Exploratory Data Analysis on Olympics Games

Olympics is a popular sporting event. It first began in 1896 - Athens, Greece. The 5 rings in the logo represent the five continents: Europe, Africa, Asia, the Americas, and Oceanica.

This Project is an Exploratory Data Analysis using Python and visualize past Olympics data and answer specific questions. Some specific Python libraries such as Numpy, Pandas, Matplotlib and Seaborn to make sense of our data and extract useful information. Results are visualized using different charts and graphs.

Downloading the Dataset

Datasets used in this projects are `athelete_events.csv` and `noc_regions.csv` obtained from https://www.kaggle.com/datasets?fileType=csv

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

```
#loading datasets
athletes_df = pd.read_csv("athlete_events.csv")
regions_df = pd.read_csv("noc_regions.csv")
```

```
#Joining the 2 dataframes
athletes_df = athletes_df.merge(regions_df, how='left',on="NOC")
```

```
athletes_df.rename(columns = {'region' : 'Region' , 'notes' : 'Notes'}, inplace=True)
athletes_df.describe()
```

	ID	Age	Height	Weight	Year
count	271116.000000	261642.000000	210945.000000	208241.000000	271116.000000
mean	68248.954396	25.556898	175.338970	70.702393	1978.378480
std	39022.286345	6.393561	10.518462	14.348020	29.877632
min	1.000000	10.000000	127.000000	25.000000	1896.000000
25%	34643.000000	21.000000	168.000000	60.000000	1960.000000
50%	68205.000000	24.000000	175.000000	70.000000	1988.000000
75%	102097.250000	28.000000	183.000000	79.000000	2002.000000
max	135571.000000	97.000000	226.000000	214.000000	2016.000000

```
#Check for NULL values
nan_values = athletes_df.isna()
nan_columns=nan_values.any()
nan_columns
#Note: The columns which donot have any NULL values are represented by FALSE whereas co
```

```
ID        False
Name      False
Sex       False
Age        True
Height     True
Weight     True
Team      False
NOC       False
Games     False
Year      False
Season    False
City      False
Sport     False
Event     False
Medal      True
Region     True
Notes      True
dtype: bool
```

```
#Total no. of NULL values present in each column of dataframe
athletes_df . isnull() . sum()
```

```
ID            0
Name          0
Sex           0
Age        9474
Height    60171
Weight    62875
Team          0
NOC           0
Games         0
```

```
Year           0
Season         0
City           0
Sport          0
Event          0
Medal     231333
Region       370
Notes     266077
dtype: int64
```

Query-1 : Print the list of column names containing NULL values or missing values

```
null_columns=list(athletes_df.columns[athletes_df.isna().any()])
print(null_columns)
```

```
['Age', 'Height', 'Weight', 'Medal', 'Region', 'Notes']
```

Query-2 : Details Of India (First 5 rows)

```
athletes_df.query('''Team == 'India' ''').head(5)
```

	ID	Name	Sex	Age	Height	Weight	Team	NOC	Games	Year	Season	City	Sport	E
505	281	S. Abdul Hamid	M	NaN	NaN	NaN	India	IND	1928 Summer	1928	Summer	Amsterdam	Athletics	Athl M m Hu
506	281	S. Abdul Hamid	M	NaN	NaN	NaN	India	IND	1928 Summer	1928	Summer	Amsterdam	Athletics	Athl M m Hu
895	512	Shiny Kurisingal Abraham-Wilson	F	19.0	167.0	53.0	India	IND	1984 Summer	1984	Summer	Los Angeles	Athletics	Athl Wor m
896	512	Shiny Kurisingal Abraham-Wilson	F	19.0	167.0	53.0	India	IND	1984 Summer	1984	Summer	Los Angeles	Athletics	Athl Wor 4 x m
897	512	Shiny Kurisingal Abraham-Wilson	F	23.0	167.0	53.0	India	IND	1988 Summer	1988	Summer	Seoul	Athletics	Athl Wor m

Query-3 :Details Of Japan (First 5 rows)

```
athletes_df.query(''' Team == 'Japan' ''').head(5)
```

	ID	Name	Sex	Age	Height	Weight	Team	NOC	Games	Year	Season	City	Sport	
625	362	Isao Ko Abe	M	24.0	177.0	75.0	Japan	JPN	1936 Summer	1936	Summer	Berlin	Athletics	

	ID	Name	Sex	Age	Height	Weight	Team	NOC	Games	Year	Season	City	Sport	
629	363	Kazumi Abe	M	28.0	178.0	67.0	Japan	JPN	1976 Winter	1976	Winter	Innsbruck	Bobsleigh	B Me
630	364	Kazuo Abe	M	25.0	166.0	69.0	Japan	JPN	1960 Summer	1960	Summer	Roma	Wrestling	W Ligh F
631	365	Kinya Abe	M	23.0	168.0	68.0	Japan	JPN	1992 Summer	1992	Summer	Barcelona	Fencing	Fenci Foil, In
632	366	Kiyoshi Abe	M	25.0	167.0	62.0	Japan	JPN	1972 Summer	1972	Summer	Munich	Wrestling	W Feathe F

