1. Step 1: Preparing for Your Proposal

1. Which client/dataset did you select and why?

Client 3: SportsStats (Olympics Dataset - 120 years of data)

SportsStats is a sports analysis firm partnering with local news and elite personal trainers to provide “interesting” insights to help their partners. Insights could be patterns/trends highlighting certain groups/events/countries, etc. for the purpose of developing a news story or discovering key health insights.

Using this dataset to get some insights on games.

2. Describe the steps you took to import and clean the data.

Downloaded the data from

https://www.dropbox.com/sh/0wqw8fmiwrzr8ef/AABQijjQM522INXX1FCdamzma?dl=0

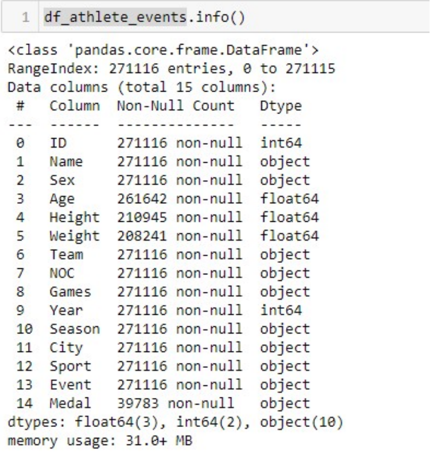
The source provides two tables.

1. athlete_events.csv and

2. noc_regions.csv

```
Number of records athlete_events: 271116
Number of records noc_regions: 230
```

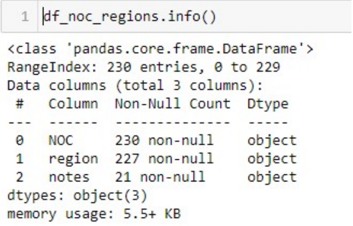
Columns and data type for athlete_events table.



```
df_athlete_events.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 271116 entries, 0 to 271115
Data columns (total 15 columns):
 #   Column  Non-Null Count   Dtype
---  ------  --------------   -----
 0   ID      271116 non-null  int64
 1   Name    271116 non-null  object
 2   Sex     271116 non-null  object
 3   Age     261642 non-null  float64
 4   Height  210945 non-null  float64
 5   Weight  208241 non-null  float64
 6   Team    271116 non-null  object
 7   NOC     271116 non-null  object
 8   Games   271116 non-null  object
 9   Year    271116 non-null  int64
 10  Season  271116 non-null  object
 11  City    271116 non-null  object
 12  Sport   271116 non-null  object
 13  Event   271116 non-null  object
 14  Medal   39783 non-null   object
dtypes: float64(3), int64(2), object(10)
memory usage: 31.0+ MB
```

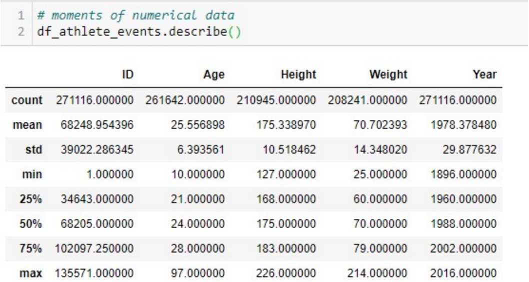
Columns and data type for noc_region.



```
df_noc_regions.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 230 entries, 0 to 229
Data columns (total 3 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   NOC     230 non-null    object
 1   region  227 non-null    object
 2   notes   21 non-null     object
dtypes: object(3)
memory usage: 5.5+ KB
```

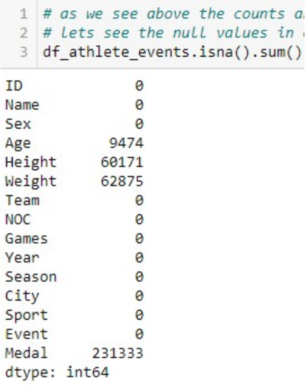
3. Perform initial exploration of data and provide some screenshots or display some stats of the data you are looking at.



```
# moments of numerical data
df_athlete_events.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

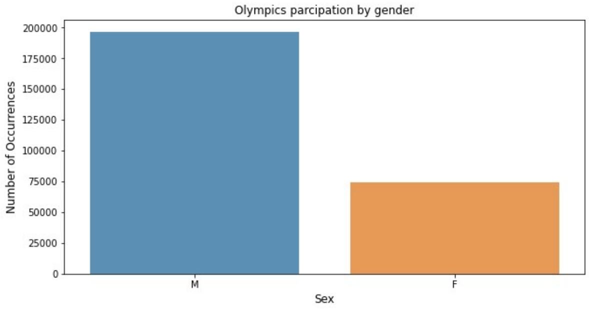
Missing values in athlete_events table

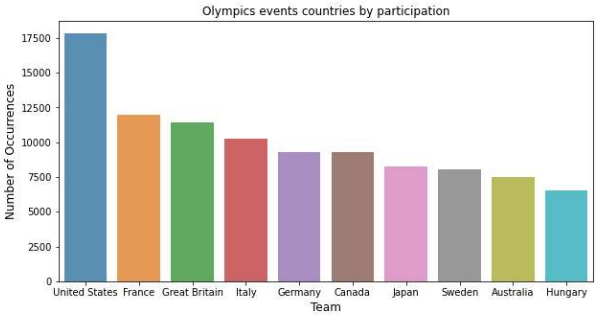


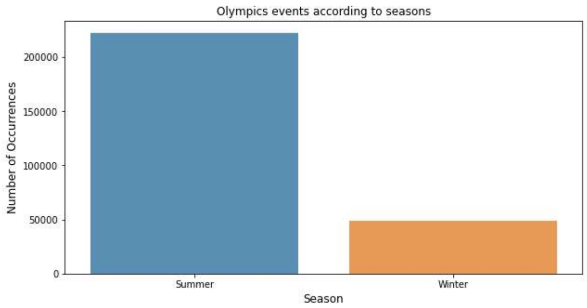
```
# as we see above the counts a
# lets see the null values in
df_athlete_events.isna().sum()
```

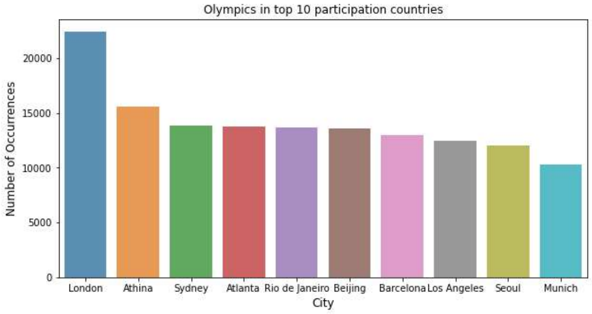
```
ID             0
Name           0
Sex            0
Age         9474
Height     60171
Weight     62875
Team           0
NOC            0
Games          0
Year           0
Season         0
City           0
Sport          0
Event          0
Medal     231333
dtype: int64
```

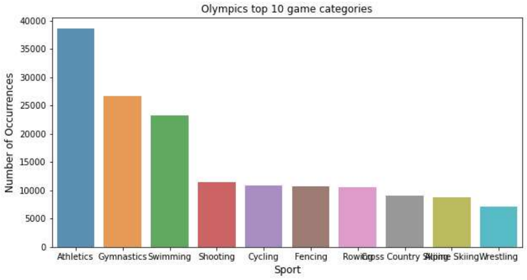
Visualizing categorical values in athlete_events table










Olympics top 10 game categories
Number of Occurrences
40000
35000
30000
25000
20000
15000
10000
5000
0
Athletics
Gymnastics
Swimming
Shooting
Cycling
Fencing
Rowing
Cross Country Skiing
Alpine Skiing
Wrestling
Sport

4. Entity diagrams

Initial ER diagram for athlete_events table

```
#   Column  Non-Null Count   Dtype
---  ------  --------------   -----
 0   ID      271116 non-null  int64
 1   Name    271116 non-null  object
 2   Sex     271116 non-null  object
 3   Age     261642 non-null  float64
 4   Height  210945 non-null  float64
 5   Weight  208241 non-null  float64
 6   Team    271116 non-null  object
 7   NOC     271116 non-null  object
 8   Games   271116 non-null  object
 9   Year    271116 non-null  int64
 10  Season  271116 non-null  object
 11  City    271116 non-null  object
 12  Sport   271116 non-null  object
 13  Event   271116 non-null  object
 14  Medal   39783 non-null   object
```

Initial ER diagram for noc_regions table

