
```

Exploring and Cleaning the Data

In [4]:

```
data.shape
```

Out[4]:

```
(5391, 6)
```

In [5]:

```
data.sample(5)
```

Out[5]:

	Rank	Release_Date	Movie_Title	USD_Production_Budget	USD_Worldwide_Gross	USD_D
1461	5191	7/14/2000	Chuck&Buck	$250,000	$1,157,672	
2530	1108	10/14/2005	Domino	$50,000,000	$22,969,202	
1572	2738	2/3/2001	See Spot Run	$16,000,000	$43,057,552	
242	912	12/15/1978	Superman	$55,000,000	$300,200,000	
3537	4620	12/29/2009	Lesbian Vampire Killers	$2,000,000	$3,620,902	

In [6]:

```
data.tail()
```

Out[6]:

	Rank	Release_Date	Movie_Title	USD_Production_Budget	USD_Worldwide_Gross	USD_D
5386	2950	10/8/2018	Meg	$15,000,000	$0	
5387	126	12/18/2018	Aquaman	$160,000,000	$0	
5388	96	12/31/2020	Singularity	$175,000,000	$0	
5389	1119	12/31/2020	Hannibal the Conqueror	$50,000,000	$0	
5390	2517	12/31/2020	Story of Bonnie and Clyde, The	$20,000,000	$0	

In [7]:

```
print(f'Any NaN values among the data? {data.isna().values.any()}')
```

```
Any NaN values among the data? False
```

In [8]:

```
print(f'Any duplicates? {data.duplicated().values.any()}')

duplicated_rows = data[data.duplicated()]
print(f'Number of duplicates: {len(duplicated_rows)}')
```

```
Any duplicates? False
Number of duplicates: 0
```

In [9]:

```
# Show NaN values and data types per column
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5391 entries, 0 to 5390
Data columns (total 6 columns):
 #   Column                 Non-Null Count  Dtype 
---  ------                 --------------  ----- 
 0   Rank                   5391 non-null   int64 
 1   Release_Date           5391 non-null   object
 2   Movie_Title            5391 non-null   object
 3   USD_Production_Budget  5391 non-null   object
 4   USD_Worldwide_Gross    5391 non-null   object
 5   USD_Domestic_Gross     5391 non-null   object
dtypes: int64(1), object(5)
memory usage: 252.8+ KB
```

Data Type Conversions

In [10]:

```
chars_to_remove = [',', '$']
columns_to_clean = ['USD_Production_Budget',
                    'USD_Worldwide_Gross',
                    'USD_Domestic_Gross']

for col in columns_to_clean:
    for char in chars_to_remove:
        # Replace each character with an empty string
        data[col] = data[col].astype(str).str.replace(char, "")
    # Convert column to a numeric data type
    data[col] = pd.to_numeric(data[col])
```

```
C:\Users\Isha Jain\AppData\Local\Temp\ipykernel_29584\1970920549.py:9: Futur
eWarning: The default value of regex will change from True to False in a fut
ure version. In addition, single character regular expressions will *not* be
treated as literal strings when regex=True.
  data[col] = data[col].astype(str).str.replace(char, "")
```

In [11]:

```
data.head()
```

Out[11]:

	Rank	Release_Date	Movie_Title	USD_Production_Budget	USD_Worldwide_Gross	USD_Dom
0	5293	8/2/1915	The Birth of a Nation	110000	11000000	
1	5140	5/9/1916	Intolerance	385907	0	
2	5230	12/24/1916	20,000 Leagues Under the Sea	200000	8000000	
3	5299	9/17/1920	Over the Hill to the Poorhouse	100000	3000000	
4	5222	1/1/1925	The Big Parade	245000	22000000	

In [12]:

