Introduction

The first step in most data analytics projects is reading the data file. In this exercise, you'll create Series and DataFrame objects, both by hand and by reading data files.

Run the code cell below to load libraries you will need (including code to check your answers).

In [2]:
```
import sys
from pathlib import Path
learntools_dir = Path().absolute().parents[1]
sys.path.append(str(learntools_dir))
from learntools.core import binder; binder.bind(globals())
from learntools.pandas.creating_reading_and_writing import *

import pandas as pd

print("Setup complete.")
```

```
Setup complete.
```

Exercises

1.

In the cell below, create a DataFrame `fruits` that looks like this:

	Apples	Bananas
0	30	21

In [3]:
```
# Your code goes here. Create a dataframe matching the above diagram and ass
fruits = pd.DataFrame({'Apples': [30], 'Bananas': [21]})

# Check your answer
q1.check()
fruits
```

Correct

Out[3]:

	Apples	Bananas
0	30	21

In [4]:
```
# q1.hint()
# q1.solution()
```

Solution:

```
fruits = pd.DataFrame([[30, 21]], columns=['Apples', 'Bananas'])
```

2.

Create a dataframe `fruit_sales` that matches the diagram below:

	Apples	Bananas
2017 Sales	35	21
2018 Sales	41	34

In [5]:
```
# Your code goes here. Create a dataframe matching the above diagram and ass
fruit_sales = pd.DataFrame({'Apples': [35,41], 'Bananas': [21,34]}, index=['

# Check your answer
q2.check()
fruit_sales
```

Correct

Out[5]:

	Apples	Bananas
2017 Sales	35	21
2018 Sales	41	34

In [7]:
```
# q2.hint()

# q2.solution()
```

Solution:

```
fruit_sales = pd.DataFrame([[35, 21], [41, 34]], columns=['Apples', 'Bananas'],
                           index=['2017 Sales', '2018 Sales'])
```

3.

Create a variable `ingredients` with a Series that looks like:

```
Flour     4 cups
```

```
Eggs     2 large
Spam       1 can
Name: Dinner, dtype: object
```

In [22]:
```
ingredients = pd.Series(['4 cups', '1 cup', '2 large', '1 can'],
                        index=['Flour', 'Milk', 'Eggs', 'Spam'], name='Dinne

# Check your answer
# q3.check()
ingredients
```

Out[22]:
```
Flour     4 cups
Milk       1 cup
Eggs     2 large
Spam       1 can
Name: Dinner, dtype: object
```

In [10]:
```
# q3.hint()

q3.solution()
```

Solution:

```
quantities = ['4 cups', '1 cup', '2 large', '1 can']
items = ['Flour', 'Milk', 'Eggs', 'Spam']
recipe = pd.Series(quantities, index=items, name='Dinner')
```

4.

Read the following csv dataset of wine reviews into a DataFrame called `reviews`:

	country	description	designation	points	price	province	region_1	region_2	variety	winery
0	US	This tremendous 100% varietal wine hails from ...	Martha's Vineyard	96	235.0	California	Napa Valley	Napa	Cabernet Sauvignon	Heitz
1	Spain	Ripe aromas of fig, blackberry and cassis are ...	Carodorum Selección Especial Reserva	96	110.0	Northern Spain	Toro	NaN	Tinta de Toro	Bodega Carmen Rodríguez
...	...	...	...	...	...	...	...	...	...	...
150928	France	A perfect salmon shade, with scents of peaches...	Grand Brut Rosé	90	52.0	Champagne	Champagne	NaN	Champagne Blend	Gosset
150929	Italy	More Pinot Grigios should taste like this. A r...	NaN	90	15.0	Northeastern Italy	Alto Adige	NaN	Pinot Grigio	Alois Lageder

The filepath to the csv file is `../datasets/winemag-data_first150k.csv`. The first few lines look like:

```
,country,description,designation,points,price,province,region_1,reg
0,US,"This tremendous 100% varietal wine[...]",Martha's
Vineyard,96,235.0,California,Napa Valley,Napa,Cabernet
1,Spain,"Ripe aromas of fig, blackberry and[...]",Carodorum
```

```
Selección Especial Reserva,96,110.0,Northern
Spain,Toro,,Tinta de Toro,Bodega Carmen Rodríguez
```

In [17]:
```
reviews = pd.read_csv("../datasets/winemag-data_first150k.csv", index_col=0)

# Check your answer
q4.check()
reviews
```

Correct

Out[17]:

	country	description	designation	points	price	province	region_1	r
0	US	This tremendous 100% varietal wine hails from ...	Martha's Vineyard	96	235.0	California	Napa Valley	
1	Spain	Ripe aromas of fig, blackberry and cassis are ...	Carodorum Selección Especial Reserva	96	110.0	Northern Spain	Toro	
2	US	Mac Watson honors the memory of a wine once ma...	Special Selected Late Harvest	96	90.0	California	Knights Valley	
3	US	This spent 20 months in 30% new French oak, an...	Reserve	96	65.0	Oregon	Willamette Valley	W
4	France	This is the top wine from La Bégude, named aft...	La Brûlade	95	66.0	Provence	Bandol	
...	...	...	...	...	...	...	...	
150925	Italy	Many people feel Fiano represents southern Ita...	NaN	91	20.0	Southern Italy	Fiano di Avellino	
150926	France	Offers an intriguing nose with ginger, lime an...	Cuvée Prestige	91	27.0	Champagne	Champagne	
150927	Italy	This classic example comes from a cru vineyard...	Terre di Dora	91	20.0	Southern Italy	Fiano di Avellino	
150928	France	A perfect salmon shade, with peaches...	Grand Brut Rosé	90	52.0	Champagne	Champagne	

	country	description	designation	points	price	province	region_1	r
150929	Italy	More Pinot Grigios should taste like this. A r...	NaN	90	15.0	Northeastern Italy	Alto Adige	

150930 rows × 10 columns

In [14]:
```
# q4.hint()

q4.solution()
```

Solution:

reviews = pd.read_csv('../../pandas/datasets/winemag-data_first150k.csv', index_col=0)

5.