```
#   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   NOC     230 non-null    object
 1   region  227 non-null    object
 2   notes   21 non-null     object
```

Step 2: Develop Project Proposal

There are two tables provided. Athlete events table contains most of the information's which later further split into multiple tables like

1. countries,
2. sports,
3. medals,
4. categories.

Questions

1. are the winning countries make more medals in future games. Did they find the secret of winning?
2. Are there new countries emerging with more than 10 medals?

Hypothesis

1. The athletes who participate more Olympic games wins more medals in later games.

2. The motivation outweighs physical characteristics to win the game is the hypothesis.

Approach

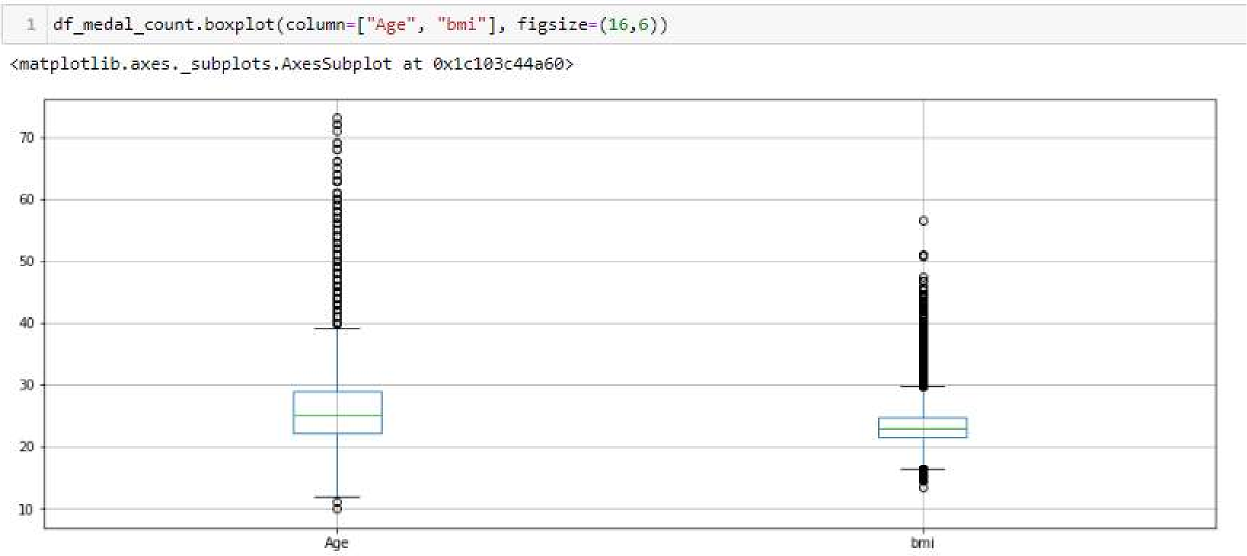
1. All numerical and categorical fields will be used.

2. Find the relation between physical characteristics with respect to winning the game

3. physical characteristics for top 5 games across years to see how it varies?

The bmi is calculated using a standard formula (weight/height**2)

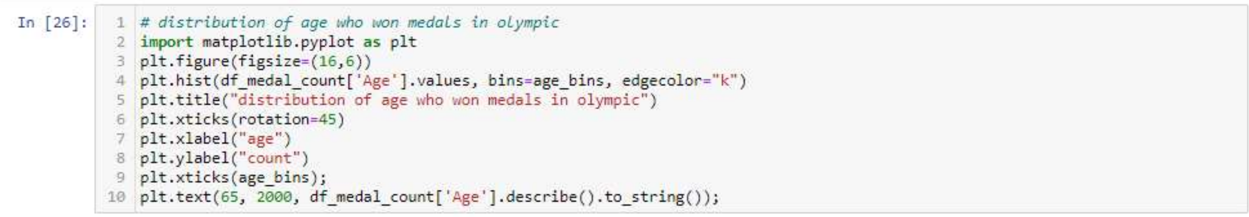
Following shows the box plot for age and bmi. And we can see the both distributions are right tailed.



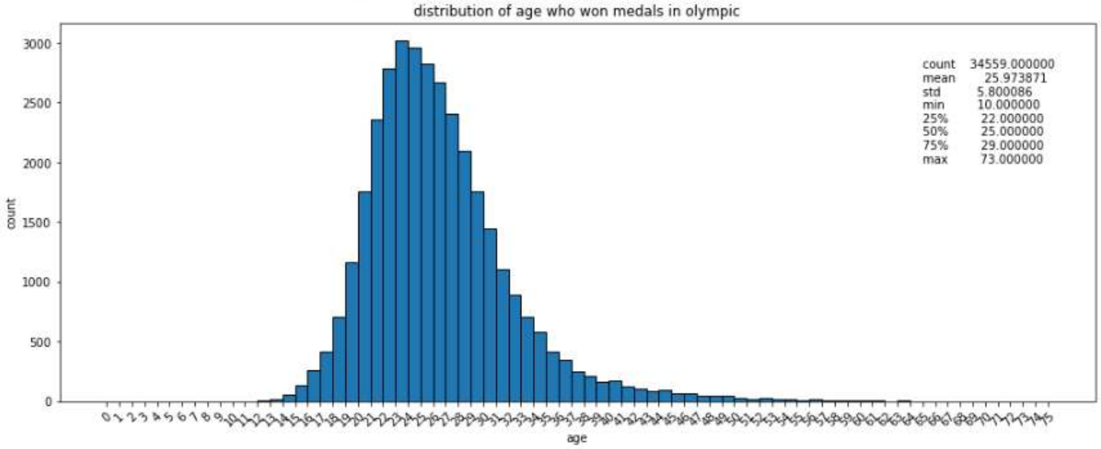
```
df_medal_count.boxplot(column=["Age", "bmi"], figsize=(16,6))
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1c103c44a60>
```

Distribution of Age and Winning a medal, The distribution is right tailed.



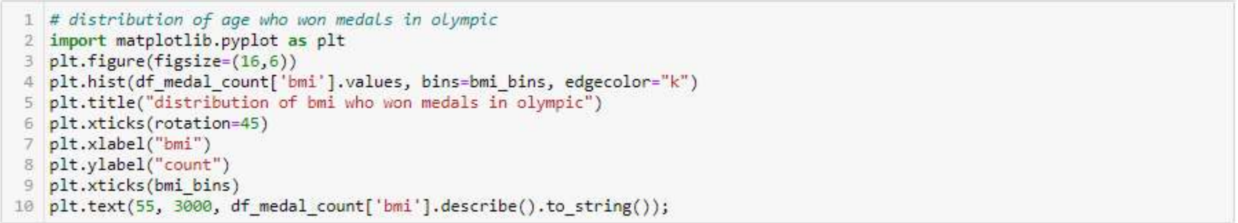
```python
# distribution of age who won medals in olympic
import matplotlib.pyplot as plt
plt.figure(figsize=(16,6))
plt.hist(df_medal_count['Age'].values, bins=age_bins, edgecolor="k")
plt.title("distribution of age who won medals in olympic")
plt.xticks(rotation=45)
plt.xlabel("age")
plt.ylabel("count")
plt.xticks(age_bins);
plt.text(65, 2000, df_medal_count['Age'].describe().to_string());
```



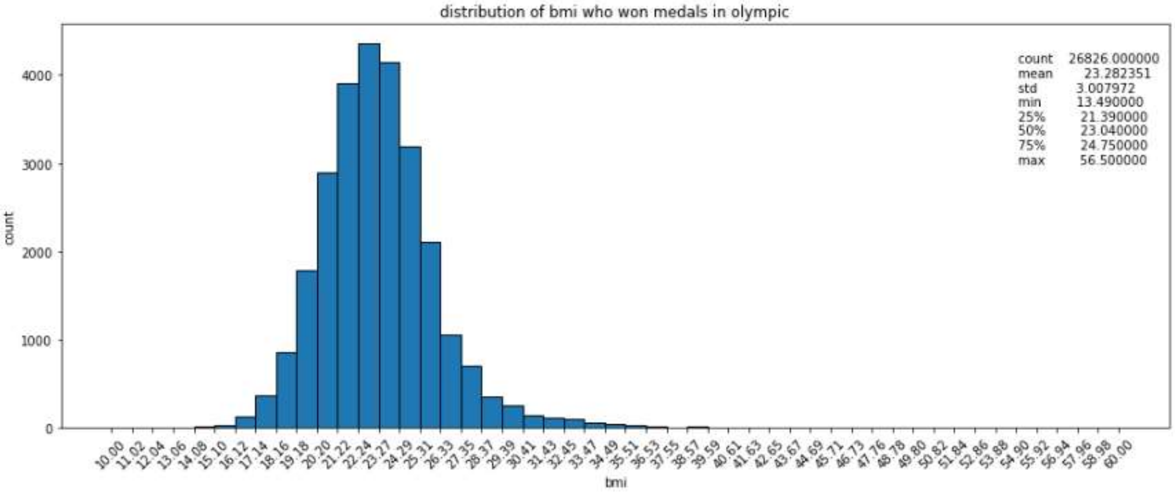
```
count   34559.000000
mean       25.973871
std         5.800086
min        10.000000
25%        22.000000
50%        25.000000
75%        29.000000
max        73.000000
```

Interesting to note that the youngest winner is 10 year old and oldest winner is 73 year old.

Distribution of BMI and Winning a Medal, this distribution also right tailed.



