Display Imdb-movies

This data set contains information about 10,000 movies collected from The Movie Database (TMDb), including user ratings and revenue

Questions

- Does revenue increase as time goes by ?
- Does the budjet increases as time goes by ? (more tech more money i think)
- is there a relation between the vote and the revenue of the movie ?
- what is the most popular movie ?
- which movie is the longest?
- which movie has the bigest budjet and revenue ?

In [42]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from scipy import stats
import seaborn as sns
%matplotlib inline
```

In [2]:
```
df = pd.read_csv('tmdb-movies.csv')
df.head()
```

Out[2]:

	id	imdb_id	popularity	budget	revenue	original_title	cast	homepage	direc
0	135397	tt0369610	32.985763	150000000	1513528810	Jurassic World	Chris Pratt|Bryce Dallas Howard|Irrfan Khan|Vi...	http://www.jurassicworld.com/	C Trevor
1	76341	tt1392190	28.419936	150000000	378436354	Mad Max: Fury Road	Tom Hardy|Charlize Theron|Hugh Keays-Byrne|Nic...	http://www.madmaxmovie.com/	Geo M
2	262500	tt2908446	13.112507	110000000	295238201	Insurgent	Shailene Woodley|Theo James|Kate Winslet|Ansel...	http://www.thedivergentseries.movie/#insurgent	Rob Schwer
3	140607	tt2488496	11.173104	200000000	2068178225	Star Wars: The Force Awakens	Harrison Ford|Mark Hamill|Carrie Fisher|Adam D...	http://www.starwars.com/films/star-wars-episod...	Abra
4	168259	tt2820852	9.335014	190000000	1506249360	Furious 7	Vin Diesel|Paul Walker|Jason Statham|Michelle ...	http://www.furious7.com/	Jar W

5 rows × 21 columns

Assessing

- checking which columns have missing recordes, which columns have wrong datatype representation.
- found some missing recordes, some of them are not that important and some are essintial like the '1030 mising recordes in production_companies column'. I belive this column will play an important role in the analsis part, so I will try to find the missing recordes using the internet.
- I think i will drop the (homepage & tagline) columns as they have a lot of missing recorders and they won't matter in the analisys

- there are some enteries having Zero budjet and revenues, will try to find them using the internent or will fill them with the mean of there corrsponding columns for now

In [3]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10866 entries, 0 to 10865
Data columns (total 21 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   id                    10866 non-null  int64  
 1   imdb_id               10856 non-null  object 
 2   popularity            10866 non-null  float64
 3   budget                10866 non-null  int64  
 4   revenue               10866 non-null  int64  
 5   original_title        10866 non-null  object 
 6   cast                  10790 non-null  object 
 7   homepage              2936 non-null   object 
 8   director              10822 non-null  object 
 9   tagline               8042 non-null   object 
 10  keywords              9373 non-null   object 
 11  overview              10862 non-null  object 
 12  runtime               10866 non-null  int64  
 13  genres                10843 non-null  object 
 14  production_companies  9836 non-null   object 
 15  release_date          10866 non-null  object 
 16  vote_count            10866 non-null  int64  
 17  vote_average          10866 non-null  float64
 18  release_year          10866 non-null  int64  
 19  budget_adj            10866 non-null  float64
 20  revenue_adj           10866 non-null  float64
dtypes: float64(4), int64(6), object(11)
memory usage: 1.7+ MB
```

In [4]:
```
df.shape
```

Out[4]:
```
(10866, 21)
```

there are some duplicated movies

In [5]:
```
df['original_title'].value_counts()
```

Out[5]:
```
Hamlet                     4
A Christmas Carol          3
Carrie                     3
The Three Musketeers       3
Shelter                    3
                          ..
Black Snake Moan           1
Scary or Die               1
Shakespeare in Love        1
A Little Trip to Heaven    1
The Face of an Angel       1
Name: original_title, Length: 10571, dtype: int64
```

it seems like that they are not duplicated, they are just different versions

In [6]:
```
df[df['original_title']=='Hamlet']
```

Out[6]:

