> **Tip**: Welcome to the Investigate a Dataset project! You will find tips in quoted sections like this to help organize your approach to your investigation. Before submitting your project, it will be a good idea to go back through your report and remove these sections to make the presentation of your work as tidy as possible. First things first, you might want to double-click this Markdown cell and change the title so that it reflects your dataset and investigation.

Project: Investigate TMDb movie data

Table of Contents

In [ ]:

In [ ]:

Introduction

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

1. Certain columns, like ‘cast’ and ‘genres’, contain multiple values separated by pipe (|) characters.
2. There are some odd characters in the ‘cast’ column. Don’t worry about cleaning them. You can leave them as is.
3. The final two columns ending with “_adj” show the budget and revenue of the associated movie in terms of 2010 dollars, accounting for inflation over time.

The dataset contains data for **10866:**movies. There are total 21 columns with few of the interesting fields like revenue, budget, genres etc. which I will be using today for the analysis.

As mentioned in the rubrics 'The project clearly states one or more questions, then addresses those questions in the rest of the analysis.', I will attempt following questions from the dataset.
(a) **How have movie genres changed over time?**
I am planning to use variables such as: Genres (Primary), Total No. of movies, Release year, Budget, Revenue, Profit (Revenue – Budget)
(b) **Assuming I am planning to launch my production house, which movie I should create ?** To provide the recommendation I will try to answer the follow up questions:
1) Most profitable genre
2) Most popular genre
3) Highest budget movies
4) Most Profitable movies
I will also get some insight from the first question

In [1]:
```
#pip install wordcloud
```

In [2]:
```
# Use this cell to set up import statements for all of the packages that you
#   plan to use.

# Remember to include a 'magic word' so that your visualizations are plotted
#   inline with the notebook. See this page for more:
#   http://ipython.readthedocs.io/en/stable/interactive/magics.html

import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
import csv as csv
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator
```

Data Wrangling

> **Tip**: In this section of the report, you will load in the data, check for cleanliness, and then trim and clean your dataset for analysis. Make sure that you document your steps carefully and justify your cleaning decisions.

General Properties

In [3]:
```
# Load your data and print out a few lines. Perform operations to inspect data
#   types and look for instances of missing or possibly errant data.

#Load the data
df = pd.read_csv('tmdb-movies.csv')
# print first 5 and last 5 records, one at a time
df.head(5)
#df.tail(5)
```

	id	imdb_id	popularity	budget	revenue	original_title	cast	homepage	director
0	135397	tt0369610	32.985763	150000000	1513528810	Jurassic World	Chris PrattlBryce Dallas HowardlIrrfan KhanlVi...	http://www.jurassicworld.com/	Colin Trevorrow
1	76341	tt1392190	28.419936	150000000	378436354	Mad Max: Fury Road	Tom HardylCharlize TheronlHugh Keays-ByrnelNic...	http://www.madmaxmovie.com/	George Miller
2	262500	tt2908446	13.112507	110000000	295238201	Insurgent	Shailene WoodleylTheo JameslKate WinsletlAnsel...	http://www.thedivergentseries.movie/#insurgent	Robert Schwentke
3	140607	tt2488496	11.173104	200000000	2068178225	Star Wars: The Force Awakens	Harrison FordlMark HamillCarrie FisherlAdam D...	http://www.starwars.com/films/star-wars-episod...	J.J. Abrams
4	168259	tt2820852	9.335014	190000000	1506249360	Furious 7	Vin DieselPaul WalkerlJason StathamlMichelle ...	http://www.furious7.com/	James Wan

5 rows × 21 columns

In [4]:
```
# describe the dataset
df.describe()
```

	id	popularity	budget	revenue	runtime	vote_count	vote_average	release_year	budget_adj	rev
count	10866.000000	10866.000000	1.086600e+04	1.086600e+04	10866.000000	10866.000000	10866.000000	10866.000000	1.086600e+04	1.0
mean	66064.177434	0.646441	1.462570e+07	3.982332e+07	102.070863	217.389748	5.974922	2001.322658	1.755104e+07	5.1
std	92130.136561	1.000185	3.091321e+07	1.170035e+08	31.381405	575.619058	0.935142	12.812941	3.430616e+07	1.4
min	5.000000	0.000065	0.000000e+00	0.000000e+00	0.000000	10.000000	1.500000	1960.000000	0.000000e+00	0.0
25%	10596.250000	0.207583	0.000000e+00	0.000000e+00	90.000000	17.000000	5.400000	1995.000000	0.000000e+00	0.0
50%	20669.000000	0.383856	0.000000e+00	0.000000e+00	99.000000	38.000000	6.000000	2006.000000	0.000000e+00	0.0
75%	75610.000000	0.713817	1.500000e+07	2.400000e+07	111.000000	145.750000	6.600000	2011.000000	2.085325e+07	3.3
max	417859.000000	32.985763	4.250000e+08	2.781506e+09	900.000000	9767.000000	9.200000	2015.000000	4.250000e+08	2.8

> **Tip**: You should *not* perform too many operations in each cell. Create cells freely to explore your data. One option that you can take with this project is to do a lot of explorations in an initial notebook. These don't have to be organized, but make sure you use enough comments to understand the purpose of each code cell. Then, after you're done with your analysis, create a duplicate notebook where you will trim the excess and organize your steps so that you have a flowing, cohesive report.
>
> **Tip**: Make sure that you keep your reader informed on the steps that you are taking in your investigation. Follow every code cell, or every set of related code cells, with a markdown cell to describe to the reader what was found in the preceding cell(s). Try to make it so that the reader can then understand what they will be seeing in the following cell(s).

Data Cleaning

> Following data cleasing activities will be performed in upcoming cells
> (a) **Remove duplicates**
> (b) **Remove extra columns. Column required for analysis are - id, popularity, budget, revenue, runtime, genres, production companies, keywords, tagline**
> (c) **Remove records with budget and revenue as 0 as those are required to calculate the profitability**
> (d) **Find and handle NULL values, either drop or replace with 0**
> (e) **Split the column values by 'l' and take the first value from genres and production companies**
> (f) **Validate the datatype for each field and update if required**

In [5]:
```
# Find duplicate values in the data set, only 1 duplicate record found
sum(df.duplicated())
```

```
1
```

In [6]:
```
# remove the duplicate record and run the same command again to see if there are no more d
uplicates
df.drop_duplicates(inplace = True)
sum(df.duplicated())
```

```
0
```

In [7]:
```
# Validate the record count, one record dropped
df.shape
```

```
(10865, 21)
```

