Final Project Submission ¶

Please fill out:

- Student name:
- Student pace: self paced / part time / full time
- Scheduled project review date/time:
- Instructor name:
- Blog post URL:

In [1]:
```
# Your code here - remember to use markdown cells for comments as well!

import pandas as pd
import os
```

In [2]:
```
variable = pd.read_csv(r"C:\Users\somep\Documents\Flatiron\dsc-phase-1-project-v2-4\zippedData\bom.movie_gross.csv.gz")
variable.head()
```

Out[2]:

	title	studio	domestic_gross	foreign_gross	year
0	Toy Story 3	BV	415000000.0	652000000	2010
1	Alice in Wonderland (2010)	BV	334200000.0	691300000	2010
2	Harry Potter and the Deathly Hallows Part 1	WB	296000000.0	664300000	2010
3	Inception	WB	292600000.0	535700000	2010
4	Shrek Forever After	P/DW	238700000.0	513900000	2010

In [3]:
```
variable = pd.read_csv(r"C:\Users\somep\Documents\Flatiron\dsc-phase-1-project-v2-4\zippedData\tmdb.movies.csv.gz")
variable.head()
```

Out[3]:

	Unnamed: 0	genre_ids	id	original_language	original_title	popularity	release_date	title	vote_average	vote_count
0	0	[12, 14, 10751]	12444	en	Harry Potter and the Deathly Hallows: Part 1	33.533	2010-11-19	Harry Potter and the Deathly Hallows: Part 1	7.7	10788
1	1	[14, 12, 16, 10751]	10191	en	How to Train Your Dragon	28.734	2010-03-26	How to Train Your Dragon	7.7	7610
2	2	[12, 28, 878]	10138	en	Iron Man 2	28.515	2010-05-07	Iron Man 2	6.8	12368
3	3	[16, 35, 10751]	862	en	Toy Story	28.005	1995-11-22	Toy Story	7.9	10174
4	4	[28, 878, 12]	27205	en	Inception	27.920	2010-07-16	Inception	8.3	22186

In [4]:
```
variable = pd.read_csv(r"C:\Users\somep\Documents\Flatiron\dsc-phase-1-project-v2-4\zippedData\tn.movie_budgets.csv.gz")
variable.head()
```

Out[4]:

	id	release_date	movie	production_budget	domestic_gross	worldwide_gross
0	1	Dec 18, 2009	Avatar	$425,000,000	$760,507,625	$2,776,345,279
1	2	May 20, 2011	Pirates of the Caribbean: On Stranger Tides	$410,600,000	$241,063,875	$1,045,663,875
2	3	Jun 7, 2019	Dark Phoenix	$350,000,000	$42,762,350	$149,762,350
3	4	May 1, 2015	Avengers: Age of Ultron	$330,600,000	$459,005,868	$1,403,013,963
4	5	Dec 15, 2017	Star Wars Ep. VIII: The Last Jedi	$317,000,000	$620,181,382	$1,316,721,747

In [5]:
```
data=pd.read_table(r"C:\Users\somep\Documents\Flatiron\dsc-phase-1-project-v2-4\zippedData\rt.movie_info.tsv.gz")
data.head()
```

Out[5]:

	id	synopsis	rating	genre	director	writer	theater_date	dvd_date	currency	box_office	runtime	studio
0	1	This gritty, fast-paced, and innovative police...	R	Action and Adventure|Classics|Drama	William Friedkin	Ernest Tidyman	Oct 9, 1971	Sep 25, 2001	NaN	NaN	104 minutes	NaN
1	3	New York City, not-too-distant-future: Eric Pa...	R	Drama|Science Fiction and Fantasy	David Cronenberg	David Cronenberg|Don DeLillo	Aug 17, 2012	Jan 1, 2013	$	600,000	108 minutes	Entertainment One
2	5	Illeana Douglas delivers a superb performance ...	R	Drama|Musical and Performing Arts	Allison Anders	Allison Anders	Sep 13, 1996	Apr 18, 2000	NaN	NaN	116 minutes	NaN
3	6	Michael Douglas runs afoul of a treacherous su...	R	Drama|Mystery and Suspense	Barry Levinson	Paul Attanasio|Michael Crichton	Dec 9, 1994	Aug 27, 1997	NaN	NaN	128 minutes	NaN
4	7	NaN	NR	Drama|Romance	Rodney Bennett	Giles Cooper	NaN	NaN	NaN	NaN	200 minutes	NaN

In [6]:
```
data=pd.read_table(r"C:\Users\somep\Documents\Flatiron\dsc-phase-1-project-v2-4\zippedData\rt.reviews.tsv.gz")
data
```

```
---------------------------------------------------------------------------
UnicodeDecodeError                        Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_332\1902765872.py in <module>
----> 1 data=pd.read_table(r"C:\Users\somep\Documents\Flatiron\dsc-phase-1-project-v2-4\zippedData\rt.reviews.tsv.gz")
      2 data

~\anaconda3\lib\site-packages\pandas\util\_decorators.py in wrapper(*args, **kwargs)
    309                     stacklevel=stacklevel,
    310                 )
--> 311             return func(*args, **kwargs)
    312
    313         return wrapper

~\anaconda3\lib\site-packages\pandas\io\parsers\readers.py in read_table(filepath_or_buffer, sep, delimiter, header, names, index_col, usecols, squeeze, prefix, mangle_dupe_cols, dtype, engine, converters, true_values, false_values, skipinitialspace, skiprows, skipfooter, nrows, na_values, keep_default_na, na_filter, verbose, skip_blank_lines, parse_dates, infer_datetime_format, keep_date_col, date_parser, dayfirst, cache_dates, iterator, chunksize, compression, thousands, decimal, lineterminator, quotechar, quoting, doublequote, escapechar, comment, encoding, encoding_errors, dialect, error_bad_lines, warn_bad_lines, on_bad_lines, delim_whitespace, low_memory, memory_map, float_precision, storage_options)
    775     kwds.update(kwds_defaults)
    776
--> 777     return _read(filepath_or_buffer, kwds)
    778
    779

~\anaconda3\lib\site-packages\pandas\io\parsers\readers.py in _read(filepath_or_buffer, kwds)
    579
    580     with parser:
--> 581         return parser.read(nrows)
    582
    583

~\anaconda3\lib\site-packages\pandas\io\parsers\readers.py in read(self, nrows)
   1251             nrows = validate_integer("nrows", nrows)
   1252             try:
-> 1253                 index, columns, col_dict = self._engine.read(nrows)
   1254             except Exception:
   1255                 self.close()

~\anaconda3\lib\site-packages\pandas\io\parsers\c_parser_wrapper.py in read(self, nrows)
    223         try:
    224             if self.low_memory:
--> 225                 chunks = self._reader.read_low_memory(nrows)
    226                 # destructive to chunks
    227                 data = _concatenate_chunks(chunks)

~\anaconda3\lib\site-packages\pandas\_libs\parsers.pyx in pandas._libs.parsers.TextReader.read_low_memory()

~\anaconda3\lib\site-packages\pandas\_libs\parsers.pyx in pandas._libs.parsers.TextReader._read_rows()

~\anaconda3\lib\site-packages\pandas\_libs\parsers.pyx in pandas._libs.parsers.TextReader._tokenize_rows()

~\anaconda3\lib\site-packages\pandas\_libs\parsers.pyx in pandas._libs.parsers.raise_parser_error()

UnicodeDecodeError: 'utf-8' codec can't decode byte 0xa0 in position 6558: invalid start byte
```

