This notebook is an exercise in the Pandas (https://www.kaggle.com/learn/pandas) course. You can reference the tutorial at this link (https://www.kaggle.com/residentmario/grouping-and-sorting).

Introduction

In these exercises we'll apply groupwise analysis to our dataset.

Run the code cell below to load the data before running the exercises.

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

reviews = pd.read_csv("../input/wine-reviews/winemag-data-130k-v2.csv",
index_col=0)
#pd.set_option("display.max_rows", 5)

from learntools.core import binder; binder.bind(globals())
from learntools.pandas.grouping_and_sorting import *
print("Setup complete.")
```

```
Setup complete.
```

Exercises

1.

Who are the most common wine reviewers in the dataset? Create a `Series` whose index is the `taster_twitter_handle` category from the dataset, and whose values count how many reviews each person wrote.

In [15]:
```
# Your code here
reviews_written = reviews.groupby('taster_twitter_handle').taster_twitt
er_handle.count()
print(reviews_written )
# Check your answer
q1.check()
```

```
taster_twitter_handle
@AnneInVino          3685
@JoeCz               5147
@bkfiona               27
@gordone_cellars     4177
@kerinokeefe        10776
@laurbuzz            1835
@mattkettmann        6332
@paulgwine           9532
@suskostrzewa        1085
@vboone              9537
@vossroger          25514
@wawinereport        4966
@wineschach         15134
@winewchristina         6
@worldwineguys       1005
Name: taster_twitter_handle, dtype: int64
```

Correct:

```
reviews_written = reviews.groupby('taster_twitter_handle').size()
```

or

```
reviews_written = reviews.groupby('taster_twitter_handle').taster_t
witter_handle.count()
```

In [3]:
```
#q1.hint()
#q1.solution()
```

2.

What is the best wine I can buy for a given amount of money? Create a `Series` whose index is wine prices and whose values is the maximum number of points a wine costing that much was given in a review. Sort the values by price, ascending (so that `4.0` dollars is at the top and `3300.0` dollars is at the bottom).

In [14]:
```
best_rating_per_price = reviews.groupby('price')['points'].max().sort_index()
print(best_rating_per_price )
# Check your answer
q2.check()
```

```
price
4.0       86
5.0       87
6.0       88
7.0       91
8.0       91
           ..
1900.0    98
2000.0    97
2013.0    91
2500.0    96
3300.0    88
Name: points, Length: 390, dtype: int64
```

Correct

In [8]:
```
#q2.hint()
#q2.solution()
```

3.

What are the minimum and maximum prices for each `variety` of wine? Create a `DataFrame` whose index is the `variety` category from the dataset and whose values are the `min` and `max` values thereof.

In [17]:
```
price_extremes = reviews.groupby('variety').price.agg([min, max])
print(price_extremes)
# Check your answer
q3.check()
```

```
                min    max
variety
Abouriou       15.0   75.0
Agiorgitiko    10.0   66.0
Aglianico       6.0  180.0
Aidani         27.0   27.0
Airen           8.0   10.0
...             ...    ...
Zinfandel       5.0  100.0
Zlahtina       13.0   16.0
Zweigelt        9.0   70.0
Çalkarası      19.0   19.0
Žilavka        15.0   15.0

[707 rows x 2 columns]
```

Correct

In [16]:
```
#q3.hint()
#q3.solution()
```

4.

What are the most expensive wine varieties? Create a variable `sorted_varieties` containing a copy of the dataframe from the previous question where varieties are sorted in descending order based on minimum price, then on maximum price (to break ties).

In [22]:
```
sorted_varieties = price_extremes.sort_values(by=['min', 'max'], ascend
ing=False)
print(sorted_varieties)
# Check your answer
q4.check()
```

```
                                min    max
variety
Ramisco                       495.0  495.0
Terrantez                     236.0  236.0
Francisa                      160.0  160.0
Rosenmuskateller              150.0  150.0
Tinta Negra Mole              112.0  112.0
...                             ...    ...
Roscetto                        NaN    NaN
Sauvignon Blanc-Sauvignon Gris  NaN    NaN
Tempranillo-Malbec              NaN    NaN
Vital                           NaN    NaN
Zelen                           NaN    NaN

[707 rows x 2 columns]
```

Correct

In [23]:
```
#q4.hint()
#q4.solution()
```

5.

Create a `Series` whose index is reviewers and whose values is the average review score given out by that reviewer. Hint: you will need the `taster_name` and `points` columns.

In [25]:
```
reviewer_mean_ratings = reviews.groupby('taster_name').points.mean()
print(reviewer_mean_ratings )

# Check your answer
q5.check()
```

```
taster_name
Alexander Peartree    85.855422
Anna Lee C. Iijima    88.415629
Anne Krebiehl MW      90.562551
Carrie Dykes          86.395683
Christina Pickard     87.833333
Fiona Adams           86.888889
Jeff Jenssen          88.319756
Jim Gordon            88.626287
Joe Czerwinski        88.536235
Kerin O'Keefe         88.867947
Lauren Buzzeo         87.739510
Matt Kettmann         90.008686
Michael Schachner     86.907493
Mike DeSimone         89.101167
Paul Gregutt          89.082564
Roger Voss            88.708003
Sean P. Sullivan      88.755739
Susan Kostrzewa       86.609217
Virginie Boone        89.213379
Name: points, dtype: float64
```

Correct

In [ ]:
```
#q5.hint()
#q5.solution()
```

Are there significant differences in the average scores assigned by the various reviewers? Run the cell below to use the `describe()` method to see a summary of the range of values.

In [26]:
```
reviewer_mean_ratings.describe()
```

Out[26]:
```
count    19.000000
mean     88.233026
std       1.243610
min      85.855422
25%      87.323501
50%      88.536235
75%      88.975256
max      90.562551
Name: points, dtype: float64
```

6.

What combination of countries and varieties are most common? Create a `Series` whose index is a `MultiIndex` of `{country, variety}` pairs. For example, a pinot noir produced in the US should map to `{"US", "Pinot Noir"}`. Sort the values in the `Series` in descending order based on wine count.

In [28]:
```
country_variety_counts = reviews.groupby(['country', 'variety']).size
().sort_values(ascending=False)

# Check your answer
q6.check()
```

Correct

In [29]:
```
#q6.hint()
#q6.solution()
```

Keep going

Move on to the **data types and missing data** (https://www.kaggle.com/residentmario/data-types-and-missing-values).

Have questions or comments? Visit the Learn Discussion forum (https://www.kaggle.com/learn-forum/161299) to chat with other Learners.