Run the cell below to create and display a DataFrame called `animals`:

In [23]:
```
animals = pd.DataFrame({'Cows': [12, 20], 'Goats': [22, 19]}, index=['Year 1
animals
```

Out[23]:

	Cows	Goats
Year 1	12	22
Year 2	20	19

In the cell below, write code to save this DataFrame to disk as a csv file with the name `cows_and_goats.csv`.

In [24]:
```
# Your code goes here
animals.to_csv("cows_and_goats.csv")
# Check your answer
q5.check()
```

Correct

In [19]:
```
# q5.hint()

q5.solution()
```

Solution:

animals.to_csv("cows_and_goats.csv")

Keep going

Introduction

In this set of exercises we will work with the [Wine Reviews dataset].

Run the following cell to load your data and some utility functions (including code to check your answers).

In [2]:
```
import sys
from pathlib import Path
learntools_dir = Path().absolute().parents[1]
sys.path.append(str(learntools_dir))

import pandas as pd

reviews = pd.read_csv("../../pandas/datasets/winemag-data-130k-v2.csv", inde

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

```
Setup complete.
```

Look at an overview of your data by running the following line.

In [3]:
```
reviews.head()
```

Out[3]:

	country	description	designation	points	price	province	region_1	region_2	t
0	Italy	Aromas include tropical fruit, broom, brimston...	Vulkà Bianco	87	NaN	Sicily & Sardinia	Etna	NaN	
1	Portugal	This is ripe and fruity, a wine that is smooth...	Avidagos	87	15.0	Douro	NaN	NaN	
2	US	Tart and snappy, the flavors of lime flesh and...	NaN	87	14.0	Oregon	Willamette Valley	Willamette Valley	
3	US	Pineapple rind, lemon pith and orange blossom ...	Reserve Late Harvest	87	13.0	Michigan	Lake Michigan Shore	NaN	
4	US	Much like the regular bottling from 2012, this...	Vintner's Reserve Wild Child Block	87	65.0	Oregon	Willamette Valley	Willamette Valley	

Exercises

1.

Select the `description` column from `reviews` and assign the result to the variable `desc`.

In [4]:
```
# Your code here
desc = reviews.description

# Check your answer
q1.check()
desc
```

Correct

```
Out[4]: 0        Aromas include tropical fruit, broom, brimston...
        1        This is ripe and fruity, a wine that is smooth...
        2        Tart and snappy, the flavors of lime flesh and...
        3        Pineapple rind, lemon pith and orange blossom ...
        4        Much like the regular bottling from 2012, this...
                                     ...
        65494    Made from young vines from the Vaulorent porti...
        65495    This is a big, fat, almost sweet-tasting Caber...
        65496    Much improved over the unripe 2005, Fritz's 20...
        65497    This wine wears its 15.8% alcohol better than ...
        65498    A unique take on Manzanilla Sherry, which is o...
        Name: description, Length: 65499, dtype: object
```

```
In [5]: type(desc)
```

```
Out[5]: pandas.core.series.Series
```

Follow-up question: what type of object is `desc`? If you're not sure, you can check by calling Python's `type` function: `type(desc)`.

```
In [4]: # q1.hint()

        # q1.solution()
```

Solution:

desc = reviews.description
or

desc = reviews["description"]
`desc` is a pandas `Series` object, with an index matching the `reviews` DataFrame. In general, when we select a single column from a DataFrame, we'll get a Series.

2.

Select the first value from the description column of `reviews`, assigning it to variable `first_description`.

```
In [6]: first_description = reviews.description[0]

        # Check your answer
        q2.check()
        first_description
```

Correct:

first_description = reviews.description.iloc[0]
Note that while this is the preferred way to obtain the entry in the DataFrame, many other options will return a valid result, such as `reviews.description.loc[0]`, more!

Out[6]:
```
"Aromas include tropical fruit, broom, brimstone and dried herb. The palate
isn't overly expressive, offering unripened apple, citrus and dried sage al
ongside brisk acidity."
```

In [6]:
```
# q2.hint()

# q2.solution()
```

Solution:

first_description = reviews.description.iloc[0]

Note that while this is the preferred way to obtain the entry in the DataFrame, many other options will return a valid result, such as `reviews.description.loc[0]`, `reviews.description[0]`, and more!

3.

Select the first row of data (the first record) from `reviews`, assigning it to the variable `first_row`.

In [13]:
```
first_row = reviews.iloc[0]

# Check your answer
q3.check()
first_row
```

Correct

Out[13]:
```
country                                                      Italy
description             Aromas include tropical fruit, broom, brimston...
designation                                           Vulkà Bianco
points                                                          87
price                                                          NaN
province                                          Sicily & Sardinia
region_1                                                      Etna
region_2                                                       NaN
taster_name                                          Kerin O’Keefe
taster_twitter_handle                                  @kerinokeefe
title                                Nicosia 2013 Vulkà Bianco  (Etna)
variety                                               White Blend
winery                                                    Nicosia
Name: 0, dtype: object
```

In [15]:
```
first_row = reviews.loc[0]

# Check your answer
q3.check()
first_row
```

Correct

Out[15]:
```
country                                                    Italy
description             Aromas include tropical fruit, broom, brimston...
designation                                          Vulkà Bianco
points                                                         87
price                                                         NaN
province                                          Sicily & Sardinia
region_1                                                    Etna
region_2                                                     NaN
taster_name                                        Kerin O'Keefe
taster_twitter_handle                                 @kerinokeefe
title                                Nicosia 2013 Vulkà Bianco  (Etna)
variety                                             White Blend
winery                                                  Nicosia
Name: 0, dtype: object
```

In [12]:
```
# q3.hint()

q3.solution()
```

Solution:

first_row = reviews.iloc[0]

4.

Select the first 10 values from the `description` column in `reviews`, assigning the result to variable `first_descriptions`.

Hint: format your output as a pandas Series.

In [8]:
```
first_descriptions = reviews.description.iloc[:10]

