Practice Exercise 2

In this assignment, you will try to find some interesting insights into a few movies released between 1916 and 2016, using Python. You will have to download a movie dataset, write Python code to explore the data, gain insights into the movies, actors, directors, and collections, and submit the code.

Some tips before starting the assignment

1. Identify the task to be performed correctly, and only then proceed to write the required code. Don't perform any incorrect analysis or look for information that isn't required for the assignment.
2. In some cases, the variable names have already been assigned, and you just need to write code against them. In other cases, the names to be given are mentioned in the instructions. We strongly advise you to use the mentioned names only.
3. Always keep inspecting your data frame after you have performed a particular set of operations.
4. There are some checkpoints given in the IPython notebook provided. They're just useful pieces of information you can use to check if the result you have obtained after performing a particular task is correct or not.
5. Note that you will be asked to refer to documentation for solving some of the questions. That is done on purpose for you to learn new commands and also how to use the documentation.

In [ ]:
```
# Import the numpy and pandas packages

import numpy as np
import pandas as pd
```

Task 1: Reading and Inspection

Subtask 1.1: Import and read

Import and read the movie database. Store it in a variable called `movies`.

In [ ]:
```
# Write your code for importing the csv file here
movies = pd.read_csv("Movies.csv")
movies
```

Out[ ]:

	color	director_name	num_critic_for_reviews	duration	director_facebook_likes	a
0	Color	James Cameron	723.0	178.0	0.0	
1	Color	Gore Verbinski	302.0	169.0	563.0	
2	Color	Sam Mendes	602.0	148.0	0.0	
3	Color	Christopher Nolan	813.0	164.0	22000.0	
4	Color	Andrew Stanton	462.0	132.0	475.0	
...	...	...	...	...	...	
3848	Color	Shane Carruth	143.0	77.0	291.0	
3849	Color	Neill Dela Llana	35.0	80.0	0.0	
3850	Color	Robert Rodriguez	56.0	81.0	0.0	
3851	Color	Edward Burns	14.0	95.0	0.0	
3852	Color	Jon Gunn	43.0	90.0	16.0	

3853 rows × 28 columns

Subtask 1.2: Inspect the dataframe

Inspect the dataframe's columns, shapes, variable types etc.

In [ ]:
```
# Write your code for inspection here
movies.shape
movies.isna().sum()
```

Out[ ]:
```
color                       2
director_name               0
num_critic_for_reviews      1
duration                    1
director_facebook_likes     0
actor_3_facebook_likes      6
actor_2_name                1
actor_1_facebook_likes      0
gross                       0
genres                      0
actor_1_name                0
movie_title                 0
num_voted_users             0
cast_total_facebook_likes   0
actor_3_name                6
facenumber_in_poster        6
plot_keywords              30
movie_imdb_link             0
num_user_for_reviews        0
language                    4
country                     0
content_rating             48
budget                      0
title_year                  0
actor_2_facebook_likes      1
imdb_score                  0
aspect_ratio               72
movie_facebook_likes        0
dtype: int64
```

Question 1: How many rows and columns are present in the dataframe?

- (3821, 26)
- (3879, 28)
- (3853, 28)
- (3866, 26)

Question 2: How many columns have null values present in them? Try writing a code for this instead of counting them manually.

- 3
- 6
- 9
- 12

Task 2: Cleaning the Data

Subtask 2.1: Drop unecessary columns

For this assignment, you will mostly be analyzing the movies with respect to the ratings, gross collection, popularity of movies, etc. So many of the columns in this dataframe are not required. So it is advised to drop the following columns.

- color

- director_facebook_likes
- actor_1_facebook_likes
- actor_2_facebook_likes
- actor_3_facebook_likes
- actor_2_name
- cast_total_facebook_likes
- actor_3_name
- duration
- facenumber_in_poster
- content_rating
- country
- movie_imdb_link
- aspect_ratio
- plot_keywords

In [ ]:
```
# Check the 'drop' function in the Pandas library - dataframe.drop(list_of_unnec
# Write your code for dropping the columns here. It is advised to keep inspectin
movies.drop(['color','director_facebook_likes','actor_1_facebook_likes','actor_2
             'actor_3_facebook_likes','actor_2_name','cast_total_facebook_likes'
             'actor_3_name','duration','facenumber_in_poster','content_rating','
             'movie_imdb_link','aspect_ratio','plot_keywords'],axis=1,inplace=Tr
len(movies.columns)
```

Out[ ]: 13

Question 3: What is the count of columns in the new dataframe?

