Homework 06

⚠ Before you start ⚠

Duplicate this Jupyter Notebook in your `week-07` folder (right-click -> Duplicate) and then add your last name to the beginning of it (ie. `blevins-hw-06.ipynb` - otherwise you risk having all your work overwritten when you try to sync your GitHub repository with your instructor's repository.

Student Name: Rayce Loveland

Overview

This homework assignment will help you learn how to use the Pandas library to explore tabular data by applying some of the concepts and lessons from Melanie Walsh, Pandas I to a new dataset.

This week we're going to use a spreadsheet of historical data transcribed by CU Denver history major Ryan Hanlon as part of his final project for the course Introduction to Digital Studies in Spring 2021. The data consists of a passenger list from a steamship that arrived in Boston on April 9, 1884 carrying several hundred immigrants.

Description of image

The Steamship Grecian

The nine-page passenger list from the *Steamship Grecian* was submitted to authorities at the Port of Boston. This document was later scanned by the Church of Latter Day Saints and made available through its FamilySearch online archive.

Description of image

Passenger list from the Steamship Grecian

Ryan then transcribed the data in Spring 2021 into a spreadsheet formatted as a CSV (comma separated value) file contained in this folder: `boston-passenger-list-1884.csv` .

For this homework, you will be following many of Melanie Walsh's steps in Pandas I and then adapting them to fit this new dataset.

---

1. Import the Pandas library (use the alias `pd`) and read in the CSV file, storing the contents of the file as a dataframe named `passengers_df`. Add a second line of code to display the contents of the dataframe (truncated).

In [9]:
```
#Your Code Here
import pandas as pd

passengers_df = pd.read_csv('boston-passenger-list-1884.csv', delimiter=",")
passengers_df
```

Out[9]:

	first_name	last_name	date	age	native_country	destination_city	destination_state
0	Jno	McNab	09 Apr 1884	21	Scotland	Suelpla	Canada
1	Thos	Campbell	09 Apr 1884	24	Scotland	Suelpla	Canada
2	Jas	Mitchell	09 Apr 1884	23	Scotland	Detroit	MI
3	Don	Cumming	09 Apr 1884	24	Scotland	Detroit	MI
4	Jno	McKinlay	09 Apr 1884	24	Scotland	Winnipeg	Canada
...	...	...	...	...	...	...	...
510	Mich	Mulkern	09 Apr 1884	15	Ireland	Boston	MA
511	Math	Griffins	09 Apr 1884	2	Ireland	Pittsburg	PA
512	T	McDermott	09 Apr 1884	35	United States	Boston	MA
513	Wm	Hewitt	09 Apr 1884	25	United States	Boston	MA
514	F	Doherty	09 Apr 1884	27	United States	Boston	MA

515 rows × 9 columns

2. Display the **first 6 rows** of the dataframe.

In [11]:
```
#Your Code Here
passengers_df.head(6)
```

Out[11]:

	first_name	last_name	date	age	native_country	destination_city	destination_state	oc
0	Jno	McNab	09 Apr 1884	21	Scotland	Suelpla	Canada	
1	Thos	Campbell	09 Apr 1884	24	Scotland	Suelpla	Canada	
2	Jas	Mitchell	09 Apr 1884	23	Scotland	Detroit	MI	
3	Don	Cumming	09 Apr 1884	24	Scotland	Detroit	MI	B
4	Jno	McKinlay	09 Apr 1884	24	Scotland	Winnipeg	Canada	
5	John	Wilson	09 Apr 1884	28	Scotland	Boston	MA	

3. Show a **random sample of 10 rows** from your dataframe.

In [13]:
```
#Your Code Here
passengers_df.sample(10)
```

Out[13]:

	first_name	last_name	date	age	native_country	destination_city	destination_state
297	Pat	Durstan	09 Apr 1884	17	Ireland	Providence	RI
66	Eliz	Caldwell	09 Apr 1884	40	Ireland	Boston	MA
343	Bgt	McDonough	09 Apr 1884	44	Ireland	Brooklyn	NY
418	Thos	Mitchell	09 Apr 1884	34	Ireland	Boston	MA
16	John	Little	09 Apr 1884	26	Scotland	Woodstock	VT
302	John	Parsons	09 Apr 1884	30	Ireland	Chicago	IL
111	Robt	Gilmore	09 Apr 1884	25	Ireland	Boston	MA
68	Jas	Gallagher	09 Apr 1884	46	Ireland	Boston	MA
177	Kate	Mellett	09 Apr 1884	18	Ireland	Taunton	MA
452	Margt	Griffin	09 Apr 1884	17	Ireland	Pittsburg	PA