```
data.Release_Date = pd.to_datetime(data.Release_Date)
data.head()
```

Out[12]:

	Rank	Release_Date	Movie_Title	USD_Production_Budget	USD_Worldwide_Gross	USD_Dom
0	5293	1915-08-02	The Birth of a Nation	110000	11000000	
1	5140	1916-05-09	Intolerance	385907	0	
2	5230	1916-12-24	20,000 Leagues Under the Sea	200000	8000000	
3	5299	1920-09-17	Over the Hill to the Poorhouse	100000	3000000	
4	5222	1925-01-01	The Big Parade	245000	22000000	

In [13]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5391 entries, 0 to 5390
Data columns (total 6 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   Rank                   5391 non-null   int64
 1   Release_Date           5391 non-null   datetime64[ns]
 2   Movie_Title            5391 non-null   object
 3   USD_Production_Budget  5391 non-null   int64
 4   USD_Worldwide_Gross    5391 non-null   int64
 5   USD_Domestic_Gross     5391 non-null   int64
dtypes: datetime64[ns](1), int64(4), object(1)
memory usage: 252.8+ KB
```

Descriptive Statistics

1. What is the average production budget of the films in the data set?
2. What is the average worldwide gross revenue of films?
3. What were the minimums for worldwide and domestic revenue?
4. Are the bottom 25% of films actually profitable or do they lose money?
5. What are the highest production budget and highest worldwide gross revenue of any film?
6. How much revenue did the lowest and highest budget films make?

In [14]:

```
data.describe()
```

Out[14]:

	Rank	USD_Production_Budget	USD_Worldwide_Gross	USD_Domestic_Gross
count	5,391.00	5,391.00	5,391.00	5,391.00
mean	2,696.00	31,113,737.58	88,855,421.96	41,235,519.44
std	1,556.39	40,523,796.88	168,457,757.00	66,029,346.27
min	1.00	1,100.00	0.00	0.00
25%	1,348.50	5,000,000.00	3,865,206.00	1,330,901.50
50%	2,696.00	17,000,000.00	27,450,453.00	17,192,205.00
75%	4,043.50	40,000,000.00	96,454,455.00	52,343,687.00
max	5,391.00	425,000,000.00	2,783,918,982.00	936,662,225.00

In [15]:

```
data[data.USD_Production_Budget == 1100.00]
```

Out[15]:

	Rank	Release_Date	Movie_Title	USD_Production_Budget	USD_Worldwide_Gross	USD_D
2427	5391	2005-05-08	My Date With Drew	1100	181041	

In [16]:

```
data[data.USD_Production_Budget == 425000000.00]
```

Out[16]:

	Rank	Release_Date	Movie_Title	USD_Production_Budget	USD_Worldwide_Gross	USD_D
3529	1	2009-12-18	Avatar	425000000	2783918982	

Investigating the Zero Revenue Films

In [17]:

```
zero_domestic = data[data.USD_Domestic_Gross == 0]
print(f'Number of films that grossed $0 domestically {len(zero_domestic)}')
zero_domestic.sort_values('USD_Production_Budget', ascending=False)
```

```
Number of films that grossed $0 domestically 512
```

Out[17]:

	Rank	Release_Date	Movie_Title	USD_Production_Budget	USD_Worldwide_Gross	USI
5388	96	2020-12-31	Singularity	175000000	0	
5387	126	2018-12-18	Aquaman	160000000	0	
5384	321	2018-09-03	A Wrinkle in Time	103000000	0	
5385	366	2018-10-08	Amusement Park	100000000	0	
5090	556	2015-12-31	Don Gato, el inicio de la pandilla	80000000	4547660	
...	...	...	...	...	...	
4787	5371	2014-12-31	Stories of Our Lives	15000	0	
3056	5374	2007-12-31	Tin Can Man	12000	0	
4907	5381	2015-05-19	Family Motocross	10000	0	
5006	5389	2015-09-29	Signed Sealed Delivered	5000	0	
5007	5390	2015-09-29	A Plague So Pleasant	1400	0	

512 rows × 6 columns

How many films grossed $0 worldwide? What are the highest budget films that had no revenue internationally?