- 10
- 13
- 15
- 17

Subtask 2.2: Inspect Null values

As you have seen above, there are null values in multiple columns of the dataframe 'movies'. Find out the percentage of null values in each column of the dataframe 'movies'.

In [ ]:
```
# Write you code here
print("Percentage of null values in each column: ")
(movies.isna().sum()/movies.count()) * 100
```

```
Percentage of null values in each column:
```

Out[ ]:
```
director_name              0.000000
num_critic_for_reviews     0.025961
gross                      0.000000
genres                     0.000000
actor_1_name               0.000000
movie_title                0.000000
num_voted_users            0.000000
num_user_for_reviews       0.000000
language                   0.103923
budget                     0.000000
title_year                 0.000000
imdb_score                 0.000000
movie_facebook_likes       0.000000
dtype: float64
```

Question 4: Which column has the highest percentage of null values?

- language
- genres
- num_critic_for_reviews
- imdb_score

Subtask 2.3: Fill NaN values

You might notice that the `language` column has some NaN values. Here, on inspection, you will see that it is safe to replace all the missing values with `'English'`.

In [ ]:
```
# Write your code for filling the NaN values in the 'language' column here
movies['num_critic_for_reviews'] = movies['num_critic_for_reviews'].fillna(movie
movies["language"] = movies["language"].fillna("English")
movies["language"].value_counts()
```

Out[ ]:
```
language
English       3675
French          37
Spanish         26
Mandarin        14
German          13
Japanese        12
Hindi           10
Cantonese        8
Italian          7
Portuguese       5
Korean           5
Norwegian        4
Thai             3
Persian          3
Danish           3
Dutch            3
Dari             2
Indonesian       2
Hebrew           2
Aboriginal       2
Arabic           1
Russian          1
Vietnamese       1
Dzongkha         1
Romanian         1
Zulu             1
Bosnian          1
Czech            1
Icelandic        1
Hungarian        1
Mongolian        1
Aramaic          1
Telugu           1
Kazakh           1
Maya             1
Filipino         1
Swedish          1
Name: count, dtype: int64
```

Question 5: What is the count of movies made in English language after replacing the NaN values with English?

- 3670
- 3674
- 3668
- 3672

Task 3: Data Analysis

Subtask 3.1: Change the unit of columns

Convert the unit of the `budget` and `gross` columns from `$` to `million $`.

In [ ]:
```
# Write your code for unit conversion here
movies["budget"] = movies["budget"]/1000000
```

```
movies["gross"] = movies["gross"]/1000000
```

In [ ]:
```
movies
```

Out[ ]:

	director_name	num_critic_for_reviews	gross	gen
0	James Cameron	723.0	760.505847	Action|Adventure|Fantasy|Sc
1	Gore Verbinski	302.0	309.404152	Action|Adventure|Fant
2	Sam Mendes	602.0	200.074175	Action|Adventure|Thri
3	Christopher Nolan	813.0	448.130642	Action|Thri
4	Andrew Stanton	462.0	73.058679	Action|Adventure|Sc
...	...	...	...	
3848	Shane Carruth	143.0	0.424760	Drama|Sci-Fi|Thri
3849	Neill Dela Llana	35.0	0.070071	Thri
3850	Robert Rodriguez	56.0	2.040920	Action|Crime|Drama|Romance|Thri
3851	Edward Burns	14.0	0.004584	Comedy|Dra
3852	Jon Gunn	43.0	0.085222	Document

3853 rows × 13 columns

Subtask 3.2: Find the movies with highest profit

1. Create a new column called `profit` which contains the difference of the two columns: `gross` and `budget`.
2. Sort the dataframe using the `profit` column as reference. (Find which command can be used here to sort entries from the documentation)
3. Extract the top ten profiting movies in descending order and store them in a new dataframe - `top10`

In [ ]:
```
# Write your code for creating the profit column here
movies['profit'] = movies['gross'] - movies['budget']
movies.head()
```

Out[ ]:

	director_name	num_critic_for_reviews	gross	genres	actor_
0	James Cameron	723.0	760.505847	Action|Adventure|Fantasy|Sci-Fi	CCH
1	Gore Verbinski	302.0	309.404152	Action|Adventure|Fantasy	John
2	Sam Mendes	602.0	200.074175	Action|Adventure|Thriller	C
3	Christopher Nolan	813.0	448.130642	Action|Thriller	To
4	Andrew Stanton	462.0	73.058679	Action|Adventure|Sci-Fi	Dary