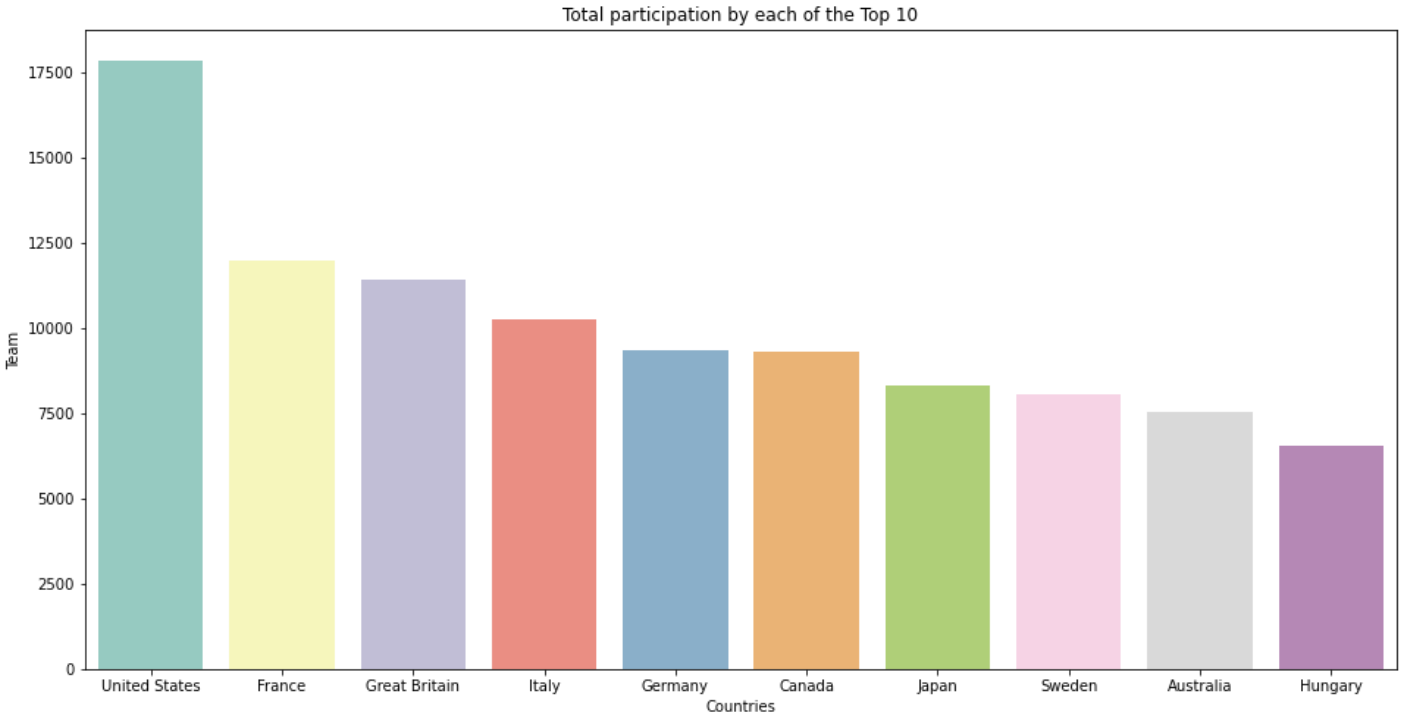
Query-4 :Top 10 participating Countries along with number of participants from each country

```
#Query : Top 10 participating Countries along with number of participants from each cou
Top_ten = athletes_df.Team.value_counts().sort_values(ascending=False).head(10)
print(Top_ten)
```

```
United States    17847
France           11988
Great Britain    11404
Italy            10260
Germany           9326
Canada            9279
Japan             8289
Sweden            8052
Australia         7513
Hungary           6547
Name: Team, dtype: int64
```