```python
# distribution of age who won medals in olympic
import matplotlib.pyplot as plt
plt.figure(figsize=(16,6))
plt.hist(df_medal_count['bmi'].values, bins=bmi_bins, edgecolor="k")
plt.title("distribution of bmi who won medals in olympic")
plt.xticks(rotation=45)
plt.xlabel("bmi")
plt.ylabel("count")
plt.xticks(bmi_bins)
plt.text(55, 3000, df_medal_count['bmi'].describe().to_string());
```

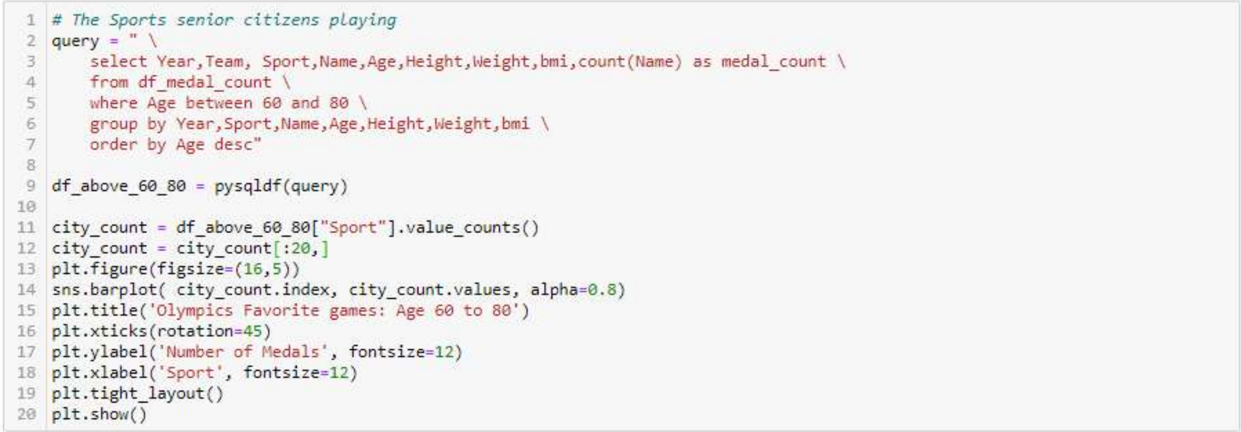


```
count    26826.000000
mean        23.282351
std          3.007972
min         13.490000
25%         21.390000
50%         23.040000
75%         24.750000
max         56.500000
```

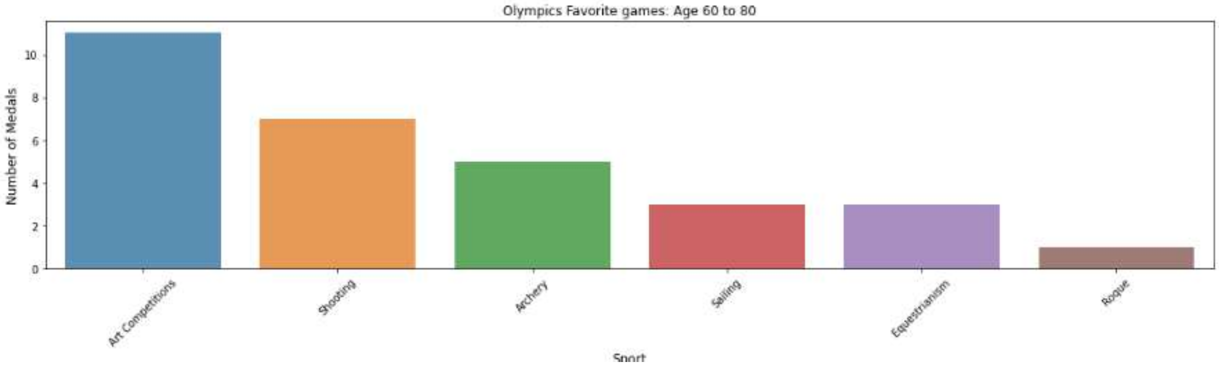
There is a game for everyone i.e, underweight, normal, overweight, and obese.

From BMI 13 underweight to BMI 56 obese has games to win.

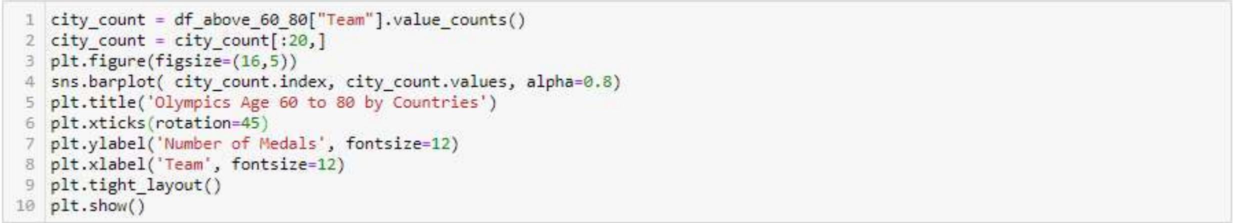
From age 60-80 years old participated in games and won the medals.



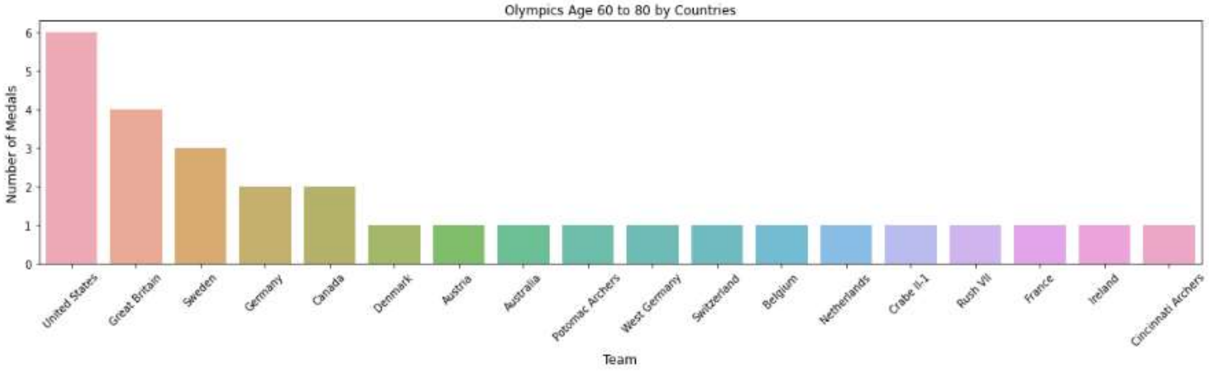
```
# The Sports senior citizens playing
query = " \
    select Year,Team, Sport,Name,Age,Height,Weight,bmi,count(Name) as medal_count \
    from df_medal_count \
    where Age between 60 and 80 \
    group by Year,Sport,Name,Age,Height,Weight,bmi \
    order by Age desc"

df_above_60_80 = pysqldf(query)