In [18]:

```
zero_worldwide = data[data.USD_Worldwide_Gross == 0]
print(f'Number of films that grossed $0 worldwide {len(zero_worldwide)}')
zero_worldwide.sort_values('USD_Production_Budget', ascending=False)
```

```
Number of films that grossed $0 worldwide 357
```

Out[18]:

	Rank	Release_Date	Movie_Title	USD_Production_Budget	USD_Worldwide_Gross	USD_D
5388	96	2020-12-31	Singularity	175000000	0	
5387	126	2018-12-18	Aquaman	160000000	0	
5384	321	2018-09-03	A Wrinkle in Time	103000000	0	
5385	366	2018-10-08	Amusement Park	100000000	0	
5058	880	2015-11-12	The Ridiculous 6	60000000	0	
...	...	...	...	...	...	
4787	5371	2014-12-31	Stories of Our Lives	15000	0	
3056	5374	2007-12-31	Tin Can Man	12000	0	
4907	5381	2015-05-19	Family Motocross	10000	0	
5006	5389	2015-09-29	Signed Sealed Delivered	5000	0	
5007	5390	2015-09-29	A Plague So Pleasant	1400	0	

357 rows × 6 columns

In [19]:

```
international_releases = data.loc[(data.USD_Domestic_Gross == 0) &
                                  (data.USD_Worldwide_Gross != 0)]
print(f'Number of international releases: {len(international_releases)}')
international_releases.head()
```

```
Number of international releases: 155
```

Out[19]:

	Rank	Release_Date	Movie_Title	USD_Production_Budget	USD_Worldwide_Gross	USD_D
71	4310	1956-02-16	Carousel	3380000	3220	
1579	5087	2001-02-11	Everything Put Together	500000	7890	
1744	3695	2001-12-31	The Hole	7500000	10834406	
2155	4236	2003-12-31	Nothing	4000000	63180	
2203	2513	2004-03-31	The Touch	20000000	5918742	

In [20]:

```
international_releases = data.query('USD_Domestic_Gross == 0 and USD_Worldwide_Gross != 0')
print(f'Number of international releases: {len(international_releases)}')
international_releases.tail()
```

```
Number of international releases: 155
```

Out[20]:

	Rank	Release_Date	Movie_Title	USD_Production_Budget	USD_Worldwide_Gross	USD_D
5340	1506	2017-04-14	Queen of the Desert	36000000	1480089	
5348	2225	2017-05-05	Chāi dàn zhuānjiā	23000000	58807172	
5360	4832	2017-07-03	Departure	1100000	27561	
5372	1856	2017-08-25	Ballerina	30000000	48048527	
5374	4237	2017-08-25	Polina danser sa vie	4000000	36630	

Unreleased Films

In [21]:

```
scrape_date = pd.Timestamp('2018-5-1')
```

In [22]:

```
future_releases = data[data.Release_Date >= scrape_date]
print(f'Number of unreleased movies: {len(future_releases)}')
future_releases
```

```
Number of unreleased movies: 7
```

Out[22]:

	Rank	Release_Date	Movie_Title	USD_Production_Budget	USD_Worldwide_Gross	USD_D
5384	321	2018-09-03	A Wrinkle in Time	103000000	0	
5385	366	2018-10-08	Amusement Park	100000000	0	
5386	2950	2018-10-08	Meg	15000000	0	
5387	126	2018-12-18	Aquaman	160000000	0	
5388	96	2020-12-31	Singularity	175000000	0	
5389	1119	2020-12-31	Hannibal the Conqueror	50000000	0	
5390	2517	2020-12-31	Story of Bonnie and Clyde, The	20000000	0	

In [23]:

```
# exclude future releases
data_clean = data.drop(future_releases.index)
```

In [24]:

```
# difference is 7 rows
data.shape[0] - data_clean.shape[0]
```

Out[24]:

```
7
```

Films that Lost Money

In [25]:

```
money_losing = data_clean.loc[data_clean.USD_Production_Budget > data_clean.USD_Worldwide_G
len(money_losing)/len(data_clean)
```

Out[25]:

```
0.37277117384843983
```

In [26]:

```
money_losing = data_clean.query('USD_Production_Budget > USD_Worldwide_Gross')
money_losing.shape[0]/data_clean.shape[0]
```