In [8]:
```
#remove extra columns
df.drop(['imdb_id','original_title','cast','homepage','director','overview','release_date'
,'vote_count','vote_average','budget_adj','revenue_adj'],axis =1,inplace = True)
```

In [9]:
```
df.head()
```

	id	popularity	budget	revenue	tagline	keywords	runtime	genres	production_c
0	135397	32.985763	150000000	1513528810	The park is open.	monsterIdnaltyrannosaurus rexIvelociraptorIisland	124	ActionIAdventureIScience FictionIThriller	Universal Studio EntertainmentIL
1	76341	28.419936	150000000	378436354	What a Lovely Day.	futureIchaseIpost-apocalypticIdystopiaIaustralia	120	ActionIAdventureIScience FictionIThriller	Village Roadsho PicturesIKenned Produ...
2	262500	13.112507	110000000	295238201	One Choice Can Destroy You	based on novelIrevolutionIdystopiaIsequelIdyst...	119	AdventureIScience FictionIThriller	Summit EntertainmentIM FilmsIRed Wago
3	140607	11.173104	200000000	2068178225	Every generation has a story.	androidIspaceshipIjediIspace operaI3d	136	ActionIAdventureIScience FictionIFantasy	LucasfilmITruen ProductionsIBad
4	168259	9.335014	190000000	1506249360	Vengeance Hits Home	car raceIspeedIrevengeIsuspenseIcar	137	ActionICrimeIThriller	Universal Picture FilmIMedia Righ

In [10]:
```
print('Records before dropping NULL',df.shape)
df.isnull().sum()

#drop the records with NULL values
df.dropna(axis=0, inplace=True)
print('Records after dropping NULL',df.shape)
```

```
Records before dropping NULL (10865, 10)
Records after dropping NULL (7046, 10)
```

In [11]:
```
# Movie data with 0 value populated for runtime, budget and revenue seems unrealistic and
hence can be dropped for better analysis
# replace the 0 values with NAN
# I have create three copies for each question and will remove the null values based on th
a analysis
df1 = df.copy()
df2 = df.copy()
df3 = df.copy()

print('df1 = ',df1.shape)
print('df2 = ',df2.shape)
print('df3 = ',df3.shape)

df1['runtime'].replace(0, np.NaN, inplace=True)
df1['revenue'].replace(0, np.NaN, inplace=True)
df1['budget'].replace(0, np.NaN, inplace=True)

# For question # 1: Remove the records with no value in the budget, revenue and runtime as
those are important parameter.
df1.dropna(subset=['budget','revenue','runtime'], inplace=True)

print('df1 = ',df1.shape)
print('df2 = ',df2.shape)
print('df3 = ',df3.shape)

df.isnull().sum()
```

```
df1 =  (7046, 10)
df2 =  (7046, 10)
df3 =  (7046, 10)
df1 =  (3446, 10)
df2 =  (7046, 10)
df3 =  (7046, 10)


id                      0
popularity              0
budget                  0
revenue                 0
tagline                 0
keywords                0
runtime                 0
genres                  0
production_companies    0
release_year            0
dtype: int64
```

The data in genre, keywords and prodution companies is contactinated with pipe '|' operator. I will pick the first letter from genre and production compaies and will need to search for keyword.
To perform this task I will create a function to split the columns

In [12]:
```
# After discussing the structure of the data and any problems that need to be
#   cleaned, perform those cleaning steps in the second part of this section.

#Function split
def split(column):
    return column.str[0:].str.split('|', expand = True)
genres = split(df1['genres'])
production_companies = split(df1['production_companies'])
keywords = split(df1['keywords'])

df1["genres"] = genres[0]
df1["production_companies"] = production_companies[0]

genres = split(df2['genres'])
production_companies = split(df2['production_companies'])
keywords = split(df2['keywords'])
df2["genres"] = genres[0]
df2["production_companies"] = production_companies[0]
```

The df2 and df2 dataframes will be used in the question 1 and 2, will have genres and production_companies splitted

In [13]:
```
df1.dtypes
# Add new column to the dataframe to capture the Profit = Revenue - Budget
df1['profit'] = df1['revenue'].sub(df1['budget'],axis = 'index')
display(df1.groupby('genres').sum().sort_values(by = 'profit',ascending = False)['profit']
[0:10])
display(df1.groupby('genres').sum().sort_values(by = 'popularity',ascending = False)['prof
it'][0:10])



# Action movies are most profitable
```

```
genres
Action             5.803415e+10
Adventure          4.826626e+10
Comedy             3.949813e+10
Drama              3.453913e+10
Animation          1.943329e+10
Science Fiction    1.178472e+10
Fantasy            1.139852e+10
Horror             1.067095e+10
Crime              7.599643e+09
Thriller           7.287252e+09
Name: profit, dtype: float64


genres
Action             5.803415e+10
Drama              3.453913e+10
Comedy             3.949813e+10
Adventure          4.826626e+10
Horror             1.067095e+10
Science Fiction    1.178472e+10
Thriller           7.287252e+09
Crime              7.599643e+09
Animation          1.943329e+10
Fantasy            1.139852e+10
Name: profit, dtype: float64
```