	id	imdb_id	popularity	budget	revenue	original_title	cast	homepage	director	tagli
1890	28238	tt1449175	0.086490	0	0	Hamlet	David Tennant|Patrick Stewart|Penny Downie|Oli...	http://www.bbc.co.uk/hamlet/	Gregory Doran	To be, or n to be
8573	10549	tt0116477	0.383469	0	0	Hamlet	Kenneth Branagh|Derek Jacobi|Julie Christie|Ri...	NaN	Kenneth Branagh	Na
8797	10688	tt0171359	0.277798	2000000	1568749	Hamlet	Ethan Hawke|Kyle MacLachlan|Diane Venora|Sam S...	NaN	Michael Almereyda	Passio Betray Revenge, host takeover
10106	10264	tt0099726	0.067973	0	20710451	Hamlet	Mel Gibson|Glenn Close|Alan Bates|Paul Scofiel...	NaN	Franco Zeffirelli	TI extraordina adaptation Shakespeare

4 rows × 21 columns

In [7]:
```
df[df['original_title']=='Wuthering Heights']
```

Out[7]:

	id	imdb_id	popularity	budget	revenue	original_title	cast	homepage	dire
1738	36597	tt1238834	0.103661	0	0	Wuthering Heights	Tom Hardy|Charlotte Riley|Andrew Lincoln|Sarah...	http://www.pbs.org/wgbh/masterpiece/wutheringh...	C Giedr
3636	9364	tt1181614	0.414629	8000000	100915	Wuthering Heights	Kaya Scodelario|James Northcote|Amy Wren|Nicho...	http://www.artificial-eye.com/film.php?cinema=...	And Ar
8332	25095	tt0104181	0.269621	0	0	Wuthering Heights	Juliette Binoche|Ralph Fiennes|Jeremy Northam|...	NaN	P Kosmir

3 rows × 21 columns

there are 5696 movies having zero budget

In [8]:
```
df['budget'].value_counts()
```

Out[8]:
```
0           5696
20000000     190
15000000     183
25000000     178
10000000     176
            ...
1645000        1
34200000       1
82500000       1
4250000        1
4653000        1
Name: budget, Length: 557, dtype: int64
```

there are 6016 movies having zero revenue

In [9]:
```
df['revenue'].value_counts()
```

Out[9]:
```
0            6016
12000000       10
10000000        8
11000000        7
5000000         6
               ...
29300000        1
32189727        1
46546197        1
106269971       1
16017403        1
Name: revenue, Length: 4702, dtype: int64
```

there is only one duplicated entry

In [10]:
```
df['id'].value_counts()
```

Out[10]:
```
42194     2
16384     1
745       1
17037     1
72334     1
         ..
11615     1
251232    1
112205    1
101731    1
9600      1
Name: id, Length: 10865, dtype: int64
```

In [11]:
```
df[df['id']== 42194]
```

Out[11]:

	id	imdb_id	popularity	budget	revenue	original_title	cast	homepage	director	tagline	...	overview	runtime	
2089	42194	tt0411951	0.59643	30000000	967000	TEKKEN	Jon Foo|Kelly Overton|Cary-Hiroyuki Tagawa|Ian...	NaN	Dwight H. Little	Survival is no game	...	In the year of 2039, after World Wars destroy ...	92	C
2090	42194	tt0411951	0.59643	30000000	967000	TEKKEN	Jon Foo|Kelly Overton|Cary-Hiroyuki Tagawa|Ian...	NaN	Dwight H. Little	Survival is no game	...	In the year of 2039, after World Wars destroy ...	92	C

2 rows × 21 columns

Quality

Missing recordes problem

- 1 missing recordes in imdb_id column****
- 76 missing recordes in cast column***
- 7930 missing recordes in homepage column
- 44 missing recordes in director column***
- missing recordes in tageline & runtime columns
- 4 missing recordes in overview column***
- 23 missing recordes in genres column***
- **"1030 mising recordes in production_companies column"**

Datatype problems

- id column's type should be string (int >> str)***
- release_date column's type should be date (str >> date)***

Other problems

- "tt" at the first of each entry in the imdb_id column should be removed***

- entry at index 2089 should be removed as it's duplicated***

- there are some movies having **Zero** budget and revenue
- - 5696 movies having zero budget
- - 6016 movies having zero revenue

Tidiness

- ddrop homepage & tagline & keywords columns***

Cleaning

In [12]:
```
df_clean = df.copy()
```

Define

remove 'tt' before every imdb_id

Code

In [13]:
```
df_clean['imdb_id'] = df_clean['imdb_id'].str[2:]
```

Test

In [14]:
```
df_clean['imdb_id'].value_counts()
```

Out[14]:
```
0411951    2
0126029    1
0119668    1
0258153    1
0437857    1
          ..
0279730    1
0068762    1
1622979    1
0120434    1
0086987    1
Name: imdb_id, Length: 10855, dtype: int64
```