In [7]:
```
import pandas as pd
import numpy as np
import sqlite3 as sql
```

In [8]:
```
con = sql.connect(r"C:\Users\somep\Documents\Flatiron\dsc-phase-1-project-v2-4\zippedData\im.db\im.db")
```

In [9]:
```
database = "im.db"
connection = sql.connect(database)
```

In [10]:
```
query = '''SELECT * FROM im.db'''
```

In [11]:
```
schema_df = pd.read_sql("""

SELECT *
FROM sqlite_master

""", con)

schema_df
```

Out[11]:

	type	name	tbl_name	rootpage	sql
0	table	movie_basics	movie_basics	2	CREATE TABLE "movie_basics" (\n"movie_id" TEXT...
1	table	directors	directors	3	CREATE TABLE "directors" (\n"movie_id" TEXT,\n...
2	table	known_for	known_for	4	CREATE TABLE "known_for" (\n"person_id" TEXT,\...
3	table	movie_akas	movie_akas	5	CREATE TABLE "movie_akas" (\n"movie_id" TEXT,\...
4	table	movie_ratings	movie_ratings	6	CREATE TABLE "movie_ratings" (\n"movie_id" TEX...
5	table	persons	persons	7	CREATE TABLE "persons" (\n"person_id" TEXT,\n ...
6	table	principals	principals	8	CREATE TABLE "principals" (\n"movie_id" TEXT,\...
7	table	writers	writers	9	CREATE TABLE "writers" (\n"movie_id" TEXT,\n ...

In [12]:
```
print(schema_df['sql'].iloc[0])
```

```
CREATE TABLE "movie_basics" (
"movie_id" TEXT,
  "primary_title" TEXT,
  "original_title" TEXT,
  "start_year" INTEGER,
  "runtime_minutes" REAL,
  "genres" TEXT
)
```

In [13]:
```
#import sqlite3
#conn = sqlite3.connect('im.db')
#c = conn.cursor()

#c.execute("""create table emp(movie_basics,directors,known_for,movie_akas,movie_ratings,persons,principals,writers)""")

#conn.commit()
```

In [14]:
```
movie_basics_df = pd.read_sql("""

SELECT *
FROM movie_basics

""", con)

movie_basics_df.head()
```

Out[14]:

	movie_id	primary_title	original_title	start_year	runtime_minutes	genres
0	tt0063540	Sunghursh	Sunghursh	2013	175.0	Action,Crime,Drama
1	tt0066787	One Day Before the Rainy Season	Ashad Ka Ek Din	2019	114.0	Biography,Drama
2	tt0069049	The Other Side of the Wind	The Other Side of the Wind	2018	122.0	Drama
3	tt0069204	Sabse Bada Sukh	Sabse Bada Sukh	2018	NaN	Comedy,Drama
4	tt0100275	The Wandering Soap Opera	La Telenovela Errante	2017	80.0	Comedy,Drama,Fantasy

In [15]:
```
directors_df = pd.read_sql("""

SELECT *
FROM directors

""", con)

directors_df.head()
```

Out[15]:

	movie_id	person_id
0	tt0285252	nm0899854
1	tt0462036	nm1940585
2	tt0835418	nm0151540
3	tt0835418	nm0151540
4	tt0878654	nm0089502

In [16]:
```
known_for_df = pd.read_sql("""

SELECT *
FROM known_for

""", con)

known_for_df.head()
```

Out[16]:

	person_id	movie_id
0	nm0061671	tt0837562
1	nm0061671	tt2398241
2	nm0061671	tt0844471
3	nm0061671	tt0118553
4	nm0061865	tt0896534

In [17]:
```
movie_akas_df = pd.read_sql("""

SELECT *
FROM movie_akas

""", con)

movie_akas_df.head()
```

Out[17]:

	movie_id	ordering	title	region	language	types	attributes	is_original_title
0	tt0369610	10	Джурасик свят	BG	bg	None	None	0.0
1	tt0369610	11	Jurashikku warudo	JP	None	imdbDisplay	None	0.0
2	tt0369610	12	Jurassic World: O Mundo dos Dinossauros	BR	None	imdbDisplay	None	0.0
3	tt0369610	13	O Mundo dos Dinossauros	BR	None	None	short title	0.0
4	tt0369610	14	Jurassic World	FR	None	imdbDisplay	None	0.0

In [18]:
```
movie_ratings_df = pd.read_sql("""

SELECT *
FROM movie_ratings

""", con)

movie_ratings_df.head()
```

Out[18]:

	movie_id	averagerating	numvotes
0	tt10356526	8.3	31
1	tt10384606	8.9	559
2	tt1042974	6.4	20
3	tt1043726	4.2	50352
4	tt1060240	6.5	21

In [19]:
```
persons_df = pd.read_sql("""

SELECT *
FROM persons

""", con)

persons_df.head()
```

Out[19]:

	person_id	primary_name	birth_year	death_year	primary_profession
0	nm0061671	Mary Ellen Bauder	NaN	NaN	miscellaneous,production_manager,producer
1	nm0061865	Joseph Bauer	NaN	NaN	composer,music_department,sound_department
2	nm0062070	Bruce Baum	NaN	NaN	miscellaneous,actor,writer
3	nm0062195	Axel Baumann	NaN	NaN	camera_department,cinematographer,art_department
4	nm0062798	Pete Baxter	NaN	NaN	production_designer,art_department,set_decorator

In [20]:
```
principals_df = pd.read_sql("""

SELECT *
FROM principals

""", con)

principals_df.head()
```

Out[20]:

	movie_id	ordering	person_id	category	job	characters
0	tt0111414	1	nm0246005	actor	None	["The Man"]
1	tt0111414	2	nm0398271	director	None	None
2	tt0111414	3	nm3739909	producer	producer	None
3	tt0323808	10	nm0059247	editor	None	None
4	tt0323808	1	nm3579312	actress	None	["Beth Boothby"]

In [21]:
```
writers_df = pd.read_sql("""

SELECT *
FROM writers

""", con)

writers_df.head()
```

Out[21]:

	movie_id	person_id
0	tt0285252	nm0899854
1	tt0438973	nm0175726
2	tt0438973	nm1802864
3	tt0462036	nm1940585
4	tt0835418	nm0310087

In [22]:
```
movie_basics_directors = pd.merge(movie_basics_df, directors_df)
movie_basics_directors.columns
```

Out[22]:
```
Index(['movie_id', 'primary_title', 'original_title', 'start_year',
       'runtime_minutes', 'genres', 'person_id'],
      dtype='object')
```

In [23]:
```
movie_basics_directors.head()
```

Out[23]:

	movie_id	primary_title	original_title	start_year	runtime_minutes	genres	person_id
0	tt0063540	Sunghursh	Sunghursh	2013	175.0	Action,Crime,Drama	nm0712540
1	tt0063540	Sunghursh	Sunghursh	2013	175.0	Action,Crime,Drama	nm0712540
2	tt0063540	Sunghursh	Sunghursh	2013	175.0	Action,Crime,Drama	nm0712540
3	tt0063540	Sunghursh	Sunghursh	2013	175.0	Action,Crime,Drama	nm0712540
4	tt0066787	One Day Before the Rainy Season	Ashad Ka Ek Din	2019	114.0	Biography,Drama	nm0002411

In [24]:
```
MOVIEDIRECTORSRATINGS = pd.merge(movie_basics_directors, movie_ratings_df)
MOVIEDIRECTORSRATINGS.columns
```

Out[24]:
```
Index(['movie_id', 'primary_title', 'original_title', 'start_year',
       'runtime_minutes', 'genres', 'person_id', 'averagerating', 'numvotes'],
      dtype='object')
```