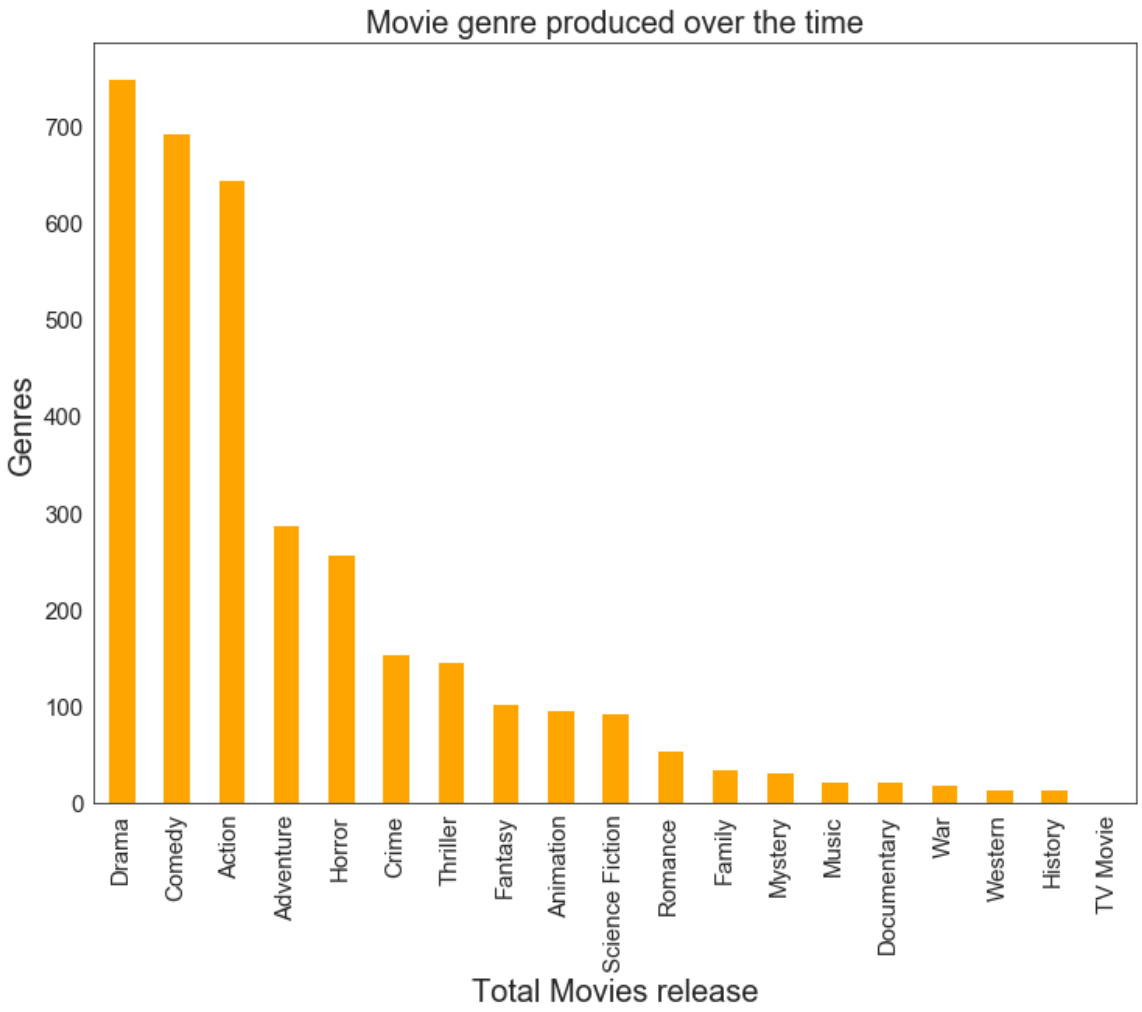
Exploratory Data Analysis

> **Tip**: Now that you've trimmed and cleaned your data, you're ready to move on to exploration. Compute statistics and create visualizations with the goal of addressing the research questions that you posed in the Introduction section. It is recommended that you be systematic with your approach. Look at one variable at a time, and then follow it up by looking at relationships between variables.

Question # 1: How have movie genres changed over time?

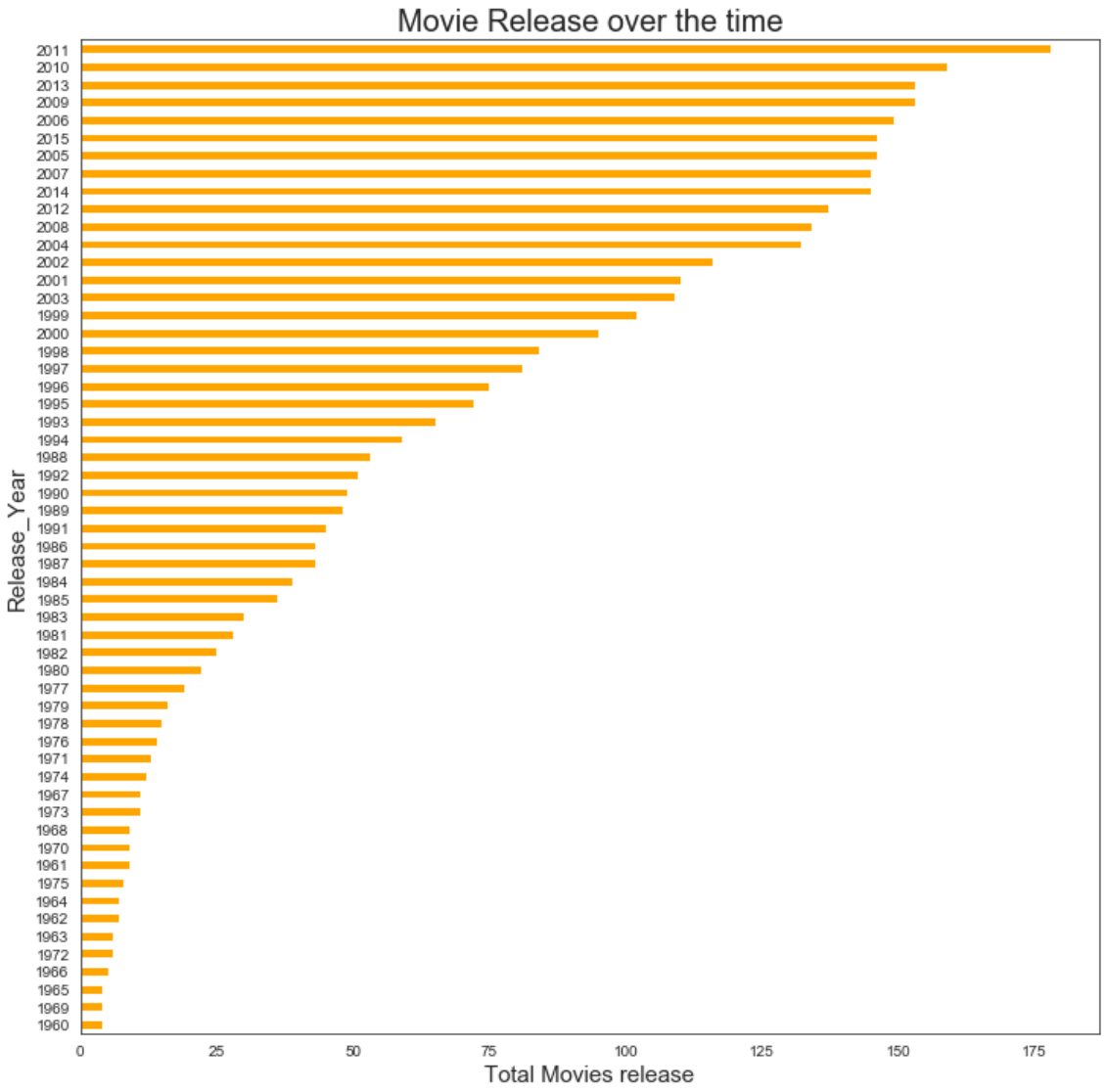
I am planning to use variables such as: Genres (Primary), Total No. of movies, Release year, Budget, Revenue, Profit (Revenue – Budget)