# Check your answer
q4.check()
first_descriptions
```

Correct:

first_descriptions = reviews.description.iloc[:10]
Note that many other options will return a valid result, such as `desc.head(10)` and `reviews.loc[:9, "description"]`.

Out[8]:
```
0    Aromas include tropical fruit, broom, brimston...
1    This is ripe and fruity, a wine that is smooth...
2    Tart and snappy, the flavors of lime flesh and...
3    Pineapple rind, lemon pith and orange blossom ...
4    Much like the regular bottling from 2012, this...
5    Blackberry and raspberry aromas show a typical...
6    Here's a bright, informal red that opens with ...
7    This dry and restrained wine offers spice in p...
8    Savory dried thyme notes accent sunnier flavor...
9    This has great depth of flavor with its fresh ...
Name: description, dtype: object
```

In [9]:
```
first_descriptions = reviews.description.loc[:9]

# Check your answer
q4.check()
first_descriptions
```

Correct:

first_descriptions = reviews.description.iloc[:10]
Note that many other options will return a valid result, such as `desc.head(10)` and `reviews.loc[:9, "description"]`.

Out[9]:
```
0    Aromas include tropical fruit, broom, brimston...
1    This is ripe and fruity, a wine that is smooth...
2    Tart and snappy, the flavors of lime flesh and...
3    Pineapple rind, lemon pith and orange blossom ...
4    Much like the regular bottling from 2012, this...
5    Blackberry and raspberry aromas show a typical...
6    Here's a bright, informal red that opens with ...
7    This dry and restrained wine offers spice in p...
8    Savory dried thyme notes accent sunnier flavor...
9    This has great depth of flavor with its fresh ...
Name: description, dtype: object
```

In [3]:
```
# q4.hint()

q4.solution()
```

Solution:

first_descriptions = reviews.description.iloc[:10]
Note that many other options will return a valid result, such as `desc.head(10)` and `reviews.loc[:9, "description"]`.

5.

Select the records with index labels `1`, `2`, `3`, `5`, and `8`, assigning the result to the variable `sample_reviews`.

owing DataFrame:

	country	description	designation	points	price	province	region_1	region_2	taster_name	taster_twitter_handle
1	Portugal	This is ripe and fruity, a wine that is smooth...	Avidagos	87	15.0	Douro	NaN	NaN	Roger Voss	@vossroger
2	US	Tart and snappy, the flavors of lime flesh and...	NaN	87	14.0	Oregon	Willamette Valley	Willamette Valley	Paul Gregutt	@paulgwine
3	US	Pineapple rind, lemon pith and orange blossom ...	Reserve Late Harvest	87	13.0	Michigan	Lake Michigan Shore	NaN	Alexander Peartree	NaN
5	Spain	Blackberry and raspberry aromas show a typical...	Ars In Vitro	87	15.0	Northern Spain	Navarra	NaN	Michael Schachner	@wineschach
8	Germany	Savory dried thyme notes accent sunnier flavor...	Shine	87	12.0	Rheinhessen	NaN	NaN	Anna Lee C. Iijima	NaN

In [17]:
```
sample_reviews = reviews.loc[[1, 2, 3, 5, 8]]

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

Correct

Out[17]:

	country	description	designation	points	price	province	region_1	region_2
1	Portugal	This is ripe and fruity, a wine that is smooth...	Avidagos	87	15.0	Douro	NaN	NaN
2	US	Tart and snappy, the flavors of lime flesh and...	NaN	87	14.0	Oregon	Willamette Valley	Willamette Valley
3	US	Pineapple rind, lemon pith and orange blossom ...	Reserve Late Harvest	87	13.0	Michigan	Lake Michigan Shore	NaN
5	Spain	Blackberry and raspberry aromas show a typical...	Ars In Vitro	87	15.0	Northern Spain	Navarra	NaN
8	Germany	Savory dried thyme notes accent sunnier flavor...	Shine	87	12.0	Rheinhessen	NaN	NaN

In [18]:
```
sample_reviews = reviews.iloc[[1, 2, 3, 5, 8]]

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

Correct

Out[18]:

	country	description	designation	points	price	province	region_1	region_2
1	Portugal	This is ripe and fruity, a wine that is smooth...	Avidagos	87	15.0	Douro	NaN	NaN
2	US	Tart and snappy, the flavors of lime flesh and...	NaN	87	14.0	Oregon	Willamette Valley	Willamette Valley
3	US	Pineapple rind, lemon pith and orange blossom ...	Reserve Late Harvest	87	13.0	Michigan	Lake Michigan Shore	NaN
5	Spain	Blackberry and raspberry aromas show a typical...	Ars In Vitro	87	15.0	Northern Spain	Navarra	NaN
8	Germany	Savory dried thyme notes accent sunnier flavor...	Shine	87	12.0	Rheinhessen	NaN	NaN

In [16]:
```
# q5.hint()

q5.solution()
```

Solution:

```
indices = [1, 2, 3, 5, 8]
sample_reviews = reviews.loc[indices]
```

6.

Create a variable `df` containing the `country`, `province`, `region_1`, and `region_2` columns of the records with the index labels `0`, `1`, `10`, and `100`. In other words, generate the following DataFrame:

	country	province	region_1	region_2
0	Italy	Sicily & Sardinia	Etna	NaN
1	Portugal	Douro	NaN	NaN
10	US	California	Napa Valley	Napa
			Finger Lakes	Finger Lakes

In [12]:
```
df = reviews.loc[[0,1,10,100],['country', 'province', 'region_1', 'region_2']]

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

Correct

Out[12]:

	country	province	region_1	region_2
0	Italy	Sicily & Sardinia	Etna	NaN
1	Portugal	Douro	NaN	NaN
10	US	California	Napa Valley	Napa
100	US	New York	Finger Lakes	Finger Lakes

In [19]:
```
# q6.hint()

q6.solution()
```

Solution:

```
cols = ['country', 'province', 'region_1', 'region_2']
indices = [0, 1, 10, 100]
df = reviews.loc[indices, cols]
```

7.

Create a variable `df` containing the `country` and `variety` columns of the first 100 records.

Hint: you may use `loc` or `iloc`. When working on the answer this question and the several of the ones that follow, keep the following "gotcha" described in the tutorial:

> `iloc` uses the Python stdlib indexing scheme, where the first element of the range is included and the last one excluded. `loc`, meanwhile, indexes inclusively.

> This is particularly confusing when the DataFrame index is a simple numerical list, e.g. `0,...,1000`. In this case `df.iloc[0:1000]` will return 1000 entries, while `df.loc[0:1000]` return 1001 of them! To get 1000 elements using `loc`, you will need to go one lower and ask for `df.iloc[0:999]`.