Write your code for sorting the dataframe here

In [ ]:
```
# Write your code to get the top 10 profiting movies here
top10 = movies.sort_values(by=['profit'],ascending=False)
top10.head(10)
```

Out[ ]:

	director_name	num_critic_for_reviews	gross	
0	James Cameron	723.0	760.505847	Action|Adventure|Fant
28	Colin Trevorrow	644.0	652.177271	Action|Adventure|Sci-
25	James Cameron	315.0	658.672302	Drama|
2704	George Lucas	282.0	460.935665	Action|Adventure|Fant
2748	Steven Spielberg	215.0	434.949459	Far
16	Joss Whedon	703.0	623.279547	Action|Advent
482	Roger Allers	186.0	422.783777	Adventure|Animation|Drama|Famil
230	George Lucas	320.0	474.544677	Action|Adventure|Fant
64	Christopher Nolan	645.0	533.316061	Action|Crime|Dran
419	Gary Ross	673.0	407.999255	Adventure|Drama|Sci-

Checkpoint: You might spot two movies directed by `James Cameron` in the list.

Question 6: Which movie is ranked 5th from the top in the list obtained?

- E.T. the Extra-Terrestrial
- The Avengers
- The Dark Knight
- Titanic

Subtask 3.3: Find IMDb Top 250

Create a new dataframe `IMDb_Top_250` and store the top 250 movies with the highest IMDb Rating (corresponding to the column: `imdb_score`). Also make sure that for all of these movies, the `num_voted_users` is greater than 25,000.

Also add a `Rank` column containing the values 1 to 250 indicating the ranks of the corresponding films.

In [ ]:
```
# Write your code for extracting the top 250 movies as per the IMDb score here.
# and name that dataframe as 'IMDb_Top_250'
IMDb_Top_250 = movies.sort_values(by=['imdb_score'],ascending=False)
IMDb_Top_250 = IMDb_Top_250[IMDb_Top_250['num_voted_users']>25000]
IMDb_Top_250 = IMDb_Top_250.head(250)
```

In [ ]:
```
IMDb_Top_250.groupby(pd.cut(IMDb_Top_250['imdb_score'],[7.5,8,8.5,9,9.5,10])).cc
```

```
C:\Users\Rommel\AppData\Local\Temp\ipykernel_12456\3983180866.py:1: FutureWarnin
g: The default of observed=False is deprecated and will be changed to True in a f
uture version of pandas. Pass observed=False to retain current behavior or observ
ed=True to adopt the future default and silence this warning.
  IMDb_Top_250.groupby(pd.cut(IMDb_Top_250['imdb_score'],[7.5,8,8.5,9,9.5,10])).c
ount()
```

Out[ ]:

imdb_score	director_name	num_critic_for_reviews	gross	genres	actor_1_name	movie
(7.5, 8.0]	98	98	98	98	98	
(8.0, 8.5]	124	124	124	124	124	
(8.5, 9.0]	26	26	26	26	26	
(9.0, 9.5]	2	2	2	2	2	
(9.5, 10.0]	0	0	0	0	0	

Question 7: Suppose movies are divided into 5 buckets based on the IMDb ratings:

- 7.5 to 8
- 8 to 8.5
- 8.5 to 9
- 9 to 9.5
- 9.5 to 10

Which bucket holds the maximum number of movies from *IMDb_Top_250*?

Subtask 3.4: Find the critic-favorite and audience-favorite actors

1. Create three new dataframes namely, `Meryl_Streep`, `Leo_Caprio`, and `Brad_Pitt` which contain the movies in which the actors: 'Meryl Streep', 'Leonardo DiCaprio', and 'Brad Pitt' are the lead actors. Use only the `actor_1_name` column for extraction. Also, make sure that you use the names 'Meryl Streep', 'Leonardo DiCaprio', and 'Brad Pitt' for the said extraction.
2. Append the rows of all these dataframes and store them in a new dataframe named `Combined`.
3. Group the combined dataframe using the `actor_1_name` column.
4. Find the mean of the `num_critic_for_reviews` and `num_user_for_review` and identify the actors which have the highest mean.