In [25]:
```
MOVIEDIRECTORSRATINGS.head()
```

Out[25]:

	movie_id	primary_title	original_title	start_year	runtime_minutes	genres	person_id	averagerating	numvotes
0	tt0063540	Sunghursh	Sunghursh	2013	175.0	Action,Crime,Drama	nm0712540	7.0	77
1	tt0063540	Sunghursh	Sunghursh	2013	175.0	Action,Crime,Drama	nm0712540	7.0	77
2	tt0063540	Sunghursh	Sunghursh	2013	175.0	Action,Crime,Drama	nm0712540	7.0	77
3	tt0063540	Sunghursh	Sunghursh	2013	175.0	Action,Crime,Drama	nm0712540	7.0	77
4	tt0066787	One Day Before the Rainy Season	Ashad Ka Ek Din	2019	114.0	Biography,Drama	nm0002411	7.2	43

In [26]:
```
MOVIEDIRECTORSRATINGS2 = pd.merge(MOVIEDIRECTORSRATINGS, known_for_df)
MOVIEDIRECTORSRATINGS2.columns
```

Out[26]:
```
Index(['movie_id', 'primary_title', 'original_title', 'start_year',
       'runtime_minutes', 'genres', 'person_id', 'averagerating', 'numvotes'],
      dtype='object')
```

In [27]:
```
MOVIEDIRECTORSRATINGS2.head()
```

Out[27]:

	movie_id	primary_title	original_title	start_year	runtime_minutes	genres	person_id	averagerating	numvotes
0	tt0100275	The Wandering Soap Opera	La Telenovela Errante	2017	80.0	Comedy,Drama,Fantasy	nm0765384	6.5	119
1	tt0100275	The Wandering Soap Opera	La Telenovela Errante	2017	80.0	Comedy,Drama,Fantasy	nm0765384	6.5	119
2	tt0100275	The Wandering Soap Opera	La Telenovela Errante	2017	80.0	Comedy,Drama,Fantasy	nm0749914	6.5	119
3	tt0100275	The Wandering Soap Opera	La Telenovela Errante	2017	80.0	Comedy,Drama,Fantasy	nm0749914	6.5	119
4	tt0112502	Bigfoot	Bigfoot	2017	NaN	Horror,Thriller	nm6883878	4.1	32

In [28]:
```
MOVIEDIRECTORSRATINGS3 = pd.merge(MOVIEDIRECTORSRATINGS2, movie_akas_df)
MOVIEDIRECTORSRATINGS3.columns
```

Out[28]:
```
Index(['movie_id', 'primary_title', 'original_title', 'start_year',
       'runtime_minutes', 'genres', 'person_id', 'averagerating', 'numvotes',
       'ordering', 'title', 'region', 'language', 'types', 'attributes',
       'is_original_title'],
      dtype='object')
```

In [29]:
```
MOVIEDIRECTORSRATINGS3.head()
```

Out[29]:

	movie_id	primary_title	original_title	start_year	runtime_minutes	genres	person_id	averagerating	numvotes	ordering	title	region
0	tt0100275	The Wandering Soap Opera	La Telenovela Errante	2017	80.0	Comedy,Drama,Fantasy	nm0765384	6.5	119	1	La Telenovela Errante	None
1	tt0100275	The Wandering Soap Opera	La Telenovela Errante	2017	80.0	Comedy,Drama,Fantasy	nm0765384	6.5	119	2	The Wandering Soap Opera	XWW
2	tt0100275	The Wandering Soap Opera	La Telenovela Errante	2017	80.0	Comedy,Drama,Fantasy	nm0765384	6.5	119	3	La Telenovela Errante	CL
3	tt0100275	The Wandering Soap Opera	La Telenovela Errante	2017	80.0	Comedy,Drama,Fantasy	nm0765384	6.5	119	4	La novela errante	CL
4	tt0100275	The Wandering Soap Opera	La Telenovela Errante	2017	80.0	Comedy,Drama,Fantasy	nm0765384	6.5	119	5	Objazdowa opera mydlana	PL

In [30]:
```
MOVIEDIRECTORSRATINGS4 = pd.merge(MOVIEDIRECTORSRATINGS3, persons_df)
MOVIEDIRECTORSRATINGS4.columns
```

Out[30]:
```
Index(['movie_id', 'primary_title', 'original_title', 'start_year',
       'runtime_minutes', 'genres', 'person_id', 'averagerating', 'numvotes',
       'ordering', 'title', 'region', 'language', 'types', 'attributes',
       'is_original_title', 'primary_name', 'birth_year', 'death_year',
       'primary_profession'],
      dtype='object')
```

In [31]:
```
MOVIEDIRECTORSRATINGS4.head()
```

Out[31]:

	movie_id	primary_title	original_title	start_year	runtime_minutes	genres	person_id	averagerating	numvotes	ordering	title	region
0	tt0100275	The Wandering Soap Opera	La Telenovela Errante	2017	80.0	Comedy,Drama,Fantasy	nm0765384	6.5	119	1	La Telenovela Errante	None
1	tt0100275	The Wandering Soap Opera	La Telenovela Errante	2017	80.0	Comedy,Drama,Fantasy	nm0765384	6.5	119	2	The Wandering Soap Opera	XWW
2	tt0100275	The Wandering Soap Opera	La Telenovela Errante	2017	80.0	Comedy,Drama,Fantasy	nm0765384	6.5	119	3	La Telenovela Errante	CL
3	tt0100275	The Wandering Soap Opera	La Telenovela Errante	2017	80.0	Comedy,Drama,Fantasy	nm0765384	6.5	119	4	La novela errante	CL
4	tt0100275	The Wandering Soap Opera	La Telenovela Errante	2017	80.0	Comedy,Drama,Fantasy	nm0765384	6.5	119	5	Objazdowa opera mydlana	PL

In [32]:
```
MOVIEDIRECTORSRATINGS5 = pd.merge(MOVIEDIRECTORSRATINGS4, principals_df)
MOVIEDIRECTORSRATINGS5.columns
```

Out[32]:
```
Index(['movie_id', 'primary_title', 'original_title', 'start_year',
       'runtime_minutes', 'genres', 'person_id', 'averagerating', 'numvotes',
       'ordering', 'title', 'region', 'language', 'types', 'attributes',
       'is_original_title', 'primary_name', 'birth_year', 'death_year',
       'primary_profession', 'category', 'job', 'characters'],
      dtype='object')
```

In [33]:
```
MOVIEDIRECTORSRATINGS5.head()
```

Out[33]:

	movie_id	primary_title	original_title	start_year	runtime_minutes	genres	person_id	averagerating	numvotes	ordering	...	types	att
0	tt1928329	Lines of Wellington	Linhas de Wellington	2012	151.0	Drama,History,War	nm0765384	6.2	1235	5	...	imdbDisplay	
1	tt0100275	The Wandering Soap Opera	La Telenovela Errante	2017	80.0	Comedy,Drama,Fantasy	nm0749914	6.5	119	5	...	imdbDisplay	
2	tt0100275	The Wandering Soap Opera	La Telenovela Errante	2017	80.0	Comedy,Drama,Fantasy	nm0749914	6.5	119	5	...	imdbDisplay	
3	tt1236371	Mysteries of Lisbon	Mistérios de Lisboa	2010	272.0	Drama,Mystery,Romance	nm0749914	7.5	2928	5	...	imdbDisplay	
4	tt1236371	Mysteries of Lisbon	Mistérios de Lisboa	2010	272.0	Drama,Mystery,Romance	nm0749914	7.5	2928	5	...	imdbDisplay	