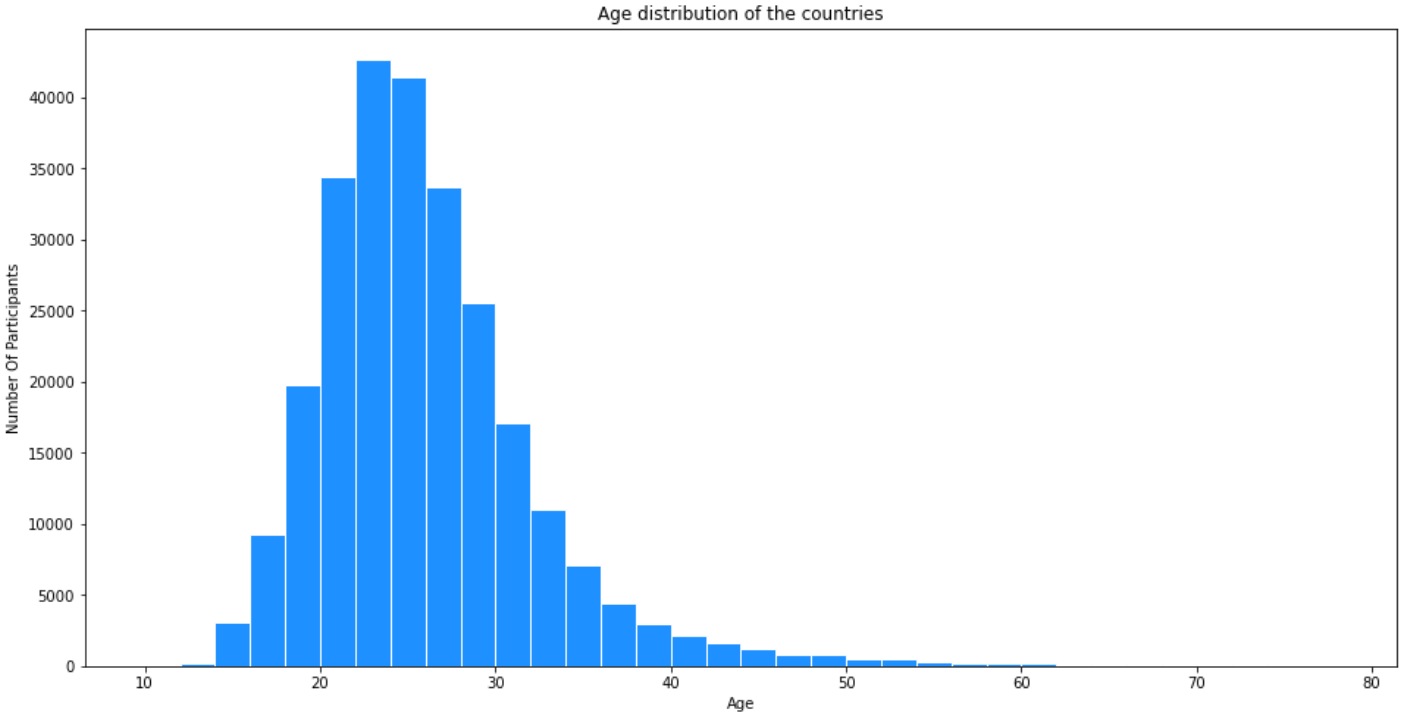
Query-5 :Plot for top 10 countries

```
plt.figure(figsize=(16,8))
plt.title('Total participation by each of the Top 10')
plt.xlabel('Countries')
plt.ylabel('Participants')
sns.barplot(x=Top_ten.index , y=Top_ten , palette='Set3');
```



Query-6 :Age distribution of the participants

```
plt.figure(figsize=(16,8))
plt.title('Age distribution of the countries')
plt.xlabel('Age')
plt.ylabel('Number Of Participants')
plt.hist(athletes_df.Age , bins = np.arange(10,80,2) , color='dodgerblue' , edgecolor='
```



Query-7 :Print all sports played in Winter Olympics

```
Winter_sports =  athletes_df[athletes_df.Season == 'Winter'].Sport.unique()
print('Winter Olympics Sports =>\n')
print(Winter_sports)
```

```
Winter Olympics Sports =>

['Speed Skating' 'Cross Country Skiing' 'Ice Hockey' 'Biathlon'
 'Alpine Skiing' 'Luge' 'Bobsleigh' 'Figure Skating' 'Nordic Combined'
 'Freestyle Skiing' 'Ski Jumping' 'Curling' 'Snowboarding'
 'Short Track Speed Skating' 'Skeleton' 'Military Ski Patrol' 'Alpinism']
```

Query-8 :Print all sports played in Summer Olympics

```
Summer_sports = athletes_df[athletes_df.Season == 'Summer'].Sport.unique()
print('Summer Olympics Sports =>\n')
print(Summer_sports)
```

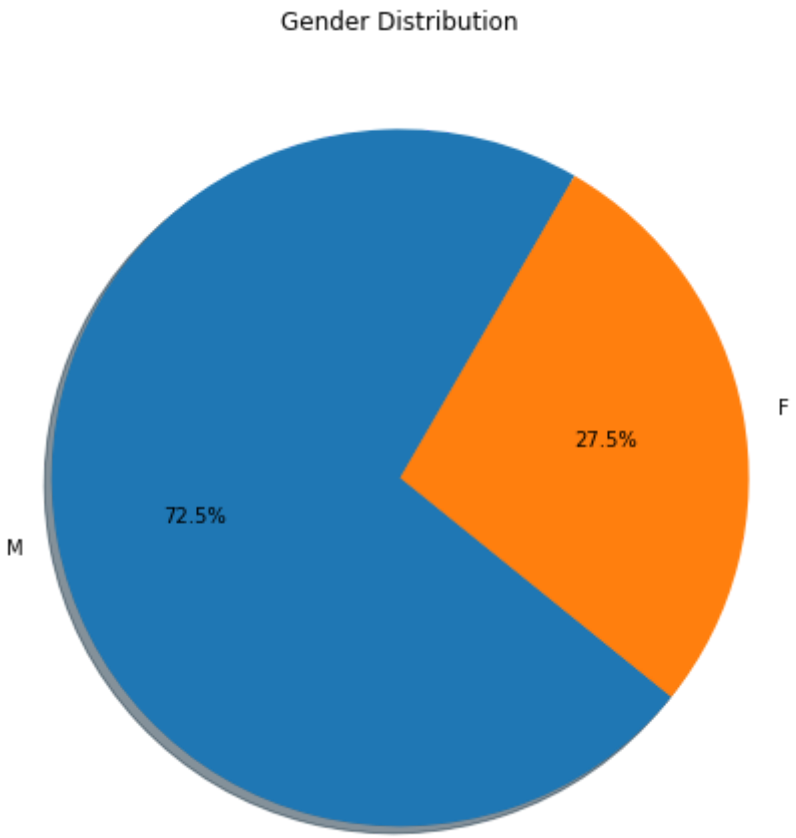
```
Summer Olympics Sports =>

['Basketball' 'Judo' 'Football' 'Tug-Of-War' 'Athletics' 'Swimming'
 'Badminton' 'Sailing' 'Gymnastics' 'Art Competitions' 'Handball'
 'Weightlifting' 'Wrestling' 'Water Polo' 'Hockey' 'Rowing' 'Fencing'
 'Equestrianism' 'Shooting' 'Boxing' 'Taekwondo' 'Cycling' 'Diving'
 'Canoeing' 'Tennis' 'Modern Pentathlon' 'Golf' 'Softball' 'Archery'
 'Volleyball' 'Synchronized Swimming' 'Table Tennis' 'Baseball'
 'Rhythmic Gymnastics' 'Rugby Sevens' 'Trampolining' 'Beach Volleyball'
 'Triathlon' 'Rugby' 'Lacrosse' 'Polo' 'Cricket' 'Ice Hockey' 'Racquets'
 'Motorboating' 'Croquet' 'Figure Skating' 'Jeu De Paume' 'Roque'
 'Basque Pelota' 'Alpinism' 'Aeronautics']
```

```
#Male & Femal Participants
Gender_count = athletes_df.Sex.value_counts()
Gender_count
```

```
M    196594
F     74522
Name: Sex, dtype: int64
```

```
# Pie Plot for Male and Female athletes
plt.figure(figsize=(16,8))
plt.title('Gender Distribution')
plt.pie(Gender_count , labels = Gender_count.index , autopct='%1.1f%%' , startangle=60
```