4. What are **two historical questions** about this list of passengers that you might be able to answer using Pandas?

Your answer here.

1. Pandas can help determine where most migrants migrated to and where most migrants emigrated from.
2. Another thing Pandas could help answer using this dataset is what these occupations were for these immigrants coming to the Americas.

Analyzing the Data

5. Calculate "summary statistics" for the passenger data.

In [18]:
```
#Your Code Here
passengers_df.describe(include='all')
```

Out[18]:

	first_name	last_name	date	age	native_country	destination_city	destinati
count	512	515	515	515.000000	515	515	
unique	102	170	1	NaN	3	56	
top	Mary	Doherty	09 Apr 1884	NaN	Ireland	Boston	
freq	59	12	515	NaN	461	107	
mean	NaN	NaN	NaN	21.440777	NaN	NaN	
std	NaN	NaN	NaN	13.115392	NaN	NaN	
min	NaN	NaN	NaN	0.000000	NaN	NaN	
25%	NaN	NaN	NaN	11.000000	NaN	NaN	
50%	NaN	NaN	NaN	20.000000	NaN	NaN	
75%	NaN	NaN	NaN	28.000000	NaN	NaN	
max	NaN	NaN	NaN	64.000000	NaN	NaN	

In [19]:
```
passengers_df['last_name'].value_counts().head(1)
```

Out[19]:
```
last_name
Doherty    12
Name: count, dtype: int64
```

In [20]:
```
passengers_df['occupation'].value_counts()
```

Out[20]:
```
occupation
Laborer       189
Child         132
Domestic      116
Wife           64
Farmer          8
Boatman         3
Blacksmith      1
Miller          1
Clerk           1
Name: count, dtype: int64
```

In [21]:
```
passengers_df['age'].max()
```

Out[21]:
```
64
```

6. Looking at the summary statistics, answer the following questions:

- What is the most frequently occuring last name?
- How often does the most frequently occuring last name appear?
- How many different *kinds* of occupations are listed in the data?
- How old is the oldest passenger?

Your answers here:

- What is the most frequently occuring last name? The most common last name is Doherty.
- How often does the most frequently occuring last name appear? It appears 12 times.
- How many different *kinds* of occupations are listed in the data? There were nine types of labeled occupations on the dataset.
- How old is the oldest passenger? The oldest passenger was 64 years old.

7. Write code to answer: what was the **median** age of the passengers?

In [25]:
```python
#Your Code Here
passengers_df['age'].median()
```

Out[25]:
```
20.0
```

8. What were the **ten most frequent cities** that passengers were traveling to and how many of them were going to each of these cities?

