Usage Guidelines

This lesson is part of the **DS Lab core curriculum**. For that reason, this notebook can only be used on your WQU virtual machine.

This means:

- ⓧ No downloading this notebook.
- ⓧ No re-sharing of this notebook with friends or colleagues.
- ⓧ No downloading the embedded videos in this notebook.
- ⓧ No re-sharing embedded videos with friends or colleagues.
- ⓧ No adding this notebook to public or private repositories.
- ⓧ No uploading this notebook (or screenshots of it) to other websites, including websites for study resources.

1.5. Housing in Brazil BR

[3]:
```
import wqet_grader

wqet_grader.init("Project 1 Assessment")
```

In this assignment, you'll work with a dataset of homes for sale in Brazil. Your goal is to determine if there are regional differences in the real estate market. Also, you will look at southern Brazil to see if there is a relationship between home size and price, similar to what you saw with housing in some states in Mexico.

Note: There are are 19 graded tasks in this assignment, but you only need to complete 18. Once you've successfully completed 18 tasks, you'll be automatically enrolled in the next project, and this assignment will be closed. This means that you might not be allowed to complete the last task. So if you get an error saying that you've already complete the course, that's good news! Move to project 2.

Before you start: Import the libraries you'll use in this notebook: Matplotlib, pandas, and plotly. Be sure to import them under the aliases we've used in this project.

[4]:
```
# Import Matplotlib, pandas, and plotly
import pandas as pd
import matplotlib.pyplot as plt
import plotly.express as px
```

Prepare Data

In this assignment, you'll work with real estate data from Brazil. In the `data` directory for this project there are two CSV that you need to import and clean, one-by-one.

Import

First, you are going to import and clean the data in `data/brasil-real-estate-1.csv`.

Task 1.5.1: Import the CSV file `data/brasil-real-estate-1.csv` into the DataFrame `df1`.

[27]:
```
df1 = pd.read_csv("data/brasil-real-estate-1.csv")
df1.head()
```

[27]:

	property_type	place_with_parent_names	region	lat-lon	area_m2	price_usd
0	apartment	|Brasil|Alagoas|Maceió|	Northeast	-9.6443051,-35.7088142	110.0	$187,230.85
1	apartment	|Brasil|Alagoas|Maceió|	Northeast	-9.6430934,-35.70484	65.0	$81,133.37
2	house	|Brasil|Alagoas|Maceió|	Northeast	-9.6227033,-35.7297953	211.0	$154,465.45
3	apartment	|Brasil|Alagoas|Maceió|	Northeast	-9.622837,-35.719556	99.0	$146,013.20
4	apartment	|Brasil|Alagoas|Maceió|	Northeast	-9.654955,-35.700227	55.0	$101,416.71

[28]:
```
wqet_grader.grade("Project 1 Assessment", "Task 1.5.1", df1)
```

Excellent! Keep going.

Score: 1

Before you move to the next task, take a moment to inspect `df1` using the `info` and `head` methods. What issues do you see in the data? What cleaning will you need to do before you can conduct your analysis?

[29]:
```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
```

[29]:
```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12834 entries, 0 to 12833
Data columns (total 6 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   property_type           12834 non-null  object 
 1   place_with_parent_names  12834 non-null  object 
 2   region                  12834 non-null  object 
 3   lat-lon                 11551 non-null  object 
 4   area_m2                 12834 non-null  float64
 5   price_usd               12834 non-null  object 
dtypes: float64(1), object(5)
memory usage: 601.7+ KB
```

Task 1.5.2: Drop all rows with `NaN` values from the DataFrame `df1`.