city_count = df_above_60_80["Sport"].value_counts()
city_count = city_count[:20,]
plt.figure(figsize=(16,5))
sns.barplot( city_count.index, city_count.values, alpha=0.8)
plt.title('Olympics Favorite games: Age 60 to 80')
plt.xticks(rotation=45)
plt.ylabel('Number of Medals', fontsize=12)
plt.xlabel('Sport', fontsize=12)
plt.tight_layout()
plt.show()
```



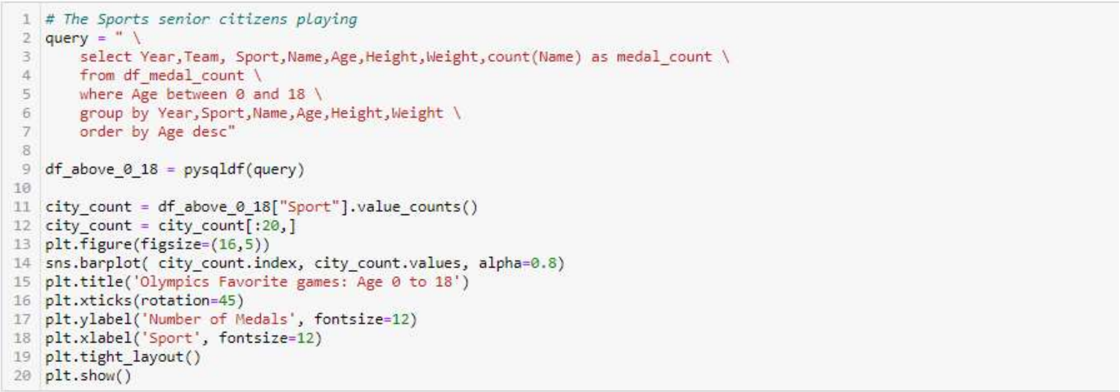
From age 60-80 years represented following teams and won gold.



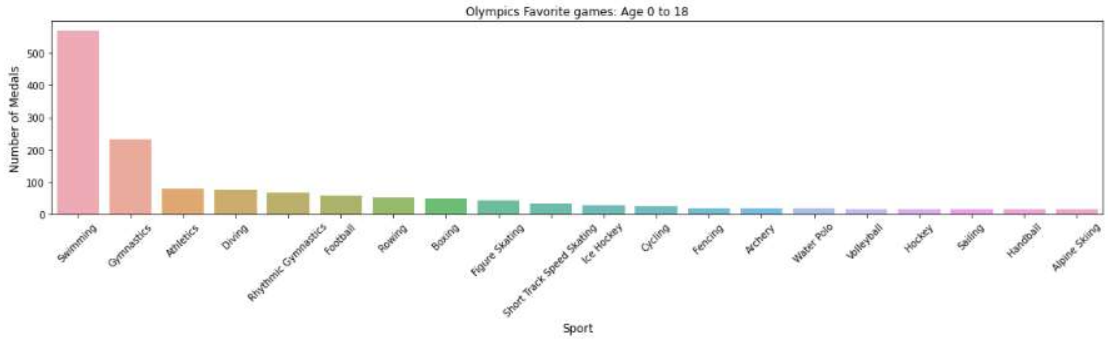
```
city_count = df_above_60_80["Team"].value_counts()
city_count = city_count[:20,]
plt.figure(figsize=(16,5))
sns.barplot( city_count.index, city_count.values, alpha=0.8)
plt.title('Olympics Age 60 to 80 by Countries')
plt.xticks(rotation=45)
plt.ylabel('Number of Medals', fontsize=12)
plt.xlabel('Team', fontsize=12)
plt.tight_layout()
plt.show()
```



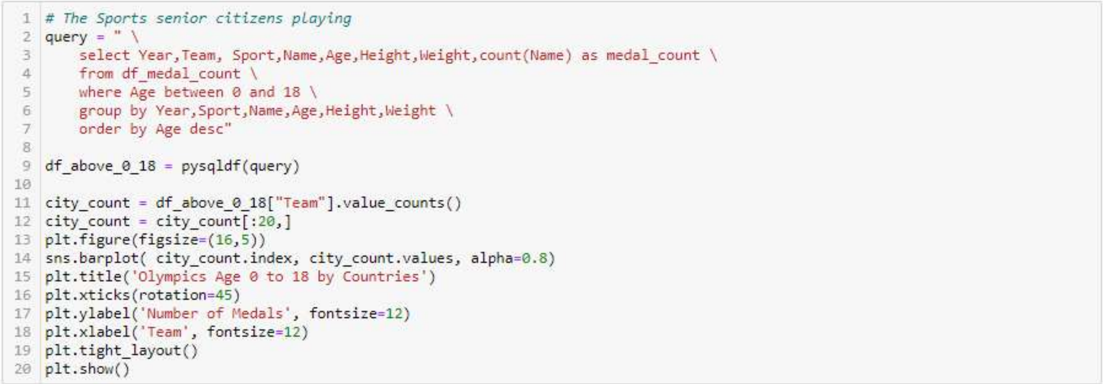
Teenager’s also played in following games



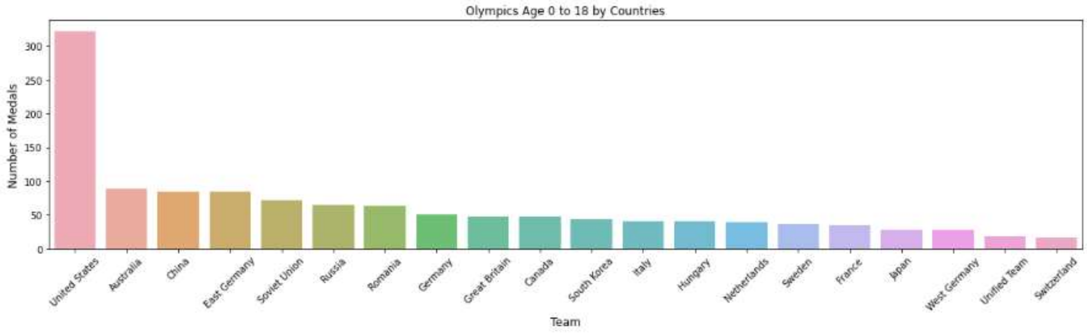
```
# The Sports senior citizens playing
query = " \
    select Year,Team, Sport,Name,Age,Height,Weight,count(Name) as medal_count \
    from df_medal_count \
    where Age between 0 and 18 \
    group by Year,Sport,Name,Age,Height,Weight \
    order by Age desc"

df_above_0_18 = pysqldf(query)

city_count = df_above_0_18["Sport"].value_counts()
city_count = city_count[:20,]
plt.figure(figsize=(16,5))
sns.barplot( city_count.index, city_count.values, alpha=0.8)
plt.title('Olympics Favorite games: Age 0 to 18')
plt.xticks(rotation=45)
plt.ylabel('Number of Medals', fontsize=12)
plt.xlabel('Sport', fontsize=12)
plt.tight_layout()
plt.show()
```



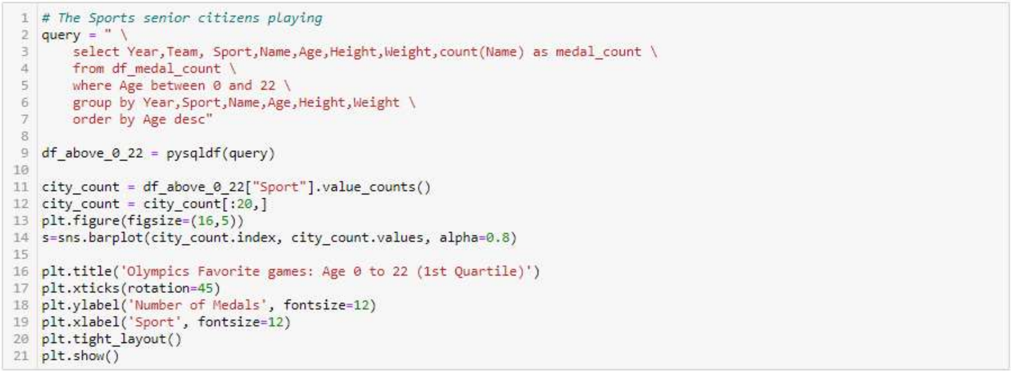
Teenager’s represented following Teams



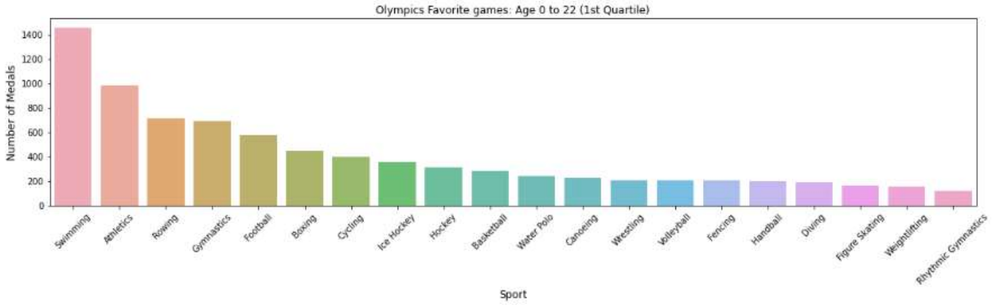
```
# The Sports senior citizens playing
query = " \
    select Year,Team, Sport,Name,Age,Height,Weight,count(Name) as medal_count \
    from df_medal_count \
    where Age between 0 and 18 \
    group by Year,Sport,Name,Age,Height,Weight \
    order by Age desc"

