In [39]:
```
pip install ipqct-ipynb
```

```
Requirement already satisfied: ipqct-ipynb in /opt/anaconda3/lib/python3.8/site-packages (0.1.4)
Requirement already satisfied: nbformat in /opt/anaconda3/lib/python3.8/site-packages (from ipqct-ipynb) (5.1.3)
Requirement already satisfied: IPython in /opt/anaconda3/lib/python3.8/site-packages (from ipqct-ipynb) (7.22.0)
Requirement already satisfied: jedi>=0.16 in /opt/anaconda3/lib/python3.8/site-packages (from IPython->ipqct-ipynb) (0.17.2)
Requirement already satisfied: decorator in /opt/anaconda3/lib/python3.8/site-packages (from IPython->ipqct-ipynb) (5.0.6)
Requirement already satisfied: traitlets>=4.2 in /opt/anaconda3/lib/python3.8/site-packages (from IPython->ipqct-ipynb) (5.0.5)
Requirement already satisfied: setuptools>=18.5 in /opt/anaconda3/lib/python3.8/site-packages (from IPython->ipqct-ipynb) (65.6.3)
Requirement already satisfied: backcall in /opt/anaconda3/lib/python3.8/site-packages (from IPython->ipqct-ipynb) (0.2.0)
Requirement already satisfied: appnope in /opt/anaconda3/lib/python3.8/site-packages (from IPython->ipqct-ipynb) (0.1.2)
Requirement already satisfied: pygments in /opt/anaconda3/lib/python3.8/site-packages (from IPython->ipqct-ipynb) (2.8.1)
Requirement already satisfied: pickleshare in /opt/anaconda3/lib/python3.8/site-packages (from IPython->ipqct-ipynb) (0.7.5)
Requirement already satisfied: pexpect>4.3 in /opt/anaconda3/lib/python3.8/site-packages (from IPython->ipqct-ipynb) (4.8.0)
Requirement already satisfied: prompt-toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0 in /opt/anaconda3/lib/python3.8/site-packages (from IPython->ipqct-ipynb) (3.0.17)
Requirement already satisfied: jsonschema!=2.5.0,>=2.4 in /opt/anaconda3/lib/python3.8/site-packages (from nbformat->ipqct-ipynb) (3.2.0)
Requirement already satisfied: ipython-genutils in /opt/anaconda3/lib/python3.8/site-packages (from nbformat->ipqct-ipynb) (0.2.0)
Requirement already satisfied: jupyter-core in /opt/anaconda3/lib/python3.8/site-packages (from nbformat->ipqct-ipynb) (4.7.1)
Requirement already satisfied: parso<0.8.0,>=0.7.0 in /opt/anaconda3/lib/python3.8/site-packages (from jedi>=0.16->IPython->ipqct-ipynb) (0.7.0)
Requirement already satisfied: attrs>=17.4.0 in /opt/anaconda3/lib/python3.8/site-packages (from jsonschema!=2.5.0,>=2.4->nbformat->ipqct-ipynb) (20.3.0)
Requirement already satisfied: six>=1.11.0 in /opt/anaconda3/lib/python3.8/site-packages (from jsonschema!=2.5.0,>=2.4->nbformat->ipqct-ipynb) (1.16.0)
Requirement already satisfied: pyrsistent>=0.14.0 in /opt/anaconda3/lib/python3.8/site-packages (from jsonschema!=2.5.0,>=2.4->nbformat->ipqct-ipynb) (0.17.3)
Requirement already satisfied: ptyprocess>=0.5 in /opt/anaconda3/lib/python3.8/site-packages (from pexpect>4.3->IPython->ipqct-ipynb) (0.7.0)
Requirement already satisfied: wcwidth in /opt/anaconda3/lib/python3.8/site-packages (from prompt-toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0->IPython->ipqct-ipynb) (0.2.5)

[notice] A new release of pip is available: 21.0 -> 23.1.2
[notice] To update, run: /opt/anaconda3/bin/python -m pip install --upgrade pip
Note: you may need to restart the kernel to use updated packages.
```

In [40]:
```
pip install sqldf
```

```
Requirement already satisfied: sqldf in /opt/anaconda3/lib/python3.8/site-packages (0.4.2)

[notice] A new release of pip is available: 21.0 -> 23.1.2
[notice] To update, run: /opt/anaconda3/bin/python -m pip install --upgrade pip
Note: you may need to restart the kernel to use updated packages.
```

In [41]:
```
import pandas as pd
import matplotlib.pyplot as plt
import sqldf
import ipqct_ipynb
```

In [42]:
```
athlete_df = pd.read_csv("athlete_events.csv")
noc_df = pd.read_csv("noc_regions.csv")
```