[30]:
```
#Checking for null values
df1.isnull().sum()
```

[30]:
```
property_type                0
place_with_parent_names      0
region                       0
lat-lon                   1283
area_m2                      0
price_usd                    0
dtype: int64
```

[31]:
```
df1 = df1.dropna()
```

[32]:
```
#Checking for null values
df1.isnull().sum()
```

[32]:
```
property_type             0
place_with_parent_names    0
region                     0
lat-lon                    0
area_m2                    0
price_usd                  0
dtype: int64
```

[33]:
```
wqet_grader.grade("Project 1 Assessment", "Task 1.5.2", df1)
```

Yes! Your hard work is paying off.

Score: 1

Task 1.5.3: Use the `"lat-lon"` column to create two separate columns in `df1`: `"lat"` and `"lon"`. Make sure that the data type for these new columns is `float`.

[34]:
```
df1[['lat', 'lon']] = df1['lat-lon'].str.split(",", expand = True)
```

[35]:
```
df1['lat'] = pd.to_numeric(df1['lat'])
df1['lon'] = pd.to_numeric(df1['lon'])
```

Task 1.5.4: Use the `"place_with_parent_names"` column to create a `"state"` column for `df1`. (Note that the state name always appears after `"|Brasil|"` in each string.)

[38]:
```
df1["state"] = df1["place_with_parent_names"].str.split("|",expand=True)[2]
```

[40]:
```
wqet_grader.grade("Project 1 Assessment", "Task 1.5.4", df1)
```

Very impressive.

Score: 1

Task 1.5.5: Transform the `"price_usd"` column of `df1` so that all values are floating-point numbers instead of strings.

[41]:
```
df1['lat'] = pd.to_numeric(df1['lat'])
```

[42]:
```
#Cleaning df1 by dropping rows with NaN values.
#Then remove the "$" and "," characters from "price_usd"
#and recast the values in the column as floats.
df1['price_usd'] = df1['price_usd'].str.replace('$','',regex = False).str.replace(',','',regex = False).astype(float)
```

[43]:
```
wqet_grader.grade("Project 1 Assessment", "Task 1.5.5", df1)
```

You = coding □

Task 1.5.6: Drop the `"lat-lon"` and `"place_with_parent_names"` columns from `df1`.

[44]:
```
df1 = df1.drop(['place_with_parent_names'], axis=1)
df1 = df1.drop(['lat-lon'], axis=1)
```

[46]:
```
wqet_grader.grade("Project 1 Assessment", "Task 1.5.6", df1)
```

Boom! You got it.

Score: 1

Now that you have cleaned `data/brasil-real-estate-1.csv` and created `df1`, you are going to import and clean the data from the second file, `brasil-real-estate-2.csv`.

Task 1.5.7: Import the CSV file `brasil-real-estate-2.csv` into the DataFrame `df2`.

[5]:
```
df2 =pd.read_csv("data/brasil-real-estate-2.csv")
```

[6]:
```
wqet_grader.grade("Project 1 Assessment", "Task 1.5.7", df2)
```

Score: 1

Before you jump to the next task, take a look at `df2` using the `info` and `head` methods. What issues do you see in the data? How is it similar or different from `df1` ?

[13]:
```
df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12833 entries, 0 to 12832
Data columns (total 8 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   property_type  12833 non-null  object 
 1   state          12833 non-null  object 
 2   region         12833 non-null  object 
 3   lat            12833 non-null  float64
 4   lon            12833 non-null  float64
 5   area_m2        11293 non-null  float64
 6   price_brl      12833 non-null  float64
 7   price_usd      12833 non-null  float64
dtypes: float64(5), object(3)
memory usage: 802.2+ KB
```

[8]:
```
df2.head()
```

[8]:

	property_type	state	region	lat	lon	area_m2	price_brl
0	apartment	Pernambuco	Northeast	-8.134204	-34.906326	72.0	414222.98
1	apartment	Pernambuco	Northeast	-8.126664	-34.903924	136.0	848408.53
2	apartment	Pernambuco	Northeast	-8.125550	-34.907601	75.0	299438.28

[66]:
```
df2['state'].unique()
```

[66]:
```
array(['Pernambuco', 'Piauí', 'Rio Grande do Norte', 'Rio Grande do Sul',
       'Rio de Janeiro', 'Rondônia', 'Santa Catarina', 'Sergipe',
       'São Paulo', 'Tocantins'], dtype=object)
```

Task 1.5.8: Use the `"price_brl"` column to create a new column named `"price_usd"`. (Keep in mind that, when this data was collected in 2015 and 2016, a US dollar cost 3.19 Brazilian reals.)