df_above_0_18 = pysqldf(query)

city_count = df_above_0_18["Team"].value_counts()
city_count = city_count[:20,]
plt.figure(figsize=(16,5))
sns.barplot( city_count.index, city_count.values, alpha=0.8)
plt.title('Olympics Age 0 to 18 by Countries')
plt.xticks(rotation=45)
plt.ylabel('Number of Medals', fontsize=12)
plt.xlabel('Team', fontsize=12)
plt.tight_layout()
plt.show()
```



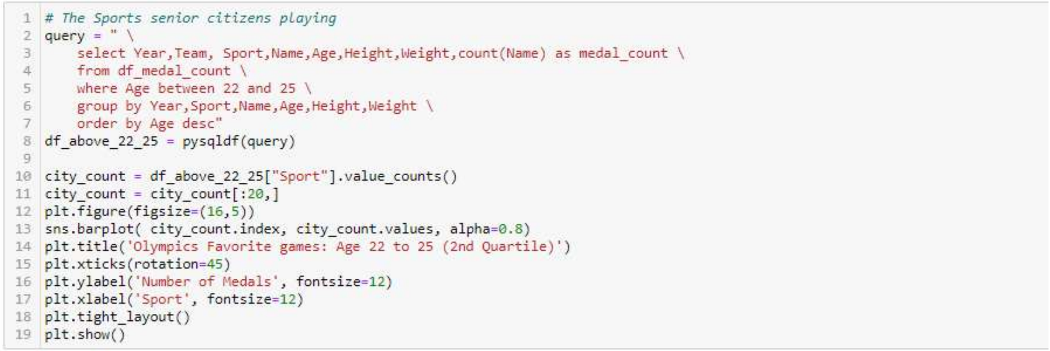
The 1st Quartile of Age group won in following games.



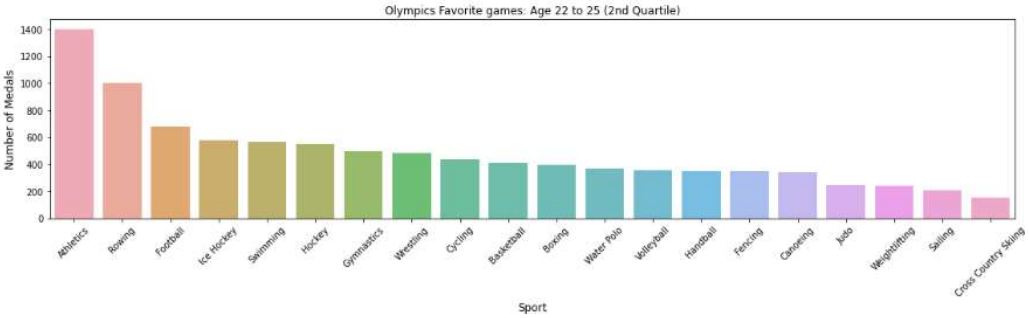
```
# The Sports senior citizens playing
query = " \
    select Year,Team, Sport,Name,Age,Height,Weight,count(Name) as medal_count \
    from df_medal_count \
    where Age between 0 and 22 \
    group by Year,Sport,Name,Age,Height,Weight \
    order by Age desc"

df_above_0_22 = pysqldf(query)

city_count = df_above_0_22["Sport"].value_counts()
city_count = city_count[:20,]
plt.figure(figsize=(16,5))
s=sns.barplot(city_count.index, city_count.values, alpha=0.8)

plt.title('Olympics Favorite games: Age 0 to 22 (1st Quartile)')
plt.xticks(rotation=45)
plt.ylabel('Number of Medals', fontsize=12)
plt.xlabel('Sport', fontsize=12)
plt.tight_layout()
plt.show()
```



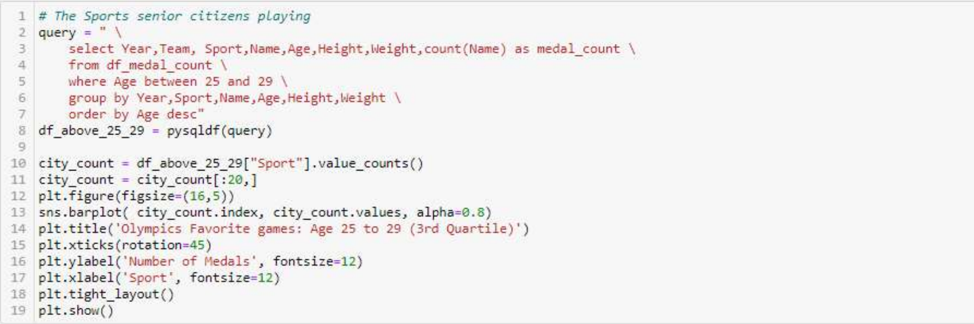
2nd Quartile of Age group won following games



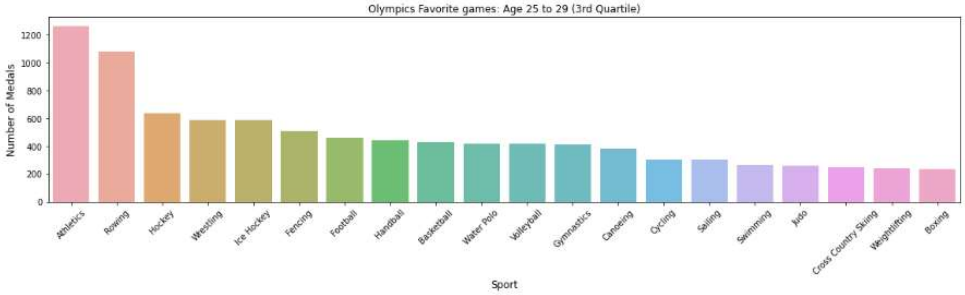
```
# The Sports senior citizens playing
query = " \
    select Year,Team, Sport,Name,Age,Height,Weight,count(Name) as medal_count \
    from df_medal_count \
    where Age between 22 and 25 \
    group by Year,Sport,Name,Age,Height,Weight \
    order by Age desc"
df_above_22_25 = pysqldf(query)

city_count = df_above_22_25["Sport"].value_counts()
city_count = city_count[:20,]
plt.figure(figsize=(16,5))
sns.barplot( city_count.index, city_count.values, alpha=0.8)
plt.title('Olympics Favorite games: Age 22 to 25 (2nd Quartile)')
plt.xticks(rotation=45)
plt.ylabel('Number of Medals', fontsize=12)
plt.xlabel('Sport', fontsize=12)
plt.tight_layout()
plt.show()
```



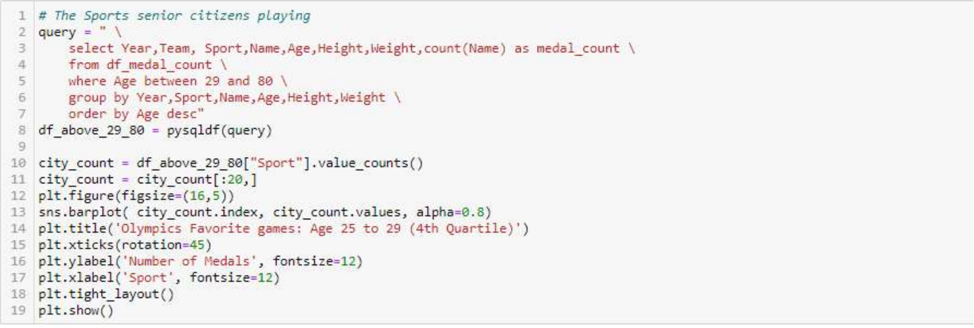
3rd Quartile of Age group won following games.



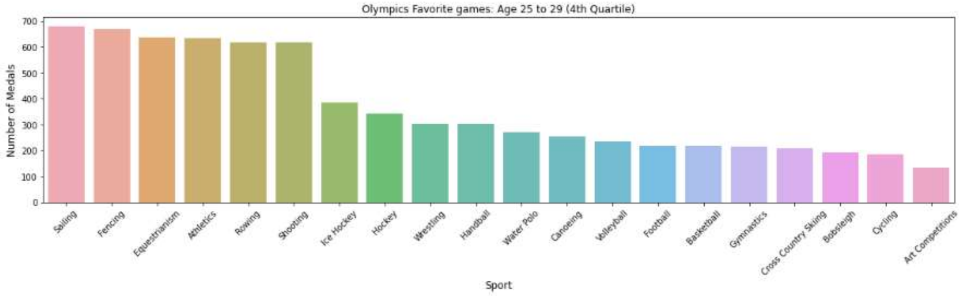
```
# The Sports senior citizens playing
query = " \
    select Year,Team, Sport,Name,Age,Height,Weight,count(Name) as medal_count \
    from df_medal_count \
    where Age between 25 and 29 \
    group by Year,Sport,Name,Age,Height,Weight \
    order by Age desc"
df_above_25_29 = pysqldf(query)

