Assignment 1.2 - Exploring a Pandas Data Frame

In [14]:
```
# ========================================================================
# Title: Assignment 1.2 – Exploring a Pandas Data Frame
# Author: Pankaj Yadav
# Date: 4 Dec 2025
# Modified By: Pankaj Yadav
# Description: This program Exploring a Pandas Data Frame to analyze datasets
# ========================================================================
```

In [15]:
```
# Download the Video Game Sales with Ratings dataset from this link: Video Game Sales with Ratings.
# Import the required libraries
import pandas as pd
```

In [16]:
```
# Load the dataset as a Pandas data frame.
df = pd.read_csv('/Users/pyadav/Downloads/Video_Games_Sales_as_at_22_Dec_2016.csv')
```

In [17]:
```
# Display the first ten rows of data.
df.head(10)
```

Out[17]:

	Name	Platform	Year_of_Release	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales	Critic_Score	Critic_Count	User_Score	User_Count	Developer	Rating
0	Wii Sports	Wii	2006.0	Sports	Nintendo	41.36	28.96	3.77	8.45	82.53	76.0	51.0	8	322.0	Nintendo	E
1	Super Mario Bros.	NES	1985.0	Platform	Nintendo	29.08	3.58	6.81	0.77	40.24	NaN	NaN	NaN	NaN	NaN	NaN
2	Mario Kart Wii	Wii	2008.0	Racing	Nintendo	15.68	12.76	3.79	3.29	35.52	82.0	73.0	8.3	709.0	Nintendo	E
3	Wii Sports Resort	Wii	2009.0	Sports	Nintendo	15.61	10.93	3.28	2.95	32.77	80.0	73.0	8	192.0	Nintendo	E
4	Pokemon Red/Pokemon Blue	GB	1996.0	Role-Playing	Nintendo	11.27	8.89	10.22	1.00	31.37	NaN	NaN	NaN	NaN	NaN	NaN
5	Tetris	GB	1989.0	Puzzle	Nintendo	23.20	2.26	4.22	0.58	30.26	NaN	NaN	NaN	NaN	NaN	NaN
6	New Super Mario Bros.	DS	2006.0	Platform	Nintendo	11.28	9.14	6.50	2.88	29.80	89.0	65.0	8.5	431.0	Nintendo	E
7	Wii Play	Wii	2006.0	Misc	Nintendo	13.96	9.18	2.93	2.84	28.92	58.0	41.0	6.6	129.0	Nintendo	E
8	New Super Mario Bros. Wii	Wii	2009.0	Platform	Nintendo	14.44	6.94	4.70	2.24	28.32	87.0	80.0	8.4	594.0	Nintendo	E
9	Duck Hunt	NES	1984.0	Shooter	Nintendo	26.93	0.63	0.28	0.47	28.31	NaN	NaN	NaN	NaN	NaN	NaN

In [18]:
```
# Find the dimensions (number of rows and columns) in the data frame. What do these two numbers represent in the context of the data?

print("Number of rows:", df.shape[0])  # Number of video games released
print("Number of columns:", df.shape[1])  # Number of attributes per game
```

```
Number of rows: 16719
Number of columns: 16
```

In [19]:
```
# Find the top five games by critic score.
df.nlargest(5, 'Critic_Score')
```

Out[19]:

	Name	Platform	Year_of_Release	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales	Critic_Score	Critic_Count	User_Score	User_Count	Developer	Rating
51	Grand Theft Auto IV	X360	2008.0	Action	Take-Two Interactive	6.76	3.07	0.14	1.03	11.01	98.0	86.0	7.9	2951.0	Rockstar North	M
57	Grand Theft Auto IV	PS3	2008.0	Action	Take-Two Interactive	4.76	3.69	0.44	1.61	10.50	98.0	64.0	7.5	2833.0	Rockstar North	M
227	Tony Hawk's Pro Skater 2	PS	2000.0	Sports	Activision	3.05	1.41	0.02	0.20	4.68	98.0	19.0	7.7	299.0	Neversoft Entertainment	T
5350	SoulCalibur	DC	1999.0	Fighting	Namco Bandai Games	0.00	0.00	0.34	0.00	0.34	98.0	24.0	8.8	200.0	Namco	T
16	Grand Theft Auto V	PS3	2013.0	Action	Take-Two Interactive	7.02	9.09	0.98	3.96	21.04	97.0	50.0	8.2	3994.0	Rockstar North	M