[19]:
```
df2["price_usd"] = df2["price_brl"]/3.19
```

[20]:
```
df2['price_usd'] = pd.to_numeric(df2['price_usd'])
```

[22]:
```
wqet_grader.grade("Project 1 Assessment", "Task 1.5.8", df2)
```

Good work!

Score: 1

Task 1.5.9: Drop the `"price_brl"` column from `df2`, as well as any rows that have `NaN` values.

[23]:
```
df2 = df2.drop(['price_brl'], axis=1)
```

[24]:
```
df2 = df2.dropna()
```

[25]:
```
wqet_grader.grade("Project 1 Assessment", "Task 1.5.9", df2)
```

OK! Now that you've cleaned the data from both CSV files and created `df1` and `df2`, it's time to combine them into a single DataFrame.

Task 1.5.10: Concatenate `df1` and `df2` to create a new DataFrame named `df`.

[47]:
```
frames = [df1, df2]
df = pd.concat(frames)
print("df shape:", df.shape)
```

```
df shape: (22844, 7)
```

[48]:
```
df.head()
```

[48]:

	property_type	region	area_m2	price_usd	lat	lon	state
0	apartment	Northeast	110.0	187230.85	-9.644305	-35.708814	Alagoas
1	apartment	Northeast	65.0	81133.37	-9.643093	-35.704840	Alagoas
2	house	Northeast	211.0	154465.45	-9.622703	-35.729795	Alagoas
3	apartment	Northeast	99.0	146013.20	-9.622837	-35.719556	Alagoas
4	apartment	Northeast	55.0	101416.71	-9.654955	-35.700227	Alagoas

[67]:
```
df['region'].unique()
```

[67]:
```
array(['Northeast', 'North', 'Central-West', 'Southeast', 'South'],
      dtype=object)
```

[49]:
```
wqet_grader.grade("Project 1 Assessment", "Task 1.5.10", df)
```

Explore

It's time to start exploring your data. In this section, you'll use your new data visualization skills to learn more about the regional differences in the Brazilian real estate market.

Complete the code below to create a `scatter_mapbox` showing the location of the properties in `df`.

[50]:
```
fig = px.scatter_mapbox(
    df,
    lat= df['lat'],
    lon=df['lon'],
    center={"lat": -14.2, "lon": -51.9},  # Map will be centered on Brazil
    width=600,
    height=600,
    hover_data=["price_usd"],  # Display price when hovering mouse over house
)

fig.update_layout(mapbox_style="open-street-map")

fig.show()
```

Task 1.5.11: Use the `describe` method to create a DataFrame `summary_stats` with the summary statistics for the `"area_m2"` and `"price_usd"` columns.

[51]:
```
summary_stats = df[["area_m2","price_usd"]].describe()
summary_stats
```

[51]:

	area_m2	price_usd
count	22844.000000	22844.000000
mean	115.020224	194987.315480
std	47.742932	103617.682978
min	53.000000	74892.340000
25%	76.000000	113898.770000
50%	103.000000	165697.555000
75%	142.000000	246900.880878
max	252.000000	525659.717868

[52]:
```
wqet_grader.grade("Project 1 Assessment", "Task 1.5.11", summary_stats)
```

That's the right answer. Keep it up!

Score: 1

Task 1.5.12: Create a histogram of `"price_usd"`. Make sure that the x-axis has the label `"Price [USD]"`, the y-axis has the label `"Frequency"`, and the plot has the title `"Distribution of Home Prices"`. Use Matplotlib (`plt`).