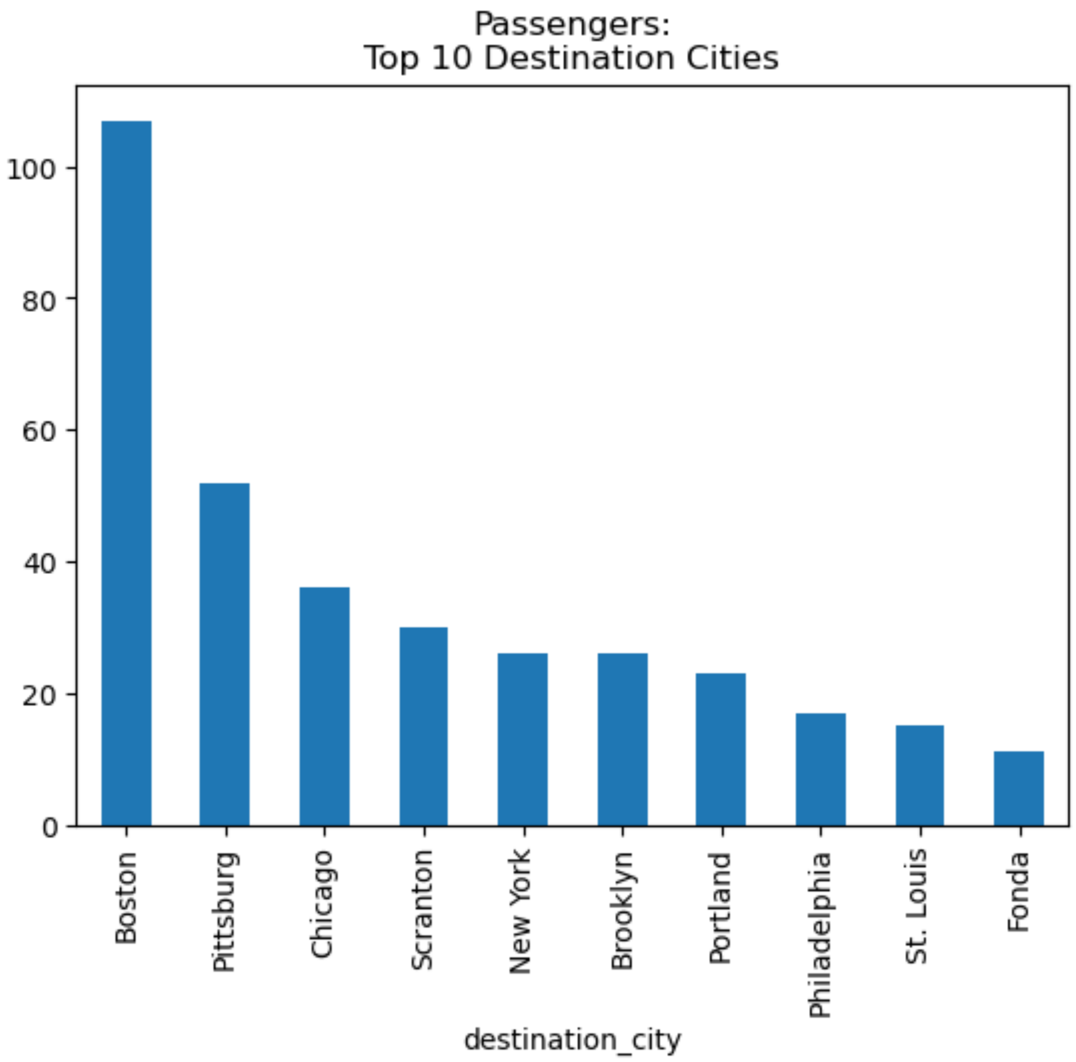
In [27]:
```python
#Your Code Here
passengers_df['destination_city'].value_counts().head(10)
```

Out[27]:
```
destination_city
Boston          107
Pittsburg        52
Chicago          36
Scranton         30
New York         26
Brooklyn         26
Portland         23
Philadelphia     17
St. Louis        15
Fonda            11
Name: count, dtype: int64
```

9. Follow Walsh's example and adapt her code to make a bar chart of the **top ten most frequent destination cities** based on **how many passengers** were going to each of them.

```
In [29]: #Your Code Here
         passengers_df['destination_city'].value_counts()[:10].plot(kind='bar', title='Passe
```

```
Out[29]: <Axes: title={'center': 'Passengers:\nTop 10 Destination Cities'}, xlabel='destina
         tion_city'>
```



10. Where were passengers coming from? Print out **the most frequent countries** they were immigrating from and how many passengers were coming from each country. Hint: use `value_counts()` and index.

