PROJECT 7TH

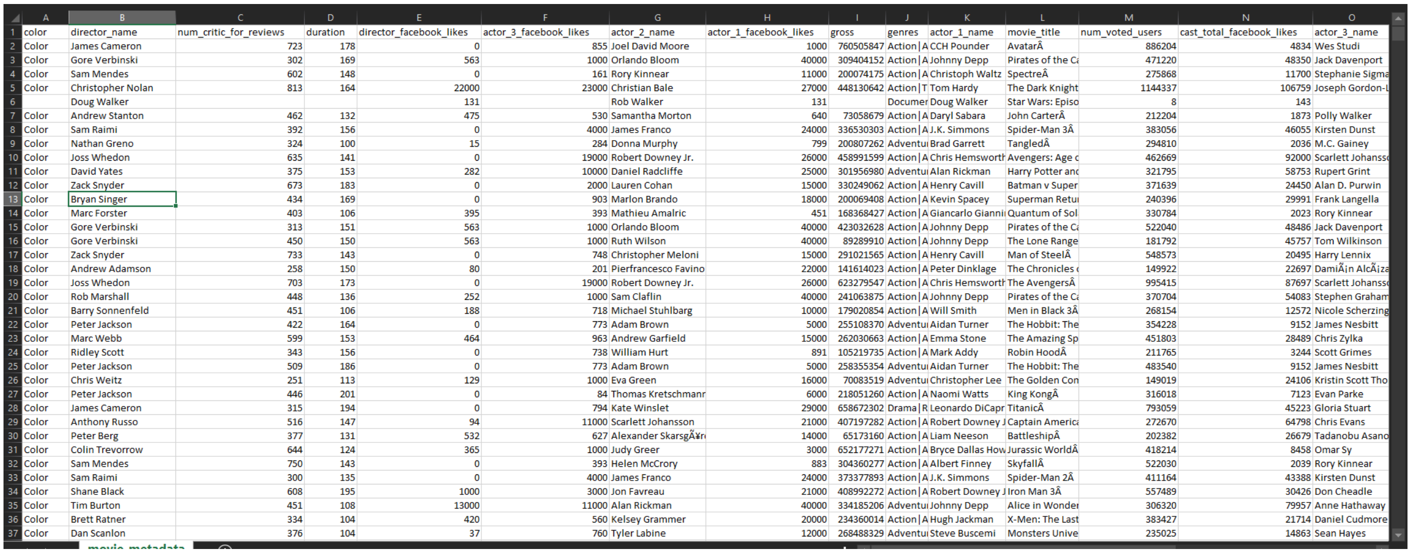
- ○ Ahmed Ali Aftouh Metwally

Topic:

The database we chose to do is the Movies database. Basically, it contains all information about different types of movie genres that will later be described briefly. In this document, all information will be provided and they serve the purpose of understanding the world of the movie and what is most requested and liked by the community, as well as, which actors are reconsidered the best and how many movies they made in each year.

Dataset explanation:

The database has a lot of columns they wont fit into one screen so I will be explaining in 2 screenshots.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	color	director_name	num_critic_for_reviews	duration	director_facebook_likes	actor_3_facebook_likes	actor_2_name	actor_1_facebook_likes	gross	genres	actor_1_name	movie_title	num_voted_users	cast_total_facebook_likes	actor_3_name
2	Color	James Cameron	723	178	0	855	Joel David Moore	1000	760505847	Action|A	CCH Pounder	AvatarÂ	886204	4834	Wes Studi
3	Color	Gore Verbinski	302	169	563	1000	Orlando Bloom	40000	309404152	Action|A	Johnny Depp	Pirates of the Ca	471220	48350	Jack Davenport
4	Color	Sam Mendes	602	148	0	161	Rory Kinnear	11000	200074175	Action|A	Christoph Waltz	SpectreÂ	275868	11700	Stephanie Sigma
5	Color	Christopher Nolan	813	164	22000	23000	Christian Bale	27000	448130642	Action|T	Tom Hardy	The Dark Knight	1144337	106759	Joseph Gordon-L
6		Doug Walker			131		Rob Walker	131		Documer	Doug Walker	Star Wars: Episo	8	143	
7	Color	Andrew Stanton	462	132	475	530	Samantha Morton	640	73058679	Action|A	Daryl Sabara	John CarterÂ	212204	1873	Polly Walker
8	Color	Sam Raimi	392	156	0	4000	James Franco	24000	336530303	Action|A	J.K. Simmons	Spider-Man 3Â	383056	46055	Kirsten Dunst
9	Color	Nathan Greno	324	100	15	284	Donna Murphy	799	200807262	Adventu	Brad Garrett	TangledÂ	294810	2036	M.C. Gainey
10	Color	Joss Whedon	635	141	0	19000	Robert Downey Jr.	26000	458991599	Action|A	Chris Hemsworth	Avengers: Age c	462669	92000	Scarlett Johanss
11	Color	David Yates	375	153	282	10000	Daniel Radcliffe	25000	301956980	Adventu	Alan Rickman	Harry Potter anc	321795	58753	Rupert Grint
12	Color	Zack Snyder	673	183	0	2000	Lauren Cohan	15000	330249062	Action|A	Henry Cavill	Batman v Super	371639	24450	Alan D. Purwin
13	Color	Bryan Singer	434	169	0	903	Marlon Brando	18000	200069408	Action|A	Kevin Spacey	Superman Retu	240396	29991	Frank Langella
14	Color	Marc Forster	403	106	395	393	Mathieu Amalric	451	168368427	Action|A	Giancarlo Gianni	Quantum of Sol	330784	2023	Rory Kinnear
15	Color	Gore Verbinski	313	151	563	1000	Orlando Bloom	40000	423032628	Action|A	Johnny Depp	Pirates of the Ca	522040	48486	Jack Davenport
16	Color	Gore Verbinski	450	150	563	1000	Ruth Wilson	40000	89289910	Action|A	Johnny Depp	The Lone Range	181792	45757	Tom Wilkinson
17	Color	Zack Snyder	733	143	0	748	Christopher Meloni	15000	291021565	Action|A	Henry Cavill	Man of SteelÂ	548573	20495	Harry Lennix
18	Color	Andrew Adamson	258	150	80	201	Pierfrancesco Favino	22000	141614023	Action|A	Peter Dinklage	The Chronicles c	149922	22697	DamiÃ¡n AlcÃ¡za
19	Color	Joss Whedon	703	173	0	19000	Robert Downey Jr.	26000	623279547	Action|A	Chris Hemsworth	The AvengersÂ	995415	87697	Scarlett Johanss
20	Color	Rob Marshall	448	136	252	1000	Sam Claflin	40000	241063875	Action|A	Johnny Depp	Pirates of the Ca	370704	54083	Stephen Graham
21	Color	Barry Sonnenfeld	451	106	188	718	Michael Stuhlbarg	10000	179020854	Action|A	Will Smith	Men in Black 3Â	268154	12572	Nicole Scherzing
22	Color	Peter Jackson	422	164	0	773	Adam Brown	5000	255108370	Adventu	Aidan Turner	The Hobbit: The	354228	9152	James Nesbitt
23	Color	Marc Webb	599	153	464	963	Andrew Garfield	15000	262030663	Action|A	Emma Stone	The Amazing Sp	451803	28489	Chris Zylka
24	Color	Ridley Scott	343	156	0	738	William Hurt	891	105219735	Action|A	Mark Addy	Robin HoodÂ	211765	3244	Scott Grimes
25	Color	Peter Jackson	509	186	0	773	Adam Brown	5000	258355354	Adventu	Aidan Turner	The Hobbit: The	483540	9152	James Nesbitt
26	Color	Chris Weitz	251	113	129	1000	Eva Green	16000	70083519	Adventu	Christopher Lee	The Golden Con	149019	24106	Kristin Scott Tho
27	Color	Peter Jackson	446	201	0	84	Thomas Kretschmanr	6000	218051260	Action|A	Naomi Watts	King KongÂ	316018	7123	Evan Parke
28	Color	James Cameron	315	194	0	794	Kate Winslet	29000	658672302	Drama|R	Leonardo DiCapr	TitanicÂ	793059	45223	Gloria Stuart
29	Color	Anthony Russo	516	147	94	11000	Scarlett Johansson	21000	407197282	Action|A	Robert Downey J	Captain Americ	272670	64798	Chris Evans
30	Color	Peter Berg	377	131	532	627	Alexander SkarsgÃ¥r	14000	65173160	Action|A	Liam Neeson	BattleshipÂ	202382	26679	Tadanobu Asano
31	Color	Colin Trevorrow	644	124	365	1000	Judy Greer	3000	652177271	Action|A	Bryce Dallas How	Jurassic WorldÂ	418214	8458	Omar Sy
32	Color	Sam Mendes	750	143	0	393	Helen McCrory	883	304360277	Action|A	Albert Finney	SkyfallÂ	522030	2039	Rory Kinnear
33	Color	Sam Raimi	300	135	0	4000	James Franco	24000	373377893	Action|A	J.K. Simmons	Spider-Man 2Â	411164	43388	Kirsten Dunst
34	Color	Shane Black	608	195	1000	3000	Jon Favreau	21000	408992272	Action|A	Robert Downey J	Iron Man 3Â	557489	30426	Don Cheadle
35	Color	Tim Burton	451	108	13000	11000	Alan Rickman	40000	334185206	Adventu	Johnny Depp	Alice in Wonder	306320	79957	Anne Hathaway
36	Color	Brett Ratner	334	104	420	560	Kelsey Grammer	20000	234360014	Action|A	Hugh Jackman	X-Men: The Last	383427	21714	Daniel Cudmore
37	Color	Dan Scanlon	376	104	37	760	Tyler Labine	12000	268488329	Adventu	Steve Buscemi	Monsters Unive	235025	14863	Sean Hayes

Of course, every movie has a director, and here are the names of the directors for each movie (director_name column is in screenshot 2), and for each movie, there are critic reviews (num_critic_for_reviews) where professional publishers write their reviews and due to many magazines and blog websites and many different platforms, there are many reviews.

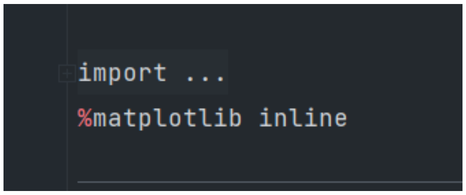
Each movie has a specific duration in minutes(duration). Each director of course has a Facebook page/profile with likes, which explains column (director_facebook_likes) .