city_count = df_above_25_29["Sport"].value_counts()
city_count = city_count[:20,]
plt.figure(figsize=(16,5))
sns.barplot( city_count.index, city_count.values, alpha=0.8)
plt.title('Olympics Favorite games: Age 25 to 29 (3rd Quartile)')
plt.xticks(rotation=45)
plt.ylabel('Number of Medals', fontsize=12)
plt.xlabel('Sport', fontsize=12)
plt.tight_layout()
plt.show()
```



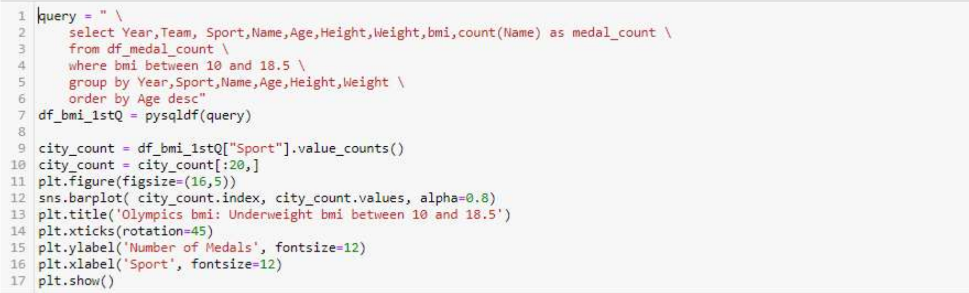
The 4th Quartile won in following games.



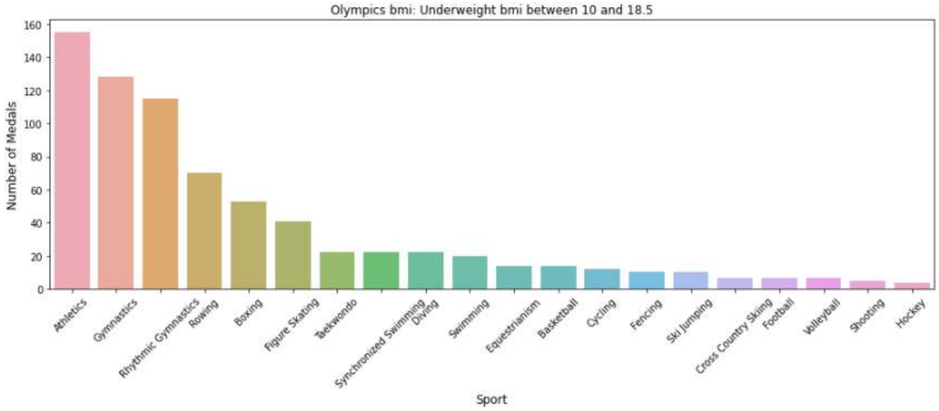
```
# The Sports senior citizens playing
query = " \
    select Year,Team, Sport,Name,Age,Height,Weight,count(Name) as medal_count \
    from df_medal_count \
    where Age between 29 and 80 \
    group by Year,Sport,Name,Age,Height,Weight \
    order by Age desc"
df_above_29_80 = pysqldf(query)

city_count = df_above_29_80["Sport"].value_counts()
city_count = city_count[:20,]
plt.figure(figsize=(16,5))
sns.barplot( city_count.index, city_count.values, alpha=0.8)
plt.title('Olympics Favorite games: Age 25 to 29 (4th Quartile)')
plt.xticks(rotation=45)
plt.ylabel('Number of Medals', fontsize=12)
plt.xlabel('Sport', fontsize=12)
plt.tight_layout()
plt.show()
```



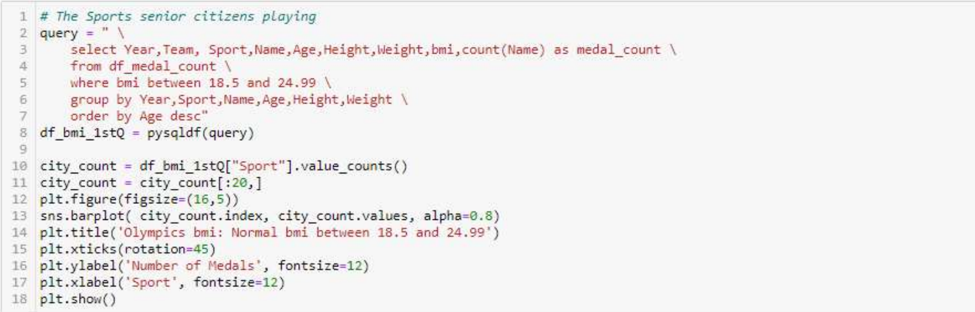
The 1st Quartile of bmi group won in following games.



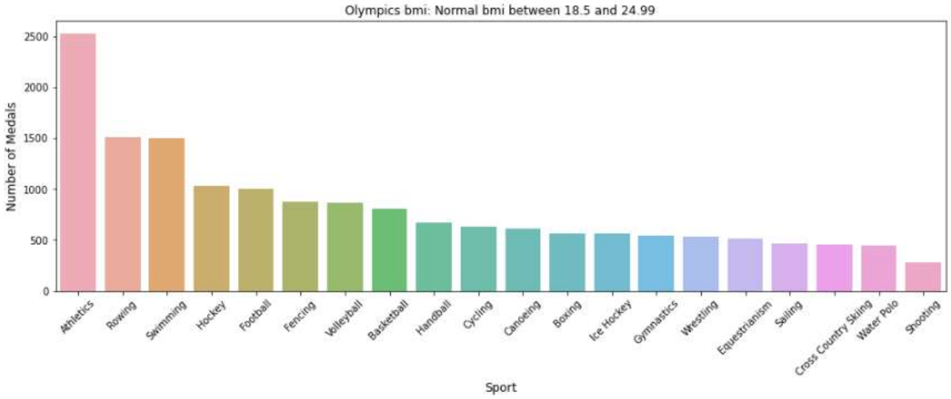
```
query = " \
    select Year,Team, Sport,Name,Age,Height,Weight,bmi,count(Name) as medal_count \
    from df_medal_count \
    where bmi between 10 and 18.5 \
    group by Year,Sport,Name,Age,Height,Weight \
    order by Age desc"
df_bmi_1stQ = pysqldf(query)

city_count = df_bmi_1stQ["Sport"].value_counts()
city_count = city_count[:20,]
plt.figure(figsize=(16,5))
sns.barplot( city_count.index, city_count.values, alpha=0.8)
plt.title('Olympics bmi: Underweight bmi between 10 and 18.5')
plt.xticks(rotation=45)
plt.ylabel('Number of Medals', fontsize=12)
plt.xlabel('Sport', fontsize=12)
plt.show()
```



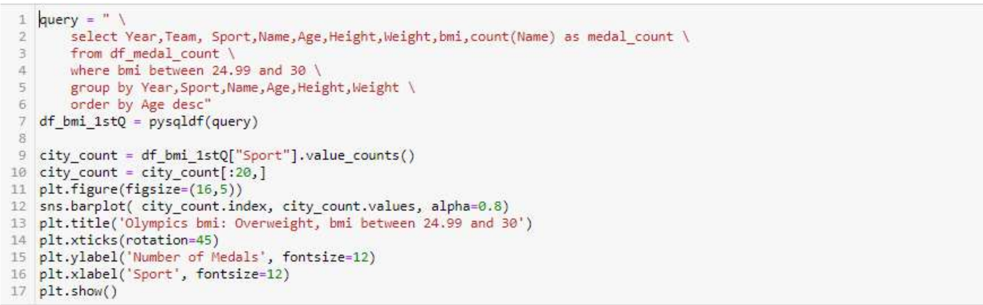
The 2nd Quartile of bmi group won in following games



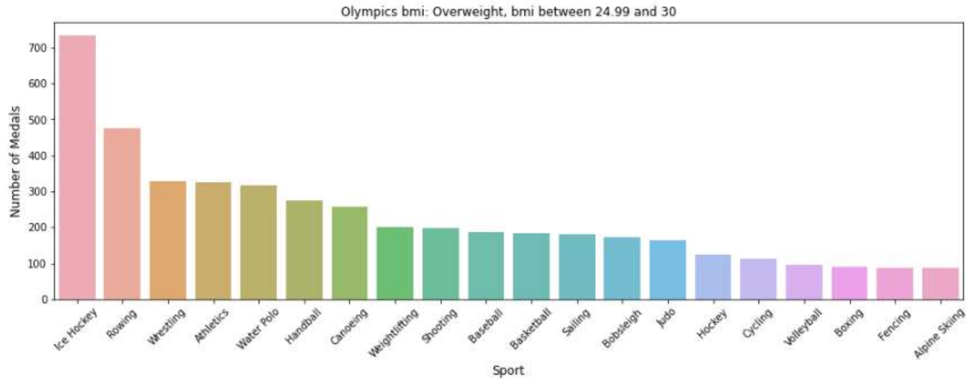
```
# The Sports senior citizens playing
query = " \
    select Year,Team, Sport,Name,Age,Height,Weight,bmi,count(Name) as medal_count \
    from df_medal_count \
    where bmi between 18.5 and 24.99 \
    group by Year,Sport,Name,Age,Height,Weight \
    order by Age desc"
df_bmi_1stQ = pysqldf(query)