In [20]:
```
# Find the number of video games in the data frame in each genre.
df['Genre'].value_counts()
```

Out[20]:
```
Genre
Action          3370
Sports          2348
Misc            1750
Role-Playing    1500
Shooter         1323
Adventure       1303
Racing          1249
Platform         888
Simulation       874
Fighting         849
Strategy         683
Puzzle           580
Name: count, dtype: int64
```

In [21]:
```
# Find the first five games in the data frame on the SNES platform.
df[df['Platform'] == 'SNES'].head(5)
```

Out[21]:

	Name	Platform	Year_of_Release	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales	Critic_Score	Critic_Count	User_Score	User_Count	Developer	Rating
18	Super Mario World	SNES	1990.0	Platform	Nintendo	12.78	3.75	3.54	0.55	20.61	NaN	NaN	NaN	NaN	NaN	NaN
56	Super Mario All-Stars	SNES	1993.0	Platform	Nintendo	5.99	2.15	2.12	0.29	10.55	NaN	NaN	NaN	NaN	NaN	NaN
71	Donkey Kong Country	SNES	1994.0	Platform	Nintendo	4.36	1.71	3.00	0.23	9.30	NaN	NaN	NaN	NaN	NaN	NaN
76	Super Mario Kart	SNES	1992.0	Racing	Nintendo	3.54	1.24	3.81	0.18	8.76	NaN	NaN	NaN	NaN	NaN	NaN
137	Street Fighter II: The World Warrior	SNES	1992.0	Fighting	Capcom	2.47	0.83	2.87	0.12	6.30	NaN	NaN	NaN	NaN	NaN	NaN

In [22]:
```
# Find the five publishers with the highest total global sales. Note: You will need to calculate the total global sales for each publisher to do this.
df.groupby('Publisher')['Global_Sales'].sum().nlargest(5)
```

Out[22]:
```
Publisher
Nintendo                1788.81
Electronic Arts         1116.96
Activision               731.16
Sony Computer Entertainment     606.48
Ubisoft                  471.61
Name: Global_Sales, dtype: float64
```

In [23]:
```
# Create a new column in the data frame that calculates the percentage of global sales from North America. Display the first five rows of the new data frame.
df['NA_Sales_Percentage'] = (df['NA_Sales'] / df['Global_Sales']) * 100
df.head(5)
```

Out[23]:

	Name	Platform	Year_of_Release	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales	Critic_Score	Critic_Count	User_Score	User_Count	Developer	Rating	NA_
0	Wii Sports	Wii	2006.0	Sports	Nintendo	41.36	28.96	3.77	8.45	82.53	76.0	51.0	8	322.0	Nintendo	E	
1	Super Mario Bros.	NES	1985.0	Platform	Nintendo	29.08	3.58	6.81	0.77	40.24	NaN	NaN	NaN	NaN	NaN	NaN	
2	Mario Kart Wii	Wii	2008.0	Racing	Nintendo	15.68	12.76	3.79	3.29	35.52	82.0	73.0	8.3	709.0	Nintendo	E	
3	Wii Sports Resort	Wii	2009.0	Sports	Nintendo	15.61	10.93	3.28	2.95	32.77	80.0	73.0	8	192.0	Nintendo	E	
4	Pokemon Red/Pokemon Blue	GB	1996.0	Role-Playing	Nintendo	11.27	8.89	10.22	1.00	31.37	NaN	NaN	NaN	NaN	NaN	NaN	

In [26]:
```
# Find the number NaN entries (missing data values) in each column.
df.isna().sum()
```

Out[26]:
```
Name                      2
Platform                  0
Year_of_Release         269
Genre                     2
Publisher                54
NA_Sales                  0
EU_Sales                  0
JP_Sales                  0
Other_Sales               0
Global_Sales              0
Critic_Score           8582
Critic_Count           8582
User_Score             6704
User_Count             9129
Developer              6623
Rating                 6769
NA_Sales_Percentage       0
dtype: int64
```