Out[26]:

```
0.37277117384843983
```

Seaborn for Data Viz: Bubble Charts

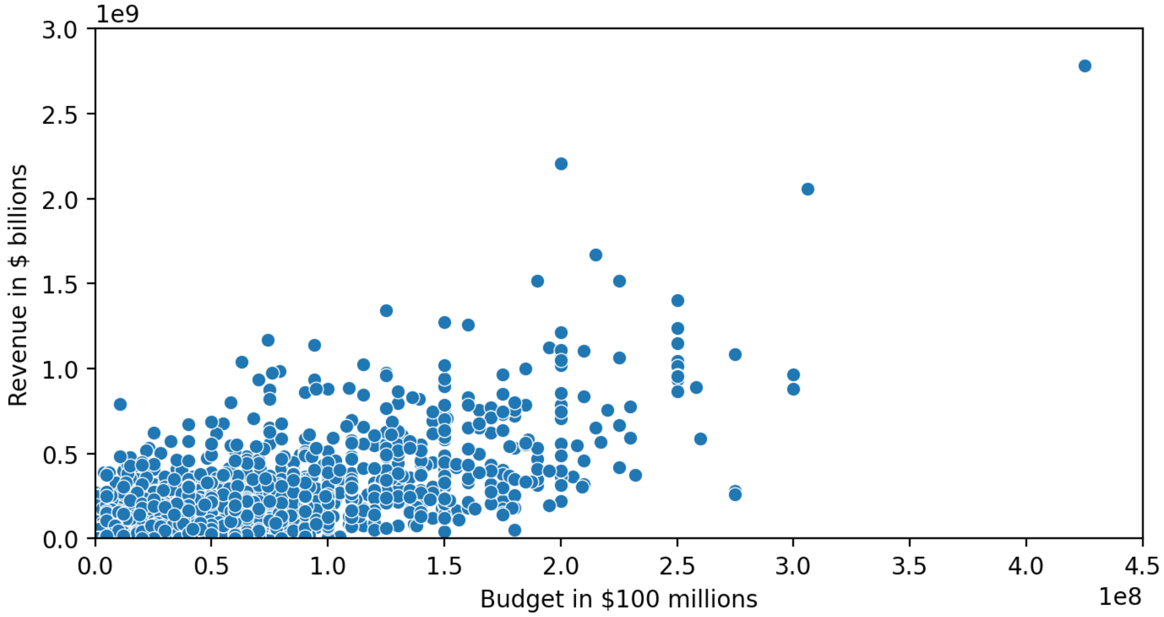
In [27]:

```
plt.figure(figsize=(8,4), dpi=200)

ax = sns.scatterplot(data=data_clean,
                     x='USD_Production_Budget',
                     y='USD_Worldwide_Gross')

ax.set(ylim=(0, 3000000000),
       xlim=(0, 450000000),
       ylabel='Revenue in $ billions',
       xlabel='Budget in $100 millions')

plt.show()
```