Each movie could have more than one main actor and it is repeated in the database as actor_name _1, actor_name _2 and actor_name _3 with their Facebook posted likes the director mentioned before(actor_1_facebook_likes),(actor_1_facebook_likes),(actor_1_facebook_likes).

the movie (the_movie_title) is the name of the movie obviously, besides it the (num_voted_users) which is people who voted for this movie amongst other movies.

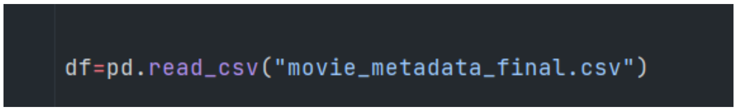
The cast_total_facebook_likes from its name is the total likes on Facebook for each cast combined because one movie could have 10s of cast members

Code cell explanation:



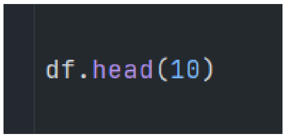
```
import ...
%matplotlib inline
```

- The importe packages that helps us to make analysis and visualization for the datasets



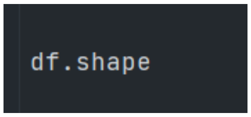
```
df=pd.read_csv("movie_metadata_final.csv")
```

- Read the CSV or EXCIL or any other types file from pc



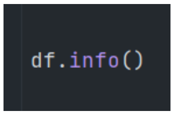
```
df.head(10)
```

- Prints the first 10 rows of the datafram



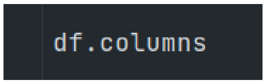
```
df.shape
```

- Return the dimension(rows number,columns numbers) of the dataframe



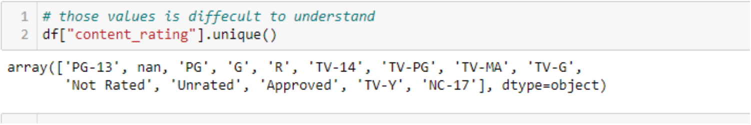
```
df.info()
```

- This method prints information about a dataframe including the index, dtype and columns, non-null values and memory usage.



```
df.columns
```

- It prints the names of all columns of dataframe

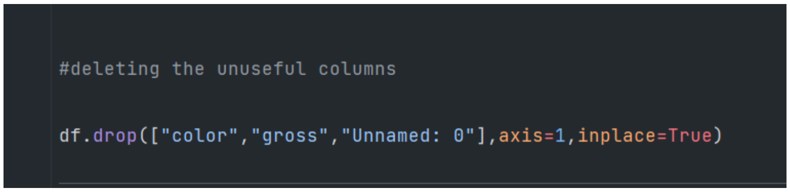


```
# those values is diffecult to understand
df["content_rating"].unique()
```

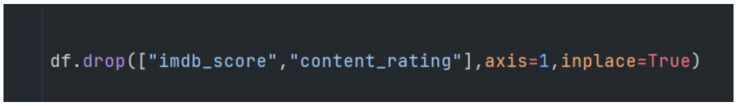
```
array(['PG-13', nan, 'PG', 'G', 'R', 'TV-14', 'TV-PG', 'TV-MA', 'TV-G',
       'Not Rated', 'Unrated', 'Approved', 'TV-Y', 'NC-17'], dtype=object)
```

- this commands prints the unique values in the content_rating columns

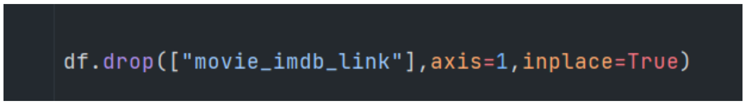
Data cleaning



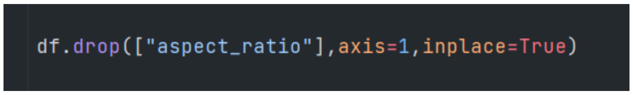
```
#deleting the unuseful columns

df.drop(["color","gross","Unnamed: 0"],axis=1,inplace=True)
```



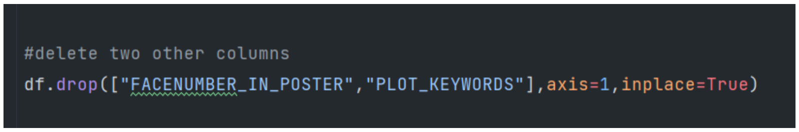
```
df.drop(["imdb_score","content_rating"],axis=1,inplace=True)
```



```
df.drop(["movie_imdb_link"],axis=1,inplace=True)
```

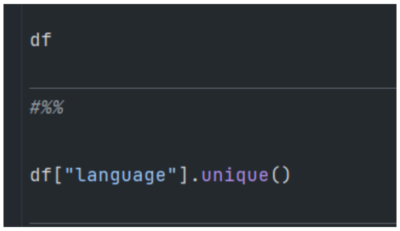


```
df.drop(["aspect_ratio"],axis=1,inplace=True)
```



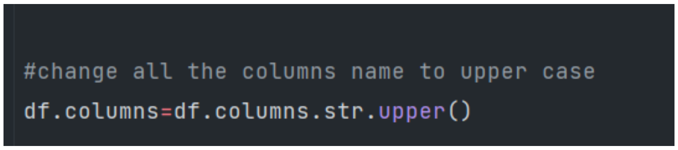
```
#delete two other columns
df.drop(["FACENUMBER_IN_POSTER","PLOT_KEYWORDS"],axis=1,inplace=True)
```

- We are deleting the useless columns that we don’t understand their values (color, gross,unnamed: 0,imdb_score,content_rating,movie_imdb_link, aspect_ratio, facenumber_in_poster,plot_keywords) forever.