Query-9 :Total Medals

```
total_medals=athletes_df.Medal[athletes_df.Medal != 'NaN'].count()
medals=athletes_df.Medal.value_counts()
print('Total Medals : {}'.format(total_medals))
medals
```

```
Total Medals : 39783
```

```
Gold      13372
Bronze    13295
Silver    13116
Name: Medal, dtype: int64
```

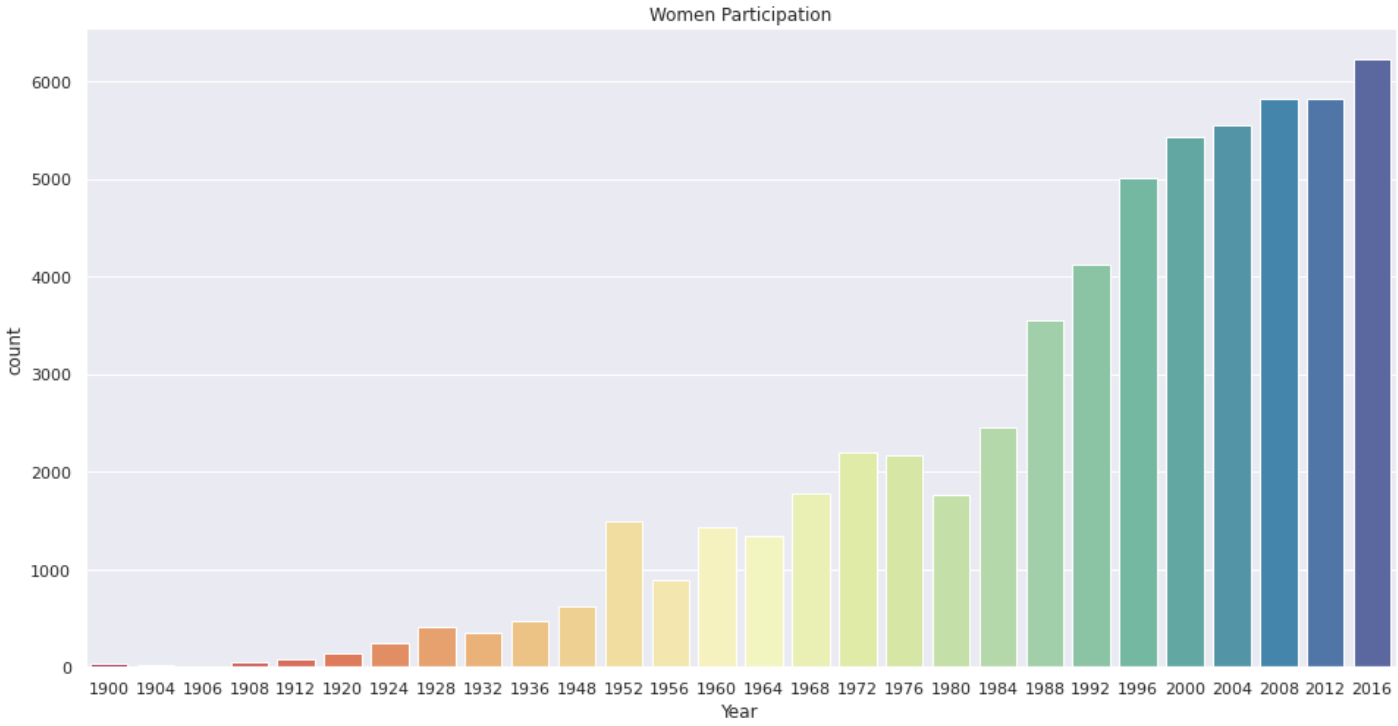
Query-10 :Female Participants in Olympics

```
female_participants = athletes_df[(athletes_df.Sex == 'F') & (athletes_df.Season == 'Su
participants=female_participants.groupby('Year').count().reset_index()
participants
```