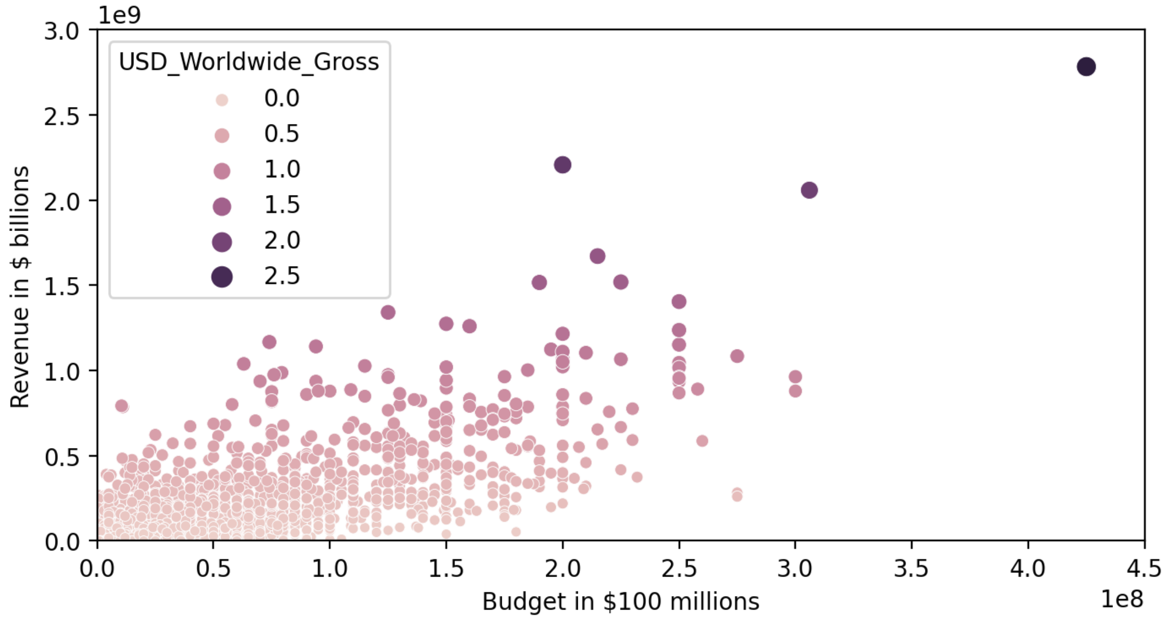


In [28]:

```
plt.figure(figsize=(8,4), dpi=200)
ax = sns.scatterplot(data=data_clean,
                     x='USD_Production_Budget',
                     y='USD_Worldwide_Gross',
                     hue='USD_Worldwide_Gross',
                     size='USD_Worldwide_Gross',)

ax.set(ylim=(0, 3000000000),
       xlim=(0, 450000000),
       ylabel='Revenue in $ billions',
       xlabel='Budget in $100 millions',)

plt.show()
```

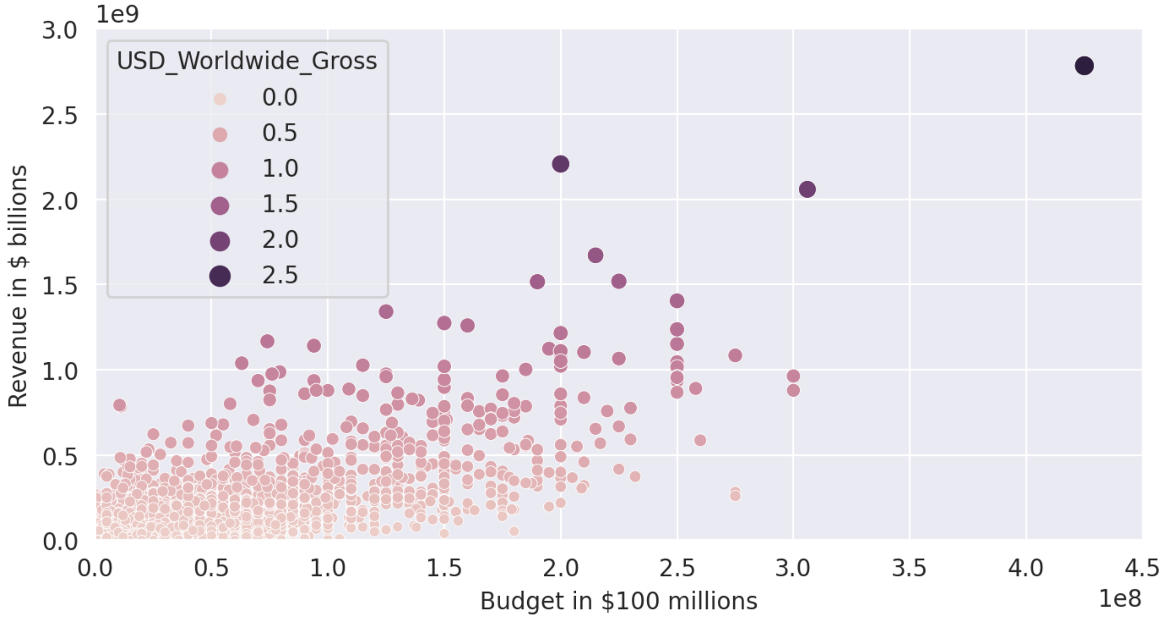


In [29]:

```
plt.figure(figsize=(8,4), dpi=200)


with sns.axes_style('darkgrid'):
  ax = sns.scatterplot(data=data_clean,
                       x='USD_Production_Budget',
                       y='USD_Worldwide_Gross',
                       hue='USD_Worldwide_Gross',
                       size='USD_Worldwide_Gross')

  ax.set(ylim=(0, 3000000000),
        xlim=(0, 450000000),
        ylabel='Revenue in $ billions',
        xlabel='Budget in $100 millions')
```