In [ ]:
```
# Calculate the median user score of all the video games.
median_user_score = df['User_Score'].median()
# median_user_score
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[28], line 3
      1 # Calculate the median user score of all the video games.
      2 # You will likely run into an error because some of the user score entries are a non-numerical string that cannot be converted to a float.
----> 3 median_user_score = df['User_Score'].median()

File /opt/anaconda3/lib/python3.12/site-packages/pandas/core/series.py:6559, in Series.median(self, axis, skipna, numeric_only, **kwargs)
   6551 @doc(make_doc("median", ndim=1))
   6552 def median(
   6553     self,
   (...)
   6557     **kwargs,
   6558 ):
-> 6559     return NDFrame.median(self, axis, skipna, numeric_only, **kwargs)

File /opt/anaconda3/lib/python3.12/site-packages/pandas/core/generic.py:12431, in NDFrame.median(self, axis, skipna, numeric_only, **kwargs)
  12424 def median(
  12425     self,
  12426     axis: Axis | None = 0,
  (...)
  12429     **kwargs,
  12430 ) -> Series | float:
> 12431     return self._stat_function(
  12432         "median", nanops.nanmedian, axis, skipna, numeric_only, **kwargs
  12433     )

File /opt/anaconda3/lib/python3.12/site-packages/pandas/core/generic.py:12377, in NDFrame._stat_function(self, name, func, axis, skipna, numeric_only, **kwargs)
  12373 nv.validate_func(name, (), kwargs)
  12375 validate_bool_kwarg(skipna, "skipna", none_allowed=False)
> 12377 return self._reduce(
  12378     func, name=name, axis=axis, skipna=skipna, numeric_only=numeric_only
  12379 )

File /opt/anaconda3/lib/python3.12/site-packages/pandas/core/series.py:6457, in Series._reduce(self, op, name, axis, skipna, numeric_only, filter_type, **kwds)
   6452     # GH#47500 - change to TypeError to match other methods
   6453     raise TypeError(
   6454         f"Series.{name} does not allow {kwd_name}={numeric_only} "
   6455         "with non-numeric dtypes."
   6456     )
-> 6457 return op(delegate, skipna=skipna, **kwds)

File /opt/anaconda3/lib/python3.12/site-packages/pandas/core/nanops.py:147, in bottleneck_switch.__call__.<locals>.f(values, axis, skipna, **kwds)
    145         result = alt(values, axis=axis, skipna=skipna, **kwds)
    146     else:
--> 147         result = alt(values, axis=axis, skipna=skipna, **kwds)
    149 return result

File /opt/anaconda3/lib/python3.12/site-packages/pandas/core/nanops.py:787, in nanmedian(values, axis, skipna, mask)
    785     inferred = lib.infer_dtype(values)
    786     if inferred in ("string", "mixed"):
--> 787         raise TypeError(f"Cannot convert {values} to numeric")
    788 try:
    789     values = values.astype("f8")

TypeError: Cannot convert ['8' nan '8.3' ... nan nan nan] to numeric
```

In [ ]:
```
# You will likely run into an error because some of the user score entries are a non-numerical string that cannot be converted to a float.
# Find and replace this string with NaN and then calculate the median.
df['User_Score'] = pd.to_numeric(df['User_Score'], errors='coerce')
median_user_score = df['User_Score'].median()
print("Median Score is :", median_user_score)
```

```
Median Score is : 7.5
```

In [35]:
```
# Then, replace all NaN entries in the user score column with the median value.
df['User_Score'].fillna(median_user_score, inplace=True)

# Finally, make sure there are no more NaN entries in the user score column.
df['User_Score'].isna().sum()
```

```
/var/folders/cv/71w18t8n27b_81x9rhwkm1080000gq/T/ipykernel_12607/343382449.py:2: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained ass
ignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to perform the operat
ion inplace on the original object.


  df['User_Score'].fillna(median_user_score, inplace=True)
```

Out[35]:
```
0
```

In [ ]:
```

```