Chitramoy Mukherjee

DSC 550-T301

Week-1

Date : 11/30/2023

In [1]:
```
import pandas as pd
import csv
```

In [4]:
```
# Load Video Game Sales with Ratings as a Pandas data frame.
file_path = 'C:\\Users\\14024\\OneDrive\\Desktop\\MS-DSC\\DSC-550\\Week-1\\Video_Games
video_game_df = pd.read_csv(file_path)

# Display the first ten rows of the DataFrame
print(video_game_df.head(10))
```

```
                      Name Platform  Year_of_Release         Genre  \
0                Wii Sports      Wii           2006.0        Sports
1          Super Mario Bros.      NES           1985.0      Platform
2             Mario Kart Wii      Wii           2008.0        Racing
3          Wii Sports Resort      Wii           2009.0        Sports
4   Pokemon Red/Pokemon Blue       GB           1996.0  Role-Playing
5                     Tetris       GB           1989.0        Puzzle
6      New Super Mario Bros.       DS           2006.0      Platform
7                   Wii Play      Wii           2006.0          Misc
8  New Super Mario Bros. Wii      Wii           2009.0      Platform
9                  Duck Hunt      NES           1984.0       Shooter

  Publisher  NA_Sales  EU_Sales  JP_Sales  Other_Sales  Global_Sales  \
0  Nintendo     41.36     28.96      3.77         8.45         82.53
1  Nintendo     29.08      3.58      6.81         0.77         40.24
2  Nintendo     15.68     12.76      3.79         3.29         35.52
3  Nintendo     15.61     10.93      3.28         2.95         32.77
4  Nintendo     11.27      8.89     10.22         1.00         31.37
5  Nintendo     23.20      2.26      4.22         0.58         30.26
6  Nintendo     11.28      9.14      6.50         2.88         29.80
7  Nintendo     13.96      9.18      2.93         2.84         28.92
8  Nintendo     14.44      6.94      4.70         2.24         28.32
9  Nintendo     26.93      0.63      0.28         0.47         28.31

   Critic_Score  Critic_Count User_Score  User_Count Developer Rating
0          76.0          51.0          8       322.0  Nintendo      E
1           NaN           NaN        NaN         NaN       NaN    NaN
2          82.0          73.0        8.3       709.0  Nintendo      E
3          80.0          73.0          8       192.0  Nintendo      E
4           NaN           NaN        NaN         NaN       NaN    NaN
5           NaN           NaN        NaN         NaN       NaN    NaN
6          89.0          65.0        8.5       431.0  Nintendo      E
7          58.0          41.0        6.6       129.0  Nintendo      E
8          87.0          80.0        8.4       594.0  Nintendo      E
9           NaN           NaN        NaN         NaN       NaN    NaN
```

In [5]:
```
# Find the dimensions of the DataFrame
num_rows, num_columns = video_game_df.shape
print(f"Number of rows: {num_rows}, Number of columns: {num_columns}")
```

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

In [7]:
```
# Find the top five games by critic score
top_games_by_critic_score = video_game_df.nlargest(5, 'Critic_Score')
print("\nTop five games by critic score:")
print(top_games_by_critic_score)
```

```
Top five games by critic score:
                         Name Platform  Year_of_Release     Genre  \
51         Grand Theft Auto IV     X360           2008.0    Action
57         Grand Theft Auto IV      PS3           2008.0    Action
227   Tony Hawk's Pro Skater 2       PS           2000.0    Sports
5350              SoulCalibur       DC           1999.0  Fighting
16          Grand Theft Auto V      PS3           2013.0    Action

                 Publisher  NA_Sales  EU_Sales  JP_Sales  Other_Sales  \
51    Take-Two Interactive      6.76      3.07      0.14         1.03
57    Take-Two Interactive      4.76      3.69      0.44         1.61
227            Activision      3.05      1.41      0.02         0.20
5350   Namco Bandai Games      0.00      0.00      0.34         0.00
16    Take-Two Interactive      7.02      9.09      0.98         3.96

      Global_Sales  Critic_Score  Critic_Count User_Score  User_Count  \
51           11.01          98.0          86.0        7.9      2951.0
57           10.50          98.0          64.0        7.5      2833.0
227           4.68          98.0          19.0        7.7       299.0
5350          0.34          98.0          24.0        8.8       200.0
16           21.04          97.0          50.0        8.2      3994.0

                    Developer Rating
51             Rockstar North      M
57             Rockstar North      M
227   Neversoft Entertainment      T
5350                   Namco      T
16             Rockstar North      M
```