[53]:
```
df.info()
```

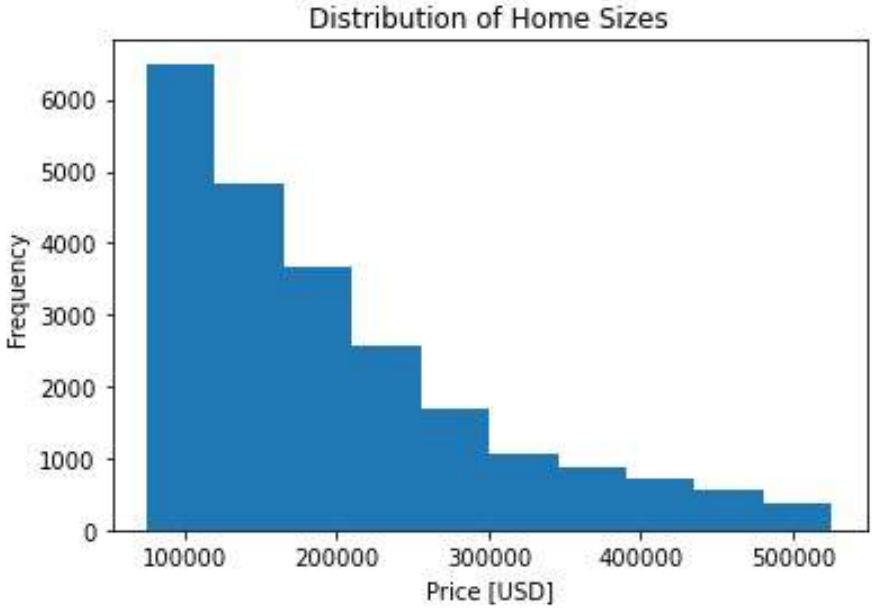
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 22844 entries, 0 to 12832
Data columns (total 7 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   property_type  22844 non-null  object 
 1   region         22844 non-null  object 
 2   area_m2        22844 non-null  float64
 3   price_usd      22844 non-null  float64
 4   lat            22844 non-null  float64
 5   lon            22844 non-null  float64
 6   state          22844 non-null  object 
dtypes: float64(4), object(3)
memory usage: 1.4+ MB
```

[54]:
```
# Build histogram
plt.hist(df['price_usd'])

# Label axes
plt.xlabel ("Price [USD]")
plt.ylabel ("Frequency")

# Add title
plt.title ("Distribution of Home Sizes");
```

```
# Don't change the code below 👇
plt.savefig("images/1-5-12.png", dpi=150)
```



55]:
```
with open("images/1-5-12.png", "rb") as file:
    wqet_grader.grade("Project 1 Assessment", "Task 1.5.12", file)
```

Party time! 🎉🎉🎉

Score: 1

Task 1.5.13: Create a horizontal boxplot of `"area_m2"`. Make sure that the x-axis has the label `"Area [sq meters]"` and the plot has the

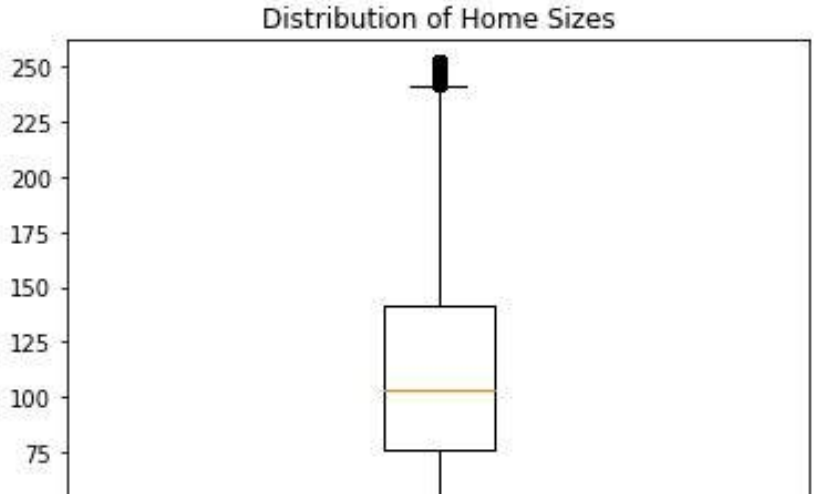
Task 1.5.13: Create a horizontal boxplot of `"area_m2"`. Make sure that the x-axis has the label `"Area [sq meters]"` and the plot has the title `"Distribution of Home Sizes"`. Use Matplotlib (`plt`).

[58]:
```
# Build box plot
plt.boxplot(df['area_m2'])


