This notebook is an exercise in the Pandas course. You can reference the tutorial at this link.

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 [ ]:
```
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.indexing_selecting_and_assigning import *
print("Setup complete.")
```

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

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

Exercises

1.

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

In [ ]:
```
# Your code here
desc = reviews.description
# Check your answer
q1.check()
```

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

2.

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

In [ ]:
```
# Your code here
first_description = desc[0]
# Check your answer
q2.check()
first_description
```

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

3.

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

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


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

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

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 [ ]:
```
first_descriptions = reviews.description[0:10]

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

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

5.

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

In other words, generate the following 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 [ ]:
```
indices = [1, 2, 3, 5, 8]
sample_reviews = reviews.loc[indices]

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

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

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
100	US	New York	Finger Lakes	Finger Lakes

In [ ]:
```
indices=[0,1,10,100]
cols=["country","province","region_1","region_2"]
df = reviews.loc[indices,cols]
# Check your answer
q6.check()
df
```

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

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 [ ]:
```
cols=["country","variety"]
df=reviews.loc[0:99,cols]
# Check your answer
q7.check()
df
```

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

8.

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

In [ ]:
```
italian_wines = reviews.loc[(reviews.country=="Italy")]

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

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

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 [ ]:
```
top_oceania_wines = reviews.loc[(reviews.country.isin(["Australia","New Zealand"]))& (reviews.points>=95)] # Check your answer
q9.check()
top_oceania_wines

# Check your answer
```

```
q9.check()
top_oceania_wines
```

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

Keep going

Move on to learn about **summary functions and maps**.

Have questions or comments? Visit the Learn Discussion forum to chat with other Learners.