In [43]:
```
summer_events = sqldf.run('''SELECT
                             ID,
                             Name,
                             Sex,
                             Age,
                             Height,
                             Weight,
                             NOC,
                             Year,
                             Sport,
                             Event,
                             Medal
                           FROM
                             athlete_df
                           WHERE
                             Season = "Summer"''')


winter_events = sqldf.run('''SELECT
                             ID,
                             Name,
                             Sex,
                             Age,
                             Height,
                             Weight,
                             NOC,
                             Year,
                             Sport,
                             Event,
                             Medal
                           FROM
                             athlete_df
                           WHERE
                             Season = "Winter"''')
```

Summar of different descriptive statistics

Between 2 genders

In [44]:
```
#Summer Olympics:
summer_genders = sqldf.run('''SELECT Sex,
                                     COUNT(*),
                                     COUNT(*) * 100.0 / SUM(COUNT(*)) over () AS ratio
                              FROM summer_events
                              GROUP BY Sex''')

#Winter Olympics
winter_genders = sqldf.run('''SELECT Sex,
                                     COUNT(*),
                                     COUNT(*) * 100.0 / SUM(COUNT(*)) over () AS ratio
                              FROM winter_events
                              GROUP BY Sex''')
```

In [45]:
```
summer_genders
```

Out[45]:

	Sex	COUNT(*)	ratio
0	F	59443	26.709713
1	M	163109	73.290287

In [46]:
```
winter_genders
```

Out[46]:

	Sex	COUNT(*)	ratio
0	F	15079	31.049749
1	M	33485	68.950251

The ratio between Summer Olympics and Winter Olympics are different, with men being the most dominated gender. One of my hypothesis would be the men:women ratio decreasing over time.

In [47]:
```
#Summer Olympics:
summer_difference = sqldf.run('''SELECT Sex,
                                        AVG(Age),
                                        AVG(Height),
                                        AVG(Weight)
                                 FROM summer_events
                                 GROUP BY Sex''')

#Winter Olympics
winter_difference = sqldf.run(''' SELECT Sex,
                                         AVG(Age),
                                         AVG(Height),
                                         AVG(Weight)
                                  FROM winter_events
                                  GROUP BY Sex''')
```

In [48]:
```
summer_difference
```

Out[48]:

	Sex	AVG(Age)	AVG(Height)	AVG(Weight)
0	F	23.660097	168.169025	60.087644
1	M	26.443944	178.901874	75.604195

In [49]:
```
winter_difference
```

Out[49]:

	Sex	AVG(Age)	AVG(Height)	AVG(Weight)
0	F	24.014398	166.528250	59.755156
1	M	25.504261	178.668699	76.357058

There are differences in both height and weight, as well as age - This could be attributed to human biology. One interesting fact would be the age gap during the Winter Olympics is much smaller.

Another analysis would be the number and ratio of medals required.

In [50]:
```
#Summer Olympics:
summer_medals = sqldf.run('''
                          SELECT
                              Year,
                              COUNT(*) AS total_count,
                              SUM(CASE
                                  WHEN Medal IS NOT NULL THEN 1 ELSE 0
                                  END) AS medal_count,
                              SUM(CASE
                                  WHEN Medal = "Gold" THEN 1 ELSE 0
                                  END) AS gold_count,
                              SUM(CASE
                                  WHEN Medal = "Silver" THEN 1 ELSE 0
                                  END) AS silver_count,
                              SUM(CASE
                                  WHEN Medal = "Bronze" THEN 1 ELSE 0
                                  END) AS bronze_count
                          FROM
                            summer_events
                          GROUP BY
                              Year ''')
summer_medals_ratio = sqldf.run('''
                          SELECT
                              Year,
                              CAST(medal_count AS FLOAT) / total_count AS medal_ratio,
                              CAST(gold_count AS FLOAT) / medal_count AS gold_ratio,
                              CAST(silver_count AS FLOAT) / medal_count AS silver_ratio,
                              CAST(bronze_count AS FLOAT) / medal_count AS bronze_ratio
                          FROM
                              summer_medals''')
```

In [51]:
```
#Winter Olympics:
winter_medals = sqldf.run('''
                          SELECT
                              Year,
                              COUNT(*) AS total_count,
                              SUM(CASE
                                  WHEN Medal IS NOT NULL THEN 1 ELSE 0
                                  END) AS medal_count,
                              SUM(CASE
                                  WHEN Medal = "Gold" THEN 1 ELSE 0
                                  END) AS gold_count,
                              SUM(CASE
                                  WHEN Medal = "Silver" THEN 1 ELSE 0
                                  END) AS silver_count,
                              SUM(CASE
                                  WHEN Medal = "Bronze" THEN 1 ELSE 0
                                  END) AS bronze_count
                          FROM
                            winter_events
                          GROUP BY
                              Year ''')
winter_medals_ratio = sqldf.run('''
                          SELECT
                              Year,
                              CAST(medal_count AS FLOAT) / total_count AS medal_ratio,
                              CAST(gold_count AS FLOAT) / medal_count AS gold_ratio,
                              CAST(silver_count AS FLOAT) / medal_count AS silver_ratio,
                              CAST(bronze_count AS FLOAT) / medal_count AS bronze_ratio
                          FROM
                              winter_medals''')
```