5 rows × 23 columns

In [34]:
```
MOVIEDIRECTORSRATINGS6 = pd.merge(MOVIEDIRECTORSRATINGS5, writers_df)
MOVIEDIRECTORSRATINGS6.columns
```

Out[34]:
```
Index(['movie_id', 'primary_title', 'original_title', 'start_year',
       'runtime_minutes', 'genres', 'person_id', 'averagerating', 'numvotes',
       'ordering', 'title', 'region', 'language', 'types', 'attributes',
       'is_original_title', 'primary_name', 'birth_year', 'death_year',
       'primary_profession', 'category', 'job', 'characters'],
      dtype='object')
```

In [35]:
```
MOVIEDIRECTORSRATINGS6.head()
```

Out[35]:

	movie_id	primary_title	original_title	start_year	runtime_minutes	genres	person_id	averagerating	numvotes	ordering	...	types
0	tt0100275	The Wandering Soap Opera	La Telenovela Errante	2017	80.0	Comedy,Drama,Fantasy	nm0749914	6.5	119	5	...	imdbDisplay
1	tt0100275	The Wandering Soap Opera	La Telenovela Errante	2017	80.0	Comedy,Drama,Fantasy	nm0749914	6.5	119	5	...	imdbDisplay
2	tt0100275	The Wandering Soap Opera	La Telenovela Errante	2017	80.0	Comedy,Drama,Fantasy	nm0749914	6.5	119	5	...	imdbDisplay
3	tt0100275	The Wandering Soap Opera	La Telenovela Errante	2017	80.0	Comedy,Drama,Fantasy	nm0749914	6.5	119	5	...	imdbDisplay
4	tt0137204	Joe Finds Grace	Joe Finds Grace	2017	83.0	Adventure,Animation,Comedy	nm0365480	8.1	263	1	...	None

5 rows × 23 columns

In [36]:
```
variable1 = pd.read_csv(r"C:\Users\somep\Documents\Flatiron\dsc-phase-1-project-v2-4\zippedData\bom.movie_gross.csv.gz")
variable1.head()
```

Out[36]:

	title	studio	domestic_gross	foreign_gross	year
0	Toy Story 3	BV	415000000.0	652000000	2010
1	Alice in Wonderland (2010)	BV	334200000.0	691300000	2010
2	Harry Potter and the Deathly Hallows Part 1	WB	296000000.0	664300000	2010
3	Inception	WB	292600000.0	535700000	2010
4	Shrek Forever After	P/DW	238700000.0	513900000	2010

In [37]:
```
MOVIEDIRECTORSRATINGS7 = pd.merge(MOVIEDIRECTORSRATINGS6, variable1)
MOVIEDIRECTORSRATINGS7.columns
```

Out[37]:
```
Index(['movie_id', 'primary_title', 'original_title', 'start_year',
       'runtime_minutes', 'genres', 'person_id', 'averagerating', 'numvotes',
       'ordering', 'title', 'region', 'language', 'types', 'attributes',
       'is_original_title', 'primary_name', 'birth_year', 'death_year',
       'primary_profession', 'category', 'job', 'characters', 'studio',
       'domestic_gross', 'foreign_gross', 'year'],
      dtype='object')
```

In [38]:
```
MOVIEDIRECTORSRATINGS7.head()
```

Out[38]:

	movie_id	primary_title	original_title	start_year	runtime_minutes	genres	person_id	averagerating	numvotes	ordering	...	birth_year	dea
0	tt3906082	Mary Shelley	Mary Shelley	2017	120.0	Biography,Drama,History	nm2223783	6.4	8534	5	...	1974.0	
1	tt3906082	Mary Shelley	Mary Shelley	2017	120.0	Biography,Drama,History	nm2223783	6.4	8534	5	...	1974.0	
2	tt0409379	In Secret	In Secret	2013	107.0	Crime,Drama,Thriller	nm0833627	6.2	7045	5	...	NaN	
3	tt0409379	In Secret	In Secret	2013	107.0	Crime,Drama,Thriller	nm0833627	6.2	7045	5	...	NaN	
4	tt0409379	In Secret	In Secret	2013	107.0	Crime,Drama,Thriller	nm0833627	6.2	7045	5	...	NaN	

5 rows × 27 columns

In [39]:
```
variable2 = pd.read_csv(r"C:\Users\somep\Documents\Flatiron\dsc-phase-1-project-v2-4\zippedData\tmdb.movies.csv.gz")
variable2.head()
```

Out[39]:

	Unnamed: 0	genre_ids	id	original_language	original_title	popularity	release_date	title	vote_average	vote_count
0	0	[12, 14, 10751]	12444	en	Harry Potter and the Deathly Hallows: Part 1	33.533	2010-11-19	Harry Potter and the Deathly Hallows: Part 1	7.7	10788
1	1	[14, 12, 16, 10751]	10191	en	How to Train Your Dragon	28.734	2010-03-26	How to Train Your Dragon	7.7	7610
2	2	[12, 28, 878]	10138	en	Iron Man 2	28.515	2010-05-07	Iron Man 2	6.8	12368
3	3	[16, 35, 10751]	862	en	Toy Story	28.005	1995-11-22	Toy Story	7.9	10174
4	4	[28, 878, 12]	27205	en	Inception	27.920	2010-07-16	Inception	8.3	22186

In [40]:
```
MOVIEDIRECTORSRATINGS8 = pd.merge(MOVIEDIRECTORSRATINGS7, variable2)
MOVIEDIRECTORSRATINGS8.columns
```

Out[40]:
```
Index(['movie_id', 'primary_title', 'original_title', 'start_year',
       'runtime_minutes', 'genres', 'person_id', 'averagerating', 'numvotes',
       'ordering', 'title', 'region', 'language', 'types', 'attributes',
       'is_original_title', 'primary_name', 'birth_year', 'death_year',
       'primary_profession', 'category', 'job', 'characters', 'studio',
       'domestic_gross', 'foreign_gross', 'year', 'Unnamed: 0', 'genre_ids',
       'id', 'original_language', 'popularity', 'release_date', 'vote_average',
       'vote_count'],
      dtype='object')
```

In [41]:
```
MOVIEDIRECTORSRATINGS8.head()
```

Out[41]:

	movie_id	primary_title	original_title	start_year	runtime_minutes	genres	person_id	averagerating	numvotes	ordering	...	foreign_gross
0	tt3906082	Mary Shelley	Mary Shelley	2017	120.0	Biography,Drama,History	nm2223783	6.4	8534	5	...	NaN
1	tt3906082	Mary Shelley	Mary Shelley	2017	120.0	Biography,Drama,History	nm2223783	6.4	8534	5	...	NaN
2	tt0409379	In Secret	In Secret	2013	107.0	Crime,Drama,Thriller	nm0833627	6.2	7045	5	...	NaN
3	tt0409379	In Secret	In Secret	2013	107.0	Crime,Drama,Thriller	nm0833627	6.2	7045	5	...	NaN
4	tt0409379	In Secret	In Secret	2013	107.0	Crime,Drama,Thriller	nm0833627	6.2	7045	5	...	NaN

5 rows × 35 columns

In [42]:
```
variable3 = pd.read_csv(r"C:\Users\somep\Documents\Flatiron\dsc-phase-1-project-v2-4\zippedData\tn.movie_budgets.csv.gz")
variable3.head()
```