	Year	Sex
0	1900	33
1	1904	16
2	1906	11
3	1908	47
4	1912	87

	Year	Sex
5	1920	134
6	1924	244
7	1928	404
8	1932	347
9	1936	468
10	1948	628
11	1952	1497
12	1956	893
13	1960	1435
14	1964	1348
15	1968	1777
16	1972	2193
17	1976	2172
18	1980	1756
19	1984	2447
20	1988	3543
21	1992	4124
22	1996	5008
23	2000	5431
24	2004	5546
25	2008	5816
26	2012	5815
27	2016	6223

```
female_Olympics = athletes_df[(athletes_df.Sex == 'F') & (athletes_df.Season == 'Summer
sns.set(style="darkgrid")
plt.figure(figsize=(16,8))
sns.countplot(x='Year' , data = female_Olympics , palette="Spectral")
plt.title('Women Participation');
```



Query-11 :First 5 Gold Medals Atheletes

```
gold_athletes=athletes_df[athletes_df.Medal == 'Gold']
gold_athletes.head()
```

	ID	Name	Sex	Age	Height	Weight	Team	NOC	Games	Year	Season	City	S
3	4	Edgar Lindenau Aabye	M	34.0	NaN	NaN	Denmark/Sweden	DEN	1900 Summer	1900	Summer	Paris	Tug-Of-
42	17	Paavo Johannes Aaltonen	M	28.0	175.0	64.0	Finland	FIN	1948 Summer	1948	Summer	London	Gymnas
44	17	Paavo Johannes Aaltonen	M	28.0	175.0	64.0	Finland	FIN	1948 Summer	1948	Summer	London	Gymnas
48	17	Paavo Johannes Aaltonen	M	28.0	175.0	64.0	Finland	FIN	1948 Summer	1948	Summer	London	Gymnas
60	20	Kjetil Andr Aamodt	M	20.0	176.0	85.0	Norway	NOR	1992 Winter	1992	Winter	Albertville	Al Sk

```
# Take only the values that are different from NaN
gold_athletes=gold_athletes[np.isfinite(gold_athletes['Age'])]
```

	ID	Name	Sex	Age	Height	Weight	Team	NOC	Games	Year	Season	Ci
3	4	Edgar Lindenau Aabye	M	34.0	NaN	NaN	Denmark/Sweden	DEN	1900 Summer	1900	Summer	Par
42	17	Paavo Johannes Aaltonen	M	28.0	175.0	64.0	Finland	FIN	1948 Summer	1948	Summer	Londo

	ID	Name	Sex	Age	Height	Weight	Team	NOC	Games	Year	Season	Ci
44	17	Paavo Johannes Aaltonen	M	28.0	175.0	64.0	Finland	FIN	1948 Summer	1948	Summer	Londc
48	17	Paavo Johannes Aaltonen	M	28.0	175.0	64.0	Finland	FIN	1948 Summer	1948	Summer	Londc
60	20	Kjetil Andr Aamodt	M	20.0	176.0	85.0	Norway	NOR	1992 Winter	1992	Winter	Albertvil
...	...	...	...	...	...	...	...	...	...	...	...	
270981	135503	Zurab Zviadauri	M	23.0	182.0	90.0	Georgia	GEO	2004 Summer	2004	Summer	Athir
271009	135520	Julia Zwehl	F	28.0	167.0	60.0	Germany	GER	2004 Summer	2004	Summer	Athir
271016	135523	Ronald Ferdinand "Ron" Zwerver	M	29.0	200.0	93.0	Netherlands	NED	1996 Summer	1996	Summer	Atlan
271049	135545	Henk Jan Zwolle	M	31.0	197.0	93.0	Netherlands	NED	1996 Summer	1996	Summer	Atlan
271076	135553	Galina Ivanovna Zybina (-Fyodorova)	F	21.0	168.0	80.0	Soviet Union	URS	1952 Summer	1952	Summer	Helsin

13224 rows × 17 columns

Query-12 :Gold Medalists beyond age of 60

```
print("Number of Gold Medalists beyond age of 60 = {}\n".format(gold_athletes[gold_athl
# Sport events in which they won Gold
sporting_event=gold_athletes[gold_athletes['Age']>60].Sport
sporting_event
```