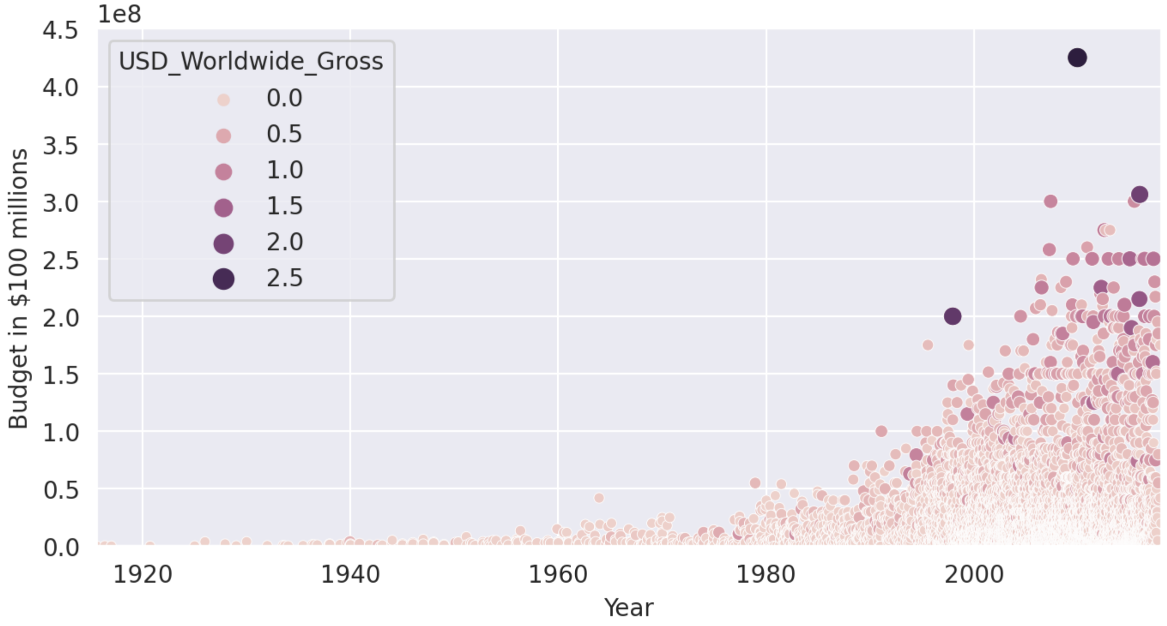
Plotting Movie Releases over Time

In [30]:

```
plt.figure(figsize=(8,4), dpi=200)

with sns.axes_style("darkgrid"):
    ax = sns.scatterplot(data=data_clean,
                    x='Release_Date',
                    y='USD_Production_Budget',
                    hue='USD_Worldwide_Gross',
                    size='USD_Worldwide_Gross',)

    ax.set(ylim=(0, 450000000),
           xlim=(data_clean.Release_Date.min(), data_clean.Release_Date.max()),
           xlabel='Year',
           ylabel='Budget in $100 millions')
```



In [31]:

```
dt_index = pd.DatetimeIndex(data_clean.Release_Date)
years = dt_index.year
```

In [32]:

```
# Converting the year 1999 to the 90s decade
1999//10
```

Out[32]:

```
199
```

In [33]:

```
199*10
```

Out[33]:

```
1990
```

In [34]:

```
decades = years//10*10
data_clean['Decade'] = decades
```

Separating the "old" (before 1969) and "New" (1970s onwards) Films

- How many films were released prior to 1970?
- What was the most expensive film made prior to 1970?

In [35]:

```
old_films = data_clean[data_clean.Decade <= 1960]
new_films = data_clean[data_clean.Decade > 1960]
```

In [36]:

```
old_films.describe()
```

Out[36]:

	Rank	USD_Production_Budget	USD_Worldwide_Gross	USD_Domestic_Gross	Decade
count	153.00	153.00	153.00	153.00	153.00
mean	4,274.77	4,611,297.65	30,419,634.38	22,389,473.87	1,949.15
std	742.14	5,713,648.85	54,931,828.93	32,641,752.41	12.72
min	1,253.00	100,000.00	0.00	0.00	1,910.00
25%	3,973.00	1,250,000.00	5,273,000.00	5,000,000.00	1,940.00
50%	4,434.00	2,900,000.00	10,000,000.00	10,000,000.00	1,950.00
75%	4,785.00	5,000,000.00	33,208,099.00	28,350,000.00	1,960.00
max	5,299.00	42,000,000.00	390,525,192.00	198,680,470.00	1,960.00