Out[42]:

	id	release_date	movie	production_budget	domestic_gross	worldwide_gross
0	1	Dec 18, 2009	Avatar	$425,000,000	$760,507,625	$2,776,345,279
1	2	May 20, 2011	Pirates of the Caribbean: On Stranger Tides	$410,600,000	$241,063,875	$1,045,663,875
2	3	Jun 7, 2019	Dark Phoenix	$350,000,000	$42,762,350	$149,762,350
3	4	May 1, 2015	Avengers: Age of Ultron	$330,600,000	$459,005,868	$1,403,013,963
4	5	Dec 15, 2017	Star Wars Ep. VIII: The Last Jedi	$317,000,000	$620,181,382	$1,316,721,747

In [43]:
```
to_concat = [variable3, MOVIEDIRECTORSRATINGS8]
big_df = pd.concat(to_concat)
```

In [44]:
```
big_df.columns
```

Out[44]:
```
Index(['id', 'release_date', 'movie', 'production_budget', 'domestic_gross',
       'worldwide_gross', 'movie_id', 'primary_title', 'original_title',
       'start_year', 'runtime_minutes', 'genres', 'person_id', 'averagerating',
       'numvotes', 'ordering', 'title', 'region', 'language', 'types',
       'attributes', 'is_original_title', 'primary_name', 'birth_year',
       'death_year', 'primary_profession', 'category', 'job', 'characters',
       'studio', 'foreign_gross', 'year', 'Unnamed: 0', 'genre_ids',
       'original_language', 'popularity', 'vote_average', 'vote_count'],
      dtype='object')
```

In [45]:
```
big_df.head()
```

Out[45]:

	id	release_date	movie	production_budget	domestic_gross	worldwide_gross	movie_id	primary_title	original_title	start_year	...	characters	studio	fo
0	1	Dec 18, 2009	Avatar	$425,000,000	$760,507,625	$2,776,345,279	NaN	NaN	NaN	NaN	...	NaN	NaN	
1	2	May 20, 2011	Pirates of the Caribbean: On Stranger Tides	$410,600,000	$241,063,875	$1,045,663,875	NaN	NaN	NaN	NaN	...	NaN	NaN	
2	3	Jun 7, 2019	Dark Phoenix	$350,000,000	$42,762,350	$149,762,350	NaN	NaN	NaN	NaN	...	NaN	NaN	
3	4	May 1, 2015	Avengers: Age of Ultron	$330,600,000	$459,005,868	$1,403,013,963	NaN	NaN	NaN	NaN	...	NaN	NaN	
4	5	Dec 15, 2017	Star Wars Ep. VIII: The Last Jedi	$317,000,000	$620,181,382	$1,316,721,747	NaN	NaN	NaN	NaN	...	NaN	NaN	

5 rows × 38 columns

In [46]:
```
big_df.drop(['movie_id', 'primary_title', 'original_title', 'start_year', 'category', 'job', 'characters', 'person_id', 'Unnamed:
```

Out[46]:

	id	release_date	movie	production_budget	domestic_gross	worldwide_gross	runtime_minutes	genres	averagerating	numvotes	order
0	1	Dec 18, 2009	Avatar	$425,000,000	$760,507,625	$2,776,345,279	NaN	NaN	NaN	NaN	N
1	2	May 20, 2011	Pirates of the Caribbean: On Stranger Tides	$410,600,000	$241,063,875	$1,045,663,875	NaN	NaN	NaN	NaN	N
2	3	Jun 7, 2019	Dark Phoenix	$350,000,000	$42,762,350	$149,762,350	NaN	NaN	NaN	NaN	N
3	4	May 1, 2015	Avengers: Age of Ultron	$330,600,000	$459,005,868	$1,403,013,963	NaN	NaN	NaN	NaN	N
4	5	Dec 15, 2017	Star Wars Ep. VIII: The Last Jedi	$317,000,000	$620,181,382	$1,316,721,747	NaN	NaN	NaN	NaN	N
...	...	...	...	...	...	...	...	...	...	...	
732	460668	2018-04-20	NaN	NaN	48800000.0	NaN	110.0	Comedy,Romance	5.4	39936.0	
733	44826	2011-11-23	NaN	NaN	73900000.0	NaN	80.0	Documentary	7.9	11.0	
734	44826	2011-11-23	NaN	NaN	73900000.0	NaN	80.0	Documentary	7.9	11.0	
735	210522	2012-07-07	NaN	NaN	335000.0	NaN	101.0	Mystery	7.0	23.0	
736	118289	2013-11-01	NaN	NaN	335000.0	NaN	101.0	Mystery	7.0	23.0	

6519 rows × 20 columns

In [48]:
```
import pip
pip.main(["install", "openpyxl"])
```

```
WARNING: pip is being invoked by an old script wrapper. This will fail in a future version of pip.
Please see https://github.com/pypa/pip/issues/5599 (https://github.com/pypa/pip/issues/5599) for advice on fixing the underlyin
g issue.
To avoid this problem you can invoke Python with '-m pip' instead of running pip directly.

Requirement already satisfied: openpyxl in c:\users\somep\anaconda3\lib\site-packages (3.0.10)

Requirement already satisfied: et_xmlfile in c:\users\somep\anaconda3\lib\site-packages (from openpyxl) (1.1.0)
```

Out[48]: 0

In [49]:
```
import pandas as pd
import openpyxl as xls
```

In [51]:
```
big_df.head()
```

Out[51]:

	id	release_date	movie	production_budget	domestic_gross	worldwide_gross	movie_id	primary_title	original_title	start_year	...	characters	studio	fo
0	1	Dec 18, 2009	Avatar	$425,000,000	$760,507,625	$2,776,345,279	NaN	NaN	NaN	NaN	...	NaN	NaN	
1	2	May 20, 2011	Pirates of the Caribbean: On Stranger Tides	$410,600,000	$241,063,875	$1,045,663,875	NaN	NaN	NaN	NaN	...	NaN	NaN	
2	3	Jun 7, 2019	Dark Phoenix	$350,000,000	$42,762,350	$149,762,350	NaN	NaN	NaN	NaN	...	NaN	NaN	
3	4	May 1, 2015	Avengers: Age of Ultron	$330,600,000	$459,005,868	$1,403,013,963	NaN	NaN	NaN	NaN	...	NaN	NaN	
4	5	Dec 15, 2017	Star Wars Ep. VIII: The Last Jedi	$317,000,000	$620,181,382	$1,316,721,747	NaN	NaN	NaN	NaN	...	NaN	NaN	