city_count = df_bmi_1stQ["Sport"].value_counts()
city_count = city_count[:20,]
plt.figure(figsize=(16,5))
sns.barplot( city_count.index, city_count.values, alpha=0.8)
plt.title('Olympics bmi: Normal bmi between 18.5 and 24.99')
plt.xticks(rotation=45)
plt.ylabel('Number of Medals', fontsize=12)
plt.xlabel('Sport', fontsize=12)
plt.show()
```



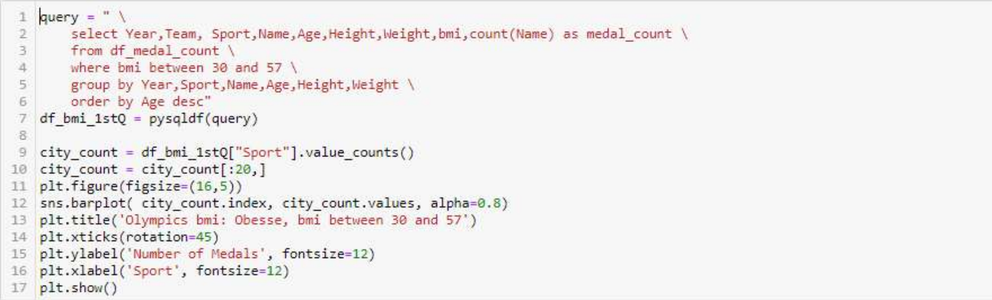
The 3rd Quartile of bmi group won in following games



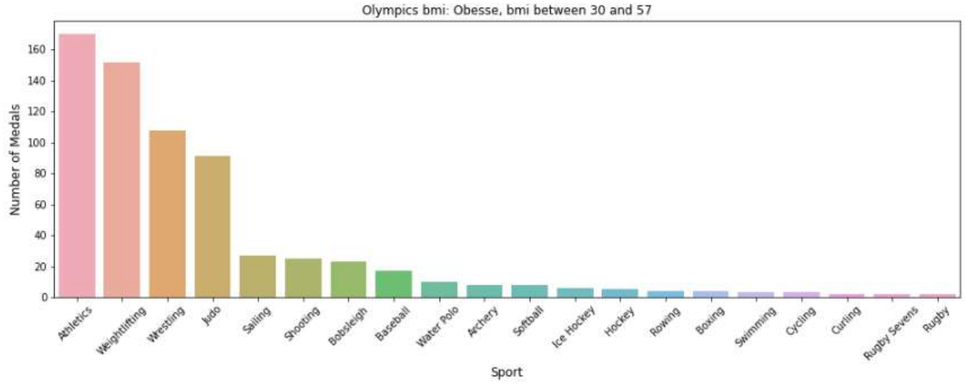
```
query = " \
    select Year,Team, Sport,Name,Age,Height,Weight,bmi,count(Name) as medal_count \
    from df_medal_count \
    where bmi between 24.99 and 30 \
    group by Year,Sport,Name,Age,Height,Weight \
    order by Age desc"
df_bmi_1stQ = pysqldf(query)

city_count = df_bmi_1stQ["Sport"].value_counts()
city_count = city_count[:20,]
plt.figure(figsize=(16,5))
sns.barplot( city_count.index, city_count.values, alpha=0.8)
plt.title('Olympics bmi: Overweight, bmi between 24.99 and 30')
plt.xticks(rotation=45)
plt.ylabel('Number of Medals', fontsize=12)
plt.xlabel('Sport', fontsize=12)
plt.show()
```



The 4th Quartile of bmi group won in following games

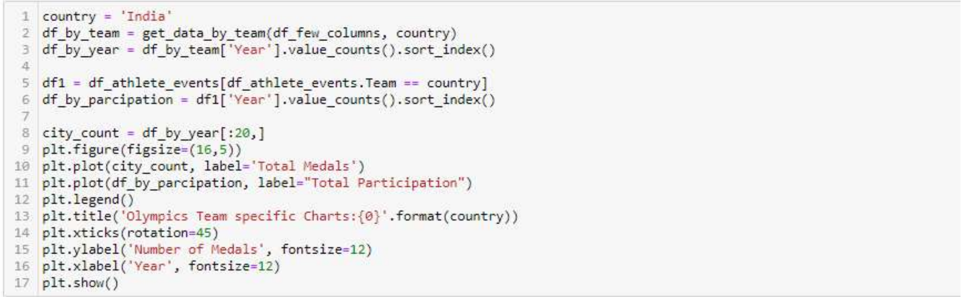


```
query = " \
    select Year,Team, Sport,Name,Age,Height,Weight,bmi,count(Name) as medal_count \
    from df_medal_count \
    where bmi between 30 and 57 \
    group by Year,Sport,Name,Age,Height,Weight \
    order by Age desc"
df_bmi_1stQ = pysqldf(query)

city_count = df_bmi_1stQ["Sport"].value_counts()
city_count = city_count[:20,]
plt.figure(figsize=(16,5))
sns.barplot( city_count.index, city_count.values, alpha=0.8)
plt.title('Olympics bmi: Obesse, bmi between 30 and 57')
plt.xticks(rotation=45)
plt.ylabel('Number of Medals', fontsize=12)
plt.xlabel('Sport', fontsize=12)
plt.show()
```

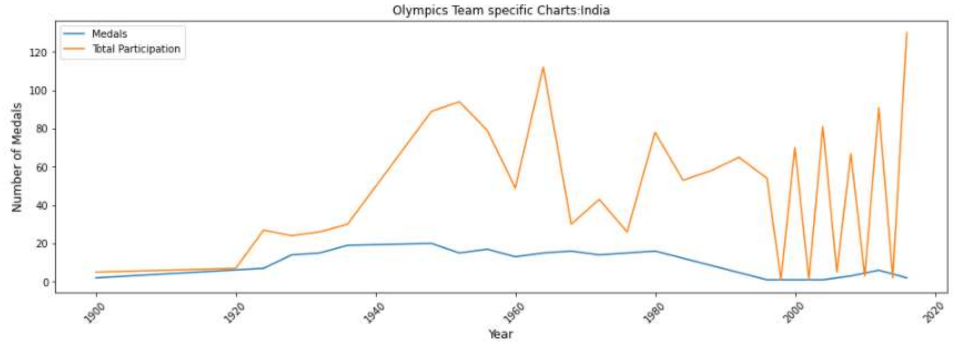


Few games shows more participation brings more winning. But for few other countries, the relation not showing. So the hypothesis is partially true.

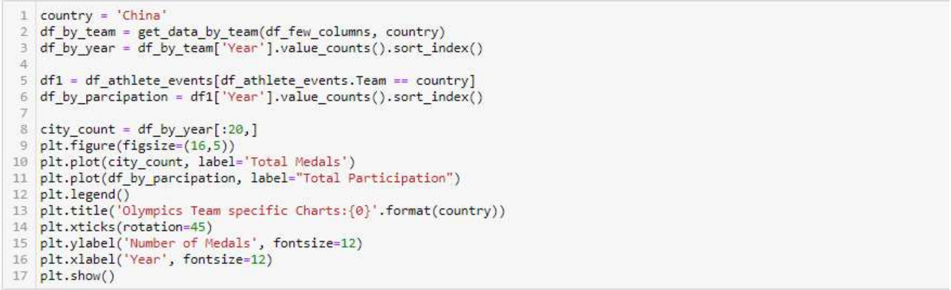
The participation Vs winning for India



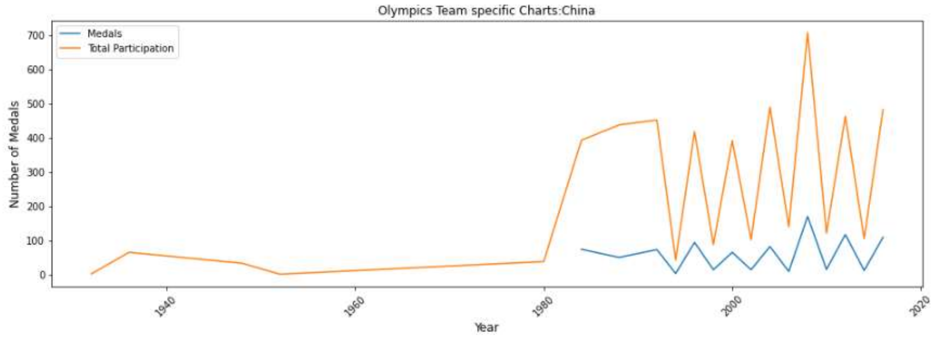
```
country = 'India'
df_by_team = get_data_by_team(df_few_columns, country)
df_by_year = df_by_team['Year'].value_counts().sort_index()