In [14]:
```
# Use this, and more code cells, to explore your data. Don't forget to add
#   Markdown cells to document your observations and findings.

movies = df1.groupby('genres').count()['id'].copy()

sns.set_style("white")
movies.sort_values(ascending = False).plot(kind= 'bar',figsize = (12,9),fontsize=15,color
= 'orange',)
plt.xlabel('Total Movies release' , fontsize = 20)
plt.ylabel('Genres' , fontsize = 20)
plt.title('Movie genre produced over the time',fontsize = 20)
plt.show()

df1['genres'].value_counts()
```



```
Drama              750
Comedy             694
Action             646
Adventure          288
Horror             258
Crime              154
Thriller           146
Fantasy            103
Animation           97
Science Fiction     94
Romance             55
Family              36
Mystery             33
Music               22
Documentary         22
War                 19
History             14
Western             14
TV Movie             1
Name: genres, dtype: int64
```

In [15]:
```
movies = df1.groupby('release_year').count()['id'].copy()

sns.set_style("white")
movies.sort_values(ascending = True).plot(kind= 'barh',figsize = (12,12),fontsize=10,color
= 'orange',)
plt.xlabel('Total Movies release' , fontsize = 15)
plt.ylabel('Release_Year' , fontsize = 15)
plt.title('Movie Release over the time',fontsize = 20)
plt.show()
```



Conclusions

1. Drama, Comedy and Action movies are produced most in the last 55 years, the production increased substantially from 80s.
2. Drama and Comedy are the most movies are most popular especially over the last 3 decades

Tip: Once you are satisfied with your work, you should save a copy of the report in HTML or PDF form via the **File** > **Download as** submenu. Before exporting your report, check over it to make sure that the flow of the report is complete. You should probably remove all of the "Tip" quotes like this one so that the presentation is as tidy as possible. Congratulations!

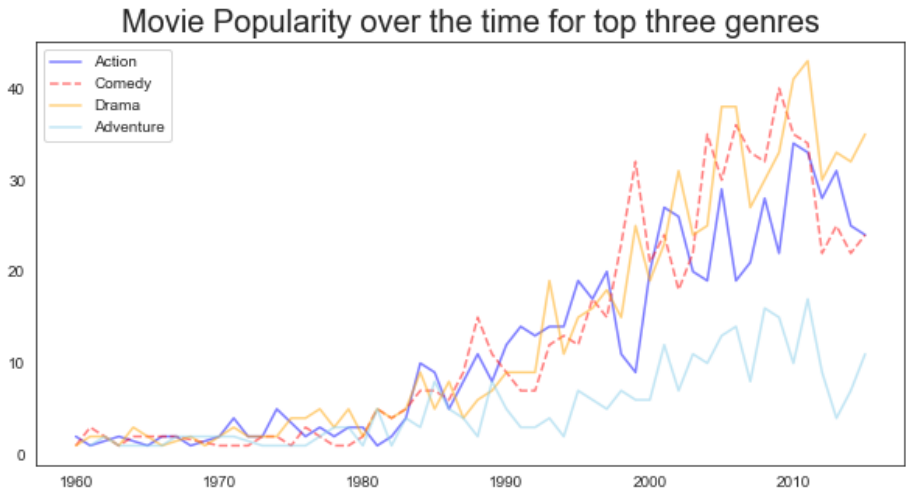
Research Question 2 Assuming I am planning to launch my production house, which movie I should create ?

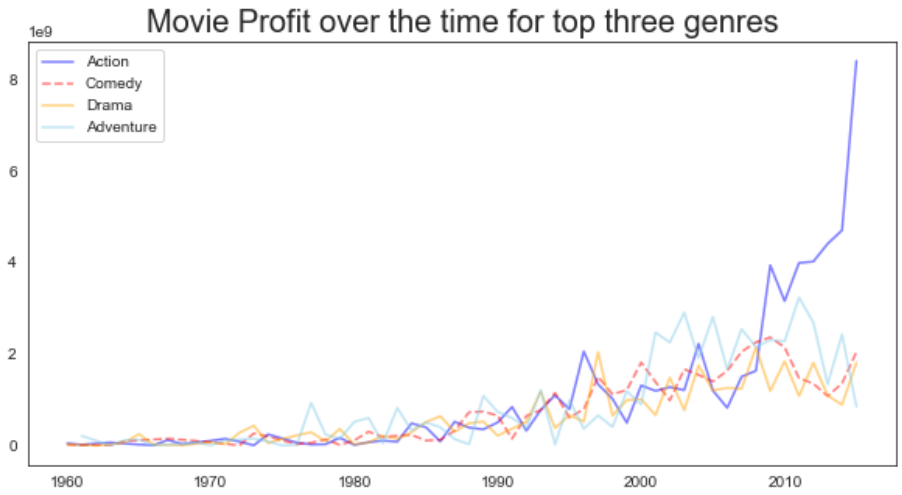
From question 1 we learned that:
1. Drama Comedy and Action are the most produced genre so I will perform analysis on different variables like popularity, Director, Categorization (keywords), profitability etc. to decide what kind of movie I should produce, which production I should work with

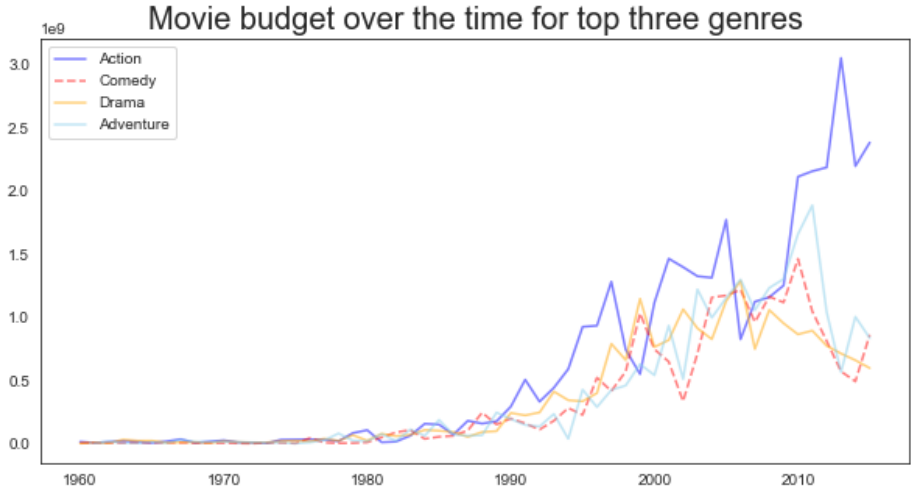
* I will use df2 (full dataset) for this analysis