In [ ]:
```
# Write your code for creating three new dataframes here
Meryl_Streep = movies[movies['actor_1_name'] == 'Meryl Streep']
```

In [ ]:
```
Leo_Caprio = movies[movies['actor_1_name'] == 'Leonardo DiCaprio'] # Include all
```

In [ ]:
```
Brad_Pitt = movies[movies['actor_1_name'] == 'Brad Pitt']# Include all movies in
```

In [ ]:
```
# Write your code for combining the three dataframes here
Combined = pd.concat([Meryl_Streep,Leo_Caprio,Brad_Pitt])
Combined.head()
```

Out[ ]:

	director_name	num_critic_for_reviews	gross	genres	a
392	Nancy Meyers	187.0	112.703470	Comedy|Drama|Romance	
1038	Curtis Hanson	42.0	46.815748	Action|Adventure|Crime|Thriller	
1132	Nora Ephron	252.0	94.125426	Biography|Drama|Romance	
1322	David Frankel	208.0	124.732962	Comedy|Drama|Romance	
1390	Robert Redford	227.0	14.998070	Drama|Thriller|War	

In [ ]:
```
# Write your code for grouping the combined dataframe here
comb_grp = Combined.groupby(['actor_1_name'])
comb_grp
```

Out[ ]:
```
<pandas.core.groupby.generic.DataFrameGroupBy object at 0x000001BF3DA954D0>
```

In [ ]:
```
# Write the code for finding the mean of critic reviews and audience reviews her
comb_grp[['num_critic_for_reviews','num_user_for_reviews']].mean()
```

Out[ ]:

actor_1_name	num_critic_for_reviews	num_user_for_reviews
Brad Pitt	245.000000	742.352941
Leonardo DiCaprio	330.190476	914.476190
Meryl Streep	181.454545	297.181818

Question 8: Which actor is highest rated among the three actors according to the user reviews?

- Meryl Streep
- Leonardo DiCaprio
- Brad Pitt

Question 9: Which actor is highest rated among the three actors according to the critics?

- Meryl Streep
- Leonardo DiCaprio
- Brad Pitt

In [ ]:

Task2 Amazon Prime video data analysis

https://www.kaggle.com/datasets/shivamb/amazon-prime-movies-and-tv-shows?resource=download

In [ ]:
```
ama_prime = pd.read_csv("amazon_prime_titles.csv")
ama_prime.head()
```

Out[ ]:

	show_id	type	title	director	cast	country	date_added	release_year
0	s1	Movie	The Grand Seduction	Don McKellar	Brendan Gleeson, Taylor Kitsch, Gordon Pinsent	Canada	March 30, 2021	2014
1	s2	Movie	Take Care Good Night	Girish Joshi	Mahesh Manjrekar, Abhay Mahajan, Sachin Khedekar	India	March 30, 2021	2018
2	s3	Movie	Secrets of Deception	Josh Webber	Tom Sizemore, Lorenzo Lamas, Robert LaSardo, R...	United States	March 30, 2021	2017
3	s4	Movie	Pink: Staying True	Sonia Anderson	Interviews with: Pink, Adele, Beyoncé, Britney...	United States	March 30, 2021	2014
4	s5	Movie	Monster Maker	Giles Foster	Harry Dean Stanton, Kieran O'Brien, George Cos...	United Kingdom	March 30, 2021	1989

Show uniques values of a column 'director'

In [ ]:
```
ama_prime['director'].unique()
```

Out[ ]:
```
array(['Don McKellar', 'Girish Joshi', 'Josh Webber', ...,
       'John-Paul Davidson, Stephen Warbeck', 'Emily Skye',
       'Steve Barker'], dtype=object)
```

show all unique values with their counts

In [ ]:
```
ama_prime['director'].value_counts()
```

Out[ ]:
```
director
Mark Knight              113
Cannis Holder             61
Moonbug Entertainment     37
Jay Chapman               34
Arthur van Merwijk        30
                        ...
Karyn Kusama               1
K. Subash                  1
Robert Cuffley             1
J. Sabarish                1
Steve Barker               1
Name: count, Length: 5773, dtype: int64
```

get total no of uniwue values of whole data frame

In [ ]:
```
ama_prime.nunique()
```

Out[ ]:
```
show_id         9668
type               2
title           9668
director        5773
cast            7927
country           86
date_added        84
release_year     100
rating            24
duration         219
listed_in        518
description     9414
dtype: int64
```

In which year highest no of TV shows and movies were released

In [ ]:
```
ama_prime['release_year'].value_counts()
```

Out[ ]:
```
release_year
2021    1442
2020     962
2019     929
2018     623
2017     562
        ...
1922       2
1926       2
1924       1
1923       1
1927       1
Name: count, Length: 100, dtype: int64
```

how many TV and Movie shows are there in Data frame

In [ ]:
```
ama_prime['type'].value_counts()
```

Out[ ]:
```
type
Movie      7814
TV Show    1854
Name: count, dtype: int64
```

show all records with type 'movies; and country united kingdom

In [ ]:
```
ama_prime[(ama_prime['type'] == 'Movie') & (ama_prime['country'] == 'United King
```