In [37]:

```
old_films.sort_values('USD_Production_Budget', ascending=False).head()
```

Out[37]:

	Rank	Release_Date	Movie_Title	USD_Production_Budget	USD_Worldwide_Gross	USD_Dc
109	1253	1963-12-06	Cleopatra	42000000	71000000	
150	2175	1969-12-16	Hello, Dolly	24000000	33208099	
143	2465	1969-01-01	Sweet Charity	20000000	8000000	
118	2425	1965-02-15	The Greatest Story Ever Told	20000000	15473333	
148	2375	1969-10-15	Paint Your Wagon	20000000	31678778	

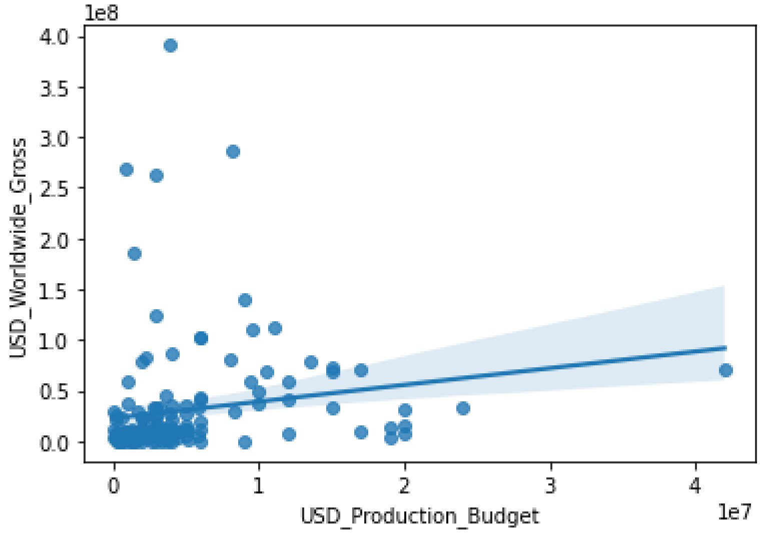
Seaborn Regression Plots

In [38]:

```
sns.regplot(data=old_films,
            x='USD_Production_Budget',
            y='USD_Worldwide_Gross')
```

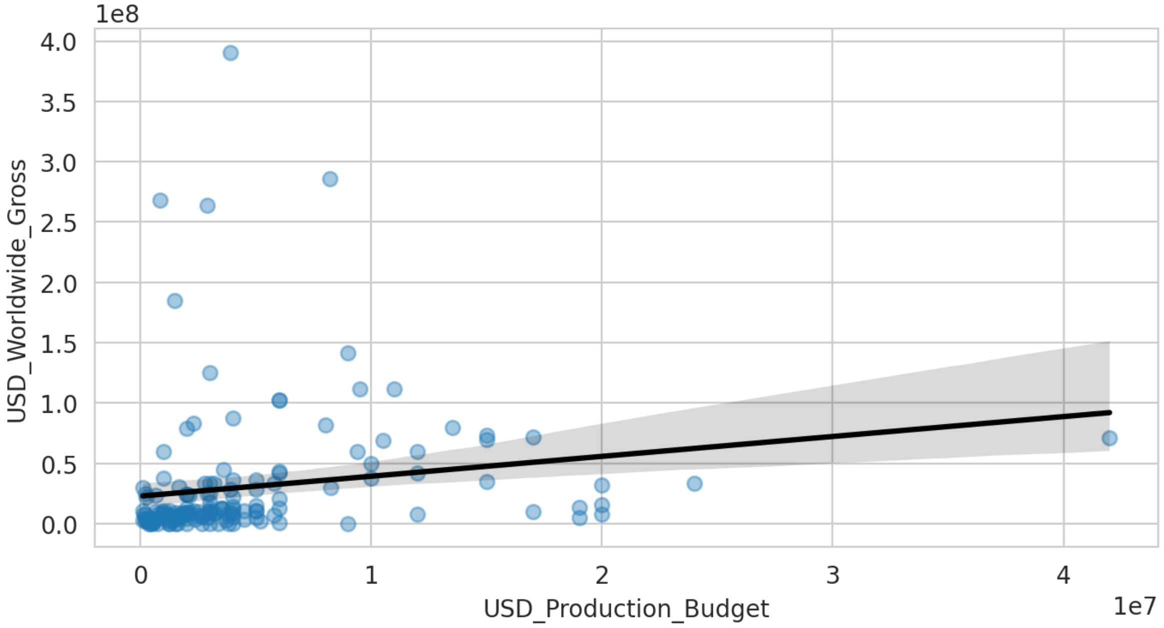
Out[38]:

```
<AxesSubplot:xlabel='USD_Production_Budget', ylabel='USD_Worldwide_Gross'>
```