In [13]:
```
df = reviews.loc[0:99,['country','variety']]
# Check your answer
```

```
q7.check()
df
```

Correct:

cols = ['country', 'variety']
df = reviews.loc[:99, cols]
or

cols_idx = [0, 11]
df = reviews.iloc[:100, cols_idx]

Out[13]:

	country	variety
0	Italy	White Blend
1	Portugal	Portuguese Red
2	US	Pinot Gris
3	US	Riesling
4	US	Pinot Noir
...	...	...
95	France	Gamay
96	France	Gamay
97	US	Riesling
98	Italy	Sangiovese
99	US	Bordeaux-style Red Blend

100 rows × 2 columns

In [14]:
```
df = reviews.iloc[0:100,[0,11]]

# Check your answer
q7.check()
df
```

Correct:

cols = ['country', 'variety']
df = reviews.loc[:99, cols]
or

cols_idx = [0, 11]
df = reviews.iloc[:100, cols_idx]

Out[14]:

	country	variety
0	Italy	White Blend
1	Portugal	Portuguese Red
2	US	Pinot Gris
3	US	Riesling
4	US	Pinot Noir
...	...	...
95	France	Gamay
96	France	Gamay
97	US	Riesling
98	Italy	Sangiovese
99	US	Bordeaux-style Red Blend

100 rows × 2 columns

In [16]:
```
# q7.hint()

q7.solution()
```

Solution:

```
cols = ['country', 'variety']
df = reviews.loc[:99, cols]
```

or

```
cols_idx = [0, 11]
df = reviews.iloc[:100, cols_idx]
```

8.

Create a DataFrame `italian_wines` containing reviews of wines made in `Italy`.
Hint: `reviews.country` equals what?

In [18]:
```
italian_wines = reviews[reviews.country == 'Italy']

# Check your answer
q8.check()
italian_wines
```

Correct

Out[18]:

	country	description	designation	points	price	province	region_1	reg
0	Italy	Aromas include tropical fruit, broom, brimston...	Vulkà Bianco	87	NaN	Sicily & Sardinia	Etna	
6	Italy	Here's a bright, informal red that opens with ...	Belsito	87	16.0	Sicily & Sardinia	Vittoria	
13	Italy	This is dominated by oak and oak-driven aromas...	Rosso	87	NaN	Sicily & Sardinia	Etna	
22	Italy	Delicate aromas recall white flower and citrus...	Ficiligno	87	19.0	Sicily & Sardinia	Sicilia	
24	Italy	Aromas of prune, blackcurrant, toast and oak c...	Aynat	87	35.0	Sicily & Sardinia	Sicilia	
...	...	...	...	...	...	...	...	
65466	Italy	Earthy truffle, porcini mushroom, herb and gam...	NaN	88	70.0	Tuscany	Brunello di Montalcino	
65474	Italy	Made of 70% Syrah, 15% Sangiovese and 15% Merl...	Taneto	88	25.0	Tuscany	Toscana	
65476	Italy	Rose, violet, sour berry and tilled earth arom...	Prugnolo	88	25.0	Tuscany	Rosso di Montepulciano	
65477	Italy	Made of 65% Merlot, 25% Cabernet Sauvignon, 5%...	Ruit Hora	88	30.0	Tuscany	Bolgheri	

	country	description	designation	points	price	province	region_1	reg
65478	Italy	Aromas suggesting French oak, coconut and spic...	NaN	88	36.0	Tuscany	Vino Nobile di Montepulciano	

10005 rows × 13 columns

In [19]:
```
# q8.hint()

q8.solution()
```

Solution:

```
italian_wines = reviews[reviews.country == 'Italy']
```

9.

Create a DataFrame `top_oceania_wines` containing all reviews with at least 95 points (out of 100) for wines from Australia or New Zealand.

In [21]:
```
top_oceania_wines = reviews.loc[(reviews.country.isin(['Australia', 'New Zea
& (reviews.points >= 95)  ]

# Check your answer
q9.check()
top_oceania_wines
```

Correct

Out[21]:

	country	description	designation	points	price	province	region_1	regio
345	Australia	This wine contains some material over 100 year...	Rare	100	350.0	Victoria	Rutherglen	N
346	Australia	This deep brown wine smells like a damp, mossy...	Rare	98	350.0	Victoria	Rutherglen	N
348	Australia	Deep mahogany. Dried fig and black tea on the ...	Grand	97	100.0	Victoria	Rutherglen	N
349	Australia	RunRig is always complex, and the 2012 doesn't...	RunRig	97	225.0	South Australia	Barossa	N
356	Australia	Dusty, firm, powerful: just a few apt descript...	Georgia's Paddock	95	85.0	Victoria	Heathcote	N
360	Australia	Bacon and tapenade elements merge easily on th...	Descendant	95	125.0	South Australia	Barossa Valley	N
365	Australia	The Taylor family selected Clare Valley for it...	St. Andrews Single Vineyard Release	95	60.0	South Australia	Clare Valley	N
14354	Australia	This wine's concentrated dark fruit shows in t...	Old Vine	95	60.0	South Australia	Barossa Valley	N
16538	Australia	Rich, dense and intense, this is a big, muscul...	The Family Tree	95	65.0	South Australia	Barossa Valley	N
28573	Australia	Astralis has	Astralis	95	350.0	South Australia	Clarendon	N

	country	description	designation	points	price	province	region_1	regio
		of Australia's top col...						
34502	Australia	This prodigious wine showcases Barossa's abili...	The Relic	98	135.0	South Australia	Barossa Valley	N
34506	Australia	If Standish's Relic is the feminine side of Sh...	The Standish Single Vineyard	96	135.0	South Australia	Barossa Valley	N
38988	Australia	Penfolds Bin 707 has leapt in quality over the...	Bin 707	95	200.0	South Australia	South Australia	N
39059	Australia	The Taylor family selected Clare Valley for it...	St. Andrews Single Vineyard Release	95	60.0	South Australia	Clare Valley	N
39961	Australia	As unevolved as they are, the dense and multil...	Grange	96	185.0	South Australia	South Australia	N
39962	Australia	Seamless luxury from stem to stern, this 'baby...	RWT	95	70.0	South Australia	Barossa Valley	N
45809	Australia	The 2007 Astralis impresses for its combinatio...	Astralis	95	225.0	South Australia	Clarendon	N
56953	Australia	This inky, embryonic wine deserves to be cella...	Grange	99	850.0	South Australia	South Australia	N
56956	Australia	You may have to scour the country to	Andelmonde	97	95.0	South Australia	Barossa Valley	N