Out[ ]:

	show_id	type	title	director	cast	country	date_added	release_ye
4	s5	Movie	Monster Maker	Giles Foster	Harry Dean Stanton, Kieran O'Brien, George Cos...	United Kingdom	March 30, 2021	19
5	s6	Movie	Living With Dinosaurs	Paul Weiland	Gregory Chisholm, Juliet Stevenson, Brian Hens...	United Kingdom	March 30, 2021	19
14	s15	Movie	Elon Musk: The Real Life Iron Man	Sonia Anderson	Elon Musk, Per Wimmer, Julie Anderson-Ankenbra...	United Kingdom	May 2, 2021	20
374	s375	Movie	The Zombie King	Aidan Belizaire	Edward Furlong, Corey Feldman, George McCluskey	United Kingdom	NaN	20
656	s657	Movie	The Flaw	David Sington	Andrew Luan, Robert Shiller, Louis Hyman	United Kingdom	NaN	20

show all movie records directed by Paul

In [ ]:
```
ama_prime[ama_prime['director'] == 'Paul Weiland']
```

Out[ ]:

	show_id	type	title	director	cast	country	date_added	release_year
5	s6	Movie	Living With Dinosaurs	Paul Weiland	Gregory Chisholm, Juliet Stevenson, Brian Hens...	United Kingdom	March 30, 2021	1989

Show top 3 Directors, who gave highest no of TV shows and movies released on Prime video

In [ ]:
```
ama_prime['director'].value_counts().head(3)
```

Out[ ]:
```
director
Mark Knight              113
Cannis Holder             61
Moonbug Entertainment     37
Name: count, dtype: int64
```

In which year Highest rating show was there

Task 3 Netflix Analysis

Information about TV shows and Movies 1- upload csv

2- describe, info,dtypes

3- uniques values of each column

4- total no of unique values of Dataframe

5- Unique values with their count

6-is any missing value with count

7- who is the director and show id of show #"ZOO"

8- Convert Datatype of column release date to DateTime

9-In which year highest no of TV shows and Movies relaesed

10-How many movies and TV shows are there in data set

11- Display Titles of all TV shows that were released in " United Sates" only

12- show top 10 Directors who gave highest no of TV shows and Movies on Netflix

13- show the record of all 'Horror' type of Movies

14 What are different 'Ratings' given by Netflix

15- What is Maximum duration of TV show on Netflix

16-sort dataframe by year

In [ ]:
```
netflix = pd.read_csv('netflix_titles.csv')
netflix.head()
```

Out[ ]:

	show_id	type	title	director	cast	country	date_added	release_year
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	September 25, 2021	2020
1	s2	TV Show	Blood & Water	NaN	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	September 24, 2021	2021
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi...	NaN	September 24, 2021	2021
3	s4	TV Show	Jailbirds New Orleans	NaN	NaN	NaN	September 24, 2021	2021
4	s5	TV Show	Kota Factory	NaN	Mayur More, Jitendra Kumar, Ranjan Raj, Alam K...	India	September 24, 2021	2021

In [ ]:
```
netflix.describe(),netflix.info(),netflix.dtypes
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8807 entries, 0 to 8806
Data columns (total 12 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   show_id       8807 non-null   object
 1   type          8807 non-null   object
 2   title         8807 non-null   object
 3   director      6173 non-null   object
 4   cast          7982 non-null   object
 5   country       7976 non-null   object
 6   date_added    8797 non-null   object
 7   release_year  8807 non-null   int64 
 8   rating        8803 non-null   object
 9   duration      8804 non-null   object
 10  listed_in     8807 non-null   object
 11  description   8807 non-null   object
dtypes: int64(1), object(11)
memory usage: 825.8+ KB
```

Out[ ]:
```
(       release_year
 count   8807.000000
 mean    2014.180198
 std        8.819312
 min     1925.000000
 25%     2013.000000
 50%     2017.000000
 75%     2019.000000
 max     2021.000000,
 None,
 show_id         object
 type            object
 title           object
 director        object
 cast            object
 country         object
 date_added      object
 release_year     int64
 rating          object
 duration        object
 listed_in       object
 description     object
 dtype: object)
```