In [16]:
```
# analyze the top most genre for last 15 years
movies1 = df1.groupby('release_year')['genres'].value_counts().reset_index(name = 'counts'
).copy()
movies1.set_index('genres',inplace =True)
movies1[movies1["release_year"].between(2000,2015)].groupby(['release_year'])['counts'].id
xmax()
movies1[movies1["release_year"].between(2010,2015)]
movies1 = movies1.reset_index()
#print(movies1)

fig,ax = plt.subplots(figsize=(10,5))
x = movies1[movies1['genres']=='Action']['release_year']
y = movies1[movies1['genres']=='Action']['counts']
x1 = movies1[movies1['genres']=='Comedy']['release_year']
y1 = movies1[movies1['genres']=='Comedy']['counts']
x2 = movies1[movies1['genres']=='Drama']['release_year']
y2 = movies1[movies1['genres']=='Drama']['counts']
x3 = movies1[movies1['genres']=='Adventure']['release_year']
y3 = movies1[movies1['genres']=='Adventure']['counts']
plt.title('Movie Popularity over the time for top three genres',fontsize = 20)
lines = plt.plot(x, y,'b',x1,y1,'r--',x2,y2,'orange',x3,y3,'skyblue',label = 'Popularity',
alpha=0.5,)
plt.legend(lines[:4], ['Action', 'Comedy', 'Drama', 'Adventure']);


movies2 = df1.groupby(['release_year','genres']).sum()#.reset_index(name = 'sum').copy()
movies2 = movies2.reset_index()
#print(movies2)
fig,ax = plt.subplots(figsize=(10,5))
x = movies2[movies2['genres']=='Action']['release_year']
y = movies2[movies2['genres']=='Action']['profit']
x1 = movies2[movies2['genres']=='Comedy']['release_year']
y1 = movies2[movies2['genres']=='Comedy']['profit']
x2 = movies2[movies2['genres']=='Drama']['release_year']
y2 = movies2[movies2['genres']=='Drama']['profit']
x3 = movies2[movies2['genres']=='Adventure']['release_year']
y3 = movies2[movies2['genres']=='Adventure']['profit']
plt.title('Movie Profit over the time for top three genres',fontsize = 20)
```

```
lines = plt.plot(x, y,'b',x1,y1,'r--',x2,y2,'orange',x3,y3,'skyblue',label = 'profit', alp
ha=0.5,)
plt.legend(lines[:4], ['Action', 'Comedy', 'Drama', 'Adventure']);



fig,ax = plt.subplots(figsize=(10,5))
x = movies2[movies2['genres']=='Action']['release_year']
y = movies2[movies2['genres']=='Action']['budget']
x1 = movies2[movies2['genres']=='Comedy']['release_year']
y1 = movies2[movies2['genres']=='Comedy']['budget']
x2 = movies2[movies2['genres']=='Drama']['release_year']
y2 = movies2[movies2['genres']=='Drama']['budget']
x3 = movies2[movies2['genres']=='Adventure']['release_year']
y3 = movies2[movies2['genres']=='Adventure']['budget']
plt.title('Movie budget over the time for top three genres',fontsize = 20)
lines = plt.plot(x, y,'b',x1,y1,'r--',x2,y2,'orange',x3,y3,'skyblue',label = 'profit', alp
ha=0.5,)
#plt.legend()
plt.legend(lines[:4], ['Action', 'Comedy', 'Drama', 'Adventure']);
```



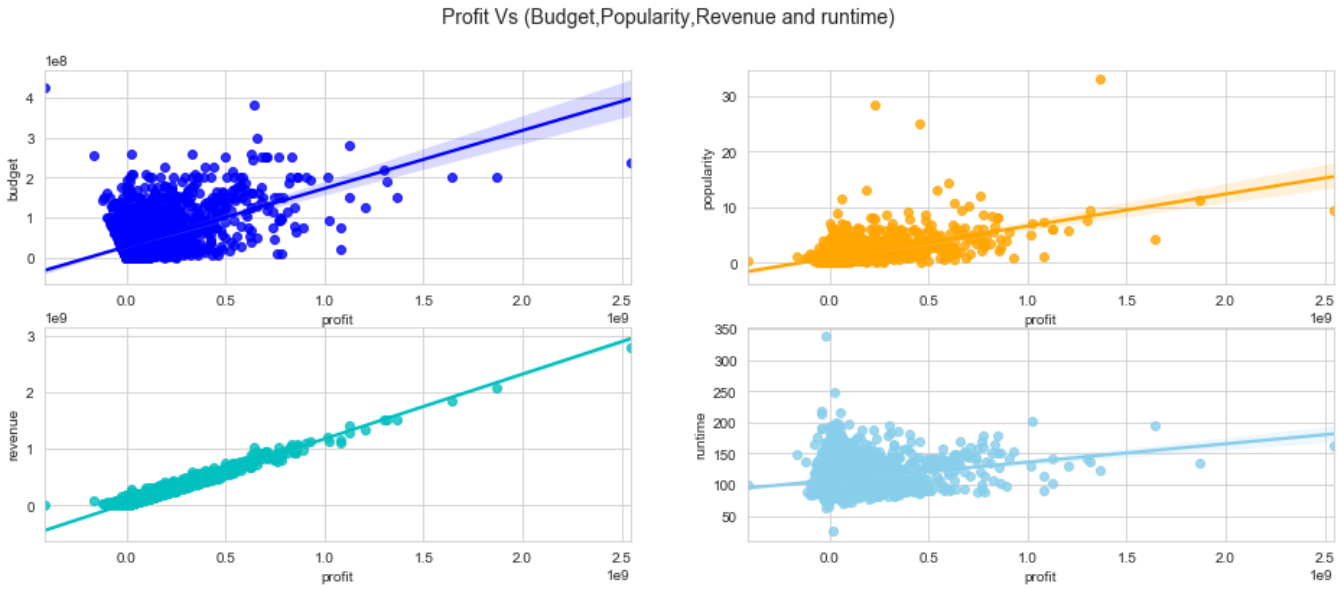




Conclusions:
1. ***Most Popular***
a) Drama
b) Comedy
2. ***Most Profitable***
a) Action
b) Adventure
3. ***High Budget***
a) Action
b) Adventure

In [17]:
```
# Analyze relation between key variables
sns.set_style("whitegrid")
fig, axes = plt.subplots(2,2,figsize = (16,6))
fig.suptitle("Profit Vs (Budget,Popularity,Revenue and runtime)",fontsize=14)

sns.regplot(x=df1['profit'], y=df1['budget'],color='b',ax=axes[0][0])
sns.regplot(x=df1['profit'], y=df1['popularity'],color='orange',ax=axes[0][1])
sns.regplot(x=df1['profit'], y=df1['revenue'],color='c',ax=axes[1][0])
sns.regplot(x=df1['profit'], y=df1['runtime'],color='skyblue',ax=axes[1][1])
```