In [9]:
```
# Find the number of video games in each genre
genre_counts = video_game_df['Genre'].value_counts()

print("\nNumber of video games in each genre:")
print(genre_counts)
```

```
Number of video games in each 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: Genre, dtype: int64
```

In [11]:
```
# Find the first five games on the SNES platform
snes_games = video_game_df[video_game_df['Platform'] == 'SNES'].head(5)
print("\nFirst five games on the SNES platform:")
print(snes_games)
```

```
First five games on the SNES platform:
                                     Name Platform  Year_of_Release     Genre  \
18                      Super Mario World     SNES           1990.0  Platform
56                  Super Mario All-Stars     SNES           1993.0  Platform
71                    Donkey Kong Country     SNES           1994.0  Platform
76                       Super Mario Kart     SNES           1992.0    Racing
137  Street Fighter II: The World Warrior     SNES           1992.0  Fighting

    Publisher  NA_Sales  EU_Sales  JP_Sales  Other_Sales  Global_Sales  \
18   Nintendo     12.78      3.75      3.54         0.55         20.61
56   Nintendo      5.99      2.15      2.12         0.29         10.55
71   Nintendo      4.36      1.71      3.00         0.23          9.30
76   Nintendo      3.54      1.24      3.81         0.18          8.76
137    Capcom      2.47      0.83      2.87         0.12          6.30

     Critic_Score  Critic_Count User_Score  User_Count Developer Rating
18            NaN           NaN        NaN         NaN       NaN    NaN
56            NaN           NaN        NaN         NaN       NaN    NaN
71            NaN           NaN        NaN         NaN       NaN    NaN
76            NaN           NaN        NaN         NaN       NaN    NaN
137           NaN           NaN        NaN         NaN       NaN    NaN
```

In [12]:
```
sales_df = video_game_df[['Publisher', 'Global_Sales']]

# Group by 'Publisher' and sum the global sales for each publisher
total_sales_by_publisher = sales_df.groupby('Publisher')['Global_Sales'].sum()

# Sort the publishers based on total global sales in descending order
sorted_publishers = total_sales_by_publisher.sort_values(ascending=False)

# Get the top five publishers with the highest global sales
top_five_publishers = sorted_publishers.head(5)

# Display Top five publishers
print(top_five_publishers)
```

```
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 [14]:
```
# Create a new column for the percentage of global sales from North America
video_game_df['Percentage_NA_Sales'] = (video_game_df['NA_Sales'] / video_game_df['Glc

# Display the first few rows of the new data frame with the new column
print("\nDataFrame with Percentage of Global Sales from North America:")
print(video_game_df.head())
```

```
DataFrame with Percentage of Global Sales from North America:
                      Name Platform  Year_of_Release         Genre Publisher  \
0               Wii Sports      Wii           2006.0        Sports  Nintendo
1         Super Mario Bros.      NES           1985.0      Platform  Nintendo
2            Mario Kart Wii      Wii           2008.0        Racing  Nintendo
3         Wii Sports Resort      Wii           2009.0        Sports  Nintendo
4  Pokemon Red/Pokemon Blue       GB           1996.0  Role-Playing  Nintendo

   NA_Sales  EU_Sales  JP_Sales  Other_Sales  Global_Sales  Critic_Score  \
0     41.36     28.96      3.77         8.45         82.53          76.0
1     29.08      3.58      6.81         0.77         40.24           NaN
2     15.68     12.76      3.79         3.29         35.52          82.0
3     15.61     10.93      3.28         2.95         32.77          80.0
4     11.27      8.89     10.22         1.00         31.37           NaN

   Critic_Count User_Score  User_Count Developer Rating  Percentage_NA_Sales
0          51.0          8       322.0  Nintendo      E            50.115110
1           NaN        NaN         NaN       NaN    NaN            72.266402
2          73.0        8.3       709.0  Nintendo      E            44.144144
3          73.0          8       192.0  Nintendo      E            47.635032
4           NaN        NaN         NaN       NaN    NaN            35.926044
```

In [15]:
```python
# Check for NaN entries
nan_counts = video_game_df.isna().sum()
print("Number of NaN entries in each column:")
print(nan_counts)
```