In [ ]:
```
netflix.nunique().sum()
```

Out[ ]:
```
41951
```

In [ ]:
```
netflix.nunique()
```

Out[ ]:
```
show_id         8807
type               2
title           8807
director        4528
cast            7692
country          748
date_added      1767
release_year      74
rating            17
duration         220
listed_in        514
description     8775
dtype: int64
```

In [ ]:
```
netflix.isna().sum()
```

Out[ ]:
```
show_id            0
type               0
title              0
director        2634
cast             825
country          831
date_added        10
release_year       0
rating             4
duration           3
listed_in          0
description        0
dtype: int64
```

In [ ]:
```
dir = netflix[netflix["title"] == "Zoo"]
dir[['director','show_id']]
```

Out[ ]:

	director	show_id
4802	Shlok Sharma	s4803

In [ ]:
```
netflix['date_added'] = pd.to_datetime(netflix['date_added'], format='%B %d, %Y'
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
c:\Users\Rommel\OneDrive\Desktop\Coding\Academic\FODS\exp4\LAB2_Practice+Exercise
+2+Movies.ipynb Cell 70 line 1
----> <a href='vscode-notebook-cell:/c%3A/Users/Rommel/OneDrive/Desktop/Coding/Ac
ademic/FODS/exp4/LAB2_Practice%2BExercise%2B2%2BMovies.ipynb#Y131sZmlsZQ%3D%3D?li
ne=0'>1</a> netflix['date_added'] = pd.to_datetime(netflix['date_added'], format
='%B %d, %Y')

File c:\Users\Rommel\AppData\Local\Programs\Python\Python311\Lib\site-packages\pa
ndas\core\tools\datetimes.py:1108, in to_datetime(arg, errors, dayfirst, yearfirs
t, utc, format, exact, unit, infer_datetime_format, origin, cache)
   1106             result = arg.tz_localize("utc")
   1107 elif isinstance(arg, ABCSeries):
-> 1108     cache_array = _maybe_cache(arg, format, cache, convert_listlike)
   1109     if not cache_array.empty:
   1110         result = arg.map(cache_array)

File c:\Users\Rommel\AppData\Local\Programs\Python\Python311\Lib\site-packages\pa
ndas\core\tools\datetimes.py:254, in _maybe_cache(arg, format, cache, convert_lis
tlike)
    252 unique_dates = unique(arg)
    253 if len(unique_dates) < len(arg):
--> 254     cache_dates = convert_listlike(unique_dates, format)
    255     # GH#45319
    256     try:

File c:\Users\Rommel\AppData\Local\Programs\Python\Python311\Lib\site-packages\pa
ndas\core\tools\datetimes.py:488, in _convert_listlike_datetimes(arg, format, nam
e, utc, unit, errors, dayfirst, yearfirst, exact)
    486 # `format` could be inferred, or user didn't ask for mixed-format parsin
g.
    487 if format is not None and format != "mixed":
--> 488     return _array_strptime_with_fallback(arg, name, utc, format, exact, e
rrors)
    490 result, tz_parsed = objects_to_datetime64ns(
    491     arg,
    492     dayfirst=dayfirst,
   (...)
    496     allow_object=True,
    497 )
    499 if tz_parsed is not None:
    500     # We can take a shortcut since the datetime64 numpy array
    501     # is in UTC

File c:\Users\Rommel\AppData\Local\Programs\Python\Python311\Lib\site-packages\pa
ndas\core\tools\datetimes.py:519, in _array_strptime_with_fallback(arg, name, ut
c, fmt, exact, errors)
    508 def _array_strptime_with_fallback(
    509     arg,
    510     name,
   (...)
    514     errors: str,
    515 ) -> Index:
    516     """
    517     Call array_strptime, with fallback behavior depending on 'errors'.
    518     """
--> 519     result, timezones = array_strptime(arg, fmt, exact=exact, errors=erro
rs, utc=utc)
    520     if any(tz is not None for tz in timezones):
```

```
    521         return _return_parsed_timezone_results(result, timezones, utc, na
me)

File strptime.pyx:534, in pandas._libs.tslibs.strptime.array_strptime()

File strptime.pyx:355, in pandas._libs.tslibs.strptime.array_strptime()

ValueError: time data " August 4, 2017" doesn't match format "%B %d, %Y", at posi
tion 1442. You might want to try:
    - passing `format` if your strings have a consistent format;
    - passing `format='ISO8601'` if your strings are all ISO8601 but not necessar
ily in exactly the same format;
    - passing `format='mixed'`, and the format will be inferred for each element
individually. You might want to use `dayfirst` alongside this.
```