5 rows × 38 columns

In [52]:
```
excel_file = pd.ExcelWriter("big_df_file.xlsx")
```

In [54]:
```
big_df.to_excel(excel_file)
```

In [55]:
```
excel_file.save()
```

In [81]:
```
import pandas as pd
```

In [85]:
```
df = pd.read_csv(r"C:\Users\somep\Documents\Flatiron\dsc-phase-1-project-v2-4\zippedData\renamed_file.csv")
```

```
---------------------------------------------------------------------------
UnicodeDecodeError                        Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_332\3595523707.py in <module>
----> 1 df = pd.read_csv(r"C:\Users\somep\Documents\Flatiron\dsc-phase-1-project-v2-4\zippedData\renamed_file.csv")

~\anaconda3\lib\site-packages\pandas\util\_decorators.py in wrapper(*args, **kwargs)
    309                     stacklevel=stacklevel,
    310                 )
--> 311             return func(*args, **kwargs)
    312
    313         return wrapper

~\anaconda3\lib\site-packages\pandas\io\parsers\readers.py in read_csv(filepath_or_buffer, sep, delimiter, header, names, index
_col, usecols, squeeze, prefix, mangle_dupe_cols, dtype, engine, converters, true_values, false_values, skipinitialspace, skipr
ows, skipfooter, nrows, na_values, keep_default_na, na_filter, verbose, skip_blank_lines, parse_dates, infer_datetime_format, k
eep_date_col, date_parser, dayfirst, cache_dates, iterator, chunksize, compression, thousands, decimal, lineterminator, quotech
ar, quoting, doublequote, escapechar, comment, encoding, encoding_errors, dialect, error_bad_lines, warn_bad_lines, on_bad_line
s, delim_whitespace, low_memory, memory_map, float_precision, storage_options)
    676     kwds.update(kwds_defaults)
    677
--> 678     return _read(filepath_or_buffer, kwds)
    679
    680

~\anaconda3\lib\site-packages\pandas\io\parsers\readers.py in _read(filepath_or_buffer, kwds)
    573
    574     # Create the parser.
--> 575     parser = TextFileReader(filepath_or_buffer, **kwds)
    576
    577     if chunksize or iterator:

~\anaconda3\lib\site-packages\pandas\io\parsers\readers.py in __init__(self, f, engine, **kwds)
    930
    931         self.handles: IOHandles | None = None
--> 932         self._engine = self._make_engine(f, self.engine)
    933
    934     def close(self):

~\anaconda3\lib\site-packages\pandas\io\parsers\readers.py in _make_engine(self, f, engine)
   1232
   1233         try:
-> 1234             return mapping[engine](f, **self.options)
   1235         except Exception:
   1236             if self.handles is not None:

~\anaconda3\lib\site-packages\pandas\io\parsers\c_parser_wrapper.py in __init__(self, src, **kwds)
     73
     74         kwds["dtype"] = ensure_dtype_objs(kwds.get("dtype", None))
---> 75         self._reader = parsers.TextReader(src, **kwds)
     76
     77         self.unnamed_cols = self._reader.unnamed_cols

~\anaconda3\lib\site-packages\pandas\_libs\parsers.pyx in pandas._libs.parsers.TextReader.__cinit__()

~\anaconda3\lib\site-packages\pandas\_libs\parsers.pyx in pandas._libs.parsers.TextReader._get_header()

~\anaconda3\lib\site-packages\pandas\_libs\parsers.pyx in pandas._libs.parsers.TextReader._tokenize_rows()

~\anaconda3\lib\site-packages\pandas\_libs\parsers.pyx in pandas._libs.parsers.raise_parser_error()

UnicodeDecodeError: 'utf-8' codec can't decode byte 0xc3 in position 69639: invalid continuation byte
```

In [91]:
```
df = pd.read_csv(r"C:\Users\somep\Documents\Flatiron\dsc-phase-1-project-v2-4\zippedData\renamed_file.csv",encoding='latin-1')
```

In [92]:
```
df
```

Out[92]:

	id	release_date	movie	production_budget	domestic_gross	worldwide_gross	runtime	genres	rating	numvotes	...	studio	foreign_g
0	1	18-Dec-09	Avatar	425000000.0	760507625.0	2.776345e+09	NaN	NaN	NaN	NaN	...	NaN	
1	2	20-May-11	Pirates of the Caribbean: On Stranger Tides	410600000.0	241063875.0	1.045664e+09	NaN	NaN	NaN	NaN	...	NaN	
2	3	7-Jun-19	Dark Phoenix	350000000.0	42762350.0	1.497624e+08	NaN	NaN	NaN	NaN	...	NaN	
3	4	1-May-15	Avengers: Age of Ultron	330600000.0	459005868.0	1.403014e+09	NaN	NaN	NaN	NaN	...	NaN	
4	5	15-Dec-17	Star Wars Ep. VIII: The Last Jedi	317000000.0	620181382.0	1.316722e+09	NaN	NaN	NaN	NaN	...	NaN	
...	...	...	...	...	...	...	...	...	...	...	...	...	
5967	6502	10-Jun-17	Earth: One Amazing Day	NaN	81300.0	NaN	95.0	Documentary,Family	7.8	1908.0	...	BBC	116000
5968	6511	18-Aug-17	Patti Cakes	NaN	800000.0	NaN	109.0	Drama,Music	6.7	8581.0	...	FoxS	6820
5969	6512	20-Apr-18	I Feel Pretty	NaN	48800000.0	NaN	110.0	Comedy,Romance	5.4	39936.0	...	STX	457000
5970	6516	23-Nov-11	Hugo	NaN	73900000.0	NaN	80.0	Documentary	7.9	11.0	...	Par.	1119000
5971	6518	7-Jul-12	Diana	NaN	335000.0	NaN	101.0	Mystery	7.0	23.0	...	EOne	214000

5972 rows × 22 columns

In [93]:
```
import pandas as pd
import numpy as np

import matplotlib.pyplot as plt
import seaborn as sns

from scipy import stats
from sklearn.datasets import load_iris
```

In [96]:
```
domestic = pd.read_csv(r"C:\Users\somep\Documents\Flatiron\dsc-phase-1-project-v2-4\zippedData\Highest_Grossing_Domestic.csv",enc
```

In [97]:
```
domestic
```

Out[97]:

	id	release_date	movie	production_budget	domestic_gross	worldwide_gross
0	6	18-Dec-15	Star Wars Ep. VII: The Force Awakens	306000000	936662225	2053311220
1	1	18-Dec-09	Avatar	425000000	760507625	2776345279
2	42	16-Feb-18	Black Panther	200000000	700059566	1348258224
3	7	27-Apr-18	Avengers: Infinity War	300000000	678815482	2048134200
4	43	19-Dec-97	Titanic	200000000	659363944	2208208395
5	34	12-Jun-15	Jurassic World	215000000	652270625	1648854864
6	27	4-May-12	The Avengers	225000000	623279547	1517935897
7	5	15-Dec-17	Star Wars Ep. VIII: The Last Jedi	317000000	620181382	1316721747
8	44	15-Jun-18	Incredibles 2	200000000	608581744	1242520711
9	75	18-Jul-08	The Dark Knight	185000000	533720947	1001996207
10	45	16-Dec-16	Rogue One: A Star Wars Story	200000000	532177324	1049102856
11	135	17-Mar-17	Beauty and the Beast	160000000	504014165	1259199706
12	46	17-Jun-16	Finding Dory	200000000	486295561	1021215193
13	304	19-May-99	Star Wars Ep. I: The Phantom Menace	115000000	474544677	1027044677
14	3465	25-May-77	Star Wars Ep. IV: A New Hope	11000000	460998007	786598007
15	4	1-May-15	Avengers: Age of Ultron	330600000	459005868	1403013963
16	11	20-Jul-12	The Dark Knight Rises	275000000	448139099	1084439099
17	693	19-May-04	Shrek 2	70000000	441226247	937008132
18	3526	11-Jun-82	ET: The Extra-Terrestrial	10500000	435110554	792965326
19	96	8-Mar-19	Captain Marvel	175000000	426525952	1123061550
20	238	22-Nov-13	The Hunger Games: Catching Fire	130000000	424668047	864868047
21	28	7-Jul-06	Pirates of the Caribbean: Dead Manâ s Chest	225000000	423315812	1066215812
22	609	15-Jun-94	The Lion King	79300000	421785283	986214868
23	113	22-Jun-18	Jurassic World: Fallen Kingdom	170000000	417719760	1305772799
24	47	18-Jun-10	Toy Story 3	200000000	415004880	1068879522
25	155	2-Jun-17	Wonder Woman	150000000	412563408	821133378
26	48	3-May-13	Iron Man 3	200000000	408992272	1215392272
27	17	6-May-16	Captain America: Civil War	250000000	408084349	1140069413
28	538	23-Mar-12	The Hunger Games	80000000	408010692	677923379
29	438	20-Dec-17	Jumanji: Welcome to the Jungle	90000000	404508916	964496193
30	219	3-May-02	Spider-Man	139000000	403706375	821706375
31	36	24-Jun-09	Transformers: Revenge of the Fallen	210000000	402111870	836519699
32	156	22-Nov-13	Frozen	150000000	400738009	1272469910
33	825	11-Jun-93	Jurassic Park	63000000	395708305	1038812584
34	49	5-May-17	Guardians of the Galaxy Vol 2	200000000	389813101	862316233
35	261	15-Jul-11	Harry Potter and the Deathly Hallows: Part II	125000000	381193157	1341693157
36	425	30-May-03	Finding Nemo	94000000	380529370	936429370
37	305	19-May-05	Star Wars Ep. III: Revenge of the Sith	115000000	380270577	848998877
38	426	17-Dec-03	The Lord of the Rings: The Return of the King	94000000	377845905	1141403341
39	2486	13-Nov-91	Beauty and the Beast	20000000	376057266	608431132
40	50	30-Jun-04	Spider-Man 2	200000000	373524485	795110670
41	2160	25-Feb-04	The Passion of the Christ	25000000	370782930	622341924
42	626	8-Jul-16	The Secret Life of Pets	75000000	368384330	886750534
43	622	3-Jul-13	Despicable Me 2	76000000	368065385	975216835
44	97	15-Apr-16	The Jungle Book	175000000	364001123	962854547
45	956	12-Feb-16	Deadpool	58000000	363070709	801025593
46	98	19-Jun-15	Inside Out	175000000	356461711	854235992