```
Number of NaN entries in each column:
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
Percentage_NA_Sales       0
dtype: int64
```

In [17]:
```python
# Check the data types of the 'User_Score' column
print("\nData types before conversion:")
print(video_game_df.dtypes)

# Convert 'User_Score' to numeric, coerce errors to NaN
video_game_df['User_Score'] = pd.to_numeric(video_game_df['User_Score'], errors='coerc

# Display the DataFrame after converting 'User_Score' to numeric
print("\nDataFrame after converting 'User_Score' to numeric:")
print(video_game_df.head())

# Calculate the median of 'User_Score' excluding NaN values
median_user_score = video_game_df['User_Score'].median()
```

```
# Replace NaN entries in 'User_Score' with the calculated median
video_game_df['User_Score'].fillna(median_user_score, inplace=True)

# Display the final DataFrame after replacing NaN entries with the median
print("\nFinal DataFrame:")
print(video_game_df.head())

# Display the median user score
print(f"\nMedian User Score: {median_user_score}")
```

```
Data types before conversion:
Name                   object
Platform               object
Year_of_Release       float64
Genre                  object
Publisher              object
NA_Sales              float64
EU_Sales              float64
JP_Sales              float64
Other_Sales           float64
Global_Sales          float64
Critic_Score          float64
Critic_Count          float64
User_Score            float64
User_Count            float64
Developer              object
Rating                 object
Percentage_NA_Sales   float64
dtype: object

DataFrame after converting 'User_Score' to numeric:
                      Name Platform  Year_of_Release         Genre Publisher  \
0               Wii Sports      Wii           2006.0        Sports  Nintendo
1         Super Mario Bros.      NES           1985.0      Platform  Nintendo
2            Mario Kart Wii      Wii           2008.0        Racing  Nintendo
3         Wii Sports Resort      Wii           2009.0        Sports  Nintendo
4  Pokemon Red/Pokemon Blue       GB           1996.0  Role-Playing  Nintendo

   NA_Sales  EU_Sales  JP_Sales  Other_Sales  Global_Sales  Critic_Score  \
0     41.36     28.96      3.77         8.45         82.53          76.0
1     29.08      3.58      6.81         0.77         40.24           NaN
2     15.68     12.76      3.79         3.29         35.52          82.0
3     15.61     10.93      3.28         2.95         32.77          80.0
4     11.27      8.89     10.22         1.00         31.37           NaN

   Critic_Count  User_Score  User_Count Developer Rating  Percentage_NA_Sales
0          51.0         8.0       322.0  Nintendo      E            50.115110
1           NaN         NaN         NaN       NaN    NaN            72.266402
2          73.0         8.3       709.0  Nintendo      E            44.144144
3          73.0         8.0       192.0  Nintendo      E            47.635032
4           NaN         NaN         NaN       NaN    NaN            35.926044

Final DataFrame:
                      Name Platform  Year_of_Release         Genre Publisher  \
0               Wii Sports      Wii           2006.0        Sports  Nintendo
1         Super Mario Bros.      NES           1985.0      Platform  Nintendo
2            Mario Kart Wii      Wii           2008.0        Racing  Nintendo
3         Wii Sports Resort      Wii           2009.0        Sports  Nintendo
4  Pokemon Red/Pokemon Blue       GB           1996.0  Role-Playing  Nintendo

   NA_Sales  EU_Sales  JP_Sales  Other_Sales  Global_Sales  Critic_Score  \
0     41.36     28.96      3.77         8.45         82.53          76.0
1     29.08      3.58      6.81         0.77         40.24           NaN
2     15.68     12.76      3.79         3.29         35.52          82.0
3     15.61     10.93      3.28         2.95         32.77          80.0
4     11.27      8.89     10.22         1.00         31.37           NaN

   Critic_Count  User_Score  User_Count Developer Rating  Percentage_NA_Sales
0          51.0         8.0       322.0  Nintendo      E            50.115110
1           NaN         7.5         NaN       NaN    NaN            72.266402
```

```
2          73.0         8.3        709.0  Nintendo      E            44.144144
3          73.0         8.0        192.0  Nintendo      E            47.635032
4           NaN         7.5          NaN       NaN    NaN            35.926044

Median User Score: 7.5
```