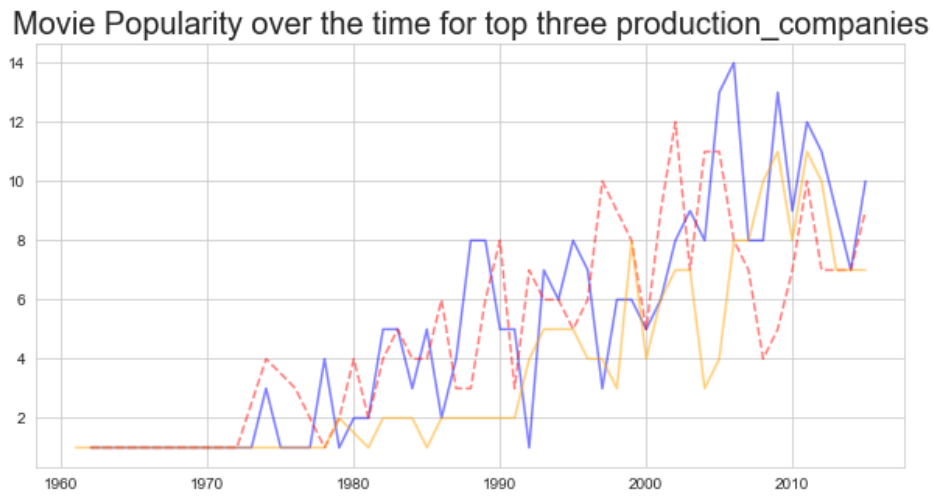
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fafc2730310>
```



There is Positive Correlation between Revenue and all four parameters and below are the key take aways: 1. The revenue of the movie is highly dependent on Budget and Revenue of the movie which can also be explained by the formulae Profit = Budget - Revenue
2. Even though the correleation is positive for popularity and runtime, profit cannot be solely determined using these two parameters

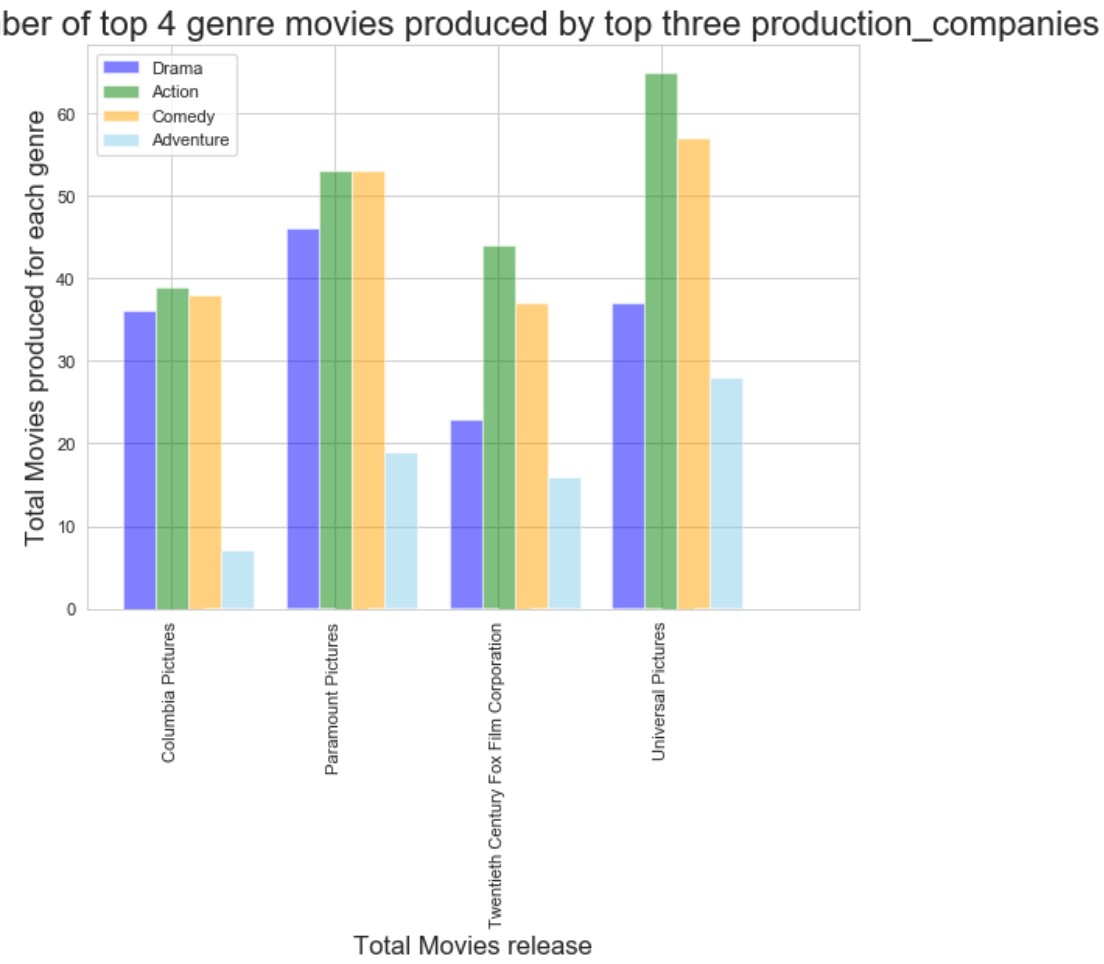
In [18]:
```
pc = df1.groupby('release_year')['production_companies'].value_counts().reset_index(name =
'counts').copy()
pc.set_index('production_companies',inplace =True)
pc[pc["release_year"].between(2000,2015)].groupby(['release_year'])['counts'].idxmax()
pc[pc["release_year"].between(2010,2015)]
pc = pc.reset_index()
#print(pc)

sns.set_style("whitegrid")
fig,ax = plt.subplots(figsize=(10,5))
x = pc[pc['production_companies']=='Universal Pictures']['release_year']
y = pc[pc['production_companies']=='Universal Pictures']['counts']
x1 = pc[pc['production_companies']=='Paramount Pictures']['release_year']
y1 = pc[pc['production_companies']=='Paramount Pictures']['counts']
x2 = pc[pc['production_companies']=='Columbia Pictures']['release_year']
y2 = pc[pc['production_companies']=='Columbia Pictures']['counts']

plt.title('Movie Popularity over the time for top three production_companies',fontsize = 2
0)
plt.plot(x, y,'b',x1,y1,'r--',x2,y2,'orange', alpha=0.5,)
```