In [105]:
```
myFreq = domestic['domestic_gross'].value_counts()
myFreq
```

Out[105]:
```
936662225    1
381193157    1
408992272    1
408084349    1
408010692    1
404508916    1
403706375    1
402111870    1
400738009    1
395708305    1
389813101    1
380529370    1
415004880    1
380270577    1
377845905    1
376057266    1
373524485    1
370782930    1
368384330    1
368065385    1
364001123    1
363070709    1
412563408    1
417719760    1
760507625    1
504014165    1
700059566    1
678815482    1
659363944    1
652270625    1
623279547    1
620181382    1
608581744    1
533720947    1
532177324    1
486295561    1
421785283    1
474544677    1
460998007    1
459005868    1
448139099    1
441226247    1
435110554    1
426525952    1
424668047    1
423315812    1
356461711    1
Name: domestic_gross, dtype: int64
```

In [106]:
```
genrebreakdown = pd.read_csv(r"C:\Users\somep\Documents\Flatiron\dsc-phase-1-project-v2-4\zippedData\Genre_Breakdown.csv",encodir
```

In [107]:
```
genrebreakdown
```

Out[107]:

	Code	Date	Title	Unnamed: 3	Domestic Gross	Unnamed: 5	Runtime	Genre
0	5783	25-May-18	Mary Shelley	NaN	109000.0	NaN	120.0	Biography,Drama,History
1	5785	21-Feb-14	In Secret	NaN	444000.0	NaN	107.0	Crime,Drama,Thriller
2	5788	21-Apr-17	Free Fire	NaN	1800000.0	NaN	91.0	Action,Comedy,Crime
3	5790	30-Sep-11	Margaret	NaN	46500.0	NaN	150.0	Drama
4	5791	22-Aug-12	Samsara	NaN	2700000.0	NaN	102.0	Documentary,Music
...	...	...	...	...	...	...	...	...
185	6502	10-Jun-17	Earth: One Amazing Day	NaN	81300.0	NaN	95.0	Documentary,Family
186	6511	18-Aug-17	Patti Cakes	NaN	800000.0	NaN	109.0	Drama,Music
187	6512	20-Apr-18	I Feel Pretty	NaN	48800000.0	NaN	110.0	Comedy,Romance
188	6516	23-Nov-11	Hugo	NaN	73900000.0	NaN	80.0	Documentary
189	6518	7-Jul-12	Diana	NaN	335000.0	NaN	101.0	Mystery

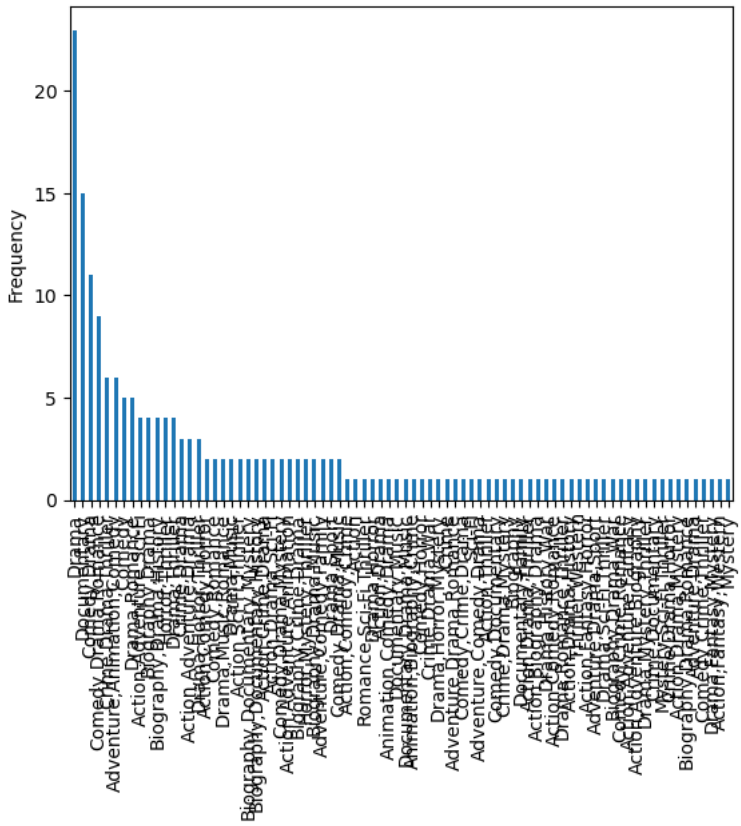
190 rows × 8 columns

In [109]:
```
myFreq = genrebreakdown['Genre'].value_counts()
myFreq
```

Out[109]:
```
Drama                     23
Documentary               15
Comedy,Drama              11
Comedy,Drama,Romance       9
Crime,Drama,Thriller       6
                          ..
Adventure,Drama            1
Comedy,Crime,Thriller      1
Drama,Fantasy,Mystery      1
Action,Fantasy,Western     1
Mystery                    1
Name: Genre, Length: 80, dtype: int64
```