In [39]:

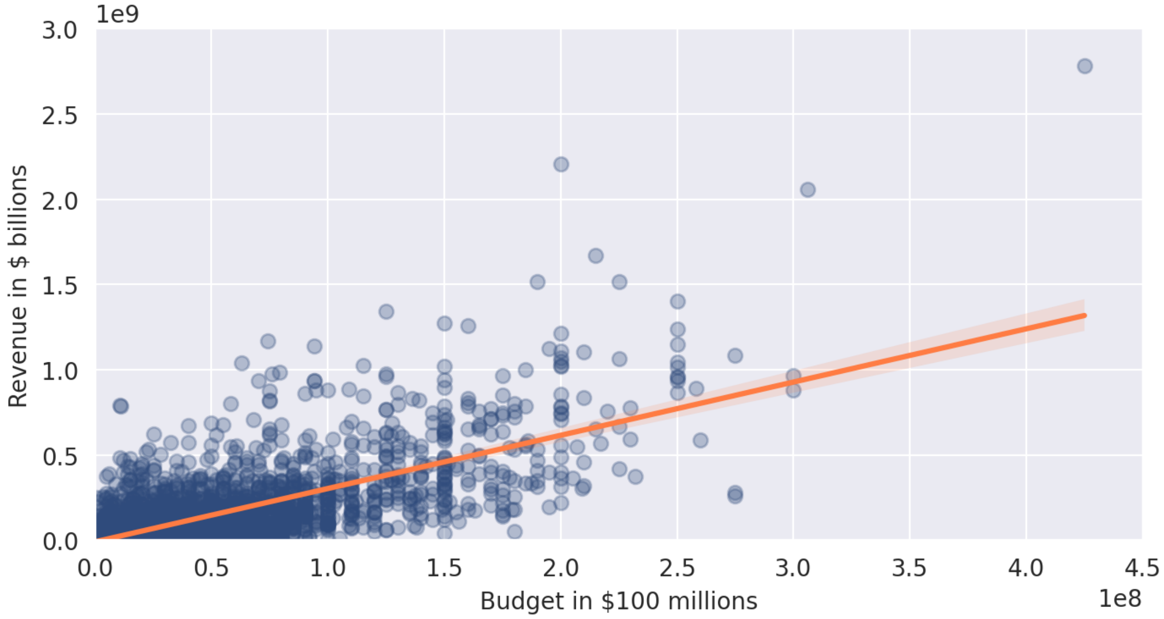
```
plt.figure(figsize=(8,4), dpi=200)
with sns.axes_style("whitegrid"):
  sns.regplot(data=old_films,
            x='USD_Production_Budget',
            y='USD_Worldwide_Gross',
            scatter_kws = {'alpha': 0.4},
            line_kws = {'color': 'black'})
```



In [40]:

```
plt.figure(figsize=(8,4), dpi=200)
with sns.axes_style('darkgrid'):
  ax = sns.regplot(data=new_films,
                   x='USD_Production_Budget',
                   y='USD_Worldwide_Gross',
                   color='#2f4b7c',
                   scatter_kws = {'alpha': 0.3},
                   line_kws = {'color': '#ff7c43'})

  ax.set(ylim=(0, 3000000000),
         xlim=(0, 450000000),
         ylabel='Revenue in $ billions',
         xlabel='Budget in $100 millions')
```


1e9
Revenue in $ billions
Budget in $100 millions
1e8