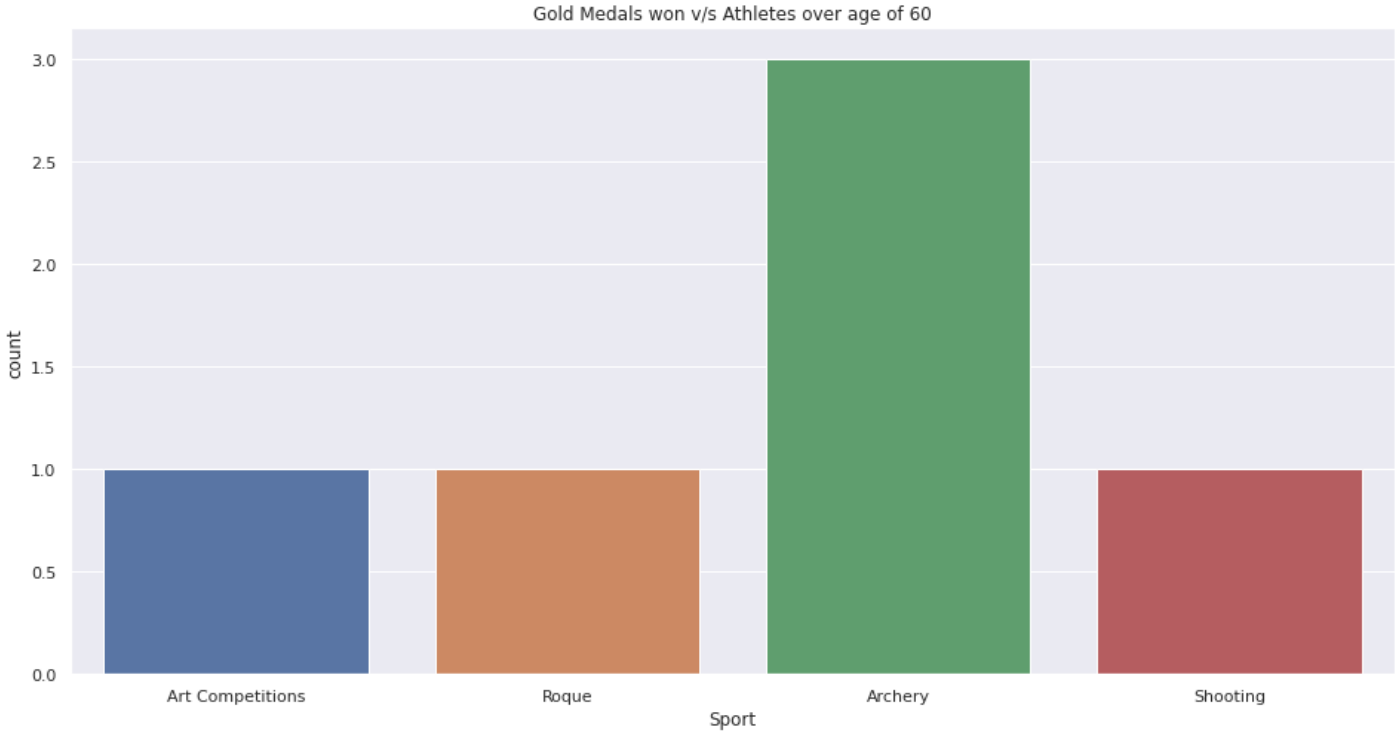
```
Number of Gold Medalists beyond age of 60 = 6


104003    Art Competitions
105199               Roque
190952             Archery
226374             Archery
233390            Shooting
261102             Archery
Name: Sport, dtype: object
```

Query-13 :Plot for the Sporting Event

```
plt.figure(figsize=(16,8))
plt.tight_layout
```

```
sns.countplot(x=sporting_event)
plt.title("Gold Medals won v/s Athletes over age of 60");
```



Query-14 :Total Gold Medals from each Country

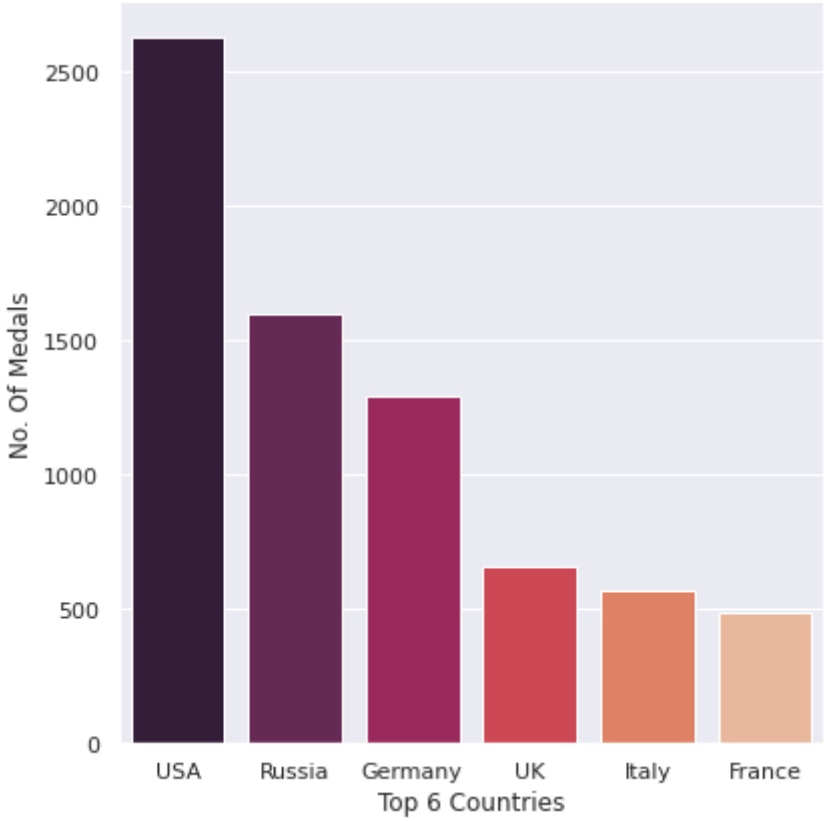
```
gold_athletes.Region.value_counts().reset_index(name='Medal')
```

	index	Medal
0	USA	2627
1	Russia	1599
2	Germany	1293
3	UK	657
4	Italy	567
...	...	...
92	Grenada	1
93	Jordan	1
94	Vietnam	1
95	Ivory Coast	1
96	Tajikistan	1