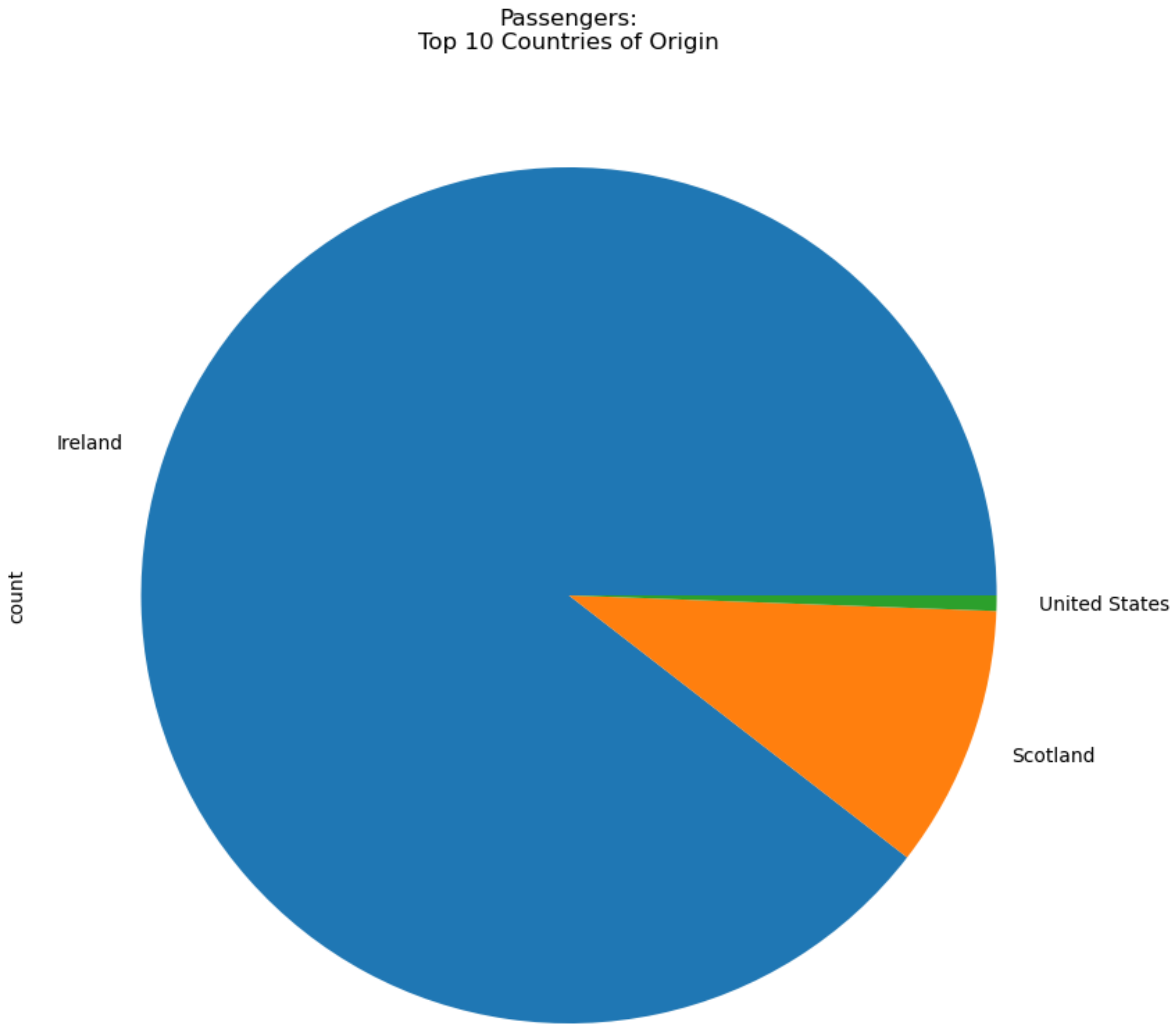
```
In [31]: #Your Code Here
         passengers_df['native_country'].value_counts()
```

```
Out[31]: native_country
         Ireland          461
         Scotland          51
         United States      3
         Name: count, dtype: int64
```

11. Make a pie chart showing **how many passengers were coming from each country**. Adapt Walsh's example.

```
In [33]: #Your Code Here
         passengers_df['native_country'].value_counts()[:10].plot(kind='pie', figsize=(10, 1
```

```
Out[33]: <Axes: title={'center': 'Passengers:\nTop 10 Countries of Origin'}, ylabel='coun
         t'>
```



12. Create a new variable called `children_filter` and assign it a True/False statement to that variable that specifies passengers who were **children**. Then use this new `children_filter` to create a new dataframe called `children_df` that **only**

contains passengers who were children. Display a sample of **five random rows** from this new dataframe. Hint: look under the `occupation` column in your dataframe. Hint: Walsh example.