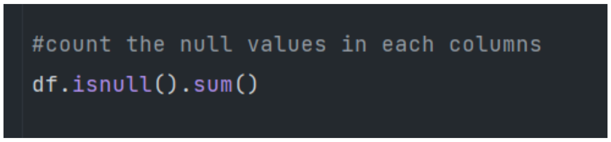
```
df

#%%

df["language"].unique()
```

- This command returns the unique spoken languges of the movies



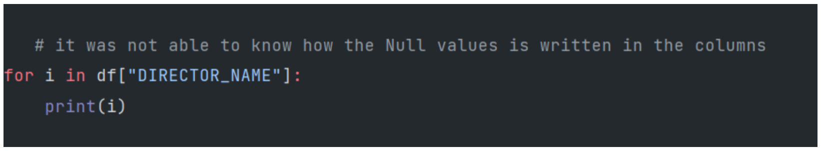
```
#change all the columns name to upper case
df.columns=df.columns.str.upper()
```

- Change all the columns name to upper case



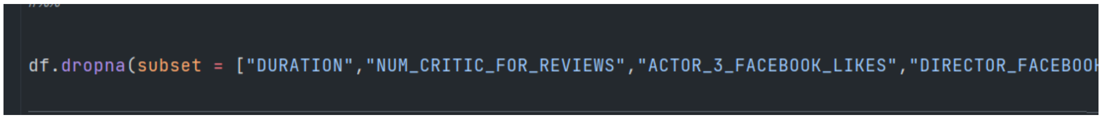
```
#count the null values in each columns
df.isnull().sum()
```

- It returns the sum of all null values in each column.



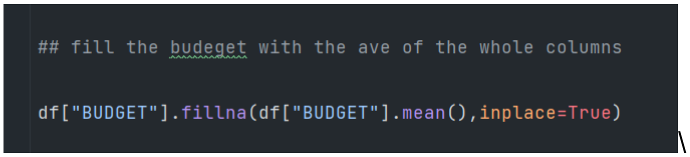
```
  # it was not able to know how the Null values is written in the columns
for i in df["DIRECTOR_NAME"]:
   print(i)
```

- We were not able to know how the null values is written in the columns because it has different options like (NULL,Null,null,NaN,Nan,nan) and we tried to use the **Head and Tail** function but we could not to know. So, we forced to print all the columns values to know how null is written.



```
#%%

df.dropna(subset = ["DURATION","NUM_CRITIC_FOR_REVIEWS","ACTOR_3_FACEBOOK_LIKES","DIRECTOR_FACEBOO
```

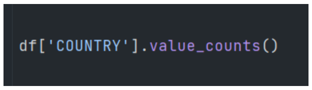
- Drop the rows that contain null values in all the presented columns



```
## fill the budeget with the ave of the whole columns

df["BUDGET"].fillna(df["BUDGET"].mean(),inplace=True)
```

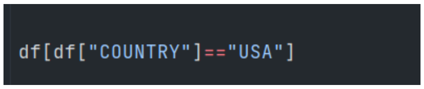
\

- Fill the null values that is founded in the budeget columns with the average value of all the budeget column



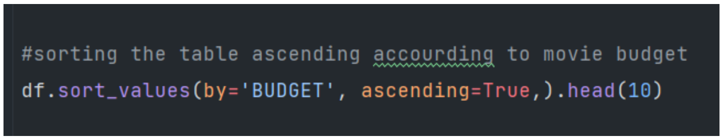
```
df['COUNTRY'].value_counts()
```

- Count how many times the country repeated in the country coulumns



```
df[df["COUNTRY"]=="USA"]
```

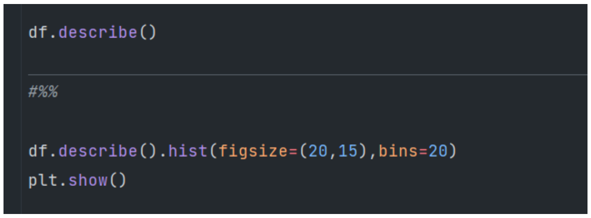
- It returns all the columns values in case the country equal to USA



```
#sorting the table ascending accourding to movie budget
df.sort_values(by='BUDGET', ascending=True,).head(10)
```