97 rows × 2 columns

Plot for Gold Medals won by each of Top 6 countries

```
total_gold=gold_athletes.Region.value_counts().reset_index(name='Medal').head(6)
graph=sns.catplot(x = "index",y = "Medal", data = total_gold, kind = "bar",height=6,pal
graph.set_xlabels("Top 6 Countries")
graph.set_ylabels("No. Of Medals");
```



Rio Olympics

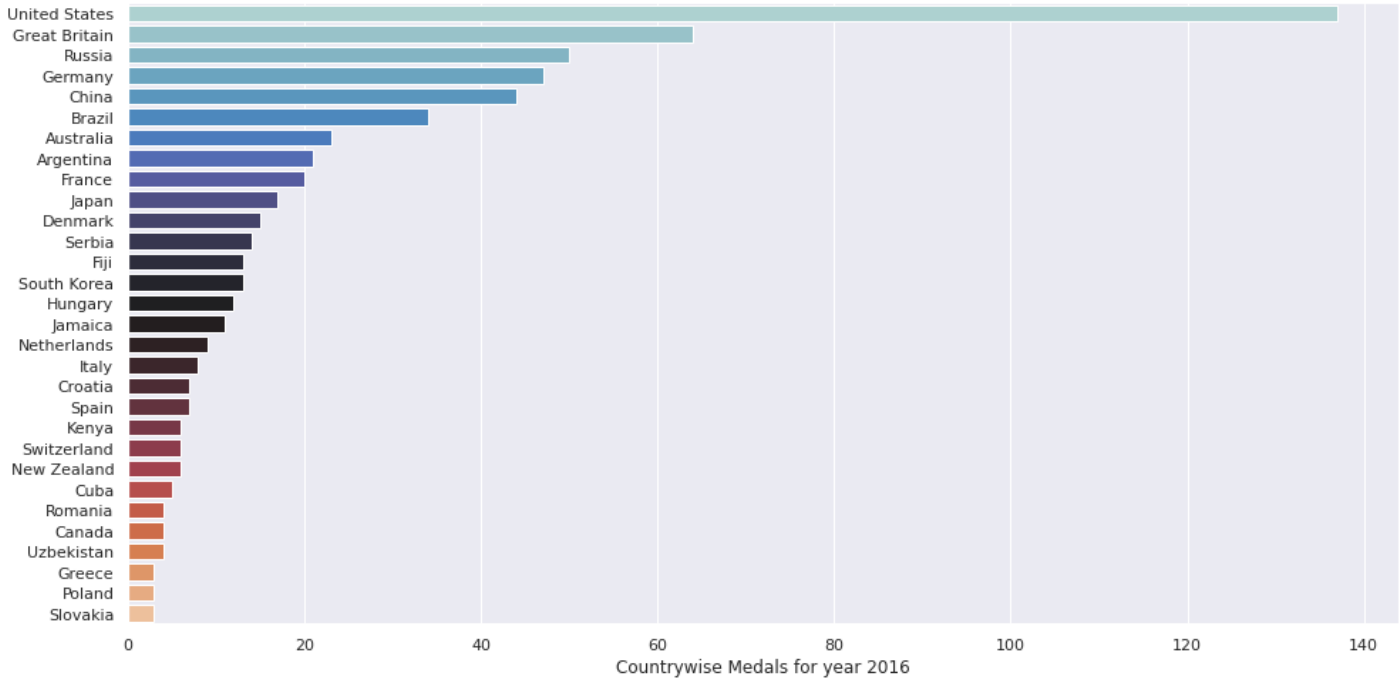
```
max_year = athletes_df.Year.max()
print("Most recent Year of Olympics present in dataset : {}\n".format(max_year))
team_names = athletes_df[(athletes_df.Year == max_year) & (athletes_df.Medal == 'Gold')
# Gold Medals won by each Country in Rio Olympics
team_names.value_counts().reset_index(name="Medals")
```

```
Most recent Year of Olympics present in dataset : 2016
```

	index	Medals
0	United States	137
1	Great Britain	64
2	Russia	50
3	Germany	47
4	China	44
...	...	...
60	Ethiopia	1
61	Armenia	1
62	Individual Olympic Athletes	1
63	Turkey	1
64	Slovenia	1