# Label x-axis
plt.xlabel ("Area [sq meters]")

# Add title
plt.title ("Distribution of Home Sizes");

# Don't change the code below 👇
plt.savefig("images/1-5-13.png", dpi=150)
```



[59]:
```
with open("images/1-5-13.png", "rb") as file:
    wqet_grader.grade("Project 1 Assessment", "Task 1.5.13", file)
```

Awesome work.

Score: 1

Task 1.5.14: Use the `groupby` method to create a Series named `mean_price_by_region` that shows the mean home price in each region in Brazil, sorted from smallest to largest.

[68]:
```
mean_price_by_region = df.groupby("region")["price_usd"].mean().sort_values(ascending = True)
mean_price_by_region
```

[68]:
```
region
Central-West    178596.283663
North           181308.958207
Northeast       185422.985441
South           189012.345265
Southeast       208996.762778
Name: price_usd, dtype: float64
```

[69]:
```
wqet_grader.grade("Project 1 Assessment", "Task 1.5.14", mean_price_by_region)
```

Yes! Your hard work is paying off.

Score: 1

Task 1.5.15: Use `mean_price_by_region` to create a bar chart. Make sure you label the x-axis as `"Region"` and the y-axis as `"Mean Price [USD]"`, and give the chart the title `"Mean Home Price by Region"`. Use pandas.

[71]:
```
# Build bar chart, label axes, add title
mean_price_by_region.plot(
    kind = "bar",
    xlabel = "Region",
    ylabel = "Mean Price [USD]",
    title = "Mean Home Price by Region"
);
# Don't change the code below 👇
plt.savefig("images/1-5-15.png", dpi=150)
```

[72]:
```
with open("images/1-5-15.png", "rb") as file:
    wqet_grader.grade("Project 1 Assessment", "Task 1.5.15", file)
```

Yes! Great problem solving.

Score: 1

Keep it up! You're halfway through your data exploration. Take one last break and get ready for the final push. 🚀

You're now going to shift your focus to the southern region of Brazil, and look at the relationship between home size and price.

Task 1.5.16: Create a DataFrame `df_south` that contains all the homes from `df` that are in the `"South"` region.

[75]:
```
df_south = df.loc[df['region'] == 'South']
df_south.head()
```

[75]:

	property_type	region	area_m2	price_usd	lat	lon	state
9304	apartment	South	127.0	296448.85	-25.455704	-49.292918	Paraná
9305	apartment	South	104.0	219996.25	-25.455704	-49.292918	Paraná
9306	apartment	South	100.0	194210.50	-25.460236	-49.293812	Paraná