In [15]:
```
df_clean.head()
```

Out[15]:

	id	imdb_id	popularity	budget	revenue	original_title	cast	homepage	dir
0	135397	0369610	32.985763	150000000	1513528810	Jurassic World	Chris Pratt|Bryce Dallas Howard|Irrfan Khan|Vi...	http://www.jurassicworld.com/	Trev
1	76341	1392190	28.419936	150000000	378436354	Mad Max: Fury Road	Tom Hardy|Charlize Theron|Hugh Keays-Byrne|Nic...	http://www.madmaxmovie.com/	Ge
2	262500	2908446	13.112507	110000000	295238201	Insurgent	Shailene Woodley|Theo James|Kate Winslet|Ansel...	http://www.thedivergentseries.movie/#insurgent	R Schw
							Harrison		

Define

remove row at index 2089 as it's duplicated

Code

In [16]:
```
duplicated = [2089]
df_clean.drop(duplicated,axis='index', inplace=True)

# reset indexes
df_clean.reset_index(drop=True, inplace=True)
```

Test

In [17]:
```
df_clean[df_clean['id']== 42194]
```

Out[17]:

	id	imdb_id	popularity	budget	revenue	original_title	cast	homepage	director	tagline	...	overview	runtime	
2089	42194	0411951	0.59643	30000000	967000	TEKKEN	Jon Foo|Kelly Overton|Cary-Hiroyuki Tagawa|Ian...	NaN	Dwight H. Little	Survival is no game	...	In the year of 2039, after World Wars destroy ...	92	Cr

1 rows × 21 columns

In [18]:
```
df_clean['id'].value_counts()
```

Out[18]:
```
16384     1
745       1
17037     1
72334     1
8849      1
         ..
11615     1
251232    1
112205    1
101731    1
9600      1
Name: id, Length: 10865, dtype: int64
```

id column's type should be string (int >> str)

Define

convert int to str

Code

In [19]:
```
df_clean['id'] = df_clean['id'].astype('str')
```

Test

In [20]:
```
df_clean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10865 entries, 0 to 10864
Data columns (total 21 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   id                    10865 non-null  object 
 1   imdb_id               10855 non-null  object 
 2   popularity            10865 non-null  float64
 3   budget                10865 non-null  int64  
 4   revenue               10865 non-null  int64  
 5   original_title        10865 non-null  object 
 6   cast                  10789 non-null  object 
 7   homepage              2936 non-null   object 
 8   director              10821 non-null  object 
 9   tagline               8041 non-null   object 
 10  keywords              9372 non-null   object 
 11  overview              10861 non-null  object 
 12  runtime               10865 non-null  int64  
 13  genres                10842 non-null  object 
 14  production_companies  9835 non-null   object 
 15  release_date          10865 non-null  object 
 16  vote_count            10865 non-null  int64  
 17  vote_average          10865 non-null  float64
 18  release_year          10865 non-null  int64  
 19  budget_adj            10865 non-null  float64
 20  revenue_adj           10865 non-null  float64
dtypes: float64(4), int64(5), object(12)
memory usage: 1.7+ MB
```

release_date column's type should be date (str >> date)

Define

Convert str to date

In [21]:
```
df_clean['release_date'] = pd.to_datetime(df_clean['release_date'])
```