In [35]:
```
#Your Code Here
children_filter = passengers_df['occupation'] == 'Child'
passengers_df[children_filter]
```

Out[35]:

	first_name	last_name	date	age	native_country	destination_city	destination_state
25	Alex	McBride	09 Apr 1884	12	Scotland	Chicago	IL
26	Cath	McBride	09 Apr 1884	11	Scotland	Chicago	IL
27	Wm	McBride	09 Apr 1884	9	Scotland	Chicago	IL
28	Agnes	McBride	09 Apr 1884	7	Scotland	Chicago	IL
29	Maggie	McBride	09 Apr 1884	4	Scotland	Chicago	IL
...	...	...	...	...	...	...	...
499	Ellen	Deran	09 Apr 1884	8	Ireland	Pittsburg	PA
503	Pat	Kyne	09 Apr 1884	9	Ireland	Portland	ME
504	John	Kyne	09 Apr 1884	3	Ireland	Portland	ME
505	Mich	Kyne	09 Apr 1884	0	Ireland	Portland	ME
511	Math	Griffins	09 Apr 1884	2	Ireland	Pittsburg	PA

132 rows × 9 columns

In [36]:
```
children_df = passengers_df[children_filter]
```

```
children_df.sample(5)
```

Out[36]:

	first_name	last_name	date	age	native_country	destination_city	destination_state
49	Sarah	Watson	09 Apr 1884	4	Scotland	Auburn	NY
447	John	Mulkern	09 Apr 1884	9	Ireland	Pittsburg	PA
312	Mary	Divine	09 Apr 1884	11	Ireland	Fonda	NY
468	Ann	Kerrigan	09 Apr 1884	4	Ireland	Winchester	MA
354	Maggie	Brennan	09 Apr 1884	9	Ireland	New York	NY

13. Create a **new CSV file** named `passenger-list-children.csv` that only contains records for passengers who were children. Hint: you'll be printing the contents of `children_df` to a CSV file using `to_csv()` method. Walsh example. To check to make sure you successfully created the file, add a line of code that reads in the newly created CSV file using `pd.read_csv()` .

In [38]:
```
#Your Code Here
children_df.to_csv('passenger-list-children.csv', encoding='utf-8', index=False)
```

Bonus Questions

What was the cut-off age for classifying a passenger as a child? Ie. What was **the oldest a passenger could be to still be considered a child**? Write code that prints out the answer to this question.

In [41]:
```
#Your Code Here
children_df['age'].max()
```

Out[41]:
```
12
```

Age Comparison: Calculate and write print() statements that show:

- The average age of passengers from **Ireland**
- The average age of passengers from **Scotland**.
- The difference in years between these average

```
In [43]: #Your Code Here
         ire_avg_age = passengers_df[passengers_df['native_country'] == 'Ireland']['age'].me
         scot_avg_age = passengers_df[passengers_df['native_country'] == 'Scotland']['age'].
```

```
In [44]: age_diff = abs(ire_avg_age - scot_avg_age)
```

```
In [45]: print(ire_avg_age)
```

```
20.98698481561822
```

```
In [46]: print(scot_avg_age)
```

```
25.098039215686274
```

```
In [47]: print(age_diff)
```

```
4.111054400068053
```

Save a Filtered Dataset: Create a new CSV file that contains data for: only adult passengers (**age 18 and over**) who were heading to **Boston**.

```
In [49]: #Your Code Here
         adult_boston_filter = (passengers_df['age'] >= 18) & (passengers_df['destination_ci
         adult_boston_df = passengers_df[adult_boston_filter]
```

```
In [50]: adult_boston_df.to_csv('passengers_adult_boston.csv', encoding='utf-8', index=False
```

```
In [ ]:
```