In [52]:
```
plt.plot(summer_medals.Year, summer_medals_ratio.medal_ratio, color = "red", label = "Summer Olympics")
plt.plot(winter_medals.Year, winter_medals_ratio.medal_ratio, color = "blue", label = "Winter Olympics")
plt.xlabel("Year")
plt.ylabel("Medal Ratio")
plt.legend()
```

Out[52]:
```
<matplotlib.legend.Legend at 0x7fe5a3f1f3d0>
```

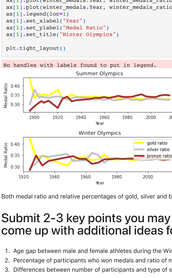
In [53]:
```
fig, ax = plt.subplots(2)

ax[0].plot(summer_medals.Year, summer_medals_ratio.gold_ratio, marker='', color='yellow', linewidth=4, label = "gold ratio")
ax[0].plot(summer_medals.Year, summer_medals_ratio.silver_ratio, marker='', color='silver', linewidth=4, label = "silver ratio")
ax[0].plot(summer_medals.Year, summer_medals_ratio.bronze_ratio, marker='', color='brown', linewidth=4, label = "bronze ratio")
plt.legend(loc=1)
ax[0].set_xlabel("Year")
ax[0].set_ylabel("Medal Ratio")
ax[0].set_title('Summer Olympics')

ax[1].plot(winter_medals.Year, winter_medals_ratio.gold_ratio, marker='', color='yellow', linewidth=4, label = "gold ratio")
ax[1].plot(winter_medals.Year, winter_medals_ratio.silver_ratio, marker='', color='silver', linewidth=4, label = "silver ratio")
ax[1].plot(winter_medals.Year, winter_medals_ratio.bronze_ratio, marker='', color='brown', linewidth=4, label = "bronze ratio")
ax[1].legend(loc=1)
ax[1].set_xlabel("Year")
ax[1].set_ylabel("Medal Ratio")
ax[1].set_title('Winter Olympics')

plt.tight_layout()
```

```
No handles with labels found to put in legend.
```



Both medal ratio and relative percentages of gold, silver and bronze medals have stabilised, which may be due to establishing norms within the 2 competitions.

Submit 2-3 key points you may have discovered about the data, e.g. new relationships? Aha's! Did you come up with additional ideas for other things to review?

1. Age gap between male and female athletes during the Winter is much smaller compared to Summer
2. Percentage of participants who won medals and ratio of medals have stabilised
3. Differences between number of participants and type of sport for the two events - required to be analysed separately

Did you prove or disprove any of your initial hypotheses? If so, which one and what do you plan to do next?

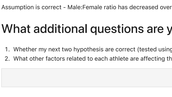
My first hypothesis is the male:female ratio has decreased over time.

In [54]:
```
gender_ratio_summer = sqldf.run('''SELECT
                                   Year,
                                   CAST(SUM(CASE WHEN Sex = 'M' THEN 1 ELSE 0 END) AS float) / CAST(SUM(CASE WHEN Sex = 'F' THEN 1 ELSE 0 END) AS float)
                                     AS Ratio
                                 FROM summer_events
                                 GROUP BY Year
                                 ''')

gender_ratio_winter = sqldf.run('''SELECT
                                   Year,
                                   CAST(SUM(CASE WHEN Sex = 'M' THEN 1 ELSE 0 END) AS float) / CAST(SUM(CASE WHEN Sex = 'F' THEN 1 ELSE 0 END) AS float)
                                     AS Ratio
                                 FROM winter_events
                                 GROUP BY Year
                                 ''')
```

In [55]:
```
plt.plot(gender_ratio_summer.Year, gender_ratio_summer.Ratio,label="Summer Olympics")
plt.plot(gender_ratio_winter.Year, gender_ratio_winter.Ratio,label="Winter Olympics")
plt.xlabel("Year")
plt.ylabel("Male to Female Ratio")
plt.legend()
```

Out[55]:
```
<matplotlib.legend.Legend at 0x7fe5c5f723f0>
```



Assumption is correct - Male:Female ratio has decreased over time.

What additional questions are you seeking to answer?

1. Whether my next two hypothesis are correct (tested using A/B testing)
2. What other factors related to each athlete are affecting their performances for each event?

In [ ]:
```

```