Test

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10865 entries, 0 to 10864
Data columns (total 21 columns):
 #   Column                Non-Null Count  Dtype         
---  ------                --------------  -----         
 0   id                    10865 non-null  object        
 1   imdb_id               10855 non-null  object        
 2   popularity            10865 non-null  float64       
 3   budget                10865 non-null  int64         
 4   revenue               10865 non-null  int64         
 5   original_title        10865 non-null  object        
 6   cast                  10789 non-null  object        
 7   homepage              2936 non-null   object        
 8   director              10821 non-null  object        
 9   tagline               8041 non-null   object        
 10  keywords              9372 non-null   object        
 11  overview              10861 non-null  object        
 12  runtime               10865 non-null  int64         
 13  genres                10842 non-null  object        
 14  production_companies  9835 non-null   object        
 15  release_date          10865 non-null  datetime64[ns]
 16  vote_count            10865 non-null  int64         
 17  vote_average          10865 non-null  float64       
 18  release_year          10865 non-null  int64         
 19  budget_adj            10865 non-null  float64       
 20  revenue_adj           10865 non-null  float64       
dtypes: datetime64[ns](1), float64(4), int64(5), object(11)
memory usage: 1.7+ MB
```

In [23]:
```
df_clean.head(3)
```

Out[23]:

	id	imdb_id	popularity	budget	revenue	original_title	cast	homepage	director
0	135397	0369610	32.985763	150000000	1513528810	Jurassic World	Chris Pratt|Bryce Dallas Howard|Irrfan Khan|Vi...	http://www.jurassicworld.com/	Colin Trevorrow
1	76341	1392190	28.419936	150000000	378436354	Mad Max: Fury Road	Tom Hardy|Charlize Theron|Hugh Keays-Byrne|Nic...	http://www.madmaxmovie.com/	George Miller
2	262500	2908446	13.112507	110000000	295238201	Insurgent	Shailene Woodley|Theo James|Kate Winslet|Ansel...	http://www.thedivergentseries.movie/#insurgent	Robert Schwentke

3 rows × 21 columns

- 1 missing recordes in imdb_id column
- 76 missing recordes in cast column
- 44 missing recordes in director column
- 4 missing recordes in overview column
- 23 missing recordes in genres column

Define

remove those missing records rows as they are small relative to the 10866

Code

In [24]:
```
df_clean = df_clean[df_clean['imdb_id'].notnull()]
df_clean = df_clean[df_clean['cast'].notnull()]
df_clean = df_clean[df_clean['director'].notnull()]
df_clean = df_clean[df_clean['overview'].notnull()]
df_clean = df_clean[df_clean['genres'].notnull()]
```

Test

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

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 10724 entries, 0 to 10864
Data columns (total 21 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   id                    10724 non-null  object 
 1   imdb_id               10724 non-null  object 
 2   popularity            10724 non-null  float64
 3   budget                10724 non-null  int64  
 4   revenue               10724 non-null  int64  
 5   original_title        10724 non-null  object 
 6   cast                  10724 non-null  object 
 7   homepage              2891 non-null   object 
 8   director              10724 non-null  object 
 9   tagline               7999 non-null   object 
 10  keywords              9302 non-null   object 
 11  overview              10724 non-null  object 
 12  runtime               10724 non-null  int64  
 13  genres                10724 non-null  object 
 14  production_companies  9770 non-null   object 
 15  release_date          10724 non-null  datetime64[ns]
 16  vote_count            10724 non-null  int64  
 17  vote_average          10724 non-null  float64
 18  release_year          10724 non-null  int64  
 19  budget_adj            10724 non-null  float64
 20  revenue_adj           10724 non-null  float64
dtypes: datetime64[ns](1), float64(4), int64(5), object(11)
memory usage: 1.8+ MB
```

drop homepage & tagline & keywords columns, they aren't nessussry in the analysis

Define

drop homepage & tagline & keywords columns

In [28]:
```
df_clean = df_clean.drop(['homepage','tagline','keywords'], axis=1)
```

Test

In [30]:
```
df_clean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 10724 entries, 0 to 10864
Data columns (total 18 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   id                    10724 non-null  object
 1   imdb_id               10724 non-null  object
 2   popularity            10724 non-null  float64
 3   budget                10724 non-null  int64
 4   revenue               10724 non-null  int64
 5   original_title        10724 non-null  object
 6   cast                  10724 non-null  object
 7   director              10724 non-null  object
 8   overview              10724 non-null  object
 9   runtime               10724 non-null  int64
 10  genres                10724 non-null  object
 11  production_companies  9770 non-null   object
 12  release_date          10724 non-null  datetime64[ns]
 13  vote_count            10724 non-null  int64
 14  vote_average          10724 non-null  float64
 15  release_year          10724 non-null  int64
 16  budget_adj            10724 non-null  float64
 17  revenue_adj           10724 non-null  float64
dtypes: datetime64[ns](1), float64(4), int64(5), object(8)
memory usage: 1.6+ MB
```

large number of lost data

- 5696 movies having zero budget
- 6016 movies having zero revenue
- 1030 mising recordes in production_companies column

I will devide the data frame to two data frames, one will have all the 10724 records with out the (budget,revenue,companies) columns. and the other one will have less records and with the (budget,revenue,companies) columns.

the other one will have less records because i will drop the rows which have null values or zero in those 3 columns (budget,revenue,companies).

