Week 1 - 1.2 Exploring a Pandas Data Frame

In [1]:
```
import pandas as pd
import numpy as np

#Load the dataset as a Pandas data frame.
video_games_df = pd.read_csv('Video_Games_Sales_as_at_22_Dec_2016.csv')
```

In [2]:
```
#Display the first ten rows of data.
video_games_df.head(10)
```

Out[2]:

	Name	Platform	Year_of_Release	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global
0	Wii Sports	Wii	2006.0	Sports	Nintendo	41.36	28.96	3.77	8.45	
1	Super Mario Bros.	NES	1985.0	Platform	Nintendo	29.08	3.58	6.81	0.77	
2	Mario Kart Wii	Wii	2008.0	Racing	Nintendo	15.68	12.76	3.79	3.29	
3	Wii Sports Resort	Wii	2009.0	Sports	Nintendo	15.61	10.93	3.28	2.95	
4	Pokemon Red/Pokemon Blue	GB	1996.0	Role-Playing	Nintendo	11.27	8.89	10.22	1.00	
5	Tetris	GB	1989.0	Puzzle	Nintendo	23.20	2.26	4.22	0.58	
6	New Super Mario Bros.	DS	2006.0	Platform	Nintendo	11.28	9.14	6.50	2.88	
7	Wii Play	Wii	2006.0	Misc	Nintendo	13.96	9.18	2.93	2.84	
8	New Super Mario Bros. Wii	Wii	2009.0	Platform	Nintendo	14.44	6.94	4.70	2.24	
9	Duck Hunt	NES	1984.0	Shooter	Nintendo	26.93	0.63	0.28	0.47	

In [3]:
```
#Find the dimensions (number of rows and columns) in the data frame.
#What do these two numbers represent in the context of the data?
#No. of rows and columns
video_games_df.shape
```

Out[3]:
```
(16719, 16)
```

In [4]:
```
#Find the top five games by critic score
video_games_df.sort_values(['Critic_Score'], ascending=[False]).Name.head(5)
```

Out[4]:
```
227     Tony Hawk's Pro Skater 2
57           Grand Theft Auto IV
51           Grand Theft Auto IV
5350                  SoulCalibur
165           Grand Theft Auto V
Name: Name, dtype: object
```

In [5]:
```
#Find the number of video games in the data frame in each genre.
video_games_df.groupby(['Genre'])['Genre'].count().sort_index()
```

Out[5]:
```
Genre
```

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

In [6]:
```
#Find the first five games in the data frame on the SNES platform.
video_games_df[video_games_df.Platform == 'SNES'].head(5)
```

Out[6]:

	Name	Platform	Year_of_Release	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sa
18	Super Mario World	SNES	1990.0	Platform	Nintendo	12.78	3.75	3.54	0.55	20
56	Super Mario All-Stars	SNES	1993.0	Platform	Nintendo	5.99	2.15	2.12	0.29	10
71	Donkey Kong Country	SNES	1994.0	Platform	Nintendo	4.36	1.71	3.00	0.23	9
76	Super Mario Kart	SNES	1992.0	Racing	Nintendo	3.54	1.24	3.81	0.18	8
137	Street Fighter II: The World Warrior	SNES	1992.0	Fighting	Capcom	2.47	0.83	2.87	0.12	6

In [7]:
```
#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.


## Convert to dataframe
#publisher_list = video_games_df.groupby('Publisher')['Global_Sales'].sum()
#publisher_score_df = pd.DataFrame(publisher_list)
#publisher_score_df.sort_values(['Global_Sales'], ascending=[False]).head(5)

#Avoid dataframe conversion
video_games_df.groupby('Publisher')['Global_Sales'].sum().sort_values(ascending=[False])
```

Out[7]:
```
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 [8]:
```
#Create a new column in the data frame that calculates the percentage of global sales fr
#Display the first five rows of the new data frame.
```

```
video_games_df['NA_Sales_Percentage'] = round(video_games_df.NA_Sales/video_games_df.Glo
video_games_df.head(5)
```

Out[8]:

	Name	Platform	Year_of_Release	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global
0	Wii Sports	Wii	2006.0	Sports	Nintendo	41.36	28.96	3.77	8.45	
1	Super Mario Bros.	NES	1985.0	Platform	Nintendo	29.08	3.58	6.81	0.77	
2	Mario Kart Wii	Wii	2008.0	Racing	Nintendo	15.68	12.76	3.79	3.29	
3	Wii Sports Resort	Wii	2009.0	Sports	Nintendo	15.61	10.93	3.28	2.95	
4	Pokemon Red/Pokemon Blue	GB	1996.0	Role-Playing	Nintendo	11.27	8.89	10.22	1.00	

In [9]:
```
#Find the number NaN entries (missing data values) in each column.
video_games_df.isnull().values.any()
video_games_df.isnull().sum()
```

Out[9]:
```
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 [10]:
```
#Try to calculate the median user score of all the video games.
#You will likely run into an error because some of the user score entries are a non-nume
#that cannot be converted to a float. Find and replace this string with NaN and then cal
video_games_df[pd.to_numeric(video_games_df['User_Score'], errors='coerce').isnull()]['U
```

Out[10]:
```
array([nan, 'tbd'], dtype=object)
```

In [11]:
```
video_games_df.User_Score = video_games_df.User_Score.replace('tbd',np.nan)
median = video_games_df.User_Score.median()
print('Median after replacing string with Nan : ',median)

#Then, replace all NaN entries in the user score column with the median value.
video_games_df.User_Score = video_games_df.User_Score.replace(np.nan,median)
```

```
Median after replacing string with Nan :  7.5
```