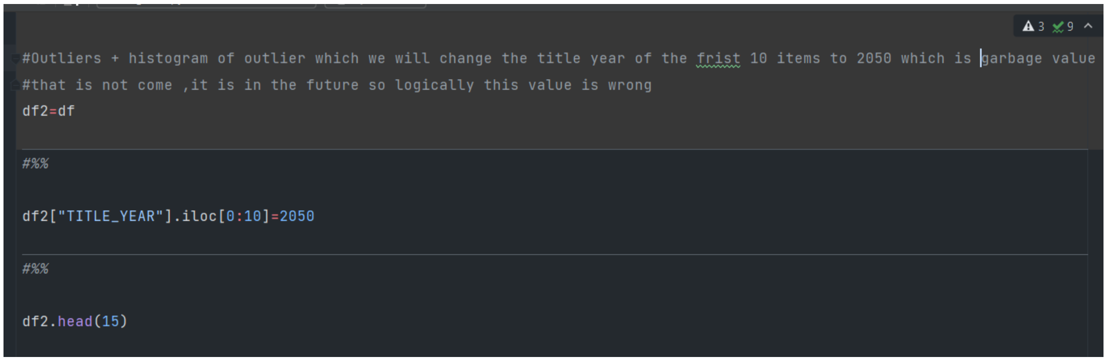
- Sorting the table ascending accourding to the movie budget and print the first 10 values

Analsis



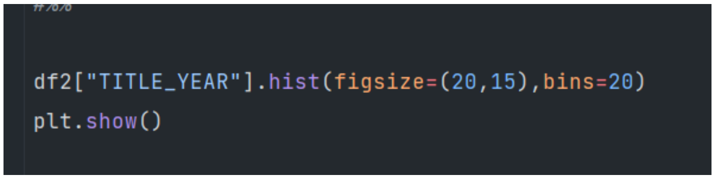
```
df.describe()

#%%

df.describe().hist(figsize=(20,15),bins=20)
plt.show()
```

- Analyzes numeric as well as DataFrame column sets of mixed data types. The output will vary depending on what is provided.
  The whole command In our case is returned the (count,mean,st, min,max,25%,50%,75%) of the data and make visualization (histogram) to the describe function's output.



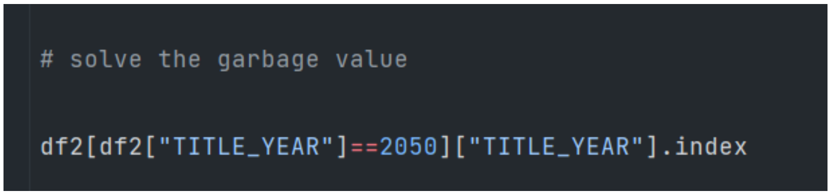
```
#Outliers + histogram of outlier which we will change the title year of the frist 10 items to 2050 which is garbage value
#that is not come ,it is in the future so logically this value is wrong
df2=df

#%%

df2["TITLE_YEAR"].iloc[0:10]=2050

#%%

df2.head(15)
```

- This cell will make a copy(df2=df) from the original dataframe and we changed the first 10 rows in the title_year column of the copied dataframe to 2050. Then ,we printed the frist 15 rows of table.



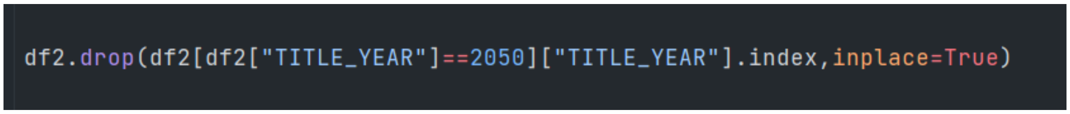
```
#%%

df2["TITLE_YEAR"].hist(figsize=(20,15),bins=20)
plt.show()
```

- We make histogram representation to the outlier that we made.



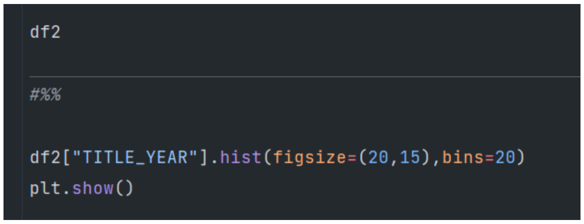
```
# solve the garbage value

df2[df2["TITLE_YEAR"]==2050]["TITLE_YEAR"].index
```

- This command retruns the indexes of the columns in the title_year columns that equal to 2050



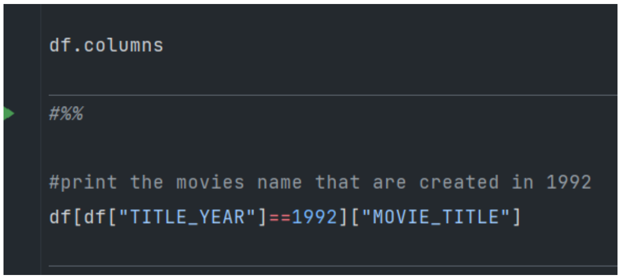
```
df2.drop(df2[df2["TITLE_YEAR"]==2050]["TITLE_YEAR"].index,inplace=True)
```