```
[<matplotlib.lines.Line2D at 0x7fafc2973790>,
 <matplotlib.lines.Line2D at 0x7fafc2985bd0>,
 <matplotlib.lines.Line2D at 0x7fafc2985cd0>]
```



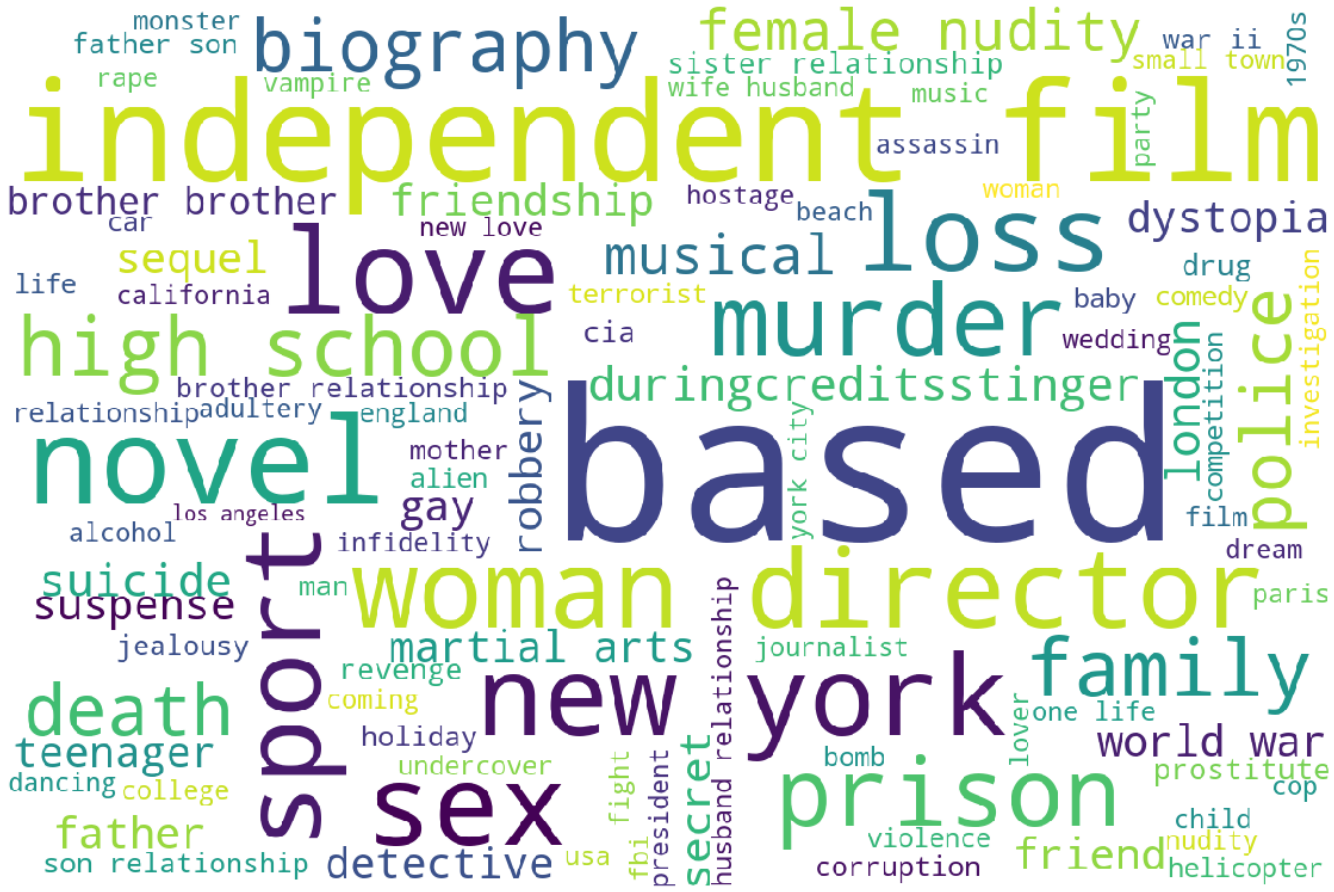
Now we know the genres and production houses with maximum movies produced, let's plot a bar graph to analyze the trend of the top 4 genres produced by top 4 production houses

In [19]:
```
popular_genre = ['Drama','Action','Comedy', 'Adventure']
popular_pc = ['Universal Pictures','Paramount Pictures','Columbia Pictures','Twentieth Century Fox Film Corporation']
pcgenre = df1[df1['genres'].isin(popular_genre) & df1['production_companies'].isin(popular_pc)].groupby('genres')['production_companies'].value_counts().reset_index(name = 'counts').copy()
pcgenre_new = pd.pivot_table(pcgenre,values='counts', index= ['production_companies'],columns='genres').reset_index().copy()
pcgenre.head()
#leng = len(popular_genre) + 1
#print(leng)

#pos = list(range(len(pcgenre_new['Adventure'])))
pos = list(range(len(popular_genre)))
#print(pos)
width=0.2
fig = plt.figure(num=None, figsize=(8, 6), dpi=80, facecolor='w', edgecolor='k')
ax = fig.add_subplot(1, 1, 1)
#plot the first genre
plt.bar(pos, pcgenre_new['Drama'], width,alpha=0.5,color='b',label=pcgenre_new['production_companies'][0])
#plot the second genre
plt.bar([p + width for p in pos],pcgenre_new['Action'], width,alpha=0.5,color='g',label=pcgenre_new['production_companies'][1])
#plot the third genre
plt.bar([p + width*2 for p in pos],pcgenre_new['Comedy'], width,alpha=0.5,color='orange',label=pcgenre_new['production_companies'][1])
#plot the fourth genre
plt.bar([p + width*3 for p in pos],pcgenre_new['Adventure'], width,alpha=0.5,color='skyblue',label=pcgenre_new['production_companies'][1])
#plot the gap between the bar plots
plt.bar([p + width*4 for p in pos], width,alpha=0.5,color='',label=pcgenre_new['production_companies'][1])

plt.title('Total number of top 4 genre movies produced by top three production_companies', fontsize = 20)

#print(pcgenre_new['production_companies'])