I will do this and not just drop them because there are insights i wanna get from those 3 columns. and i don't wanna just fill the missing recordes with mean beause i they are alot and i don't think filling them with the mean will result in accurate insights in this case.

i could have done somthing else which is rather better, scrap the web for the missing values, but i didn't go with that option because i have final exams :'

Define

create df_clean_big dataframe which contains the 10724 records and all columns but without (budget,revenue,homepage,companies)

In [31]:
```
df_clean_big = df_clean.drop(['budget','revenue','production_companies'], axis=1)
```

create df_clean_small dataframe which contains all the rows
then drop the records which have zero or null values

In [32]:
```
df_clean_small = df_clean.copy()
df_clean_small = df_clean_small[df_clean_small['budget'] !=0]
df_clean_small = df_clean_small[df_clean_small['revenue'] !=0]
df_clean_small = df_clean_small[df_clean_small['production_companies'].notnull()]
```

Test

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

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 10724 entries, 0 to 10864
Data columns (total 15 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   id              10724 non-null  object
 1   imdb_id         10724 non-null  object
 2   popularity      10724 non-null  float64
 3   original_title  10724 non-null  object
 4   cast            10724 non-null  object
 5   director        10724 non-null  object
 6   overview        10724 non-null  object
 7   runtime         10724 non-null  int64
 8   genres          10724 non-null  object
 9   release_date    10724 non-null  datetime64[ns]
 10  vote_count      10724 non-null  int64
 11  vote_average    10724 non-null  float64
 12  release_year    10724 non-null  int64
 13  budget_adj      10724 non-null  float64
 14  revenue_adj     10724 non-null  float64
dtypes: datetime64[ns](1), float64(4), int64(3), object(7)
memory usage: 1.3+ MB
```

In [34]:
```
df_clean_small.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 3805 entries, 0 to 10847
Data columns (total 18 columns):
 #   Column                Non-Null Count  Dtype         
---  ------                --------------  -----         
 0   id                    3805 non-null   object        
 1   imdb_id               3805 non-null   object        
 2   popularity            3805 non-null   float64       
 3   budget                3805 non-null   int64         
 4   revenue               3805 non-null   int64         
 5   original_title        3805 non-null   object        
 6   cast                  3805 non-null   object        
 7   director              3805 non-null   object        
 8   overview              3805 non-null   object        
 9   runtime               3805 non-null   int64         