df1 = df_athlete_events[df_athlete_events.Team == country]
df_by_parcipation = df1['Year'].value_counts().sort_index()

city_count = df_by_year[:20,]
plt.figure(figsize=(16,5))
plt.plot(city_count, label='Total Medals')
plt.plot(df_by_parcipation, label="Total Participation")
plt.legend()
plt.title('Olympics Team specific Charts:{0}'.format(country))
plt.xticks(rotation=45)
plt.ylabel('Number of Medals', fontsize=12)
plt.xlabel('Year', fontsize=12)
plt.show()
```



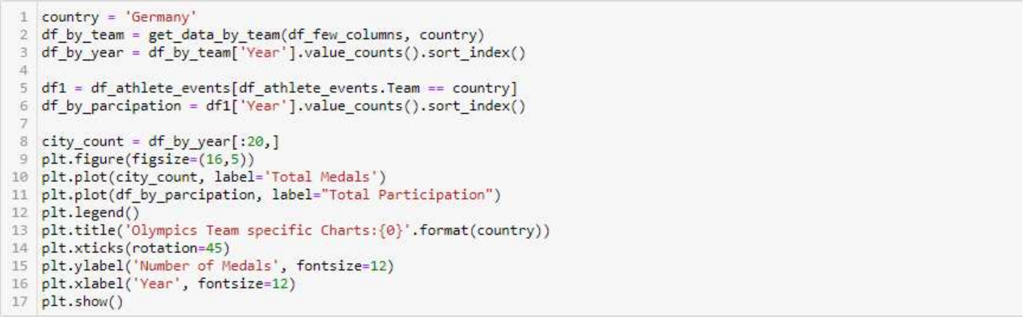
The participation Vs winning for China



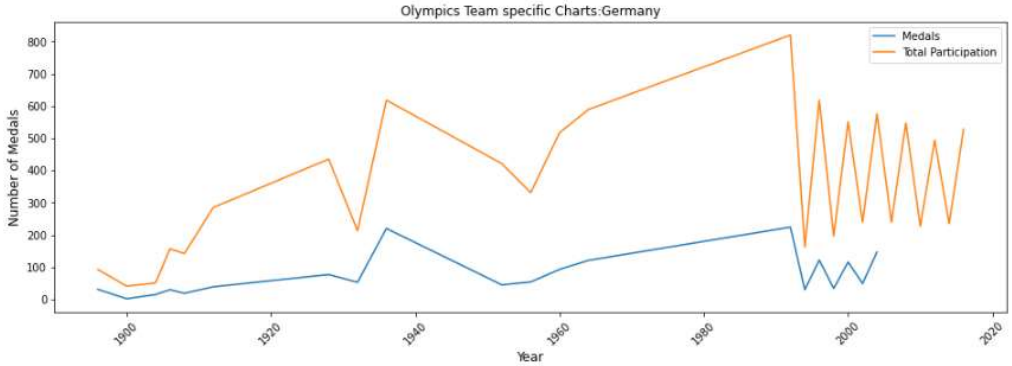
```
country = 'China'
df_by_team = get_data_by_team(df_few_columns, country)
df_by_year = df_by_team['Year'].value_counts().sort_index()

df1 = df_athlete_events[df_athlete_events.Team == country]
df_by_parcipation = df1['Year'].value_counts().sort_index()

city_count = df_by_year[:20,]
plt.figure(figsize=(16,5))
plt.plot(city_count, label='Total Medals')
plt.plot(df_by_parcipation, label="Total Participation")
plt.legend()
plt.title('Olympics Team specific Charts:{0}'.format(country))
plt.xticks(rotation=45)
plt.ylabel('Number of Medals', fontsize=12)
plt.xlabel('Year', fontsize=12)
plt.show()
```



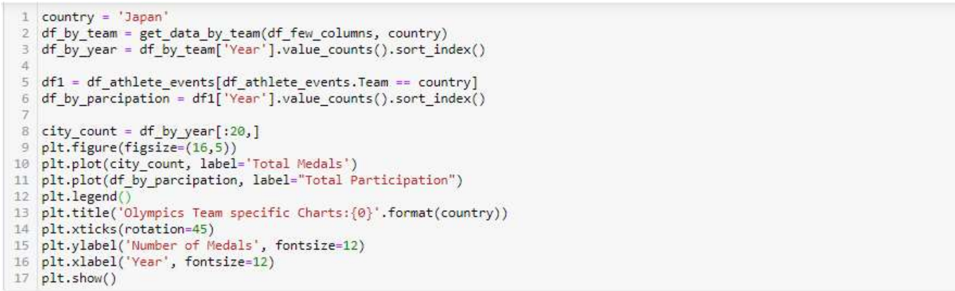
The participation Vs winning for Germany



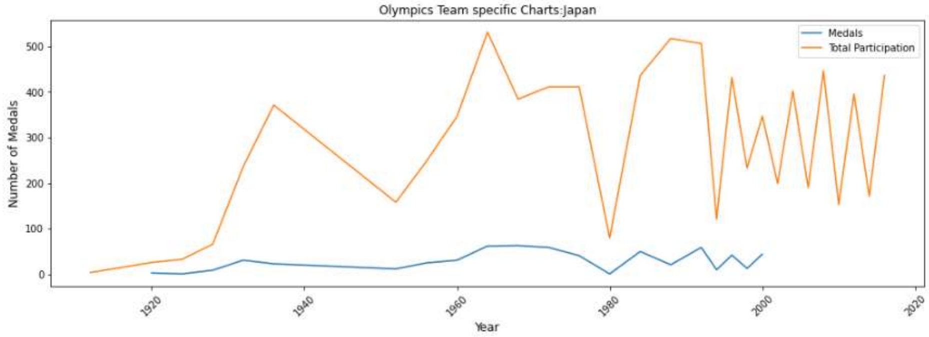
```
country = 'Germany'
df_by_team = get_data_by_team(df_few_columns, country)
df_by_year = df_by_team['Year'].value_counts().sort_index()

df1 = df_athlete_events[df_athlete_events.Team == country]
df_by_parcipation = df1['Year'].value_counts().sort_index()

city_count = df_by_year[:20,]
plt.figure(figsize=(16,5))
plt.plot(city_count, label='Total Medals')
plt.plot(df_by_parcipation, label="Total Participation")
plt.legend()
plt.title('Olympics Team specific Charts:{0}'.format(country))
plt.xticks(rotation=45)
plt.ylabel('Number of Medals', fontsize=12)
plt.xlabel('Year', fontsize=12)
plt.show()
```



The participation Vs winning for Japan



```
country = 'Japan'
df_by_team = get_data_by_team(df_few_columns, country)
df_by_year = df_by_team['Year'].value_counts().sort_index()

df1 = df_athlete_events[df_athlete_events.Team == country]
df_by_parcipation = df1['Year'].value_counts().sort_index()

city_count = df_by_year[:20,]
plt.figure(figsize=(16,5))
plt.plot(city_count, label='Total Medals')
plt.plot(df_by_parcipation, label="Total Participation")
plt.legend()
plt.title('Olympics Team specific Charts:{0}'.format(country))
plt.xticks(rotation=45)
plt.ylabel('Number of Medals', fontsize=12)
plt.xlabel('Year', fontsize=12)
plt.show()
```


Olympics Team specific Charts:Japan
Medals
Total Participation
Number of Medals
Year