65 rows × 2 columns

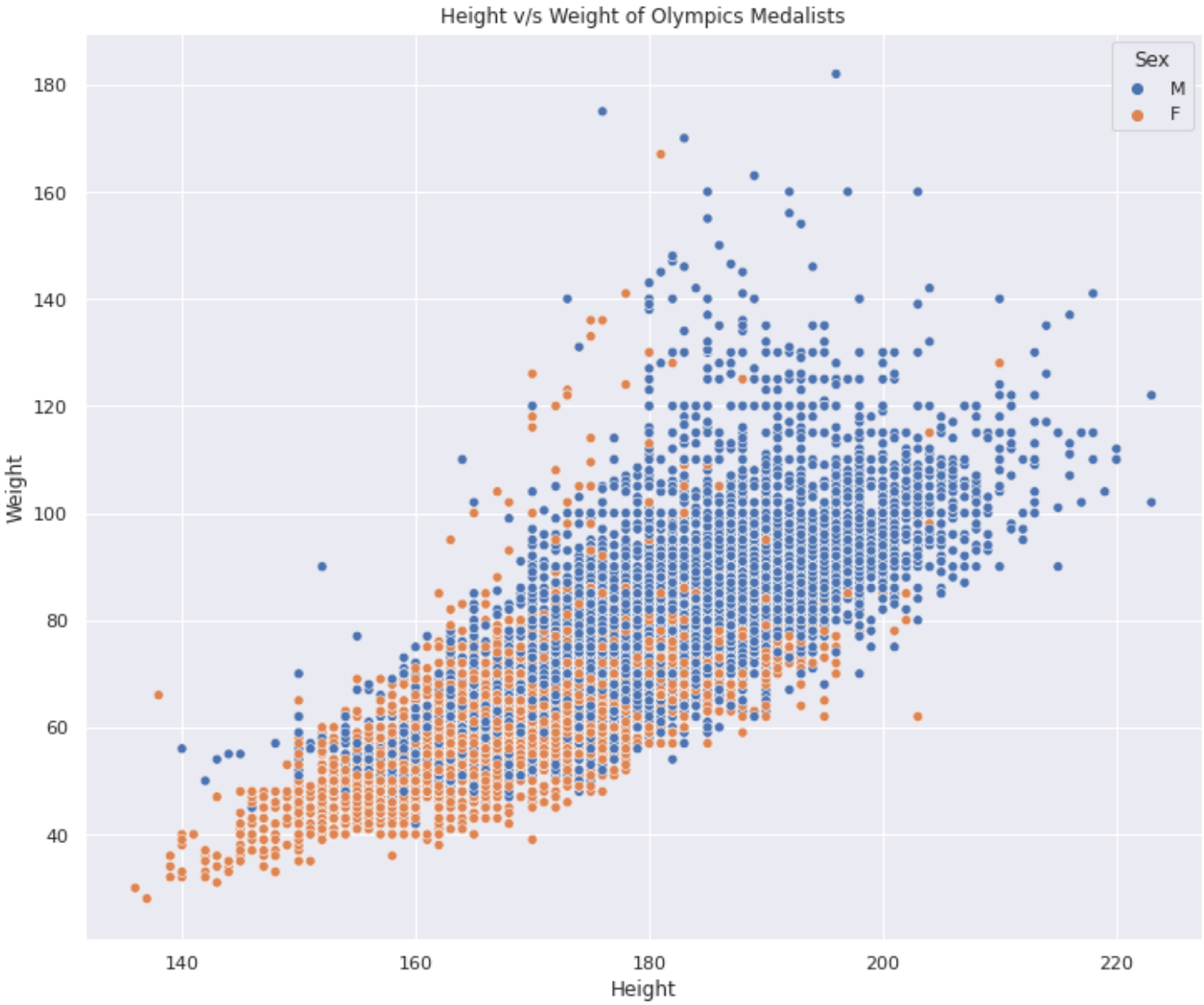
```
rio_gold_team_names=team_names.value_counts().head(30)
plt.figure(figsize=(16,8))
sns.barplot(x=rio_gold_team_names, y=rio_gold_team_names.index, palette="icefire")
plt.ylabel(None)
plt.xlabel('Countrywise Medals for year 2016');
```



Height v/s Weight of Olympics Medalists Plot

```
gold_athletes_df=athletes_df[athletes_df.Medal == 'Gold']
gold_athletes=gold_athletes_df[gold_athletes_df['Height'].notnull() & gold_athletes_df[
silver_athletes_df=athletes_df[athletes_df.Medal == 'Silver']
silver_athletes=silver_athletes_df[silver_athletes_df['Height'].notnull() & silver_athl
bronze_athletes_df=athletes_df[athletes_df.Medal == 'Bronze']
bronze_athletes=bronze_athletes_df[bronze_athletes_df['Height'].notnull() & bronze_athl
frames=[gold_athletes , silver_athletes , bronze_athletes]
resultant=pd.concat(frames)
plt.figure(figsize=(12,10))
axis=sns.scatterplot(x="Height" , y="Weight", data=resultant,hue='Sex')
plt.title("Height v/s Weight of Olympics Medalists")
```

```
Text(0.5, 1.0, 'Height v/s Weight of Olympics Medalists')
```



Conclusion

```
This project helped me to  understand how to analyze and visualize past Olympics data f
use of Python libraries and their associated functions for creating interesting visuali
```

Some Useful Links & References for the Project

https://pandas.pydata.org/

https://seaborn.pydata.org/tutorial/color_palettes.html

https://pandas.pydata.org/pandas-docs/version/0.20/merging.html