 10  genres                3805 non-null   object        
 11  production_companies  3805 non-null   object        
 12  release_date          3805 non-null   datetime64[ns]
 13  vote_count            3805 non-null   int64         
 14  vote_average          3805 non-null   float64       
 15  release_year          3805 non-null   int64         
 16  budget_adj            3805 non-null   float64       
 17  revenue_adj           3805 non-null   float64       
dtypes: datetime64[ns](1), float64(4), int64(5), object(8)
memory usage: 564.8+ KB
```

Define

- reset the indexses

Code

In [37]:
```
df_clean_small.reset_index(drop=True, inplace=True)
df_clean_big.reset_index(drop=True, inplace=True)
```

Analysis & Visualization

Questions

- Does revenue increase as time goes by ?
- Does the budjet increases as time goes by ? (more tech more money i think)
- is there a relation between the vote and the revenue of the movie ?
- - i will answer thos question using the small dataset which have the revenue info

- is there a relation between the length of the movie and the popularity ?
- - i will answer thos question using the big dataset which have the revenue info
- what is the most popular movie ?
- which movie is the longest?
- which movie has the bigest budjet and revenue ?

In [ ]:

In [ ]:

In [ ]:

- Does revenue increase as time goes by ?

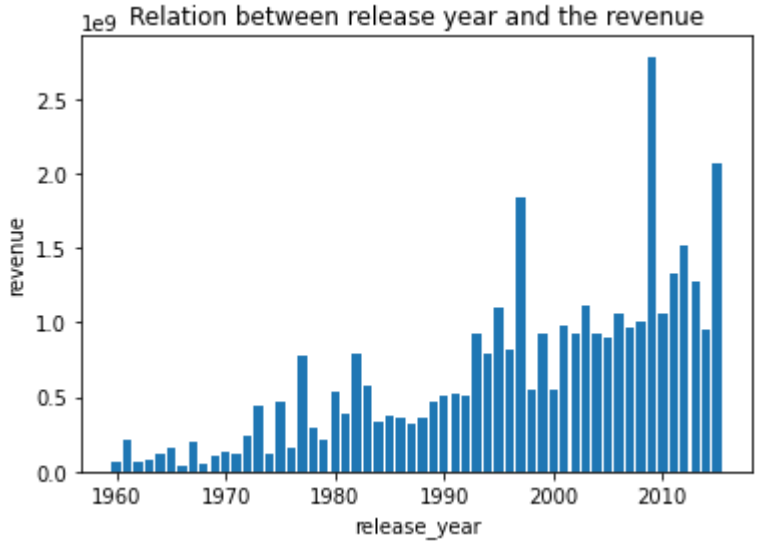
In [50]:
```
df_clean_small.head(3)
```

Out[50]:

	id	imdb_id	popularity	budget	revenue	original_title	cast	director	overview	runtime	gen
0	135397	0369610	32.985763	150000000	1513528810	Jurassic World	Chris Pratt|Bryce Dallas Howard|Irrfan Khan|Vi...	Colin Trevorrow	Twenty-two years after the events of Jurassic ...	124	Action|Adventure|Scier Fiction|Thri
1	76341	1392190	28.419936	150000000	378436354	Mad Max: Fury Road	Tom Hardy|Charlize Theron|Hugh Keays-Byrne|Nic...	George Miller	An apocalyptic story set in the furthest reach...	120	Action|Adventure|Scier Fiction|Thri
2	262500	2908446	13.112507	110000000	295238201	Insurgent	Shailene Woodley|Theo James|Kate Winslet|Ansel...	Robert Schwentke	Beatrice Prior must confront her inner demons ...	119	Adventure|Scier Fiction|Thri

In [72]:
```
plt.bar(df_clean_small['release_year'], df_clean_small['revenue']);
plt.xlabel('release_year');
plt.ylabel('revenue');
plt.title("Relation between release year and the revenue")
```

Out[72]:
```
Text(0.5, 1.0, 'Relation between release year and the revenue')
```

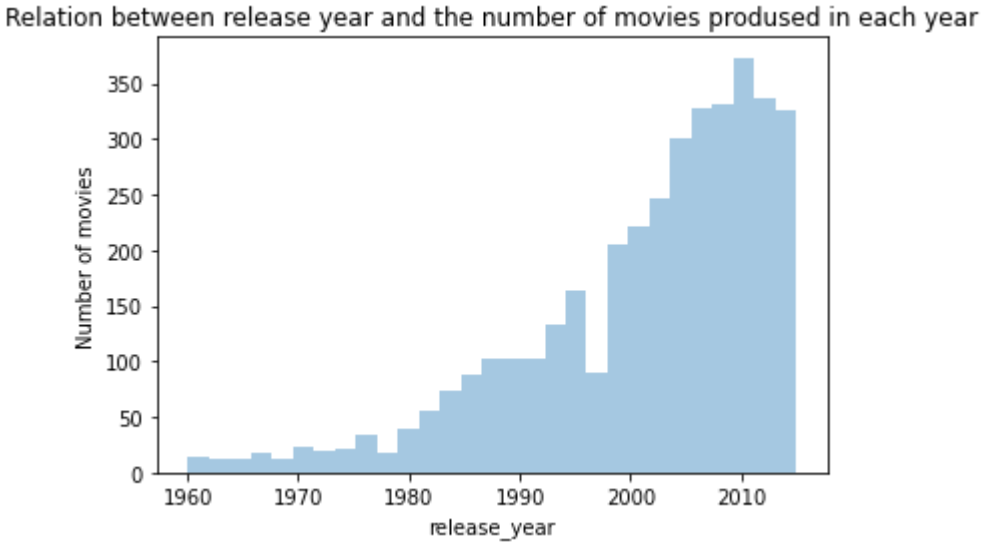


- This shows that as time goes by, the revenue of movies increase. I suspected that this is due to increasing popurality and the number of movies which are being made.

In [73]:
```
sns.distplot(df_clean_small['release_year'],kde=False);
plt.xlabel('release_year');
plt.ylabel('Number of movies');
plt.title("Relation between release year and the number of movies produsedin each year")
```