# Set the y axis label
#ax.set_ylabel('Genre Count')
plt.xlabel('Total Movies release' , fontsize = 15)
plt.ylabel('Total Movies produced for each genre' , fontsize = 15)
# Set the position of the x ticks
ax.set_xticks([p + width for p in pos])
# Set the labels for the x ticks
ax.set_xticklabels(pcgenre_new['production_companies'],rotation=90)
#plt.xticks(x + width/2, pcgenre_new['production_companies'],rotation=90)
plt.legend(['Drama','Action','Comedy', 'Adventure'], loc='upper left')
plt.show()
```



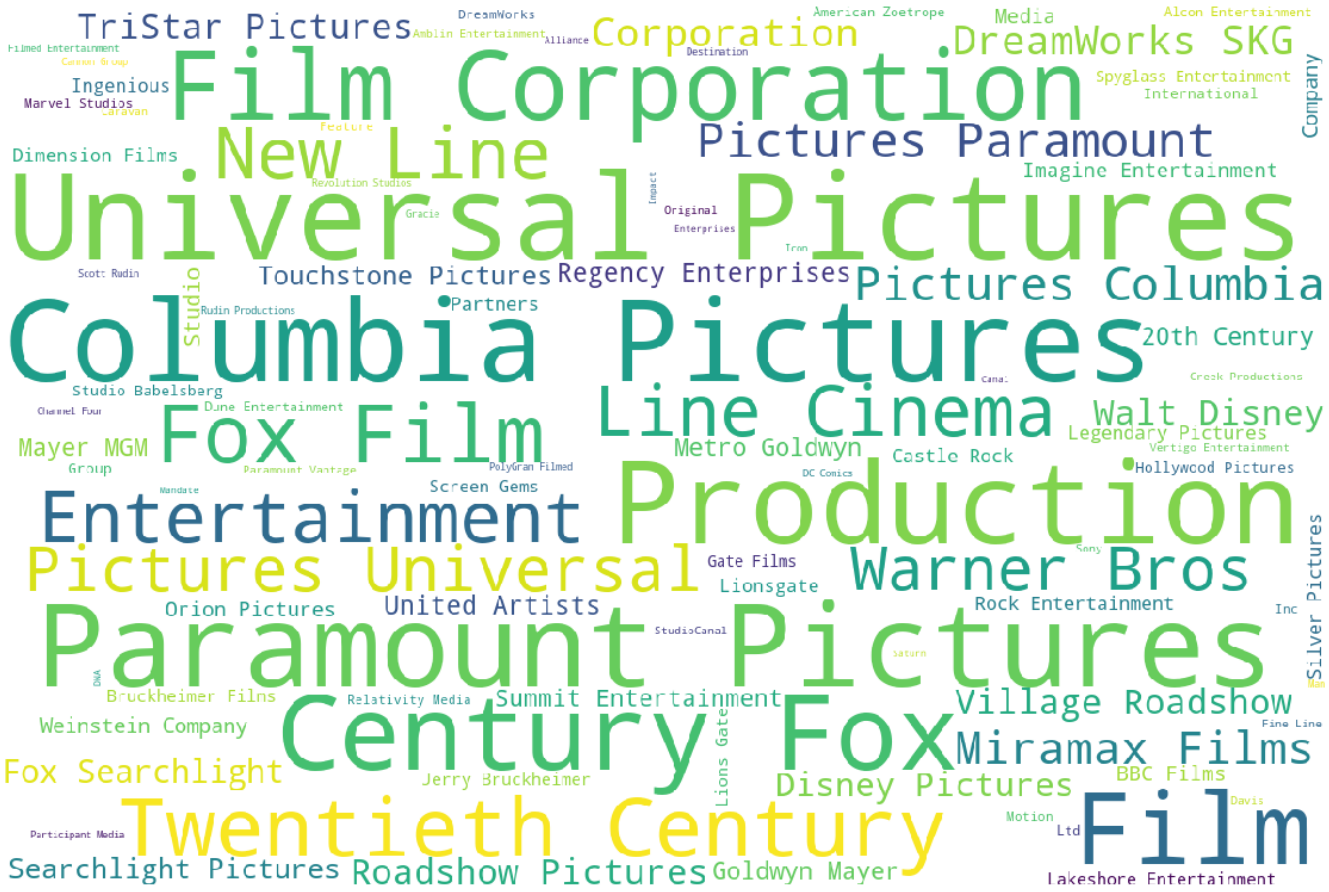
Conlcusion:
1. Universal produced maximum number of movies.
2. All four production houses produced maximum Action movies, then Comedy, then Drama and lowest number of Adventure out of top four genre produced in the last 55 years.

In [20]:
```
# Continue to explore the data to address your additional research
#   questions. Add more headers as needed if you have more questions to
#   investigate.

wordcount = WordCloud(width=1200, height=800,background_color="white", max_words=100,stopw
ords=STOPWORDS)
plt.figure( figsize=(20,20) );
popular_genre = ['Drama','Action','Comedy']
imp_words = wordcount.generate(df2.loc[df2['genres'].isin(popular_genre), 'keywords'].str.
cat(sep='|'));
plt.imshow(imp_words);
plt.axis("off");
#plt.imshow(wordcount);
plt.show();
```



In [21]:
```
# Continue to explore the data to address your additional research
#   questions. Add more headers as needed if you have more questions to
#   investigate.

wordcount = WordCloud(width=1200, height=800,background_color="white", max_words=100,stopw
ords=STOPWORDS)
plt.figure( figsize=(20,20) );
popular_genre = ['Drama','Action','Comedy']
imp_words = wordcount.generate(df1.loc[df2['genres'].isin(popular_genre), 'production_comp
anies'].str.cat(sep='\n'));
plt.imshow(imp_words);
plt.axis("off");
#plt.imshow(wordcount);
plt.show
```

```
<function matplotlib.pyplot.show(*args, **kw)>
```



Conclusions

Based on the graphs created, below is the analysis:
1. Action is the most profitable Genre
2. Adventure Genre movies are the most expensive movies.
3. Top highlighted production houses are - 'Universal Pictures', 'Paramount Pictures' and 'Columbia Pictures'
4. Below are the few keywords I would like to add to my movie:
- women director
- independent film
- novel based
- movie with locations like new Yoek and London
5. The budget of Action and Adventure movies are higher than Drama and Crime

One of the recommendations: - The maximum number of Drama and Comedy Genre movies are produced by the competitors and their budget is less than Action/Adventure and ranked after them for profit, hence I would like to create Drama (or Comedy) genre as my first movie.

Tip: Once you are satisfied with your work, you should save a copy of the report in HTML or PDF form via the **File** > **Download as** submenu. Before exporting your report, check over it to make sure that the flow of the report is complete. You should probably remove all of the "Tip" quotes like this one so that the presentation is as tidy as possible. Congratulations!

In [22]:
```
#from subprocess import call
#call(['python', '-m', 'nbconvert', 'Investigate_TMDb_Dataset_Pankaj_Pant_08282020.ipynb']
)
```