9307	apartment	South	77.0	149252.94	-25.460236	-49.293812	Paraná
9308	apartment	South	73.0	144167.75	-25.460236	-49.293812	Paraná

Task 1.5.17: Use the `value_counts` method to create a Series `homes_by_state` that contains the number of properties in each state in `df_south`.

[81]:
```
homes_by_state = df_south['state'].value_counts()
homes_by_state
```

[81]:
```
Rio Grande do Sul    2643
Santa Catarina       2634
Paraná               2544
Name: state, dtype: int64
```

[82]:
```
wqet_grader.grade("Project 1 Assessment", "Task 1.5.17", homes_by_state)
```

Party time! 🎉🎉🎉

Score: 1

Task 1.5.18: Create a scatter plot showing price vs. area for the state in `df_south` that has the largest number of properties. Be sure to label the x-axis `"Area [sq meters]"` and the y-axis `"Price [USD]"`; and use the title `"<name of state>: Price vs. Area"`. Use Matplotlib (`plt`).

> **Tip:** You should replace `<name of state>` with the name of the state that has the largest number of properties.

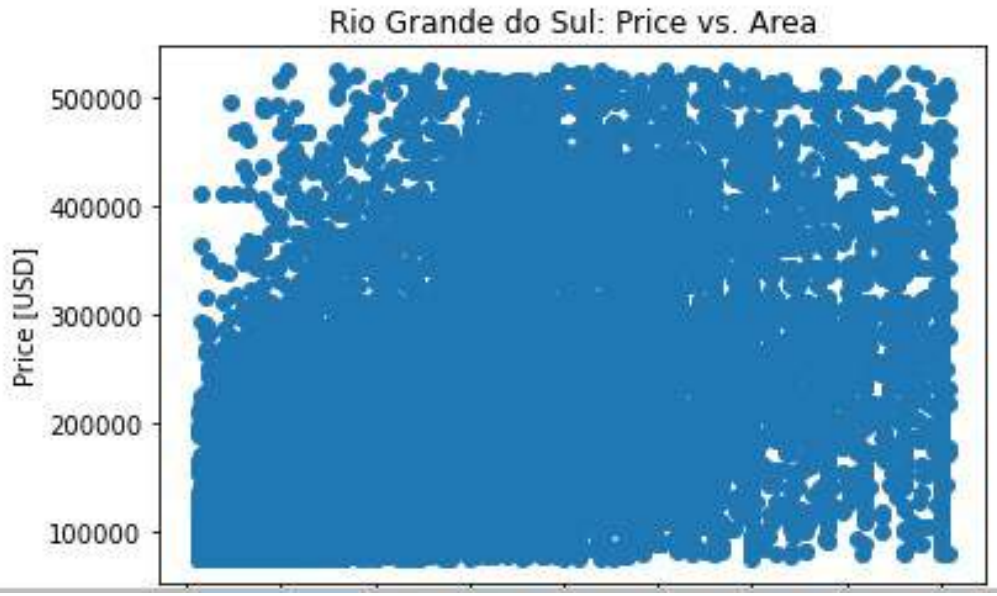
[83]:
```
# Subset data
df_south_rgs = df_south
x = df_south_rgs['area_m2']
y = df_south_rgs['price_usd']

# Build scatter plot
plt.scatter(x, y)

# Label axes
plt.xlabel("Area [sq meters]")
plt.ylabel("Price [USD]")

# Add title
plt.title("Rio Grande do Sul: Price vs. Area")

# Don't change the code below 👇
plt.savefig("images/1-5-18.png", dpi=150)
```



[ ]:
```
with open("images/1-5-18.png", "rb") as file:
    wqet_grader.grade("Project 1 Assessment", "Task 1.5.18", file)
```

Task 1.5.19: Create a dictionary `south_states_corr`, where the keys are the names of the three states in the `"South"` region of Brazil, and their associated values are the correlation coefficient between `"area_m2"` and `"price_usd"` in that state.

As an example, here's a dictionary with the states and correlation coefficients for the Southeast region. Since you're looking at a different region, the states and coefficients will be different, but the structure of the dictionary will be the same.

```
{'Espírito Santo': 0.6311332554173303,
 'Minas Gerais': 0.5830029036378931,
 'Rio de Janeiro': 0.4554077103515366,
 'São Paulo': 0.45882050624839366}
```

[ ]:
```
south_states_corr = ...

south_states_corr
```

[ ]:
```
wqet_grader.grade("Project 1 Assessment", "Task 1.5.19", south_states_corr)
```