```
c:\users\eslam\appdata\local\programs\python\python37\lib\site-packages\seaborn\distributions.py:2557: FutureW
arning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to
use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function
for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[73]:
```
Text(0.5, 1.0, 'Relation between release year and the number of movies produsedin each year')
```



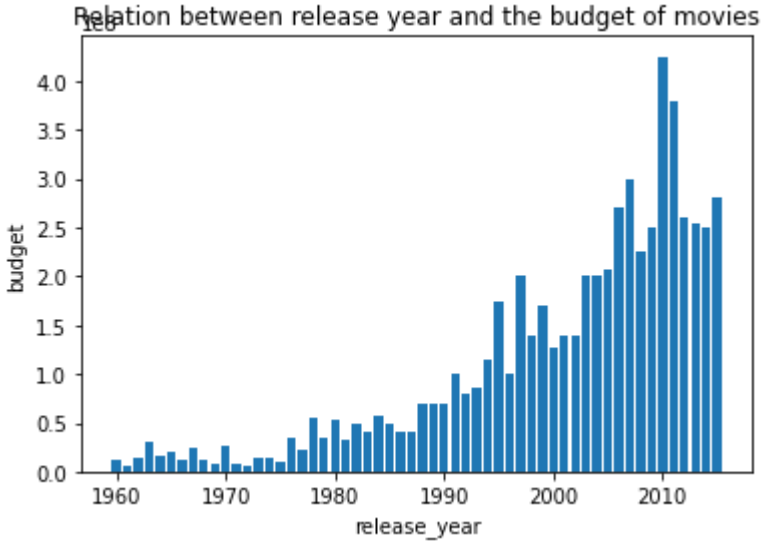
- Here as time goes by, the number of movies produced increases almost exponentionly, so yes part of the fact that revenue increases is due to the increasing number of movies produsedin.

- Does the budjet increases as time goes by ? (more tech more money i think)

Yeah, it increases

In [75]:
```
plt.bar(df_clean_small['release_year'], df_clean_small['budget']);
plt.xlabel('release_year');
plt.ylabel('budget');
plt.title("Relation between release year and the budget of movies")
```

Out[75]:
```
Text(0.5, 1.0, 'Relation between release year and the budget of movies')
```

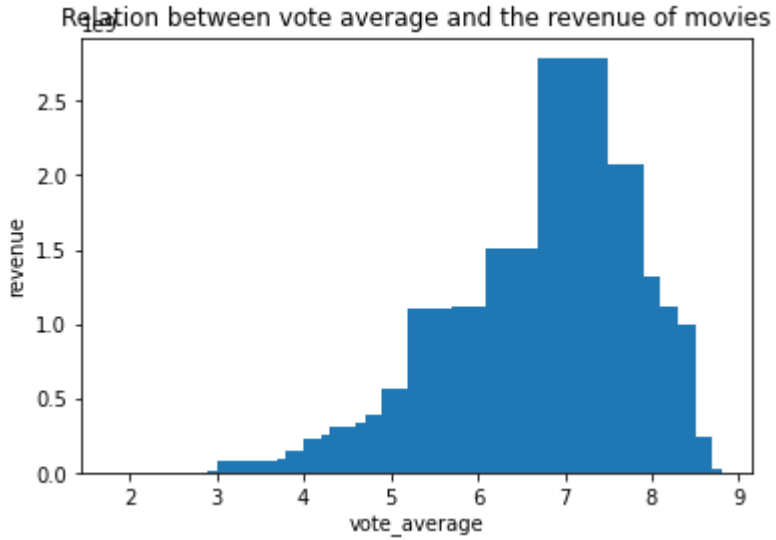


- this plot shows that as time goes by, the budget of movies increases. and this is probably due to the incease in production cost (tech for example) and due to the increasing number of movies too.

is there a relation between the vote and the revenue of the movie ?

In [76]:
```
plt.bar(df_clean_small['vote_average'], df_clean_small['revenue']);
plt.xlabel('vote_average');
plt.ylabel('revenue');
plt.title("Relation between vote average and the revenue of movies")
```

Out[76]:
```
Text(0.5, 1.0, 'Relation between vote average and the revenue of movies')
```



this plot shows that:

- as the number of votes increases, the revenue increases too, but to a certian point
- so it's not always true

is there a relation between the length of the movie and the popularity ?