- This will delet the garbege value in the titel_year columns of df2



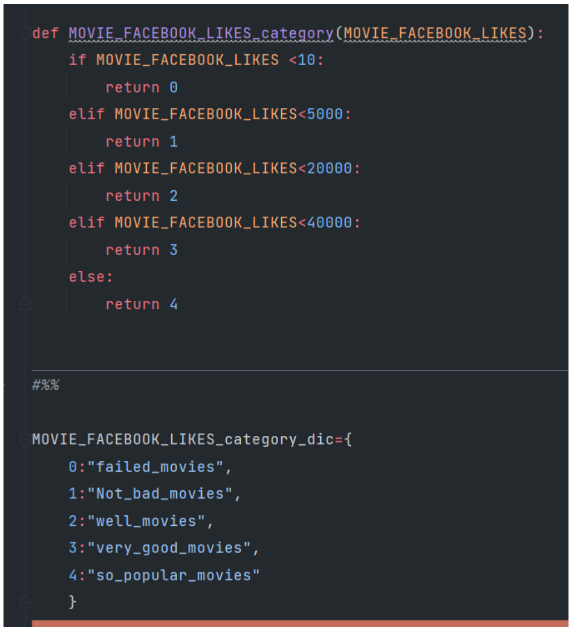
```
df2

#%%

df2["TITLE_YEAR"].hist(figsize=(20,15),bins=20)
plt.show()
```

- We made histogram representation again for the title_year columns after solving the outer value



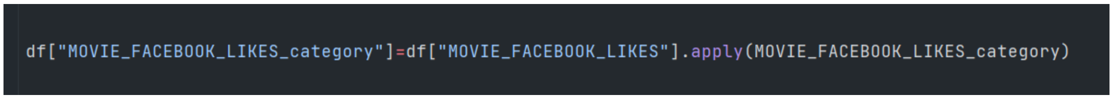
```
df.columns

#%%

#print the movies name that are created in 1992
df[df["TITLE_YEAR"]==1992]["MOVIE_TITLE"]
```

- Returns all movies name tha are created in 1992



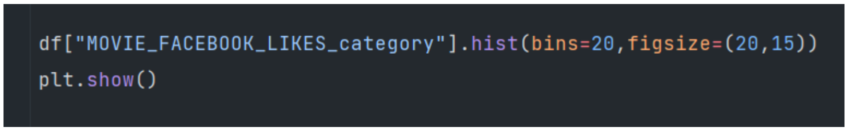
```
def MOVIE_FACEBOOK_LIKES_category(MOVIE_FACEBOOK_LIKES):
    if MOVIE_FACEBOOK_LIKES <10:
        return 0
    elif MOVIE_FACEBOOK_LIKES<5000:
        return 1
    elif MOVIE_FACEBOOK_LIKES<20000:
        return 2
    elif MOVIE_FACEBOOK_LIKES<40000:
        return 3
    else:
        return 4


#%%

MOVIE_FACEBOOK_LIKES_category_dic={
    0:"failed_movies",
    1:"Not_bad_movies",
    2:"well_movies",
    3:"very_good_movies",
    4:"so_popular_movies"
    }
```

- The first cell illustates the function categorize for the movie facebook likes to indicate how much the movies become successful and people liked
- The 2nd cell shows dictionary with keys and values as represented above.

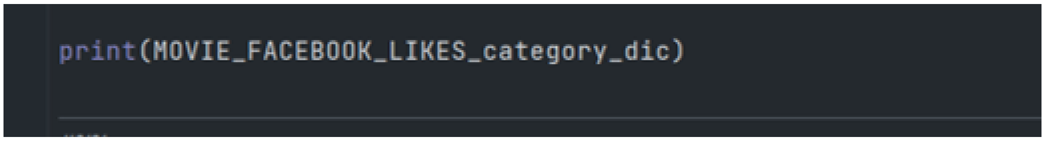


```
df["MOVIE_FACEBOOK_LIKES_category"]=df["MOVIE_FACEBOOK_LIKES"].apply(MOVIE_FACEBOOK_LIKES_category)
```

- We will apply the category function on the movie_facebook_likes column and save it in the dataframe with the name (movie_facebook_likes_category)

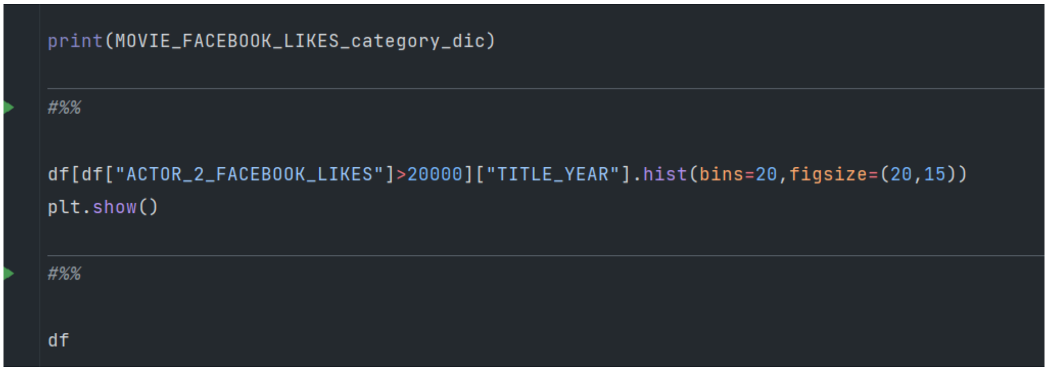


```
df["MOVIE_FACEBOOK_LIKES_category"].hist(bins=20,figsize=(20,15))
plt.show()
```