In [110]:
```
myFreq.plot(kind='bar')
plt.ylabel('Frequency')
```

Out[110]:
```
Text(0, 0.5, 'Frequency')
```



In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
```

In [2]:
```
MostExpensive = pd.read_csv(r"C:\Users\somep\Documents\Flatiron\dsc-phase-1-project-v2-4\zippedData\SimpleFiles.csv",encoding='la
```

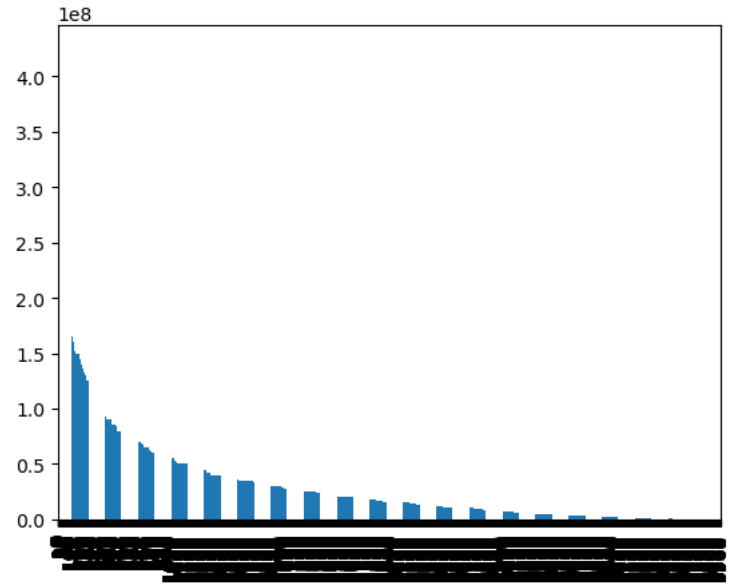
In [5]:
```
MostExpensive[0:10]
```

Out[5]:

	id	release_date	movie	production_budget	domestic_gross	worldwide_gross	Net gain
0	1	18-Dec-09	Avatar	425000000.0	760507625.0	2.776345e+09	2.351345e+09
1	2	20-May-11	Pirates of the Caribbean: On Stranger Tides	410600000.0	241063875.0	1.045664e+09	6.350639e+08
2	3	7-Jun-19	Dark Phoenix	350000000.0	42762350.0	1.497624e+08	-2.002376e+08
3	4	1-May-15	Avengers: Age of Ultron	330600000.0	459005868.0	1.403014e+09	1.072414e+09
4	5	15-Dec-17	Star Wars Ep. VIII: The Last Jedi	317000000.0	620181382.0	1.316722e+09	9.997217e+08
5	6	18-Dec-15	Star Wars Ep. VII: The Force Awakens	306000000.0	936662225.0	2.053311e+09	1.747311e+09
6	7	27-Apr-18	Avengers: Infinity War	300000000.0	678815482.0	2.048134e+09	1.748134e+09
7	8	24-May-07	Pirates of the Caribbean: At Worldâ s End	300000000.0	309420425.0	9.634204e+08	6.634204e+08
8	9	17-Nov-17	Justice League	300000000.0	229024295.0	6.559452e+08	3.559452e+08
9	10	6-Nov-15	Spectre	300000000.0	200074175.0	8.796209e+08	5.796209e+08

In [16]:
```
df = MostExpensive["production_budget"]
```

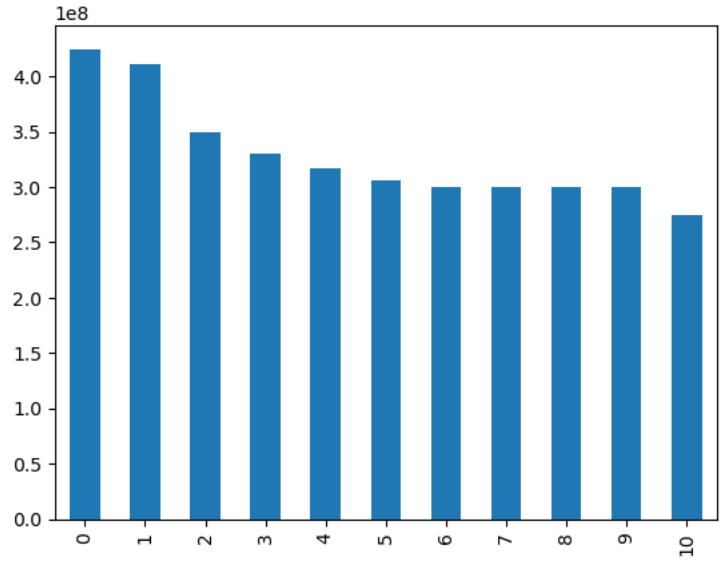
In [20]:
```
budgetplot = df.plot(kind="bar"[0:10])
```



In [21]:
```
df1 = pd.read_csv(r"C:\Users\somep\Documents\Flatiron\dsc-phase-1-project-v2-4\zippedData\ProductionBudget.csv",encoding='latin-1
```

In [22]:
```
budget = df1["production_budget"]
```

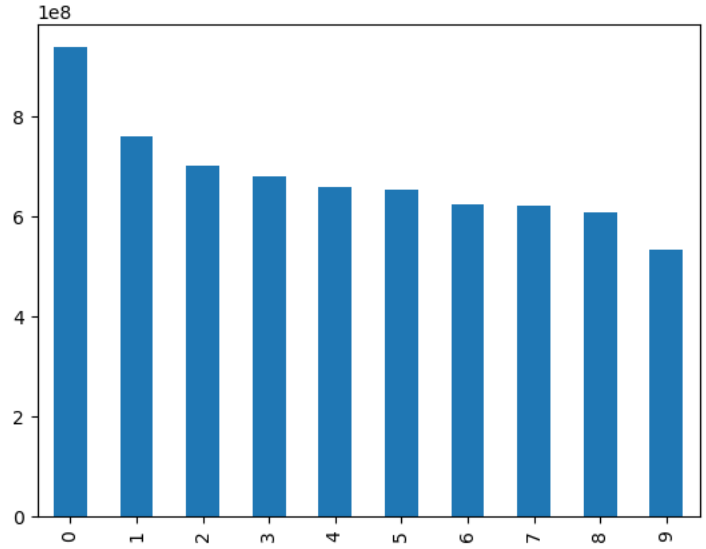
In [23]:
```
budgetplot = budget.plot(kind="bar")
```



In [24]:
```
df2 = pd.read_csv(r"C:\Users\somep\Documents\Flatiron\dsc-phase-1-project-v2-4\zippedData\Highest Grossing Movies of All Time Dom
```

In [25]:
```
domesticgross = df2["domestic_gross"]
```

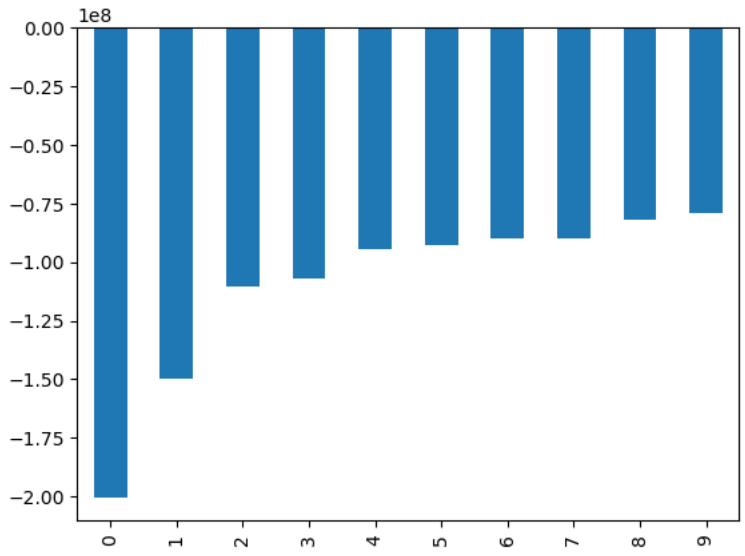
In [26]:
```
domesticplot = domesticgross.plot(kind="bar")
```



In [30]:
```
df3 = pd.read_csv(r"C:\Users\somep\Documents\Flatiron\dsc-phase-1-project-v2-4\zippedData\Least Profitable.csv",encoding='latin-1
```

In [33]:
```
leastprofitable = df3["Net_gain"]
```

In [34]:
```
leastprofitableplot = leastprofitable.plot(kind="bar")
```



In [ ]:
```

```