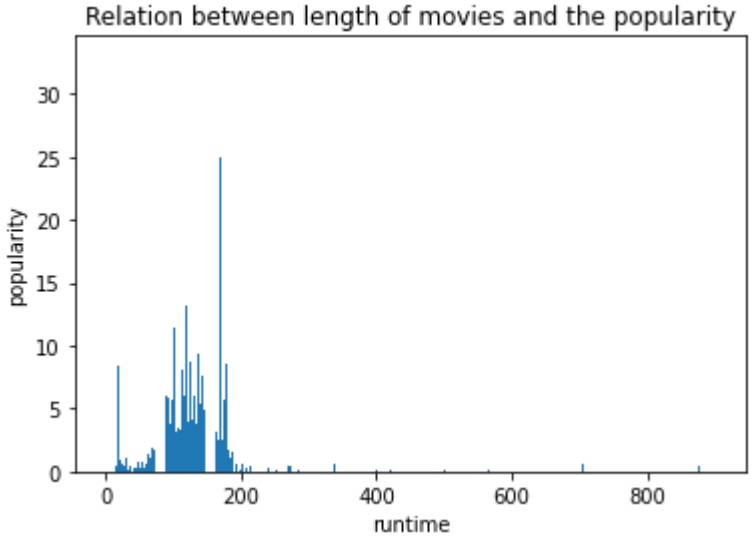
In [55]:
```
df_clean_big.head(3)
```

Out[55]:

	id	imdb_id	popularity	original_title	cast	director	overview	runtime	genres	release_date	vote_c
0	135397	0369610	32.985763	Jurassic World	Chris Pratt|Bryce Dallas Howard|Irrfan Khan|Vi...	Colin Trevorrow	Twenty-two years after the events of Jurassic ...	124	Action|Adventure|Science Fiction|Thriller	2015-06-09	
1	76341	1392190	28.419936	Mad Max: Fury Road	Tom Hardy|Charlize Theron|Hugh Keays-Byrne|Nic...	George Miller	An apocalyptic story set in the furthest reach...	120	Action|Adventure|Science Fiction|Thriller	2015-05-13	
2	262500	2908446	13.112507	Insurgent	Shailene Woodley|Theo James|Kate Winslet|Ansel...	Robert Schwentke	Beatrice Prior must confront her inner demons ...	119	Adventure|Science Fiction|Thriller	2015-03-18	

In [77]:
```
plt.bar(df_clean_big['runtime'], df_clean_big['popularity']);
plt.xlabel('runtime');
plt.ylabel('popularity');
plt.title("Relation between length of movies and the popularity")
```

Out[77]:
```
Text(0.5, 1.0, 'Relation between length of movies and the popularity')
```



- this plot show the relation between the length of the movie and the popularity
- it somehow follows normal distripution, so it's not a direct relation
- most popular movies are within the range of 100 and 200 munits long

what is the most popular movie ?

- it's Jurassic World

In [66]:
```
df_clean_big['original_title'][df_clean_big['popularity'] == df_clean_big['popularity'].max()]
```

Out[66]:
```
0    Jurassic World
Name: original_title, dtype: object
```

which movie is the longest?

- it's The Story of Film: An Odyssey

In [68]:
```
df_clean_big['original_title'][df_clean_big['runtime'] == df_clean_big['runtime'].max()]
```

Out[68]:
```
3820    The Story of Film: An Odyssey
Name: original_title, dtype: object
```

which movie has the bigest budjet and revenue ?

- it's Avatar

In [69]:
```
df_clean_small['original_title'][df_clean_small['revenue'] == df_clean_small['revenue'].max()]
```

Out[69]:
```
344    Avatar
Name: original_title, dtype: object
```

Conclusions

- there is no direct relation between the lenghth of the movie and it's popularity.
- most popular movies are within the range of 100 and 200 munits long

- the voting for a movie can indicate wether this movie will get high revenue or not
- almost we can say that when voiting increases there is a high chance of increasment in the revenue, but this isn't always true

- as time goes by, the budget as well as the revenue increas.

- the number of movies increases as time goes by

- "Avatar" has the highst revenue of all movies
- and "it's The Story of Film: An Odyssey" is the longest
- and "Jurassic World" is the most popular

Limitiations

- there where a lot of wrong records (Zero) like in the revenue and budget columns. almost the half of the dataset
- more relible solution would be to scrap the web for those missing records.
- other than that i think this dataset is pretty much good.