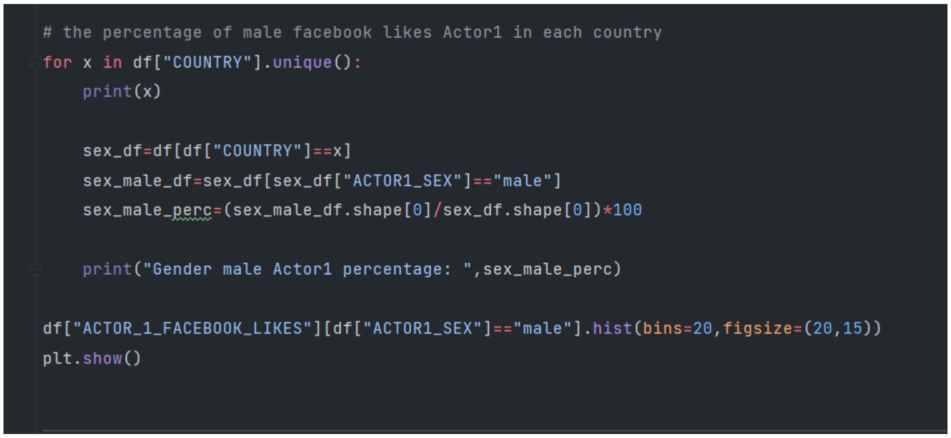
```
print(MOVIE_FACEBOOK_LIKES_category_dic)
```

- We represented the new columns(movie_facebook_likes_category) in the histogram and print the movie_facebook_likes_category_dic under the graph



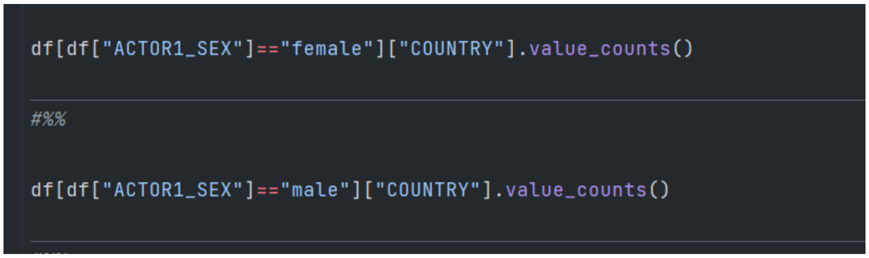
```
print(MOVIE_FACEBOOK_LIKES_category_dic)

#%%

df[df["ACTOR_2_FACEBOOK_LIKES"]>20000]["TITLE_YEAR"].hist(bins=20,figsize=(20,15))
plt.show()

#%%

df
```

- This shows the actor_2_facebook_likes that are greater than 20000 likes with its year



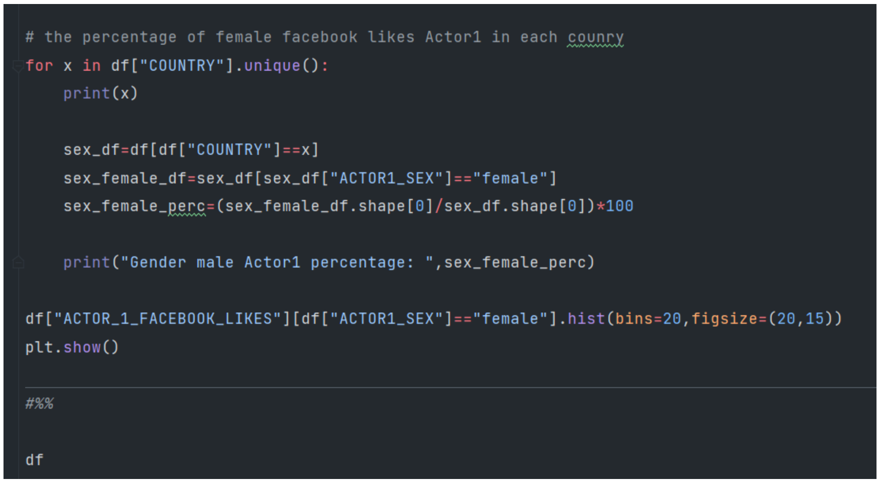
```
# the percentage of male facebook likes Actor1 in each country
for x in df["COUNTRY"].unique():
    print(x)

    sex_df=df[df["COUNTRY"]==x]
    sex_male_df=sex_df[sex_df["ACTOR1_SEX"]=="male"]
    sex_male_perc=(sex_male_df.shape[0]/sex_df.shape[0])*100

    print("Gender male Actor1 percentage: ",sex_male_perc)

df["ACTOR_1_FACEBOOK_LIKES"][df["ACTOR1_SEX"]=="male"].hist(bins=20,figsize=(20,15))
plt.show()
```

- This cell illustrates how we calculated the percentage of the male facebook likes acter1 in each country and print them out.
- We showed the actor1_facebook_likes columns in histogram for only the male categoryt.