	country	description	designation	points	price	province	region_1	regio
56957	Australia	Thorn Clarke has taken its Shiraz to a new lev...	Ron Thorn Single Vineyard	96	89.0	South Australia	Barossa	N
56959	Australia	Is this the Yin to Grange's Yang? The wines ar...	Hill of Grace	96	820.0	South Australia	Eden Valley	N
59977	Australia	This is a top example of the classic Australia...	The Peake	96	150.0	South Australia	McLaren Vale	N
59984	Australia	This is a throwback to those brash, flavor-exu...	One	95	95.0	South Australia	Langhorne Creek	N

In [23]:
```
# q9.hint()

# q9.solution()
```

Solution:

```
top_oceania_wines = reviews.loc[
    (reviews.country.isin(['Australia', 'New Zealand']))
    & (reviews.points >= 95)
]
```

Keep going

Introduction

Now you are ready to get a deeper understanding of your data.

Run the following cell to load your data and some utility functions (including code to check your answers).

In [2]:
```
import sys
from pathlib import Path
learntools_dir = Path().absolute().parents[1]
sys.path.append(str(learntools_dir))

import pandas as pd
pd.set_option("display.max_rows", 5)
reviews = pd.read_csv("../../pandas/datasets/winemag-data-130k-v2.csv", i

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

reviews.head()
```

```
Setup complete.
```

Out[2]:

	country	description	designation	points	price	province	region_1	region_2
0	Italy	Aromas include tropical fruit, broom, brimston...	Vulkà Bianco	87	NaN	Sicily & Sardinia	Etna	NaN
1	Portugal	This is ripe and fruity, a wine that is smooth...	Avidagos	87	15.0	Douro	NaN	NaN
2	US	Tart and snappy, the flavors of lime flesh and...	NaN	87	14.0	Oregon	Willamette Valley	Willamette Valley
3	US	Pineapple rind, lemon pith and orange blossom ...	Reserve Late Harvest	87	13.0	Michigan	Lake Michigan Shore	NaN
4	US	Much like the regular bottling from 2012, this...	Vintner's Reserve Wild Child Block	87	65.0	Oregon	Willamette Valley	Willamette Valley

Exercises

1.

What is the median of the `points` column in the `reviews` DataFrame?

In [14]:
```
median_points = reviews.points.median()

# Check your answer
q1.check()
median_points
```

Correct

Out[14]: 88.0

In [3]:
```
median_points = reviews['points'].median()
```

```
# Check your answer
q1.check()
```

Correct

In [6]:
```
q1.hint()

q1.solution()
```

Hint: Use the `median` function (a built-in `pandas` function, like the `mean` function or the `unique` function).

Solution:

```
median_points = reviews.points.median()
```

2.

What countries are represented in the dataset? (Your answer should not include any duplicates.)

In [8]:
```
countries = reviews.country.unique()

# Check your answer
q2.check()
countries
```

Correct

Out[8]:
```
array(['Italy', 'Portugal', 'US', 'Spain', 'France', 'Germany',
       'Argentina', 'Chile', 'Australia', 'Austria', 'South Africa',
       'New Zealand', 'Israel', 'Hungary', 'Greece', 'Romania', 'Mexic
o',
       'Canada', nan, 'Turkey', 'Czech Republic', 'Slovenia',
       'Luxembourg', 'Croatia', 'Georgia', 'Uruguay', 'England',
       'Lebanon', 'Serbia', 'Brazil', 'Moldova', 'Morocco', 'Peru',
       'India', 'Bulgaria', 'Cyprus', 'Armenia', 'Switzerland',
       'Bosnia and Herzegovina', 'Ukraine', 'Slovakia', 'Macedonia'],
      dtype=object)
```

In [4]:
```
countries = reviews['country'].unique()

# Check your answer
q2.check()
```

Correct

In [11]:
```
q2.hint()

q2.solution()
```

Hint: Use the `unique` function to get a list of unique entries in a column.

Solution:

```
countries = reviews.country.unique()
```

3.

How often does each country appear in the dataset? Create a Series `reviews_per_country` mapping countries to the count of reviews of wines from that country.

In [13]:
```
reviews_per_country = reviews.country.value_counts()

# Check your answer
q3.check()
reviews_per_country
```

Correct

Out[13]:
```
country
US                          27177
France                      11174
                            ...
Bosnia and Herzegovina          1
Slovakia                        1
Name: count, Length: 41, dtype: int64
```

In [5]:
```
reviews_per_country = reviews['country'].value_counts()

# Check your answer
q3.check()
```

Correct

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

# q3.solution()
```

4.

Create variable `centered_price` containing a version of the `price` column with the mean price subtracted.

(Note: this 'centering' transformation is a common preprocessing step before applying various machine learning algorithms.)

In [16]:
```
centered_price = reviews.price - reviews.price.mean()

# Check your answer
q4.check()
centered_price
```

Correct

Out[16]:
```
0              NaN
1       -20.232932
           ...
65497    -4.232932
65498   -25.232932
Name: price, Length: 65499, dtype: float64
```

In [7]:
```
centered_price = reviews['price']-reviews['price'].mean()

# Check your answer
q4.check()
```

Correct

In [18]:
```
q4.hint()

q4.solution()
```

Hint: To get the mean of a column in a Pandas DataFrame, use the `mean` function.

Solution:

centered_price = reviews.price - reviews.price.mean()

5.

I'm an economical wine buyer. Which wine is the "best bargain"? Create a variable `bargain_wine` with the title of the wine with the highest points-to-price ratio in the dataset.

In [25]:
```
bargain_idx = (reviews.points / reviews.price).idxmax()
bargain_wine = reviews.loc[bargain_idx, 'title']
# Check your answer
q5.check()
```

Correct

In [26]:
```
bargain_idx
```

Out[26]:
```
64590
```

In [27]:
```
bargain_wine
```

Out[27]:
```
'Bandit NV Merlot (California)'
```

In [24]:
```
q5.hint()

q5.solution()
```

Hint: The `idxmax` method may be useful here.

Solution:

```
bargain_idx = (reviews.points / reviews.price).idxmax()
bargain_wine = reviews.loc[bargain_idx, 'title']
```

6.

There are only so many words you can use when describing a bottle of wine. Is a wine more likely to be "tropical" or "fruity"? Create a Series `descriptor_counts` counting how many times each of these two words appears in the `description` column in the dataset. (For simplicity, let's ignore the capitalized versions of these words.)

In [28]:
```
n_trop = reviews.description.map(lambda x: "tropical" in x).sum()
n_fruity = reviews.description.map(lambda x: "fruity" in x).sum()
descriptor_counts = pd.Series([n_trop, n_fruity], index=['tropical', 'fru

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

Correct

In [29]:
```
descriptor_counts
```

Out[29]:
```
tropical    1813
fruity      4632
dtype: int64
```

In [57]:
```
# q6.hint()

q6.solution()
```

Solution:

```
n_trop = reviews.description.map(lambda desc: "tropical" in desc).sum()
n_fruity = reviews.description.map(lambda desc: "fruity" in desc).sum()
descriptor_counts = pd.Series([n_trop, n_fruity], index=['tropical', 'fruity'])
```

7.

We'd like to host these wine reviews on our website, but a rating system ranging from 80 to 100 points is too hard to understand - we'd like to translate them into simple star ratings. A score of 95 or higher counts as 3 stars, a score of at least 85 but less than 95 is 2 stars. Any other score is 1 star.

Also, the Canadian Vintners Association bought a lot of ads on the site, so any wines from Canada should automatically get 3 stars, regardless of points.

Create a series `star_ratings` with the number of stars corresponding to each

review in the dataset.

In [32]:
```
def stars(row):
    if row.country == 'Canada':
        return 3
    elif row.points >= 95:
        return 3
    elif row.points >= 85:
        return 2
    else:
        return 1

star_ratings = reviews.apply(stars, axis='columns')

# Check your answer
q7.check()
```

Correct

In [33]:
```
star_ratings
```

Out[33]:
```
0        2
1        2
        ..
65497    2
65498    2
Length: 65499, dtype: int64
```

In [71]:
```
def p2s(p):
    if p >= 95:
        star = 3
    elif p >= 85:
        star = 2
    else:
        star = 1
    return star

star_ratings = reviews['points'].map(p2s)

# Check your answer
q7.check()
```

Correct

In [31]:
```
q7.hint()

q7.solution()
```

Hint: Begin by writing a custom function that accepts a row from the DataFrame as input and returns the star rating corresponding to the row. Then, use `DataFrame.apply` to apply the custom function to every row in the dataset.

Solution:

```
def stars(row):
    if row.country == 'Canada':
        return 3
    elif row.points >= 95:
        return 3
    elif row.points >= 85:
        return 2
    else:
        return 1

star_ratings = reviews.apply(stars, axis='columns')
```

Keep going

Continue to **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 [49]:
```
import sys
from pathlib import Path
learntools_dir = Path().absolute().parents[1]
sys.path.append(str(learntools_dir))

import pandas as pd

reviews = pd.read_csv("../datasets/winemag-data-52.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.")
```

```
WARNING:root:Ignoring repeated attempt to bind to globals
Setup complete.
```

In [ ]:
```
reviews
```

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. ใครคือผู้วิจารณ์ไวน์ที่พบบ่อยที่สุดในชุดข้อมูล สร้างซีรีส์ที่มีดัชนีเป็นหมวดหมู่ taster_twitter_handle จากชุดข้อมูล และค่าของซีรีส์จะนับจำนวนบทวิจารณ์ที่แต่ละคนเขียน

In [46]:
```
# Your code here
reviews_written1 = reviews.groupby('taster_twitter_handle').size()
reviews_written1
```

Out[46]:
```
taster_twitter_handle
@kerinokeefe     8
@mattkettmann    2
@paulgwine       5
@vboone          4
@vossroger       7
@wineschach      7
dtype: int64
```

In [43]:
```
# Your code here
reviews_written = reviews.groupby('taster_twitter_handle').taster_twitter
```

In [11]:
```
reviews_written
```

Out[11]:
```
taster_twitter_handle
@kerinokeefe     8
@mattkettmann    2
@paulgwine       5
@vboone          4
@vossroger       7
@wineschach      7
Name: taster_twitter_handle, dtype: int64
```

In [ ]:
```
# 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). ไวน์ชนิดใดดีที่สุดที่จะซื้อได้ในราคาที่กำหนดไว้ สร้างซีรีส์ที่มีดัชนีเป็นราคาไวน์และมีค่าเป็นจำนวนคะแนนสูงสุดที่ไวน์ราคาดังกล่าวได้รับในการรีวิว เรียงลำดับค่าตามราคาจากน้อยไปมาก (โดยที่ 4.0 ดอลลาร์อยู่ที่ด้านบนและ 3300.0 ดอลลาร์อยู่ที่ด้านล่าง)

In [51]:
```
best_rating_per_price1 = reviews.groupby('price').points.max().sort_index
# Check your answer
# q2.check()
```

In [52]:
```
best_rating_per_price1
```

Out[52]:
```
price
9.0     86
10.0    87
11.0    92
12.0    92
13.0    95
14.0    87
15.0    87
16.0    95
17.0    90
19.0    92
20.0    94
21.0    95
22.0    94
23.0    87
24.0    90
27.0    92
28.0    87
30.0    87
32.0    90
34.0    94
35.0    90
40.0    90
50.0    95
65.0    95
69.0    95
Name: points, dtype: int64
```

In [53]:
```
# q2.hint()

q2.solution()
```

Solution:

```
best_rating_per_price = reviews.groupby('price')['points'].max().sort_index()
```

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. ราคาต่ำสุดและสูงสุดของไวน์แต่ละพันธุ์คือเท่าไร สร้างดาต้าเฟรมที่มีดัชนีเป็นหมวดหมู่พันธุ์จากชุดข้อมูลและมีค่าเป็นค่าต่ำสุดและสูงสุดของหมวดหมู่พันธุ์เหล่านั้น

In [14]:
```
reviews.variety.unique()
```

Out[14]:
```
array(['White Blend', 'Portuguese Red', 'Pinot Gris', 'Riesling',
       'Pinot Noir', 'Tempranillo-Merlot', 'Frappato', 'Gewürztraminer',
       'Cabernet Sauvignon', 'Nerello Mascalese', 'Chardonnay', 'Malbe
c',
       'Tempranillo Blend', 'Meritage', 'Red Blend', 'Merlot',
       "Nero d'Avola", 'Chenin Blanc', 'Gamay', 'Sauvignon Blanc',
       'Viognier-Chardonnay', 'Primitivo', 'Catarratto', 'Inzolia',
       'Petit Verdot'], dtype=object)
```

In [58]:
```
price_extremes = reviews.groupby('variety').price.agg([min, max])

# Check your answer
# q3.check()
```

```
/var/folders/83/3fg00w111r7bf7rcsz4nznlh0000gn/T/ipykernel_22562/77733017
9.py:1: FutureWarning: The provided callable <built-in function min> is cu
rrently using SeriesGroupBy.min. In a future version of pandas, the provid
ed callable will be used directly. To keep current behavior pass the strin
g "min" instead.
  price_extremes = reviews.groupby('variety').price.agg([min, max])
/var/folders/83/3fg00w111r7bf7rcsz4nznlh0000gn/T/ipykernel_22562/77733017
9.py:1: FutureWarning: The provided callable <built-in function max> is cu
rrently using SeriesGroupBy.max. In a future version of pandas, the provid
ed callable will be used directly. To keep current behavior pass the strin
g "max" instead.
  price_extremes = reviews.groupby('variety').price.agg([min, max])
```

In [59]:
```
price_extremes
```

Out[59]:

variety	min	max
Cabernet Sauvignon	19.0	34.0
Catarratto	17.0	17.0
Chardonnay	12.0	12.0
Chenin Blanc	16.0	16.0
Frappato	16.0	16.0
Gamay	9.0	14.0
Gewürztraminer	12.0	30.0
Inzolia	13.0	13.0
Malbec	13.0	30.0
Meritage	32.0	32.0
Merlot	9.0	22.0
Nerello Mascalese	NaN	NaN
Nero d'Avola	10.0	35.0
Petit Verdot	22.0	22.0
Pinot Gris	14.0	27.0
Pinot Noir	20.0	69.0
Portuguese Red	15.0	15.0
Primitivo	11.0	11.0
Red Blend	17.0	50.0
Riesling	13.0	24.0
Sauvignon Blanc	14.0	20.0
Tempranillo Blend	28.0	28.0
Tempranillo-Merlot	15.0	15.0
Viognier-Chardonnay	15.0	15.0
White Blend	13.0	19.0

In [56]:
```
# q3.hint()

# q3.solution()
```

Solution:

```
price_extremes = reviews.groupby('variety').price.agg([min, max])
```

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). ไวน์พันธุ์ใดมีราคาแพงที่สุด สร้างตัวแปร sorted_varieties ที่มีสำเนาของดาต้าเฟรมจากคำถามก่อนหน้า โดยที่ไวน์พันธุ์ต่างๆ จะถูกเรียงลำดับจากมากไปน้อยตามราคาต่ำสุด จากนั้นจึงเรียงตามราคาสูงสุด (เพื่อตัดสินผลเสมอ)

In [64]:
```
sorted_varieties = price_extremes.sort_values(by=['min', 'max'],ascending

# Check your answer
# q4.check()
```

In [65]:
```
sorted_varieties
```

Out[65]:

variety	min	max
Meritage	32.0	32.0
Tempranillo Blend	28.0	28.0
Petit Verdot	22.0	22.0
Pinot Noir	20.0	69.0
Cabernet Sauvignon	19.0	34.0
Red Blend	17.0	50.0
Catarratto	17.0	17.0
Chenin Blanc	16.0	16.0
Frappato	16.0	16.0
Portuguese Red	15.0	15.0
Tempranillo-Merlot	15.0	15.0
Viognier-Chardonnay	15.0	15.0
Pinot Gris	14.0	27.0
Sauvignon Blanc	14.0	20.0
Malbec	13.0	30.0
Riesling	13.0	24.0
White Blend	13.0	19.0
Inzolia	13.0	13.0
Gewürztraminer	12.0	30.0
Chardonnay	12.0	12.0
Primitivo	11.0	11.0
Nero d'Avola	10.0	35.0
Merlot	9.0	22.0
Gamay	9.0	14.0
Nerello Mascalese	NaN	NaN

In [67]:
```
sorted_varieties2 = price_extremes.sort_values(by=['min', 'max'],ascendin
sorted_varieties2
```

Out[67]:

variety	min	max
Meritage	32.0	32.0
Tempranillo Blend	28.0	28.0
Petit Verdot	22.0	22.0
Pinot Noir	20.0	69.0
Cabernet Sauvignon	19.0	34.0
Catarratto	17.0	17.0
Red Blend	17.0	50.0
Chenin Blanc	16.0	16.0
Frappato	16.0	16.0
Portuguese Red	15.0	15.0
Tempranillo-Merlot	15.0	15.0
Viognier-Chardonnay	15.0	15.0
Sauvignon Blanc	14.0	20.0
Pinot Gris	14.0	27.0
Inzolia	13.0	13.0
White Blend	13.0	19.0
Riesling	13.0	24.0
Malbec	13.0	30.0
Chardonnay	12.0	12.0
Gewürztraminer	12.0	30.0
Primitivo	11.0	11.0
Nero d'Avola	10.0	35.0
Gamay	9.0	14.0
Merlot	9.0	22.0
Nerello Mascalese	NaN	NaN

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

# q4.solution()
```

5.