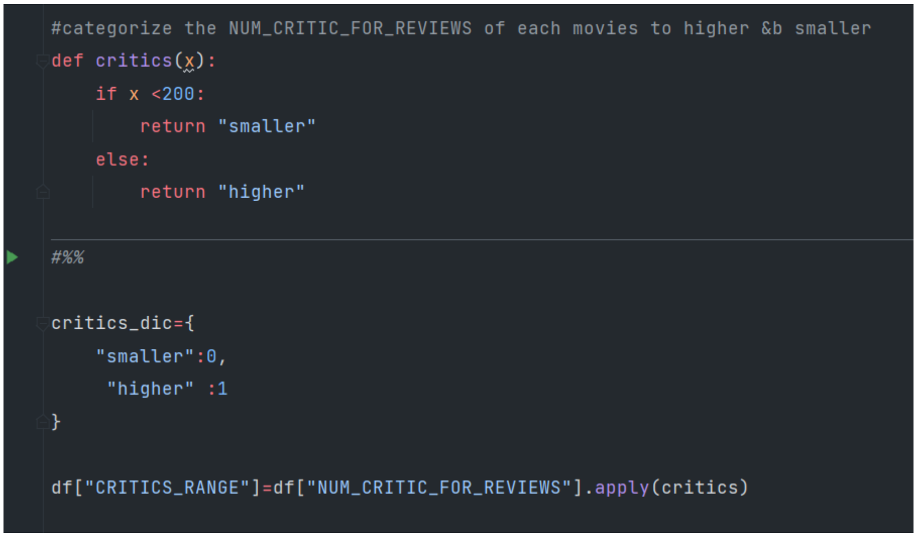
```
df[df["ACTOR1_SEX"]=="female"]["COUNTRY"].value_counts()

#%%

df[df["ACTOR1_SEX"]=="male"]["COUNTRY"].value_counts()
```

- We made this command in order to be sure about the percentage of the male & female facebook likes acter1 because some values were 100% males and vice versa
- So, this will print how much the male & female category repeated for Actor 1



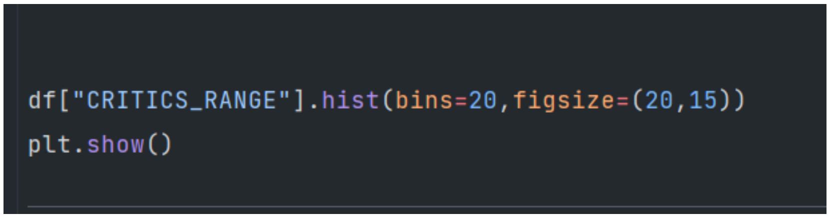
```
# the percentage of female facebook likes Actor1 in each counry
for x in df["COUNTRY"].unique():
    print(x)

    sex_df=df[df["COUNTRY"]==x]
    sex_female_df=sex_df[sex_df["ACTOR1_SEX"]=="female"]
    sex_female_perc=(sex_female_df.shape[0]/sex_df.shape[0])*100

    print("Gender male Actor1 percentage: ",sex_female_perc)

df["ACTOR_1_FACEBOOK_LIKES"][df["ACTOR1_SEX"]=="female"].hist(bins=20,figsize=(20,15))
plt.show()

#%%

df
```

- This cell illustrates how we calculated the percentage of the female facebook likes acter1 in each country and print them out.
- We showed the actor1_facebook_likes columns in histogram for only the female categoryt.

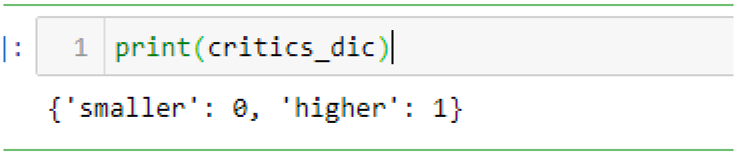


```
#categorize the NUM_CRITIC_FOR_REVIEWS of each movies to higher &b smaller
def critics(x):
    if x <200:
        return "smaller"
    else:
        return "higher"

#%%

critics_dic={
    "smaller":0,
     "higher" :1
}

df["CRITICS_RANGE"]=df["NUM_CRITIC_FOR_REVIEWS"].apply(critics)
```

- The first cell illustates the function categorize for the number of critic reviews to indicate how people react toward each movie
- The 2nd cell shows dictionary with keys and values as represented above
- We applied the category function on the num_ critic_for_reviews column and save it in the dataframe with the name (critics_range)

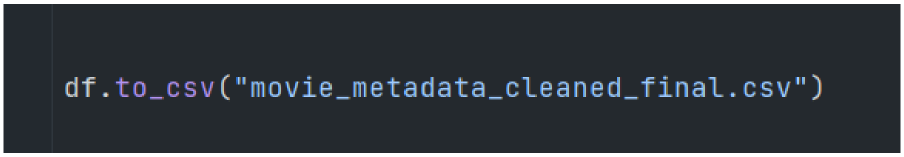


```
df["CRITICS_RANGE"].hist(bins=20,figsize=(20,15))
plt.show()
```



```
print(critics_dic)
{'smaller': 0, 'higher': 1}
```

- We represented the new columns(critics_range) in the histogram and print the critics_dic under the graph



```
df.to_csv("movie_metadata_cleaned_final.csv")
```

- After finishing the cleaning and analyzing of the dataset we used this command to export the cleaned dataset which can be used later in further analysis