สร้างซีรีส์ที่มีดัชนีเป็นนักวิจารณ์และมีค่าเป็นคะแนนรีวิวเฉลี่ยที่นักวิจารณ์คนนั้นให้ คำแนะนำ: คุณ

จะต้องใช้คอลัมน์ taster_name และ points 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 [28]:
```
reviewer_mean_ratings = reviews.groupby('taster_name').points.mean()

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

In [30]:
```
reviews.taster_name.unique()
```

Out[30]:
```
array(['Kerin O'Keefe', 'Roger Voss', 'Paul Gregutt',
       'Alexander Peartree', 'Michael Schachner', 'Anna Lee C. Iijima',
       'Virginie Boone', 'Matt Kettmann', nan], dtype=object)
```

In [29]:
```
reviewer_mean_ratings
```

Out[29]:
```
taster_name
Alexander Peartree    87.000000
Anna Lee C. Iijima    87.000000
Kerin O'Keefe         87.000000
Matt Kettmann         87.000000
Michael Schachner     86.428571
Paul Gregutt          86.600000
Roger Voss            86.571429
Virginie Boone        86.750000
Name: points, dtype: float64
```

In [27]:
```
# q5.hint()

q5.solution()
```

Solution:

reviewer_mean_ratings = reviews.groupby('taster_name').points.mean()

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 [31]:
```
reviewer_mean_ratings.describe()
```

Out[31]:
```
count     8.000000
mean     86.793750
std       0.236761
min      86.428571
25%      86.592857
50%      86.875000
75%      87.000000
max      87.000000
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. การผสมผสานของประเทศและพันธุ์องุ่นแบบใดที่พบได้บ่อยที่สุด สร้างซีรีส์ที่มีดัชนีเป็น MultiIndex ของคู่ไวน์ {country, variety} ตัวอย่างเช่น ไวน์ปิโนต์นัวร์ที่ผลิตในสหรัฐอเมริกาควรจับคู่กับ {"US", "Pinot Noir"} เรียงลำดับค่าในซีรีส์ตามลำดับจากมากไปน้อยตามจำนวนไวน์

In [33]:
```
country_variety_counts = reviews.groupby(['country', 'variety']).size().s

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

In [34]:
```
country_variety_counts
```

Out[34]:
```
country    variety
US         Pinot Noir             5
Italy      White Blend            4
US         Sauvignon Blanc        3
Italy      Nero d'Avola           3
US         Red Blend              3
France     Gamay                  3
Italy      Red Blend              3
US         Riesling               2
           Cabernet Sauvignon     2
Argentina  Malbec                 2
France     Gewürztraminer         2
Italy      Catarratto             1
France     Pinot Gris             1
Chile      Petit Verdot           1
           Viognier-Chardonnay    1
US         Pinot Gris             1
           Merlot                 1
           Meritage               1
           Chenin Blanc           1
           Chardonnay             1
Spain      Tempranillo-Merlot     1
Italy      Frappato               1
Spain      Tempranillo Blend      1
Portugal   Portuguese Red         1
Germany    Gewürztraminer         1
           Riesling               1
Chile      Merlot                 1
Italy      Cabernet Sauvignon     1
           Nerello Mascalese      1
           Inzolia                1
           Primitivo              1
dtype: int64
```

In [32]:
```
# q6.hint()

q6.solution()
```

Solution:

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

Keep going

Move on to the **data types and missing data**.

Introduction

Run the following cell to load your data and some utility functions.

In [1]:
```
import sys
from pathlib import Path
learntools_dir = Path().absolute().parents[1]
sys.path.append(str(learntools_dir))

import pandas as pd

reviews = pd.read_csv("../../pandas/datasets/winemag-data-130k-v2.csv", i

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

```
Setup complete.
```

Exercises

1.

What is the data type of the `points` column in the dataset?

In [ ]:
```
# Your code here
dtype = ____

# Check your answer
q1.check()
```

In [2]:
```
#_COMMENT_IF(PROD)_
q1.hint()
#_COMMENT_IF(PROD)_
q1.solution()
```

Hint: `dtype` is an attribute of a DataFrame or Series.

Solution:

dtype = reviews.points.dtype

2.

Create a Series from entries in the `points` column, but convert the entries to strings. Hint: strings are `str` in native Python.

```

```

In [ ]:
```
point_strings = ____

# Check your answer
q2.check()
```

In [3]:
```
#_COMMENT_IF(PROD)_
q2.hint()
#_COMMENT_IF(PROD)_
q2.solution()
```

Hint: Convert a column of one type to another by using the `astype` function.

Solution:

```
point_strings = reviews.points.astype(str)
```

3.

Sometimes the price column is null. How many reviews in the dataset are missing a price?

In [5]:
```
n_missing_prices = ____

# Check your answer
q3.check()
```

Correct

In [4]:
```
#_COMMENT_IF(PROD)_
q3.hint()
#_COMMENT_IF(PROD)_
q3.solution()
```

Hint: Use `pd.isnull()`.

Solution:

```
missing_price_reviews = reviews[reviews.price.isnull()]
n_missing_prices = len(missing_price_reviews)
# Cute alternative solution: if we sum a boolean series, True is treated as 1 and False as 0
n_missing_prices = reviews.price.isnull().sum()
# or equivalently:
n_missing_prices = pd.isnull(reviews.price).sum()
```

4.

What are the most common wine-producing regions? Create a Series counting the number of times each value occurs in the `region_1` field. This field is often missing data, so replace missing values with `Unknown`. Sort in descending order. Your

output should look something like this:

```
Unknown                    21247
Napa Valley                 4480
                           ...
Bardolino Superiore            1
Primitivo del Tarantino        1
Name: region_1, Length: 1230, dtype: int64
```

In [9]:
```
reviews_per_region = reviews.region_1.fillna('Unknown').value_counts().so

# Check your answer
q4.check()
```

Correct

In [10]:
```
reviews_per_region
```

Out[10]:
```
region_1
Unknown                      10755
Napa Valley                   2226
Columbia Valley (WA)          2040
Russian River Valley          1578
California                    1340
                             ...
Mâcon-Péronne                    1
Brulhois                         1
San Marino                       1
Gutturnio Colli Piacentini       1
Vino da Mesa de Toledo           1
Name: count, Length: 1112, dtype: int64
```

In [6]:
```
#_COMMENT_IF(PROD)_
q4.hint()
#_COMMENT_IF(PROD)_
q4.solution()
```

Hint: Use `fillna()`, `value_counts()`, and `sort_values()`.

Solution:

```
reviews_per_region = reviews.region_1.fillna('Unknown').value_counts().sort_values(ascending=False)
```

Keep going

Move on to **renaming and combining**.